A Novel Signature Scheme and its application to Image Authentication

Ken-ichi Sakina
QRTechnology LLC
1-2-3-109 Miwamidoriyama, Machida, Tokyo

ABSTRACT
In this paper, a novel digital signature scheme called the RS-signature scheme is proposed, which combines Reed-Solomon (RS) codes and the Elliptic Curve Digital Signature Algorithm (ECDSA). In conventional signature schemes, embedding a digital signature of the original file into the file itself changes its hash value, and this makes it impossible to generate a signature-embedded file in which the file itself is the target of the signature. However, by employing the proposed scheme, it becomes possible to generate such a signature-embedded file.
In this paper, the theory and algorithm of the proposed scheme are first presented. Next, to demonstrate its validity, the scheme was applied to image authentication experiments, yielding results that show practical-level authentication speed and reliability. The RS signature scheme can, in principle, be applied to a wide variety of file formats. To demonstrate its extensibility, the scheme was also applied to experiments using HTML files, and satisfactory results were obtained.

General Terms
Authentication, Coding Theory, Digital Signature, Security, Image, HTML

Keywords
Image authentication, Digital signature, Reed–Solomon code, ECDSA. HTML Authentication

1. INTRODUCTION
In recent years, with the advancement of AI, methods of forgery and tampering have also become more sophisticated, making it an urgent priority to enhance the reliability of digital data. As is well known, digital signature technology includes a variety of signature schemes such as DSA, ECDSA, and RSA [1][2], which play a crucial role in ensuring the integrity and authenticity of digital content.

Here, we consider a signature-embedded file, which is important from the viewpoint of usability and security, in which the file itself is the target of a signature. Such a file cannot be generated using conventional signature schemes, because embedding a signature into the original file changes its hash value and thus invalidates the signature. Therefore, they typically require either a separate file to store the signature (i.e. the signature file) or apply a partial signature that covers only a specific portion of the original file. In the former case, the separation of the original file and the signature file introduces not only security vulnerabilities but also practical issues during transmission or verification. For example, when both the original file and the signature file need to be transmitted, such as via email, they are often bundled into a ZIP archive for convenience. However, due to security policies or system restrictions, the recipient's system may reject ZIP file attachments [3]. In the latter case, partial signatures may introduce security vulnerabilities. For example, in the case of XML documents [4] or PDF files, such partial signatures are vulnerable to attacks such as the Wrapping Attack [5][6][7].

To address these issues, this paper proposes a novel digital signature scheme, called the RS-signature scheme, which combines Reed–Solomon (RS) codes [8] with digital signatures. By using the RS-signature scheme, it becomes possible to generate such signature-embedded files. As described in Section 4, the RS-signature scheme, first, RS-encodes the signer's information (such as the signer's name, affiliation, and signing date), and embeds or appends the resulting RS codes into the original file. Then, using an exclusive OR operation, the corresponding signature for that file is embedded into the RS-codes within the file, thereby generating a signature-embedded file in which the file itself becomes the target of the signature.

The RS-signature scheme is similar to previously published code-based signature schemes [9]–[11] in that it uses error-correcting codes; however, it differs from them in the following respects:
- Code-based signature schemes cannot generate a signature-embedded file in which the file itself is the target of a signature, but the RS-signature scheme makes it possible.
- Code-based signature schemes do not utilize a public key infrastructure (PKI), but the RS-signature scheme makes it possible.

In principle, the RS-signature scheme can generate signature-embedded files for virtually any type of file, including image files, PDF documents, MS Word documents, MS Excel spreadsheets, and XML files. Furthermore, one of the key features of the RS-signature scheme is that the signer's information is displayed during signature verification. One particularly important application of the RS-signature scheme is image authentication. This generates a signed image file by replacing the LSBs of the original image file with an RS-code and embedding the file's digital signature into the RS-code using an exclusive OR. Image authentication experiments were conducted using this approach, and the expected results were obtained.

The rest of this paper is organized as follows: Section 2 and 3 review ECDSA and RS-code, respectively. Section 4 presents the proposed RS-signature scheme in detail. Section 5 provides experiments using RS-signature scheme for image-based certificates. Section 6 demonstrates the extensibility of RS-signature scheme through experiments using HTML files. Section 7 describes a security discussion of the proposed scheme, and finally, the concluding remarks are provided in Section 8

2. ECDSA

The ECDSA (Elliptic Curve Digital Signature Algorithm) is a widely used digital signature algorithm based on the algebraic

structure of elliptic curves over finite fields. It is a variant of the DSA (Digital Signature Algorithm), adapted to work within the elliptic curve cryptographic framework, offering equivalent security with significantly smaller key sizes. The signing and verification processes in ECDSA, which will be used in later sections, are briefly described below.

(a) Domain parameters

Let p be the prime number, and let F_p be the finite field of order p. An elliptic curve over F_p, denoted by E/F_p, is defined by

$$y^2 = x^3 + ax + b, \tag{1}$$

where $a, b \in F_p$.

