International Journal of Information Security (2025) 24:106
https://doi.org/10.1007/s10207-025-01022-2

REGULAR CONTRIBUTION

Analyzing shared keys in X.509 certificates with domain ownership

Kashif Junaid[1] · **Muhammad Umar Janjua**[1]

Published online: 29 March 2025

Abstract
X.509 certificates play a crucial role in establishing trust and security in Public Key Infrastructure (PKI) systems, particularly in web communications. These certificates, issued by trusted certificate authorities (CAs), bind identities to public keys and play a critical role in verifying the identity of devices and endpoints during secure communication. However, the sharing of public keys among multiple certificates, termed "Key Sharing," can present security risks if not properly managed or if best practices are not followed. This study examines the implications, security threats, and potential misuse of shared public keys in X.509 certificates. Analyzing a dataset of 12,286,887 certificates, we identify 9245 unique sets with shared public keys. We introduce a novel association learning approach for domain ownership classification to distinguish between safe and unsafe sharing scenarios. The performance of this method is evaluated using real-world and synthetic datasets, demonstrating its effectiveness. To mitigate unsafe key-sharing practices, we actively collaborate with CAs and domain owners. Our research underscores the critical need for detecting and preventing risky key-sharing practices, providing actionable insights to enhance PKI system security and ensure safer online communications.

Keywords X.509 Certificates · Domain ownership · Web PKI · Key sharing

1 Introduction

X.509 certificates play a crucial role in establishing trust and identifying devices and endpoints within a Public Key Infrastructure (PKI) system. These certificates are issued by trusted third-party certificate authorities (CAs) to facilitate secure communication of sensitive data for end-users. During the TLS protocol handshake, the server presents an endpoint X.509 certificate for authentication and identification. This certificate contains a reference to the website's domain name, along with one or more intermediate certificates, allowing the client's browser to establish a trust chain from one of its end certificates to the root certificate. In each X.509 certificate, a public key is associated with the certificate's subject, typically represented by a domain name[8]. It's common for a certificate owner to possess multiple certificates for the same subject domain name, resulting in multiple certificates sharing the same public key. This prevalent scenario is what we refer to as 'Key Sharing'. Within the context of this study, we designate the collections of such shared keys as a 'Sharing Set' for thorough examination and analysis.

Key sharing is permissible and safe in cases such as certificate renewal, cross-signing, and CDN utilization. However, improper or unauthorized key sharing introduces significant security risks, necessitating a systematic analysis.

Improper key sharing can enable attackers to impersonate domains, facilitating man-in-the-middle (MITM) attacks and typosquatting. For example, attackers can deceive users by registering visually similar domain names, such as *azuree.com* instead of *azure.com*, and obtaining legitimate certificates for these domains[42]. Hence, it becomes imperative to determine the safety, legitimacy, and legality of the scenario when the same key is found to be shared among various certificates. Addressing these risks requires scalable, automated methods, which this study aims to provide through novel association learning techniques.

To bridge these gaps, this study addresses the following key questions:

– What are the potential risks and misuse scenarios associated with public key sharing?

✉ Kashif Junaid
phdcs21004@itu.edu.pk

Muhammad Umar Janjua
umar.janjua@itu.edu.pk

[1] Information Technology University of the Punjab, Lahore, Pakistan

– How can we distinguish between legitimate and unsafe key-sharing practices in X.509 certificates?
– How can domain ownership and association learning techniques be leveraged to automate the detection of risky key-sharing instances?

Prominent organizations like Facebook,[1] Amazon,[2] and Microsoft,[3] with their extensive ecosystems of services and products, manage thousands of X.509 certificates worldwide. Ensuring the detection, analysis, and classification of key-sharing instances into safe and unsafe categories is essential for maintaining trust and security within their Public Key Infrastructure (PKI) systems. To meet these challenges, there is a critical need for systematic, automated, and scalable solutions that can swiftly implement mitigations to prevent potential risks and ensure robust security.

In this paper, we discuss the potential reasons, associated security threats, sharing of valid as well as invalid public keys, and their use cases. For this purpose, we collect 12,286,887 different certificates issued by different CAs and perform a public key-sharing analysis. It's interesting to note that there are 9245 such unique sharing sets where the same public key is utilized by many certificates. To address the identified gaps, We propose a novel technique of domain ownership using association learning, which enables us to classify these sharing sets into safe and unsafe classes. Using a real-world 12 M certificates dataset and a synthetic dataset designed to assess the robustness of our proposed models, we compare the performance of our domain ownership association models. Then, using the best model among the provided models, we uncover potential unsafe sharing sets out of the 9245 unique sharing sets.

We actively engage with CAs and domain owners by sharing identified potential unsafe sharing sets. This collaboration aims to bolster the security of the PKI system. Our research highlights the significance of identifying worrisome key-sharing practices and potential threats. Additionally, we perform a manual analysis of predicted unsafe key sharing and demonstrate the utility of our domain ownership model in detecting risky domains and subdomains. We anticipate that our findings will contribute to the detection and prevention of unsafe key sharing, promoting a more secure online environment.

Contributions of the Paper: The main contributions of our paper are described as follows.

1. We address the problem of safety of public key sharing within and across X.509 certificates, and find 9245 key-sharing sets in the 12 M real-world certificates.
2. We use association learning techniques to learn domain ownership of keys and effectively use it to classify sharing instances into safe vs. unsafe categories.
3. We develop synthetic attack vectors for domains to test the association models.
4. We perform manual analysis of predicted unsafe sharing sets by involving CAs as well as domain owners and share our findings of non-compliance as acknowledged by CAs.

Organization of the Paper: The rest of the paper is organized as follows. Section 2 presents an overview of the literature review and identifies the challenges and gaps in existing research. Section 3 outlines the proposed concept and approach, while Sect. 4 details the problem formulation. Section 5 illustrates data and experimental design, and Sect. 6 presents the results of our experiments. Section 7 discusses the implications of our findings and their relevance to the field, it also mentions limitations and provides directions for future research. Finally, Sect. 8 concludes the paper. However, Sections 9, 10, and 11 highlight ethical considerations, discuss competing interests, and outline the research data policy, respectively.

2 Literature review

Understanding the risks associated with public key sharing in X.509 certificates is critical for enhancing Public Key Infrastructure (PKI) security. Despite extensive research on cryptographic vulnerabilities and PKI compliance, the specific challenges of public key sharing have received limited attention. In this section, we provide an overview of existing research, limitations, and the needs of our work.

2.1 Related attacks

In the realm of cybersecurity, attackers continually seek ways to compromise security measures and pilfer sensitive data from online sources. Among the notable security threats, the Man-in-the-Middle (MITM) attack stands out as a prominent adversary tactic.

DNS spoofing is a type of MITM attack that compromises communication between users and DNS responders by forging DNS responses. Attackers exploit this vulnerability to redirect users to malicious domains, creating an illusion of secure communication. Conti et al. highlight that DNS spoofing can be weaponized to intercept encrypted communication [9]. This becomes particularly critical when combined with improper public key-sharing practices, as attackers can exploit shared keys to impersonate legitimate entities, escalating the attack's impact.

[1] https://www.facebook.com/
[2] https://www.amazon.com/
[3] https://www.microsoft.com/

In a renegotiation attack, the client and server engage in renegotiation upon resuming the previously negotiated session. This attack exploits a vulnerability within this renegotiation process. By delaying the client's request for renegotiation, the attacker establishes a session with the server and transmits a message [20]. Shared public keys further exacerbate this risk, as vulnerabilities in one session can compromise multiple certificates associated with the same key.

Version rollback and cryptographic downgrade attacks, such as the Poodle attack, exploit weaknesses in protocol negotiation to enforce the use of outdated or insecure cryptographic algorithms. Han et al. demonstrate how attackers leverage these vulnerabilities to compromise encrypted communication.[22] When shared public keys are associated with weaker algorithms, attackers gain access to multiple systems relying on those keys, amplifying the risk.

There are attacks on public keys, such as multi-protocol attacks [3], that primarily target the security of public key authentication protocols. Public key (or asymmetric key) algorithms are theoretically considered to be more susceptible to attack compared to symmetric key algorithms due to the availability of the public key used for message encryption. This availability can potentially aid attackers because the message might reveal the specific public-key encryption technique used.

A prominent attack in this context is the Key Search attack, where the attacker attempts to derive private keys from given public keys and seeks weaknesses in the algorithm used to generate those public keys [37]. This highlights the significance of our study, as in this case, all the certificates using compromised public keys would become exploited.

