DNS SECURITY

Leonardo Querzoni
querzoni@diag.uniroma1.it

SAPIENZA
UNIVERSITÀ DI ROMA

CIS SAPIENZA
CYBER INTELLIGENCE AND INFORMATION SECURITY

THE DNS NAMESPACE

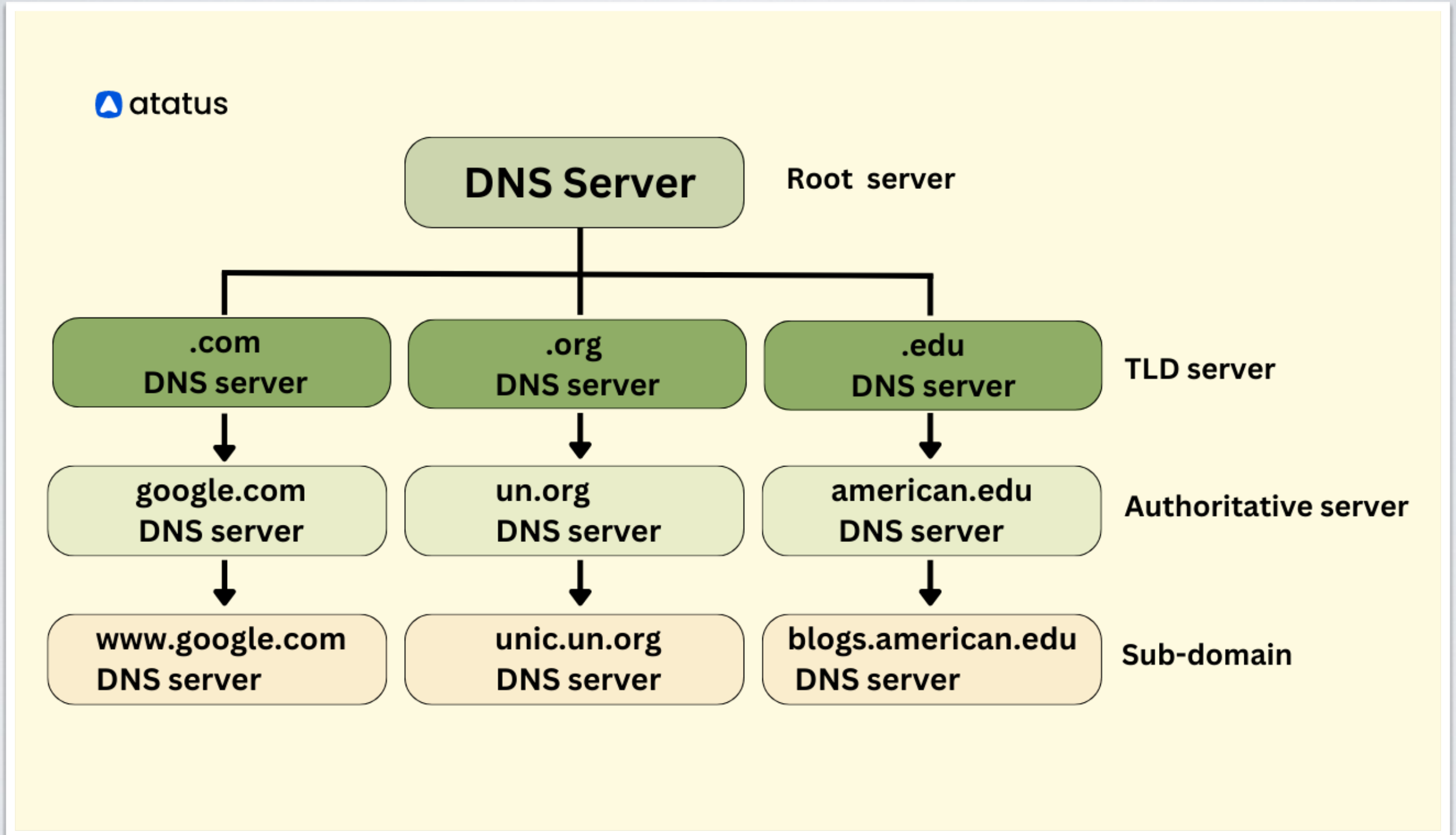
The **Domain Name System** implements a distributed database.

DNS is a strict **hierarchy** starting from the Root (.)

- **Root** (.): 13 Logical Clusters (A-M), Managed by IANA.
- **TLD** (.com, .edu): Managed by Registries (e.g., Verisign).
- **SLD** (example.com): Managed by Registrars/Owners.

It is rooted on the concept of **delegation**: the parent zone does not contain the data; it contains pointers (NS records) to the child's authoritative servers.