Let G be the base point on E/F_p with prime order l, and let the hash function be

$$H: \{0,1\}^* \to \{0,1\}^{|l|-1}. \tag{2}$$

The domain parameters are then given by

$$D = \{a, b, p, l, G, H\}. \tag{3}$$

(b) key pair

First, choose a non-zero random number X from the set $Z_l^* = \{1, \cdots . l-1\}$, and let the number X be the private key. The public key Q is then computed by

$$Q = XG \tag{4}$$

where Q is the point on E/F_p.

(c) Signature generation

The algorithm consists of the following steps:

- Let the message that a signer wants to sign be M.
- Generate non zero random number $r \in Z_l^*$ and obtain x-component c of rG, that is

$$c = (rG)_x \quad mod\ l \tag{5}$$

- Compute the equation

$$s = r^{-1}(H(M) + cX) \qquad mod\ l \tag{6}$$

-The digital signature σ for M is given by

$$\sigma = (c, s). \tag{7}$$

(d) Signature verification

The verifier checks the signature σ for M as follows:

- Compute following equations (8)~(10):

$$u = s^{-1} \quad mod\ l \tag{8}$$

$$v = H(M)u \quad mod\ l \tag{9}$$

$$vG + cuQ = (x', y') \tag{10}$$

- If $x' = c \ \ (mod\ l)$, the signature σ is valid, otherwise is invalid.

3. RS (REED-SOLOMON) CODES

RS-codes are one of the well-known error correcting codes and provide strong error correction capabilities. This section provides a brief overview of RS-codes, which will serve as the basis for the subsequent discussions.

3.1 Generation of RS-codes

We first consider m-th extension field F_q of the finite field $F_2 = \{0,1\}$, where $q = 2^m$. Let $f(x)$ be a minimum polynomial of degree m over F_2, and let α be a primitive root of $f(x)$. The extension field F_q can then be represented as

$$F_q = \{0, 1, \alpha, \cdots . \alpha^{2^m-2}\}, \tag{11}$$

where each element in F_q corresponds to an m-bit unit, commonly referred to as a *symbol*.

Now consider (n, k)RS-codes over F_q, where n is the codelength (in symbols) and k is the number of information symbols. Let t be the symbol error-correcting capability of the RS-codes. Any codeword $\boldsymbol{w}$ of (n, k)RS-codes can be represented in vector form as follows.

$$\boldsymbol{w} = (w_0, w_1, \cdots, w_{n-1}) \in F_q^n\ , \tag{12}$$

where $F_q^n \equiv F_q \times F_q \times \cdots \times F_q$ (taken n times) and each component $w_i \in F_q$ for $i = 0,1, \cdots, n$-1. The corresponding code polynomial $w(x) \in F_q[x]$ of degree less than n is given by

$$w(x) = w_0 + w_1 x + \cdots + w_{n-1}x^{n-1}. \tag{13}$$

The RS-codes are generated using a generator polynomial $g(x)$ and an information polynomial $a(x)$ as follows:

$$w(x) = a(x)g(x), \tag{14}$$

where

$$g(x) = \prod_{i=1}^{n-k}(x - \alpha^i), \tag{15}$$

$$a(x) = \sum_{i=0}^{k-1} a_i x^i\ ,\ a_i \in F_q\ . \tag{16}$$

Note that the set of all vectors $\boldsymbol{w} = (w_0, w_1, \cdots, w_{n-1})$ satisfying (14) forms k-dimensional subspace of F_q^n. In general, $(n. k, d)$RS-codes have a minimum Hamming distance $d = n - k + 1$, making it a Maximum Distance Separable (MDS) code. The relationship among *n*, *k*, and *t* is determined by the number of consecutive roots of $w(x)$, $n - k$, which is expressed as $k = n - 2t$.

3.2 Decoding of RS-codes

We consider $(n, k, 2t + 1)$ RS-codes with generator polynomial $g(x)$ and information polynomial $a(x)$ as defined in (15) and (16), respectively. The codes can correct up to t symbol errors.

In practical communication systems, the received words often differ from the transmitted codewords due to channel noise. In this subsection, we briefly outline the Euclidean decoding algorithm [12], which is used to recover the original codewords from received words. Let the set of error positions in the received word be

$$I = \{i_1, i_2, \cdots, i_s\}\ , \tag{17}$$

where $s \le t$. The error polynomial $e(x)$ is then given by

$$e(x) = e_{i_1}x^{i_1} + e_{i_2}x^{i_2} + \cdots + e_{i_s}x^{i_s} = \sum_{k=1}^{s} e_{i_k}x^{i_k} \tag{18}$$