Delignat and Bhargavan [14] have shed light on confusion attacks affecting virtual hosts that share TLS cache or ticket encryption keys. Additionally, they have identified potential threats targeting websites associated with CDNs and wildcard certificates. Building upon this research, Zhang et al. [48] have delved into the HTTPS Context Confusion Attack (SCC Attack) on shared certificates, illustrating how attackers can compromise a domain and its associated domains. Squarcina et al. [45] have focused on related-domain attacks, investigating the security implications of domains controlled by malicious actors. Their study encompasses various scenarios demonstrating how these attackers, in privileged positions, can pose security risks. These risks extend to affecting web browsers' site isolation mechanisms and exploiting subdomain takeover vulnerabilities. Brinkmann et al. [5] have explored cross-protocol attacks stemming from wildcard certificates. Their research illustrates how attackers can redirect encrypted traffic to alternative domains, highlighting potential vulnerabilities in the encryption process. Heninger et al. reveal widespread vulnerabilities in RSA and DSA keys due to insufficient entropy during key generation for TLS and SSH servers. These vulnerabilities, affecting a significant percentage of hosts, pose security risks in shared key-based encryption, particularly in headless or embedded devices.[24] Felsch et al. uncovered cross-protocol authentication bypasses and vulnerabilities related to key reuse in IPsec's Internet Key Exchange (IKE) protocols.[19] These findings underscore the significance of effective key management and the potential risks associated with shared key problems, aligning with our focus on analyzing key sharing and its implications in our research.

More real-world incidents underline the dangers and urgency of addressing improper key sharing. The DROWN attack exploited shared RSA public keys between SSLv2 and modern TLS protocols, allowing attackers to decrypt secure HTTPS traffic by leveraging weaknesses in SSLv2. This vulnerability affected over 33% of HTTPS servers at the time, emphasizing the critical risks posed by improper public key sharing [1].

Our paper focuses on the security risks posed by the sharing of a single public key across multiple X.509 certificates, particularly when associated with different domain owners. This issue is exacerbated in IoT devices, where shared public keys are prevalent. Kilgallin et al. analyzed 75 million certificates and identified key compromise risks, revealing that 1 in 172 keys could potentially be exploited.[30] While their work highlighted the dangers of RSA key compromises, it did not explore systematic approaches for classifying and mitigating unsafe key-sharing instances, leaving a critical gap addressed by our study.

2.2 Compliance with guidelines

This category focuses on compliance with CA/B Forum guidelines, surveys of HTTPS security, and evaluation of the proposals to enhance the security aspects of the CA/B model [7],[25]. Delignat-Lavaud et al. [13] perform a detailed analysis of compliance with CA/B Forum guidelines. By performing analysis of publicly trusted certificates for compliance with specific guidelines they provide a clustering mechanism of certificates that automatically derives templates describing CA behavior. Thus allowing visualization of the PKI in terms of the issuance policies and violations. This further helps to discover vulnerabilities in certificate validation libraries and the templates used by different CAs to issue certificates. However, their work lacks the quantitative assessment of compliance with the guidelines. Also, it doesn't quantify the level of CAs.

Farhan et al. [18] explored the evolution of TLS certificates over eight years, highlighting persistent issues like invalid certificates and misuse of template certificates. However, their work did not address the implications of private key sharing among certificates, leaving a gap in understanding post-issuance risks. Kumar et al. [31] introduced the ZLint

tool to detect errors and warnings in certificate issuance, focusing on reducing misissuance by certificate authorities (CA). Their analysis was purely rule-based and did not use machine learning models to identify patterns of mismanagement or predict risky behaviors. We use the domain ownership and association methods to classify unsafe key sharing.

Stark et al. [46] emphasize Certificate Transparency (CT) log monitoring and the importance of domain owners tracking improper certificates. However, their work did not address how CT logs or similar mechanisms can detect unsafe key-sharing practices. Khan et al. [28] extensively discussed PKI issues and certificate revocation mechanisms, emphasizing the potential integration of blockchain for transparency and security. Their survey emphasized certificate revocation mechanisms and blockchain-based improvements but did not delve into post-issuance vulnerabilities like key sharing, which can compromise systems despite strong revocation mechanisms. Our approach addresses these gaps by introducing a classification system to detect unsafe key sharing and improving PKI security practices.

Durumeric et al. [17] propose recommendations for the PKI system after regularly scanning the entire IPv4 address space for certificates. Their work is focused on the analysis of the category of organizations involved in the HTTPS ecosystem, practices followed by CA, and the adoption of different features in the site certificates.

DigiNotar Certificate Authority breach (2011) enabled attackers to issue over 500 fraudulent certificates, which shared the same compromised keys. This breach facilitated man-in-the-middle attacks targeting Iranian Gmail users, resulting in the collapse of DigiNotar and loss of trust in digital certificates [15]. The ROCA vulnerability arose from improper RSA key generation, making keys predictable and easy to factor [41]. Although it did not involve shared keys directly, it showcased how flawed cryptographic implementations can compromise entire systems. Like the Estonian ID Card issue, ROCA highlights the dangers of weak key generation, underscoring the need for robust cryptographic standards.[38]

Larimer et al. [35] proposed the certificate pinning technique associating a domain with specific public keys to enhance security. However, this approach is limited by scalability issues and is vulnerable to first-visit exploitation, particularly in cases of improper key sharing. These limitations underscore the need for proactive mechanisms to detect and manage unsafe key-sharing practices. Our study fills this gap by introducing a domain ownership model that systematically identifies and mitigates risky key-sharing instances, complementing techniques like certificate pinning.

Cangialosi et al. raise concerns about the security and privacy implications of private key sharing which is violating the practice that all private keys should be kept private. They highlight that private key sharing is widespread, and a small number of hosting providers possess the keys of the majority of popular websites which is alarming, as it involves the transfer of sensitive information without most users being aware of it[6]. Similarly, the Heartbleed vulnerability (2014) in OpenSSL allowed attackers to read server memory and potentially expose private keys. If such compromised private keys were associated with multiple certificates sharing the same public key, all those certificates would be at risk. This incident underscores the importance of unique key pairs for each certificate to minimize widespread exposure [23].

Hue et al. scrutinize the security of WPA2-Enterprise Wi-Fi services provided by educational institutions, highlighting vulnerabilities stemming from weak connection configurations and TLS setups that may lead to credential theft. The paper also delves into various misconfiguration issues and potential cases of private key reuse [26].

Another article investigates TLS proxies used by antivirus and parental-control software, highlighting significant vulnerabilities. These proxies, when intercepting TLS traffic, can potentially enable server impersonation, which raises concerns about the security of shared keys used in TLS communication. This research underscores the risks associated with TLS interception tools that could affect millions of users and their shared key-based security.[12]

Additionally, systemic vulnerabilities in IoT devices have been revealed, with over 250,000 routers found to share identical SSH keys, creating a substantial attack surface [44]. Hence, highlighting the need for robust key-sharing detection.

Zhang et al. [47] discuss the ownership of domains with dangling DNS records and present the hosting checker framework. Our work focuses on the domain name ownership model, whether a certain domain name belongs to a particular owner or not.

2.3 Anomalies detection using machine learning

Machine Learning is playing an interesting role in the security domain. Dong et al. [16] use machine learning techniques to find potentially rogue certificates. Li et al. use different machine learning algorithms to detect benign and malicious certificates[32].

These studies address general anomaly detection but do not tackle the specific challenges posed by key sharing. To bridge this gap, we use association learning techniques to classify key-sharing instances by domain ownership, providing a scalable way to determine if key-sharing is safe or unsafe.

3 Understating public keys association with domain names

Certificates are issued to specific domain owners against a single domain name or a set of particular domain names, allowing each public key to be associated with either a single domain name or a set of multiple domain names [4]. Consequently, there are multiple types of associations, which are defined below.

(a) Same Key associated with Different Domains - Within the same certificate: This is a common scenario where certificates cover multiple domains (e.g., SAN entries). It is considered secure when issued correctly.
(b) Same Domain associated with Different Keys - Across different certificates: This occurs during certificate renewals or key rotations. This scenario is also considered secure.
(c) Same Key associated with Different Domains - Across different certificates: This scenario raises significant security concerns and is the focus of this study.
This scenario raises significant security concerns and is the focus of this study. It is highly improbable that the Subject Alternative Names (SAN) list of one organization's X.509 certificate would include the domain or sub-domain of an entirely unrelated organization. For example, Microsoft's X.509 certificate should not contain amazon.com in its SAN list. These two organizations are distinct entities; consequently, their X.509 certificates should possess different public keys. Nevertheless, concerns regarding the security and issuance of X.509 certificates arise when unrelated entities with different domains and sub-domains inexplicably share the same public key within their X.509 certificates. Figure 1 illustrates examples of the same public key found in different certificates.

Fig. 1 Same keys in different certificates. In both certificates *Public Key* is the same

4 Problem formulation

In the Public Key Infrastructure (PKI) system, certificates associate public keys with domain names, playing a vital role in identification and trust-building. The validity of sharing the same public key across different certificates hinges on specific conditions. Legitimate scenarios for key sharing encompass certificate renewals, cross-signing, and the utilization of certificates within Content Delivery Networks (CDNs). However, invalid key sharing can manifest for a variety of reasons, including public key forgery, misconfiguration during certificate issuance, vulnerabilities in signature algorithms, or fraudulent certificate issuance.

Within the PKI framework, a key pair serves as a representation of an individual's identity. Consider the following scenario: an IT administrator transitions from one company to a competitor and retains their previous company's key pair. In this scenario, the individual gains access to the former company's configurations, potentially enabling them to intercept and reroute network traffic.

