DNSSEC: Domain Name System Security Extensions

Devendra Pratap Singh, Sreya Gunda

April 20, 2025

Word count: 3363

1 Introduction

1.1 Background of the Domain Name System (DNS)

The Domain Name System (DNS) is the internet's phonebook, translating human-readable domain names such as `www.example.com` to machine-readable IP addresses. Created in the 1980s, DNS has matured into an essential part of the internet ecosystem, allowing people to visit websites and online services easily without needing to remember numeric IP addresses. The DNS takes place in hierarchical structure, comprised of root servers, top-level domains (TLDs), and authoritative name servers that aid in resolving varied levels of requests.

1.2 Motivation for DNS Security

Even though it plays a critical function, the original DNS protocol wasn't developed with robust security controls. This has exposed it to numerous attacks such as DNS spoofing, cache poisoning, and man-in-the-middle attacks. These attacks can redirect users to fake websites, compromise data integrity, and facilitate phishing schemes. The absence of data origin authentication and message integrity in DNS responses has given rise to a critical need for a more secure infrastructure. As threats spread through cyberspace, ensuring that DNS data is authentic and its integrity ensured has become of utmost concern for governments, companies, and individuals.

1.3 Overview of DNSSEC

DNS Security Extensions (DNSSEC) is a set of protocols that provide security for the DNS by allowing authentication of DNS information. It accomplishes this through digitally signing DNS information

using public-key cryptography. DNSSEC ensures that users are communicating with the correct domain, not an imposter or spoofing version. It does not encrypt DNS queries, but it ensures responses are valid and authentic. DNSSEC adds a few new types of DNS records and a chain of trust system that enables resolvers to authenticate the validity of received information from root to the level of the particular domain.

2 Understanding DNS

2.1 Basics of DNS Resolution

The Domain Name System (DNS) is a distributed and decentralized system to map domain names to IP addresses. When a user types a URL like `www.example.com` in a browser, a DNS request is sent to resolve this name to the associated IP address. This is referred to as DNS resolution.

The resolution starts at the recursive resolver, usually run by the user's Internet Service Provider (ISP). If the resolver has the domain name in cache, it simply returns the corresponding IP address. Otherwise, the resolver queries one of the root name servers. The root server returns a referral to the correct Top-Level Domain (TLD) server, e.g., those serving
texttt.com or
texttt.org domains.

The resolver next asks the TLD server, which gives a referral to the authoritative name server for the requested domain. The authoritative server contains the real DNS records (e.g., A, AAAA, MX, CNAME, etc.) and gives the IP address of the domain. Lastly, the resolver returns the IP address to the user's system, which proceeds to connect to the destination server.

This multi-step query procedure occurs in milliseconds and is optimized by caching at multiple levels. Recursive resolvers cache answers to make it faster and minimize load on upstream servers, while authoritative servers can outsource subdomain management to other servers, creating a scalable resolution system.

2.2 DNS Hierarchy and Zone Files

The DNS hierarchy is naturally hierarchical, structured as an inverted tree with the root at the apex. Immediately below the root are the Top-Level Domains (TLDs) like `.com`, `.net`, `.org`, and country-code TLDs like `.uk` and `.in`. These TLDs hold second-level domains like `example.com`, which may further hold subdomains like `mail.example.com` or `blog.example.com`.

Each level within the DNS hierarchy may be operated by a different administrative organization and thus can offer decentralized control. For instance, whereas the `.com` domain is operated by a

TLD operator, the owner of `example.com` is supposed to operate its subdomains.

DNS data for each domain is stored in zone files, which are plain text files containing mappings between domain names and IP addresses, along with other relevant information. A zone file typically includes various resource records (RRs), such as:

- **A (Address) Record:** Maps a domain name to an IPv4 address.
- **AAAA Record:** Maps a domain name to an IPv6 address.
- **MX (Mail Exchange) Record:** Specifies mail servers for the domain.
- **NS (Name Server) Record:** Lists the authoritative name servers.
- **CNAME (Canonical Name) Record:** Points a domain to another domain.

