INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2023, VOL. 69, NO. 4, PP. 803–810
Manuscript received August 31, 2023; revised October, 2023. DOI: 10.24425/ijet.2023.147704

RSA Keys Quality in a Real-world Organizational Certificate Dataset: a Practical Outlook

Konrad Kamiński and Wojciech Mazurczyk

***Abstract*—This research investigates the intricacies of X.509 certificates within a comprehensive corporate infrastructure. Spanning over two decades, the examined enterprise has heavily depended on its internal certificate authority and Public Key Infrastructure (PKI) to uphold its data and systems security. With the broad application of these certificates, from personal identification on smart cards to device and workstation authentication via Trusted Platform Modules (TPM), our study seeks to address a pertinent question on how prevalent are weak RSA keys within such a vast internal certificate repository. Previous research focused primarily on key sets publicly accessible from TLS and SSH servers or PGP key repositories. On the contrary, our investigation provides insights into the private domain of an enterprise, introducing new dimensions to this problem. Among our considerations are the trustworthiness of hardware and software solutions in generating keys and the consequential implications of identified vulnerabilities on organizational risk management. The obtained results can contribute to enhancing security strategies in enterprises.**

***Keywords*—Certificates; X.509; RSA keys; PKI; vulnerabilities; RSA factorization**

I. Introduction

X.509 certificates have been extensively used presently, meeting numerous ICT (Information and Communication Technologies) security requirements. Depending on usage, these certificates offer some of the following features:

- Confidentiality: through encrypted TLS (Transport Layer Security) protocol and email encryption (S/MIME);
- Integrity: including non-repudiation and authenticity, by linking the name in the certificate with operations cryptographically related to the certificate holder;
- Accountability: can be established through the use of two-way identity-proofing mechanisms as well as digital signatures.

It is worth noting that nearly all major websites employ HTTPS (Hypertext Transfer Protocol Secure), and web browsers have been alerting users for quite some time now about the absence of encryption when using the unencrypted protocol. In addition to implementing HTTPS, the extensive adoption of X.509 standard certificates has also ensured that tools that use certificates and methods to protect private keys are widely available. The use of smart cards is a common way to implement strong authentication methods or MFA (Multi-Factor Authentication), particularly in enterprise environments. By generating a private key on a smart card, you can be sure that there is only one instance of the key and that only a person who knows the correct PIN can use the private key. Certificates are frequently used in enterprises to provide secure access to internal resources and can also effectively replace the need for passwords. On the other hand, services that require enhanced security can use Hardware Security Modules (HSM) to protect the most sensitive keys.

K. Kamiński is with Faculty of Electrical Engineering and Communication, Warsaw University of Technology, Warsaw, Poland and Security Technology Development and Transformation Division, Orange Polska S.A., Warsaw, Poland (ORCID: 0009-0004-0031-7507).

W. Mazurczyk is with Faculty of Electrical Engineering and Communication, Warsaw University of Technology, Warsaw, Poland (ORCID: 0000-0002-8509-4127).

The quality of cryptographic keys in certificates used for an organization's needs affects the security of its resources, i.e., data and infrastructure. Certificates can authenticate not only within the corporate network but also at the Internet perimeter. It is common to use certificates for authentication in edge services - VPN (Virtual Private Network), WiFi, and sites accessible from the Internet using mTLS (Mutual TLS). A weakness in the key used in a personal certificate (such as the ability to factor a public key) gives an attacker the ability to impersonate a user and gain direct access to corporate resources from anywhere in the world. Detecting weak keys in publicly accessible corporate website certificates can result in Man-in-the-Middle (MitM) attacks, wherein the data sent to sites authenticated with a corporate certificate can be intercepted and credentials stolen or sessions manipulated. The way an attacker exploits the vulnerability varies depending on the type of certificate, but it can lead to data leakage and intercepting access to an organization's internal infrastructure.

This article details research on the use of certificates by a large enterprise. For over two decades, the company has relied on PKI and X.509 certificates from an internal Certification Authority to enhance data and system security and ensure compliance with various regulations. Certificates are used for a variety of purposes within the company, but three main groups of uses can be distinguished: personal certificates issued on smart cards, certificates for servers and devices (including mobile devices) where keys are mostly generated in software, and certificates for workstations using the Trusted Platform Module (TPM). The objective of the study was to identify how much of relatively large RSA key collection contains weak keys. Similar studies have been conducted previously by other researchers, but they used keys from certificates of publicly available TLS and HTTPS services [1] and SSH (Secure Shell) servers [2] and PGP (Pretty Good Privacy) keys [3].