THE DNS NAMESPACE

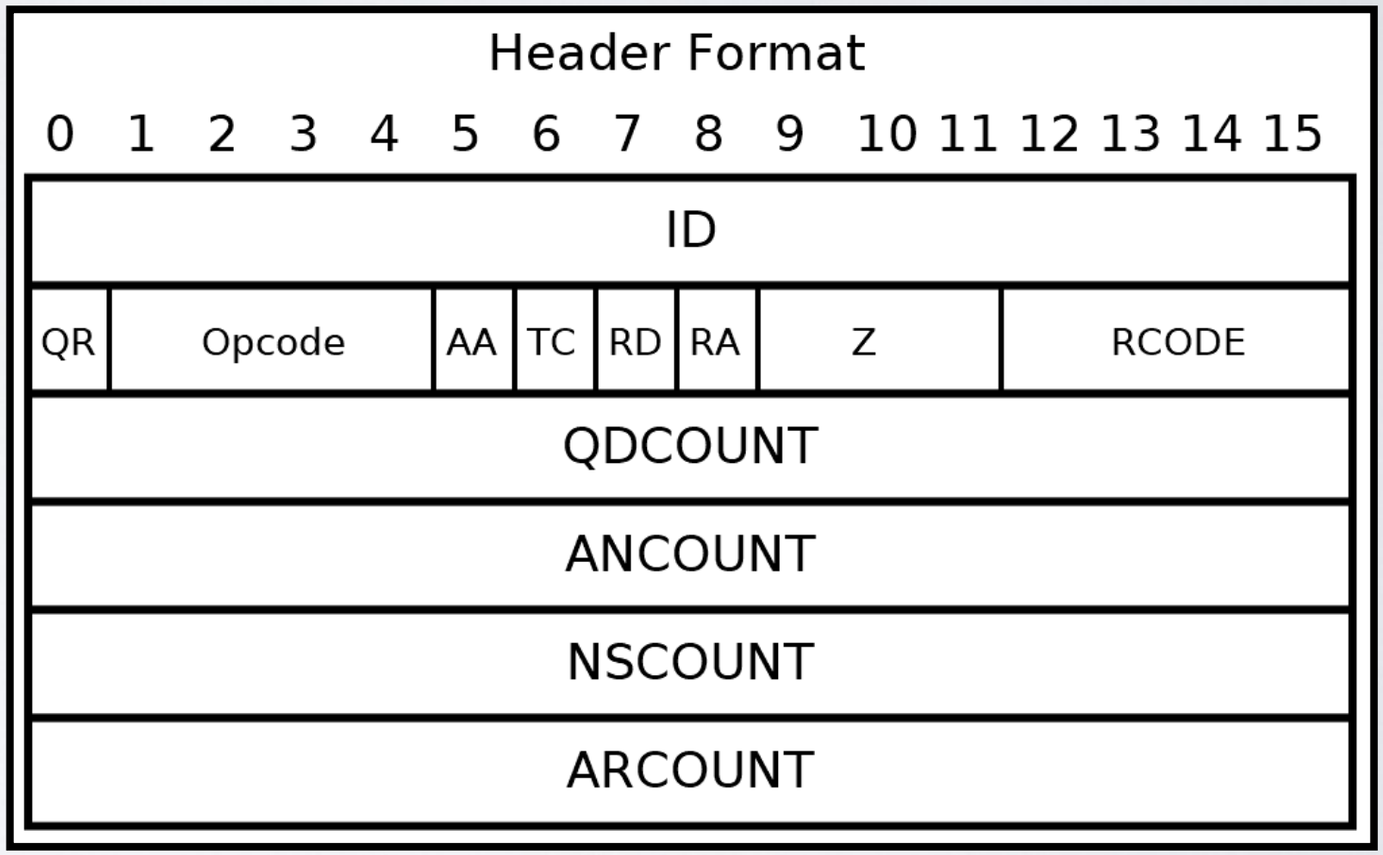


THE DNS PACKET FORMAT (RFC 1035)

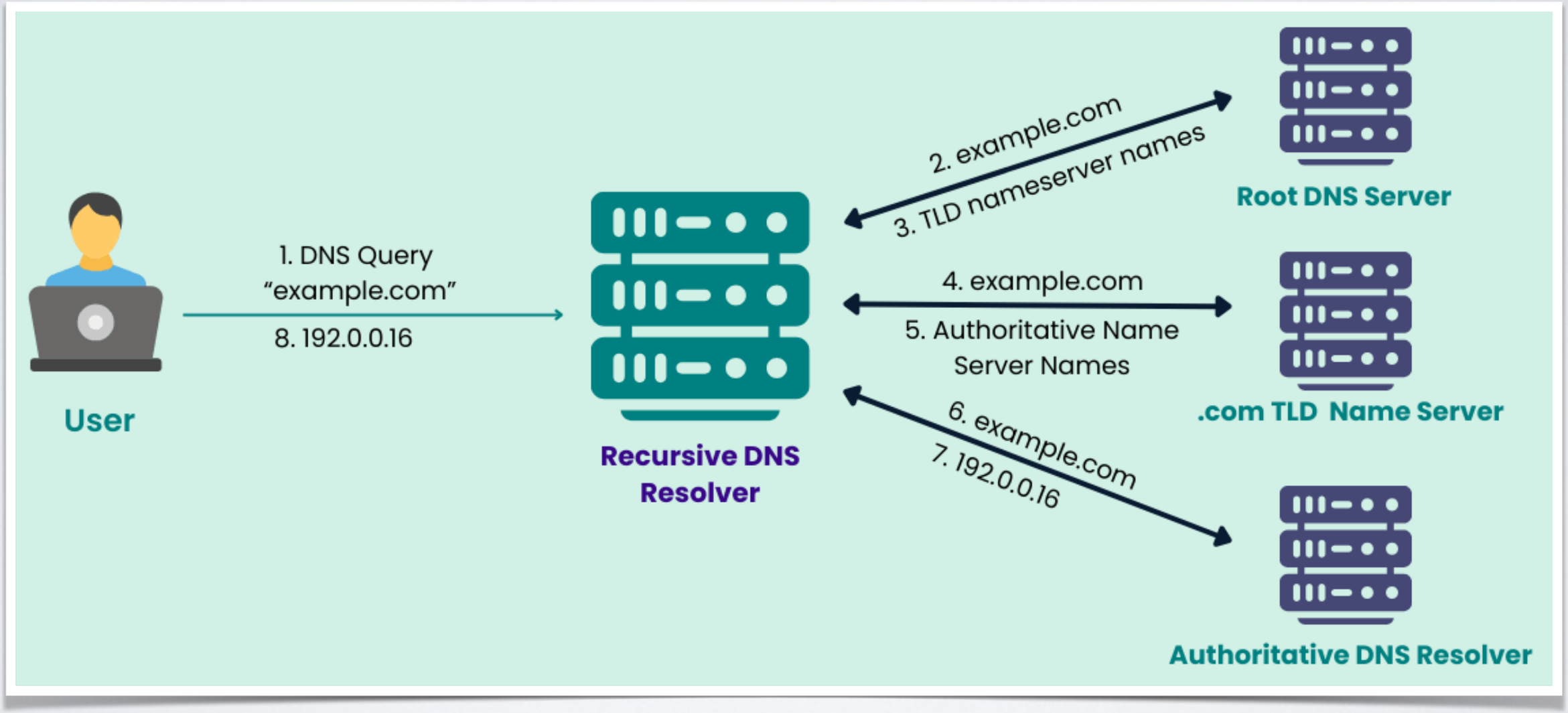
To understand attacks, we must understand the bits on the wire.

- Transaction ID (16-bit): The only mechanism in original DNS to match queries to responses.
- Flags: QR (Query/Response), AA (Authoritative), RD (Recursion Desired).
- Counts: Number of questions, answers, authority, and additional records.

Packets usually travel on UDP



NAME RESOLUTION PROCESS



VULNERABILITIES IN DNS

CONFIDENTIALITY

Status: Non-existent in standard DNS.

Threat: Passive surveillance. ISPs, governments, and local attackers can see every domain you request.

INTEGRITY

Status: Weak (ID only).

Threat: Cache Poisoning. Attackers can modify answers in transit or inject fake data into caches.
Spoofing. You cannot prove a response actually came from the owner of the zone.

AVAILABILITY

Status: Generally high (Anycast), but vulnerable.

Threat: DDoS Amplification, NXDOMAIN floods, Phantom Domain attacks.

SPOOFING AND POISONING

DNS does not include any authentication mechanisms. Weak integrity is only provided through transaction IDs.

The DNS Transaction ID is only 16 bits (0 to 65,535).

A resolver accepts a response if:

- Source IP matches query Destination IP.
- Destination Port matches Source Port.
- Transaction ID matches.

