Secure Transcoding with JPSEC Confidentiality and Authentication

Susie Wee, John Apostolopoulos
Mobile and Media Systems Laboratory
HP Laboratories Palo Alto

HPL-2004-185
October 20, 2004*

image security, transcoding, JPEG-2000, JPSEC, confidentiality, authentication, secure scalable streaming, secure transcoding

The emerging JPEG-2000 Part 8 Security standard (JPSEC) is being defined to provide security services for JPEG-2000 images [1, 2]. This paper describes how confidentiality and authentication can be applied in a manner that allows mid-network adaptation of protected JPSEC streams while preserving end-to-end security. We achieved this by designing the JPSEC syntax to support the principles of secure scalable streaming and secure transcoding [3,4,5]. Specifically, we designed JPSEC encryption methods and signaling syntax that enable an entity to securely adapt or transcode the resulting JPSEC-protected stream without requiring decryption. We discuss tradeoffs in protection, transcoding flexibility, and complexity for the different encryption methods. Furthermore, we show how authentication can be applied to verify that the secure transcoding operation was performed in a valid and permissible manner.

Invited paper in 'Media Security Issues in Streaming and Mobile Applications', IEEE ICIP 2004.

SECURE TRANSCODING WITH JPSEC CONFIDENTIALITY AND AUTHENTICATION

Susie Wee and John Apostolopoulos

Multimedia Communications and Networking Department
Hewlett-Packard Laboratories, Palo Alto, CA USA

ABSTRACT

The emerging JPEG-2000 Part 8 Security standard (JPSEC) is being defined to provide security services for JPEG-2000 images [1, 2]. This paper describes how confidentiality and authentication can be applied in a manner that allows mid-network adaptation of protected JPSEC streams while preserving end-to-end security. We achieved this by designing the JPSEC syntax to support the principles of secure scalable streaming and secure transcoding [3, 4, 5]. Specifically, we designed JPSEC encryption methods and signaling syntax that enable an entity to securely adapt or transcode the resulting JPSEC-protected stream without requiring decryption. We discuss tradeoffs in protection, transcoding flexibility, and complexity for the different encryption methods. Furthermore, we show how authentication can be applied to verify that the secure transcoding operation was performed in a valid and permissible manner.

1. INTRODUCTION

It is often desireable to give senders and mid-network entities, such as caching servers, the ability to adapt or transcode compressed images for downstream client capabilities or network conditions. The JPEG-2000 image coding standard [1] was designed for compression and scalability, where scalability refers to the ability to extract portions of a coded image from a JPEG-2000 codestream without requiring full decoding. This inherent scalability greatly facilitates adaptive streaming and transcoding for applications such as remote browsing and streaming [6, 7].

Current efforts in the JPEG-2000 Part 8 Security standard (JPSEC) are underway to incorporate security services into the JPEG-2000 standard [2]. These include many aspects of security such as confidentiality [8], authentication [9], integrity, conditional access [10, 11], and ownership protection. These security services are achieved using techniques such as encryption, authentication, key generation and management, scrambling, and watermarking.

Protecting the confidentiality of a JPEG-2000 codestream means encrypting the data so that eavesdroppers can not decode the image. The simplest way to do this is to encrypt the entire codestream and to do so with a key that is only held by the allowed users. This protects the data from eavesdroppers who do not have the key. It also provides the desirable property of end-to-end security, where the content is locked by the sender, unlocked by allowed receivers, and remains encrypted at all points in between.

When the entire codestream is encrypted, the inherent scalability property is lost, and the only way to transcode the stream is to give the transcoder the key to allow it to decrypt and transcode the content. In this case, every transcoder becomes a possible point of attack for the system. Furthermore, this mid-network decryption compromises the end-to-end security of the system.

Secure scalable streaming (SSS) and secure transcoding are technologies that simultaneously enable the seemingly conflicting goals of end-to-end security and mid-network transcoding [3, 4]. This is achieved by co-designing the coding, encryption, and packetization in a manner that allows transcoding to be performed without decryption, which we call secure transcoding. SSS and secure transcoding were applied to JPEG-2000 in [5]. In this paper, we show how we designed JPSEC to provide confidentiality and authentication in a manner that allows mid-network transcoding of protected JPSEC codestreams while preserving end-to-end security.

