

## Chapter 4

# Layered Access Control on Watermarked Scalable Media

In this chapter, we describe the design of a layered protection scheme on scalable digital media. It combines three concepts: scalable coding, watermarking, and cipher chaining. This chapter is organized as follows. Firstly, we describe the motivation and goals of this design in Sec. 4.1. In Sec. 4.2, we describe the recursive coding forms for scalable coding and chained encryption. Then we propose a scheme to combine these techniques together to provide a self-synchronized protection architecture in Sec. 4.3. To verify the feasibility of the scheme, we construct two simple applications and perform simulations in Sec. 4.4 and Sec. 4.5. At the end of this chapter, we summarize the features of this technique.

### 4.1 Motivation and Goals

With the widespread use of Internet and the growing demand of multimedia delivery, digital media, including images, audio and video clips, are easily acquired in our daily life. The current network environments make scalable coding of multimedia a necessary requirement when multiple users try to access the same information through different communication links [39][40]. Scalability implies that a set of

multimedia data is partitioned into hierarchical layers such that the lowest layer, called the base layer, has the most reduced content quality. The reduction may be in spatial frame size, temporal frame rate, or signal-to-noise ratio (SNR) from fine to coarse quantization levels. To reproduce content a higher quality, the enhancement layers provide additional data to improve the multimedia quality. Enhancement layers accomplish the scalability of the content coding, namely, spatial, temporal, or SNR scalability. Therefore, scalable coding of multimedia is suitable for delivering digital contents to different users and devices with various capabilities [41].

Under a wide spectrum of applications, it requires to deliver multimedia contents securely. However, the channel for multimedia broadcasting is an open environment; thus, if the user data and information are not protected by an adequate means, it might be illegally used or altered by hackers. To protect privacy and intellectual property rights, people often use cryptographic techniques to ensure secure delivery [42][43]. One shortcoming of cryptographic approach is that an error in a ciphertext may lead to a partially or totally failed decryption depending on the cipher mode [44]. In a broadcast environment, where erroneous transmission may occur from time to time and no retransmission is allowed, this may induce serious quality degradation in service. To alleviate such a shortcoming, we employ digital watermarking techniques with cryptographic approaches, which will be described in later sections.

There are two issues in delivering encrypted contents over a broadcast environment. A common design incorporates frequent key change and transmission. The key distribution induces the two problems: the first one is that the key requires stronger protection from transmission errors because it is used for decryption. The other is, when keys and contents are transmitted over a channel that does not guarantee receiving order, the synchronization between contents and keys becomes an issue [45]. To meet these two requirements, we propose a combined intellectual property (IP) protection scheme that consists of multi-layer content encryption and key-content synchronization.

## 4.2 Recursive Data Flow Structures

As discussed in Sec. 4.1, scalable coding is a solution for broadcasting contents to devices with various playback capabilities. With the help of scalable coding, the entire media can be partitioned into layers of data. Thus, we can organize the content for different groups of users. On the one hand, it is straightforward to group receivers of different playback capabilities by sending different combinations of data partitions; on the other hand, conditional access (CA) is also assured by encrypting separate data partitions by different access keys.

### 4.2.1 Scalable Coding

The goal of scalable coding is to provide multi-grade services. At the encoder side, the given content is first analyzed and hierarchically partitioned into several layers. The scalability can be designed in one or more of the spatial, temporal, or SNR domains. The spatial-domain scalability provides different spatial resolutions of the image and video contents; the temporal scalability provides various video and audio playback frame rates; and the SNR scalability allows different levels of quantization noise in reconstructed data. Among all of them, the most basic layer, or the lowest quality layer, is called the *base layer*, and all the others are called the *enhancement layers*. At the decoder side, after receiving the base layer, the enhancement layers are added to improve the decoded quality of the content. Due to the scalable decoding process, a receiver can select or use the available layers to construct the content to the desired quality level. Two practical examples are provided as follows:

- A low-end receiver device may select fewer layers to decode to match its processing power, resulting in an output with a lower quality;
- A player device receives scalable coded data from an unreliable channel, and it may suffer from data corruption or data loss. By using the “available” data (optionally with error recovery techniques), the device can restore the content

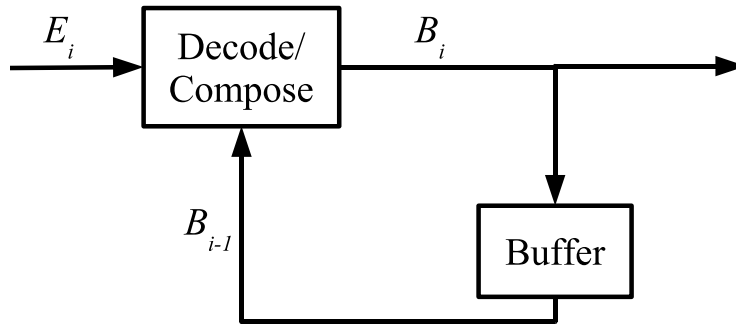


Figure 4.1: Recursive form of scalable decoding.

quality as much as possible.

In the above examples, the decoder may use as many available layers as possible to reproduce the contents.

Based on the scalable decoding process, it may be represented as an architecture with recursive enhancement layers. As shown in Fig. 4.1, assuming that  $B_0$  is the base layer, i.e.,  $B_i$  for  $i = 0$ . For  $i \geq 1$ , the enhanced content  $B_i$  is the composition of the low-quality content  $B_{i-1}$  and the received enhancement data  $E_i$ . To obtain the highest quality, the iterative process is applied throughout all the layers. The multi-grade service is provided by terminating the process at a proper layer which is suitable for the processing power and/or the availability for decoding received data.

