Chapter 4
Online Image Authentication with Security Protection of Authentication Certificates

4.1 Overview of Proposed Method

In Section 4.1.1, the motivation of the proposed method for online image authentication with security protection of authentication certificates is described. And in Section 4.1.2, we give a briefly review of the Secure Hash Algorithm-1 (SHA-1). In Section 4.2, the online image authentication and tampering detection procedures are described. Finally, in Section 4.3, some discussions and a summary are made.

4.1.1 Motivation

Traditionally, image owners go to the central image authentication center to register their images. When they find copyright-infringed images, they must go to the central image authentication center again to authenticate the copyright-infringed images and request the center to issue physical certificates about this fact for later uses. But this is an inefficient way.

An online image authentication system is proposed in our study. Image owners also need to go to the central image authentication center to register their images. But after the registration, once they find copyright-infringed images, they can authenticate those images immediately by giving the URLs of those images to the central image authentication center online and requesting digital certificates. Illegal users will have no time to eliminate the evidence then, as mentioned previously.

In addition, since the record in a certificate, which indicates infringed or non-infringed, is of great importance, we propose a method for certificate tampering detection using a commonly used hash function, the Secure Hash Algorithm-1 (SHA-1).

4.1.2 Review of Secure Hash Algorithm-1

The SHA family is a set of related cryptographic hash function, and the most commonly used function in the family is the SHA-1, which is employed in a large variety of popular security applications and protocols like Transport Layer Security (TLS), Secure Sockets Layer (SSL), Secure Shell (SSH), etc.

The SHA-1 produces a 160-bit hash value. The hash value is generated by a function H of the form h = H(M) where M is a message with a maximum length of 2^{64} bits, and H(M) is the 160-bit hash value, which is also called the *message digest*. Figure 4.1 shows message digest generation using the SHA-1. At first, the input message is padded so that its length is congruent to 448 modulo 512. That is, the length of the padded message is 64 bits less than an integer multiple of 512 bits and then a 64-bit representation of the original message length is appended. Then the outcome yields a message that is an integer multiple of 512 bits in length and it can be processed in 512-bit blocks. Assume the length of the input message after padding is $L \times 512$ bits. We segment the input message into message blocks Y_0 to Y_{L-1}, and each has 512 bits in length. The heart of the SHA-1 is a module that consists of 80 iterations of processing. Each modulo is assumed to be H_0 to H_{L-1} and the output of a module is the input to the next module. The elementary SHA-1 operation in the *t*th iteration is shown in Figure 4.2. ABCDE is a 160-bit buffer value, and the contents of the five 32-bit buffers A, B, C, D and E are updated in each iteration in the following way:

A, B, C, D, E $\leftarrow$ (E + $F(t$, B, C, D) + S^5(A) + W_t + K_t), A, S^{30}(B), C, D,

where the notations are explained as follows.

(1) $F(t$, B, C, D) is a primitive logical function for iteration t where t is the iteration number from 0 to 79. Each primitive function is defined as follows:

$$F(t,B,C,D)=\begin{cases}(B\wedge C)\vee(\overline{B}\wedge D) & \text{if } 0\le t\le 19;\\ B\oplus C\oplus D & \text{if } 20\le t\le 39;\\ (B\wedge C)\vee(B\wedge D)\vee(C\wedge D) & \text{if } 40\le t\le 59;\\ B\oplus C\oplus D & \text{if } 60\le t\le 79,\end{cases}$$

(2) S^n denotes a circular left shift of the 32-bit argument by n bits, and symbol + denotes addition modulo 2^{32}.

(3) W_t is a 32-bit word derived from the current 512-bit input block. The first 16 values of W_t are taken directly from the 16 words of the current block. The remaining values are defined as $W_t = S^1(W_{t-16}\oplus W_{t-14}\oplus W_{t-8}\oplus W_{t-3})$.