During a change in website or domain ownership, the previous owner may retain control of old certificates and the associated security keys, allowing them to impersonate the website even when it's under new ownership. This key sharing presents security risks, as it can be exploited by domain squatters and attackers. They might misuse these old certificates or exploit registrar policy loopholes to acquire certificates for websites they don't legitimately own. This underscores the critical need to address key-sharing issues during domain ownership changes to prevent unauthorized access and misuse of keys. [34]

Another motivation for this research lies in the vulnerability associated with attackers efficiently computing the prime

factors of cryptographic keys. This capability allows them to derive private keys, posing a significant threat to the decryption and impersonation of secure communications relying on those keys [30]. As the number of keys in use proliferates, so does the probability of discovering weakly generated factors within public keys. Such discoveries could compromise certificates that share those keys.

Real-world incidents further emphasize the risks of improper key sharing. A summary of the attacks related to key sharing is presented in Table 1, which draws attention to the analysis of key sharing and distinguish between legitimate and malicious entities. It becomes imperative to assess the safety, legitimacy, and legality of scenarios where key sharing occurs among various certificates with different owners. Currently, analyzing a single public key shared among a vast number of certificates is an arduous task. It is labor-intensive, necessitating domain expertise and substantial time and resources to scrutinize each scenario.

To address the aforementioned limitations, we aim to develop an association learning approach that tackles the problem of determining the safety and legitimacy of public key sharing across multiple certificates.

4.1 Defining sharing sets

To detect key sharing, we first need to find a set of X.509 certificates having same public keys (PK). Since the same key is used by several distinct certificates, we call these sets **sharing sets**. We define a sharing set *SS* as below:

$$SS = \{K_i = K_j \& h(C_i) \neq h(C_j)\} \quad (1)$$

where SS is the sharing set, K_i is the public key of certificate C_i and K_j is the public key of certificate C_j and $h(C_i)$ is the hash of the certificate C_i and $h(C_j)$ is the hash of certificate (C_j).

4.2 Identify domain ownership with sharing keys

To address the previously mentioned problem, we can utilize the concept of domain ownership. Domain ownership can be defined at various levels of granularity, but in this paper, we focus on determining domain ownership at the highest organizational level within the structure of a company. Since the Subject Alternative Names (SAN) list encompasses multiple domain and sub-domain names, potentially associated with a single key and indirectly with a specific owner, our goal is to learn and establish the association between SAN entries and their respective owners.

4.3 Valid use cases for public key sharing

Not all key sharing is unsafe; there are valid scenarios where public keys are shared across certificates. Below, we categorize these scenarios based on our analysis.

1. *Renewal of Expired Certificates*: Certificate renewal involves the extension of X.509 certificates with new expiration dates while retaining the same public key. This ensures continuity and compliance with evolving security standards, making it a valid key-sharing scenario.
2. *Domain Names hosted with CDN's X.509 Certificates*: In some cases, Content Delivery Networks (CDNs) issue X.509 certificates for websites, reducing costs for domain owners. This practice involves using the same public key across multiple certificates to provide secure connections between the origin server, CDN server, and end-users.
3. *Cross Signing of Certificates*: Cross-signing involves multiple trusted Certificate Authorities (CAs) validating certificates to create redundant trust paths. This is a legitimate key-sharing case that enhances reliability within the PKI framework.

4.4 Risks and attacks exploiting public key sharing

Improper public key sharing introduces significant vulnerabilities, even in valid use cases such as certificate renewal, cross-signing, or using the same key for different purposes (e.g., encryption and digital signatures). Table 1 highlights real-world attacks that exploit these scenarios, emphasizing the risks associated with improper key-sharing practices.

5 Data and methods

In this section, we describe our experiments. We start by first describing our data processing methodology and then explaining the experimental design in detail. Finally, we present how our domain ownership model can be useful for detecting safe and unsafe key sharing and protecting users/industry against relevant threats and compliance issues. Figure 2 presents an overview of our workflow.

5.1 Smart data processing

We use two crawlers-SSL Crawler and Local Crawler-to collect X.509 certificate data from Alexa's Top 1 Million Websites [2] and Rapid7's Project Sonar datasets [39]. Subsequently, we conduct feature engineering tasks on the collected data, as detailed in the following sections.

Table 1 Attacks Exploiting Key Sharing

Attack name	Description	Public key sharing essential?	Without public key sharing?
Shodan SSH Key Sharing[44]	Over 250,000 IoT devices were found sharing identical SSH keys, creating a significant attack surface for botnets	Yes, sharing identical keys is critical	Not Possible
Estonian ID Card Attack[38]	Flawed key generation led to shared keys among users, enabling impersonation attacks	Yes, sharing flawed keys caused the issue	Not Possible
ROCA Vulnerability[41]	Weak key generation due to improper implementation of RSA libraries, making keys factorable	No, weak key generation is the cause, not sharing	Possible
DigiNotar Breach[15]	Attackers issued over 500 fraudulent certificates using compromised keys, enabling large-scale MITM attacks	Yes, public key misuse was central	Not Possible
Related Domain Attack[45]	Attackers exploit misconfigured or closely related domains that share TLS certificates to confuse users, redirect traffic to insecure sites, and carry out phishing attacks	Yes, sharing certificates is required	Not Possible
Confusion Attack (ALPACA)[48]	Exploiting shared certificates to reroute traffic between protocols for phishing and session hijacking	Yes, shared certificates are required	Not Possible
Context Confusion Attack[14]	HTTPS MITM attacks leveraging shared certificates between multiple domains	Yes, shared certificates are required	Not Possible
Version Rollback Attack[22]	Exploiting protocol negotiation vulnerabilities to enforce insecure cryptographic algorithms	No, protocol negotiation flaw is exploited	Possible
Typosquatting Attack[29]	Attacks leveraging visually similar domain names, e.g., ‘azuree.com‘ instead of ‘azure.com‘, to mislead users	No, it relies on domain names, not keys	Possible

5.1.1 Data collection and description

We developed custom SSL and Local Crawlers for data collection. SSL crawler takes input from Alexa's Top 1 Million Websites URL list and sends the HTTPS request to each of the URLs, upon the response of this request, we get the X.509 certificate string in PEM format and then we build a chain from this certificate string and extract data in JSON format as described in Step 2 and 3 in Fig. 2. Whereas, the local crawler is used to extract certificate strings downloaded from Rapid7's Sonar dataset. We use these certificate strings directly as input to build trust chains and extract data in JSON format (as described in Steps 2 and 3 in Fig. 2).

Moreover, we locally dump each of the files on local storage separately for each of the certificate chains in JSON as well as in PEM format as shown in step 4. Each extracted JSON file contains all the certificate data fields. The collection includes certificate fields such as serial length, version, issuer, public key size, validity period with dates, subject information, signature algorithm information, raw PEM format strings, self-signed status, X.509 chain status, and certificate extensions.

5.1.2 Feature engineering task

The data is pre-processed before forming the feature vector. After extracting the certificate data fields, we perform different CA/B Forum guidelines check using one of the linter programs named zlint [31]. The zlint program performs nearly 220 different checks on each of the X.509 certificates separately to check the compliance requirement according to the pre-defined standards and store the result of all the checks in JSON format. Some of the checks that may include are, checking the version of the certificate whether it's a version (V3) or any other version of the certificate, algorithms used to generate key pairs, etc. After executing the zlint program (as described in Steps 5 and 6 of Fig. 2) on each of the certificates, data grows exponentially. We only keep valid and trusted certificate files as per zlint checks (see Step 7 of Fig. 2). For a single certificate, we are collectively producing three extra files storing zlint checks results, trust chain data fields, and certificate strings. To store these large numbers of files in an optimum fashion, we use a NoSQL database MongoDB due to its efficiency in handling large-scale JSON data storage and quick retrieval of certificate records for analysis.(see Step 8 in Fig. 2). We then extract CSV from MongoDB for

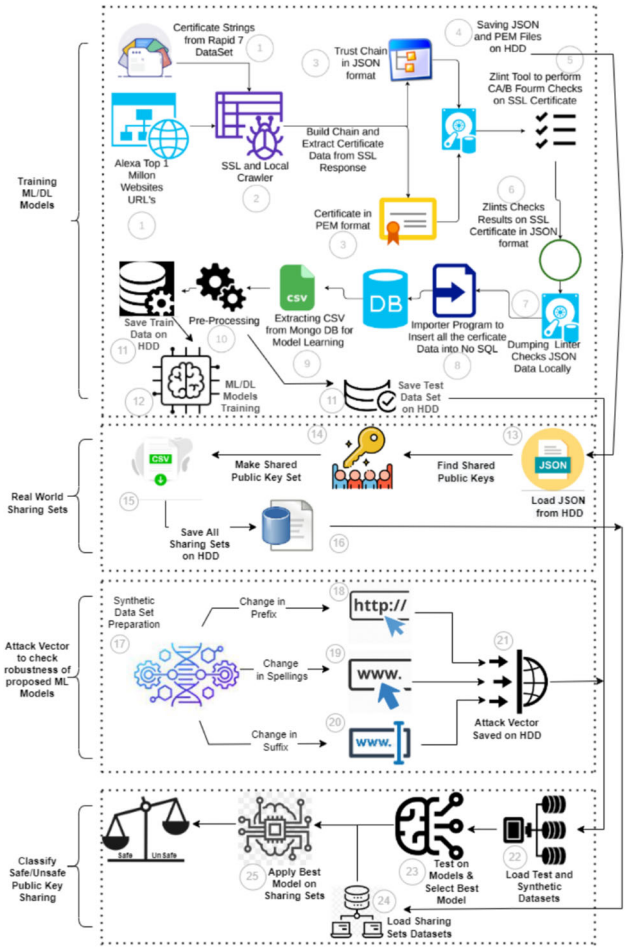

Fig. 2 Complete Workflow: Data gathering, Extracting Sharing Sets, Training/Testing Models, Synthetic Vectors and Classification of Safe/Unsafe Sharing

pre-processing in Step 9. Some of the pre-processing steps that we perform on our data are described below.