As stated previously, most research uses publicly available key sets. The existence of weak keys presents certain vulnerabilities and, consequently, the chance of successful attacks. In this paper, a dataset of keys used for internal corporate purposes is studied, which complements the previous research because:

- Some keys come from personal and workstation certificates, where the keys are generated on smart cards, in TPMs, or in a CA (Certification Authority) environment. On the basis of the research results, the level of trust in the hardware and software solutions used can be determined.
- As a result of the research and analysis of the results, it will be possible to develop and implement countermeasures in the enterprise to minimize the negative impact of the vulnerabilities found. The management of enterprise resources, such as certificates and keys, enables the identification of the teams responsible for specific systems and the monitoring of work progress to mitigate the negative impact of vulnerabilities.
- The research methods developed will make it possible to assess the effectiveness of measures to reduce or eliminate the generation of certificates with vulnerable keys in the future. In the context of internally generated keys and certificates, establishing an efficient process for enhancing key quality can be achieved in a relatively short timeframe through internal standards, recommendations, and collaboration with vendors.

The remainder of this paper is structured as follows. First, in Section II, we present the fundamentals of RSA keys, certificates, and public key infrastructure (PKI) and briefly introduce the RSA key vulnerabilities that are the main focus of this work. Then, in Section III, related work is presented. Next, Section IV explains the research methodology utilized, that is, the procedures for acquiring the set of certificates, how they were obtained and how the tests were performed. Section V discusses the results of the conducted investigation, including an attempt to determine the root cause of each type of vulnerability present and suggestions for minimizing their impact. Finally, Section VII provides a summary of the findings, proposed solutions, and suggestions for further research.

II. Background

Asymmetric cryptography is a fundamental part of the Public Key Infrastructure (PKI), enabling secure data exchange without the need to establish a separate channel for exchanging cryptographic keys. The RSA algorithm, named after its creators (Rivest, Shamir, and Adleman), is based on the computational complexity required to factor large complex numbers, making it an efficient tool for encryption and digital signing. In basic terms, asymmetric cryptography uses two cryptographically related keys. One key is a private key, known only to its owner. The second key is a public key that is available to other parties. Deriving the private key from the public key must be a very complex and time-consuming operation, resulting in a lack of practical methods for performing this operation.

The use of public keys in communication requires confirmation of ownership of the keys. Therefore, the concept of PKI [4] was proposed in 1978 with the aim of cryptographically linking user data identifying the entity with its public key, verified by a Trusted Third Party (TTP). Digital certificates are an integral part of PKI. They are digital documents that cryptographically link the identity of an entity (owner, subject) to a public key. Independent entities that perform TTP functions, called Certification Authorities (CAs), issue certificates. Digitally signed by this CA, the certificates confirm the identity of the entity along with the public key associated with that identity.

However, despite their effectiveness, PKI systems are not without vulnerabilities [5], [6]. PKI relies on trust in certificate issuers and cryptography. In the past, there have been cases where TLS certificates were issued to unauthorized parties without sufficient verification of the applicant's DNS name credentials [7]. There have also been cases where Certificate Authorities have been compromised as a result of attackers gaining access [8]. To minimize the negative effects of these cases, a number of mechanisms have been proposed to mitigate the risk in the above cases, including machine learning [9]. Finally, the HTTP Public Key Pinning (HPKP) and Certificate Transparency mechanisms have been introduced [10], as well as a CAA record in the DNS [11].

As stated above, the quality of the key used is a crucial factor in securing certificates. The quality of an RSA key, which refers to its security capability, is vital to preserving the confidentiality and integrity of data. This article focuses on examining the quality of RSA keys in certificates issued by Orange Polska internal CA using known vulnerability tests such as RSA private key factor sharing [12], ROCA (Return of the Coppersmith's Attack) [13], and Debian weak keys [14], [15].

A. A brief description of the vulnerabilities targeted by the research

Over the past two decades, several vulnerabilities have been discovered in RSA keys. These vulnerabilities violate the rule with respect to the extremely high computational complexity required to determine a private key from a public key. These vulnerabilities completely or significantly weaken the strength of certificates containing such keys, rendering them untrustworthy. To verify the quality of a key, one must use existing tools or create new ones. Researchers who publish information about discovered vulnerabilities often create tools to test keys against the vulnerabilities described.

A division of key vulnerabilities can be introduced. Some vulnerabilities are individual weaknesses of single keys (ROCA, Debian weak keys). The other type of vulnerability is due to dependencies between keys (BatchGCD). The above vulnerabilities are briefly described below.

1) Debian weak keys: The Debian weak keys vulnerability (CVE-2008-0166) is caused by a programming error in the prime number generator and results in a significant reduction of the possible key space. It affects software-generated keys in the Debian Linux distribution (and derivatives, including the popular Ubuntu) with the vulnerable OpenSSL package. The vulnerability allows the cataloging a list of all RSA keys that can be generated on vulnerable Debian family systems. A database of vulnerable public keys is currently available. The test tool checks that the public key is included in this database.

2) Sharing parts of RSA keys : The vulnerability is the use of the same prime number in different RSA keys as one of the two components of the private key *p* and *q*. The ease of determining the private key in such a case is shown by a sample calculation in the SageMath package:

```
sage: p = random_prime(2^1024)
sage: q1 = random_prime(2^1024)
sage: q2 = random_prime(2^1024)
sage: K1=p*q1
sage: K2=p*q2
sage: time g = gcd(K1,K2)
CPU times: user 34 µs, sys: 0 ns, total: 34 µs
Wall time: 34.1 µs
sage: g == p
True
sage: d1=K1/p
sage: d1 == q1
True
```

Fig. 1. SageMath example showing how easy and quickly find shared primes.