(4) K_t is four distinct predefined constants as follows:

$$K_t=\begin{cases}\texttt{5A827999} & \text{if } 0\le t\le 19\\ \texttt{6ED9EBA1} & \text{if } 20\le t\le 39\\ \texttt{8F1BBCDC} & \text{if } 40\le t\le 59\\ \texttt{CA62C1D6} & \text{if } 60\le t\le 79\end{cases}$$

The contents of five buffers after 80 iterations in module H_q are added to the input buffers of the first iteration to produce the next certificate value CV_{q+1}. The addition is done using addition modulo 2^{32} which is shown in Figure 4.3. Then CV_{q+1} are the outputs of module H_q and are taken as inputs to module H_{q+1}.

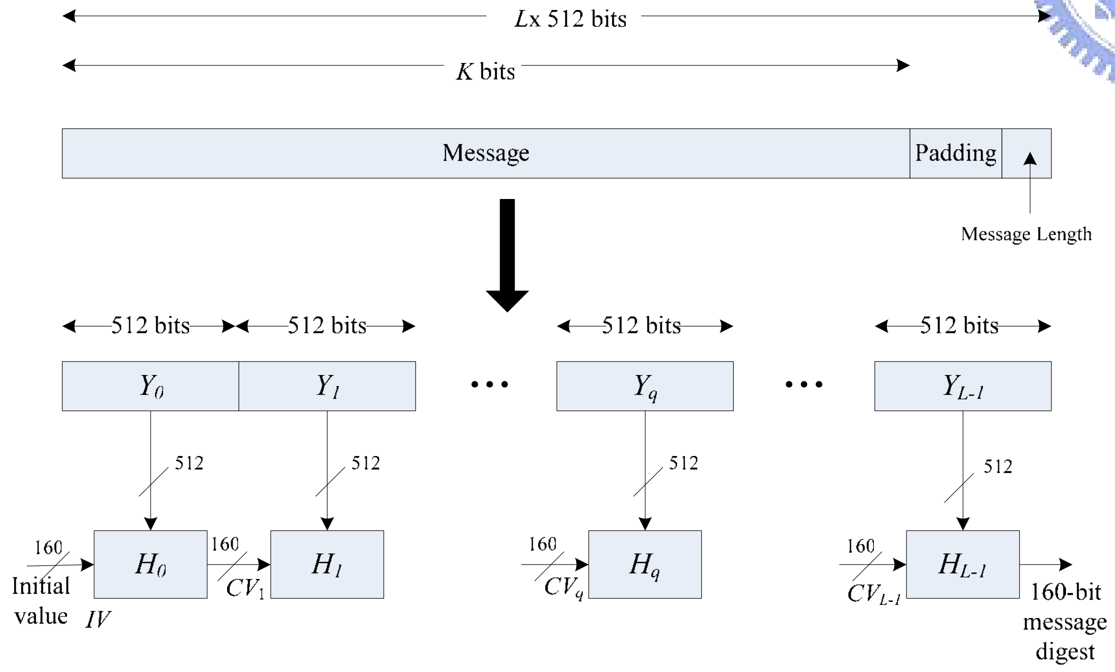

Figure 4.1 Message digest generation using SHA-1.

