ELSEVIER

Performance Evaluation 36–37 (1999) 213–232

PERFORMANCE EVALUATION
An International Journal

www.elsevier.com/locate/peva

Analytical performance evaluation of nested certificates

Albert Levi *, M. Ufuk Çağlayan [1]

Boğaziçi University, Department of Computer Engineering, Bebek, Istanbul 80815, Turkey

Abstract

The *classical certificate* systems are computationally inefficient, since they use signature operations based on public key cryptosystems. The *nested certificates* (A. Levi, Design and performance evaluation of the nested certification scheme and its applications in public key infrastructures, Ph.D. Thesis, Department of Computer Engineering, Boğaziçi University, Istanbul, Turkey) are proposed to improve the performance of the certificate verification. A nested certificate is a certificate for another, say *subject*, certificate. The subject certificates can be classical or other nested certificates. A subject certificate can be verified without using the public key cryptosystem operations. In this way, the nested certificates improve the performances of the certificate and certificate path verification. In this paper, analytical formulations and graphical analyses of the computational performance improvement of the nested certificate usage are given for both single subject certificate verification and certificate path verification cases. Moreover, it is also shown that the usage of nested certificates always improves the computational performances of the verification of a single certificate and the verification of a certificate path.

Keywords: Computer network security; Public key infrastructure; Digital signature; Digital certificate; Performance evaluation

1. Introduction

Public key cryptosystems are used extensively in network security and authentication applications. One of the most important reasons behind this popularity is the ease of key distribution in public key cryptography. The public–private key pairs are created by the owner of the key. The private keys are used to decrypt messages and digitally sign information. Therefore, they must be kept secret. On the other hand, the public keys are used to encrypt messages and to verify digital signatures. Since these operations can be carried out by anyone, the public keys can be known by everyone. Since the public–private key pairs are created by the owners themselves, there should be a mechanism to introduce them as valid users to other users. The rationale for this requirement is to avoid *name spoofing*. A public key owner may introduce him/herself with a different name and the other users of the application reckon this user as he/she is the claimed person; however, he/she is not the person that he/she says so.

* Corresponding author. E-mail: levi@boun.edu.tr
[1] E-mail: caglayan@boun.edu.tr

0166-5316/99/$ – see front matter
PII: S0166-5316(99)00022-X

The general practice to avoid name spoofing is to use some trusted entities to introduce the users to the system. These trusted introducers digitally sign the binding between the public key and the real identity of a user. The application partners verify the signature of the introducer over that binding using the introducer's public key and make sure about the validity of the public key of the user. Such a system can work if and only if the introducer is a trusted entity for the verifier and the verifier knows the introducer's public key, because otherwise the verifier cannot verify and comment on the signature of the introducer on the identity–public key binding of the user. This digitally signed information, which contains the public key of a person with the identity information and some managerial details, is called *certificate* and the introducer is called *Certification Authority* (*CA*). Throughout the paper, such *certificates* will be named as *classical certificates*, in order to distinguish from the nested certificates that will be explained later. International Telecommunication Union (ITU) X.509 standard [12] is a standard on the classical certificates. Pretty Good Privacy (PGP) [13,14] software also uses classical certification techniques widely for public key distribution.

It is obvious that a single CA is not sufficient for a network with large number of users. Therefore, there must be several CAs throughout the global network. Moreover, there must be a classical certificate network to connect the CAs and other users. Such network is named as *Public Key Infrastructure* (*PKI*). In that way, the users, who have classical certificates from different CAs, will be able to verify each other's certificate.

In order to issue a classical certificate, the CA digitally signs the certificate content using its private key. That classical certificate can be verified using the public key of the same CA. Thus, in order to verify the classical certificate and to find out the public key of the target entity *T*, the verifier *V* must know the correct public key of the CA of the classical certificate of *T*. If *V* does not know it, then it has to verify the classical certificate for that CA. To do so, *V* has to know the public key of the CA of the CA of *T*. This loop goes on until *V* is faced with a classical certificate that it can directly verify (i.e. it knows the public key of the corresponding CA). The classical certificates in this loop constitute a path, which is named as *classical certificate path*. This path is a directed one and the starting point is a CA, of which *V* knows the public key, and the ending point is *T*. In this path, each classical certificate is verified to find out the public key of the next CA and each public key is used to verify the next classical certificate. The verifier has to know the public key of the first CA and has to trust all the CAs on the path. The certificates of such a classical certificate path are drawn from the PKI. A generic classical certificate path is shown in Fig. 1.

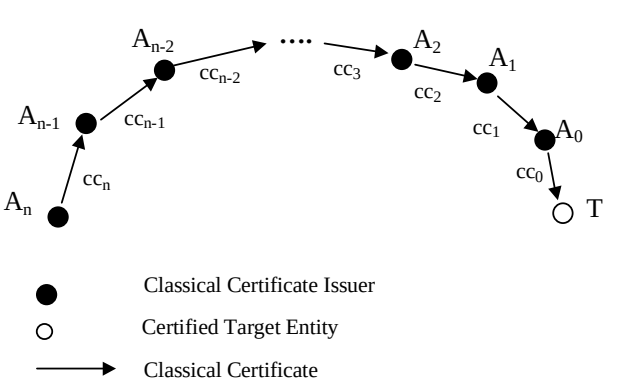

Fig. 1. A generic classical certificate path.

The basic rule of the classical certificate verification is the existence of a legitimate signature, which has been issued by a trusted CA, over the certificate content. The digital signatures are issued and verified by employing a public key cryptosystem, like the RSA [8] or the DSA [3] cryptosystems. Moreover, one way hash algorithms [7,4,10] are also used. The CA first calculates the hash of the certificate content and signs this hash instead of the whole certificate content. In order to verify a classical certificate, *cert*, using a *cryptographic certificate verification* mechanism, the verifier must know the correct public key of the issuer of *cert*. Assuming that this public key is known by the verifier, the verifier follows the following steps to cryptographically verify *cert*:

(1) The verifier first applies the one way hash algorithm to the content of *cert*.

(2) The verifier applies the public key cryptosystem based signature verification procedure to the signature part of *cert* using the public key of the issuer of *cert*.

(3) The verifier compares the outcomes of the above steps. If they are the same, then the verifier makes sure about: (i) the integrity of the content of *cert* and (ii) the legitimacy of the signature of the issuer of *cert* over the content of *cert*.

(4) If the issuer of *cert* is trusted for the verifier, then the above two results of the verification implies that: (iii) the information given in *cert* is correct. In other words, the public key of the entity specified in *cert* is legitimate.

In order to verify the classical certificate of a target entity *T* via a certificate path, the verifier *V* verifies all of the classical certificates one by one sequentially. The verification starts with the first certificate of the path and *V* must know the correct public key of the first CA. However, *V* may not be the first CA, it may be any user. On the other hand, the trust of *V* to all of the CAs on the path is essential in order to verify the path. The cryptographic certificate verification steps are applied for the verification of all of the classical certificates on the path. Each certificate verification yields a public key, which is to be used to verify the next certificate on the path. This loop goes on until the target entity is reached.

The public key cryptosystem operations employed in the classical certificate verification are computationally complex. Therefore, there is an efficiency bottleneck. This handicap becomes more serious especially for the classical certificate paths in which several public key cryptosystem based signature verifications are performed. The nested certificates [5] are proposed to improve the computational efficiency of the certificate verification. The nested certificates are used to verify their *subject* certificates without using public key cryptosystem operations. Therefore, subject certificate verification is computationally more efficient than the cryptographic verification. The *Nested Certificate based PKI* (*NPKI*) [6,5] is proposed to take the advantages of the nested certificates in the PKIs. In NPKI certificate paths, both classical and nested certificates are used together. The contribution of this paper is the analytical performance evaluation of the nested certificate usage in the subject certificate verification and the NPKI certificate path verification methods. The performance measure is the relative computational speed-up factors. Two such speed-up factors are analyzed in this paper. One of them is the speed-up factor for the single subject certificate verification time over the cryptographic certificate verification time. The other speed-up factor is the one for the NPKI certificate path verification time over the classical certificate path verification time. In order to analyze these speed-up factors, formulations of them will be derived. Moreover, these formulations will prove that the usage of nested certificates always improves the computational efficiency of the certificate verification. Such an efficiency also exists for the NPKI certificate paths.

