

## A FORMAT-COMPLIANT CONFIGURABLE ENCRYPTION FRAMEWORK FOR ACCESS CONTROL OF MULTIMEDIA

Jiangtao Wen, Mike Severa, Wenjun Zeng, Max Luttrell, and Weiyin Jin\*

{gwen, severa, wzeng, luttrell}@packetvideo.com

PacketVideo Corporation

San Diego CA 92121 USA

\* EE Dept., Univ. of Southern California

**Abstract** – We present a framework for access control of standard-compliant video bitstreams for entertainment purposes. The approach leverages well-known encryption algorithms and maintains standard-compliance of the encrypted bitstream. The standard compliance feature guarantees the inheritance of the error resiliency properties of the video compression standards, and works with many existing network bandwidth adaptation and error control techniques that have been developed for standard-compliant compressed video, thus making it especially suitable for wireless multimedia applications. Standard compliance also allows subsequent signal processing techniques to be applied to the encrypted bitstream. The approach can be applied to any of the common video coding standards. It is also capable of providing layered compromises between security, complexity, delay, bit overhead, etc.

### 1. INTRODUCTION

One of the major goals of content access control for entertainment purposes (as opposed to for top secret communications) is to disallow unauthorized users to view the video with satisfactory quality, similar to a set top cable TV scrambler. One common method for access control is through encryption.

If one takes a look at today's food chain from the content creator, to the content copyright owner, to the distribution channel, to the final content consumer, digital compression is often performed at various stages, possibly with different compression techniques and digital formats. If one takes into account factors such as the platform, bandwidth, application, value of the content, etc., the task of designing the optimal encryption method for each combination quickly becomes prohibitively expensive. Therefore, it is very desirable to design a flexible encryption framework that could be configured for applications to various existing and forthcoming compression standards, media types, and security requirements. It should be noted that similar content in different application scenarios might have quite different requirements. This is particularly true for wireless multimedia distribution, due to several unique characteristics of the wireless networks. From the content access control point of view, there are a few requirements specific to wireless applications:

**Low processing overhead:** Due to the high data rate of video, encryption may add a large amount of processing overhead, making real-time secure video delivery

difficult to achieve. For wireless multimedia streaming to low-power devices such as smart phones and PDAs, low processing overhead is a particularly desirable feature.

**Network friendliness:** Given the time-varying nature of some wireless channels, to guarantee quality of service for wireless applications, on-line dynamic bandwidth allocation is extremely critical, and needs to be performed under very stringent delay and cell loss constraints. Therefore it is desirable to make the encryption transparent to network adaptation processes such as transcoding or rate shaping.

**Error resiliency:** Most video compression standards have been developed to provide certain level of resiliency to bit error or packet loss. Many error control techniques have been developed to address the effects of channel errors. It is desirable that most of these existing tools can be readily applied to the transmission of an encrypted video bitstream.

Aside from the above requirements, that may be more specific to wireless multimedia distribution, there are other desirable features that most multimedia applications have in common. These include multiple levels of security, proof of security, and applicability of various signal processing such as watermarking, random access, scene change detection, content-based searching, etc. to the encrypted bitstream in the various links of an end-to-end chain [2][3].

Encryption of content in the compressed domain can be achieved in various ways, the simplest of which is to encrypt the entire compressed media bitstream with a cipher. As discussed above, various links in the end-to-end chain may perform various signal processing. For these modules to operate properly for such encrypted content, the content usually has to be decrypted, processed, and then re-encrypted. The decryption/re-encryption introduces significant processing overhead, and the module has to be encryption/decryption capable, which is undesirable given the wide spectrum of encryption algorithms in use today, many of which are proprietary. From a content provider's point of view, inline decryption/re-encryption means that clear content needs to surface before it reaches the end-user, which poses significant security threats. Finally, direct encryption may create marker and header emulation. Markers and headers are special bit patterns in a compressed bitstream used for synchronization purposes and are usually designed so that they cannot be emulated by valid concatenation of other bits in a compressed bitstream. Marker and header emulation can introduce problems when the encrypted content is transmitted over certain network protocols designed for unencrypted compressed bitstreams. As an example, the Motion Marker in the data partition mode of MPEG-4 video syntax [4] is 17 bits and is not byte-aligned, therefore, on average it can be emulated in every  $2^{17}=131072$  cipher text bits, which is not a large number for even low bitrate video.

In this paper, we present a framework under which encryption of compressed content can be achieved in the compressed domain securely and with low complexity while still maintaining compliance to the compression format. If used properly, this framework will enable many of the signal processing techniques that

people would apply directly to compressed bitstreams to the encrypted bitstream. We will also present some tools [3] we have developed that enable this framework.