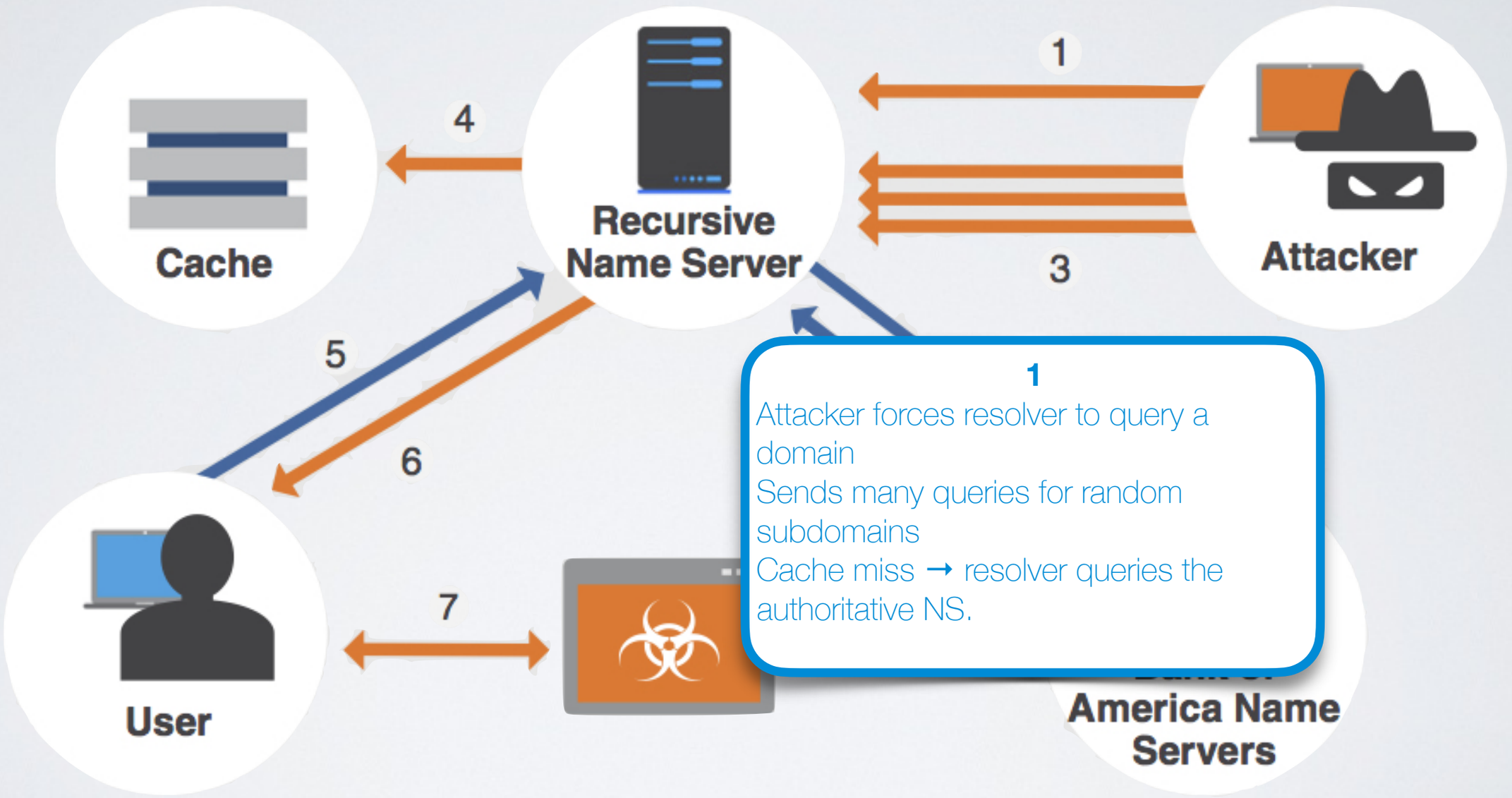
If an attacker can flood the resolver with guesses before the real server answers (race condition), they can poison the resolver's cache.

SPOOFING AND POISONING

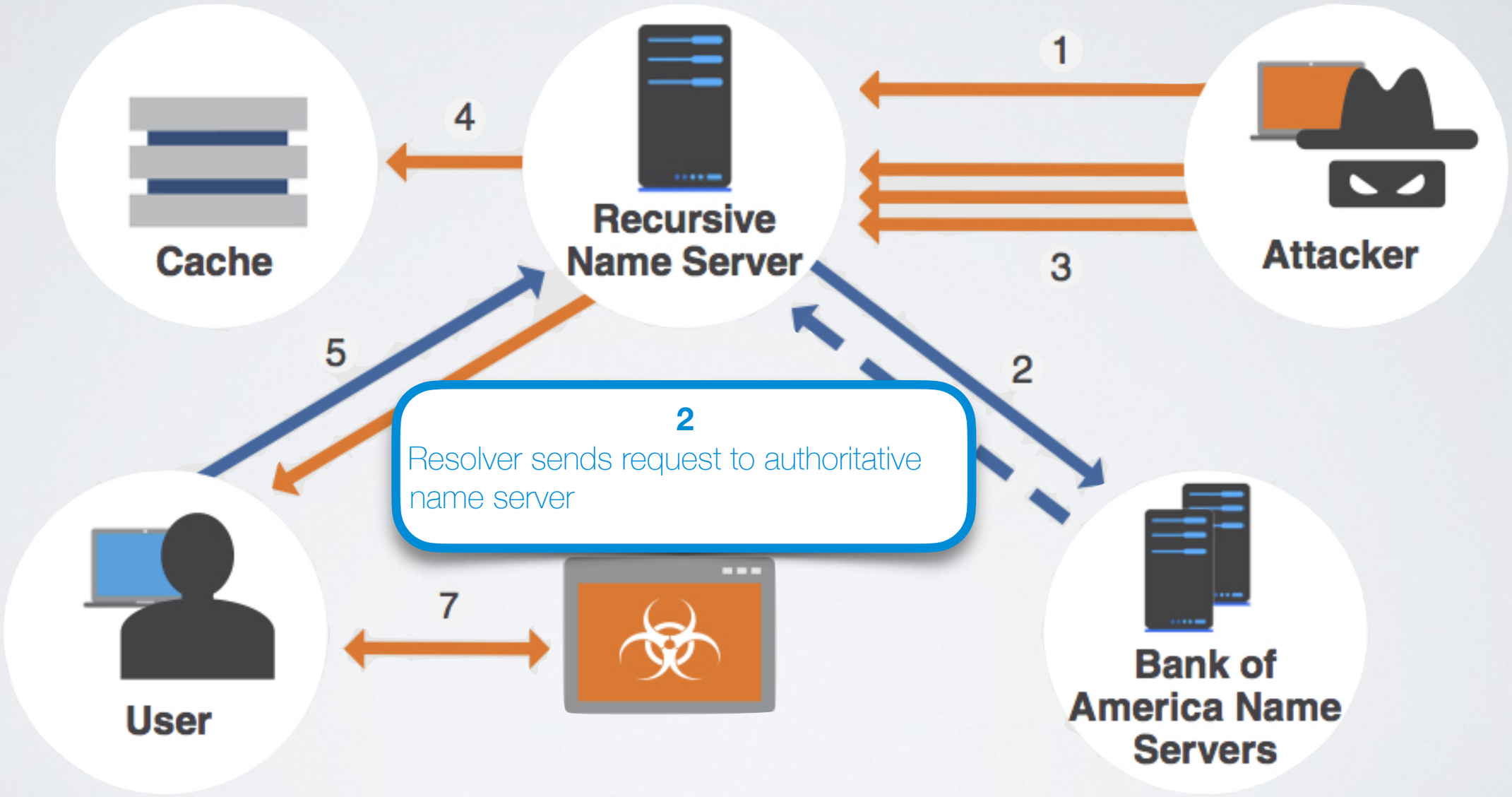
Core Elements the Attacker Must Guess/Control:

- Transaction ID (TXID)
  - 16-bit field → 65,536 possibilities.
  - Older resolvers used predictable or sequential TXIDs.
- Source Port of the Resolver's Query
  - If not randomized, easy to guess.
  - With port randomization, entropy increases to ~$2^{16} \times 2^{16} \approx 4$ billion combinations.
- Query Name
  - The attacker triggers the resolver to send a query by requesting a domain under attacker control (e.g., x123.attacker.com).
- Race Timing
  - The forged response must arrive before the real authoritative response.