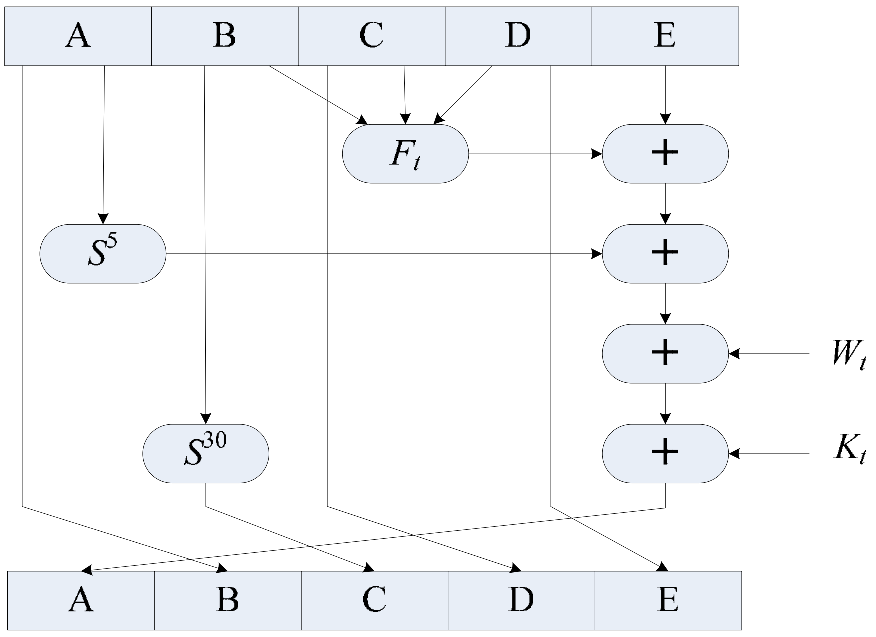

Figure 4.2 Elementary SHA-1 operation in iteration *t*.

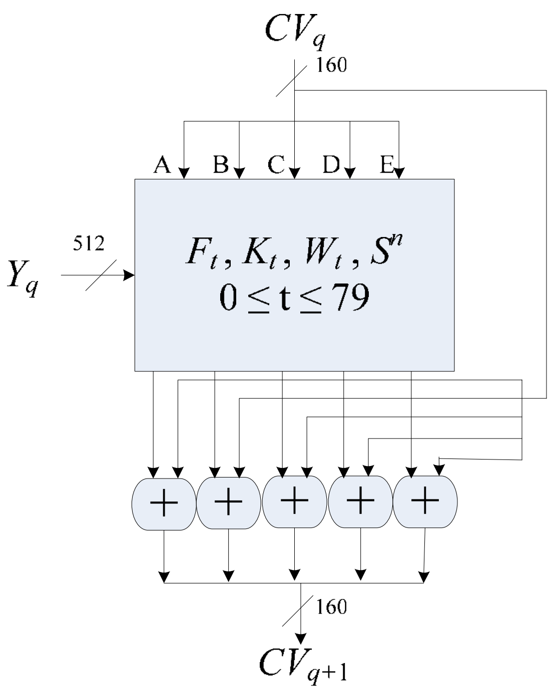

Figure 4.3 Elementary SHA-1 operation in iteration *t*.

On account of security protection, it must be prohibitively difficult to invert the hashing process to construct a message with a given hash value. That is, the hash value should be long enough to make the construction an intractable problem, which means that it is not solvable in polynomial time. And using a brute-force technique, the difficulty of producing any message having a given message digest is on the order of 2^{160} operations. It will spend over one hundred years to solve this kind of problem even using the fastest computer of nowadays. In addition, the SHA-1 has the characteristic of avalanche effect, which is the fundamental property of block ciphers. The avalanche effect means that even if the input message is flipped a single bit, the output message digest changes significantly. So it is difficult to find certain rules to make a forged message with a given message digest.

In our study, the function input M is a certificate, so we call the 160-bit output as a *certificate digest*. To prevent a digital certificate from being tampered with during the transmission, the central center can compute the certificate digest of the certificate before sending it to the local center. Then the local center can authenticate the certificate by re-computing a new certificate digest. If the new certificate digest is the same as that the central center computes before sending, then the authentication results recorded in the certificate is trustworthy.

4.2 Online Image Authentication Procedures

In Section 4.2.1, the procedure for registration as a local image authentication center is introduced. And the process of online authentication for copyright-infringed images is described in Section 4.2.2.

4.2.1 Procedure for Registration as A Local Image Authentication Center

In the proposed system, an image owner must register his/her digital images at the central image authentication center first. The purpose of registration is to keep a record of the owner of copyrighted images and his/her pre-defined digital watermark, and this step of registration makes the whole system more equitable. Because local image authentication centers have the software package InfoProtector, a person there might embed his/her own watermark into images which do not belong to him/her and claim that he/she is the owner of the images. Thus, the central image authentication center only accepts the request for authenticating a *registered* image in order to prevent local image authentication centers from misuse of InfoProtector.