A JPSEC secure transcoding system is shown in Figure 1. An original image is encoded into a JPEG-2000 codestream, which is then locked with a key to form a JPSEC codestream. This JPSEC codestream can be transcoded without the key to form a transcoded JPSEC codestream. The original and transcoded JPSEC codestreams can be unlocked and decoded by authorized entities with the appropriate key to reconstruct the full and transcoded images.

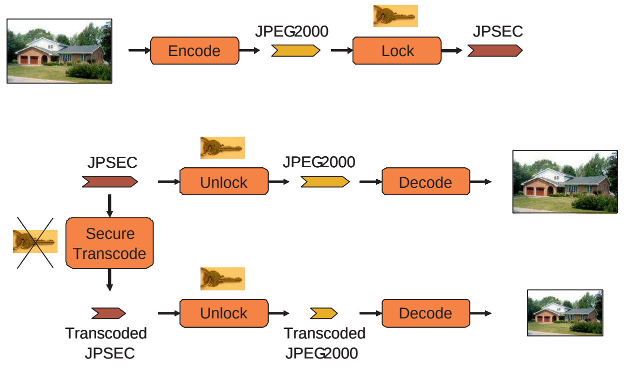

Fig. 1. Secure transcoding can be achieved with JPSEC.

2. JPSEC OVERVIEW

JPSEC provides security services for JPEG-2000 by defining a number of protection tools, such as encryption and authentication, that can be applied to JPEG-2000 codestreams. One or more protection tools can be applied to a single stream. The resulting JPSEC codestream, shown on the right of Figure 2, contains signaling information for the protection tools and the modified data that may have resulted from their application. The signaling information is placed in a SEC (security) marker segment that is added to the JPEG-2000 header, and it includes the parameters that a JPSEC consumer needs to interpret and process the protected stream.

JPSEC provides protection templates for decryption, authentication, data integrity, and scrambling that can be used to protect a JPEG-2000 codestream. These templates specify parameters for these protection methods, e.g., a decryption template can specify using AES decryption in cipher block chaining mode and an authentication template can specify using HMAC with SHA-1. For each template protection tool, the signaling information conveys three types of information:

- Zone of Influence: Describes the area influenced by the protection tool using *image- and bitstream-related parameters*.
- Template parameters: Specify the template type and its parameters.
- Processing parameters: Describe how the protection method is applied to the data, including the domain where the data is processed and the granularity with which it is applied.

The Zone of Influence (ZOI) can serve a number of purposes. First, it tells a JPSEC consumer which part of the data to process by using the ZOI's bitstream-related parameters. For example, in the case of decryption it can tell a JPSEC consumer to decrypt bytes 0 through 2078 using the specified decryption method. The ZOI's bitstream-related parameters can specify data by byte ranges or JPEG-2000 packet indices.

In addition, the ZOI can be used to semantically describe the image area influenced by the protection method by using the ZOI's image-related parameters. For example, in the case of authentication the ZOI can be used to indicate that an authentication code has been computed for the lowest-resolution component, so a JPSEC consumer can verify that low-resolution transcoding was performed in a permissible manner. Also, in the case of selective encryption, the ZOI can indicate that the higher resolution levels have been protected and thus need a key to be decrypted. The ZOI's image-related parameters can specify a combination of image areas, tiles, precincts, color components, resolution levels, and quality layers.

Finally, the ZOI can be used to identify the correspondence between image features and bitstream segments by specifying both image-related parameters and bitstream-related parameters. For instance, if the image-related parameters specify that the low-resolution component of the image was encrypted and the bitstream-related parameters specify a list of byte ranges that should be decrypted, then the ZOI is indicating that the specified list of byte ranges contain the low-resolution component. This ability to represent correspondences between image features and bitstream segments is a key feature of the ZOI and of JPSEC. This is an important feature for enabling low-complexity, secure transcoding.