1. Splitting the Domain Names based on '.' to find the Sub Domains and Top-Level Domains
2. Removal of duplicate instances
3. Filtering out instances with missing values
4. Generating combinations of Sub Domains of single Domain Name

The pre-processed data contains a public key, certificate hash, issuance, and expiry date of the certificate, organization name to whom the X.509 certificate is issued, organization from which the X.509 certificate is issued, and domain name. The pre-processed data is split into train and test data and saved on HDD for association learning experiments (as described in Steps 10 and 11 of Fig. 2). After that, different association models like N-Gram, CBOW, and Skip-Gram are trained on training data.

5.2 Experimental design

This section outlines the methodology employed in our research, detailing our approach for data collection and experiments. Figure 2 presents an overview of our approach for data collection and experiments. We collect data and apply different training models to learn Domain Ownership. We also find real-world sharing sets from the above-collected data. To test the resilience of our proposed association learning models we prepare attack vectors of synthetic data (see Fig. 2).

We formally define a set of rules to classify sharing sets into safe and unsafe classes. Upon analyzing the domain owners of all the certificates in our sharing sets, we categorize a particular sharing set as safe if all of its certificates belong to the same single organization. To strengthen our assumption we also further examine other fields of the certificate, such as the SAN, to identify anomalies in the SAN lists of the certificate. We define an anomaly in SAN as the presence of an ambiguous, unrelated, or attack domain name in the SAN field of the certificate. To detect anomalies, we utilize the Concept of Domain Ownership, which associates domain names with their respective owners. This association learning helps us analyze domain names and their actual owners by extracting association scores from the data.

5.2.1 Real world sharing sets extraction

The collected data contains approximately 12 Million unique instances of X.509 certificates. After pre-processing and cleaning the data, we produce the sharing sets as per the criteria defined in Eq. 1 and collect 9245 sharing sets in nearly 12 million crawled certificates as shown in Steps 13 to 16 of Fig. 2.

5.2.2 Association learning for domain ownership

To learn the associations between the domain names and their respective owners we use three different association models; N-Gram [10], Skip-Gram [21] and Continuous Bag of Words (CBOW) [33]. For the N-Gram model, we split the Domain Names into their subparts based on the dot (".") and create all the bigram combinations between each sub-domain and its owner, as shown in Table 2. To find the bigram combinations probabilities, we utilize the Markov assumption mentioned in Eq. 3 where we find the probability score of the given word using the adjacent previous word only rather than utilizing the Chain Rule of probability defined in Eq. 2, where we consider all the previous words in the text. So in our case, the owner name and sub-domain are placed adjacently in the input.

We calculate the overall association score of each of the sub-domains with its associated owner and take the average

Table 2 Combinations of Domains and sub-domains with the owner

Splitted domains	Owners
Pay	Expedia
Expediagroup	Expedia
Com	Expedia
expediagroup.com	Expedia
pay.expediagroup.com	Expedia

Table 3 Sub-domain Scores

(Domain name,owner)	Scores
Expediagroup,Expedia	60.3
Com,Expedia	75.6
expediagroup.com,Expedia	89.6
pay.expediagroup.com,Expedia	94.5

of the probabilities to assign a single value to the domain name as shown in Table 3. The algorithm 1 describes the procedure for quantifying the domain name against a particular owner considering all the sub-domains of the Domain Name.

We then apply CBOW and Skip-Gram models to find the association scores of domain names and owners on the same bigram combination's data as described above and shown in Table 2. CBOW model focuses on the current target word based on the source context words while Skip-Gram behavior is the reverse of CBOW and it focuses on different contexts in which a word can be used. So, we apply the same technique as mentioned in Eq. 3 and algorithm 1 to calculate the association scores for these models too.

$$P = \{W_i | W_1, W_2, W_3, W_4, ... W_{i-1}\} \tag{2}$$

$$P = \{W_i | W_{i-1}\} \tag{3}$$

5.2.3 Synthetic certificates preparation

To address the class imbalance within our dataset, where we have only 9245 sharing sets among 12 million X.509 certificates, we apply oversampling techniques to the minority class, representing sharing sets. This imbalance arises due to the limited number of sharing sets available.

To enrich our dataset with more sharing sets, we create synthetic certificates by generating X.509 certificates with the same public keys. These synthetic certificates undergo controlled modifications to their Domain Names, mimicking various potential attack scenarios. This synthetic dataset includes a range of attack vectors, providing a robust testing ground for our models.

Algorithm 1 Algorithm for Finding a Domain Ownership Association Score

```
1: procedure FIND_DOMAIN_OWNERSHIP_SCORE(DomainName, Owner)
2:     Input: DomainName: The domain name, Owner: The owner
   of the domain
3:     Output: Domain ownership association score
4:     d[] ← SplitDomain(DomainName)
5:     P ← 0
6:     for i ← 1 to n do
7:         p[i] ← AssociationProbByModel(d[i], Owner)
8:         P ← P + p[i]
9:     end for
10:    P ← (P/n) × 100
11:    return P
12: end procedure
```

This strategy mitigates class imbalance while enhancing evaluation validity and comprehensiveness. This enables us to effectively differentiate between valid and invalid instances of public key sharing in the context of shared keys. We refer to these artificially generated X.509 certificates as "Synthetic certificates".

In our model testing process, we utilize the fuzz testing technique to produce synthetic certificate data. We manipulate the associations between Domain Names and their respective owners, following steps 17 to 21 outlined in Fig. 2. One common attack scenario we consider is, where attackers exploit slight variations in domain names. To evaluate our model's resilience against potential Domain Name variations, we introduce controlled changes to various components of Domain Names, including suffixes, prefixes, and spellings[11][29].

The described modifications, such as changing prefixes, suffixes, spellings, and domain name owners, are useful for testing and identifying safe and unsafe key-sharing scenarios in the context of public key-sharing and domain ownership. Here's how they can help in this regard:

- *Modifying Prefixes of Domain Names:* By altering the prefixes of domain names, we can simulate scenarios where attackers attempt to use domain names that closely resemble legitimate ones. Testing how well our model distinguishes between valid prefixes and modified ones helps identify potential unsafe key sharing scenarios. If the model correctly classifies these modified prefixes as unsafe, it indicates that it can detect attempts to use similar domain names for malicious purposes.
- *Modifying Suffixes of Domain Names:* Changing the suffixes of domain names introduces variations in the top-level domains (TLDs). This can simulate cases where attackers use slightly different TLDs to create domain names that appear valid but are not. Detecting these

Table 4 Synthetic Data prepared by fuzzing technique

Valid domain name	Valid owner	Change in prefix	Change in suffix	Change in spellings
pay.expediagroup.com	Expedia	prorewards.pay.expediagroup.com	pay.expediagroup.com.ir	pay.expediaa group.com
azure.com	Microsoft	aws.azure.com	azure.com.pk	azuree.com
aws.com	Amazon	google.aws.com	aws.com.ir	awws.com
cloud.google.com	Google	s3.cloud.google.com	cloud.google.com.edu	cloud.gooogle.com

modifications as unsafe indicates the model's ability to identify potentially malicious domain variations within key-sharing scenarios.

- *Altering Domain Name Spelling:* Modifying the spelling of domain names mimics typo-squatting attacks, where attackers register misspelled versions of legitimate domains. If our model can flag these slight spelling changes as unsafe, it demonstrates its capability to detect key-sharing scenarios that involve maliciously crafted domain names.
- *Changing Owner of the Domain Names:* Deliberately altering domain ownership associations tests the model's ability to identify key-sharing scenarios where unrelated owners attempt to use the same public key. If the model correctly identifies such changes as unsafe, it indicates its capacity to recognize unauthorized sharing of keys among different owners.

In summary, these modifications allow us to create synthetic scenarios that resemble real-world attack vectors and invalid use cases of public key sharing. By evaluating how well our model performs in classifying these modifications as unsafe, we can assess its effectiveness in identifying unsafe key sharing scenarios and enhancing security within the domain ownership and public key infrastructure. Table 4 contains a sample set of synthetic modifications.