4.2.2 Online Procedure for Authentication of Suspected Images

After the step of registration mentioned above, the local image authentication center will have two software packages, which are the InfoProtector and the package in which the proposed methods in this study are implemented. The local image authentication center can use InfoProtector to embed his/her registered watermark into his/her registered images. Then he/she posts the watermarked images on the Internet. When he/she finds that someone copies his/her images without permission, he/she can use the latter software package to request immediately the central image authentication center to issue a certificate to keep a record of the illegal user's infringement of the copyrighted images.

In the step of authentication, the local image authentication center not only can choose to send the URL of a suspected image to the central image authentication center but also can choose to send the URL of a webpage or a public FTP server. Which way should be done depends on the network loading and the computing power of the central image authentication center, as mentioned previously.

Relatively, the central image authentication center will probably receive a URL of a suspected image, a URL of a webpage, or a URL of a public FTP server. When the central image authentication center receives a URL of a suspected image, the central center will download the image according to the received URL. First, he/she will check whether there is an embedded signal in the suspected image or not. If no signal can be extracted, then it means that the image has not been processed by InfoProtector before. And the central center will then send a digital certificate with no extracted information back to the local image authentication center to indicate that the suspected image is not a copyright-infringed one. If there is indeed an embedded signal in the suspected image, then all relevant information will be extracted out of the image, including the image owner, the embedded watermark, the embedded annotation, etc. The central center will subsequently issue a certificate with the extracted information and send it back to the local image authentication center. If the extracted watermark is the *same* as the local image authentication center's watermark, then the suspected image is regarded as a copyright-infringed one.

However, when the central image authentication center receives a URL of a webpage or a URL of a public FTP server, the central center will first detect all images in the webpage or in the public FTP server by different methods. After downloading all suspected images, the central center deals with each image as just mentioned above. First, it checks whether there is an embedded signal in it. Second, all information in the image is extracted if needed. Third, a certificate with all extracted information is issued and sent back to the local image authentication center.

Between the process after receiving a URL of an image and the process after receiving a URL of a webpage or a URL of a public FTP server, the loading of the network and the computing power of the central image authentication center's computer are the major differences. The central image authentication center might have to issue many certificates and send these certificates back to the local image authentication center if a local image authentication center just gives the central center the URL of a webpage or the URL of a public FTP server. Figure 4.4 is a flowchart of online image authentication.