## 4.2.2 Typical Streaming Encryption

To protect the data for transmitting over broadcast channels or the Internet, data encryption algorithms are widely used. There are several well-known encryption algorithms developed for commercial or military use, such as the DES [46], AES [47], and RSA algorithms. There are two categories of the data encryption algorithms: one is called symmetric key algorithms, and the other is called asymmetric key algorithms. They can be briefly described as follows:

- In a symmetric key algorithm, a single secret key is shared between the encryption side and the decryption side, which is used to encrypt and decrypt data. In this kind of system, the encrypted data is transmitted in the open environment. However, the key should be delivered by another means to ensure that only the authorized peers can access the key, otherwise unauthorized users would access the decrypted data.
- In an asymmetric key algorithm, the secret (the key pair) has two parts: one is called the public key, and the other is called the private key. To protect the data, the sender encrypts the data using the public key, and the receiver decrypts it using the private key. In this kind of systems, the encrypted data and the public key can be distributed through the open environment, while the private key is always kept in the decryption side.

To encrypt a large amount of data at very high bit rate, such as multimedia streaming, several factors have to be considered for practical implementation. Among them, one is the complexity issue. Ideally we can use very long keys or sophisticated algorithms to provide a stronger security level, but it would also require very expensive hardware and thus makes it impractical. There are several techniques to trade-off between security and complexity. A commonly used scheme is to divide the original data into blocks, and the encryption key of the current block is derived from the previous blocks and/or keys. This is called the cipher block chaining, and is often used with symmetric key algorithms. Another commonly used technique is to protect the decryption key by other encryption schemes or algorithms. Because symmetric encryption algorithms are often much simpler than asymmetric algorithms, the multi-layer encryption may be designed as: to protect the data using symmetric algorithm with short and frequently changed keys, and to encrypt the key stream using a stronger encryption method.

The second implementation consideration is that data transmission may be captured and replayed by a malicious attacker. To deal with this type of problems, most

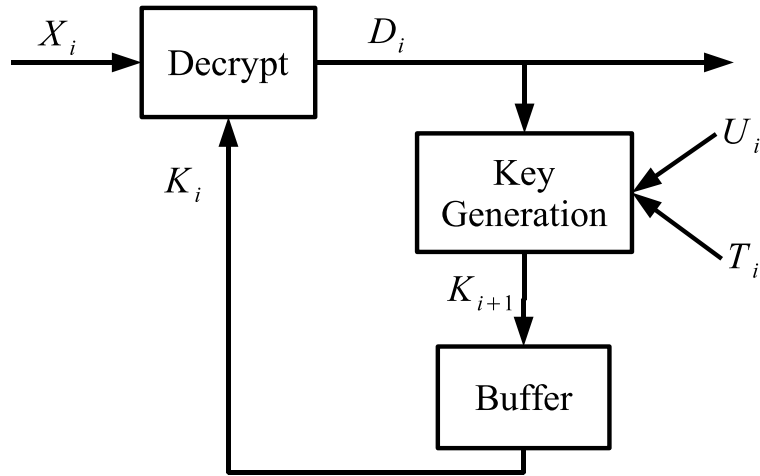


Figure 4.2: Concept of protecting streaming data.

of the on-line transaction systems encrypt the data depending on session-specific information. Briefly speaking, the encryption key depends not only on the user/data but also on the time. Combining the aforementioned data protection concepts, a high-level conceptual architecture is shown in Fig. 4.2. To decrypt the received data  $X_i$  into the original plaintext  $D_i$ , the decryption key  $K_i$  is derived from the previous recovered data  $D_{i-1}$ , the user specific information  $U_{i-1}$ , and the session/time specific information  $T_{i-1}$ .

### 4.3 Proposed Protection Scheme

In this section, we first discuss the combination of scalable coding and streaming protection architectures. Then, we describe the layered decryption and decoding operations at the receiver side. Because the associated encryption and encoding operations vary depending on the scalable coding, we provide two examples in the next sections.

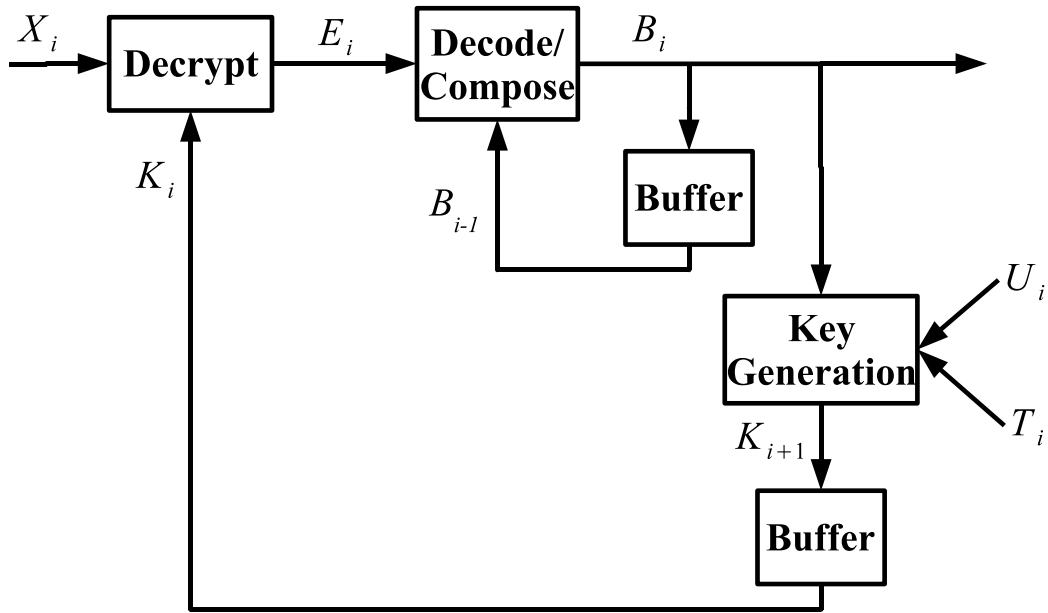


Figure 4.3: Combination of decryption and decoding.