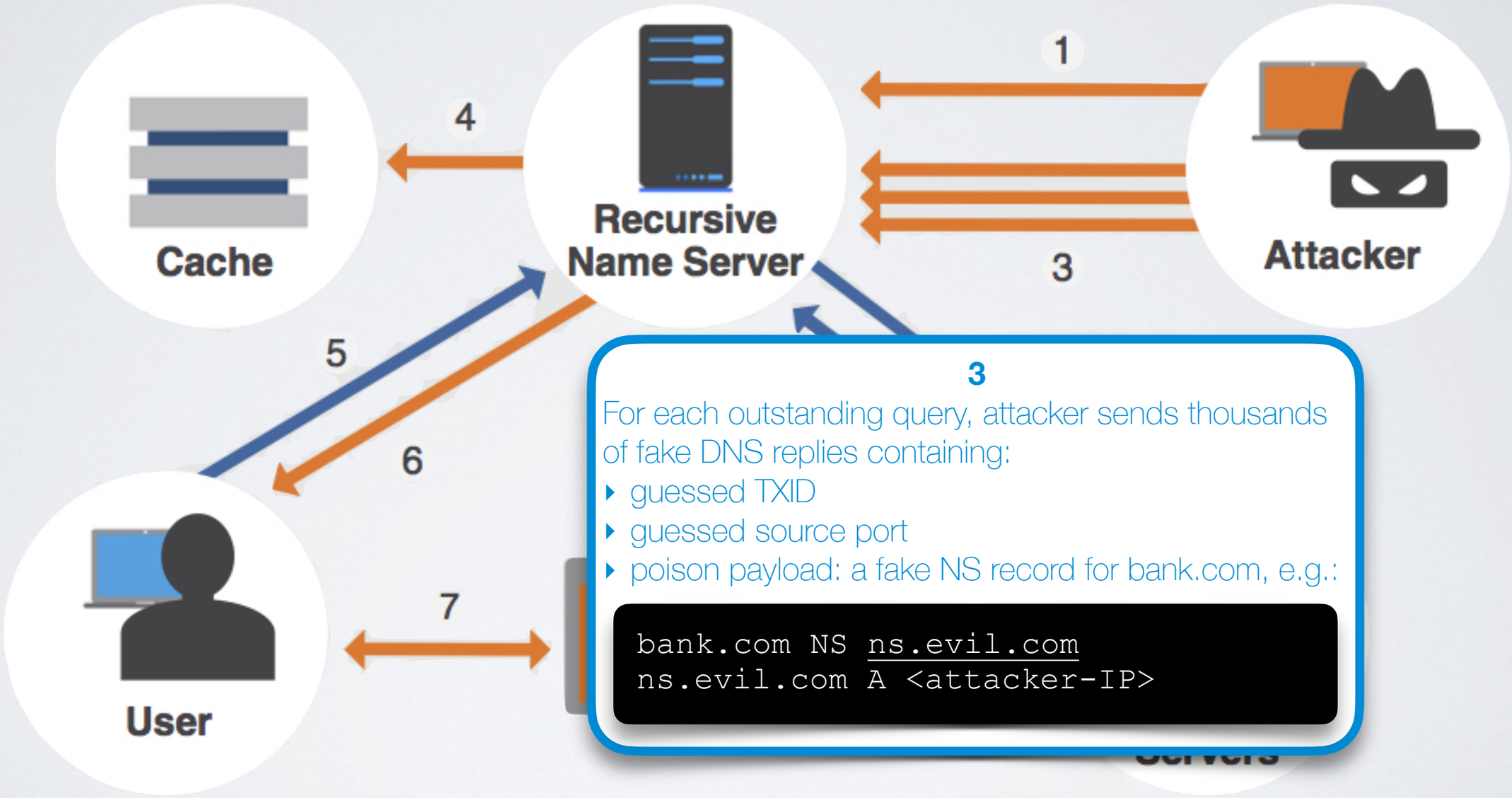
SPOOFING AND POISONING



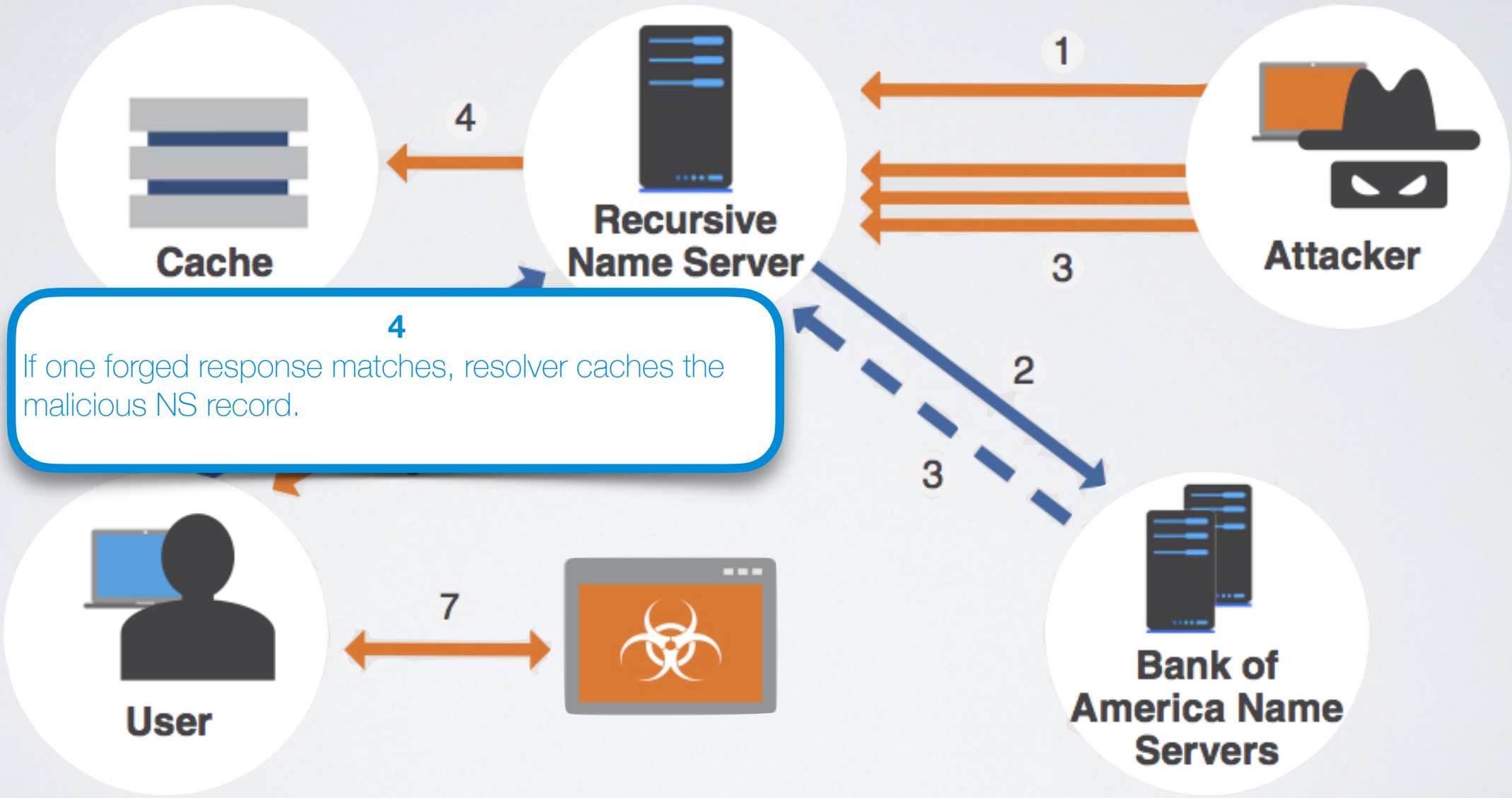
SPOOFING AND POISONING



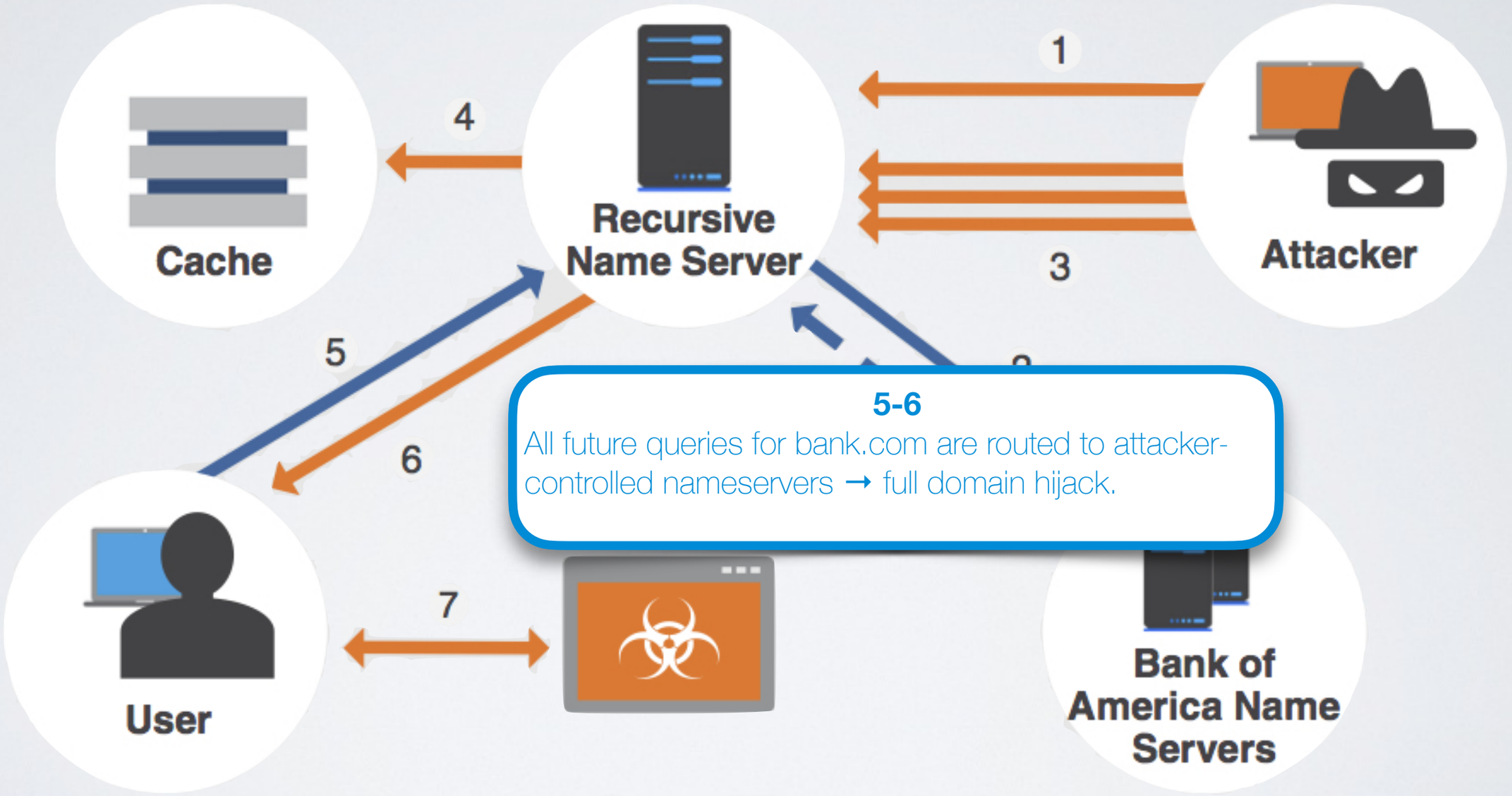
SPOOFING AND POISONING



SPOOFING AND POISONING



SPOOFING AND POISONING



SPOOFING AND POISONING

Conditions That Make Poisoning Easier

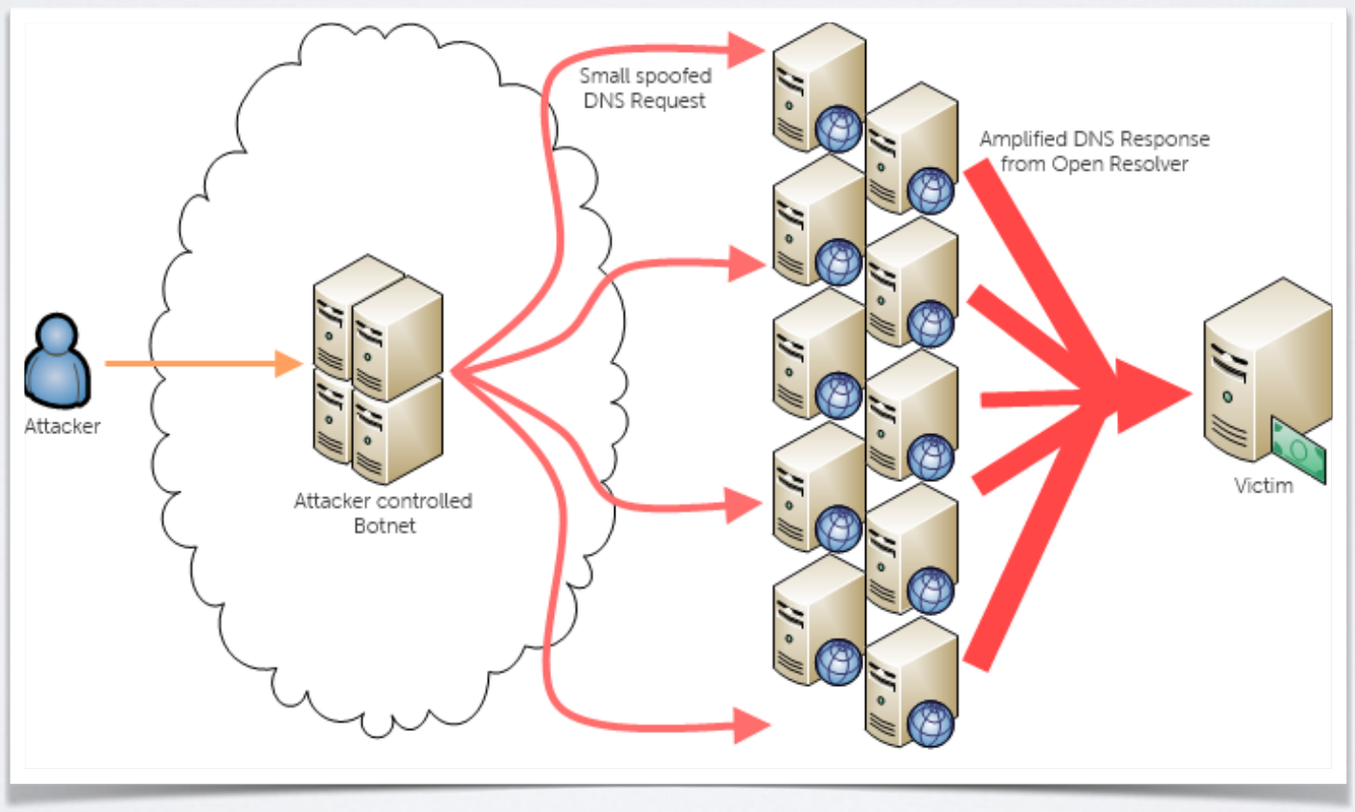
- Weak or absent TXID randomization
- No source port randomization
- Open resolvers reachable on the Internet
- Long TTLs (if attacker succeeds once, the effect persists)
- Resolver accepting out-of-bailiwick records in the Additional/Authority sections

DDOS AMPLIFICATION WITH DNS

DDoS attack may use the DNS service to amplify the response.

Asymmetric Bandwidth

- Request: 60 bytes (UDP).
- Response: 3000+ bytes (EDNS0, ANY query).
- Amplification Factor: ~50x - 70x.



DNS TUNNELING

DNS tunneling is a technique that encodes data (commands, exfiltrated info) within DNS queries and responses.

Malware uses it to bypass firewalls, proxy restrictions, and network monitoring, because DNS traffic is usually trusted and allowed.

Various possible approaches:

- Data is encoded in the queried domain name
  - E.g. h39&/hd09JW78$.evil.com
- Data is encoded in the response fields
  - Use returned IP addresses
  - Return values of TXT queries
- Data is encoded in response latencies (timing channel)

DNS TUNNELING

- Data Encoding
  - Malware encodes information into a format suitable for DNS labels.
- Query Types Used
  - TXT records: Store arbitrary text in responses.
  - A / AAAA records: Encodes data as IP addresses.
  - CNAME / MX / other resource records: Occasionally used for obfuscation.
- Communication Flow
  - Malware generates domain query with payload → sends to DNS server.
  - Query reaches attacker's authoritative server.