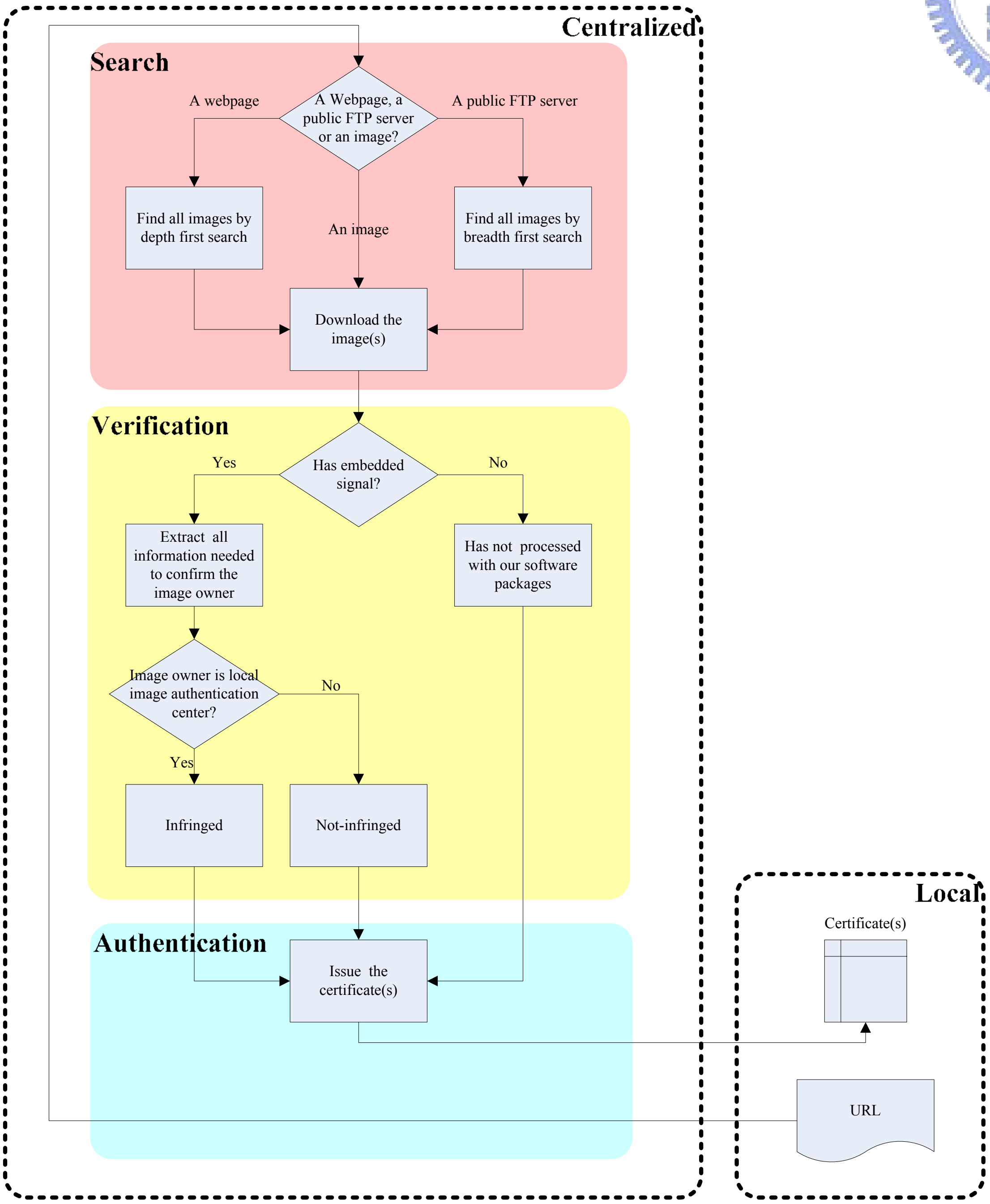
Centralized
Search
A webpage
A Webpage, a public FTP server or an image?
A public FTP server
Find all images by depth first search
An image
Find all images by breadth first search
Download the image(s)
Verification
Yes
Has embedded signal?
No
Extract all information needed to confirm the image owner
Has not processed with our software packages
Image owner is local image authentication center?
No
Yes
Infringed
Not-infringed
Authentication
Issue the certificate(s)
Local
Certificate(s)
URL

Figure 4.4 A flowchart of online image authentication.

4.3 Proposed Method for Security Protection of Authentication Certificates

In Section 4.3.1, the definition and the file format of authentication certificates are given. And the proposed method for tampering detection of certificates is described in Section 4.3.2.

4.3.1 Definition and File Format of Authentication Certificates

A certificate, which is an authentication result issued by the central image authentication center, is a record of whether the suspected image is a copyright-infringed one or not, and can be a trusted evidence in a lawsuit.

Figure 4.5 shows the proposed format of a certificate, which includes following elements:

1. Issuing time: An issuing time is the time when the certificate is issued.
2. Image source: An image source is the URL of a downloaded image.
3. Embedded version: An embedded version is the version of the software InfoProtector we use to embed information into the image and it is one of the information that can be embedded into the image.
4. Embedded owner: An embedded owner is the image owner, which is one of the information that can be embedded into the image by the software InfoProtector.
5. Embedded author: An embedded author is the image author, which is also one of the information that can be embedded into the image by the software InfoProtector.