As shown on Fig. 1, identifying a shared private component p is a trivial task. In addition, knowing it easily leads to knowing the other components of the key. Similar dependencies exist in the modulo arithmetic used in RSA. A fast algorithm that implements the above dependencies in large sets of RSA public keys is called BatchGCD [16].

3) ROCA: ROCA (CVE-2017-15361) is another vulnerability resulting from flaws in the RSA key generator in hardware cryptographic chips based on the RSALib library provided by Infineon Technologies. It allows the private key to be derived from the public key in practical attacks for RSA keys up to 2048 bits in length. Researchers have made available a tool to detect the presence of the vulnerability in RSA keys, covering a variety of formats (certificate files, SSH and PGP keys, and others).

III. Related Work

Research on the quality of cryptographic keys has been going on for many years. With the announcement of new vulnerabilities in asymmetric keys, new studies are announced that focus on different areas. Most studies are based on publicly available keys using existing certificate databases, Internet services scan results [1]–[3], [17], or Certificate Transparency logs. Another approach is to use data collected by browsers [18], which requires cooperation with the browser vendor.

Some research examines not only the quality of cryptographic keys, but also the correctness of certificates and their compliance with the current requirements of the CA/Browser Forum [19]. In the context of internal enterprise certificates, some of these requirements will be useless. Determining which of the hundreds of requirements makes sense (improve security) in the context of internal certificates may be the subject of further research.

This article addresses the problem of weak RSA keys that have been generated internally within an organization for more than 20 years as part of an internal PKI service. The certificate base is not very large (about 650,000 certificates) compared to the more than 80 million described in the literature [2]. The uniqueness of the studied dataset is the long history as well as the access to keys generated from more exotic environments. Often, these are keys that cannot be obtained by means described in other publications. Examples include personal or TLS client certificates, which cannot be found in publicly available databases.

As an example of the different nature of the studied base, it suffices to cite the example of finding 14 IBM RSA II card certificates. In a study based on a collection of more than 80 million certificates, which is more than 100 times greater, relatively few were found, only 51 [2]. The dissimilarity of the collection studied in this article is due to the availability of certificates and keys that cannot be obtained, for example, by scanning open connections on port 443/TCP (HTTPS).

Among the few examples of research based on certificates from collections that are not available to the public is the vulnerability study of Estonian e-documents [13]. However, it was based on samples rather than on the entire key collection. The above study also disaggregated the different sources of keys and determined the ease of access to them. In this article, most of the keys examined can be classified as not publicly available. The collection used in this article includes more than 100,000 keys generated in the TPM, or nearly 300,000 keys from personal certificates.

IV. Methodology

Key research in an organization consists of several stages. The first stage is gaining access to a set of certificates. The second stage is processing the data set into a form that allows further testing. Further stages involve developing methodologies and applying them to individual vulnerability tests in a repeatable and efficient manner. Each of these stages has required the development of specialized tools. The following sections describe the basic solutions used in the testing.

A. The Datasets

The study utilized two sets of certificates from internal certification authorities at Orange Polska, a major telecommunications operator in Poland and part of the Orange Group [20] that offers telecommunications services in Europe and Africa. The first authority is responsible for smart card and infrastructure certificate issuance. The Certification Authority has been in operation for over 20 years and issued both qualified and globally trusted certificates between the years 2002 and 2006. These certificates met the WebTrust$^{SM/TM}$ [21] requirements set forth by the CA/Browser Forum. Orange Poland eventually terminated its public certification service operations, but the center's technical infrastructure, procedures, and organization still largely meet the standards for public certification centers.

The dataset includes around 450,000 certificates that were issued between October 2002 and July 2023. It was exported from the CA software as a single binary-encoded certificate file. To facilitate further processing, all certificates were saved in individual files. Because of the magnitude of certificates, a simple Bash loop to divide the collection into individual certificates consumes significant time. Hence, a dedicated program was employed for this task. This method facilitates saving all certificates into separate files in one go while also extracting individual certificate metadata (subject, serial number, expiration date, and public key module) into a text file, thus streamlining future research. Implementing a dedicated program reduces the time spent saving certificates from 2 hours in a Bash loop to about 3 minutes. The mentioned script produced a directory with close to 0.5 million files. However, the EXT4 file system efficiently manages this volume, obviating the need for a multi-level directory structure. The resulting certificates can be tested later using specialized tools. Furthermore, the script creates a metadata file extracted from the certificates and RSA key modules, which are utilized in the subsequent tests.