Zone files are maintained by domain administrators and served by authoritative name servers. They define the boundaries of DNS zones, which may encompass a single domain or multiple subdomains under common administrative control.

2.3 Vulnerabilities in the Traditional DNS

Even though DNS was designed with performance and scalability, it has no inherent security features. Its largest vulnerability is that it relies on unauthenticated data transfer. Ancient DNS responses were sent in plaintext and can be intercepted, manipulated, or spoofed by bad actors.

DNS Spoofing or **cache poisoning** is a common attack whereby bad guys introduce forged data into a resolver's cache. If poisoned, the resolver will give an incorrect IP address for certain domain names, causing possible malicious diversion of users to harmful websites without their knowledge.

Another weakness comes from the absence of data origin authentication. Since resolvers are unable to authenticate the data they receive, they can be manipulated into accepting spoofed responses. This vulnerability puts users at risk of **man-in-the-middle attacks**, phishing, and malware distribution.

In addition, DNS is vulnerable to **Denial-of-Service (DoS)** attacks, wherein attackers send large amounts of bogus requests to DNS servers, which flood the system and result in service disruptions. Amplification attacks also take advantage of DNS's capability to send big responses for small queries, magnifying the effect of DoS campaigns.

These weaknesses have created a necessity for improved security measures such as DNSSEC, which covers data integrity and origin authentication, thus making the trust model of the DNS environment more secure.

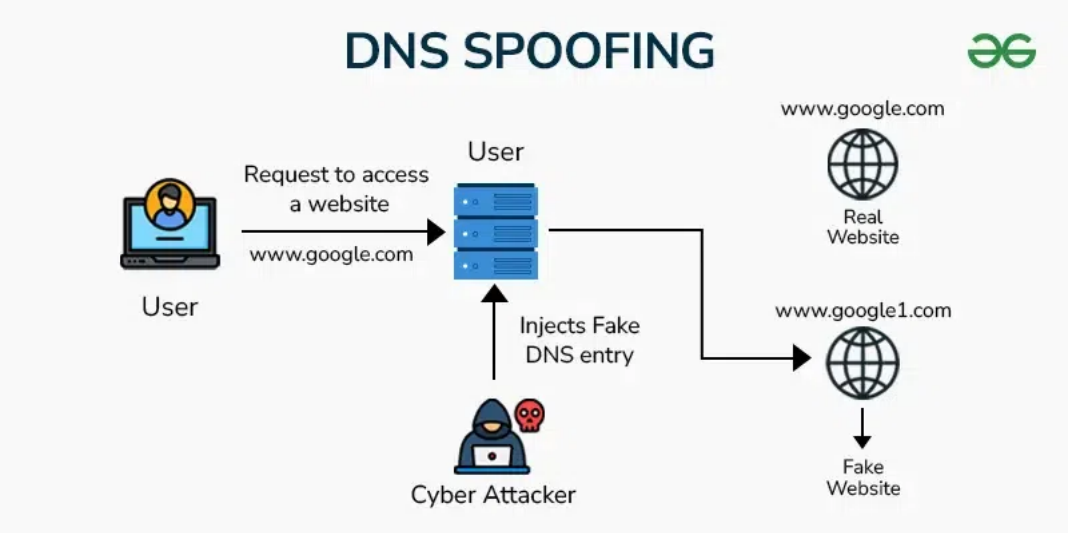

Figure 1: DNS spoofing

3 What is DNSSEC?

3.1 Definition and Goals

DNS Security Extensions (DNSSEC) is a security protocol framework that protects the integrity and authenticity of information contained in the Domain Name System (DNS). Developed by the Internet Engineering Task Force (IETF), DNSSEC emerged as a solution to address increasing concerns regarding the security vulnerabilities of traditional DNS, especially its vulnerability to cache poisoning and spoofing attacks.

The main objective of DNSSEC is to verify that DNS responses come from a valid source and have not been tampered with during transit. In contrast to standard DNS, which provides no way to confirm the validity of a DNS record, DNSSEC offers a cryptographic guarantee that the received data is precisely what the domain owner published. It achieves this through the application of digital signatures and public key infrastructure (PKI) to authenticate the authenticity and integrity of DNS data.