Let $r(x)$ be the received polynomial. Then we have

$$r(x) = w(x) + e(x). \tag{19}$$

Since $w(\alpha^i) = 0\ \ (i = 1, \cdots, n-k)$ and $n - k = 2t$, the syndrome S_j can be expressed using (18) and (19) as follows:

$$S_j = r(\alpha^j) = e(\alpha^j) = \sum_{k=1}^{s} e_{i_k}(\alpha^j)^{i_k} \quad (j = 1, \cdots, 2t)\ . \tag{20}$$

We define the syndrome polynomial $S(z)$ and the error locator polynomial $\rho(z)$ as

$$S(z) = \sum_{j=1}^{2t}\sum_{k=1}^{s} e_{i_k}(\alpha^j)^{i_k}\, z^{j-1} \tag{21}$$

and

$$\rho(z) = \prod_{j=1}^{s}(1 - \alpha^{i_j}z)\ . \tag{22}$$

In general, the following identity holds:

$$\frac{e_{i_k}\alpha^{i_k}}{1-\alpha^{i_k}z} = \sum_{j=1}^{2t} e_{i_k}(\alpha^{i_k})^j\, z^{j-1} \quad (mod\ z^{2t}). \tag{23}$$

Substituting (23) into (21), we get

$$S(z) = \sum_{k=1}^{s} \frac{e_{i_k}\alpha^{i_k}}{1-\alpha^{i_k}z} \qquad (mod\ z^{2t}). \tag{24}$$

Multiplying (22) on both sides of (24) yields

$$\rho(z)S(z) + \varphi(z)z^{2t} = \eta(z), \tag{25}$$

where $\varphi(z)$ is a polynomial over F_q, and the error evaluator polynomial $\eta(z)$ is given by

$$\eta(z) = \sum_{k=1}^{s} e_{i_k}\alpha^{i_k}\prod_{j=1,\neq k}^{s}(1 - \alpha^{i_j}z)\ , \tag{26}$$

where $t \ge deg\ \rho(z) > deg\ \eta(z)$,.

Here, we define the sequence of functions $\{h_i(z)\}$ by the extended Euclidean algorithm, where the initial functions are given by $h_0(z) = z^{2t}$ and $h_1(z) = S(z)$, as follows:

$$h_{i-2}(z) = q_i(z)h_{i-1}(z) + h_i(z), \quad i = 2,3, \cdots, L, \tag{27}$$

where $\deg h_L(z) \le t - 1$.

From (27), $h_L(z)$ can be expressed in the following form:

$$h_L(z) = A(z)h_0(z) + B(z)h_1(z) \tag{28}$$

By comparing the equation (28) with (25), we obtain

$$\rho(z) = \gamma B(z)\ ,\ \ \eta(z) = \gamma h_L(z)\ . \tag{29}$$

where γ is chosen such that the constant term of $\rho(z)$ is equal to 1.

To compute the error values, we first evaluate $\eta(z)$ at $z = \alpha^{-i_k}$. From (26), we have

$$\eta(\alpha^{-i_k}) = \prod_{j=1,\neq k}^{s}(1-\alpha^{i_j}\alpha^{-i_k})\, e_{i_k}\alpha^{i_k}. \quad (30)$$

Next, differentiating $\rho(z)$ and evaluating at $z = \alpha^{-i_k}$ yields

$$\rho'(\alpha^{-i_k}) = -\alpha^{i_k}\prod_{j=1,\neq k}^{s}(1-\alpha^{i_j}\alpha^{-i_k}). \quad (31)$$

Taking the ratio of (30) and (31), we obtain the error values

$$e_{i_k} = -\frac{\eta(\alpha^{-i_k})}{\rho'(\alpha^{-i_k})} \quad (k = 1,2,\cdots,s), \quad (32)$$

which is known as Forney's formula. To determine the error positions, we evaluate $\rho(z)$ at $z = 1, \alpha, \cdots . \alpha^{2^m-2}$. The values of z for which $\rho(z) = 0$ correspond to the inverses α^{-i_k}, which reveal the error positions i_k.

Finally, having obtained both the error values and their corresponding positions, we reconstruct the original codeword $w(x)$ from the received word $r(x)$ using

$$w(x) = r(x) + e(x). \quad (33)$$

4. RS-SIGNATURE SCHEME

In the RS-signature scheme, it is possible to generate a signature-embedded file, in which the file itself is the target of the signature. The RS-signature is constructed by combining RS-codes with ECDSA. This section presents the methods for signature generation and verification in the RS-signature scheme, based on the foundations outlined in the preceding sections.

4.1 Generation Algorithm

[A] Generation of RS-code embedded file $\widetilde{\boldsymbol{M}}$.