### 4.3.1 Combination

The combination of scalable coding and content protection is to cascade the decryption and the decoding processes, such a structure is shown in Fig. 4.3, and it is a combination of Fig. 4.1 and Fig. 4.2. Although the key generation (key reconstruction) may require an input of  $B_i$  or  $E_i$ , we choose  $B_i$  because it results in a more general architecture; that is,  $B_i$  is produced by using  $E_i$ .

To broadcast the content in an open environment without feedback channels, we need to find a way to securely deliver the user information  $U_i$  and the time dependent information  $T_i$ . For distributing  $U_i$ , there are many key distribution and management schemes developed, and it is outside the scope of this paper. Here, we are more interested in how to deliver the session and time specific secret  $T_i$ . In a typical electronic commerce system, it can be obtained by negotiations among servers and clients. Since our target application does not have feedback channels, the  $T_i$  is actively updated by the server. To distribute  $T_i$  to clients, a simple way is to encrypt  $T_i$  and multiplex it with the encrypted content. However, this scheme

might suffer from synchronization problem between  $T_i$  and the content. Another problem arises from this proposal is that it might be easy to distinguish the  $T_i$  packets and the content packets, so that the attackers have more chances to corrupt the bit-stream.

To overcome these problems, we propose a sophisticated method to reduce the risk of errors. In our method, the session/time specific secret is encrypted by the user key, and then it is embedded directly into the original content by a robust watermarking algorithm. Because the information is embedded in the previous content, the synchronization problem is implicitly solved. Furthermore, robust watermarking gives more protection on  $T_i$  than on the content data. That is,  $T_i$  can survive better when transmission errors or attacks occur. Overall, the layered protection structure proposed in this paper is a reliable method that meets the conditional access and layered coding requirements at the same time. The detailed architecture is described in the next section.

### 4.3.2 Receiver Architecture

Scalable coding is composed of one base layer and several enhancement layers to match the network/receiver diversity. The enhancement layer operation is illustrated by Fig. 4.4.

Assuming that the initial base layer has been received, the subsequent composing (decoding) operations can be expressed by

$$B_i = compose(B_{i-1}, E_i), \quad (4.1)$$

and

$$E_i = decrypt_e(X_i, K_i). \quad (4.2)$$

In Eq. 4.1,  $B_{i-1}$  is the previously reconstructed content, and  $E_i$  is the enhancement layer to increase quality from  $B_{i-1}$  to  $B_i$ . In transmission,  $E_i$  is protected by a cryptographic algorithm with  $K_i$  as the key, and the transmitted data is  $X_i$  in Eq. 4.2.