3.2 Key Features

DNSSEC introduces several features that distinguish it from traditional DNS:

- **Data Integrity:** DNSSEC uses digital signatures to confirm that DNS data has not been modified.
- **Origin Authentication:** Clients can verify that the DNS data was provided by the legitimate authoritative server.

- **Public Key Cryptography:** Domain owners use private keys to sign records, and resolvers use the corresponding public keys to validate them.
- **Chain of Trust:** Trust is established through a hierarchical chain that begins at the DNS root and extends down to individual domain zones.
- **New Record Types:** DNSSEC adds several new DNS record types, including RRSIG (digital signatures), DNSKEY (public keys), DS (delegation signer), and NSEC/NSEC3 (to prove non-existence of records).

While DNSSEC does not provide encryption or confidentiality of data, it significantly strengthens the trust model by ensuring the authenticity of DNS records.

3.3 How DNSSEC Works (High-Level Overview)

At a broad level, DNSSEC operates by digitally signing DNS information. Upon securing a domain zone using DNSSEC, every collection of DNS records (RRsets) is signed with a private key. The resulting digital signature is kept in an RRSIG record. To ensure data authenticity, DNS resolvers utilize an equivalent public key, published inside a DNSKEY record.

If a resolver requests a DNSSEC-enabled domain, it gets both the DNS record and its signature. Then it gets the DNSKEY and checks against the signature. To authenticate, the public key itself needs to be authenticated through a chain of trust, starting from the DNS root zone. This is accomplished through Delegation Signer (DS) records inserted into parent zones that lead to the public key in the child zone.

If the validation is successful, the resolver is aware that the data is genuine and intact. Otherwise, the resolver rejects the response, hence not allowing forged or malicious DNS information to be utilized. This procedure greatly helps in eliminating threats such as cache poisoning, making the internet a safer place for users and services.

4 Cryptographic Foundations of DNSSEC

DNSSEC relies heavily on mature cryptographic notions to provide authenticity and integrity to DNS data. In this section, the main cryptographic components that form the basis of DNSSEC are discussed.

4.1 Public Key Cryptography

DNSSEC uses public key or asymmetric cryptography to verify that DNS data has not been tampered with. Each zone in DNSSEC contains a private key to sign DNS records and a public key that is published using DNSKEY records. The resolver uses the public key to verify that the data was indeed signed by the owner of the zone.

4.2 Digital Signatures

A digital signature ensures the integrity and authenticity of DNS data. If a DNS record set (RRset) is signed by a zone owner, then the private key is utilized to generate the signature. It is included in an RRSIG record with the information. DNS resolvers will verify this signature using the associated public key from the DNSKEY record. If the signature is accurate, then it confirms that data was not manipulated during transportation and is, in fact, from the authentic source.

4.3 Hash Functions (e.g., SHA-2)

Hashing algorithms such as SHA-2 are employed within DNSSEC to generate fixed-size digests from variable-length inputs. The digests are employed in the signature process and DS record generation. The DS record within a parent zone holds a hash of the child zone's DNSKEY, creating a secure and verifiable connection between the two. Secure hash algorithms make it so that even minor variations in DNS records will yield markedly different digests, making it easy to detect tampering.

4.4 Key Pair (KSK and ZSK)

DNSSEC typically uses two types of cryptographic key pairs for better management and security:

- **Zone Signing Key (ZSK):** Used to sign DNS data within the zone (e.g., A, MX, TXT records).
- **Key Signing Key (KSK):** Used only to sign the DNSKEY RRset that contains the public ZSK.

This separation allows administrators to roll over ZSKs more frequently without impacting the trust chain, while the KSK remains stable and is only updated occasionally. The KSK's corresponding DS record is published in the parent zone to maintain the chain of trust.

5 DNSSEC Architecture and Components

DNSSEC enhances the traditional DNS infrastructure by introducing new components and mechanisms designed to ensure authenticity and integrity of DNS data. Its architecture integrates crypto-

graphic elements into the DNS hierarchy, forming a secure framework for DNS resolution.