Orange Polska has a secondary certification authority, based on the Active Directory Certification Service (ADCS), which has been operational since 2018. This authority has already provided about 200,000 certificates. It issues certificates based on native Active Directory (AD) mechanisms, specifically for workstations and the Mobile Device Management (MDM) system. The ADCS service console and other native AD tools lack an export function for all issued certificates. The sole method of exporting involves executing a database backup and extracting certificates from said backup. The number of tools to access the Extensible Storage Engine Data Base (ESEDB) utilized by the ADCS service is limited, although a Linux-supported toolkit called *libesedb-utils* is present for managing this particular database type. However, the authors acknowledge and demonstrate that the tool's usage is excessively slow, rendering it impractical for a 2 GB database export. As an alternative, a basic dumper tool was developed to locate all certificates in the ADCS database.

This tool identifies each certificate byte by byte, searching for the ASN.1 (Abstract Syntax Notation One) sequence with which the certificate begins. After locating each sequence, the size of the ASN.1 structure is retrieved. Based on the size ranges, a determination is made for further processing. An attempt is made to decode the data as an X.509 certificate, and if successful, a comparable action is taken for the first set's prepared tool. Although the method used is not optimal as it does not rely on the database file's structure, it still accomplishes the goal in a relatively short time (a few minutes). The algorithm's validity assumes the continuity of the certificate record in the examined file. An added benefit of this method is the possibility to search for certificates in any file, regardless of their structure. Furthermore, the accuracy of the tool was validated through tests on the first dataset.

B. Vulnerability Tests used

The Debian weak keys tool for checking vulnerable keys was available in the *openssl-blacklist* package in the Debian distribution. For the latest Debian releases, it is no longer included, but nothing prevents you from downloading and installing the deb file from the repository of an older version of the distribution. The package provides a database of vulnerable keys and an *openssl-vulnkey* script in Python. It was necessary to modify it to directly support certificates in DER (Distinguished Encoding Rules, binary encoding) format. In the examined collections of certificates, their number did not allow the command to be called directly, as the list of arguments turned out to be too long. Therefore, the collection was divided into 100 parts and the test was executed sequentially for each subset using two nested loops in the bash shell. The time to run the tests sequentially for each subset of the larger set exceeds 2 hours (on a computer with an Intel(R) Core(TM) i7-8665U CPU @ 1.90GHz). On a multiprocessor machine, it is easy to speed things up by parallelizing the tasks called in the loop. A simple script made it possible to achieve parallel processing quite effectively using the features of the Linux shell. After using parallel processing, the test execution time for a larger set is less than 30 minutes (using all 4 CPU cores with HT enabled or 8 threads). Of course, the gain would be greater if a machine with a CPU with more cores was used.

The BatchGCD algorithm was used to detect common factors in RSA key sets. The original BatchGCD algorithm [22] used the SageMath computing environment to detect shared factors in RSA key parts. Many researchers have optimized implementation of this algorithm to work efficiently on large key sets. In 2019, a C++ implementation [23] was created that runs faster and uses multithreading. The program requires an array of RSA modules, so it takes advantage of the previously described properties of the author's tools for exporting certificates from datasets. In addition, duplicate modules created by renewing certificates with the same RSA keys had to be removed, as they unnecessarily obscure the results. The tool, written in C++, is easy to use as it requires a CSV formatted input file containing the index and key module columns. As a result, the tool generates files containing indexes of vulnerable keys. The corresponding script generates the required CSV file, and after the tool runs, it searches for certificates based on the indexes and presents the results.

The tool *roca-detect* [24] developed by vulnerability discoverers was used to detect ROCA vulnerabilities. It allows working with different file formats, including a flat file with a key module on each line or a directory of certificate files. When working with a directory of files, if a vulnerable key is found, a JSON format message is generated with the full details of the certificate. The *roca-detect* tool is quite fast, but again it uses a multithreaded script. This reduces the processing time for a large set of certificates to a few minutes. As mentioned above, the results of the tool are saved to a file in JSON format, so the *jq* tool from the *jquery* package was used to present the results.

V. Results

As a result of the investigation, a number of vulnerable RSA keys were identified. The table displays the number of certificates by certificate type and the corresponding vulnerability found in the key.

TABLE I
DISTRIBUTION OF VULNERABLE CERTIFICATES BY TYPE OF APPLICATION AND VULNERABILITY

Certicate type	**Number of certificates**				
	Issued	***Debian***	***Shared***	***ROCA***	***% of vuln.***
Personal	283813	0	3	0	0.00%
TLS Servers	32844	8	14	2	0.07%
VPN Servers	477	0	0	0	0.00%
Old MDM	131242	0	0	0	0.00%
AD Controllers	373	0	0	0	0.00%
TLS Inspection	253	0	0	0	0.00%
MDM	49590	0	0	0	0.00%
WorkstationTPM	155845	0	0	4204	2.70%

In Table I the large number of vulnerable keys in the workstation certificates should be noted, which will be discussed below. The second notable observation is the occurrence of all types of vulnerabilities in only one type of certificate: TLS servers. The explanation for this observation is quite simple: the keys for the different types of certificate are generated in different environments. For the TLS servers mentioned above, the generation takes place in an environment that is extremely heterogeneous in terms of both hardware and software, which is why all three vulnerabilities were found here. For the other types of certificates, the keys are generated on a limited number of device types. For example, personal certificate keys are generated either on a limited set of smart cards (two vendors) or in CA software (archived encryption keys). Similarly, the workstation keys are generated only in TPM modules. A detailed analysis of the results obtained for each vulnerability is given below.