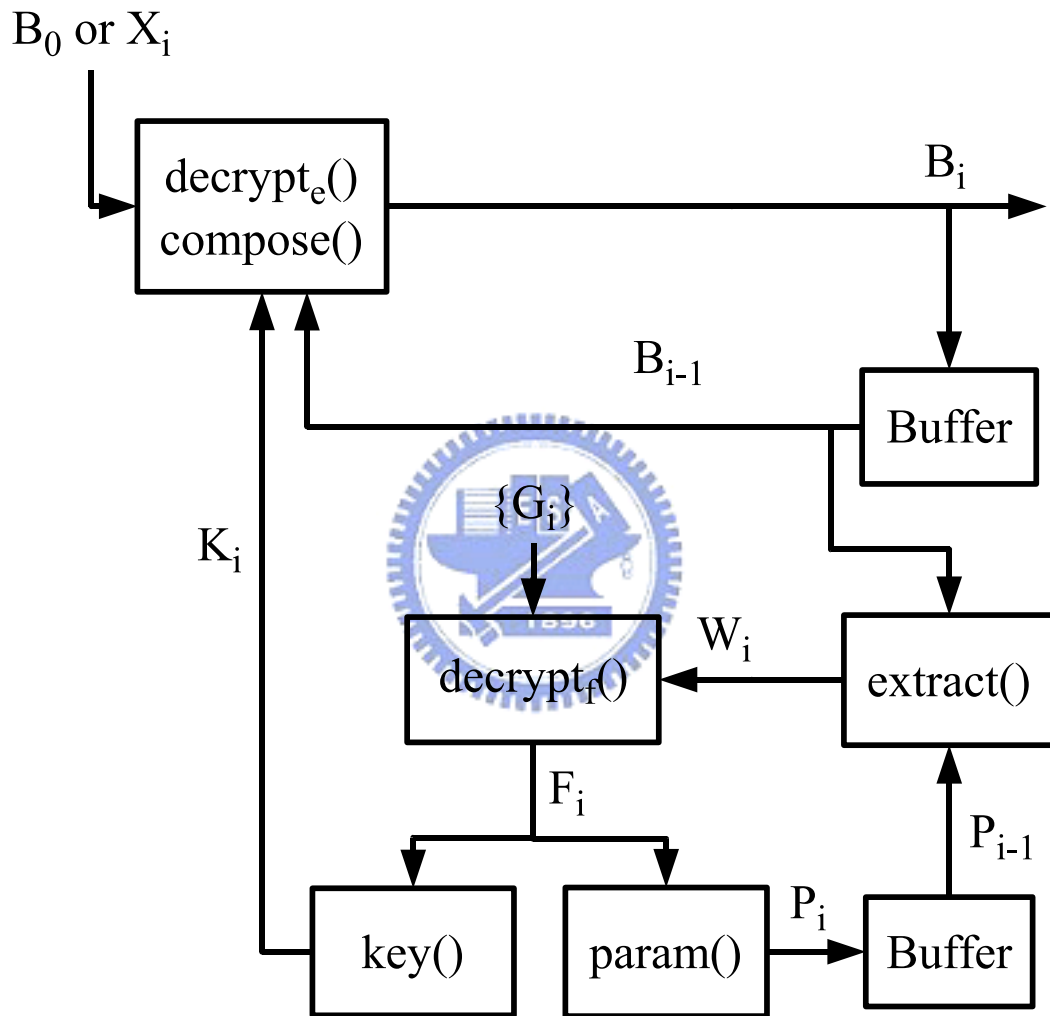


Figure 4.4: Decryption and decoding of layer-protected content.

There are some secret information to be obtained prior to decrypting  $E_i$ , and the associated operations can be expressed as follows:

$$W_i = \text{extract}(B_{i-1}, P_{i-1}) \quad (4.3)$$

$$F_i = \text{decrypt}_f(W_i, G_i) \quad (4.4)$$

$$K_i = \text{key}(F_i) \quad (4.5)$$

$$P_i = \text{param}(F_i) \quad (4.6)$$

$W_i$  is the digital watermark extracted from the reconstructed content  $B_{i-1}$  with the extraction parameter  $P_{i-1}$ . As described in Sec. 4.2,  $W_i$  represents the protected secret information. Thus, we have the secret information  $F_i$  by decrypting the watermark using user-specific key  $G_i$ . After parsing  $F_i$ , we obtain the decryption key  $K_i$  and the next watermark extraction parameter  $P_i$ .

As Fig. 4.4 illustrates, the decryption and composition blocks are iterative processes. This diagram is drawn according to the previous discussions, except that we merge the decryption and decoding blocks into one function block. Although in a typical application, decryption and decoding can be separable tasks, we do not exclude the possibility that there might be a scheme that decrypts and decodes the received data together in one operation. There are several initial parameters required to activate these processes. Some discussions of how to obtain the initial parameters are as follows.

- When the whole content is protected, namely,  $B_0$  is encrypted, we need  $K_0$  to decrypt  $X_0$ . In this case,  $K_0$  should be obtained by a separate channel.
- One scenario is that  $B_0$  is the “preview” layer; i.e.,  $B_0$  is not encrypted, we simply bypass the decryption.
- Depending on the watermarking algorithm, the extraction process may require specific parameters. If it does, the first watermark extraction parameter  $P_0$  should be obtained from a separate channel to activate subsequent extraction process.

The receiver should obtain all the user keys  $G_i$  before receiving the media data. For instance, the user keys may be distributed manually or automatically through a reliable channel. Because this is a key distribution and management issue, which is outside the scope of this paper and thus we do not discuss it further here.

## 4.4 Application 1: Scalable-Compressed Images

In this section, we apply our scheme on images with spatial scalability [48]. As shown in Fig. 4.5, the encoder/encryption architecture is almost the inverse of the receiver. The watermark  $W_i$  is the encrypted version of the key  $K_i$  and the embedding parameter  $P_i$ . The  $B'_{i-1}$  is the un-watermarked content with lower quality. After embedding  $W_i$  into  $B'_{i-1}$ , we have the watermarked content  $B_{i-1}$ . The enhancement layers are generated as the differences between  $B_i$  and  $B_{i-1}$ . All the  $K_i$ ,  $P_i$ , and  $G_i$  have been described before.

To test the feasibility of this architecture, the test image Lena of size  $1024 \times 1024$  is in use. The original Lena is first converted to a  $512 \times 512$  base-layer picture. The DES [46] key (8 ASCII letters **NCTU-DEE**) to encrypt the enhancement layer is also encrypted using DES by the user key ( $G_i$ ) to generate the 8-byte (or 64-bit) secret. The secret is then repeated for 32 times to form the binary watermark, as shown in Fig. 4.6(a). The watermarking method we adopt is a modified version of [23]. It is well known that in designing robust watermarking algorithms, both the watermarked image quality and watermark robustness play important roles. On the one hand, watermarked image quality is measured by the error between the original and watermarked images, generally measured by Peak Signal-to-Noise Ratio (PSNR). On the other hand, watermark robustness is measured by the ratio between the correctly extracted bits and the total embedded bits, generally measured by Bit Correct Ratio (BCR). Although the two requirements are desirable, they conflict with each other. For watermark embedding in the transform domain, a balanced solution between these two requirements is to embed the watermark or