6 Evaluation and results

In this section, we first present the experimental results of our domain ownership association models proposed in section 5.2. We then evaluate these models on synthetic datasets in sub-section 6.3 to check their robustness. Furthermore, we compare these models. Importantly, in section 6.5, we share our discoveries about the key sharing sets gathered from the real-world dataset, along with other discrepancies and observations observed while working with our association score models.

6.1 Analyzing chi-test and cut-off value

In this section, we discuss the chi-test and cut-off value to check the sensibility of our proposed model. We then present the results of all three proposed models discussed in section 5.2.

6.1.1 Chi-test analysis

We conducted a chi-test to examine the relationship between the scores in sharing sets and domain owners. The results indicate strong evidence, with a significance level of 1%, suggesting a dependence between the variable score and sharing set owners (see Table 8). This supports the efficacy of the domain ownership association model in identifying safe sharing sets. Additionally, the model proves valuable in detecting sharing sets where the same public key is used for multiple certificates. These findings affirm that our approach, which involves domain ownership analysis to identify safe and unsafe sharing, is a promising and suitable direction.

6.1.2 Cut-off value

We propose the need for a cut-off value[43] to safely associate a domain name with its original owner. Selecting this value randomly can lead to many valid ownerships being reported as invalid, or illegal domain name ownership being reported as valid by our models. Therefore, we calculate it based on our dataset. First, we find the probability score of all valid domain names and owner's associations using our proposed models. Then, we take the average of all the reported probabilities separately for each model. This average serves as our cut-off value, which is 52.39 for the N-Gram Model, 23.9 for the Skip-Gram Model, and 24.8 for the CBOW Model. Now, we present the results of our N-Gram, Skip-Gram, and CBOW models using the domain ownership association models.

6.2 Evaluation of domain ownership models for key sharing classification

In this section, we rigorously assess the performance of our domain ownership models within the context of key sharing

Table 5 Predicted owner of domains using N-Gram model

Domain name	Owner	Predicted owner	Predicted score
*.portal-prod-seasiap.azurewebsites.net	Microsoft Corporation	Microsoft Corporation	79.99
cloudfront.net	Amazon.com, Inc.	Microsoft Corporation	10.02
www.hellobayer.ca	Bayer AG	Bayer AG	67.01
*.c.docs.google.com	Google LLC	Google LLC	74.57
maven.repository.redhat.com	Red Hat	Red Hat	73.53

Table 6 Predicted owner of domains using Skip-Gram model

Domain name	Owner	Predicted owner	Predicted score
*.portal-prod-seasiap.azurewebsites.net	Microsoft Corporation	Microsoft Corporation	35.99
cloudfront.net	Amazon.com, Inc.	Microsoft Corporation	0.10
www.hellobayer.ca	Bayer AG	Bayer AG	49.50
*.c.docs.google.com	Google LLC	Google LLC	32.55
maven.repository.redhat.com	Red Hat	Red Hat	37.58

Table 7 Predicted owner of domains using CBOW model

Domain name	Owner	Predicted owner	Predicted score
*.portal-prod-seasiap.azurewebsites.net	Microsoft Corporation	Microsoft Corporation	30.95
cloudfront.net	Amazon.com, Inc.	Microsoft Corporation	0.52
www.hellobayer.ca	Bayer AG	Bayer AG	50.32
[0.5ex] *.c.docs.google.com	Google LLC	Google LLC	32.55
maven.repository.redhat.com	Red Hat	Red Hat	34.95

scenarios using a real-world test dataset prepared in accordance with step 11 of Fig. 2. A detailed analysis of the results is presented below:

6.2.1 N-gram model results on the test dataset

In this section, we delve into the results of our N-Gram model, which is augmented with the domain ownership association technique. Table 5 displays domain ownership predictions accompanied by their associated probabilities, referred to as the association score. The "Predicted Owner" column specifies the predicted owner's name, while the "Predicted Score" column quantifies the association score for the predicted owner concerning a particular Domain Name.

To confidently attribute a domain name to its owner, the association score must meet or exceed a predetermined threshold value. For instance, the model predicts Microsoft Corporation as the owner of "cloudfront.net," with an association score of 10.02. Since this score falls below the threshold of 52.39, the model does not assign ownership. In contrast, scores above the threshold enable confident ownership attribution. This mechanism ensures robust and reliable classification.

Table 8 Chi Square Test Result

Score	Same owners	Different owners	Total
High	4442	88	4530
Low	2951	736	3687
Mix	760	268	1028
Total	8153	1092	**9245**

The chi-square statistic is 859.3296. The result is significant at $p < 0.01$

6.2.2 Skip-gram model results on the test dataset

Table 6 presents insights into the performance of the Skip-Gram model. We set the threshold value at 23.9, derived from the model's cut-off calculations in section 6.1. The results reveal that the Skip-Gram model underperforms compared to the N-Gram model. Even valid domain associations yield scores below the threshold for reliable classification. This is attributed to the model's reliance on limited context, which is less effective when domain names contain few words.

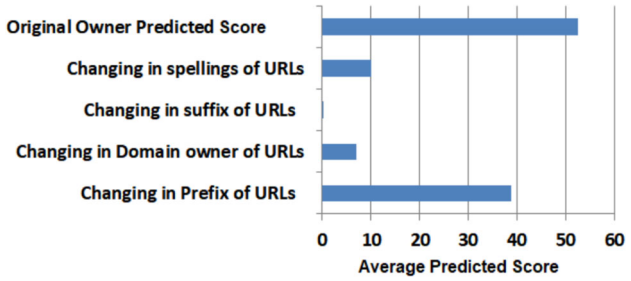

Fig. 3 Average association score using N-Gram Model

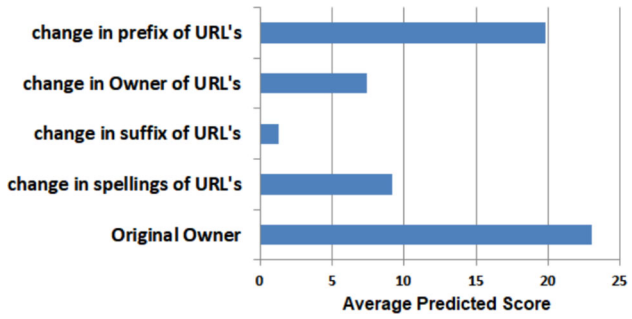

Fig. 4 Average association score using Skip-Gram model

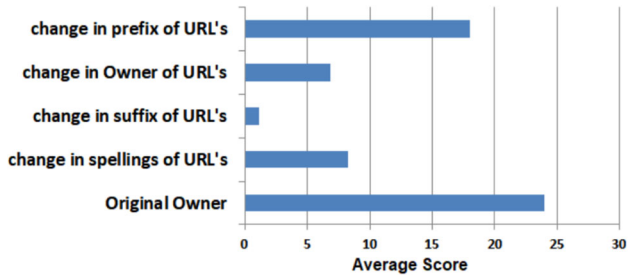

Fig. 5 Average predicted score using CBOW model

6.2.3 CBOW model results on the test dataset

This section presents the results of the CBOW model, integrated with the domain ownership association technique. Table 7 outlines domain ownership predictions alongside their scores. The CBOW model's threshold value is set at 24.8, as detailed in section 6.1. The CBOW model exhibits noticeable limitations. Even valid domain associations struggle to surpass the N-Gram model's threshold of 52.39. This is due to the CBOW model's focus on predicting target words using surrounding context, which becomes challenging with domain names containing few contextual words.

6.3 Evaluation of domain ownership models using synthetic dataset

Within this section, we scrutinize the robustness of our proposed models using a synthetic dataset generated in section 5.2.3. A detailed analysis of the results is provided below:

6.3.1 N-gram model results on the synthetic dataset

Table 9 offers insights into the average predicted association scores for various use cases using the N-Gram model. We assess the robustness of our N-Gram model against a synthetic test dataset, with the association score displaying a gradual decrease across the various scenarios. Notably, the attack involving the modification of the prefix part of Domain Names yields a score close to 40, while all other scenarios exhibit scores below 10. This consistent trend highlights the robustness of our N-Gram model, as none of the synthetic data manages to surpass our established threshold value.

Figure 3 graphically illustrates the comparison of average association scores against valid and invalid use cases using the N-Gram Model.

6.3.2 Skip-gram model results on the synthetic dataset

In this section, we evaluate the robustness of our proposed models using the Skip-Gram model with the synthetic dataset. The average scores obtained with the Skip-Gram model are reported in Table 10. Notably, the average score is 23.9, approximately half of the N-Gram model's threshold value (52.39). The performance of the Skip-Gram model significantly diminishes when confronted with synthetic data. Of particular interest is the scenario involving the modification of the prefix part of Domain Names, which yields a score approaching 18.67, while all other scenarios result in scores below 10.

Figure 4 provides a graphical representation of the overall association scores achieved by the Skip-Gram model against various synthetic data scenarios.