A. *Analysis of the results of the Debian weak keys test*

During the test, eight certificates with vulnerable keys were identified. These certificates were issued between 2007 and 2011 and are now long expired. The limited number of vulnerable keys can be attributed to the infrequent use of Debian-based systems within Orange Poland. Red Hat, CentOS, and, more recently, Rocky Linux are the primary Linux distributions used by organizations, including Orange, in which the described vulnerability did not occur.

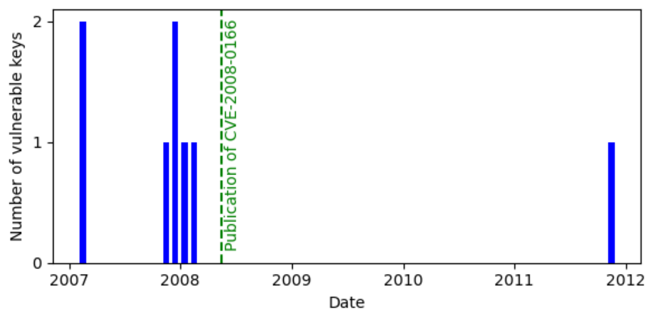

Fig. 2. Distribution of Debian weak keys over time, before and after publication security updates.

Fig. 2 illustrates that the systems have been updated, as the only exception to the vulnerable keys being generated before the vulnerability was published. It is difficult to explain the reason for the appearance of the vulnerable key 3 years after the vulnerability was published. It is likely that the key was generated on the old version of the system before the system update addressed the vulnerability. Therefore, it is advisable to carry out the update right after system installation and proceed with the remaining configuration tasks only afterward.

B. *Analysis of the results of vulnerabilities resulting from the sharing components of keys*

The study found 17 vulnerable keys in the first set and no vulnerable keys in the second set. 14 keys were generated in 2007 in IBM's Remote Supervisor Adapter II (RSAII) server access and management cards (Fig. 3). The cause of the vulnerable keys is a problem identified by the vulnerability identifier CVE-2012-2187. These cards use only 9 different key factors, which allows only 36 unique public keys to be generated. The rest of the vulnerable keys (3 pieces) are

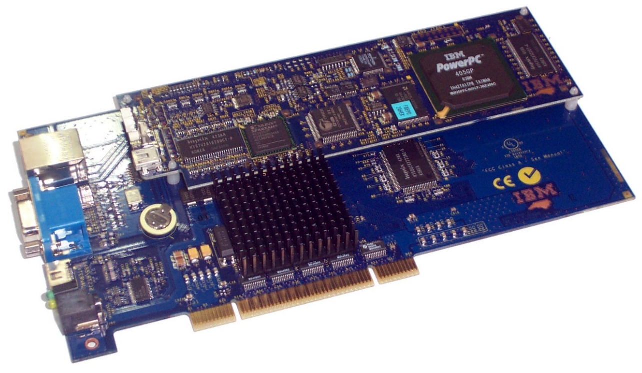

Fig. 3. Remote Supervisor Adapter II card, which generated predictable keys for several years.

from personal certificates from 2003, 2004 and 2007 (long expired) generated on most likely corrupted GemPLUS type smart cards from 2000 year. Fig. 4 shows incorrect modulus from this smart card. These cards were withdrawn from use at Orange in 2006 and replaced by cards from the SafeNet eToken family in the form of a USB key.

```
RSA Public-Key: (1024 bit)
    Modulus:
        00:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:
        ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:
        ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:
        ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:fe:ff:
        ff:ff:ff:ff:ff:00:00:00:00:00:00:00:00:00:00:00:
        00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:
        00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:
        00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:
        00:00:01:00:00:00:00:00:00
    Exponent: 65537 (0x10001)
```

Fig. 4. Example of a faulty key generated on a damaged smart card GemPlus GPK2000.

C. *Analysis of ROCA test results*

The examination of the certificates for ROCA vulnerabilities revealed a significant number of vulnerable keys. Virtually all of the cases originated from the second set of data, from the ADCS-based authority. Two cases from the first dataset were generated in software using a TPM for test purposes. Further analysis revealed that all vulnerable keys from the second dataset came from Windows-based workstations, where keys are generated in TPM modules (via Microsoft Platform Crypto Provider). The study found that above 2% of the workstation certificate keys were vulnerable. An analysis of the hardware types revealed that these are older models where the firmware for the TPM has not been updated. These workstations are being systematically retired, as shown in the chart on Fig. 5. Furthermore, the task of updating the TPM firmware has