5.1 Resource Record Types

DNSSEC introduces several new DNS resource record (RR) types, each serving a specific function within the security framework:

- **RRSIG (Resource Record Signature):** This record contains the digital signature of a DNS record set (RRset). It is generated using a private key and is used by resolvers to verify the authenticity of DNS responses.
- **DNSKEY:** Stores the public keys that correspond to private keys used to sign RRsets. Resolvers use DNSKEY records to validate RRSIGs. Each zone can have two types of keys: the Zone Signing Key (ZSK) and the Key Signing Key (KSK).
- **DS (Delegation Signer):** Located in the parent zone, this record provides a hash of the child zone's DNSKEY. It establishes a link between the parent and child zones, forming part of the chain of trust.
- **NSEC and NSEC3:** These records are used to provide authenticated denial of existence. NSEC lists the next valid domain name, while NSEC3 adds cryptographic hashing to obscure zone contents, mitigating zone enumeration attacks.

These records work in unison to validate DNS responses, prove data non-existence, and maintain a secure link between DNS zones.

5.2 Chain of Trust

The chain of trust is the basis of DNSSEC's security model. It enables resolvers to authenticate responses by following signatures starting from the zone being queried right up to the root of the DNS. The root zone itself is the anchor of trust that signs the DNSKEYs for TLDs (e.g., `.com`, `.org`). Each TLD then signs DS records for second-level domains, and so forth.

To be able to validate a DNS record, a resolver must check whether:

1. The RRSIG on the record is authentic through the use of the DNSKEY.
2. The DNSKEY is associated with a DS record in the parent zone.
3. The DS record is authenticated by the parent zone's RRSIG and DNSKEY.

This chain of recursive authentication continues all the way to the root zone, fulfilling the chain of trust.

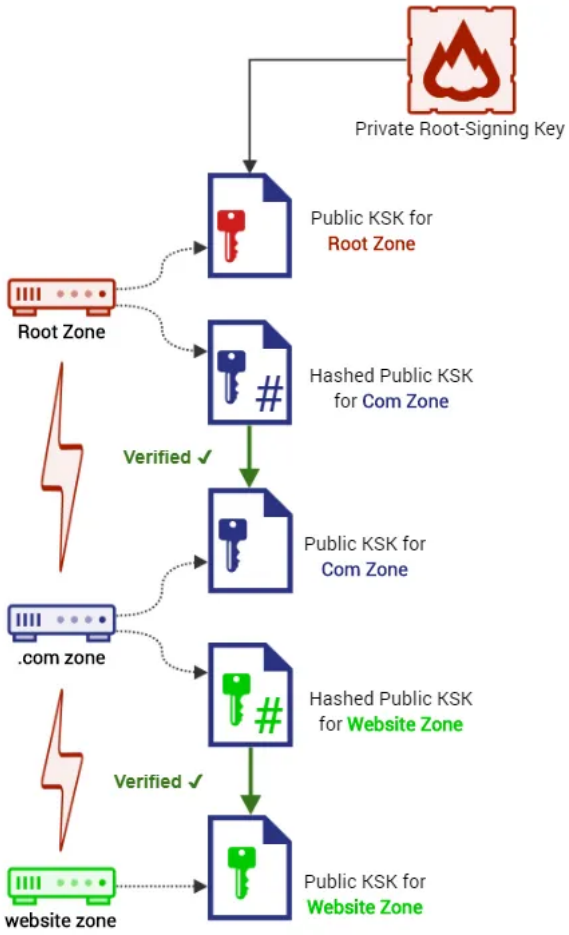

Figure 2: Chain of Trust Flow

5.3 Key Management and Rollover

Key management is a critical component of DNSSEC operations. Zones typically use two key pairs:

- **Zone Signing Key (ZSK):** Used to sign DNS data within the zone.
- **Key Signing Key (KSK):** Used to sign the zone's DNSKEY RRset.

KSKs are typically more stable and less frequently changed, while ZSKs are rolled over regularly to reduce exposure. Proper key rollover procedures are essential to maintain trust without interrupting service.