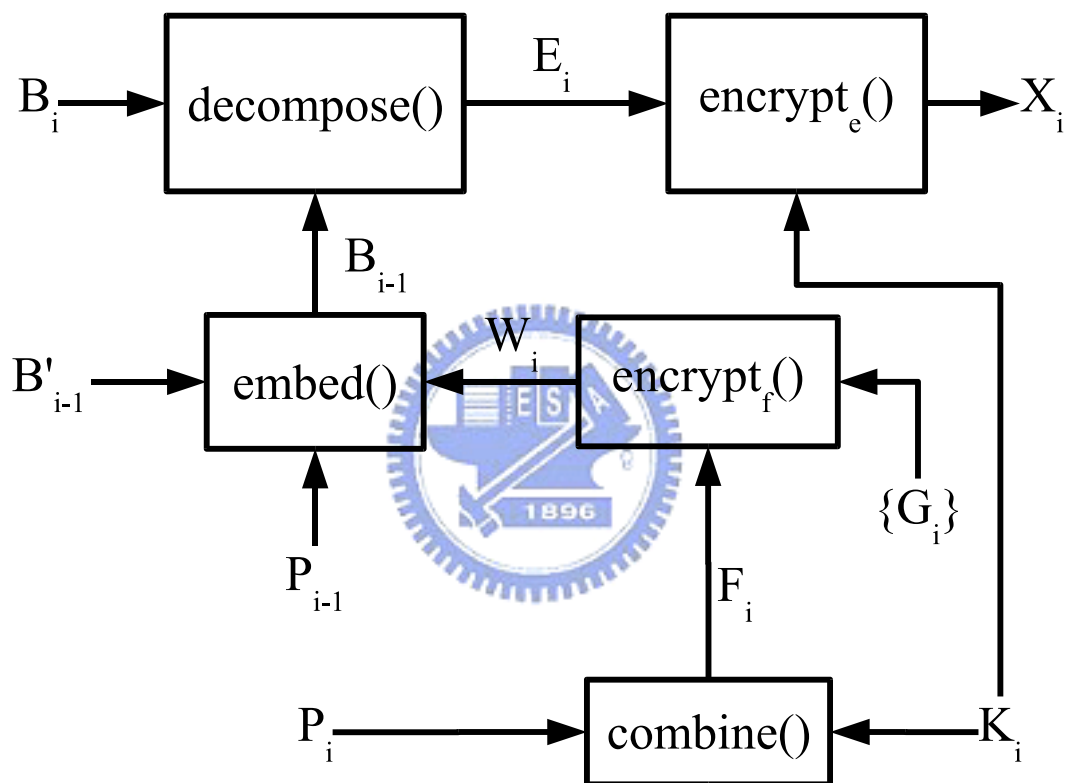


Figure 4.5: A layered protected image encoder.

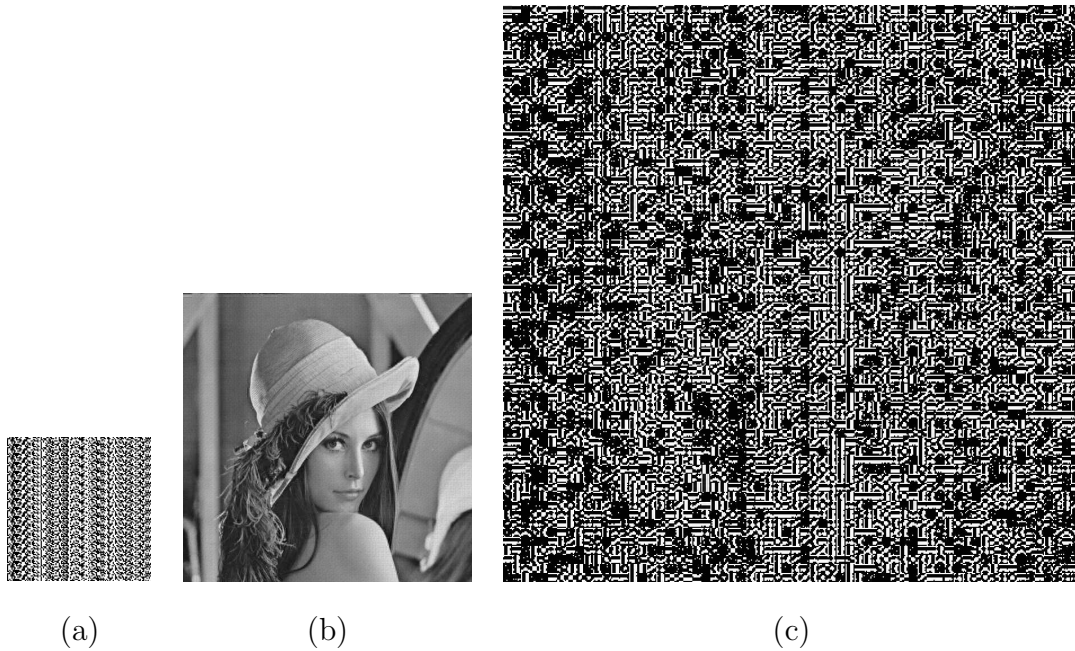


Figure 4.6: A layer-protected image for transmission. (a) The  $128 \times 128$  watermark of the encrypted string **NCTU-DEE**. (b) The watermarked base layer. (c) The  $1024 \times 1024$  enhancement layer.

secret information into the so-called “middle-frequency” coefficients. After training with genetic algorithms (GA), we obtain the appropriate frequency coefficients for embedding the secret information. We can see that both the watermark robustness and watermarked image quality can be retained; consequently, the secret can be correctly decrypted.

Fig. 4.6(b) and (c) show the transmitted base layer and the enhancement layers. Before transmission, the watermarked base layer has an acceptable visual quality, with the PSNR of 39.24 dB in Fig. 4.6(b). We then extract the watermark from the base layer picture, derive the decryption key, decrypt the transmitted enhancement data in the next layer, and finally reconstruct the original  $1024 \times 1024$  picture.