In Section 2, the analytical performance evaluation of the subject certificate verification method will be given. In that section, an overview of the nested certificates will also be provided. Section 3 is about

the performance evaluation of the NPKI certificate paths. An overview of the NPKI certificate paths will be the introductory part of Section 3. The conclusions of this work will be given in Section 4.

2. Analytical performance evaluation of the subject certificate verification

In this section, the performance of the subject certificate verification will be analyzed. The performance measure is the relative speed-up factor for the subject certificate verification time over the cryptographic verification time of the same certificate. Firstly, an overview of the nested certificates will be given in Section 2.1. The verification techniques will be briefly explained in this overview. The formulation of the speed-up factor will be derived in Section 2.2. The graphical analysis of the speed-up factor will be given in Section 2.3.

2.1. Overview of the nested certificates

Nested certification is proposed as a new and alternative certificate scheme in [5]. The certificates issued in this scheme are named *nested certificates*. A nested certificate is used to certify another, say *subject*, certificate. Therefore, it is considered as 'a certificate for another certificate'. The basic idea behind the nested certification is to delegate the subject certificate signature verification responsibility to the nested certificate issuer. In this way, the signature over the subject certificate can be verified via a nested certificate. Moreover, the verifier need not know and find out the public key of the subject certificate issuer. The subject certificates can be classical or other nested certificates. For example in Fig. 2, certificate 2 is a nested certificate for a classical certificate (certificate 1). However, certificate 3 is a nested certificate for another nested certificate (certificate 2). The nested certificates are issued by *Nested Certificate Authorities* (*NCAs*).

It is worthwhile to mention that the nested certificates are not designed to replace all the functions of the classical certificates. On the contrary, the nested certificates are designed to improve the performance and flexibility of the classical certificate usage. Therefore, both classical and nested certificates can be used together in PKIs and certificate paths.

The subject certificate of a nested certificate is a previously issued classical or nested certificate. The nested certificate issuance is similar to classical certificate issuance. The nested certificates are issued by the digital signature of the NCA over the nested certificate content. The content of a nested certificate is related to its requirements. The two requirements of a nested certificate are:

(i) to certify that the subject certificate content has been signed by the claimed CA or NCA;

(ii) to certify that the subject certificate content has not been maliciously modified.

In order to satisfy the first requirement, a nested certificate contains the existing signature over the

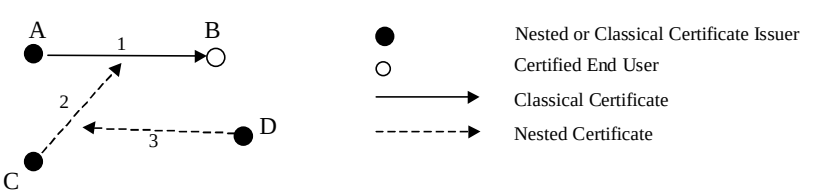

Fig. 2. The relationships between nested and subject certificates.

subject certificate content. In order to satisfy the second requirement, a nested certificate contains the hash of its subject certificate content. The hash of the subject certificate content can be obtained by applying an irreversible one way hash function [7,4,10] to the subject certificate content.

In order to verify a subject certificate via a nested certificate, the certifier nested certificate must have been verified successfully beforehand, because otherwise the nested certificate content would be suspicious. The nested certificates can be verified like the classical certificates. That means, the signature over the nested certificate content can be verified cryptographically. Another way of nested certificate verification is to verify it as a subject certificate of another nested certificate. The subject certificate verification method can also be applied to classical certificates. Cryptographic nested certificate verification is given in Section 2.1.1. The subject certificate verification method is explained in Section 2.1.2.

2.1.1. Cryptographic verification of a nested certificate

The cryptographic verification of a classical certificate was explained in Section 1. The idea behind the cryptographic verification of a nested certificate is the same as the verification of a classical certificate. Cryptographic verification of a nested certificate also requires a legitimate signature due to a trusted NCA over the nested certificate content. The verifier applies the following steps to verify that signature:

(1) The one-way irreversible hash [7,4,10] of the nested certificate content is calculated.

(2) The signature part of the nested certificate is cryptographically verified using the public key of the NCA.

(3) If the outcomes of the previous two steps match, then it can be concluded that the signature over the nested certificate is legitimate, otherwise either the certificate has been modified or the signature is not correct.

(4) If the NCA is trusted for the verifier, then the verifier concludes that the information within the nested certificate is correct. That means, the verifier finds out the correct hash and the correct signature over the subject certificate content.

2.1.2. Subject certificate verification

The information found out by nested certificate verification is not sufficient to verify its subject certificate. By the verification of a nested certificate, only the correct hash value and correct signature over the subject certificate are found. In order to verify the subject certificate, the actual hash and the actual signature over the subject certificate must be compared with the ones stored in the nested certificate. Verification of a certificate as the subject certificate of a nested certificate is called *subject certificate verification*. Although the subject certificate verification method is a consequence of the nested certificates, it can be used to verify both the nested certificates and classical certificates, since the subject certificates can be of both types. Having verified a nested certificate, *nc*, in order to verify its subject certificate, *sc*, the verifier follows the following two steps:

(a) The hash of the content of the actual *sc* is recalculated. This recalculated hash must be the same as the one stored within the *nc*.

(b) The actual signature over the content of the *sc* is compared with the subject certificate signature stored in the *nc*. These two signature values must be the same.

If the conditions in the above steps are met and the issuer of *sc* is trusted as the verifier, then the verifier concludes that the *sc* is legitimate. The verifier must trust the issuer of *sc*, since the verification

of *sc*, using the above steps, does not mean that the information stored within *sc* is correct. The verifier makes sure about correctness of the information contained in *sc*, if the issuer of *sc* is trusted. It is very important to point out that, in this way, the subject certificate *sc* becomes verified, with the same confidence, but without employing a signature verification method based on a public key cryptosystem. The phrase 'the same confidence' means that the correctness level of the information found out by a subject certificate verification is the same as that of cryptographic certificate verification. The formal proof of this conclusion can be found in [5].

The subject certificate verification method does not employ a signature verification scheme based on a public key cryptosystem. Therefore, verification of a subject certificate via a nested certificate is computationally more efficient than the cryptographic verification. The detailed performance evaluation is given in the subsequent sections.

2.2. Formulation of subject certificate verification performance

The most important advantage of the nested certification scheme is the computational efficiency of the subject certificate verification method as compared to the cryptographic certificate verification method. The reason behind this efficiency is the difference between the cryptographic certificate verification and the subject certificate verification methods. The subject certificate verification method does not employ any public key cryptosystem operation. A single hash computation and two comparisons are enough to verify a subject certificate via a nested certificate. Hash calculation is also a part of the cryptographic certificate verification method. Therefore, the time spent for the public key cryptosystem operation is the saving of the subject certificate verification method.

In this subsection, the relative *speed-up factor* for the subject certificate verification time over the public key cryptosystem based certificate verification time is formulated. The speed-up factor indicates how many times the subject certificate verification method is faster than the cryptographic certificate verification method. Moreover, it will be proven that the subject certificate verification method is always computationally more efficient than the cryptographic certificate verification method. In order to formulate the speed-up factor, the total cryptographic certificate verification time and the total subject certificate verification time are also formulated.

The first step of the cryptographic verification of a certificate is the hash computation of the certificate content. Then, the public key cryptosystem based signature verification operation is applied to the signature part of the certificate. In this way, the verifier finds out the correct hash that the CA had signed. Finally, the computed hash is compared with the verified hash for equality. $T_{pkc}(h, c, cert)$ is the total time of the cryptographic verification of the certificate *cert* which uses the hash algorithm h and the public key cryptosystem c. Neglecting the time for the comparison, $T_{pkc}(h, c, cert)$ is formulated as:

$$T_{pkc}(h, c, cert) = t_h + \frac{S(cert)}{\lambda_h} + t_c \tag{1}$$

where, $S(m)$ is the *size-of* function and returns the bit length of its argument m. h is the hash algorithm used, like the MD5 [7] or SHA-1 [10] algorithms. t_h is the fixed setup time for the hash algorithm h in milliseconds (ms). λ_h is the throughput of the hash algorithm h in bits/ms. c is the public key cryptosystem used, like the RSA [8] or DSA [3] cryptosystems. t_c is the time necessary for the verification in the public key cryptosystem c in ms. The *cert* is the certificate content, which is to be verified.