We now consider $(n, k, 2t+1)$RS-codes over F_{2^8}. Let the information vector $\boldsymbol{a}$ corresponding to (16) be

$$\boldsymbol{a} = (a_0, a_1, \cdots, a_{k-1}), \quad a_i \in F_{2^8}. \quad (34)$$

Then the RS-codeword can be obtained from (14),(15) and (16) in the vector form as

$$\boldsymbol{w} = (w_0, w_1, \cdots, w_{n-1}) \in F_{2^8}^n. \quad (35)$$

In general, a digital file M can be expressed in the vector form as:

$$\boldsymbol{M} = (m_0, m_1, \cdots, m_{N-1}), \quad m_i \in F_{2^8}\ (i = 0,1,\cdots,N-1). \quad (36)$$

An RS-code-embedded file $\widetilde{\boldsymbol{M}}$ is obtained by embedding or appending RS-codeword $\boldsymbol{w}$ into $\boldsymbol{M}$. This is defined by the following equation:

$$\widetilde{\boldsymbol{M}} = \boldsymbol{w} \oplus_E \boldsymbol{M}, \quad (37)$$

where the symbol $\oplus_E$ denotes an embedding or appending operator. If $\boldsymbol{M}$ is a text file (for example, XML files), use it as the appending operation; if $\boldsymbol{M}$ is a binary file (for example, image files), use it as the embedding operation.

Note that the RS-code-embedded file $\widetilde{\boldsymbol{M}}$ is the target of RS-signature.

[B] Algorithm for generating RS-signed file $\widetilde{\boldsymbol{M}}_s$.

Let the *RS-signature* $\widetilde{\boldsymbol{w}}$ be defined as

$$\widetilde{\boldsymbol{w}} = \boldsymbol{\sigma}_B \oplus \boldsymbol{w}, \quad (38)$$

where $\oplus$ is the exclusive OR (XOR) operator and $\boldsymbol{\sigma}_B$ is a byte representation of digital signature σ.

The algorithm for generating the RS-signed file (i.e. RS-signature embedded file) $\widetilde{\boldsymbol{M}}_s$ is as follows:

- Generate the RS-codeword $\boldsymbol{w}$ for the information vector $\boldsymbol{a} = (a_0, a_1, \cdots, a_{k-1}) \in F_{2^8}^k$.
- Generate $\widetilde{\boldsymbol{M}}$, and let $\widetilde{\boldsymbol{M}}$ be the target file to be signed (see [A]).
- Compute the digital signature $\sigma = (c, s)$ for $\widetilde{\boldsymbol{M}}$ using ECDSA (see Section 2. (c)).
- Compute l_b, which is the number of bytes obtained by converting l (see eq. (2)) into a byte sequence.
- Express $\sigma = (c, s)$ in bytes, and denote them as $\sigma_{Bi} = (c_i, s_i)$, $c_i, s_i \in F_{2^8}$ ($i = 1, \cdots, l_b$), where l_b must satisfy $l_b \leq t/2$.
- Generate the RS-signature $\widetilde{\boldsymbol{w}}$ using (38).
- Finally, generate the *RS-signed file* $\widetilde{\boldsymbol{M}}_s$ by replacing $\boldsymbol{w}$ in $\widetilde{\boldsymbol{M}}$ with $\widetilde{\boldsymbol{w}}$:

$$\widetilde{\boldsymbol{M}}_s = \widetilde{\boldsymbol{w}} \oplus_E \boldsymbol{M}. \quad (39)$$

Figure 1 shows the algorithm for generating the RS-signed file $\widetilde{\boldsymbol{M}}_s$.

> As can be seen from (39), when $\widetilde{\boldsymbol{w}}$ in $\widetilde{\boldsymbol{M}}_s$ is decoded, the target file $\widetilde{\boldsymbol{M}}$ of the signature $\boldsymbol{\sigma}_B$ is automatically obtained. In this sense, $\widetilde{\boldsymbol{M}}_s$ itself can be regarded as the target file of $\boldsymbol{\sigma}_B$.

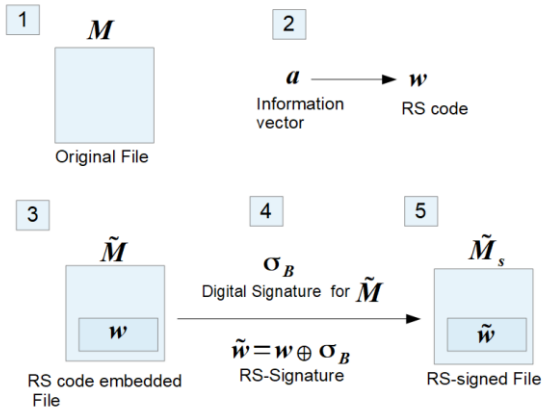

Figure 1: Generation Algorithm for RS-signed file $\widetilde{\boldsymbol{M}}_s$

4.2 Verification Algorithm

The algorithm to verify the RS-signed file $\widetilde{\boldsymbol{M}}_s$ are as follows:

- Decode the RS-signature $\widetilde{\boldsymbol{w}}$ in $\widetilde{\boldsymbol{M}}_s$ using the Euclidean decoding algorithm (see Section 3.2). Then the digital signature $\boldsymbol{\sigma}_B$ in (38) can be extracted as errors and the RS-codeword $\boldsymbol{w}$ can be obtained by $\boldsymbol{w} = \boldsymbol{\sigma}_B \oplus \widetilde{\boldsymbol{w}}$. Therefore, $\widetilde{\boldsymbol{M}}_s$ can be separated into $\widetilde{\boldsymbol{M}}$ and $\boldsymbol{\sigma}_B$:

$$\widetilde{\boldsymbol{M}}_s \rightarrow \{\widetilde{\boldsymbol{M}}, \boldsymbol{\sigma}_B\}. \quad (40)$$

- Recover the digital signature string $\sigma = (c, s)$ from $\boldsymbol{\sigma}_B$.
- Verify $\sigma = (c, s)$ for $\widetilde{\boldsymbol{M}}$ (see Section 2. (d)). If the digital signature σ is valid, the signer's information vector $\boldsymbol{a}$ is displayed.

Figure 2 shows the algorithm for verifying $\widetilde{\boldsymbol{M}}_s$

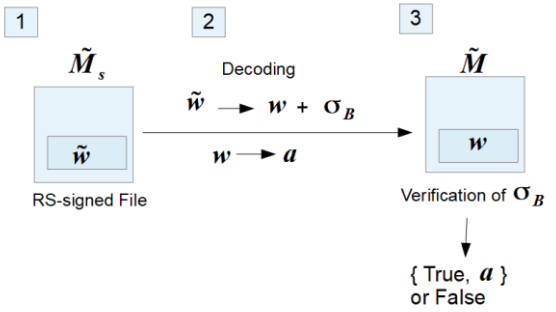

Figure 2: Verification algorithm for RS-signed file $\widetilde{\boldsymbol{M}}_s$

5. EXPERIMENT ON IMAGE AUTHENTICATION

The key features of the RS-signature scheme are its ability to generate a single signed file in which the file itself serves as the target of the signature, enable the verification of its authenticity without relying on external data, and display the signer's information which is a part of RS-code. As can be seen from the algorithms in Section 4, the RS-signature scheme is fundamentally applicable to a wide range of file formats, including PNG, XML, PDF, DOCX, and XLSX. To demonstrate the extensibility of the RS-signature scheme, Section 6 briefly discusses the application of RS-signature to HTML files. This section focuses on one of the useful applications, namely the authentication of image files, and presents experimental results of signing and verifying image-based digital certificates.

5.1 Experimental Environment

- Desktop PC
  CPU: Intel Core i5-1235U (10 cores, 1.3 GHz)
  Memory: 8 GB
  OS: Windows 11 Home 64-bit
  Java Environment: JDK 17.0.9.9 (HotSpot)

- Website to check $\widetilde{\boldsymbol{M}}_s$
  OS: CentOS Linux 7
  Servlet Container: Apache Tomcat 9.0.84
  Java Environment: Java HotSpot™ 64-Bit Server VM

5.2 Experimental Setup

The following experiment was conducted under the setting conditions described below:

- ECDSA Configuration:
  The NIST-recommended ECDSA was used with the secp160r1 parameter set [13], generating a 160-bit digital signature.
- Digital signature in bytes $\boldsymbol{\sigma}_B$:
  $\boldsymbol{\sigma_B} = (\sigma_{B1}, \cdots, \sigma_{B20})$,
  where $\sigma_{Bi} = (c_i, s_i) \in F_{2^8} \times F_{2^8}$ $(i = 1, \cdots, 20)$.
- λth RS-codeword $\boldsymbol{w_\lambda}$ $(\lambda = 1. \cdots, 5)$:
  Each codeword consists of parameters $(n, k, 2t + 1)$ = (42,16,27) in bytes.
- RS-codeword set $\boldsymbol{W}$:
  $\boldsymbol{W} = \{\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_5\}$, total bytes: 210 bytes.
- RS-signature set $\widetilde{\boldsymbol{W}}$:
  The digital signature $\boldsymbol{\sigma}_B$ is embedded into the codewords via XOR operation as follows:

$$\widetilde{\boldsymbol{W}} = \boldsymbol{\sigma}_B \oplus \boldsymbol{W} = ((\boldsymbol{\sigma}_{B1}, \cdots, \boldsymbol{\sigma}_{B4}) \oplus \boldsymbol{w_1}, \cdots, (\boldsymbol{\sigma}_{B17}, \cdots, \boldsymbol{\sigma}_{B20}) \oplus \boldsymbol{w_5})) = (\widetilde{w}_1, \cdots, \widetilde{\boldsymbol{w}}_5).$$

  For example:

$$(\boldsymbol{\sigma}_{B1}, \cdots, \boldsymbol{\sigma}_{B4}) \oplus \boldsymbol{w_1} = (c_1, s_1, \cdots, c_4, s_4) \oplus \boldsymbol{w_1} = \widetilde{\boldsymbol{w}}_1$$

- Error correction capacity:
  Each RS-codeword can correct up to 13 bytes errors, so the entire set $\boldsymbol{W}$ can correct a total of 65 bytes errors. In the experiment, a 320-bit ECDSA digital signature (160 bits × 2) was embedded into the codeword set $\boldsymbol{W}$.

