SECURED MPEG-21 DIGITAL ITEM ADAPTATION FOR H.264 VIDEO

Razib Iqbal, Shervin Shirmohammadi, and Abdulmotaleb El Saddik

Distributed and Collaborative Virtual Environments Research laboratory (DISCOVER Lab)
School of Information Technology and Engineering (SITE)
University of Ottawa, Ottawa, Ontario, Canada.
[riqbal | shervin | elsaddik] @site.uottawa.ca

ABSTRACT

Seamless adaptation and transcoding techniques to adapt the digital content have achieved significant focus to serve the consumers with the desired content in a feasible way. With the succession of time we sense that secured adaptation should also be taken care of for not only serving sensitive digital contents but also to offer security as an embedded feature of the adaptation practice to ensure digital right management and confidentiality. In this paper, we propose an encryption framework for a transcoder while adapting H.264 video conforming to MPEG-21 DIA. Encryption mechanism is applied on the adapted video content thus reducing computational overhead compared to that on the original content.

1. INTRODUCTION

The diversity of devices in both wired and wireless network via which multimedia contents are desired to be accessed and interacted with has grown significantly. The requirement for these devices differs in terms of bitrate, resolution and color. The driving concept of Universal Multimedia Access (UMA) depicts that other than having multiple types of the same content, it is preferred to have the original content adapted accordingly. This concept motivates the initiatives of the ISO/IEC 21000 Multimedia Framework, which aims to enable transparent and augmented use of multimedia resources across a wide range of networks and devices.

In terms of security and digital rights management, watermarking has been used in multimedia arena for assessing authenticity and/or confidentiality. To restrain eavesdroppers and to maximize the confidentiality, digital content should be encrypted while adapting. Simplified secure adaptation technique deserves attention for the reason that tradeoff between bulky data and slow speed, compression and encryption, and format compliance need to be considered for real time video processing and streaming. If the video is encrypted before adaptation, it is inconvenient for the transcoders to perform adaptation. Conventional transcoders apply the programmability to adapt digital contents by decompression, adaptation and re-compression steps. So an easy solution to security and adaptability can be seen as allocating some trusted intermediary nodes to perform manifold operations like decryption, decompression, adaptation, re-compression and re-encryption. But if the adaptation can be performed in the compressed domain without decompression and re-compression, it can be considered as a significant improvement. In our implementation, we have executed adaptation and encryption at the media resource server where we have embedded the encryption engine as a standalone module in the transcoder and encryption is performed on the compressed adapted bitstream.

Rest of the paper is organized as follows. Section 2 highlights the features of MPEG-21 Digital Item Adaptation and ITU-T H.264. Relevant encryption preliminaries are described in Section 3. Some related work is provided in Section 4, while the design of our system is depicted in Section 5. Section 6 illustrates implementation and result. Finally, we draw the conclusion in Section 7.

2. MPEG-21 DIA AND H.264

ISO Digital Item Adaptation (DIA), Part 7 of MPEG-21 standard [1], which has recently been finalized as part of the MPEG-21 Multimedia Framework, provides tools for adapting multimedia content. In the framework, a bit stream and all its relevant descriptions are denoted as a Digital Item. Generic BitStream Description (gBSD) [1] provides a flexible description method to describe the resource's high level structure. A Digital Item is subject to a resource adaptation engine and a description adaptation engine, which together produce the adapted Digital Item, whereas the adaptation engines are non-normative tools of DIA. MPEG-21 DIA are particularly advantageous in secure adaptation because intermediary nodes need not be aware of the content or the bitstream while applying manifold transcoding operations during adapting and/or encrypting.

H.264, the latest coding and compression standard in the digital-video lexicon is currently dominating the field by offering a flexible architecture and compression gain of up to 50% [2]. H.264/AVC design covers Video Coding Layer (VCL) which efficiently represents the video content; there is also Network Abstract Layer (NAL) which formats VCL presentation for the appropriate conveyance by particular transport layers or storage media. Advanced compression technique, improved perceptual quality, network friendliness and versatility of the codec [3], [4] drives it to outperform all the previous video coding standards. As streaming video is bandwidth intensive and time consuming to process so it is believed that H.264 is opening a new era of speed and storage options for the media resource providers.