There are two main types of rollovers:

- **Pre-Publish Method:** The new key is published prior to use so resolvers have an opportunity to cache it before validation starts.
- **Double-Signature Method:** Both the new and old keys sign data during the transition, providing a smooth rollover.

Mismanagement of key rollovers can break the chain of trust, leading to resolution failures. Automated tools and DNSSEC-aware registrars are now available to help manage this process more reliably.

6 DNSSEC Resolution Process

DNSSEC resolution procedure enhances standard DNS resolution with cryptographic authentication steps. DNSSEC guarantees the data received by a resolver was not tampered with and is actually from the source claimed.

6.1 How DNSSEC Validates Responses

When a DNSSEC-enabled resolver queries a signed zone, it receives additional records such as `RRSIG`, `DNSKEY`, and possibly `DS`. The resolver uses the following steps to validate the response:

When the DNSSEC resolver asks for only a signed zone, it gets extra records like `RRSIG`, `DNSKEY`, and sometimes `DS`. The resolver performs the following steps to authenticate the response:

1. The resolver gets the public key (`DNSKEY`) of the zone.
2. It verifies the public key to check the DNSSEC digital signature in the `RRSIG` record of the requested RRset.

3. It checks that the public key itself is trusted, either because it has been securely set up (as with the root zone) or because its hash matches a `DS` record from a trusted parent zone.

4. This process recursively continues up the DNS hierarchy to the trusted root, creating a chain of trust.

If each verification step succeeds, the response is considered authentic.

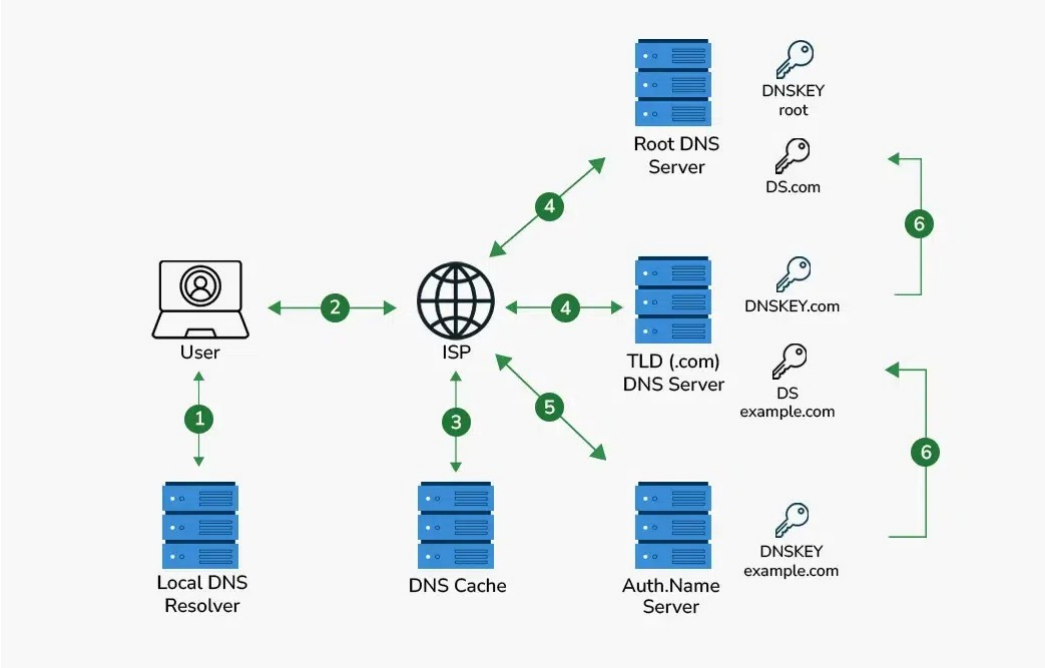

Figure 3: DNS Validation Process

6.2 Authenticated vs. Unauthenticated Data

DNSSEC makes a distinction between two categories of data:

- **Authenticated Data (AD):** This means that the data was authenticated by means of DNSSEC signatures and is reliable. DNS resolvers can tag data with such authenticity with the AD bit in the DNS response header.
- **Unauthenticated Data:** If a domain is not signed using DNSSEC or if the resolver is not validating, the data is taken without any form of cryptographic confirmation. This older behavior makes the data susceptible to spoofing and tampering.