## **2. A FRAMEWORK FOR CONFIGURABLE FORMAT-COMPLIANT ENCRYPTION**

The basic idea of the approach is to encrypt only certain fields of the compressed bitstream in a manner that allows the encrypted bitstream to maintain format compliance, but to have unacceptable quality for the target application without decryption of these fields. The ability to maintain format compliance provides *ONE* satisfactory solution to all of the requirements discussed in Section 1. By reducing the amount of data to be encrypted to only “important” information carrying fields, it provides light-weight encryption. Syntax compliance inherits network friendliness, error resilience as well as the possibility of performing various types of signal processing on the encrypted bitstream directly. In addition, the encrypted bitstream will work nicely with protocols designed for the transport of standards-based compressed bitstreams. There will be no marker emulation problem. An old system with no DRM solution included can deal with secured content gracefully.

Selective encryption of compressed bitstreams by itself is not a new idea. Techniques proposed in some of the previous work [2][5][6] could result in an encrypted bitstream that still conforms to the standard. However, the importance and value of maintaining standard compliance has not been generally recognized except for in [1][7][8], where syntactical logic structure is preserved in a way that is outside the scope of syntax, and in [2] where the features such as processing overhead, data selectivity, error resiliency, different levels of security, transcodability and applicability of signal processing without decryption were discussed to some extent in a joint encryption and compression framework. The main contribution of this paper is the introduction of a framework that could achieve any level of syntax compliance in any format/standard through a unique compliance-preserving encryption method of variable length coded fields in compressed bitstreams.

In the proposed framework, a compressed bitstream can be divided into information-carrying and not information-carrying portions. Fields such as markers usually have a fixed pattern, and their locations can often be determined based on other information carrying fields. Therefore these fields do not really need to be encrypted. The information carrying fields are what need to be encrypted. These fields are selected fixed length code (FLC) or variable length code (VLC) codewords. One can also design a layered approach in which more information carrying fields are encrypted to provide more security at a cost of more computation.

To achieve format compliance, we extract bits from the fields that we’ve chosen to encrypt, concatenate them in an appropriate way, encrypt the concatenation using a common encryption algorithm such as DES, and then put the encrypted bits back into their original positions. For FLC coded fields, this operation usually results in a compliant bitstream. When it doesn’t, it is because the FLC codespace is not full. In

this case, the FLCs can be assigned indices and treated like VLCs as outlined below. For VLC coded fields, encrypting concatenated codewords may not result in another valid codeword concatenation. To maintain compliance, we assign a fixed length index to each codeword in the VLC code table, and instead of encrypting the code word concatenation, we encrypt the index concatenation, and then map the encrypted index concatenation back to codewords. To illustrate the process, suppose we have a 4 codeword table 0, 10, 110, 111, and a two-codeword concatenation of 010. If we assign two-bit indices 00, 01, 10, 11 to the codewords, then the original codeword concatenation will correspond to an index concatenation of 0001. Any pattern resulted from the encryption of 0001 can be mapped back to a compliant codeword concatenation. Because the distribution of the encrypted index will in general be more random, this method will in general introduce an overhead. The actual percentage of the overhead depends on the type of information that the VLC was used to encode, as well as on the content. Additionally, it is recognized that assigning fixed length indices restricts the mapping to a power-of-2 subset of the possible codewords. Code-tables with non-power-of-2 number of codewords can be divided into subsets of possibly different powers-of-2 numbers of codewords, and indexed separately. One should note that although the framework is very flexible, care needs to be taken when dependencies exist between various fields, such fields can not be encrypted independently while still maintaining format compliance.

### 3. SIMULATIONS

In our simulations, we focus on MPEG-4 video error resilient mode with data partitioning [4]. In this mode, the VLC-coded macro block (MB) coding type information and motion vector (MV) information is separated from the texture information for each video packet by the aforementioned motion marker. The texture information is also VLC-coded, with a 1-bit "sign" field present in all codewords. Video packets are delimited by a 17-bit resynchronization marker. We chose the FLC coded DCT sign, DQUANT, and INTRA DC information as the candidates for FLC encryption, and motion vectors (MVDs) for VLC encryption. This is appropriate because all these fields are critical to the correct interpretation of compressed video data. Fields such as DCT signs are also very random and are therefore secure from brute force guessing. For encryption of MVDs, each motion vector codeword is first assigned a fixed length index. The indices for each codeword are concatenated to form a sequence of indices  $S$ , which is encrypted with a cipher to result in encrypted motion vector index sequence  $S'$ .  $S'$  is mapped into MV codewords, which then replace the original codewords.