5.3 Experimental Result

In this experiment, the image-based digital certificate $\boldsymbol{M}$, with a resolution of 426 × 640 pixels (420KB, bit depth 32) as shown in Figure 3, was used.

Experimental Steps

1. Generate $\boldsymbol{W}$, consisting of five RS-codewords, based on the following five information data fields (see Figure 4):
   ID: 00000001
   Title: Sample
   Signer: Dr. John Smith
   Issuer: QRTechnology
   Date: Auto input
2. Generate the RS-code embedded image $\widetilde{\boldsymbol{M}}$ using $\widetilde{\boldsymbol{M}} = \boldsymbol{W} \oplus_E \boldsymbol{M}$.
3. Generate the digital signature $\boldsymbol{\sigma}_B$ for $\widetilde{\boldsymbol{M}}$, (see Figure 5).
4. Generate the RS-signature set $\widetilde{\boldsymbol{W}}$ using $\widetilde{\boldsymbol{W}} = \boldsymbol{\sigma}_B \oplus \boldsymbol{W}$.
5. Generate the RS-signed digital certificate $\widetilde{\boldsymbol{M}}_s$, by replacing $\boldsymbol{W}$ in $\widetilde{\boldsymbol{M}}$ with $\widetilde{\boldsymbol{W}}$ (see Figure 6).
6. Verify the authenticity of $\widetilde{\boldsymbol{M}}_s$ using the process described in Section 4.2.

Verification Results

The desktop application was used to generate $\widetilde{\boldsymbol{M}}_s$, but in the verification, either the desktop application or the web application was used depending on the purpose. The experimental results are as follows:

– Visual quality: The Peak Signal-to-Noise Ratio (PSNR) between the RS-signed digital certificate $\widetilde{\boldsymbol{M}}_s$ and the original certificate $\boldsymbol{M}$ was 74.83 dB, indicating that embedding the RS code and signature caused no perceptible visual degradation.
– Verification speed: Verification on the desktop application took 0.386 seconds.
– Reliability: 200,000 repeated verifications on the desktop application produced zero authentication failures.
– Web verification: Verification output on the website is shown in Figure 7, which demonstrates that the authenticity judgment of $\widetilde{\boldsymbol{M}}_s$ is true, and the data (ID, title, signer, issuer, date) are coincident with the input data in Experimental Step 1 in Section 5.3.
- Validity of screenshots: Captured screenshot of $\widetilde{\boldsymbol{M}}_s$ was not valid in the signature verification.

These results confirm the high reliability of the proposed verification system. In these experiments, the SHA-256 cryptographic hash function was employed. However, since cryptographic hash functions are highly sensitive to minor distortions and geometric transformations in images, it is possible to use perceptual image hash functions such as pHash, aHash, or dHash [15][16] to mitigate these effects.

In a real-world scenario, the certificate authority (i.e., the issuer of $\widetilde{\boldsymbol{M}}_s$) is only required to maintain a digital certificate verification website. Any party possessing $\widetilde{\boldsymbol{M}}_s$ can verify its authenticity and integrity by uploading it to the verification website. Furthermore, supplementary information (e.g., learning history) can be embedded in the LSB bit plane of the certificate image, which is also the targt of the RS-signature.

Figure 3: Original digital certificate (sample) *M* (419KB)

```
W=
{(41,42.43,44,45,00,00,00,00,00,00,00,00,00,00,26,ac,47,99,c7,
    43,c2,df,c8,5a,83,52,39,32,36,3b,3f,27,96,47,1f,1e,bd,02,58,e6).
 (32,30,32,35,2f,30,35,2f,32,37,2f,31,31,2f,33,31,dc,50,e3,3a,56,
    7e,46,d4,a3,e0,e1,98,09,85,53,82,70,ed,90,ba,71,02,78,66,69,3c),
 (64,75,6d,79,31,00,00,00,00,00,00,00,00,00,00,00,ad,07,aa,cd,1e,
      47,0f,0d,43,0a,aa,e7,51,2e,2b,c5,cd,d7,4d,05,6c,11,c5,90,4c,01),
 (64,75,6d,79,32,00,00,00,00,00,00,00,00,00,00,00,33,2d,a3,64,a1,
      bf,52,4c,0d,c3,76,e4,4f,51,f5,40,a3,1a,57,94,8d,19,4a,3b,f5,73),
 (64,75,6d,79,33,00,00,00,00,00,00,00,00,00,00,00,b2,c0,a4,03,
   3f,1c,92,73,37,84,c9,e5,45,8f,bf,c8,72,aa,aa,10,d2,ea,c4,a9,69,5d)},
```