The size of the signature over the hash *cert* does not depend on the size of *certificate* and the hash algorithm used. It solely depends on the cryptosystem used for signing. For example, in the RSA cryptosystem, the signature size is the same as the modulus size of the signing key, for the DSA cryptosystem, the signature has two 160-bits long parts. That is why t_c does not depend on $S(cert)$ and the hash size.

The time necessary to compute the hash of a certificate content has two parts: (i) fixed setup time, t_h, which is for the initializations; (ii) hash calculation time, which depends on the size of the certificate content and calculated as $S(cert)/\lambda_h$.

On the other hand, the subject certificates are verified by one hash computation and two comparisons as described in Section 2.1.2. $T_s(z, scert)$ is the total time of the verification of the certificate *scert* as a subject certificate using the hash algorithm z. Neglecting the time for the comparisons, $T_s(z, scert)$ is formulated as:

$$T_s(z, scert) = t_z + \frac{S(scert)}{\lambda_z} \tag{2}$$

where $S(m)$ is the *size-of* function and returns the bit length of its argument m. z is the hash algorithm used, like the MD5 or SHA-1 algorithms. t_z is the fixed setup time for the hash algorithm z in ms. λ_z is the throughput of the hash algorithm z in bits/ms. The *scert* is the subject certificate content that is to be verified.

The relative improvement of the subject certificate verification over the cryptographic certificate verification must be analyzed for the same certificates, that means, $cert = scert$. Further assume that the same hash algorithms are employed for both type of verifications, that means, $h = z$. $f(h, c, cert)$ is the relative computational speed-up factor for the verification of the certificate *cert* as a subject certificate over the cryptographic verification of it. h is the hash algorithm and c is the public key cryptosystem used in the verifications. $f(h, c, cert)$ is formulated as:

$$f(h, c, cert) = \frac{T_{pkc}(h, c, cert)}{T_s(h, cert)} = \frac{t_h + \dfrac{S(cert)}{\lambda_h} + t_c}{t_h + \dfrac{S(cert)}{\lambda_h}} = 1 + \frac{t_c}{t_h + \dfrac{S(cert)}{\lambda_h}} = 1 + \frac{1}{\dfrac{t_h}{t_c} + \dfrac{S(cert)}{\lambda_h \cdot t_c}}. \tag{3}$$

The values of t_h, t_c, λ_h and $S(cert)$ are all positive. Therefore, as Eq. (3) implies, the relative speed-up factor $f(h, c, cert)$ is always larger than 1. This conclusion proves the following theorem.

Theorem 1. *Verification of a certificate as a subject certificate is always computationally more efficient than the verification of the same certificate by employing a signature verification scheme based on a public key cryptosystem, assuming that the same hash algorithms are used in both cases.*

2.3. Analysis of the subject certificate verification speed-up factor

The hash computation is a more computationally efficient process than the cryptographic verification of a signature. Bosselaers et al. [1,2] reported that $\lambda_{\text{MD5}} = 136$ kbits/ms, $\lambda_{\text{SHA-1}} = 55$ kbits/ms on a 90 MHz Pentium computer. On the other hand, Wiener [11] has reported that $t_{\text{RSA1024}} = 0.6$ ms, $t_{\text{DSA1024}} = 27$ ms on a 200 MHz Pentium computer. However, such performance values for hash functions and public key cryptosystem operations are not obtained on the same platform. For our

Table 1
Execution times for public key cryptosystem verifications and hash calculations

Algorithm	$t_{\text{Algorithm}}$ (ms)	$\lambda_{\text{Algorithm}}$ (bits/ms)
DSA512	33.909	–
DSA1024	113.968	–
RSA512	1.256	–
RSA1024	4.457	–
RSA2048	17.152	–
SHA-1	0.0093	36057.0
MD5	0.009	68267.0

analyses, we have measured the execution times of the MD5 [7] and SHA-1 [10] hash functions and the RSA [8] and DSA [3] cryptosystems with different key sizes. The measurements are obtained by running on a 166 MHz Pentium computer. The cryptographic library of the SECUDE toolkit [9] is used. Table 1 gives these measurements.

The behavior of the speed-up factor, $f(h, c, cert)$, versus the certificate size, $S(cert)$, for different cryptosystem and hash algorithm combinations is given by the graphs in Figs. 3 and 4. These graph are drawn by using the formula given in Eq. (3) and the t_c, t_h, λ_h values of Table 1.

The speed-up factor in Eq. (3), $f(h, c, cert)$, is directly related to cryptographic verification time, t_c. The speed-up factor is inversely related to the certificate size, $S(cert)$. These conclusions can be seen from Figs. 3 and 4. The speed-up factor varies between 8 and 3000 for the public key cryptosystem–hash function pairs considered. The speed-up factor becomes larger for the slower cryptosystems like the DSA512, DSA1024 and RSA2048, as shown in Fig. 3. Moreover, the speed-up factor decreases while the certificate size increases. However, there is a remarkable improvement even for the large certificate sizes. For example, where $S(cert) = 6000$ bits, then the speed-up factor varies between 8 and 640; in other words, the subject certificate verification is 8 to 640 times faster than the cryptographic verification. On the other hand, the typical certificate sizes are 3000 to 4500 bits. The speed-up factors for the typical certificate sizes are between 10 and 1220, depending on the algorithms used.

The speed-up factor for the RSA512 cryptosystem is the one with the smallest values as shown in Fig. 4. However, the average speed-up factor for RSA512 is 20. That means, the subject certificate verification method is 20 times faster than the RSA512 based cryptographic certificate verification. Although this factor is smaller than other cryptosystems, it is still a good improvement.

The hash algorithms also effect the speed-up factor. However, since the hash algorithms are much faster than the public key cryptographic operations, the effect of the hash computation time over the speed-up factor is not as significant as the execution time of the public key cryptosystem operations.

As can be seen from Figs. 3 and 4, the effect of the MD5 hash algorithm over the speed-up factor is better than the effect of the SHA-1 hash algorithm for the same cryptosystems. The throughput of the MD5 algorithm is greater than the one of the SHA-1 algorithm and the setup times are almost the same in both cases, as can be seen from Table 1. The effect of the MD5 algorithm over the speed-up factor is better, since the speed-up factor given by Eq. (3) is directly related to the hash throughput.

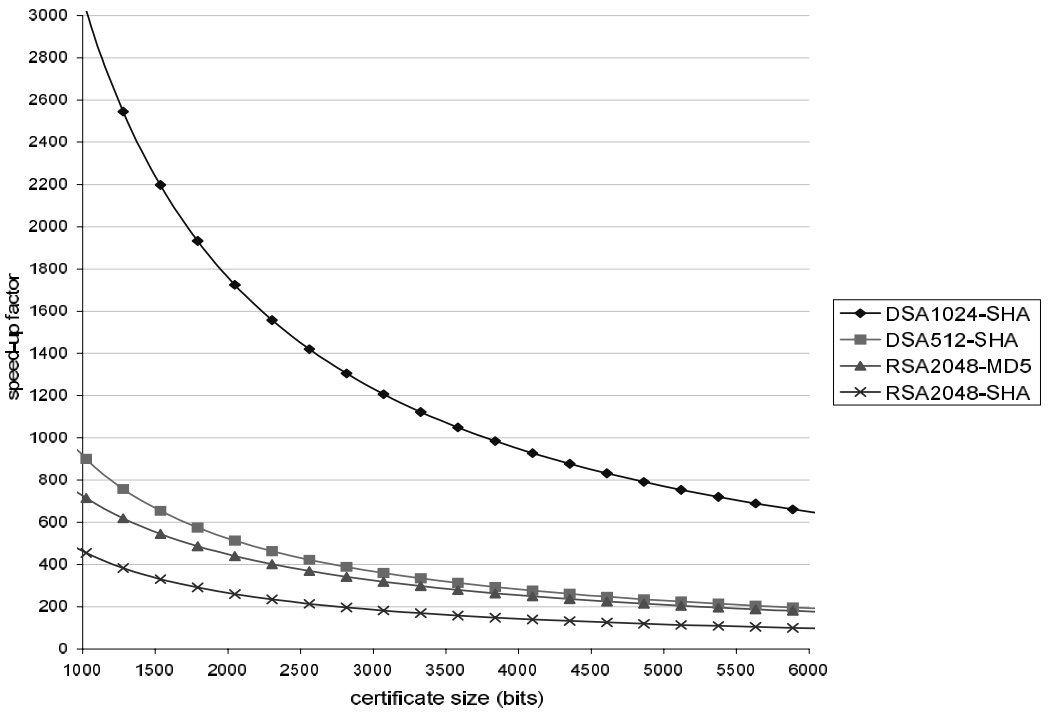

Fig. 3. Change of speed-up factor with respect to certificate size for different algorithm combinations.