6.3.3 CBOW model results on the synthetic dataset

Within this section, we assess the robustness of our proposed models using the CBOW model in conjunction with the synthetic dataset. Table 11 presents the average scores obtained for each use case. Notably, the average score reported by the CBOW model stands at 24.8, considerably lower than the threshold value set by the N-Gram model (52.39). The CBOW model exhibits a noticeable decrease in performance when subjected to the synthetic dataset, with all scenarios, except the one involving the modification of the prefix part of Domain Names, resulting in scores below 10.

Figure 5 offers a visual representation of the overall association scores obtained with the CBOW model when subjected to the synthetic dataset.

Table 9 Average association scores against synthetic data using the N-Gram Model

Use cases	Average score
Changing the Prefix of Domain Names	38.68
Changing the Domain Owner of Domain Names	7.02
Changing the Suffix of Domain Names	0.05
Changing the Spellings of Domain Names	10.1

Table 10 Average association scores against synthetic data using the Skip-Gram model

Use cases	Average score
Changing the Prefix of Domain Names	18.67
Changing the Domain Owner of Domain Names	7.24
Changing the Suffix of Domain Names	0.21
Changing the Spellings of Domain Names	9.05

Table 11 Average scores against synthetic data using the CBOW model

Use cases	Average score
Changing the Prefix of Domain Names	19.82
Changing the Domain Owner of Domain Names	8.26
Changing the Suffix of Domain Names	0.30
Changing the Spellings of Domain Names	8.05

6.4 Comparison of skip-gram, N-gram and CBOW models

We compared the performance of the N-Gram, Skip-Gram, and CBOW models. Both Skip-Gram and CBOW models rely on vector embeddings and cosine similarity to predict domain ownership. However, their performance is limited due to the small word context in domain names. The N-Gram (bigram) model [10] consistently outperformed both, achieving higher accuracy in predicting domain ownership for real and synthetic datasets.

Table 12 displays the average association scores for all three models on the test data. Our synthetic data analysis further confirms the consistent superior performance of the N-Gram model over both the Skip-Gram and CBOW models, as evident in Fig. 3, Fig. 4, and Fig. 5. Specifically, the N-Gram model outperforms both Skip-Gram and CBOW models in predicting ownership, not only for real certificates but also for synthetic certificates. Figure 6 present the comparison of the average association scores of different use cases across all the models for both test and synthetic data. We can see that N-Gram results for all cases are better than others except the *Chnage-Owner Test Data* case. Figure 7 shows the average model association scores among different use cases for both synthetic and test data. We can see for each case N-Gram results are on the top. These results reinforce the practical applicability of the N-Gram model in real-world PKI scenarios.

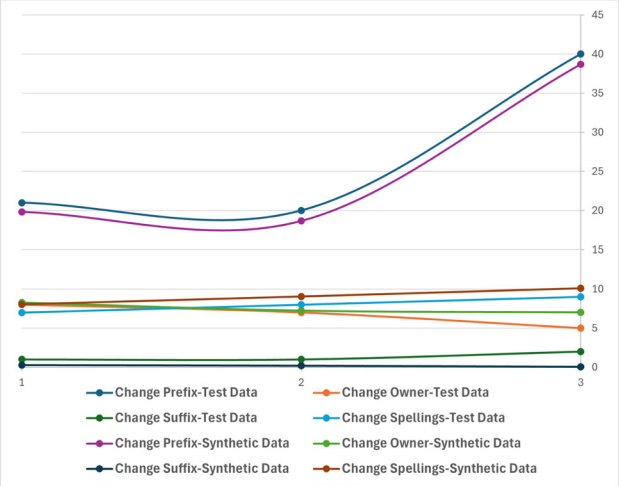

Fig. 6 Comparison of Average Scores of Different Use Cases across CBOW (left), SKip-Gram (center) and N-Gram (right) Models for both Test Data and Synthetic Data

Table 12 Association score among all the 3 models

Model	Average association score
Skip-Gram	23.9
CBOW	24.8
N-Gram	52.39

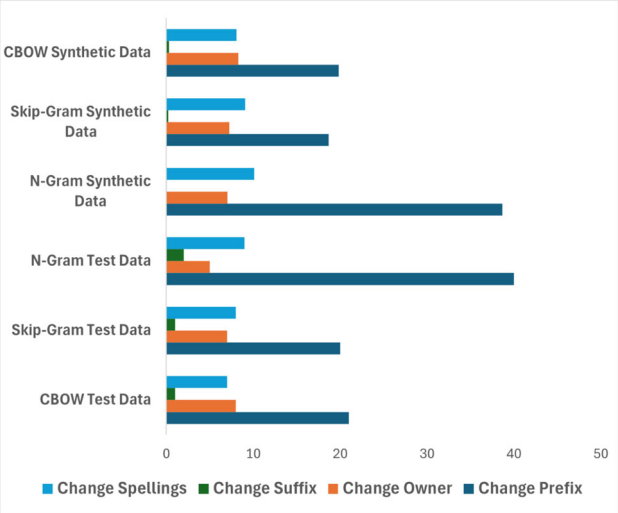

Fig. 7 Comparison of Average Model's Scores across Different Use Cases of Synthetic Data and Test Data

6.5 Classification of sharing sets from real world data using domain ownership model

In this section, we conduct a thorough analysis of the 9245 sharing sets that were extracted in Sect. 5.2. Leveraging the superior performance of the N-Gram model compared to other models, we apply it to these sharing sets to establish association scores between domain names and their respective owners. We categorize scores as *high* if they surpass the predefined cutoff value for the N-Gram model, and as *low* if they fall below this threshold.

Our analysis of each sharing set encompasses various features, including the public key, certificate hash, expiry date, issue date, issuer, and the association score predicted by the model. Based on these features, we aim to predict the correct owners of domain names within each sharing set to determine their safety.

Among the 9245 sharing sets, we identify two predominant categories:

1. Sharing Sets with Uniform Ownership:
   Within this category, there are 8,153 sharing sets where the same key is shared among different certificates, and all certificates within these sets have the same owner. This scenario typically suggests a *safe* sharing set, as it involves certificates owned by a single entity.
2. Sharing Sets with Diverse Ownership:
   In contrast, there are 1,092 sharing sets where the same key is shared among multiple certificates, and these certificates have different owners. This situation raises concerns about potential security risks associated with shared public keys and violates standard practices within the web Public Key Infrastructure (PKI).

We further classify these sharing sets into six distinct types, each with its own characteristics, as illustrated in Table 13 and elaborated upon in the subsequent discussion.

- **Same Owner-High Score (4,442 Instances):**
  In these sharing sets, all certificates share the same owner, and our model assigns high scores to them. This pattern indicates a *safe* sharing set, where certificates are owned by a single entity.
- **Same Owner-Low Score (2,951 Instances):** Although our model assigns low scores in 2,951 cases (32% of the total sharing sets), a detailed manual analysis confirms the ownership consistency. So we classify these sharing sets as *safe*. This may be due to the following reasons:
  - Our training data have limited representation of certain organizations and domain structures.
  - Domains with minimal or generic naming conventions (e.g., acronyms or single-word domains) provide limited contextual information.
  - Multi-owner SAN fields reduces model confidence.

  Manual analysis include verifying organization names, SAN entry consistency, and website visits, e.g., in cases like `example-corp.com` and `example-group.com`, both domains are confirmed to belong to the same parent organization.
- *Same Owner-Mix Score (760 Instances):*
  Within these sharing sets, our model assigns high scores to certain domain names while predicting low scores for others. Despite this mix of scores, these sharing sets are categorized as *safe* because all certificates share the same owner.
- *Different Owners-High Score (88 Instances):*
  In this scenario, our model predicts high scores for certificates with *different owners*. Although this could initially raise concerns, the high scores suggest the possibility that these different owners may, in fact, belong to the same entity. Thus, we classify these sharing sets as *safe*.
- *Different Owners-Low Score (736 Instances):*
  These sharing sets consist of certificates with *different owners*, and our model predicts low scores. Such a situation raises significant concerns as it deviates from best practices within web PKI. Consequently, these sharing sets are categorized as *potentially unsafe*.
- *Different Owners-Mix Score (268 Instances):*
  In these sharing sets, our model predicts a mix of high and low scores for different domains. This complexity is concerning and necessitates further analysis to identify potential vulnerabilities. Due to the likelihood of non-compliance with web PKI rules and the predominance of

Table 13 Sharing Sets Categorization from Real World Data with N-Gram Model Score

Same key different certificates				
Owners	Score	Sharing sets count	Percent	Safe/ unsafe
Same	High	4442	48%	Safe
Same	Low	2951	32%	Safe
Same	Mix	760	8%	Safe
Different	High	88	1%	Safe
Different	Low	736	8%	Unsafe
Different	Mix	268	3%	Unsafe
Total		9245	100%	-

low scores in most cases, we classify these sharing sets as *potentially unsafe*.