6.3 Role of Validating Resolvers

Validating resolvers are DNS resolvers that are set up to verify the DNSSEC signatures of responses. Such resolvers make all the cryptographic verifications required using the trust anchor (in most cases,

the signed root zone) as a basis. If a response cannot be validated—because a signature has expired, because a
textttDS record is missing, or because hashes do not match—the resolver considers the data to be bogus and does not return it to the client.

Validating resolvers are key to enforcing DNSSEC's guarantee of integrity and stopping clients from obtaining forged or altered DNS information. Popular DNS resolvers such as BIND, Unbound, and PowerDNS validate DNSSEC and can be set up to automatically reject invalid information.

7 DNSSEC Deployment

7.1 Steps in Deploying DNSSEC

The deployment process typically includes:

1. Generating cryptographic key pairs (ZSK and KSK).
2. Signing DNS zone files with the ZSK.
3. Publishing the public DNSKEY records in the zone.
4. Generating and submitting DS records to the parent zone.
5. Configuring and testing validation with resolvers.

These steps ensure that DNS responses can be validated by clients using a trusted chain of signatures.

7.2 Tools and Software for Implementation

Several tools assist in DNSSEC deployment, including:

- **BIND**: A widely-used DNS server with built-in DNSSEC support.
- **OpenDNSSEC**: Automates key generation and signing.
- **Unbound**: A validating DNS resolver supporting DNSSEC.
- **LDNS/Drill**: A suite of tools for signing and debugging DNSSEC records.

7.3 Real-World DNSSEC Deployment

Many high-visibility domains implement DNSSEC. The
texttt.gov top-level domain was an early adopter, mandating use of DNSSEC by all U.S. federal domains. Similarly,

texttt.org, operated by the Public Interest Registry, implemented DNSSEC in 2009. The root zone itself was signed in 2010, providing a global trust anchor and incentivizing broader use.

8 Challenges and Limitations

Despite its advantages, DNSSEC has several practical and technical difficulties.

8.1 Performance and Latency

DNSSEC answers are larger because of extra records (e.g., RRSIG, DNSKEY), which can lengthen DNS query times and cause problems with older systems or firewalls that drop large UDP packets.

8.2 Key Management Complexity

Secure and timely rollovers of keys need to be planned carefully. Key publication or rollover mistakes can compromise the chain of trust, leading to unsuccessful DNS resolutions.

8.3 Zone Enumeration Issues

DNSSEC's original NSEC records enabled attackers to list all names in an NSEC signed zone. NSEC3 came to counteract this, but increased complexity without removing enumeration in some configurations fully, leading to controversy regarding privacy and efficiency.

8.4 Backward Compatibility

DNSSEC is not backward-compatible with legacy DNS infrastructure. Clients and resolvers that do not support DNSSEC ignore its records entirely, meaning DNSSEC only adds value in fully-aware environments. This limits its effectiveness in mixed deployments.

9 Comparison with Other DNS Security Approaches

While DNSSEC focuses on securing DNS data integrity and authenticity, other approaches, such as DNS-over-HTTPS (DoH) and DNS-over-TLS (DoT), address different aspects of DNS security, primarily confidentiality and privacy.

9.1 DNS-over-HTTPS (DoH)

DNS-over-HTTPS (DoH) encrypts DNS queries and responses over HTTPS, thus denying third parties the ability to intercept or manipulate DNS traffic. The approach uses the widely accepted HTTPS protocol, which is generally permitted by firewalls and proxies and is hence firewall-friendly compared to regular DNS. DoH has the capability of circumventing DNS-based censorship as well as improve user privacy since DNS traffic cannot be differentiated from other web traffic.

But whereas DoH encrypts the transport layer, it does not offer data integrity or authentication of DNS responses, which is the main function of DNSSEC. Moreover, DoH can also be challenging for network administrators, as it can circumvent traditional DNS filtering and monitoring tools.