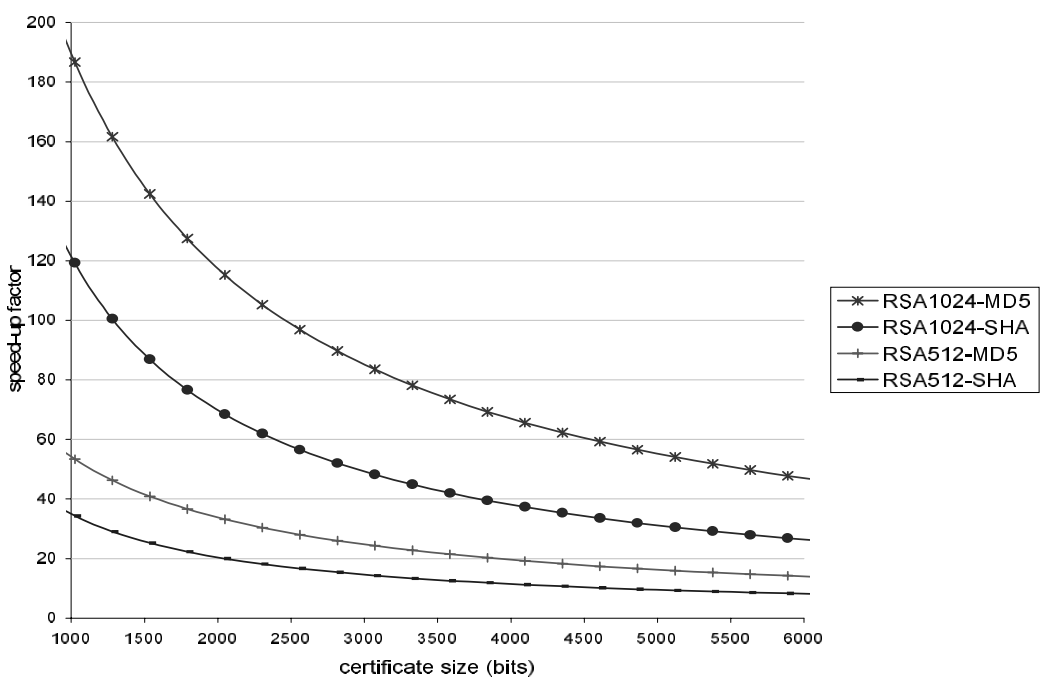

Fig. 4. Change of speed-up factor with respect to certificate size for different algorithm combinations.

3. Analytical performance evaluation of NPKI certificate path verification

In this section, the performance of the NPKI certificate path verification method will be analyzed. The performance measure will be the relative speed-up factor for the NPKI certificate path verification time over the classical certificate path verification time. Firstly, an overview of the NPKI certificate paths will be given in Section 3.1. The formulation of the speed-up factor will be derived in Section 3.2. An example graphical analysis of the speed-up factor will be given in Section 3.3. The analytical comparison of the speed-up factors of the subject certificate verification and the NPKI certificate path verification methods will be given in Section 3.4.

3.1. Overview of NPKI certificate paths

Nested certificates need to be embedded into the PKIs for their usages to become widespread. A PKI, which is capable of handling nested certificates as well, is called the *Nested certificate based PKI* (*NPKI*) [6,5]. In NPKI, classical certificates are used together with nested certificates. Moreover, in order to verify the binding between the identity and the correct public key for a target entity, the verifier must find a valid certificate path for which the end point is the target entity. Such a path is named as the *NPKI certificate path* and it is derived from the NPKI. The last certificate of the NPKI certificate path must be a classical certificate, because this certificate is used to verify the public key of the target entity and only classical certificates can certify public keys. The intermediary certificates on the NPKI certificate path can be either classical or nested certificates. An example NPKI certificate path is shown in Fig. 5. In this example path, certificates 1, 3, 4, 7 and 8 are classical certificates, whereas certificates 2, 5 and 6 are nested certificates.

All of the certificates on a NPKI certificate path are verified one by one sequentially. The verification process starts with the first certificate on the NPKI certificate path and ends with the certificate of the target entity. Each classical certificate is verified to find out the public key of the CA or the NCA of the next certificate. On the other hand, the nested certificates on the path are verified to validate their subject certificates. All of the certificates must be verified successfully, in order to consider the NPKI certificate path as valid and consequently to validate the public key of the target entity.

There are two verification methods for the certificates on a NPKI certificate path: (1) cryptographic verification method; (2) subject certificate verification method. The former one is the classical approach to verify a public key signature, which is explained in Section 1 for classical certificates and in Section 2.1.1 for the nested certificates. The latter method is explained in Section 2.1.2. The type of the predecessor certificate determines the verification method of a certificate on a NPKI certificate path. The first certificate has no predecessor and it is verified cryptographically using the public key of its issuer. This public key must be known by the verifier. If the predecessor certificate is a classical certificate,

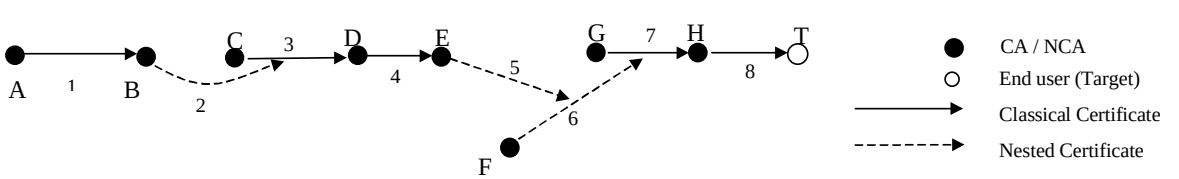

Fig. 5. An example of a NPKI certificate path.

then the current certificate (whatever its type is) is verified cryptographically using the public key of its issuer. This public key is obtained from the verification of the predecessor certificate. If the predecessor certificate is a nested certificate, then the current certificate (whatever its type is) is verified as the subject certificate of its predecessor. Therefore, the total number of subject certificate verifications is equal to the number of nested certificates on a NPKI certificate path. Consequently, the total number of cryptographic certificate verifications is equal to the number of classical certificates on the NPKI certificate path. For example, in Fig. 5, certificates 1, 2, 4, 5 and 8 are verified cryptographically, since they have either classical certificate predecessors or no predecessor. On the other hand, certificates 3, 6 and 7 are verified as subject certificates, since their predecessors are nested certificates.

The detailed analysis and the correctness proof of the NPKI certificate path processing algorithm can be found in [5]. The efficiency of the subject certificate verification also improves the NPKI certificate path verification time as compared to pure classical certificate paths. The detailed performance evaluation of the NPKI certificate path verification method is given below.

3.2. Formulation of NPKI certificate path verification performance

The verification of a certificate as a subject certificate is more efficient than the cryptographic verification of the same certificate as given by Theorem 1. In this subsection, the relative speed-up factor for the NPKI certificate path verification time over the classical certificate path verification time is formulated. The speed-up factor indicates how many times the NPKI certificate path verification method is faster than the classical certificate path verification method. Moreover, it is proven that the NPKI certificate path verification is always computationally more efficient than the classical certificate path verification. In order to formulate the speed-up factor, the NPKI and classical certificate path verification times are also formulated.

