Solutions to Selected Problems

Guide to Internet Cryptography

Companion Material

February 17, 2026

Preface This document provides solutions to selected problems from the book *Guide to Internet Cryptography: Security Protocols and Real-World Attack Implications*. The material is intended for educational use in courses and self-study.

Book website: https://link.springer.com/book/10.1007/978-3-031-19439-9

1 Chapter 15: DNS

Problem 15.1 Domain Name System

(a) What is the difference between a zone and a domain?

(b) Which types of resource records are used for hostname resolution?

(c) When is caching more effective? With recursive queries or with iterative queries?

Solution

(a) A domain is a logical concept: a complete subtree in the DNS tree. A zone is an administrative concept: It may be a complet subtree, or a subtree minus subtrees of this subtree.

(b) Mentioned in the book are `A`-RRs (resolving domain names to IPv4 addresses), `AAAA`-RRs (resolving domain names to IPv6 addresses), and `CNAME`-RRs (resolving domain names tocanonical domain names).

(c) With recursive queries, because the answer passes more caches.

Problem 15.2 Attacks on DNS

(a) What is the difference between DNS Spoofing and DNS Cache Poisoning?

(b) What is the birthday paradox?

(c) A standard cache poisoning attack can only be performed until the correct DNS response arrives at the target name server – after such an event, the attacker has to wait for the cached entry to expire. Why is this not a problem for the Kaminski attack?

Solution

(a) For DNS Spoofing, the attacker needs Man-in-the-Middle privileges. Classical DNS (Do53, DNS over UDP Port 53) is always vulnerable to this. Only DNSSEC or DNS variants like DNS-over-HTTPS (DoH) may protect against these attacks.
DNS Cacche Poisoning attacks are much more dangerous, because here the attacker does not need Man-in-the-Middle privileges.

(b) The Birthday Paradox is a combinatorial result used in statistics which contradicts human intuition about probabilities.

The Birthday Paradox The "paradox" is this: in a class of just **30 students**, the probability that **at least two share a birthday** is about **70%** — which strikes most people as surprisingly high.

Why it feels wrong intuitively Most people think: "there are 365 days in a year and only 30 students, so the chance must be quite low." This is because we instinctively think about the probability that *someone shares a birthday with **us*** — which is indeed low ($\approx 8\%$). But the question is about **any two students** sharing a birthday with **each other**, which is a much larger net.

Calculating it It is easier to calculate the complement — the probability that **no two students share a birthday** — and subtract from 1.
Imagine students entering the room one by one:

- **Student 1** can have any birthday: $\frac{365}{365}$
- **Student 2** must avoid student 1's birthday: $\frac{364}{365}$
- **Student 3** must avoid both previous birthdays: $\frac{363}{365}$
- $\vdots$
- **Student 30** must avoid all previous birthdays: $\frac{336}{365}$

So the probability that **all 30 birthdays are different** is:

$$P(\text{no match}) = \frac{365}{365} \cdot \frac{364}{365} \cdot \frac{363}{365} \cdots \frac{336}{365} = \prod_{k=0}^{29} \frac{365-k}{365} \approx 0.294 \tag{1}$$

Therefore:

$$P(\text{at least one match}) = 1 - \prod_{k=0}^{29} \frac{365-k}{365} \approx 1 - 0.294 = \boxed{0.706 \approx 70\%} \tag{2}$$

General Formula For n students, the probability of at least one shared birthday is:

$$P(n) = 1 - \frac{365!}{(365-n)! \cdot 365^n} \tag{3}$$

Why it grows so fast With 30 students, the number of **pairs** is:

$$\binom{30}{2} = \frac{30 \cdot 29}{2} = 435 \text{ pairs} \tag{4}$$

Each pair has a probability of sharing a birthday of:

$$\frac{1}{365} \approx 0.27\% \tag{5}$$

With **435 independent chances** for a match, a collision becomes far more likely than intuition suggests. Notably, the probability crosses 50% at just $n = \mathbf{23}$ students:

$$P(23) = 1 - \prod_{k=0}^{22} \frac{365 - k}{365} \approx 0.5073 > 50\% \tag{6}$$

(c) Because in the Kaminski attack, the attacker doesn't query for the target domain (which would be answered from the cache during the lifetime of teh cache entry), but for a random, nonexistent subdomain. Such subdomains are never cached, so the queries cannot be answered from any long-lived cache.