6. Embedding date: An embedding date is the date when we embedded all needed information into the image by the software InfoProtector and it is also one of the information that can be embedded into the image.
7. Embedded identity code: An embedded identity code is a character representing the legal user of the InfoProtector used to embed information and it is also one of the information that can be embedded into the image.
8. Embedded watermark type: An embedded watermark type is a character indicating what kind of watermark is embedded in the image and it is also one of the information that can be embedded into the image.
9. Embedded annotation: An embedded annotation is a description of the image and it is also one of the information that can be embedded into the image.
10. Embedded visible watermark: An embedded visible watermark is a visible watermark representing the image owner and it could be one of the embedded information.
11. Embedded invisible watermark: An embedded invisible watermark is an invisible watermark representing the image owner and it could be one of the embedded information.
12. Downloaded image: A downloaded image is the suspected image which we download on the Internet and then a certificate is issued according to this image.

No matter whether the suspected image is a copyright-infringed one or not, an issuing time, an image source, and a downloaded image are always recorded in an authentication certificate. These three are the most important records among all elements because they are the only proofs of an illegal user's infringement of the image(s) if an illegal user has removed copyright-infringed images before he/she is sued. We can still know who uses a copyright-infringed image at what time according to these three records in the certificate. That is, we can infer that the copyright infringement has happened before even when there is no copyright-infringed image on his/her webpage now.

However, the rest of the records in the certificate are important, too. We judge the suspected image as an infringed or a non-infringed one by these records. If the embedded owner is the local image authentication center who requests for issuing the certificate, or if the embedded watermark is the registered watermark which belongs to the local image authentication center who requests for issuing the certificate, then we can say that the suspected image is a copyright-infringed one.

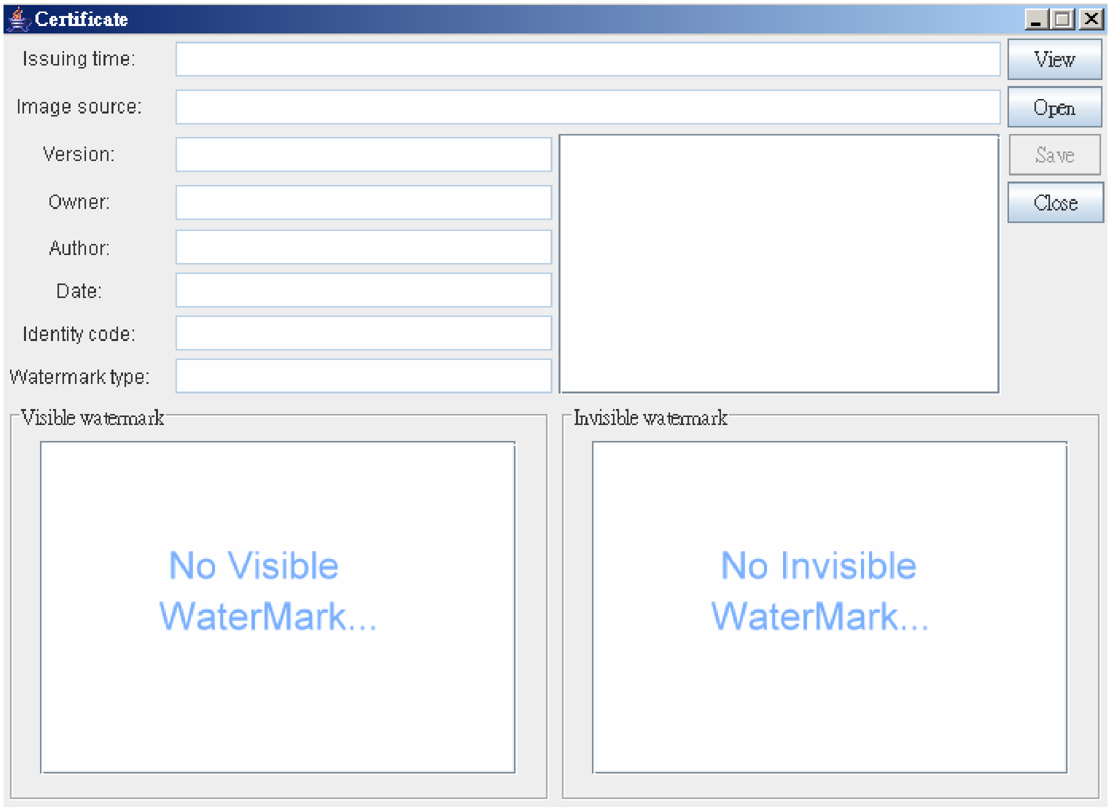

Figure 4.5 Proposed format of certificate.