Figure 4: The RS-signature set W=$\{w_1, w_2, \cdots, w_5\}$, with the total size of 210 bytes

```
σ_B={(58,14),(2C,3D),(C2,A6),(F2,4E),(17,84),
     (18,5A),(97,83),(F2, B7),(D6,E9), (B8,35),
     (DF,2B),(B0,2A),(B3,5A),(0C,14),(3E, 7A),
     (2B,08),(86,FA),(03 ,21),(FC,CD),(BD,E3)}
```

Figure 5: Digital signature σ_B

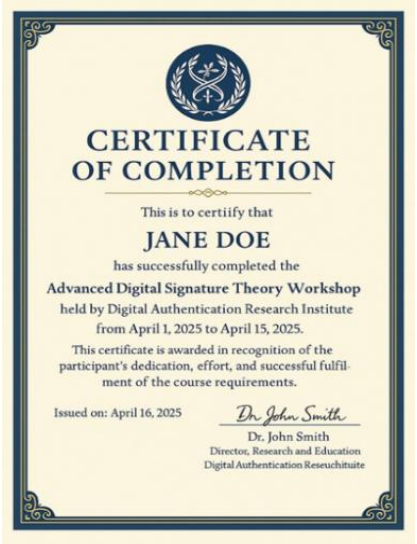

Figure 6: RS-Signed digital certificate $\widetilde{M}_s$ (424KB)

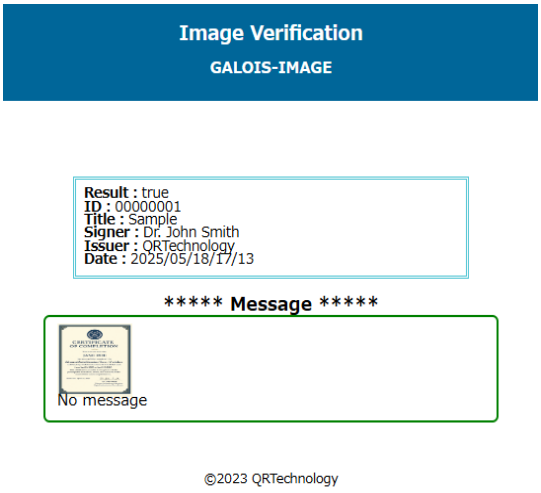

Figure 7: Verification result using the website [14].

6. EXTENSIBILITY OF RS-SIGNATURE SCHEME

The RS-signature scheme can, in principle, be applied to a wide variety of file formats. In this section, to demonstrate the extensibility of the RS-signature scheme, the scheme is applied to HTML files, and the experimental results on RS-signature generation and verification for HTML files are briefly presented.

The experiment was conducted using desktop applications for signing and verification. Figure 8 shows the RS-signed HTML file that serves as the login page. As can be seen from Figure 8, the RS-signature set consisting of five RS-signatures is embedded into the original HTML file. In this case, the signer information is as follows:

Signer: DEMO
Date: 2025/08/24/16/45 (auto input)

Figure 9 shows the result of the verification experiment for the RS-signed HTML file shown in Figure 8.

By adding an X.509 digital certificate to the <hidden> element, a trust chain can be established with the certification authority as the root.

RS-signature set

Figure 8: RS-signed HTML file $\widetilde{M}_s$

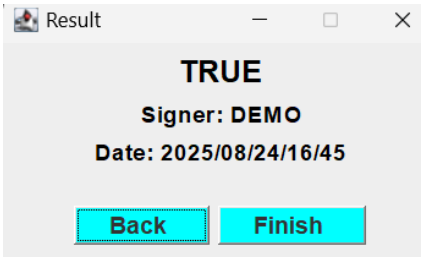

Figure 9: Verification Result on Desktop App

Phishing is still a serious cybersecurity threat. Attackers use fake websites and messages to steal personal information. Even with filters and user education, new and targeted attacks are difficult to block. The RS-signature method, when used as a web browser extension, can instantly verify whether a web page is authentic. This helps protect users from phishing attacks.

7. SECURITY DISCUSSION

The digital signature algorithm used in RS-signature scheme is ECDSA, which is widely utilized as Cryptographic Standards and has high security [17][18]. Therefore, the security strength in the RS-signature scheme can be regarded as equivalent to that of ECDSA. Furthermore, it is extremely difficult to obtain the ECDSA digital signature $\boldsymbol{\sigma}_B$ from the RS-signature set $\widetilde{\boldsymbol{W}}$. The reasons are as follows:

- $\widetilde{\boldsymbol{W}}$ and $\boldsymbol{\sigma}_B$ consist of 210 bytes and 40 bytes, respectively (see Section 5).
- Each $\widetilde{\boldsymbol{w}}_{\boldsymbol{i}} \in \widetilde{\boldsymbol{W}}$ $(i = 1, \cdots, 5)$ consists of 42 bytes and can correct up to 13 bytes. Therefore, $\widetilde{\boldsymbol{W}}$, can correct up to 65 bytes.
- Suppose 25 bytes of dummy data are embedded into $\tilde{W}$ in addition to $\boldsymbol{\sigma}_B$.
- The decoding algorithm for $\widetilde{\boldsymbol{W}}$ extracts 65 erroneous bytes E.