We then test the packet loss case on the base layer [49]. The packet loss rate in our simulations is set to 10%. The received image is shown in Fig. 4.7(b), and the extracted watermark is shown in Fig. 4.7(a). The distortion is within the tolerance



Figure 4.7: A received and reconstructed image from an erroneous channel. (a) The extracted  $128 \times 128$  watermark. (b) The received watermarked base layer which contains transmission errors. (c) The reconstructed  $1024 \times 1024$  image.

range of the extracted watermark, with the bit-correct rate of 92.74%. We then use the majority vote to produce the 8-byte secret, extracted encryption key, and decrypt the cipher-text. Finally, we can recover the original key information correctly. In this example, we do not employ any error concealment technique. Therefore, the  $1024 \times 1024$  picture is reconstructed with some defects as shown in Fig. 4.7(c).

## 4.5 Application 2: Scalable-Compressed Video

Depending on the scalable coding algorithm, the design of transmitter varies. Based on the proposed conceptual architecture, we implement another example, a Motion-JPEG-like application, to verify our proposed concept [50]. We design the scalability in the temporal domain in delivering the video data. Fig. 4.8 shows one such design. Robust watermarking method is again based on a modified version of the algorithm

proposed in [23]. As shown in Fig. 4.8, every four contiguous frames are grouped together to form a scalable group (group of pictures, GOP). In one GOP, we designate the first frame as the base layer, and the remaining three frames act as the enhancement layers. At the transmitter side, we generate watermarks which carry encryption keys. The first watermark is generated from the first data key. Then, we embed the first watermark, shown in Fig. 4.8, into the first frame. The same process is repeatedly applied to the second and third frames with different user and data keys. After the watermarking step, we compress all four frames using the JPEG compression standard. In order to resist to the transmission errors, we insert one restart codes for every MCU (minimal coded unit) to limit error propagation. The final step is to encrypt the second, third, and fourth frames using the data key1, key2, and key3, respectively.

In our simulations, we take the Foreman sequence ( $352 \times 288$ , 300 frames) and the Football sequence ( $352 \times 288$ , 260 frames) as the original sources. The embedded watermarks, corresponding to the data keys in Fig. 4.8, have size  $88 \times 72$  as shown in Fig. 4.9.

International multimedia standards, such as MPEG, specify only the decoder or receiver syntax and operations. Following the same philosophy, we suggest a receiver structure that matches the transmitter design in Fig. 4.10. The transmission errors are simulated as follows. At the receiver side, we receive the first frame, extract the embedded watermark, reconstruct the first encrypted data key, and decrypt it using the first user key. When we receive the encrypted second frame, we can decrypt it using the first data key, and reconstruct the second data key. The process performs iteratively, until all frames are reconstructed or an unrecoverable error occurs in decrypting or decoding. Because the error pattern relies on the underlying channel coding scheme and channel model, we assume the error pattern viewed at the source coding level is uniformly distributed and white. After the encoding process, we simulate the erroneous transmission by attaching uniformly distributed bit errors to the entire sequence. The simulations on the two test sequences are repeated 10