Problem 15.3 DNSSEC

(a) How much bigger is the DNSSEC zone file compared to the original zone file?

(b) How many signature verifications are needed to validate `www.example.com` in Figure 15.8?

Solution

(a)

Overview When a zone is signed with DNSSEC, the zone file grows **significantly** — typically by a factor of **5x to 10x** the original size, though it can be even more depending on several factors.

What causes the size increase? DNSSEC adds several new resource record types to every zone:

- **RRSIG** — a cryptographic signature record added for *every existing RRset* in the zone. This is the biggest contributor to size growth.
- **DNSKEY** — stores the public keys (ZSK and KSK) used to verify signatures.
- **NSEC / NSEC3** — provides authenticated denial of existence, added for every name in the zone.
- **DS** — delegation signer records (stored in the *parent* zone, not the child).

Concrete Example A simple unsigned zone might look like:

```
example.com.     A       93.184.216.34
www              A       93.184.216.34
mail             MX      10 mail.example.com.
```

After signing, each record gets an accompanying `RRSIG`, the `DNSKEY` records are added, and `NSEC`/`NSEC3` records fill in the gaps between names. What was 3 records can easily become 15–20.

Key Factors that Influence the Size

- **Key algorithm and length** — RSA/2048 produces larger signatures than ECDSA P-256, which can reduce overhead by roughly 3× compared to RSA.
- **Number of RRsets** — since every RRset gets its own RRSIG, more record types per name means more signatures.
- **NSEC vs. NSEC3** — both add roughly the same amount of data per name.
- **Signature lifetime** — does not affect size, but frequent re-signing is operationally relevant.
- **Number of DNSKEY records** — using separate ZSK and KSK adds two key records plus their RRSIGs.

Recommendation The shift from RSA to **ECDSA or EdDSA** (Ed25519) is the single most effective way to reduce DNSSEC overhead, as their signatures are dramatically shorter while maintaining equivalent security. Concretely:

- An **RSA/2048** signature is **256 bytes**.
- An **ECDSA P-256** signature is **64 bytes**.
- An **Ed25519** signature is **64 bytes**.

This means switching from RSA/2048 to ECDSA or Ed25519 reduces the size of each signature by a factor of:

$$\frac{256 \text{ bytes}}{64 \text{ bytes}} = 4\times \tag{7}$$

Since `RRSIG` records dominate the size increase, this translates directly into a substantially smaller signed zone file overall.

(b) 6 signature verifications and 3 value comparisons.

Zone type	**Typical size increase**
Small zone, RSA/2048	8× – 10×
Small zone, ECDSA P-256	3× – 5×
Large zone with many records	4× – 6×

Table 1: Typical DNSSEC zone file size increases by algorithm and zone type.

Problem 15.4 DNSSEC

Consider the following minimal zone file:

Domain	Class	Type	RData
example.com.	IN	SOA	SOA data
example.com.	IN	NS	ns.example.com
ns.example.com.	IN	A	111.111.111.112
host.example.com.	IN	A	111.111.111.111

Sketch the structure of the zone file after DNSSEC signing when NSEC is used.

Solution

Domain	**Class**	**Type**	**RData**
example.com.	IN	SOA	SOA data
example.com.	IN	RRSIG	Covers SOA, signed with ZSK
example.com.	IN	NS	ns.example.com.
example.com.	IN	RRSIG	Covers NS, signed with ZSK
example.com.	IN	DNSKEY	ZSK public key
example.com.	IN	DNSKEY	KSK public key
example.com.	IN	RRSIG	Covers DNSKEY, signed with KSK
example.com.	IN	NSEC	host.example.com. (SOA NS DNSKEY NSEC RRSIG)
example.com.	IN	RRSIG	Covers NSEC, signed with ZSK
host.example.com.	IN	A	111.111.111.111
host.example.com.	IN	RRSIG	Covers A, signed with ZSK
host.example.com.	IN	NSEC	ns.example.com. (A NSEC RRSIG)
host.example.com.	IN	RRSIG	Covers NSEC, signed with ZSK
ns.example.com.	IN	A	111.111.111.112
ns.example.com.	IN	RRSIG	Covers A, signed with ZSK
ns.example.com.	IN	NSEC	example.com. (A NSEC RRSIG)
ns.example.com.	IN	RRSIG	Covers NSEC, signed with ZSK