6.6 Manual analysis of predicted unsafe sharing sets

In this section, we thoroughly examined each sharing set categorized as *Different Owners-Mix Score* and *Different Owners-Low Score*, which our Domain Ownership model identified as potentially unsafe. We manually analyzed domain owners' names with respect to domains and each other. Our findings suggest that the sets categorized as unsafe by our model may be vulnerable to context confusion[14][48], takeover[47], related-domain[45], and domain impersonation embedding attacks[40]. Below are our observations:

- Approximately 10% of the sets that use CDNs may be vulnerable to potential attacks[14], categorizing them as unsafe.
- Around 40% of the sets using wildcards fall into the unsafe category due to potential confusion attacks[14], takeover attacks[47], and subdomain attacks[45].
- Our work reduces the number of sites that can be checked for phishing using **PhishTank**[4] and **isitphishing**[5] websites. There is a possibility that any of the domains from these sets could be phishing sites if monitored over a longer period
- Some sets show a complete change in domain owner names, e.g., from *Synchronoss Technologies, Inc* to *Vonage America Inc.*. We attempted to contact these organizations via an online form[36] to inquire about the reason for sharing keys with different owner names, but many did not respond.
- Our model effectively detects attacked or modified domains with fake owner names by accurately identifying real-world live attack domains and their owners as fake ownership, providing very low scores, sometimes even below 0.01.

7 Discussion

In section 7.1, we present our findings and recommendations, which we shared with the corresponding CAs that expressed interest in our work. Section 7.2 discusses the implications of these findings for PKI security, highlight community awareness about key-sharing issues and propose actionable steps for Certificate Authorities (CAs) to mitigate unsafe practices. Finally, we reflect on the limitations and future work of our study in Sect. 7.3 and in Sect. 7.4 respectively.

7.1 Actionable insights- violation of best practices

This section highlights critical gaps in public key infrastructure (PKI) practices and provides actionable insights.

We examined 1,004 real-world unsafe sharing sets from a total of 9245 sharing sets predicted by our model, as discussed in Sect. 6.5. These 1,004 sets involve 155 unique CAs engaged in unsafe key sharing. We contact each CA individually and ask the following questions:

- Please identify whether such key sharing among certificates is intentional or in the knowledge of CAs?
- Can it not lead to spoofing or MITM attack?
- Is it not a worrisome factor that the same public key is being used by multiple owners in different certificates?
- What if the owners do not have the same single/legal entity?
- How do you verify that different domains with different owners' names belong to the same entity?
- Should CAs not check the keys reuse/shared history before issuing new certificates to avoid malicious activities?

Some CAs were unaware of the reasons for the key sharing and suggested contacting domain owners for clarification, while some CAs acknowledged the potential risks. Based on these responses, we identified six major findings:

Finding 1 (Single Issuer Different Owners- Time Overlapped): Some sharing sets show multiple certificates issued by a single CA to different owners with the same public keys before any certificate expires. This indicates ongoing unexpired key sharing.

Finding 2 (Different Issuers Different Owners- Time Overlapped):

[4] https://phishtank.org/

[5] https://isitphishing.org/

Other sets reveal that different issuers have issued multiple certificates to different owners using the same public key before any certificate's expiry. We inform CAs that our model classifies this behavior as unsafe key sharing. CAs acknowledge the potential risk of key spoofing and MITM attacks in such cases.

Finding 3 (Challenges with CDNs):

Some CDNs experience common public key reuse, making it difficult to identify authorized key sharing. CAs lack the means to distinguish between malicious and frowned-upon key sharing, highlighting the need for better mechanisms in the PKI system. Our model can help identify unsafe key sharing in such scenarios.

Finding 4 (Organizations are changing names): Certain sharing sets show multiple owners' names associated with the same domain name, indicating organizations may change names while using the same key. This poses risks, as adversaries can request certificates with others' public keys without the CA's knowledge.

Finding 5 (No Mechanism to identify original owner): Different domains with different owners' names sharing the same public key with one or more issuers raise concerns. CAs lack a mechanism to verify whether these owners belong to the same entity.

Finding 6 (Failure to Follow Key Rotation Practices): CAs recommend end-users rotate keys with each new certificate and avoid key sharing. However, many users do not comply, especially in CDN cases where shared keys are common.

7.2 Key sharing and community awareness

As the number of public-key sharing increases with the number of certificates, questions arise: Is the community aware of and addressing these issues? Is the community taking any steps to monitor CA's issuance behavior?

To the best of our knowledge, there has been no systematic effort to address public-key reuse at scale or its associated risks, beyond basic checks for compromised keys.

We recommend CAs to check key reuse/shared history before issuing new certificates to prevent possible malicious activities. Public key reuse should be discouraged to enhance PKI security. This includes mandatory key rollovers for every new certificate. Some CAs have shown interest in these recommendations, reflecting a positive step towards improving key-sharing practices.

These findings demonstrate how our approach effectively handles unsafe key sharing among certificates, reducing investigation effort from 9245 to 1,004 sharing sets (around 90% reduction) and enabling faster follow-up and mitigation with CAs.

7.3 Limitations

While our methodology demonstrates substantial potential, it has some limitations described below. Current association models focus on static features (e.g., domain names, SAN entries), which may fail to capture ambiguous or evolving ownership relationships. Our datasets, such as Alexa's Top 1 M and Rapid7, are biased towards high-visibility domains, potentially excluding smaller or emerging domains that might exhibit unsafe sharing. The methodology does not incorporate dynamic data sources such as WHOIS updates or Certificate Transparency logs, which can improve accuracy. Processing large datasets with compliance checks (e.g., zlint) requires significant storage and computational resources, which may limit scalability for smaller organizations.

7.4 Future directions

To address these limitations, we propose research directions as follows. We plan to extend our work by incorporating a larger dataset of around 100 million certificates. Additionally, we aim to improve the technique for validating whether all owners in unsafe sharing sets belong to the same legal entity by leveraging contextual information from other sources like DNS/WHOIS records. We also intend to enhance our model by considering features such as root/intermediate certificates, issuer names, expiration dates, and issuance dates.

We also plan on enhancing domain ownership classification by exploring advanced techniques, such as Graph Neural Networks, to analyze complex relationships in key-sharing scenarios and improve classification accuracy.

8 Conclusion

In this work, we introduced a concept of domain ownership score using an association model to analyze the ownership of certificates and distinguish between safe and unsafe sharing of public keys among multiple certificates. By examining almost 12 million certificates, we identified 9245 sharing sets where the same public key is shared in multiple certificates. Out of these, we classified 1,004 sharing sets as unsafe based on our model.

Our domain ownership model provides practical value by enabling Certificate Authorities (CAs) and organizations to identify unsafe key-sharing practices efficiently. By reducing investigation workloads by approximately 90%, this model helps target interventions that strengthen PKI security and mitigate risks such as spoofing and MITM attacks.

We will release the dataset of the unsafe 1,004 certificate sharing sets to enable further action with CAs[27]. This will

help in addressing the challenges posed by related-domain attackers and improving the security of the web PKI system.

Author Contributions Kashif Junaid performed the experiments and testing. Dr Muhammad Umar continuously engaged with Kashif in verifying the writing, results and the thought process of the novel technique described in the paper.

Data Availability We are committed to transparency and open science. The Rapid7 certificate data that supports the findings of this study is available with us and can be accessed by approaching us. Any additional data required to reproduce the results or test the methods reported in this article is available upon request. The authors encourage readers to contact phdcs21004@itu.edu.pk for further information regarding data availability and access.

Declarations

Conflict of interest: We declare that we have no Conflict of interest that could influence the interpretation or evaluation of the results presented in this manuscript. Conflict of interest include, but are not limited to, financial, personal, or professional relationships that may have influenced the work or could be perceived to have influenced the work.

Ethical Considerations In our research, we address the following ethical considerations: Firstly, we employ manual methods to retrieve certificate data from Alexa and Rapid7's Sonar websites. Secondly, when communicating with Certificate Authorities (CAs), we ensure that the names of any other CAs involved in key sharing are not disclosed. Thirdly, we refrain from attributing blame to any CA for participating in key-sharing activities intentionally. Instead, we focus on discussing user adherence to best practices when obtaining certificates. Throughout our investigation, we place significant emphasis on safeguarding the privacy of Certificate Authorities (CAs). Our analysis strictly avoids accessing any single entity's private key, and no information pertaining to entities containing user account details is disclosed.