3. ENCRYPTION

Encryption is a process of scrambling or converting data known as Plaintext into a form, called a Ciphertext that cannot be interpreted by an unintended user or receiver. Now, depending on the type of plaintext, data encryption systems can be classified as text encryption, audio encryption, image encryption and video encryption. Significantly of course, the encryption algorithms for digital video work mostly in the compressed domain. The simplest way to encrypt a video is perhaps to consider the whole stream as a 1-D array and then encrypt this 1-D stream with the encryption key. However with advancement of time, video encryption mechanism considering sensitivity of digital video has been developed for layered video compression techniques. Partial or selective encryption algorithms [5] [6] are deployed on selective layers. In this case the content is encrypted and thus decrypted exploiting the inherent properties of the video coding layer(s). The notion of perceptual encryption refers to partially degrading the visual data of the video content by encryption. Many researches have already been performed [7-10] for video encryption based on perceptual encryption and the researchers have successfully offered Scalability-based perceptual encryption, Perceptual encryption for wavelet-compressed images and videos, Perceptual encryption of motion vectors in MPEG-videos etc. We prefer to use the term perceptual encryption for our adaptation framework because after encrypting the Macroblocks containing the motion vectors, the perceptual quality of the content is degraded and the goal is achieved.

4. RELATED WORK

The conventional cascaded decoding/re-encoding scheme mentioned in [11] performs the full decoding of the input bit stream and then carries out resizing and/or reordering of the decoded sequence before fully re-encoding it. An alternative method of adapting encrypted streams through intelligent tagging and layering of the data content is introduced in [12]. Here separate layers of encrypted data are being sent to adapt without decrypting. For video transmission specific adaptation, additional layers with the low resolution data streams is composed to produce high resolution version. So in adaptation decision making engine, there is flexibility to drop out color components other than dropping the whole frame from a group of pictures. A methodology for Multimedia Digital Right Management which is applicable to the mobile terminal software applying MPEG-4 video is illustrated in [13]. Authors have designed two types of encryption methods which are applicable to the mobile handset. The first was an encryption of the starting 8-byte segment of macroblocks (MBs) of I-VOPs (Video Object Planes) extracted from the MPEG-4 file format applying DES (Data Encryption Standard). The second was an encryption of the starting 8-byte segment of MBs or motion vectors (MVs) of P-VOPs with the same extraction scheme as the first method. Lastly, a third way combining the previous two is experimented and suggested. Secure Scalable Streaming (SSS) framework discussed in [14] enables transcoding without decryption. It encodes video into secure scalable packets using jointly designed scalable coding and progressive encryption techniques. This combination allows downstream transcoders to perform transcoding operations by truncating or discarding packets, and without decrypting the data. It is worth mentioning that SSS framework was designed for scalable coders but in [15] authors showed the applicability of SSS framework to non-scalable coders. The later frame work enables secure transcoding of the H.264/MPEG-4 AVC (Advanced Video Coding) by encoding media into secure scalable packets thus allowing potentially untrusted nodes to perform transcoding. In the literature we did not encounter any dynamic secure adaptation framework applying MPEG-21. The objective of this paper is to study and propose a framework for adaptation in conjunction with a way to provide secure H.264 video using encryption conforming MPEG-21 DIA.

5. DESIGN

To perform secure adaptation in some intermediary node or in the media resource server, it is essential to have the bitstream along with its gBSD while it is being transmitted [See Section 2]. Fig. 1 below gives an overview of the whole architecture.