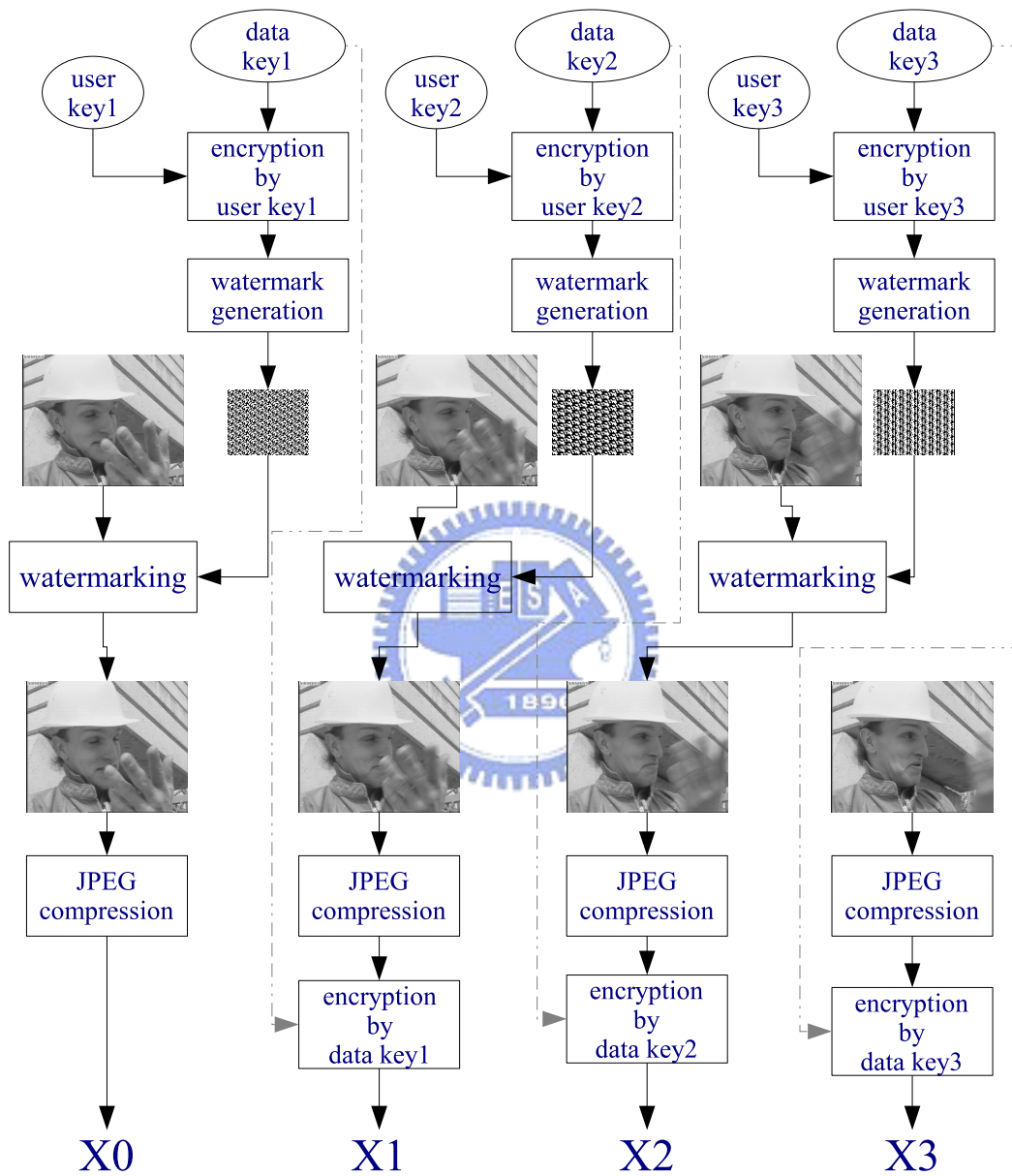


Figure 4.8: One design example for the transmitter.



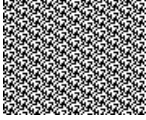

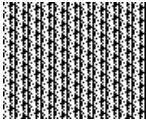
Data Keys	$\Rightarrow$	Generated watermark
Data Key1: <b>12345678</b>	User Key1: <b>CommLab1</b>	
Data Key2: <b>87654321</b>	User Key2: <b>CommLab2</b>	
Data Key3: <b>13572468</b>	User Key3: <b>CommLab3</b>	

Figure 4.9: Embedded watermarks, generated from encrypted keys, with a size of  $88 \times 72$ .

times (750 GOPs and 650 GOPs) with random errors to calculate the average BCR at three bit-error-rates:  $10^{-4}$ ,  $10^{-5}$ , and  $10^{-6}$ .

Fig. 4.11 shows the watermarked frames on the left column and the received (reconstructed) frames with transmission errors on the right column. The quality of watermarked frames is acceptable, and thus the watermarks are imperceptibly embedded. Fig. 4.12 displays an example chosen from one of the GOP simulations that can correctly decrypt the associated keys. After extracting the received frames in the right column of Fig. 4.11 with channel bit error rate (BER) of  $10^{-4}$ , we obtain the extracted watermarks in Fig. 4.12 with high bit-correction-ratios (BCR). Thus, the data keys can be correctly decrypted.

Table 4.1 shows the average watermark bit-correction-ratio for the simulations with erroneous transmission. The robustness of a watermark is measured in correctly recovered bits. The column of ideal BCR lists the results when no transmission errors occur. From the results, we can see that the higher the bit-error-rate, the lower the