  - Server decodes payload and responds with a DNS answer
  - Malware executes command or continues exfiltration cycle

DOMAIN GENERATION ALGORITHM

A Domain Generation Algorithm (**DGA**) is a deterministic algorithm used by malware to periodically compute large sets of pseudo-random domain names.

The goal is to dynamically identify Command-and-Control (C2) endpoints while evading static signatures and takedown efforts.

Why malware use DGAs?

- Resilience
- Anti-blacklisting
- Asymmetric advantage
- Obfuscation

DOMAIN GENERATION ALGORITHM

The code generates a domain list → iterates through them → attempts DNS resolution → connects to the first domain that responds.

- **Inputs / Seeds**: Current timestamp, hard-coded keys, PRNG seeds, system parameters, pseudo-random functions.
- **Output**: A large set of candidate domains (e.g., hundreds or thousands per day).

DOMAIN GENERATION ALGORITHM

Time-based DGAs

- Domains generated using date or system time.
- Easy to replicate by analysts but resilient at scale.
- Examples: Conficker, Bebloh.

PRNG-based DGAs

- Use pseudo-random number generators (LCG, Mersenne Twister, xorshift).
- Seeds may be static, shared, or derived from external values.
- Provide large search space and low predictability without reverse engineering.

DOMAIN GENERATION ALGORITHM

Seed-and-Key DGAs (Cryptographic DGAs)

- Integrate HMAC, SHA-2, AES, or CRC functions to generate domains.
- High entropy, difficult to predict without the key.
- Used by more advanced families (e.g., Bamital, Matsnu).

Wordlist-Based DGAs

- Domains composed by concatenating dictionary words
- Hybrid DGAs combine wordlists with PRNG or seeds.
- Examples: Matsnu, Suppobox.

DNS FAST-FLUXING

Evasion and resilience technique used by botnets and cybercriminal infrastructures to hide and protect malicious services (e.g. C2 servers) behind rapidly changing IP addresses.