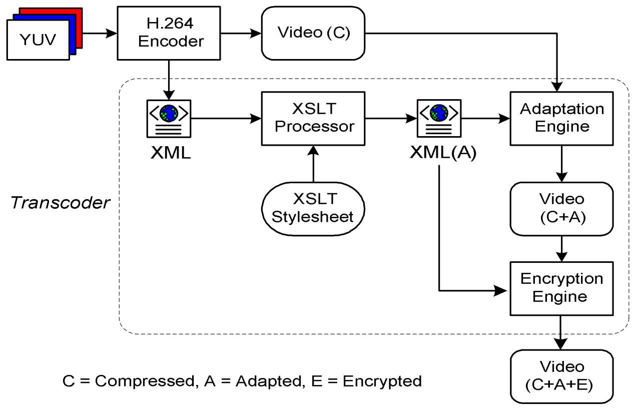


Fig. 1: Architecture of the secured H.264 Video Adaptation

In the design, gBSD of each compressed video is generated while it is being encoded inside the encoder. Resulting video (Video(C)) and its gBSD in the form of XML [16] shape the Digital Item for further use. Inside the transcoder, we have performed temporal adaptation of the Digital Item. The user or device requirement in terms of bitrate is considered as an external input. gBSD is transformed to adapted gBSD (XML(A)) by means of XSLT [17]. An adaptation engine is thus designed to generate transformed/adapted H.264 compliant video by discarding gBSD portions corresponding to specific frames and updating information. Output from the adaptation engine is the adapted H.264 video (Video(C+A)).

The first step in encrypting a bitstream is to obtain the encryptable plaintext with a known length in bits for each logical unit [18]. Macroblocks containing the motion vectors in each frame are our logical units. The encryption engine considers the level of encryption based on user's preference. If the encryption preference is highest then all the frames in the adapted video are encrypted; else, after parsing the importance level of each frame, only certain frames are being encrypted. For encryption, at first frame marker is scanned from the XML description and all the frames along with their subordinate frames having the required importance level are considered. Thus each macroblock's starting position and corresponding length is retrieved from the transformed gBSD for those frames which need to be encoded. To encrypt the macroblocks, an 8-bit encryption key is chosen and XOR operation is performed. After XOR operation, the bitstream is processed as usual for H.264 format compliance. It is worth mentioning that each logical unit can be encrypted independently.

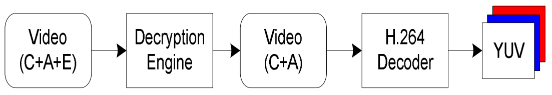


Fig. 2: Decrypting the encoded video

For decrypting the encoded video, the end node player should have the decryption engine embedded as shown in fig. 2. The operation inside the decryption is just like that of encryption engine.

6. IMPLEMENTATION AND RESULT

We have modified ITU-T reference software implementation of JM9.6 H.264 encoder available at [19] to generate the gBSD. While the encoder performs compression from the YUV input, at the same time knowledge of the frame start in terms of picture start code (PSC), frame length, number of macroblocks in each frame, start and length of each macroblock is assembled in an XML. Sample XML representation of the H.264 bitstream is shown in fig. 3. Frame rate of the H.264 video is 30fps. For transcoding, the target frame rate is an external input to the transcoder. The first step is to transform the XML (i.e. gBSD). An XSLT processor is thus designed and embedded in the transcoder. An XSLT style sheet and the original XML is the input to the XSLT processor. Based on the style sheet, the XSLT processor generates the adapted XML. Inside this adapted XML, frames that need to be parsed to generate adapted bitstream are kept and others are discarded.