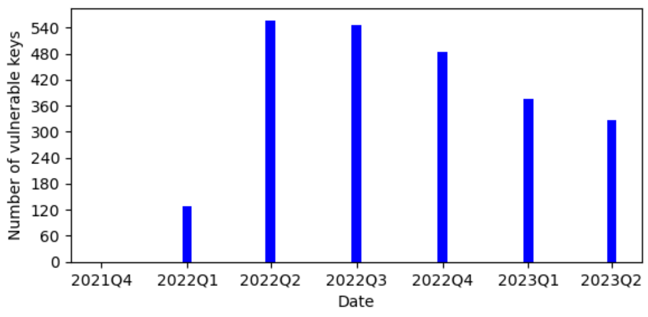

Fig. 5. Number of keys with ROCA vulnerability over time.

been undertaken, as well as reducing the validity of certificates for these stations from 4 months to a single days. The latter action is intended to reduce the sense of factoring vulnerable keys, which is still a lengthy process that takes decades on a single CPU [13]. The simplest option would be to extend all RSA keys to 3072 bits (factoring 3072-bit vulnerable keys is impractical, taking 10^{25} years on a single CPU [13]). However, RSA key lengths greater than 2048 are not supported in the Microsoft Platform Crypto Provider. Currently, the number of stations with a vulnerable TPM module is decreasing due to the TPM upgrade process and the retirement of older stations.

The above observations are consistent with similar studies cited previously. The current absence of the Debian weak-keys vulnerability is due to the presence of a vulnerable OpenSSL package only during a specific period of time. Furthermore, key sharing has been associated only with fairly old embedded solutions. The presence of the ROCA vulnerability on a significant number of stations is due to a lack of organization in the firmware update process. Firmware updates are frequently considered unimportant and even risky for hardware. This perspective may be influenced by the fact that firmware updates must be installed without launching the operating system and using a basic text interface. In contrast, operating system updates are installed in the background and only require the end user to restart the system at their convenience. Currently, Windows allows firmware updates to be included in the Windows Update service [25]. However, implementing the process can be time-consuming in a heterogeneous office environment.

VI. Discussion

The RSA key quality studies described in this article used a collection of RSA keys generated over more than 20 years. The acquisition of a large key base in the organization was made possible by the implementation of PKI services according to requirements that are much higher than those usually accepted for internal services of this type. The experience of the service implementation and maintenance team, processes and technical solutions was built based on the requirements of WebTrust$^{SM/TM}$ and the qualified certification center. This ensured the continuity of the Certification Center, including the preservation of historical certificates during subsequent software and hardware migrations. As a result, it is possible to observe how Orange Polska deals with the vulnerabilities described in the article over a fairly long period of time. A good example is the Debian weak keys vulnerability (CVE-2008-0166), which was discovered in several certificates issued in 2007 and 2008. However, it did not appear in the following years (except in one case), mainly due to the use of system updates. The BatchGCD vulnerability has been detected only in keys generated in embedded systems, such as IBM RSA II remote access cards and faulty smart cards. This confirms the theory often advanced by security researchers that embedded systems, lacking a good source of entropy and limited computing power, generate cryptographic keys of poor quality. ROCA vulnerability has been treated much worse. The significant number of workstations with a vulnerable TPM module indicates that firmware updates of workstation components are not performed regularly or at all. Both the keys used in certificates and, for example, in BitLocker disk encryption can be cracked, allowing unauthorized people to bypass security. Unfortunately, the implementation of the workstation certification at the beginning of 2022 does not allow for studying the development of the vulnerability in an earlier period. However, the discovery of the described vulnerability at the time of the article's writing triggered actions in Orange Polska to minimize the risk. The research also led to the development of tools for extracting certificates from the ADCS service database. The author of the publication did not find a ready-made tool for exporting certificates from this database. Although the ADCS service allows certificates to be published to a Active Directory in the particular configuration used in the company, there was no need to do so, and the only place to store certificates was in the ADCS service database.

A. *Significance of the results for the security of cryptographic systems based on RSA keys*

Periodic testing of system security is one of the pillars of security. As a result of testing the RSA keys, an unpatched vulnerability was discovered that affected several hundred computers. This incident highlights the need for security testing of cryptographic keys as well. Identifying repositories for such keys can be challenging; the study relied only on relatively readily available centralized X.509 certificate repositories. However, RSA keys are likely to be used in enterprises for other applications, such as SSH servers, PGP, and proprietary solutions. Further research may attempt to obtain public keys from less obvious sources. Organizations using cryptographic keys cannot rely on trust in cryptography alone; they must actively verify that trust using vulnerability detection tools. In the case of PKI, this could be an add-on module that implements periodic checks of issued certificates against known vulnerabilities, with the possibility of adding additional checking tools.

The proposed solutions to mitigate the threat of weak keys are as follows:

1) Early Verification of Certificate Requests: For key applications to PKI, it is relatively easy to perform testing in software that registers certificate requests. A certificate request in PKCS#10 format contains, in addition to the data to be placed in the certificate, the public key to be placed in the certificate. In the logical architecture of the certification center, there is a separate module responsible for registering requests, called RA (Registration Authority). In the RA, in addition to verifying the certificate data, it is relatively easy to add the function of verifying the quality of the key. Ideally, such checks should be performed automatically, and in case of a negative verification, the operator of the certificate issuing authority and the certificate applicant should be informed immediately. This would make it possible to eliminate weak keys at an early stage, identify the source of the vulnerability, and take appropriate action, such as updating the firmware of the TPM module.