Leverages features of the DNS system to make a malicious domain resolve to many different IPs over very short time intervals.

```
malicious-domain.com  -> 45.12.33.10
                         89.23.14.55
                         102.44.12.19
                         77.192.1.50

TTL = 60 seconds
```

DNS FAST-FLUXING

Rapid IP Rotation

- A single malicious domain is associated with hundreds or thousands of compromised hosts (bots).
- The DNS A records for the domain change every few minutes or seconds.
- Each response returns a different subset of bot IPs.

Extremely Short TTL Values

- DNS TTL values are intentionally set to very low values (e.g., 60 sec. or less).
- Forces clients and resolvers to re-query the DNS server frequently

Use of Compromised Machines as Proxies

- Returned IPs usually belong to compromised hosts acting as reverse proxies.
- They forward traffic to a small number of hidden backend servers

DNS FAST-FLUXING

Variants:

Single-Flux

- Only the A records (domain → IP mappings) change rapidly.
- IP rotation hides the front-end nodes.

Double-Flux

- Both A records and NS (name server) records rotate
- Much harder to take down because the DNS infrastructure itself is fluxing

DNSSEC: DESIGN GOALS

DNSSEC (DNS Security Extensions) is a set of protocol extensions that add cryptographic **integrity** and **authenticity** to DNS.

- RFCs 4033/4/5, 5155, 6781, 8145

It ensures that DNS responses cannot be tampered with or forged, solving the class of attacks that includes cache poisoning and spoofing.

DNSSEC: DESIGN GOALS

Provides:

- **Origin Authentication** - Proof data came from the zone owner.
- **Data Integrity** - Proof data wasn't modified in transit.
- **Authenticated Denial of Existence** - Proof that a domain does not exist.

