ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 2, pp: (323-331), Month: April - June 2015, Available at: www.researchpublish.com

Data Hiding in Encrypted H.264/AVC Video Streams By Codeword Substitution

[1]Mr. Ratnakar Kumbhar, [2]Mr. Sagar Yamgar, [3]Ms. Rupali Ghule, [4]Prof. Murkute P.K

[1, 2, 3] Students of BE (IT), Al-Ameen College of Engineering, Koregaon Bhima, Pune, India

[4]Guide, Al-Ameen College of engineering, Koregaon Bhima, Pune, India

Abstract: **In today's life Cloud computing becomes an important technology trend. Digital video sometimes needs to be stored and processed in an encrypted format to maintain security and privacy. For the purpose of content notation and/or tampering detection, it is necessary to perform data hiding in these encrypted videos. In this way, data hiding in encrypted domain without decryption preserves the confidentiality of the content. In addition, it is more efficient without decryption followed by data hiding and re-encryption. In this paper, a novel scheme of data hiding directly in the encrypted version of H.264/AVC video stream is proposed, which includes the following three parts, i.e., H.264/AVC video encryption, data embedding, and data extraction. By analyzing the property of H.264/AVC codec, the codewords of intraprediction modes, the codewords of motion vector differences, and the codewords of residual coefficients are encrypted with stream ciphers. Then, a data hider may embed additional data in the encrypted domain by using codeword substitution technique, without knowing the original video content. In order to adapt to different application scenarios, data extraction can be done either in the encrypted domain or in the decrypted domain. Furthermore, video file size is strictly preserved even after encryption and data embedding. Experimental results have demonstrated the feasibility and efficiency of the proposed scheme.**

Keywords: **Data hiding, encrypted domain, H.264/AVC, codeword substitution.**

I. INTRODUCTION

Cloud computing has become an important technology trend, which can provide highly efficient computation and large-scale storage solution for video data. Given that cloud services may attract more attacks and are vulnerable to untrustworthy system administrators, it is desired that the video content is accessible in encrypted form. The capability of performing data hiding directly in encrypted H.264/AVC video streams would avoid the leakage of video content, which can help address the security and privacy concerns with cloud computing [1]. For example, a cloud server can embed the additional information (e.g., video notation, or authentication data) into an encrypted version of an H.264/AVC video by using data hiding technique. With the hidden information, the server can manage the video or verify its integrity without knowing the original content, and thus the security and privacy can be protected. In addition to cloud computing, this technology can also be applied to other prominent application scenarios. For example, when medical videos or surveillance videos have been encrypted for protecting the privacy of the people, a database manager may embed the personal information into the corresponding encrypted videos to provide the data management capabilities in the encrypted domain.

Till now, few successful data hiding schemes in the encrypted domain have been reported in the open literature. In [2], a watermarking scheme in the encrypted domain using Paillier cryptosystem is proposed based on the security requirements of buyer-seller watermarking protocols.

As said the above mentioned works have been focused on image. With the increasing demands of providing video data security and privacy protection, data hiding in encrypted H.264/AVC videos will undoubtedly become popular in the near future. To the best of our knowledge, there has been no report on the implementation of data hiding in encrypted H.264/AVC video streams. Only few joint data-hiding and encryption approaches that focus on video have been

proposed. For example, in [10], during H.264/AVC compression, the intra-prediction mode (IPM), motion vector difference (MVD) and DCT coefficients' signs are encrypted, while DCT coefficients' amplitudes are watermarked adaptively. In [11], a combined scheme of encryption and watermarking is presented, which can provide the access right as well as the authentication of video content simultaneously. The IPMs of 4 × 4 luminance block, the sign bits of texture, and the sign bits of MVDs are encrypted, while IPM is used for watermarking. However, the watermarked bitstream is not fully format-compliant as a result a standard decoder may crash since it cannot parse a watermarked bitstream. Concretely, the value "−2" of IPM does not exist in the actual standard. In summary, in the existing related technologies [10]–[11], encryption and data embedding are implemented almost simultaneously during H.264/AVC compression process. However, to meet the aforementioned application requirements, it's necessary to perform data hiding directly in the encrypted domain. In addition, the approaches in [10] and [11] do not operate on the compressed bitstream. That is, encryption and watermark embedding are accomplished in the encoding process, while decryption and watermark detection are completed in the decoding process The compression/decompression cycle is time-consuming and hampers real-time implementation. Besides, encryption and watermark embedding would lead to increasing the bit-rate of H.264/AVC bitstream.