2) Shortening the validity period of certificates: In the case of weak keys with ROCA vulnerabilities, issuing short-lived certificates does not give a potential attacker enough time to break the key and use it with a still-valid certificate. The costs and factorization times given in [13] for 2048-bit RSA keys indicate that short-lived certificates (a few days) associated with these keys may defeat the purpose of launching attacks in some applications, such as WiFi authentication using EAP-TLS. However, this approach should be considered temporary because it ignores the issue of the impact of the BitLocker (disk encryption) vulnerabilities. The ultimate solution is to update the TPM firmware or decommission the computer. For other vulnerabilities, the solution is to revoke the certificate and generate a new key using updated tools.

VII. Conclusions and Future Work

In the described research, a search for vulnerable RSA keys was performed from the acquired certificate sets of the internal PKI service of Orange Polska. The tested data set contained about 650,000 certificates issued between October 2002 and July 2023. The tests revealed a serious number of vulnerable keys in the currently used certificates. Dozens of issued keys were also found to be associated with long-expired certificates. An analysis of the sources of the vulnerabilities was performed, and the methods implemented in the company were described and proposed to minimize the impact of the vulnerabilities and also to eliminate vulnerabilities in the future.

Further research should focus on two key goals. The first goal is to identify and acquire key sets for vulnerability testing. The methods described in the article, which rely on access to centralized certificate collections, are relatively easy for internal cybersecurity teams. The only challenges are organizational constraints, as access to CAs' databases and management consoles within an organization should be very tightly controlled. Other potential sources of keys may include results from network scanning of services, as previously described in the literature [1], [2]. It is important to note that data acquired will be duplicated for services that utilize the TLS protocol and certificates issued from the corporate PKI. However, the scan may discover self-signed certificates and certificates issued by ad hoc PKIs or shadow-IT systems, due to some redundancy in the scan. Furthermore, it can also obtain keys from services that do not utilize the TLS protocol, such as SSH. If implemented in the organization, key discovery as a result of scanning can be incorporated into the existing vulnerability identification process. However, restricted access to network layer services limits the effectiveness of network scanners.

It is worth noting that completely opening the traffic of all services to the organizational vulnerability scanner increases the attractiveness of attempted attacks on the scanner itself [26], [27]. Vulnerabilities in security software are not unusual [28]. If cybercriminals were to gain access to the scanner, they could take over additional servers and services without detection. Malicious traffic could be disguised as legitimate scanning activity by other security systems.

Furthermore, this article addresses three vulnerabilities related to RSA keys and proposes the expansion of the list of testing tools for future studies. In the case of keys based on different algorithms, they will, of course, be other test tools, but the use of other test tools can be based on the developed methods described in this article. It may also be worthwhile to verify certificate compliance with a specifically selected set of CA/Browser Forum requirements.

References

[1] Z. Durumeric, J. Kasten, M. Bailey, and J. A. Halderman, "Analysis of the HTTPS certificate ecosystem," ser. IMC '13. New York, NY, USA: Association for Computing Machinery, 10 2013, p. 291–304, [Online; accessed 2023-04-26]. [Online]. Available: https://dl.acm.org/doi/10.1145/2504730.2504755

[2] M. Hastings, J. Fried, and N. Heninger, "Weak Keys Remain Widespread in Network Devices," ser. IMC '16. New York, NY, USA: Association for Computing Machinery, 11 2016, p. 49–63, [Online; accessed 2023-07-30]. [Online]. Available: https://dl.acm.org/doi/10.1145/2987443.2987486

[3] A. K. Lenstra, J. P. Hughes, M. Augier, J. W. Bos, T. Kleinjung, and C. Wachter, "Ron was wrong, Whit is right," [Online; accessed 2023-04-26]. [Online]. Available: https://eprint.iacr.org/2012/064.pdf

[4] L. M. Kohnfelder, "Towards a practical public-key cryptosystem." Ph.D. dissertation, 1978, accepted: 2005-08-04T15:38:48Z. [Online]. Available: https://dspace.mit.edu/handle/1721.1/15993

[5] N. Serrano, H. Hadan, and L. J. Camp, "A Complete Study of P.K.I. (PKI's Known Incidents)," 7 2019, [Online; accessed 2023-08-15]. [Online]. Available: https://papers.ssrn.com/abstract=3425554

[6] "Timeline of Certificate Authority Failures - SSLMate," [Online; accessed 2023-08-15]. [Online]. Available: https://sslmate.com/resources/certificate_authority_failures

[7] "Replace Your Symantec SSL/TLS Certificates | Digicert & Symantec," [Online; accessed 2023-08-15]. [Online]. Available: https://www.secure128.com/blog/replace-your-symantec-ssl-tls-certificates

[8] N. van der Meulen, "Diginotar: Dissecting the First Dutch Digital Disaster," *Journal of Strategic Security*, vol. 6, no. 2, 7 2013. [Online]. Available: https://digitalcommons.usf.edu/jss/vol6/iss2/4

[9] Z. Dong, K. Kane, and L. J. Camp, "Detection of Rogue Certificates from Trusted Certificate Authorities Using Deep Neural Networks," *ACM Transactions on Privacy and Security*, vol. 19, no. 2, pp. 1–31, 9 2016.