4.3.2 Proposed Method for Certificate Tampering Detection

Since the certificate is significantly important in the proposed image authentication system, we apply a hash function, called SHA-1, to keep the accuracy of our system. The proposed method makes it simple to check whether the certificate is trustworthy or not by applying the SHA-1.

Before sending the digital certificate to the local image authentication center, we calculate a certificate digest. When the local image authentication center receives the certificate, the local center calculates a certificate digest as well. And the local image authentication center sends the certificate digest to the central image authentication center to check whether the received certificate is free from tampering and is trustworthy. So the central image authentication center compares the received certificate digest with his/her pre-calculated one. If they have the same 160 bits, then the certificate which the local image authentication center received is trustworthy. But, on the contrary, if they are not exactly the same certificate digest, it means that the certificate has been altered by someone during the transmission. Consequently, the certificate cannot be trusted any more.

The goal of the proposed method is to detect tampering when it happens in order to prevent from suing wrong people. We achieve the goal of certificate tampering detection by applying the SHA-1 to the certificate. However, we cannot exactly find which part is altered by this way. But this is not essential in our system.

The process of applying the SHA-1 to a certificate can be briefly expressed as an algorithm as follows.

Algorithm 4.1: *Applying SHA-1 to a certificate to produce a certificate digest.*

Input: A certificate *C* and five 32-bit buffers A, B, C, D, E.

Output: A certificate digest *D*.

Steps:

1. Pad the input *C* with bits starting with a “1” followed by a sufficient number of “0s” so that its length in bits is congruent to 448 modulo 512. And append a 64-bit representation of the original message length. Then assume the length of *C* after padding is $L \times 512$ bits.
2. Segment *C* into message blocks Y_0 through $Y_{\mathrm{L-1}}$, with each having 512 bits in length. Let the modules as mentioned in Section 4.1.2 be assumed to be H_0 through $H_{\mathrm{L-1}}$, with each consisting of 80 iterations.
3. Initialize five 32-bit buffers to have the following integers (hexadecimal values):

```
A = 67452301;
B = EFCDAB89;
C = 98BADCFE;
D = 10325476;
E = C3D2E1F0.
```

4. Take the initialized A, B, C, D, E and the first 512-bit message block as inputs to module H_0.
5. Process the message in H_0 for 80 iterations as mentioned in Section 4.1.2.
6. Add the contents of the five buffers after 80 iterations, using the way of modulo 2^{32} addition which is shown in Figure 4.3, to the input buffers of the first iteration to produce the output of H_0, assumed to be CV_1.
7. Take CV_1 and the next 512-bit message block Y_1 as inputs to module H_1. Obtain CV_2 by the same way as described in Steps 5 and 6 by processing the message in H_1 and doing the addition.
8. After all *L* 512-bit blocks H_0 through H_{L-1} have been processed, the output

of module H_{L-1} is the desired 160-bit certificate digest D.

The process of certificate tampering detection can be briefly expressed as an algorithm as follows. And an example of detecting a certificate which has been tampered with is shown in Figure 4.6.

Algorithm 4.2: *Certificate tampering detection*.

Input: A certificate C.

Output: A warning message W about certificate tampering if needed.

Steps:

1. Compute a certificate digest D_C using the certificate C at the central image authentication center. Then keep D_C at the central center and send C to the local image authentication center.
2. Compute a certificate digest D_L at the local image authentication center as soon as the local center receives the certificate C.
3. Send the 160-bit certificate digest D_L from the local center back to the central center.
4. Compare the received certificate digest D_L with the certificate digest D_C computed before in the following way at the central center.

   4.1 If D_L and D_C are the same, then regard the certificate C received by the local center as trustworthy.

   4.2 If D_L and D_C are not exactly the same, then regard the certificate C received by the local center as having been tampered with during the transmission, and issue a warning message W from the central center to the local center.