Let *N-path* be a NPKI certificate path. All of the certificates (nested or classical) on *N-path* must be verified either cryptographically or as subject certificates. In the *N-path* verification time formulation, three certificate sets will be used. These sets are $SubjectCerts_{N\text{-}path}$, $CryptCerts_{N\text{-}path}$ and $AllCerts_{N\text{-}path}$. The definitions of them are as follows:

$$SubjectCerts_{N\text{-}path} = \{cert_i \mid i \text{ is the index of the certificates to be verified as subject certificates}\}, \quad (4)$$

$$CryptCerts_{N\text{-}path} = \{cert_i \mid i \text{ is the index of the certificates to be verified cryptographically}\}, \quad (5)$$

$$AllCerts_{N\text{-}path} = SubjectCerts \cup CryptCerts. \quad (6)$$

In the *N-path*, the certificates with nested certificate predecessors are verified as subject certificates. Therefore, the total number of subject certificate verifications is the total number of nested certificates. Consequently, the total number of cryptographic verifications is the total number of classical certificates of the *N-path*. Thus, the numbers of elements of the above declared sets are given as:

$$|SubjectCerts_{N\text{-}path}| = n_{nc}^{N\text{-}path}, \quad (7)$$

$$|CryptCerts_{N\text{-}path}| = n_{cc}^{N\text{-}path}, \quad (8)$$

$$|AllCerts_{N\text{-}path}| = n_{total}^{N\text{-}path}. \quad (9)$$

where $n_{cc}^{N\text{-}path}$ is the total number of classical certificates, $n_{nc}^{N\text{-}path}$ is the total number of nested certificates of the *N-path*. $n_{total}^{N\text{-}path} = n_{nc}^{N\text{-}path} + n_{cc}^{N\text{-}path}$, i.e. the total number of certificates of the *N-path*.

$T_{N\text{-}path}(h, c)$ is the total *N-path* verification time. For the sake of the simplicity of the analysis, it is assumed that the same hash algorithm h and the same public key cryptosystem c are used for all of the certificates of the *N-path*. Under these assumptions, $T_{N\text{-}path}(h, c)$ is formulated as:

$$T_{N\text{-}path}(h, c) = \sum_{cert_p \in CryptCerts_{N\text{-}path}} T_{pkc}(h, c, cert_p) + \sum_{cert_t \in SubjectCerts_{N\text{-}path}} T_s(h, cert_t). \tag{10}$$

T_{pkc} and $T_s(h, cert)$ are the times for the verification of a certificate using a public key cryptosystem and the verification of a certificate as a subject certificate, respectively. Substituting Eq. (1) and Eq. (2) for $T_{pkc}(h, c, cert_p)$ and $T_s(h, cert_t)$, the total *N-path* verification time, $T_{N\text{-}path}(h, c,)$, becomes:

$$\begin{aligned} T_{N\text{-}path}(h, c) &= \sum_{cert_p \in CryptCerts_{N\text{-}path}} \left(t_h + \frac{S(cert_p)}{\lambda_h} + t_c\right) + \sum_{cert_t \in SubjectCerts_{N\text{-}path}} \left(t_h + \frac{S(cert_t)}{\lambda_h}\right) \\ &= \sum_{cert_p \in CryptCerts_{N\text{-}path}} t_c + \sum_{cert_p \in CryptCerts_{N\text{-}path}} \left(t_h + \frac{S(cert_p)}{\lambda_h}\right) + \sum_{cert_t \in SubjectCerts_{N\text{-}path}} \left(t_h + \frac{S(cert_t)}{\lambda_h}\right) \\ &= \sum_{cert_p \in CryptCerts_{N\text{-}path}} t_c + \sum_{cert_i \in AllCerts_{N\text{-}path}} \left(t_h + \frac{S(cert_i)}{\lambda_h}\right) \\ &= \sum_{cert_p \in CryptCerts_{N\text{-}path}} t_c + \sum_{cert_i \in AllCerts_{N\text{-}path}} t_h + \frac{n_{total}^{N\text{-}path}}{\lambda_h} \cdot \frac{\sum_{cert_i \in AllCerts_{N\text{-}path}} S(cert_i)}{n_{total}^{N\text{-}path}}. \end{aligned} \tag{11}$$

It has been assumed that the same hash algorithm h and the same cryptosystem c are used for all of the certificates of the *N-path*. Therefore, it can be deduced that:

$$\sum_{cert_p \in CryptCerts_{N\text{-}path}} t_c = n_{cc}^{N\text{-}path} \cdot t_c, \tag{12}$$

$$\sum_{cert_i \in AllCerts_{N\text{-}path}} t_h = n_{total}^{N\text{-}path} \cdot t_h. \tag{13}$$

Furthermore, $\sum_{cert_i \in AllCerts_{N\text{-}path}} S(cert_i)/n_{total}^{N\text{-}path}$ is the average size of all the certificates on the *N-path* and this average value is denoted as $S(cert_{avg}^{N\text{-}path})$. Under these considerations, Eq. (11) becomes:

$$T_{N\text{-}path}(h, c) = n_{cc}^{N\text{-}path} \cdot t_c + n_{total}^{N\text{-}path} \cdot t_h + n_{total}^{N\text{-}path} \cdot \frac{S(cert_{avg}^{N\text{-}path})}{\lambda_h}. \tag{14}$$

In order to examine the speed-up factor of the NPKI certificate path verification over the classical certificate path verification, the classical certificate path verification time should also be formulated. All of the certificates on a classical certificate path are classical certificates that must be verified cryptographically. Therefore, a classical certificate path can be considered as a special NPKI certificate path where there are no nested certificates. That is, for a classical certificate path, *C-path*:

$$SubjectCerts_{C\text{-}path} = \{\}, \tag{15}$$

$$CryptCerts_{C\text{-}path} = AllCerts_{C\text{-}path}. \tag{16}$$

From Eqs. (15) and (16) it can be deduced that:

$$n_{nc}^{C\text{-}path} = 0, \tag{17}$$

$$n_{cc}^{C\text{-}path} = n_{total}^{C\text{-}path}. \tag{18}$$

Under these considerations, the verification time of *C-path*, $T_{C\text{-}path}(h, c,)$, is given by Eq. (19).

$$T_{C\text{-}path}(h, c,) = n_{total}^{C\text{-}path} \cdot t_c + n_{total}^{C\text{-}path} \cdot t_h + n_{total}^{C\text{-}path} \cdot \frac{S(cert_{avg}^{C\text{-}path})}{\lambda_h}. \tag{19}$$

$f_{path}(h, c, N\text{-}path)$ is the speed-up factor for the verification of *N-path* over the verification of *C-path* in terms of the time spent for the verifications. $f_{path}(h, c, N\text{-}path)$ must be analyzed where the path lengths and the average certificate sizes of the *N-path* and *C-path* are the same. Moreover, it is also assumed that the same hash algorithm h and the same cryptosystem c are used in both *N-path* and *C-path*. These assumptions imply that:

$$n_{total}^{C\text{-}path} = n_{total}^{N\text{-}path}, \tag{20}$$

$$S(cert_{avg}^{N\text{-}path}) = S(cert_{avg}^{C\text{-}path}). \tag{21}$$

Under these considerations, $f_{path}(h, c, N\text{-}path)$ is given by Eq. (22). The derivation is as follows:

$$\begin{aligned}
f_{path}(h, c, N\text{-}path) &= \frac{T_{C\text{-}path}(h, c)}{T_{N\text{-}path}(h, c)} = \frac{n_{total}^{C\text{-}path} \cdot t_c + n_{total}^{C\text{-}path} \cdot t_h + n_{total}^{C\text{-}path} \cdot \dfrac{S(cert_{avg}^{C\text{-}path})}{\lambda_h}}{n_{cc}^{N\text{-}path} \cdot t_c + n_{total}^{N\text{-}path} \cdot t_h + n_{total}^{N\text{-}path} \cdot \dfrac{S(cert_{avg}^{N\text{-}path})}{\lambda_h}} \\
&= \frac{n_{total}^{N\text{-}path} \cdot t_c + n_{total}^{N\text{-}path} \cdot t_h + n_{total}^{N\text{-}path} \cdot \dfrac{S(cert_{avg}^{N\text{-}path})}{\lambda_h}}{n_{cc}^{N\text{-}path} \cdot t_c + n_{total}^{N\text{-}path} \cdot t_h + n_{total}^{N\text{-}path} \cdot \dfrac{S(cert_{avg}^{N\text{-}path})}{\lambda_h}} \\
&= \frac{n_{total}^{N\text{-}path} \cdot t_c + n_{total}^{N\text{-}path} \cdot t_h + n_{total}^{N\text{-}path} \cdot \dfrac{S(cert_{avg}^{N\text{-}path})}{\lambda_h}}{(n_{total}^{N\text{-}path} - n_{nc}^{N\text{-}path}) \cdot t_c + n_{total}^{N\text{-}path} \cdot t_h + n_{total}^{N\text{-}path} \cdot \dfrac{S(cert_{avg}^{N\text{-}path})}{\lambda_h}} \\
&= \frac{n_{total}^{N\text{-}path} \cdot t_c + n_{total}^{N\text{-}path} \cdot t_h + n_{total}^{N\text{-}path} \cdot \dfrac{S(cert_{avg}^{N\text{-}path})}{\lambda_h}}{n_{total}^{N\text{-}path} \cdot t_c + n_{total}^{N\text{-}path} \cdot t_h + n_{total}^{N\text{-}path} \cdot \dfrac{S(cert_{avg}^{N\text{-}path})}{\lambda_h} - n_{nc}^{N\text{-}path} \cdot t_c}
\end{aligned}$$

$$
\begin{aligned}
&= \frac{1}{1 - \dfrac{n_{nc}^{N\text{-}path} \cdot t_c}{n_{total}^{N\text{-}path} \cdot t_c + n_{total}^{N\text{-}path} \cdot t_h + n_{total}^{N\text{-}path} \cdot \dfrac{S(cert_{avg}^{N\text{-}path})}{\lambda_h}}} \\
&= \frac{1}{1 - \dfrac{n_{nc}^{N\text{-}path} \cdot t_c}{n_{total}^{N\text{-}path} \cdot \left(t_c + t_h + \dfrac{S(cert_{avg}^{N\text{-}path})}{\lambda_h} \right)}} \\
&= \frac{1}{1 - \dfrac{n_{nc}^{N\text{-}path}}{n_{total}^{N\text{-}path}} \cdot \dfrac{1}{\left(1 + \dfrac{t_h}{t_c} + \dfrac{S(cert_{avg}^{N\text{-}path})}{\lambda_c \cdot t_c} \right)}}.
\end{aligned} \tag{22}
$$

The values of t_h, t_c, λ_h and $S(cert_{avg}^{N\text{-}path})$ are all positive and finite. Therefore,

$$\frac{1}{1 + \dfrac{t_h}{t_c} + \dfrac{S(cert_{avg}^{N\text{-}path})}{\lambda_h \cdot t_c}}$$

is between, but excluding, 0 and 1. Moreover, $n_{nc}^{N\text{-}path} < n_{total}^{N\text{-}path}$. Thus, the denominator of Eq. (22) is also between, but excluding, 0 and 1. Therefore, provided that $n_{nc}^{N\text{-}path}$ is not zero, the relative speed-up factor $f_{path}(h, c, N\text{-}path)$, which is given by Eq. (22), is always larger than 1. This conclusion proves the following theorem.

Theorem 2. *Let N-path be a NPKI certificate path with at least one nested certificate on it. Let C-path be a classical certificate path with the same number of certificates as the N-path. Assuming that the average certificate sizes, the public key cryptosystems and the hash functions used are the same in both paths, the verification of the N-path is always computationally more efficient than the verification of the C-path*.

3.3. Analysis of the NPKI certificate path verification speed-up factor

The speed-up factor $f_{path}(h, c, N\text{-}path)$ given by Eq. (22) is dependent on three parameters. These are the following:

(1) The cryptosystem–hash function pair. The t_h, t_c and λ_h values are determined depending on the cryptosystem and hash function used. t_h is inversely, t_c and λ_h are directly related to the speed-up factor.

(2) The degree of nested certification. This value, which is denoted as $n_{nc}^{N\text{-}path}/n_{total}^{N\text{-}path}$ in Eq. (22), is the fraction of the number of nested certificates over the number of all certificates of the *N-path*. The degree of nested certification is a value between 0 and 1, but it cannot be 1 since there must be at least one classical certificate on the *N-path*. If it is 0, then it means there is no nested certificate on the

path, therefore the path becomes a classical certificate path. The degree of nested certification is directly related to the speed-up factor.

(3) The average certificate size $S(cert_{avg}^{N\text{-}path})$. The average certificate size is inversely related to the speed-up factor.

The NPKI certificate path verification speed-up factor is also relevant with the subject certificate verification speed-up factor. As an example case, the behavior of the both speed-up factors will be analyzed comparatively for the case where h = SHA-1 and c = RSA-512. The behavior of the speed-up factors $f(\text{SHA-1, RSA512, } cert_{avg}^{N\text{-}path})$ and $f_{path}(\text{SHA-1, RSA512, } N\text{-}path)$ versus the average certificate size, $S(cert_{avg}^{N\text{-}path})$, and the degree of nested certification, $n_{nc}^{N\text{-}path}/n_{total}^{N\text{-}path}$, is given in Fig. 6. In this figure, the upper curve, with all '*' characters, is the curve of $f(\text{SHA-1, RSA512, } cert_{avg}^{N\text{-}path})$ versus $S(cert_{avg}^{N\text{-}path})$ and it is drawn by using Eq. (3). The mesh below this curve is the mesh for $f_{path}(\text{SHA-1, RSA512, } N\text{-}path)$ versus $S(cert_{avg}^{N\text{-}path})$ and $n_{nc}^{N\text{-}path}/n_{total}^{N\text{-}path}$, which is drawn by using Eq. (22). The $t_{\text{SHA-1}}$, t_{RSA512} and the $\lambda_{\text{SHA-1}}$ values of Table 1 are used. As can be seen from Fig. 6, the $f_{path}(\text{SHA-1, RSA512, } N\text{-}path)$ values are always less than the $f(\text{SHA-1, RSA512, } cert_{avg}^{N\text{-}path})$ values for all of the average certificate sizes. Moreover, $f_{path}(\text{SHA-1, RSA512, } N\text{-}path)$ values approach $f(\text{SHA-1, RSA512, } cert_{avg}^{N\text{-}path})$ values as $n_{nc}^{N\text{-}path}/n_{total}^{N\text{-}path}$ approaches to 1.

Moreover, the change in speed-up factor $f_{path}(\text{SHA-1, RSA512, } N\text{-}path)$ is more sensitive to the degree of nested certification as compared to average certificate size. For example, $f_{path}(\text{SHA-1, RSA512, } N\text{-}path)$ is 5.13, where average certificate size is 5000 and the degree of nested certification is 0.9. For the same average certificate size, $f_{path}(\text{SHA-1, RSA1024, } N\text{-}path)$ becomes 2.68 and 1.82, where the degree of nested certification is 0.7 and 0.5, respectively. On the other hand, in the case where the degree of nested certification is 0.9, $f_{path}(\text{SHA-1, RSA1024, } N\text{-}path)$ becomes 5.60 and 6.18, where the average certificate size is 4000 and 3000, respectively.