We tested encrypting only the FLC fields identified above (therefore with no overhead in bitrate), only the VLC coded MV fields (with a bit overhead dependent on the content and bitrate), and encrypting both the FLC and VLC coded fields (with a bit overhead equal to the second scenario). The encryption method we used was DES. FLC and VLC were always encrypted separately, because they were partitioned in the original syntax and may be subject to different processing/dropping during transmission. Each video packet was also encrypted

separately to alleviate packet loss effect. The tradeoffs of this approach are complexity, security, and bitrate overhead. In our simulations, we found that for sequences such as movie trailers, the decoded video usually looked scrambled enough to prohibit use for entertainment purposes after encrypting only the FLC fields. However, because motion information was not encrypted, certain content related information could sometimes be identified (e.g. when a scene change happened, when there was global motion, etc.). In contrast, when only VLC coded MVs were encrypted, the resulted content looked scrambled enough for most of the time, however, the unencrypted INTRA-MBs might result in certain regions of a frame being momentarily clear. However, the clear regions disappeared quickly, as the true motion reference relationship was removed during MV encryption. Encrypting both FLC and VLC fields seemed to be immune to the problem of encrypting either one alone, and we believe it could offer a higher level of security at the cost of more computation. Figure 1 shows screen shots of “Soap City”, encoded at 160x112 resolution, 5fps/40kbps, with and without encryption. The overhead for encrypting the VLCs in this case is just under 9%.

The overhead resulted from applying the standard compliance encryption method to MVD information is included in Table 1, where “MaxMV N” indicates that only MVDs within the range of  $[-N, N]$  (measured in half-pel) are encrypted. “Soap City” is a sequence with medium to high motion and many scene changes, “Hanging up” is a movie trailer with fewer scene changes and high motion, and “Head” is a talking head sequence. The overhead is in general lower for bitstreams with higher bitrate, lower framerate, more scene changes, and more INTRA frames, where motion information does not take a large portion of the overall bitrate. For low bitrate, low motion (e.g. talking head) or high motion but with high frame rate sequences, because motion information does take a significant portion of the bitrate, the overhead will be high. It should be recognized that the introduced overhead should be a primary concern when designing a selective encryption algorithm using the proposed framework. The decision should be made based on the level of security that needs to be achieved, the type of the content, and bandwidth.

#### **4. DISCUSSION AND CONCLUSION**

In summary, we described a generic framework of encrypting compressed content while preserving format compliance. We found that by selectively encrypting various critical fields of the compressed bitstream, we could trade off complexity, delay, security, bit rate overhead, and functionality. For example, for wireless multimedia streaming to smart phones and PDAs where bandwidth and computation are prime considerations, one can simply encrypt FLC fields, or encrypt a smaller portion of VLC fields and thereby reduce or avoid overhead.

Note that encrypted multimedia content is subject to error concealment based attacks, which are based on trying to conceal the unbreakable encrypted data based on other available data. For entertainment applications, such attack may still render the reconstructed content valueless. We are currently investigating such attacks.

## References

- [1] J. Meyer and F. Gadegast, "Security mechanisms for multimedia-data with the example MPEG-1-video," *Proj. description of SEC MPEG*, Tech. Univ. of Berlin, Germany, 5/95.
- [2] W. Zeng and S. Lei, "Efficient frequency domain video scrambling for content access control," *Proc. ACM Multimedia'99*, pp. 285-294, Orlando, Nov. 1999.
- [3] J. Wen, M. Luttrell and M. Severa, "Access control of standard video bitstreams," *Proc. International Conf. on Media Futures*, Florence, Italy, May 2001.
- [4] *ISO/IEC/SC29/WG11*, "Information technology – Coding of audio-visual objects – Part 2: Visual ISO/IEC 14496-2", International Standards Organization, 11/98.
- [5] L. Tang, "Methods for encrypting and decrypting MPEG video data efficiently," *Proceedings ACM Multimedia '96*, pp. 219-230, Boston, MA, 11/96.
- [6] C. Shi, S. Wang and B. Bhargava, "MPEG video encryption in real-time using secret key cryptography," <http://purdue.edu/homes/bb/security99.ps>
- [7] L. Qiao and K. Nahrstedt, "Comparison of MPEG encryption algorithms," *Inter. Journal on Computer & Graphics*, 22(3), 1998.
- [8] A. S. Tosun and W. Feng, "Light-weight security mechanism for wireless video transmission," *Proc. IEEE Inter. Conf. on Information Technology: Coding and Computing*, pp. 157-161, April 2001.

sequence	resolution	fps	kbps	overhead (MaxMV 8)	overhead (MaxMV 16)	overhead (MaxMV 32)
soap city	160x112	10	263	< 1%	1%	2%
	160x112	10	71	4%	8%	11%
	160x112	5	136	< 1%	1%	2%
	160x112	5	18	6%	13%	19%
hanging up	320x224	10	158	7%	10%	17%
	320x224	10	86	10%	21%	30%
	320x224	10	531	2%	3%	5%
head	176x144	10	29	8%	13%	18%
	176x144	10	17	10%	16%	22%
	176x144	5	26	8%	13%	17%
	176x144	5	5	17%	29%	36%

Table 1: Overhead for MVD encryption



Figure 1: Example sequence (from Left to Right): Original, Encrypt VLC only, Encrypt FLC, and Encrypt VLC&FLC.