DOES NOT provide:

- **Confidentiality** - Data is still cleartext.
- **DoS Protection** - Actually increases DDoS risk!

DNSSEC: RECORDS

DNSSEC adds additional DNS record types:

- RRSIG (signature) - A cryptographic signature over a DNS resource record set (RRset).
- DNSKEY - The public key used to verify signatures.
- DS (Delegation Signer) - Links a child zone's key to the parent zone's key (forming a chain of trust).
- NSEC / NSEC3 - Authenticated denial of existence (proves "that name does not exist").

RESOURCE RECORD SIGNATURE

RRSIG (signature) - A cryptographic signature over a DNS resource record set (RRset).

- In DNSSEC, we do not sign individual records. We sign the RRset.

```
www.example.com. 3600     IN  A   192.0.2.1
www.example.com. 3600     IN  A   192.0.2.2
- - - - - - - - - - - - - - - - - - - - - - - -
All A records for “www” are bundled and signed
together
```

If a resolver receives an RRset with a missing record (e.g., in a subset), the signature verification will fail.

RESOURCE RECORD SIGNATURE

RRSIG - What does it include?

- Covered RRset Type - Specifies which type of record is being signed (e.g., A, AAAA, MX, TXT). Ensures the signature is valid only for that RRset.
- Algorithm - The cryptographic algorithm used to create the signature (e.g., RSA/SHA-256, ECDSA).
- Labels - The number of labels in the original domain name (used to prevent some types of replay attacks). e.g. www.example.com → 3 labels.
- Original TTL - The TTL of the RRset at the time of signing.
- Signature Expiration - Date/time after which the signature is considered invalid.
- Signature Inception - Date/time when the signature becomes valid.
- Key Tag - Identifier of the DNSKEY that can be used to verify this signature.
- Signer Name - The domain name of the zone that generated the signature.
- Signature Data - The actual cryptographic signature.

DNSKEY: THE PUBLIC KEY

DNSKEY - Contains the Public Key verifying the zone's RRSIGs.

```
example.com.  3600  IN  DNSKEY  256 3 8
(AwEAAagAIKlVZrpC6Ia7gEzahOR+9W29euxhJhVVLOyQbF5+...)
```

- Flags - Key properties:
  - 256 → Zone Signing Key (ZSK)
  - 257 → Key Signing Key (KSK)
- Protocol - Always 3 for DNSSEC (RFC 4034).
- Algorithm - Used crypto algorithm (8 = RSA/SHA-256).
- Public Key - The public key encoded with Base64. Used by resolvers to check RRSIG signatures.

ZSK VS. KSK

ZSK (Zone Signing Key)

- Signs the resource records (RRsets) in the zone (A, TXT, etc. but no DNSKEY).
- Protects the integrity and authenticity of the zone's data.
- Frequently rotated (days to weeks) to limit exposure if compromised
- Smaller than KSK to speed up signing and verification

KSK (Key Signing Key)

- Signs the DNSKEY RRset itself
- Establishes a trust anchor for the zone, linking it to the parent zone via DS rec.
- KSK is essentially the root of trust for that zone.

DS: DELEGATION SIGNER

DS - used to link a child zone's key to the parent zone, forming part of the chain of trust.

It allows a resolver to verify that the child zone's DNSKEY is authentic, using the parent zone as the trust anchor.

- When a child zone is delegated from a parent zone (e.g., example.com under .com), the parent stores a hash of the child's KSK in a DS record.
- This allows a validating resolver to:
  - Start at the parent zone (trusted or already validated).
  - Verify the child zone's KSK by comparing its hash to the DS record.
  - Use the child KSK to validate the signatures (RRSIG) of all other records in the child zone.

You can change the ZSK locally without talking to the Parent Zone (Registrar). You only need to talk to the Parent when rolling the KSK.