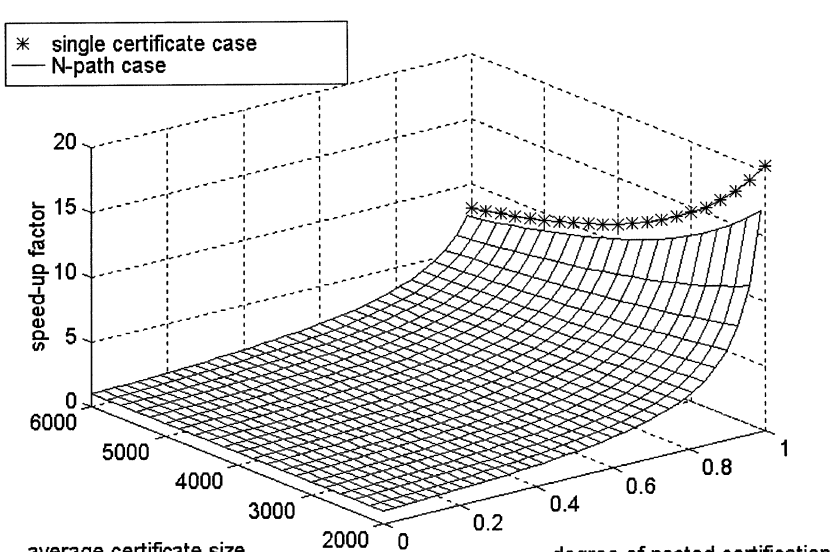

Fig. 6. Single subject certificate verification and the NPKI certificate path verification speed-up factors, where SHA-1 and RSA with 512-bit modulus are used.

3.4. Analytical comparison of the subject certificate verification and NPKI certificate path verification speed-up factors

As graphically discussed above, the NPKI certificate path verification speed-up factor is always less than the subject certificate verification speed-up factor for all certificate sizes and for a given cryptosystem–hash function pair. In this subsection, this conclusion will be proven analytically. Moreover, it will also be shown that the speed-up factor for the NPKI certificate path verification only approaches to the corresponding subject certificate verification speed-up factor.

In order to compare the speed-up factor for the verification of a certificate *cert* as a subject certificate with the speed-up factor for the verification of a NPKI certificate path *N-path*, the same cryptosystem c and hash function h are assumed to be used in both cases. Moreover, the size of *cert* is assumed to be the same as the average certificate size of *N-path*. That means, $S(cert)$ is assumed to be equal to $S(cert_{avg}^{N\text{-}path})$. The ratio of the speed-up factor for the verification of a certificate *cert* as a subject certificate over the speed-up factor for the verification of a NPKI certificate path *N-path* is denoted as $r(h, c, N\text{-}path)$. Under these assumptions, $r(h, c, N\text{-}path)$ is given by Eq. (23). The derivation is as follows:

$$
\begin{aligned}
r(h,c,N\text{-}path) &= \frac{f(h,c,cert)}{f_{path}(h,c,N\text{-}path)} = \frac{1+\dfrac{1}{\dfrac{t_h}{t_c}+\dfrac{S(cert)}{\lambda_h\cdot t_c}}}{\dfrac{1}{1-\dfrac{n_{nc}^{N\text{-}path}}{n_{total}^{N\text{-}path}}\cdot\dfrac{1}{\left(1+\dfrac{t_h}{t_c}+\dfrac{S(cert_{avg}^{N\text{-}path})}{\lambda_h\cdot t_c}\right)}}} \\
&= \frac{1+\dfrac{1}{\dfrac{t_h}{t_c}+\dfrac{S(cert_{avg}^{N\text{-}path})}{\lambda_h\cdot t_c}}}{\dfrac{1}{1-\dfrac{n_{nc}^{N\text{-}path}}{n_{total}^{N\text{-}path}}\cdot\dfrac{1}{\left(1+\dfrac{t_h}{t_c}+\dfrac{S(cert_{avg}^{N\text{-}path})}{\lambda_h\cdot t_c}\right)}}} = \frac{\dfrac{t_h}{t_c}+\dfrac{S(cert_{avg}^{N\text{-}path})}{\lambda_h\cdot t_c}+1}{\dfrac{t_h}{t_c}+\dfrac{S(cert_{avg}^{N\text{-}path})}{\lambda_h\cdot t_c}+1-\dfrac{n_{nc}^{N\text{-}path}}{n_{total}^{N\text{-}path}}} \\
&= \frac{\dfrac{1}{\dfrac{t_h}{t_c}+\dfrac{S(cert_{avg}^{N\text{-}path})}{\lambda_h\cdot t_c}}}{\dfrac{1}{\dfrac{t_h}{t_c}+\dfrac{S(cert_{avg}^{N\text{-}path})}{\lambda_h\cdot t_c}+1-\dfrac{n_{nc}^{N\text{-}path}}{n_{total}^{N\text{-}path}}}} = \frac{\dfrac{t_h}{t_c}+\dfrac{S(cert_{avg}^{N\text{-}path})}{\lambda_h\cdot t_c}+1-\dfrac{n_{nc}^{N\text{-}path}}{n_{total}^{N\text{-}path}}}{\dfrac{t_h}{t_c}+\dfrac{S(cert_{avg}^{N\text{-}path})}{\lambda_h\cdot t_c}}.
\end{aligned}
\tag{23}
$$

As discussed in Section 3.2, the degree of nested certification $n_{nc}^{N\text{-}path}/n_{total}^{N\text{-}path}$ is between, but excluding,

0 and 1. Therefore, as Eq. (23) implies, $r(h, c, N\text{-}path)$ is always greater than 1. The implication of this conclusion is as follows.

$$r(h, c, N\text{-}path) > 1 \;\Rightarrow\; \frac{f(h, c, cert)}{f_{path}(h, c, N\text{-}path)} > 1 \;\Rightarrow\; f(h, c, cert) > f_{path}(h, c, N\text{-}path).$$

That means, the speed-up factor for subject certificate verification is always greater than the speed-up factor for NPKI certificate path verification. In other words, the speed-up factor for the NPKI certificate path verification cannot reach the speed-up factor for the subject certificate verification. This conclusion proves the following theorem.

Theorem 3. *Let N-path be a NPKI certificate path with at least one nested certificate on it. Let cert be a certificate path of which the size is equal to the average certificate size of N-path. Assuming that the same public key cryptosystems and the hash functions are used in both cases, the speed-up factor for the verification of N-path is always less than the speed-up factor for the verification of cert.*

Although the speed-up factor is smaller for NPKI certificate paths, there is always improvement as proven by Theorem 2 and this speed-up factor is greater for the paths with large degree of nested certification. As discussed in Section 3.3, the speed-up factors for the NPKI certificate path verification approach to the speed-up factors for subject certificate verification as the degree of nested certification approaches to 1. This intuition is proven here. In order to prove it, the limit of the NPKI certificate path speed-up factor will be calculated as the degree of nested certification approaches to 1.

$$\begin{aligned}
\lim_{\frac{n_{nc}^{N\text{-}path}}{n_{total}^{N\text{-}path}} \to 1} f_{path}(h, c, N\text{-}path) &= \lim_{\frac{n_{nc}^{N\text{-}path}}{n_{total}^{N\text{-}path}} \to 1} \frac{1}{1 - \frac{n_{nc}^{N\text{-}path}}{n_{total}^{N\text{-}path}} \cdot \frac{1}{\left(1 + \frac{t_h}{t_c} + \frac{S(cert_{avg}^{N\text{-}path})}{\lambda_h \cdot t_c}\right)}} \\
&= \frac{1}{1 - 1 \cdot \frac{1}{\left(1 + \frac{t_h}{t_c} + \frac{S(cert_{avg}^{N\text{-}path})}{\lambda_h \cdot t_c}\right)}} = \frac{1}{\frac{\left(1 + \frac{t_h}{t_c} + \frac{S(cert_{avg}^{N\text{-}path})}{\lambda_h \cdot t_c}\right) - 1}{\left(1 + \frac{t_h}{t_c} + \frac{S(cert_{avg}^{N\text{-}path})}{\lambda_h \cdot t_c}\right)}} \\
&= \frac{\frac{t_h}{t_c} + \frac{S(cert_{avg}^{N\text{-}path})}{\lambda_h \cdot t_c} + 1}{\frac{t_h}{t_c} + \frac{S(cert_{avg}^{N\text{-}path})}{\lambda_h \cdot t_c}} = 1 + \frac{1}{\frac{t_h}{t_c} + \frac{S(cert_{avg}^{N\text{-}path})}{\lambda_h \cdot t_c}}.
\end{aligned} \tag{24}$$

The limit given by Eq. (24) is nothing but the speed-up factor for the verification of $cert_{avg}^{N\text{-}path}$ as a subject certificate. Substituting this speed-up factor as given by Eq. (3), Eq. (24) becomes the following:

$$\lim_{\frac{n_{nc}^{N\text{-}path}}{n_{total}^{N\text{-}path}} \to 1} f_{path}(h, c, N\text{-}path) = 1 + \frac{1}{\frac{t_h}{t_c} + \frac{S(cert_{avg}^{N\text{-}path})}{\lambda_h \cdot t_c}} = f(h, c, cert_{avg}^{N\text{-}path}). \tag{25}$$

As can be seen from Eq. (25), the speed-up factor for the NPKI certificate path verification approaches to the speed-up factor for a single subject certificate verification, as the degree of nested certification approaches to 1. This conclusion proves the following theorem.