References

1. Adrian, D., Bhargavan, K., Durumeric, Z et al.: Drown: Breaking tls using sslv2. In: Proceedings of the USENIX Security Symposium, pp. 689–706, (2016)
2. Alexa top 1 million websites. Accessed March 1, (2022). URL: https://www.alexa.com/topsites
3. Jim A-F: Multi-protocol attacks and the public key infrastructure. NIST, (1998)
4. Basin, D., Cremers, C., Kim, TH-J., Perrig, A., Sasse, R., Szalachowski, P: Arpki: Attack resilient public-key infrastructure. In: Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, pp. 382–393, (2014)
5. Brinkmann, M., Dresen, C., Merget, R., Poddebniak, D., Müller, J., Somorovsky, J., Schwenk, J., Schinzel, S : ALPACA: Application layer protocol confusion - analyzing and mitigating cracks in TLS authentication. In 30th USENIX Security Symposium (USENIX Security 21), pp. 4293–4310. USENIX Association, (2021)
6. Cangialosi, F., Chung, T., Choffnes, D., Levin, D., Maggs, B., Mislove, A., Wilson, C.: Measurement and analysis of private key sharing in the HTTPS ecosystem. In: ACM SIGSAC Conf. on Computer and Communications Security, pp. 628–640, (2016)
7. Clark, J., van Oorschot, P.C.: Sok: Ssl and https: Revisiting past challenges and evaluating certificate trust model enhancements. In: 2013 IEEE Symposium on Security and Privacy, pp. 511–525, (2013)
8. cloudflare. How-does-ssl-work? 2022. URL: https://www.cloudflare.com/learning/ssl/how-does-ssl-work/
9. Conti, M., Dragoni, N., Lesyk, V.: A survey of man in the middle attacks. IEEE Commun. Surv. Tutor. **18**(3), 2027–2051 (2016)
10. Dagan, I., Lee, L., Pereira, F.C.N.: Similarity-based models of word cooccurrence probabilities. Mach. Learn. **34**(1–3), 43–69 (1999)
11. Whta is typosquatting? how misspelling that domain name can cost you? NortonLifeLock Inc., (Oct 2020). Accessed December 17, (2024). URL: https://us.norton.com/internetsecurity-online-scams-what-is-typosquatting.html
12. Carnavalet de, X., Mannan, M: Killed by proxy: Analyzing client-end tls interception software. In: Proceedings of the Network and Distributed System Security Symposium (NDSS), 01 (2016)
13. Delignat-Lavaud, A., Abadi, M., Birrell, A., Mironov, I., Wobber, T., Yinglian, X.: Closing the gap between guidelines and practices. In NDSS, Web PKI (2014)
14. Delignat-Lavaud,A., Bhargavan, K: Network-based origin confusion attacks against https virtual hosting. In: Proceedings of the 24th International Conference on World Wide Web, WWW '15, pp. 227–237, Republic and Canton of Geneva, CHE, (2015). International World Wide Web Conferences Steering Committee
15. DigiNotar. Accessed 08 December (2024). URL: https://en.wikipedia.org/wiki/DigiNotar
16. Dong, Z., Kane, K., Camp, L. J: Detection of Rogue Certificates from Trusted Certificate Authorities Using Deep Neural Networks. ACM Transactions on Privacy and Security (TOPS), 19, (2016)
17. Durumeric, Z., Kasten, J., Bailey, M., Halderman, J A: Analysis of the https certificate ecosystem. In: Proceedings of the 2013 conference on Internet measurement conference, pp. 291–304, (2013)
18. Farhan, S., Chung, T: Exploring the Evolution of TLS Certificates, pp. 71–84. 03 (2023)
19. Felsch, D., Grothe, M., Schwenk, J., Czubak, A., Szymanek, M: The dangers of key reuse: Practical attacks on IPsec IKE. In 27th USENIX Security Symposium (USENIX Security 18), pp. 567–583, Baltimore, MD, (2018). USENIX Association
20. Giesen, F., Kohlar, F., Stebila, D: On the security of tls renegotiation. In: Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security, pp. 387–398, (2013)
21. Guthrie, D., Allison, B., Liu, W., Guthrie, L., Wilks, Y: A closer look at skip-gram modelling. In LREC, (2006)
22. Han, S-W., Kwon, H., Hahn, C., Koo, D., Hur, J: A survey on mitm and its countermeasures in the tls handshake protocol. In: 2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN), pp. 724–729. IEEE, (2016)
23. Heartbleed. Accessed 08 December 2024. URL: https://en.wikipedia.org/wiki/Heartbleed
24. Heninger, N., Durumeric, Z., Wustrow, E., Halderman, J. A: Mining your ps and qs: Detection of widespread weak keys in network devices. In 21st USENIX Security Symposium (USENIX Security 12), pp 205–220, Bellevue, WA, August 2012. USENIX Association
25. Holz, R., Braun, L., Kammenhuber, N., Carle, G: The ssl landscape: A thorough analysis of the x.509 pki using active and passive measurements. In: Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference, IMC '11, pp. 427–444, New York, NY, USA, (2011). Association for Computing Machinery
26. Hue, M H., Debnath, J., Leung, K M., Li, L., Minaei, M., Mazhar, M. H., Xian, K., Hoque, E., Chowdhury, O., Chau, S Y: All your credentials are belong to us: On insecure wpa2-enterprise configurations. In Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security, CCS '21, pp. 1100–1117, New York, NY, USA, (2021). Association for Computing Machinery
27. Junaid, K: Sharingc sets on github. URL: https://github.com/kashif-junaid/LowScoreSharingSets/

28. Khan, S., Luo, F., Zhang, Z., Ullah, F., Amin, F., Qadri, S.F., Belal, HMd., Bin, R., Rukhsana, W., Lu, U., Shamsher, L., Meng, L., Victor, C.M., Wu, K.: A survey on x.509 public-key infrastructure, certificate revocation, and their modern implementation on blockchain and ledger technologies. IEEE Commun. Surv. Tutor. **25**, 2529–2568 (2023)
29. Khan, T., Huo, X., Li, Z.: Kanich, C: Every second counts: Quantifying the negative externalities of cybercrime via typosquatting. IEEE Symp. Secur. Priv. **135–150**(07), 2015 (2015)
30. Kilgallin, J., Vasko, R: Factoring rsa keys in the iot era. In 2019 First IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA), pp. 184–189. IEEE, (2019)
31. Kumar, D., Wang, Z., Hyder, M., Dickinson, J., Beck, G., Adrian, D., Mason, J., Durumeric, Z., Halderman, J A., Bailey, M: Tracking certificate misissuance in the wild. In 2018 IEEE Symposium on Security and Privacy (SP), pp. 785–798. IEEE, (2018)
32. Li, J., Zhang, Z., Guo, C: Machine learning-based malicious x.509 certificates–detection. Appl. Sci., 11(5), (2021)
33. Liu, B.: Text sentiment analysis based on cbow model and deep learning in big data environment. J. Ambient Intell. Humanized Comput. **11**(2), 451–458 (2020)
34. Ma, Z., Faulkenberry, A., Papastergiou, T., Durumeric, Z., Bailey, M D., Keromytis, A D., Monrose, F., Antonakakis, M: Stale tls certificates: Investigating precarious third-party access to valid tls keys. In: Proceedings of the 2023 ACM on Internet Measurement Conference, (2023)
35. Larimer, J., Root, K: 2012. Security and Privacy in Android Apps [Online] Available: https://developergoogle.com/events/io/2012/sessions/gooio2012/107/, accessed (Oct. 2023). Accessed December 17, 2024
36. Google form. Accessed March 3, (2024). URL: https://www.wired.com/2011/09/doppelganger-domains/
37. Pakniat, N: Public key encryption with keyword search and keyword guessing attack. ISCISC, (2016)
38. Parsovs, A: Estonian electronic identity card: Security flaws in key management. In 29th USENIX Security Symposium (USENIX Security 20), pp. 1785–1802. USENIX Association, (August 2020)
39. Rapid7 data. Accessed December 17, (2024). URL: https://opendata.rapid7.com/sonar.ssl
40. Roberts, R., Goldschlag, Y., Walter, R., Chung, T., Mislove, A., Levin, D: You are who you appear to be: A longitudinal study of domain impersonation in tls certificates. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS '19, pp. 2489–2504, New York, NY, USA, (2019). Association for Computing Machinery
41. ROCA vulnerability. Accessed 08 (December 2024). URL: https://en.wikipedia.org/wiki/ROCA_vulnerability
42. Sakurai, Y., Watanabe, T., Okuda, T., Akiyama, M.: Mori, T: Identifying the phishing websites using the patterns of tls certificates. J. Cyber Secur. Mob. **10**(2), 451–486 (2021)
43. Cutoff point. Accessed on July 6, 2024. URL: https://www.sciencedirect.com/topics/mathematics/cutoff-point
44. Shodan boss finds 250,000 routers have common keys. Accessed December 17, (2024). URL:https://www.theregister.com/2015/02/20/250000_routers_have_duplicate_ssh_keys/
45. Squarcina, M., Tempesta, M., Veronese, L., Calzavara, S., Maffei, M: Can i take your subdomain? exploring Same-Site attacks in the modern web. In 30th USENIX Security Symposium (USENIX Security 21), pp. 2917–2934. USENIX Association, (2021)
46. Stark, E., Sleevi, R., Muminović, R., O'Brien, D., Messeri, E., Felt, A P., McMillion, B., Tabriz, P: Does certificate transparency break the web? measuring adoption and error rate. (2019)
47. Zhang, M., Li, X., Liu, B., Lu, J., Zhang, Y., Chen, J., Duan, H., Hao, S., Zheng, X: Detecting and measuring security risks of hosting-based dangling domains. Proc. ACM Meas. Anal. Comput. Syst., 7(1), (2023)
48. Zhang, M., Zheng, X., Shen, K., Kong, Z., Lu, C., Wang, Y., Duan, H., Hao, S., Liu, B., Yang, M: Talking with familiar strangers: An empirical study on https context confusion attacks. In: Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security, CCS '20, pp. 1939–1952, New York, NY, USA, (2020). Association for Computing Machinery