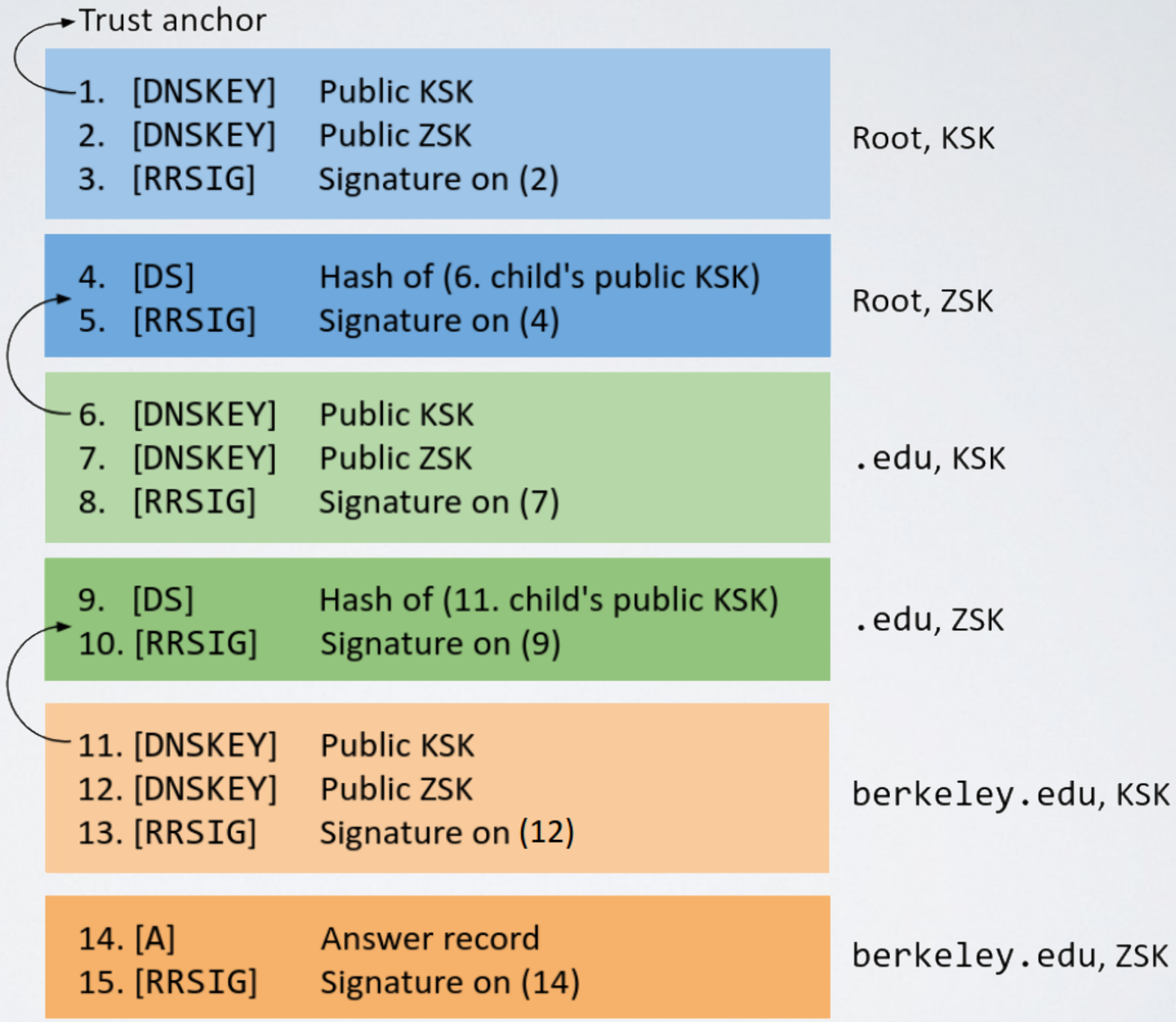
DS: DELEGATION SIGNER

DS - example: suppose we have Root (.) → TLD (.com) → example.com

1. Resolver trusts the root's public key (preconfigured trust anchor).
2. Resolver queries .com for example.com.
3. The .com zone returns:
   - NS records for example.com
   - DS record pointing to example.com's KSK
4. Resolver fetches example.com's DNSKEY RRset.
5. Resolver computes the hash of the KSK using the digest type from the DS
6. Resolver compares it with the DS record in .com.
   - Match → child KSK is trusted
   - Mismatch → validation fails
7. Resolver can now use the child KSK to verifv RRSIGs for the rest of the zone.

CHAIN OF TRUST

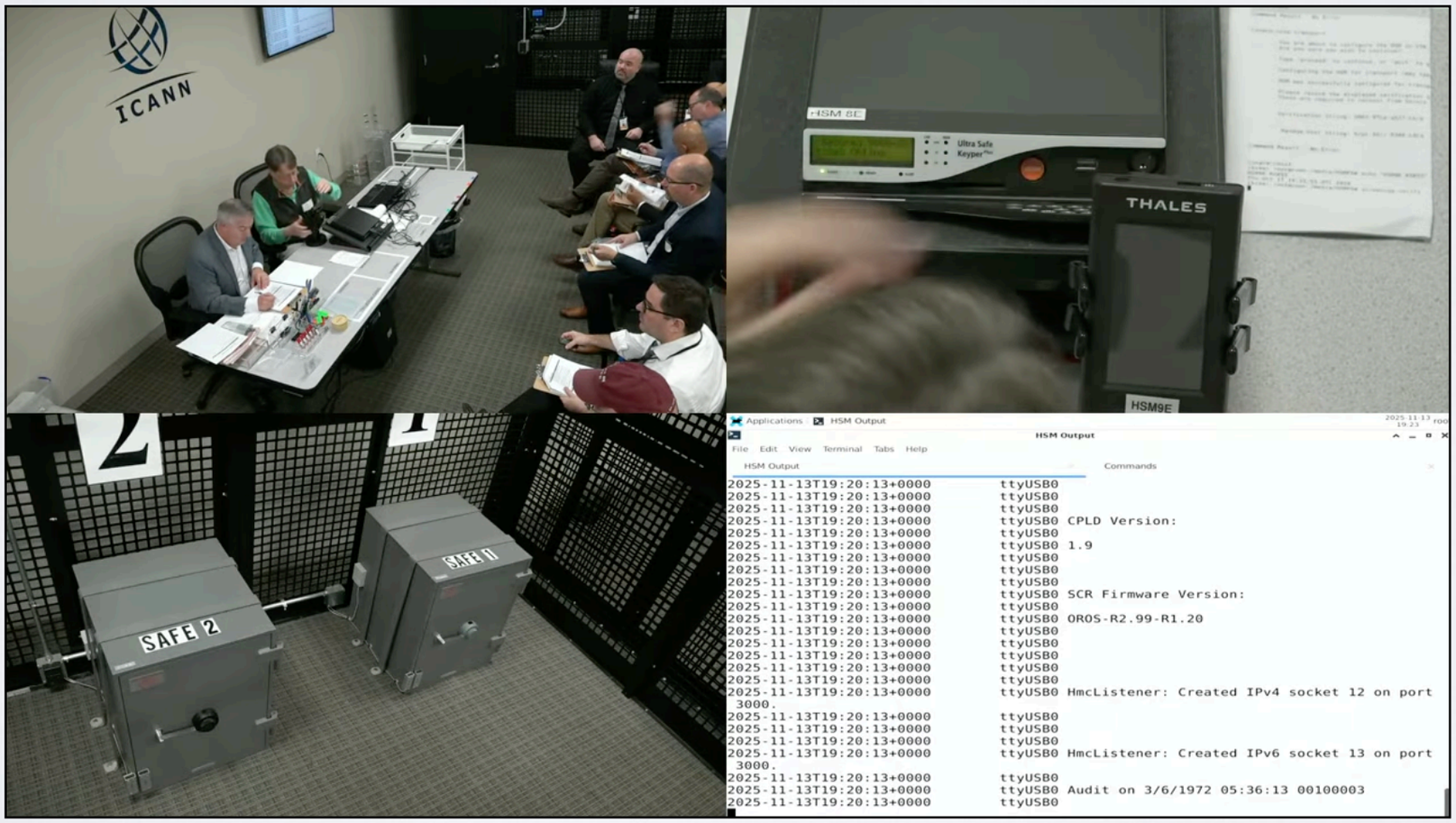
DS - example:



CHAIN OF TRUST

What about the root zone keys?

- They are managed by ICANN
- They are periodically rotated through a public “Key signing ceremony”
- Check it online: https://www.iana.org/dnssec/ceremonies



NSEC: AUTHENTICATED DENIAL

NSEC - How can we certify the “absence” of a domain name? How do you sign “Nothing"?

An NSEC record links a domain name to the next valid domain name in the zone, in canonical order.

```
bar.example.com. 3600 IN NSEC foo.example.com. A MX RRSIG
```

Interpreted as:

- Zone contains no domain names between bar. and foo.
- bar. only has A, MX, and RRSIG records; any other type does not exist.
- The record is signed by an RRSIG, so it is cryptographically verified.

Search is performed till the zone end and wraps-around the zone.

BONUS FEAT: ZONE ENUMERATION FOR FREE!!!

NSEC3

NSEC3 - enhancement to NSEC that prevents zone enumeration while still providing authenticated denial of existence.

Instead of exposing plain domain names:

- The owner names are hashed using a cryptographic hash function (usually SHA-1).
- NSEC3 records link hashed owner names to the next hashed name in “hashed order”.
- The resolver hashes the queried name and checks the NSEC3 chain.

The record defines details for hashing such as:

- Hash algorithm to be used, salt, iterations

OTHER OPTIONS

DNS over TLS (DoT) - RFC 7858

- Port: 853 (TCP).
- Encryption: Wraps DNS packets in a TLS tunnel.
- Pros: provides confidentiality/integrity/auth.
- Cons: Needs to travel on TCP. Larger resource usage. Larger delays. May be blocked by some firewalls

OTHER OPTIONS

DNS over HTTPS (DoH) - RFC 7858

- Port: 443 (TCP).
- Encryption: Mixes DNS queries with normal HTTP traffic.
- Pros: provides confidentiality/integrity/auth. Very hard to censor/block without blocking the web.
- Cons: Limited control from network admins. A channel available for attackers (C2s)