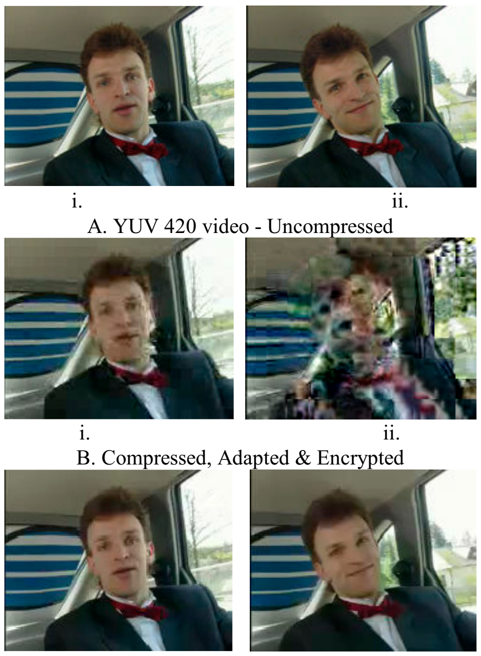
```
<?xml version="1.0" encoding="UTF-8" ?>
<dia:DIA xmlns="urn:mpeg:mpeg21:2003:01-DIA-gBSD-NS" xmlns:dia="urn:mpeg:mpeg21:2003:01-
DIA-NS" xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
 <dia:Description xsi:type="gBSDType">
  <Header>
            <ClassificationAlias alias="MV4" href="urn:mpeg:mpeg4:video:cs:syntacticalLabels"/>
            <!-- ...and so on... -->
  </Header>
  <gBSDUnit syntacticalLabel="Frame-1" start="21" length="1553" marker="importance-1">
   <Parameter name=”MB-1:I-Frame” length=”4”></Parameter>
   <!-- ...and so on... -->
   <Parameter name=”MB-99:I-Frame” length=”4”></Parameter>
  </gBSDUnit>
  <gBSDUnit syntacticalLabel="Frame-2" start="1574" length="1077" marker="importance-1">
   <Parameter name=”MB-1:P-Frame” length=”4”></Parameter>
   <!-- ...and so on... -->
   <Parameter name=”MB-99:P-Frame” length=”4”></Parameter>
  </gBSDUnit>
 </dia:Description>
</dia:DIA>
```

Fig. 3: XML representation of original H.264 bitstream

The adapted XML and the original H.264 video are the input to the adaptation engine. The adaptation engine parses the XML first. For this purpose an XML parser is designed inside the engine. While parsing the XML, each frame's starting byte and length is extracted and thus the corresponding bitstream is copied from the original H.264 to adapted H.264. Finally, other relevant layer information like NAL units is updated to make it H.264 format compliant.

Inside the encryption engine, the adapted XML and the adapted video is parsed again to extract the frame importance level and macroblock information. Finally, encryption is performed on the adapted bitstream. In above, we have mentioned that all the frames along with their subordinate frames having certain importance level are encrypted. The reason behind it is that frames are classified and compressed as I-, B- and P-frames where the reconstruction of the B-frames and the P-frames are dependent on the availability of the preceding I-frame. So the simplest decision might be to encrypt only the I-frames to protect the video stream content. But it has been figured out that a large portion of the encrypted video can be recovered from the unencrypted I-blocks in the P and the B frames and partially because of inter-frame correlation.

The encrypted video delivered to the consumer is H.264 format compliant and any standard H.264 player would be able to decode and play the video. But the frames those which are encrypted would not be perceived clearly because of the encryption. 2 frames of the results from the sample video 'Carphone' is shown in fig. 4. Fig. 4A. shows 2 frames from the raw video input. Fig. 4B. consists of previous 2 frames after compression, adaptation and encryption. For the frame 4B.i., importance level was not prioritized so this frame was not encrypted but the frame 4B.ii was encrypted. Fig. 4C. is the output of the video frames after being decrypted and decoded.



i. ii.

A. YUV 420 video - Uncompressed

i. ii.

B. Compressed, Adapted & Encrypted

i. ii.

C. Compressed, Adapted & Decrypted

Fig. 4: Sample video output (Carphone)

7. CONCLUSION

Two desirable characteristics of a transcoder for video streaming can be noted as adapting the media and protecting the security of the media. In this work, we showed that knowledge of the bitstream along with content structure in the form of metadata enables the encryption processes to be customized and easily deployable in the adaptation engine. The encryption engine is independent of bit-rate types since bit rate is engineered by the adaptation engine itself. Any sophisticated encryption algorithm suitable for H.264 video encryption can be coded and planted in the encryption engine. In this regard, we would like to recommend Motion Vector Encryption Algorithm (MVEA) [20] where concealing and distancing operations on motion vectors produce a better result in terms of encryption. On the other side, the decryption technique needs not to be implemented in all the end nodes unless it is obligatory. Since decompression is less complex than compression so the overhead added to decrypt the encoded macroblocks will not exceed the nominal threshold value for presentation as they will reside as a closely coupled tasks. Our research is now focused on providing a reasonable solution that will disallow possibly untrusted intermediary adaptation engines in the delivery path to adapt content in the encrypted domain.