3. DESIGNING ENCRYPTION & AUTHENTICATION TO SUPPORT JPSEC SECURE TRANSCODING

A key feature of JPEG-2000 is the scalability that it provides. Specifically, the standard was designed for easy access to subsets of coded image data; this greatly facilitates operations such as adaptive streaming and transcoding. JPSEC provides protection tools for JPEG-2000 codestreams. However, when applying these tools one must be aware of the impact on JPEG-2000 features such as scalability.

This section describes how encryption and authentication can be applied in JPSEC to enable secure transcoding. We consider the system shown in Figure 1, where a *lock* module creates the JPSEC protected codestream from a JPEG-2000 codestream, and an *unlock* module performs the inverse procedure that can be applied to the original or transcoded JPSEC codestream. In this example, the lock module provides two security services: encryption and authentication. The unlock module is the reverse operation and has two parts: verify authentication and decryption.

3.1. Encryption and Secure Transcoding

Transcoding a JPEG-2000 codestream involves parsing the header data to learn some of the coding parameters of the stream, then extracting or discarding the appropriate JPEG-2000 packets (terminology note: JPEG-2000 packets are different from network packets) according to the desired transcoding operation. For example, if a streamer or transcoder wishes to extract the lowest-resolution component of the codestream, then it must determine the number of resolutions, layers, components, and precincts in the image along with the data ordering information (e.g., RLCP or CPRL) and it must find the JPEG-2000 packet boundaries. With this information, it can extract only those packets that contain the low-resolution information.

A JPEG-2000 codestream can be encrypted in a number of ways. Each encryption method has different implications on the privacy and transcoding flexibility of the protected codestream and on the complexity requirements of JPSEC creators, streamers, transcoders, and consumers. These requirements are especially critical for servers that adaptively stream and transcode large numbers of streams and thin clients that have limited device capabilities. We describe a few possible encryption methods and discuss their implications.

File-based encryption: Conventional file-based encryption can be used to encrypt the entire JPEG-2000 codestream. The encryption and decryption operations are straightforward, and the resulting stream has the highest level of security because it is completely protected. However, this prohibits any form of transcoding and loses all the functionalities of JPEG-2000. For example, while JPEG-2000 provides the capability of discarding portions of the codestream, if file-based encryption is used and a portion of the JPEG-2000 codestream is discarded, the remaining portion may be undecryptable, unauthenticatable, and undecodable, and hence useless. If an entity did want to transcode the stream, it would need the key to first decrypt it; this breaks the end-to-end security of the system. For this reason, secure transcoding is not possible with file-based encryption.

We next discuss two JPSEC encryption methods that support secure transcoding.

Packet-body encryption: JPSEC allows a packet-level encryption method where only JPEG-2000 packet bodies are encrypted. The idea is presented in [8] in the context of selective encryption. Since only packet bodies are encrypted, all of the header information including coding parameters and packet headers are left exposed. Thus, with detailed knowledge of the JPEG-2000 syntax, an entity can transcode the protected codestream in the same manner as an unprotected JPEG-2000 codestream. This method provides privacy protection for the image data, but does not protect the coding parameters and packet headers. The encryption/decryption operations are somewhat complex in that they require individually extracting the packet bodies before encryption/decryption. The transcoding complexity is high because it requires the full parsing and packet extraction operations, however it allows full transcoding flexibility.

Segment-based encryption with metadata [5]: We designed a JPSEC encryption method that 1) allows segments of JPEG-2000 data to be encrypted together across packet headers and bodies and 2) inserts metadata in the SEC marker segment that provides information about the encrypted data segments. A JPSEC creator can use this method to encrypt data segments that contain certain image features, such as the low-resolution component or lowest quality layer of the image. Because larger segments of data are encrypted together across packet headers and bodies, this lowers the complexity of the encryption/decryption operation as compared to the packet-body encryption method. The metadata can include both the image-related and bitstream-related parameters in the Zone of Influence to represent the correspondence between encrypted data segments and image features. An entity can use these ZOI parameters to securely transcode the JPSEC codestream by reading the parameters and then extracting the appropriate data segments; thus, this method greatly reduces the complexity of the transcoder. On the other hand, it also reduces the transcoding flexibility of the JPSEC codestream because it only allows access to the image units that were made accessible by the JPSEC creator. An added benefit is that because the transcoding information is placed in the metadata, it also allows the

JPSEC creator to encrypt many of the JPEG-2000 marker segments and hide the boundaries of individual packets. Thus, this method allows improved privacy and reduced transcoding complexity over packet-body encryption at the expense of transcoding flexibility.