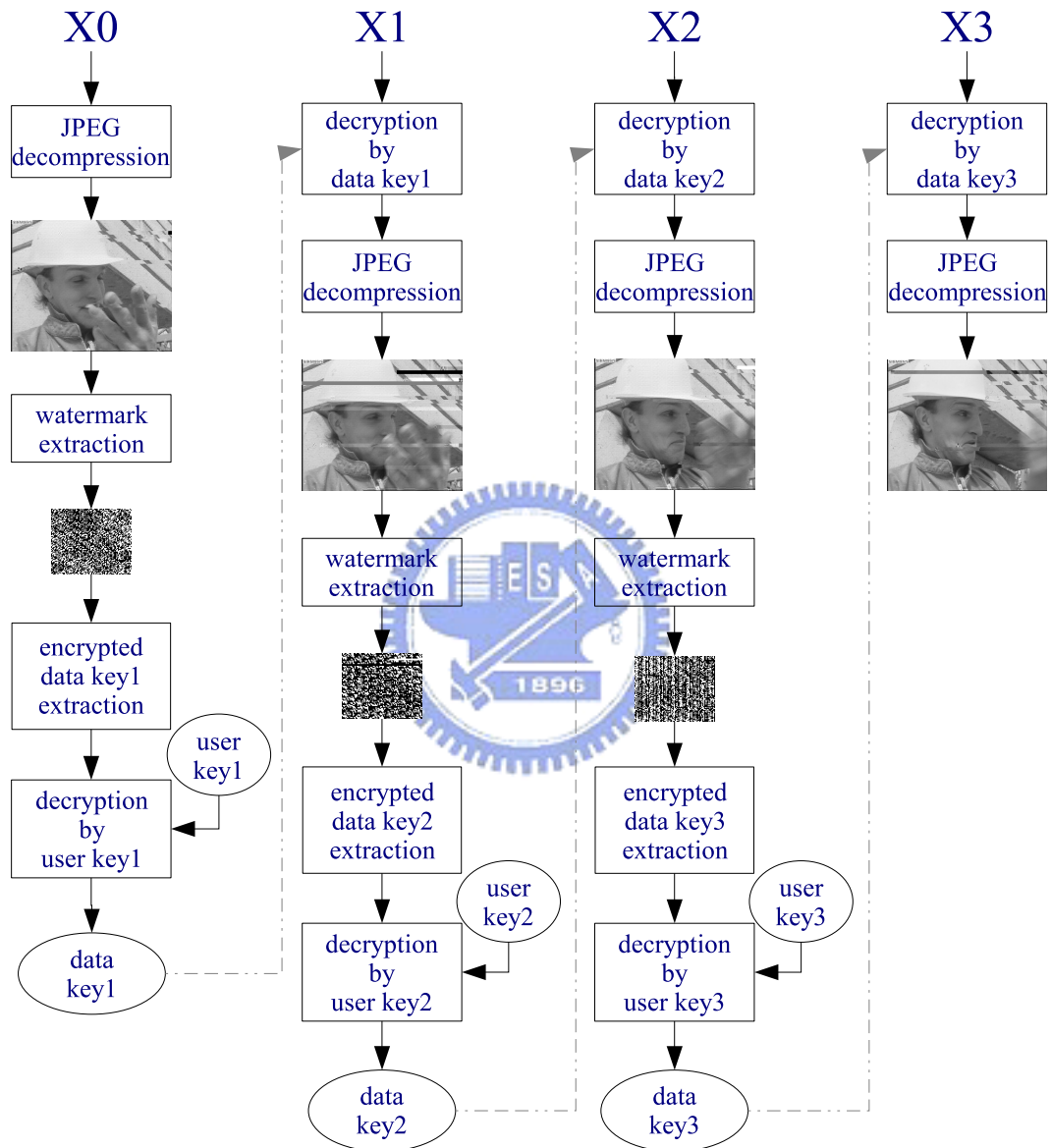


Figure 4.10: One design example for the transmitter.



Figure 4.11: Comparisons of image quality between the watermarked frames (left column) and the received frames (right column) over the channels with  $BER = 10^{-4}$ .




Extracted Watermark	$\Rightarrow$	Data Keys
	User Key1: <b>CommLab1</b>	Data Key1: <b>12345678</b>
	User Key2: <b>CommLab2</b>	Data Key2: <b>87654321</b>
	User Key3: <b>CommLab3</b>	Data Key3: <b>13572468</b>

Figure 4.12: Extracted watermarks from the received frames, and the successfully decrypted user keys.



BCR.

To measure the correctness of the recovered keys, we define the key-correction-ratio. An extracted key is correct when all its bits are identical to the original key. We can see from Table 4.2 that the higher the BER, the lower the KCR. Since the receiver is designed to block unauthorized access to the higher layers, an error key in one layer definitely blocks the extraction of the keys of the next layers. Thus, we may see that the KCR of key3 is not higher than that of key2, and so as key2 and key1.

## 4.6 Summary

In this chapter, we proposed a structure to protect the layered, or scalable, content in a broadcast environment. By combining cryptographic and scalable coding techniques, we can provide a group-based access control on the layers of the content. By incorporating robust watermarking techniques, the keys for decrypting enhancement

Table 4.1: Bit-correction-ratio after the JPEG compression and the data transmission.

<b>Foreman 75-GOPs (repeated 10 times)</b>				
BCR (Bit Correction Ratio %)				
	Ideal BCR		Average BCR	
	BER=0	BER= $10^{-4}$	BER= $10^{-5}$	BER= $10^{-6}$
watermark1	89.13	82.22	88.56	89.08
watermark2	89.67	76.55	88.20	89.49
watermark3	89.06	75.23	87.39	88.88

<b>Football 65-GOPs (repeated 10 times)</b>				
BCR (Bit Correction Ratio %)				
	Ideal BCR		Average BCR	
	BER=0	BER= $10^{-4}$	BER= $10^{-5}$	BER= $10^{-6}$
watermark1	89.58	83.79	89.03	89.51
watermark2	90.49	76.96	88.37	90.27
watermark3	89.38	74.96	87.48	89.18

Table 4.2: Key Correction Ratio (KCR) under various Bit Error Rates (BER).

<b>Foreman 75-GOPs (repeated 10 times)</b>			
	KCR (Key Correction Ratio %)		
	BER= $10^{-4}$	BER= $10^{-5}$	BER= $10^{-6}$
key1	90.53	98.53	99.33
key2	69.60	97.07	99.07
key3	58.67	95.60	98.80

<b>Football 65-GOPs (repeated 10 times)</b>			
	KCR (Key Correction Ratio %)		
	BER= $10^{-4}$	BER= $10^{-5}$	BER= $10^{-6}$
key1	87.85	99.85	100.00
key2	60.92	96.62	99.38
key3	46.92	94.92	99.38

data streams can be safely embedded in the content without changing the coding standard. The contribution in this paper is to propose an architecture that combines these techniques, and offers the advantages for scalable intellectual property protection.

In the proposed scheme, the encryption concept guarantees the access control, keeping away malicious eavesdroppers. Also, the embedding concept solves the key-content synchronization problem. Comparing to the conventional cipher-block chaining encryption, the robust watermarking concept increases the key robustness against transmission errors and distortion, so that it implicitly gives a higher data integrity protection on the keys than on the contents. To provide such features, this architecture requires more computing power than a multiple of single-layer encryption and watermarking system. For example, in addition to the decryption operations, additional complexity is needed to regenerate the keys at the receiver. The decryption keys are derived from the extracted watermarks, and the watermark extraction process is often costly. However, if the target application is a multiple grade service and thus each data layer has to be protected by separate security parameters/mechanisms, the computational complexity increase seems to be inevitable.

We also conduct two experiments based on our proposed architecture, one for spatial scalability and the other for temporal scalability. The experiments results show that our proposed scheme is feasible and promising.