Theorem 4. *Let N-path be a NPKI certificate path and cert be a certificate, of which the size is the average certificate size of N-path. The speed-up factor for the verification of N-path approaches to the speed-up factor for the verification of cert as a subject certificate, while the degree of nested certification of N-path approaches to 1.*

4. Conclusions, future and related work

The public key cryptosystem operations are secure, but computationally inefficient processes. Classical certificate systems employ public key cryptosystems. Therefore, the classical certificate verification is inefficient. This inefficiency becomes worse for the classical certificate paths, in which several classical certificate verifications are performed. The nested certificates [5] are used to verify their subject certificates without employing public key cryptosystem operations. The subject certificates can be classical or other nested certificates. Moreover, nested certificates can be used widely in the Nested Certificate based Public Key Infrastructure (NPKI) [6,5]. The NPKI certificate paths contain both classical and nested certificates.

In this paper, the analytical formulation of the relative speed-up factor of the subject certificate verification time over the cryptographic verification time is derived. Similar formulation for the NPKI certificate path verification time over the classical certificate path verification time is also given in this paper. Based on these formulations, it is proven that the verification of a certificate as a subject certificate is always computationally more efficient than the cryptographic certificate verification. Similarly, it is also proven that the usage of the nested certificates in the NPKI certificate paths also improves the computational efficiency of the certificate path verification all the time. Furthermore, the graphical analyses and some numeric examples, which are based on those formulations, are provided. From these analyses, it has been concluded that the subject certificate verification performs about 10 to 1220 times faster than the cryptographic verification for the typical certificate sizes between 3000 and 4500 bits. The factors that effect the speed-up factor are the certificate sizes, hash algorithms and the public key cryptosystems employed in the verifications. The NPKI certificate paths include both classical and nested certificates. The speed-up factor of the NPKI certificate path verification over the classical certificate path verification is directly related to the number of nested certificates on the NPKI certificate path. Because of the classical certificates on the NPKI certificate paths, the speed-up for the NPKI certificate path case is less than the single subject certificate case. However, the improvement is still noteworthy. For an example case of RSA512–SHA-1 algorithm pair, the NPKI certificate path verification speed-up factor is 2.68 with 70% nested certification.

In order to benefit from the efficient verification advantage of NPKI certificate paths, large number of nested certificates must be issued in NPKI by certification authorities. This causes *nested certification overhead* for the authorities. More nested certification means more efficiency in the NPKI certificate paths. Therefore, there is a trade-off between the NPKI certificate path verification speed-up factor and the nested certification overhead. The way of nested certification in NPKI is important in order to analyze this trade-off. Such an analysis, which uses an extensive and enforced nested certification model

over a balanced tree-shaped PKI, can be found in [5]. In this model, each authority, say *A*, is enforced to issue nested certificates to all of the certificates (nested or classical) that were issued by the CAs/NCAs that *A* has certified. In this way, the PKI turns out to be NPKI and the number of nested certificates is maximized for all of the NPKI certificate paths. Consequently, the speed-up factor is also maximized. However, this model requires a large number nested certificates to issue.

As a future work, the above analysis can be extended to consider arbitrary-shaped topologies and more flexible nested certification policies in NPKI. In flexible nested certification cases, the average efficiency improvement and the nested certification overhead are expected to be less than the enforced nested certification case. However, there is still efficiency improvement as compared to pure classical certificate systems.

Acknowledgements

This work has been supported by The Scientific and Technical Research Council of Turkey (TUBITAK) Grant EEEAG-237 and by Turkish State Planning Organization (DPT) Grant 96K120490.

References

[1] A. Bosselaers, Even faster hashing on the Pentium, Presented in the rump session of Eurocrypt '97, Konstanz, May 11–15, 1997, available from ftp://ftp.esat.kuleuven.ac.be/COSIC/bosselae/pentiumplus.ps.gz
[2] A. Bosselaers, R. Govaerts, J. Vandewalle, Fast hashing on the Pentium, in: N. Koblitz (Ed.), Advances in Cryptology, Proceedings of Crypto'96, LNCS 1109, Springer, Berlin, 1996, pp. 298–312.
[3] National Institute of Standards and Technology (NIST), Federal Information Processing Standard (FIPS) PUB 186, Digital Signature Standard (DSS), U.S. Department of Commerce, 19 May 1994.
[4] C. Kaufman, R. Perlman, M. Speciner, Network security: Private Communication in a Public World, Prentice-Hall, Englewood Cliffs, NJ, 1995, pp. 101–127.
[5] A. Levi, Design and performance evaluation of the nested certification scheme and its applications in public key infrastructures, Ph.D. Thesis, Department of Computer Engineering, Boğaziçi University, Istanbul, Turkey, 1999 (in preparation).
[6] A. Levi, M.U. Çağlayan, NPKI: nested certificate based public key infrastructure, in: Advances in Computer and Information Sciences '98 — Proceedings of the Thirteenth International Symposium on Computer and Information Sciences, ISCIS XIII, Concurrent Systems Engineering Series 53, IOS Press, Turkey, 1998, pp. 397–404.
[7] R. Rivest, The MD5 Message Digest Algorithm, RFC 1321, April 1992.
[8] R. Rivest, A. Shamir, L. Adleman, A method for obtaining digital signatures and public key cryptosystems, Commun. ACM 21 (2) (1978) 120–126.
[9] GMD Security Technology, SECUDE — A General Purpose Security Toolkit, 1998, http://www.darmstadt.gmd.de/secude/
[10] National Institute of Standards and Technology (NIST), Federal Information Processing Standard (FIPS) PUB 180-1, Secure Hash Standard (SHS), U.S. Department of Commerce, Washington, 1995.
[11] M.J. Wiener, Performance comparison of public-key cryptosystems, RSA Laboratories' Cryptobytes 4 (1) (1998) 1–5.
[12] ITU-T Recommendation X.509, ISO/IEC 9594-8, Information Technology — Open Systems Interconnection — The Directory: Authentication Framework, 1997.
[13] P. Zimmermann, PGP User's Guide Volume I: Essential Topics, available with free PGP software from http://www.pgpi.com/download
[14] P. Zimmermann, PGP User's Guide Volume II: Special Topics, available with free PGP software from http://www.pgpi.com/download

Albert Levi received the B.Sc. and M.Sc. degrees from Department of Computer Engineering, Boğaziçi University, Istanbul, Turkey in 1991 and 1993, respectively. He is currently a Ph.D. student at the same department. He is a teaching and research assistant in the Department of Computer Engineering, Boğaziçi University, Istanbul, Turkey, since 1991. His current research interests are computer and network security, cryptography, certification systems and electronic commerce. He is a member of IEEE and ACM.

Mehmet Ufuk Çağlayan received the BSEE and MSCS degrees from Middle East Technical University, Ankara, Turkey, in 1973 and 1975 respectively, and a Ph.D. degree from Northwestern University, Evanston, IL, in 1981. Dr. Çağlayan was a faculty member in DePaul University, Chicago, IL, Northwestern University, Evanston, IL and University of Petroleum and Minerals, Dhahran, Saudi Arabia. He served as a computer scientist in BASF AG, Ludwigshafen, Germany. Dr. Çağlayan is currently serving as a full professor in the Department of Computer Engineering, Boğaziçi University, Istanbul, Turkey. His current research interests are in the areas of high speed computer networks, parallel and distributed systems, operating systems and performance modeling. Dr. Çağlayan is a member of IEEE, ACM, Turkish Informatics Society, Turkish Association of Electrical Engineers and Turkish Open System Users Group.