Table 3.1 summarizes the tradeoffs of these methods in terms of privacy, transcodability, create/consume complexity, and transcoding complexity. File-based encryption provides the highest level of protection and has the lowest encryption/decryption complexity, but it does not allow secure transcoding, i.e., transcoding without decryption. At the other extreme, packet-body encryption provides full transcoding flexibility similar to transcoding unprotected JPEG-2000 codestreams, since only the coded image data itself is encrypted, and the bits describing the coding parameters are left unencrypted. However, the creation, consumption, and transcoding complexity are relatively high. The approach using segment-based encryption with metadata hits the middle ground in terms of privacy, transcodability, and create/consume complexity. Furthermore, it enables the lowest complexity transcoding, and thus is most suitable for transcoding large numbers of streams. In the next section, we show examples of this approach. Note that file-based encryption does not use metadata; packet-based encryption uses only the metadata in the JPEG-2000 header; while the segment-based approach uses additional metadata, such as ZOIs for each segment.

Method	Privacy	Trans-codability	Create/Consume Complexity	Transcode Complexity
File	Most	No	Low	N/A
Segment	Middle	Partial	Middle	Very low
Pkt body	Least	Full	High	High

Table 1. JPSEC encryption methods have different implications on the privacy and transcodability of protected codestreams and the complexity requirements of JPSEC creators, consumers, and transcoders.

3.2. Authentication

Authentication can be performed with a number of authentication tools that compute the Message Authentication Code (MAC) for a subset of data using an authentication key. This MAC value is sent to the receiver with the data. The receiver then uses the authentication key to compute the MAC of the received content and compares it with the received MAC value in order to verify the authenticity of the received content. Since the MAC is cryptographically locked, it is not practically possible for an attacker to manipulate the data and compute a new MAC which will pass authentication at the receiver without knowledge of the key. In our system we performed the authentication using HMAC, which is a keyed hash algorithm, using SHA-1 as the hash. However, other authentication methods can also be used, such as cipher-based MACs or digital signatures.

The JPSEC creator computes a MAC for each allowed transcodable result and stores these MACs in the JPSEC marker segment. A JPSEC consumer uses these MACs to verify that the received JPSEC stream was transcoded in a valid and permissible manner. It is often useful to truncate the MACs in order to provide a tradeoff between reducing the overhead from the size of the MAC and the probability of an undetected alteration.

Encryption and authentication can be combined in two ways: the encryption may be performed first, followed by authentication of the ciphertext; or alternatively the authentication may be performed on the plaintext, followed by encryption. Both approaches have merits and disadvantages, and the preferred approach often depends on the particular application. Note that JPSEC does not constrain the order of the protection services, and SSS and Secure Transcoding is possible with either order.

4. EXAMPLES OF SECURE TRANSCODING WITH JPSEC

In this section, we describe two example usages of secure transcoding with JPSEC using *segment encryption with metadata*. In the first example, the original JPEG-2000 codestream data is ordered such that the transcodable units are in contiguous segments. In the second example, the original data does not have contiguous transcodable units, therefore the data is reordered as part of the JPSEC creation and consumption processes and this reordering is signalled in the JPSEC stream.

4.1. Example usage 1

In this example, the original JPEG-2000 codestream is in RLCP ordering and the goal is to protect this stream with encryption and authentication while enabling secure transcoding by resolution on the protected codestream. Figure 2 shows the syntax of a JPEG-2000 and JPSEC codestream. Since the original JPEG-2000 codestream used an RLCP ordering, each resolution component is represented by a contiguous data segment. Thus, encryption can be performed on three contiguous data segments, and this is specified in the JPSEC header. Thus, the JPSEC header specifies three zones of influence, each describing the resolution component, bitstream segment, and encryption template used for each segment. Authentication can also be performed on each of the three data segments, either before or after encryption depending on the desired functionality. This protection method is also specified in the SEC header using the authentication template.