The total number of permutations for selecting and rearranging 40 bytes from E is approximately $P(65, 40) \approx 5.3 \times 10^{65}$, which is an astronomically large number. Consequently, obtaining a valid $\boldsymbol{\sigma}_B$ remains computationally infeasible. Thus, the security of the RS-signature scheme relies on both the cryptographic strength of the signature and the difficulty of extracting it.

In addition, by embedding an X.509 digital certificate into $\boldsymbol{M}$, it is possible to establish a verifiable trust chain rooted in a certification authority (CA). This enhances both the integrity and reliability of the digital certificate system.

8. CONCLUSION

This paper proposed the RS-signature scheme for digital file authentication and verified its effectiveness through image and HTML files authentication experiments. The RS-signature scheme is, in principle, applicable to a wide range of file formats, including PNG, XML, PDF, DOCX, and XLSX, and it offers promising potential for future applications of RS-signature scheme. Furthermore, as can be seen from the security discussion, RS-signature scheme is expected as a post-quantum signatures.

9. REFERENCES

[1] https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-5.pdf .

[2] PKCS #1 v2.2, "RSA Cryptography Standard", RSA Laboratories, RFC 8017 (2016), https://datatracker.ietf.org/doc/html/rfc8017.

[3] Microsoft. (n.d.). *Blocked attachments in Outlook*. Microsoft Learn. Retrieved July 29, 2025, from https://learn.microsoft.com/en-us/outlook/troubleshoot/security/blocked-attachments

[4] Eastlake, D., Reagle, J., Solo, D. (editors): XML-Signature Syntax and Processing: W3C Recommendation: 12 February 2002 (See: http://www.w3.org/TR/2002/REC-xmldsig-core-20020212/)

[5] Azzedine Benameur, Faisal Abdul Kadir, Serge Fenet, "XML Rewriting Attacks: Existing Solutions and their Limitations", arXiv:0812.4181,2008.

[6] Sebastian Gajek, Meiko Jensen, Lijun Liao, Jörg Schwenk, "Analysis of Signature Wrapping Attacks and Countermeasures ", Conference: IEEE International Conference on Web Services, ICWS 2009, Los Angeles, CA, USA, 6-10 July 2009

[7] V. Mladenov, C. Mainka, K. M. Selhausen, M. Grothe, J. Schwenk, "Attacks bypassing the signature validation in PDF", Ruhr University Bochum's Vulnerability Report, Nov. 2018.

[8] I. S. Reed, G. Solomon, "Polynomial Codes over Certain Finite Fields" (PDF). Journal of the Society for Industrial and Applied Mathematics. 8 (2): 300–304, 1960.

[9] R. J. McEliece , "A public-key cryptosystem based on algebraic coding theory", DSN Prog. Rep., Jet Prop. Lab., California Inst. Technol., Pasadena, CA, pages 114–116, January 1978.

[10] N. T. Courtois, M. Finiasz and N. Sendrier, "How to achieve a McEliece-based digital signature scheme", in Proc. of the 7th Int. Conf. on the Theory and Application of Cryptology and Information Security-Advances in Cryptology-ASIACRYPT 2001. Gold Coast, Australia: Springer-Verlag, Berlin, 9-13 December 2001, pp. 157–174.

[11] Farshid Makoui, Thomas Aaron Gulliver, Mohammad Dakhilalian,"A new code-based digital signature based on the McEliece cryptosystem", IET Commun. 17(10), 1199–1207, 2024

[12] Sugiyama, M Kasahara, S Hirasawa and T Namekawa, A method for solving key equation for decoding Goppa codes, Information and Control, Vol.27, pp. 87–99, 1975.

[13] SEC 2: Recommended Elliptic Curve Domain Parameters: https://www.secg.org/SEC2-Ver-1.0.pdf

[14] QRTechnology LLC. GaloaImage. Available at: https://galoaimage.com .

[15] Zauner, C. "Implementation and benchmarking of perceptual image hash functions." Upper Austria University of Applied Sciences, 2010.

[16] Monga, V., Evans, B. L. "Perceptual image hashing via feature points: performance evaluation and tradeoffs." IEEE Transactions on Image Processing, 2006.

[17] Digital Signature Standard (DSS), https://csrc.nist.gov/pubs/fips/186-5/final.

[18] ISO/IEC 14888-3:2018, https://www.iso.org/standard/76382.html.