[10] J. Amann, O. Gasser, Q. Scheitle, L. Brent, G. Carle, and R. Holz, "Mission accomplished? HTTPS security after diginotar," ser. IMC '17. New York, NY, USA: Association for Computing Machinery, 11 2017, p. 325–340, [Online; accessed 2023-08-15]. [Online]. Available: https://dl.acm.org/doi/10.1145/3131365.3131401

[11] Q. Scheitle, T. Chung, J. Hiller, O. Gasser, J. Naab, R. van Rijswijk-Deij, O. Hohlfeld, R. Holz, D. Choffnes, A. Mislove, and G. Carle, "A First Look at Certification Authority Authorization (CAA)," *ACM SIGCOMM Computer Communication Review*, vol. 48, no. 2, p. 10–23, 5 2018.

[12] D. J. Bernstein, "How to find smooth parts of integers."

[13] M. Nemec, M. Sys, P. Svenda, D. Klinec, and V. Matyas, "The Return of Coppersmith's Attack: Practical Factorization of Widely Used RSA Moduli," ser. CCS '17. New York, NY, USA: Association for Computing Machinery, 10 2017, p. 1631–1648, [Online; accessed 2023-08-14]. [Online]. Available: https://doi.org/10.1145/3133956.3133969

[14] S. Yilek, E. Rescorla, H. Shacham, B. Enright, and S. Savage, "When private keys are public: results from the 2008 Debian OpenSSL vulnerability," ser. IMC '09. New York, NY, USA: Association for Computing Machinery, 11 2009, p. 15–27, [Online; accessed 2023-07-30]. [Online]. Available: https://doi.org/10.1145/1644893.1644896

[15] "Debian – Security Information – DSA-1571-1 openssl," [Online; accessed 2023-08-14]. [Online]. Available: https://www.debian.org/security/2008/dsa-1571

[16] D. Bernstein, N. Heninger, and T. Lange, "Facthacks: RSA factorization in the real world." [Online]. Available: http://events.ccc.de/congress/2012/Fahrplan/events/5275.en.html

[17] Z. Durumeric, E. Wustrow, and J. A. Halderman, "ZMap: Fast Internet-wide Scanning and Its Security Applications," 2013, pp. 605–620, [Online; accessed 2023-08-15]. [Online]. Available: https://www.usenix.org/conference/usenixsecurity13/technical-sessions/paper/durumeric

[18] Y. Zhang, B. Liu, C. Lu, Z. Li, H. Duan, J. Li, and Z. Zhang, "Rusted Anchors: A National Client-Side View of Hidden Root CAs in the Web PKI Ecosystem," ser. CCS '21. New York, NY, USA: Association for Computing Machinery, 11 2021, p. 1373–1387, [Online; accessed 2023-08-15]. [Online]. Available: https://doi.org/10.1145/3460120.3484768

[19] "Zlint," 8 2023, original-date: 2016-11-30T18:42:16Z. [Online]. Available: https://github.com/zmap/zlint

[20] "Orange | FR0000133308 | Euronext exchange live quotes," [Online; accessed 2023-08-15]. [Online]. Available: https://live.euronext.com/en/product/equities/FR0000133308-XPAR/company-information

[21] "Baseline Requirements for the Issuance and Management of Publicly-Trusted Certificates."

[22] D. J. Bernstein, N. Heninger, and T. Lange, "FactHacks: Batch gcd." [Online]. Available: https://facthacks.cr.yp.to/batchgcd.html

[23] zugzwang, "(C++)+GMP BatchGCD algorithm," Feb. 2022, original-date 2019-11-21. [Online]. Available: https://github.com/zugzwang/batchgcd

[24] "ROCA detection tool," 7 2023, original-date: 2017-10-13T15:28:41Z. [Online]. Available: https://github.com/crocs-muni/roca

[25] T. Hudek, "Update device firmware using Windows Update - Windows drivers," 10 2022, [Online; accessed 2023-08-16]. [Online]. Available: https://learn.microsoft.com/en-us/windows-hardware/drivers/install/updating-device-firmware-using-windows-update

[26] "Exploiting (Almost) Every Antivirus Software – RACK911 Labs," [Online; accessed 2023-08-15]. [Online]. Available: https://rack911labs.ca/research/exploiting-almost-every-antivirus-software/

[27] K. W. Hamlen, V. Mohan, M. M. Masud, L. Khan, and B. Thuraisingham, "Exploiting an antivirus interface," *Computer Standards & Interfaces*, vol. 31, no. 6, pp. 1182–1189, 11 2009.

[28] "Searching for vulnerabilities in Security Products," [Online; accessed 2023-08-15]. [Online]. Available: https://www.cvedetails.com/product-search.php?vendor_id=0&search=security