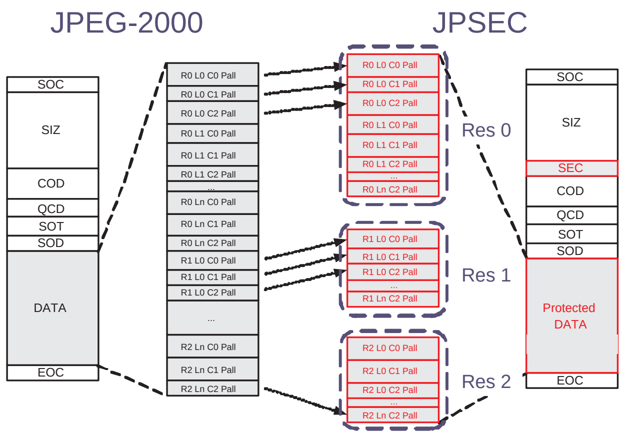

Fig. 2. Example 1: Transcodable units are contiguous in the bitstream.

In order to perform secure transcoding on the JPSEC codestream, a transcoder simply reads and parses the SEC header, identifies the locations of the resolution segments, and then retains or removes the appropriate data segments/resolutions. Notice that this transcoding operation corresponds to a simple parsing operation and that it does not require unprotecting the data. Therefore, secure transcoding-by-resolution is achieved. Authentication is performed as well by authenticating the received transcoded data with the MAC value that is placed in the SEC header during the JPSEC protection process.

Also note that with RLCP ordering this same approach can provide finer-grain transcoding, e.g., by resolution and quality layer, because it is also a contiguous unit in the bitstream.

4.2. Example usage 2

Figure 3 shows another example use case of segment-based encryption with metadata. Once again, the desired operation is to protect the codestream while allowing transcoding by resolution, however this example is slightly more complex in that the original JPEG-2000 codestream is in PCRL ordering, so the data segments corresponding to the three resolution components are not contiguous. In order to use segment-based encryption to achieve secure transcoding by resolution, the JPSEC creator first reorders the data so that the resolution components are in contiguous segments. It then encrypts the data as described in the previous usage example. Further signaling is used to inform JPSEC decoders of the permutation so that they can undo the operation.

The original JPEG-2000 codestream and the protected JPSEC codestream are shown in Figure 3. In this example, the signaling information is included both in the SEC header and in the data.

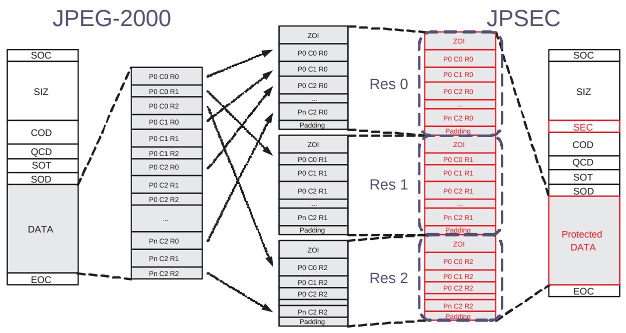

Fig. 3. Example 2: The transcodable units are not contiguous in the original bitstream, so the data units are rearranged into contiguous transcodable units. The rearrangement is specified in the ZOI parameters placed at the beginning of each transcodable unit.

The main SEC header contains ZOI parameters that describe the image-related and bitstream-related parameters associated with each data segment. In this example, after the data reordering there are three contiguous data segments that contain each resolution level. There are also additional ZOI parameters within each data segment; these ZOIs contain bitstream-related and packet-related parameters that describe the data reordering by specifying the packet information. When the start of packet headers are not used or when encryption is being performed across the packet headers and bodies, these ZOIs can be used to specify packet boundaries within the segment. This information may be used by a decoder to reassemble the data packets to the ordering used in the original JPEG-2000 codestream, or by a transcoder to perform fine-grain transcoding operations.

In this example, the SEC header contains the coarse-grain data segment information that describes corresponding image-related and bitstream-related parameters for each data segment. A secure transcoder can use this information to extract any resolution level of the codestream without decrypting the data. In this sense, the end-to-end security of the system is preserved.