9.2 DNS-over-TLS (DoT)

DNS-over-TLS (DoT) encrypts DNS traffic but employs the Transport Layer Security (TLS) protocol rather than HTTPS. Similar to DoH, DoT maintains the secrecy of DNS queries by blocking eavesdropping and man-in-the-middle attacks. It is easier to be transparent for network administrators since it employs a specific port (853) making it easier to filter and monitor DNS traffic compared to DoH that employs shared port 443 with HTTPS.

Nevertheless, DoT is plagued by the same disadvantage as DoH in that it does not ensure the integrity or authenticity of DNS information. Although DoT adds confidentiality, it needs to be combined with DNSSEC to form an overall security solution.

9.3 Complementarity and Differences

DNSSEC, DoH, and DoT complement each other by addressing different facets of DNS security:

- **DNSSEC** ensures that DNS data is authentic and hasn't been tampered with, but it does not provide confidentiality.
- **DoH** and **DoT** secure the transport layer, preventing eavesdropping and censorship, but they do not validate the authenticity of DNS data.

Together, these approaches provide a holistic security solution: DNSSEC ensures integrity, while DoH and DoT ensure confidentiality. Organizations can implement all three technologies to maximize the security and privacy of DNS queries and responses.

10 Conclusion

10.1 Summary of Key Points

In this paper, we have conducted a detailed analysis of DNSSEC (Domain Name System Security Extensions) and its pivotal role in guaranteeing the integrity and authenticity of DNS information. We have talked about the DNSSEC architecture, the cryptographic basis that sustains it, and the resolution process that confirms DNS replies. In addition, we contrasted DNSSEC with other security technologies such as DNS-over-HTTPS (DoH) and DNS-over-TLS (DoT), demonstrating how DNSSEC supports these technologies by emphasizing data authenticity, while DoH and DoT cover confidentiality.

10.2 Final Thoughts on DNSSEC's Role in Internet Security

DNSSEC is a key component in securing the Internet as a whole. While problems like key management and backward compatibility persist, its application continues to grow. DNSSEC is a key component of defending against attacks like cache poisoning and man-in-the-middle attacks, contributing to the stability of the DNS infrastructure. As security risks on the Internet evolve, the deployment of DNSSEC will remain a key strategy in making DNS a secure and trustworthy service.

References

[1] S. Ariyapperuma and C. J. Mitchell, "Security vulnerabilities in dns and dnssec," in *The Second International Conference on Availability, Reliability and Security (ARES'07)*, pp. 335–342, IEEE, 2007.

[2] A. S. Jahromi, A. Abdou, and P. C. van Oorschot, "Dnssec+: An enhanced dns scheme motivated by benefits and pitfalls of dnssec," *arXiv preprint arXiv:2408.00968*, 2024.

[3] G. Ateniese and S. Mangard, "A new approach to dns security (dnssec)," in *Proceedings of the 8th ACM conference on Computer and Communications Security*, pp. 86–95, 2001.

[4] R. van Rijswijk-Deij, A. Sperotto, and A. Pras, "Dnssec and its potential for ddos attacks: a comprehensive measurement study," in *Proceedings of the 2014 Conference on Internet Measurement Conference*, pp. 449–460, 2014.

[5] T. Chung, R. van Rijswijk-Deij, B. Chandrasekaran, D. Choffnes, D. Levin, B. M. Maggs, A. Mislove, and C. Wilson, "A longitudinal,{End-to-End} view of the {DNSSEC} ecosystem," in *26th USENIX Security Symposium (USENIX Security 17)*, pp. 1307–1322, 2017.

[6] M. Southam, "Dnssec: What it is and why it matters," *Network Security*, vol. 2014, no. 5, pp. 12–15, 2014.

[7] A. Herzberg and H. Shulman, "Dnssec: Security and availability challenges," in *2013 IEEE Conference on Communications and Network Security (CNS)*, pp. 365–366, IEEE, 2013.

(1) (2) (3) (4) (5) (6) (7)