8. REFERENCES

[1] ISO/IEC 21000-7:2004, Information Technology – Multimedia Framework – Part 7: Digital Item Adaptation.

[2] L. Sahafi, T.S. Randhawa, and R.H.S. Hardy, “Context-based complexity reduction of H.264 in video over wireless applications,” *IEEE Workshop on Multimedia Signal Processing*, pp. 23-26, 2004.

[3] S. Wenger, “H.264/AVC over IP,” *IEEE Trans. on Circuits and Systems for Video Tech.*, Vol.13, Issue 7, pp. 645-656, 2003.

[4] C. Gomila and P. Yin, “New features and applications of the H.264 video coding standard,” in *Proc. of Intl. Conf. on Info. Tech.: Research and Education*, pp. 6-10, 2003.

[5] H. Cheng and Li Xiaobo, "Partial encryption of compressed images and videos", *IEEE Trans. on Signal Processing*, Vol.48, Issue 8, pp. 2439-2451, 2000.

[6] T. Lookabaugh, “Selective encryption, information theory and compression,” *Conf. Record of the 38th Asilomar Conf. on Signals, Systems and Computers*, Vol.1, pp. 373-376, 2004.

[7] Y. Bodo, N. Laurent, and J.-L. Dugelay, “A scrambling method based on disturbance of motion vector,” in *Proc. of 10th ACM Intl. Conf. on Multimedia*, pp. 89–90, 2002.

[8] S. Lian, J. Sun, and Z. Wang, “Perceptual cryptography on SPIHT compressed images or videos,” in *Proc. of IEEE Intl. Conf. Multimedia & Expo, ICME*, Vol. 3, pp. 2195-2198, 2004.

[9] S. Lian, J. Sun, and Z. Wang, “Perceptual cryptography on JPEG2000 compressed images or videos,” in *Proc. of Intl. Conf. Computer and Info. Tech.*, pp. 78–83, 2004.

[10] S. Lian, X. Wang, J. Sun, and Z. Wang, “Perceptual cryptography on wavelet-transform encoded videos,” in *Proc. of IEEE Intl. Symposium on Intelligent Multimedia, Video and Speech Processing*, pp. 57–60, 2004.

[11] J. Zhang, A. Perkis, and N. Georganas, “H.264/AVC and Transcoding for Multimedia Adaptation,” in *Proc. of the 6th COST 276 WORKSHOP*, Greece, 2004.

[12] P. Reiher, K. Eustice, and S. Kai-Min, “Adapting encrypted data streams in open architectures,” *Third Annual Intl. Workshop on Active Middleware Services*, pp. 45-50, 2001.

[13] K. Gunhee, S. Dongkyoo, and S. Dongil, “An efficient methodology for multimedia digital rights management on mobile handset,” *IEEE Transactions on Consumer Electronics*, Vol.50, Issue 4, pp.1130-1134, 2004.

[14] S.J. Wee and J.G. Apostolopoulos, “Secure scalable streaming enabling transcoding without decryption,” in *Proc. of Intl. Conf. on Image Processing*, Vol.1, pp. 437-440, 2001.

[15] J.G. Apostolopoulos, “Secure media streaming & secure adaptation for non-scalable video,” *Intl. Conf. on Image Processing*, Vol. 3, pp. 1763–1766, 2004.

[16] W3C Consortium. “XSL Transformations (XSLT) Version 1.0”, W3C Recommendation (1999). [Online]. Available: http://www.w3.org/XML/

[17] W3C Consortium. “XSL Transformations (XSLT) Version 1.0”, W3C Recommendation (1999). [Online]. Available: http://www.w3.org/TR/xslt

[18] D. Mukherjee, H. Wang; A. Said, and S. Liu, “Format independent encryption of generalized scalable bit-streams enabling arbitrary secure adaptations,” in *Proc. of IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing*, Vol. 2, pp. ii/1033 - ii/1036, 2005.

[19] http://ftp3.itu.ch/av-arch/jvt-site/reference_software/

[20] L. Zheng and Li Xue, “Motion vector encryption in multimedia streaming,” in *Proc. of 10th Intl. Multimedia Modelling Conf.*, pp. 64-71, 2004.