4.3. Experimental results

The JPSEC system shown in Figure 1 was implemented in software. Figure 4 shows decrypted and decoded JPSEC images. A high-resolution image was encoded and encrypted into a JPSEC stream using the method described in example usage 1 with AES encryption applied in CTR mode, and authentication with HMAC using SHA-1. The data was encrypted with a granularity of resolution layer. The full JPSEC stream was decrypted and decoded to form the high-resolution image shown. The JPSEC stream was securely transcoded into a medium-resolution and low-resolution JPSEC stream using the protection tool parameters specified in the SEC marker segment. These JPSEC streams were decrypted, authenticated, and decoded to form the medium- and low-resolution images shown.

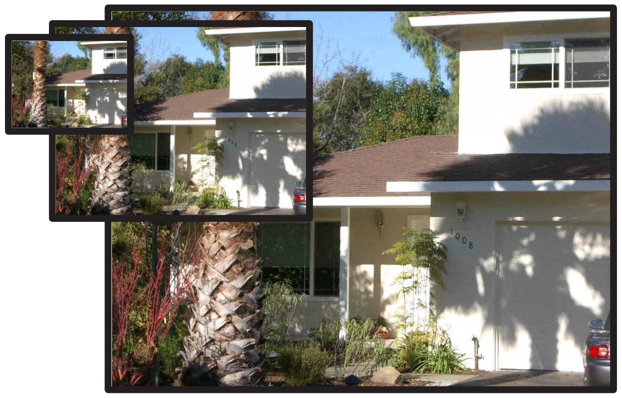

Fig. 4. The low- and medium-resolution JPSEC images were securely transcoded from the high-resolution JPSEC image.

5. CONCLUSION

This paper describes secure scalable streaming and secure transcoding with JPSEC, which enables the two seemingly conflicting properties of end-to-end security with secure transcoding at mid-network nodes. We achieve this by designing the JPSEC encryption methods and signaling syntax to enable an entity to securely adapt or transcode the resulting JPSEC-protected stream without requiring unprotecting or decrypting the content. Furthermore, this method also provides authentication that the transcoding was performed only in a valid and permissible manner. This enables a potentially untrusted server or mid-network node such as a proxy to perform secure transcoding and enables a JPSEC consumer to authenticate that the received content was transcoded properly.

6. REFERENCES

[1] *JPEG-2000 Image Coding System, ISO/IEC FCD15444-1:2000*, March 2000.

[2] *JPSEC Committee Draft version 1.0, ISO/IEC JTC1/SC29/WG1N3297*, April 2004.

[3] S.J. Wee and J.G. Apostolopoulos, “Secure scalable video streaming for wireless networks,” *IEEE ICASSP*, May 2001.

[4] S.J. Wee and J.G. Apostolopoulos, “Secure scalable streaming enabling transcoding without decryption,” *IEEE ICIP*, Oct. 2001.

[5] S.J. Wee and J.G. Apostolopoulos, “Secure scalable streaming and secure transcoding with JPEG-2000,” *IEEE ICIP*, Sept 2003.

[6] D. Taubman, “Remote browsing of JPEG-2000 images,” *IEEE ICIP*, September 2002.

[7] S. Deshpande and W. Zeng, “Scalable streaming of JPEG-2000 images using hypertext transfer protocol,” *ACM Multimedia*, October 2001.

[8] Y. Sadourny and V. Conan, “A proposal for supporting selective encryption in JPSEC,” *IEEE Transactions on Consumer Electronics*, vol. 49, no. 4, pp. 846–849, November 2003.

[9] Z. Zhang, Q. Sun, G. Qiu, Y.Q. Shi, and Z. Ni, “A unified authentication framework for JPEG2000,” *IEEE ICME*, 2004.

[10] R. Grosbois, P. Gerbelot, and T. Ebrahimi, “Authentication and access control in the JPEG 2000 compressed domain,” August 2001.

[11] Y. Wu, D. Ma, and R. Deng, “Progressive protection of jpeg2000 codestreams,” *IEEE ICIP*, 2004.