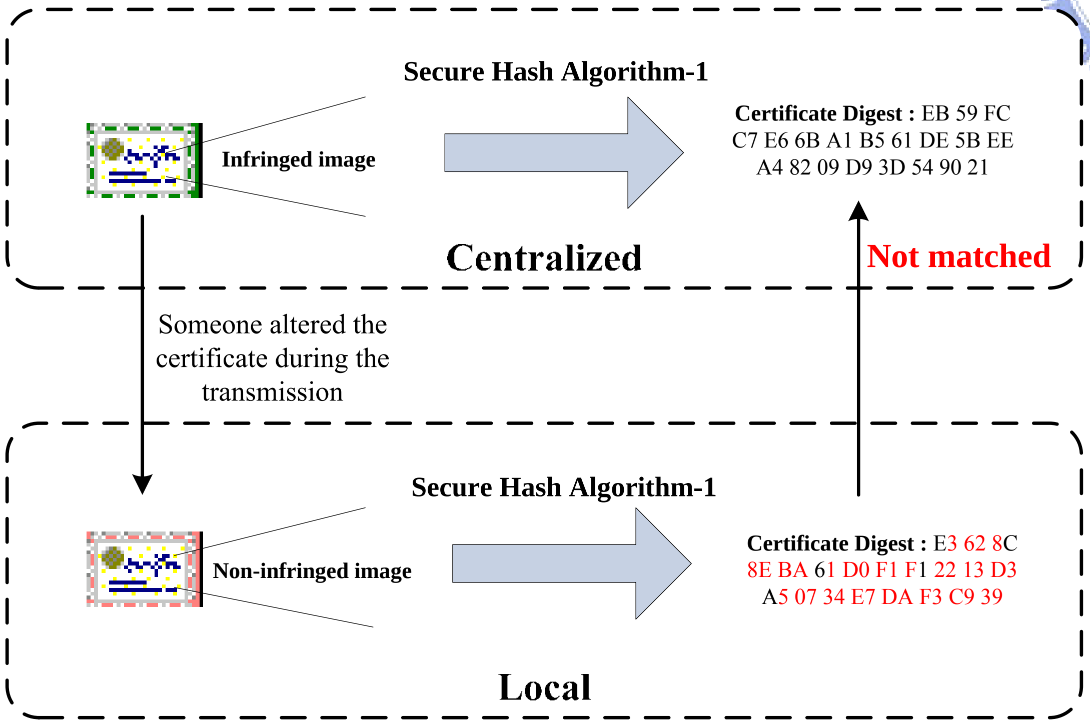
Secure Hash Algorithm-1
Infringed image
Certificate Digest : EB 59 FC C7 E6 6B A1 B5 61 DE 5B EE A4 82 09 D9 3D 54 90 21
Centralized
Not matched
Someone altered the certificate during the transmission
Secure Hash Algorithm-1
Non-infringed image
Certificate Digest : E3 62 8C 8E BA 61 D0 F1 F1 22 13 D3 A5 07 34 E7 DA F3 C9 39
Local

Figure 4.6 An example of detecting a certificate which has been tampered with in proposed system.

4.4 Discussions and Summary

In this chapter, we propose methods for online image authentication with security protection of authentication certificates. After an image owner registers his/her images at the central image authentication center and becomes one of the local image authentication centers, he/she can authenticate images online when he/she finds images which infringe his/her copyright on the Internet each time. This is an efficient way because the local image authentication center can receive issued certificates immediately via the Internet. And an illegal user will have no time to destroy the evidence of the infringement.

And to avoid suing wrong people, we apply the SHA-1 to our system. We compare the certificate digests which are calculated from the certificate sent by the central image authentication center and the certificate received by the local image authentication center. If the certificate received by the local center is exactly the one which the central center sent, the certificate digests should be the same and the received certificate is regarded trustworthy. If two certificate digests are not the same, it means that the received certificate has been altered in part and cannot be trusted any more.