Therefore, it becomes highly desirable to develop data hiding algorithms that work entirely on encoded bitstream in the encrypted domain However, there are some significant challenges for data hiding directly in compressed and encrypted bitstream.

1] The first challenge is to determine where and how the bitstream can be modified so that the encrypted bitstream with hidden data is still a compliant compressed bitstream.

2] The second challenge is to insure that decrypted videos containing hidden data can still appear to be of high visual fidelity.

3] The third challenge is to maintain the file size after encryption and data hiding, which requires that the impact on compression gain is minimal.

4] The fourth challenge is that the hidden data can be extracted either from the encrypted video stream or from the decrypted video stream, which is much more applicable in practical applications.

Based on the analysis given above, we propose a novel scheme to embed secret data directly in compressed and then encrypted H.264/AVC bitstream. Firstly, by analyzing the property of H.264/AVC codec, the codewords of IPMs, the codewords of MVDs, and the codewords of residual coefficients are encrypted with a stream cipher. The encryption algorithm is combined with the Exp-Golomb entropy coding and Context-adaptive variable-length coding (*CAVLC*) [12], which keeps the codeword length unchanged. Then, data hiding in the encrypted domain is performed based on a novel codeword substituting scheme. In contrast to the existing technologies [10]–[11] discussed above, the proposed scheme can achieve excellent performance in the following three different prospects.

• The data hiding is performed directly in encrypted H.264/AVC video bitstream.

• The scheme can ensure both the format compliance and the strict file size preservation.

• The scheme can be applied to two different application scenarios by extracting the hidden data either from the encrypted video stream or from the decrypted video stream.

II. PROPOSED SCHEME

A novel scheme of data hiding in the encrypted version of H.264/AVC videos is resented, which includes three parts, i.e., H.264/AVC video encryption, data embedding and data extraction. The content owner encrypts the original H.264/AVC video stream using standard stream ciphers with encryption keys to produce an encrypted video stream. Then, the data-hider (e.g., a cloud server) can embed the additional data into the encrypted video stream by using codeword substituting method, without knowing the original video content. At the receiver end, the hidden data extraction can be accomplished either in encrypted or in decrypted version. The diagram of the proposed framework is shown in Fig. 1, where the encryption and data embedding are depicted in part (a), and the data extraction and video decryption are shown in part (b).

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 2, pp: (323-331), Month: April - June 2015, Available at: www.researchpublish.com

A. Encryption of H.264/AVC Video Stream:

It is not practical to encrypt the whole compressed video bitstream like what the traditional ciphers do because of the following two constraints, i.e., format compliance and computational cost. Alternatively, only a fraction of video data is encrypted to improve the efficiency while still achieving adequate security. The key issue is then how to select the sensitive data to encrypt. According to the analysis given in [13], it is reasonable to encrypt both spatial information (IPMand residual data) and motion information (MVD) during H.264/AVC encoding.

In this paper, an H.264/AVC video encryption scheme with good performance including security, efficiency, and format compliance is proposed. By analyzing the property of H.264/AVC codec, three sensitive parts (i.e., IPMs, MVDs, and residual coefficients) are encrypted with stream ciphers. Compared with [13], the proposed encryption algorithm is performed not during H.264/AVC encoding but in the H.264/AVC compressed domain[14][15].

In this paper, we have improved and enhanced the previous proposed approach by encrypting more syntax elements. We encrypt the codewords of IPMs, the codewords of MVDs, and the codewords of residual coefficients. The encrypted bitstream is still H.264/AVC compliant and can be decoded by any standard-compliant H.264/AVC decoder, but the encrypted video data is treated completely different compared to plaintext video data[16].

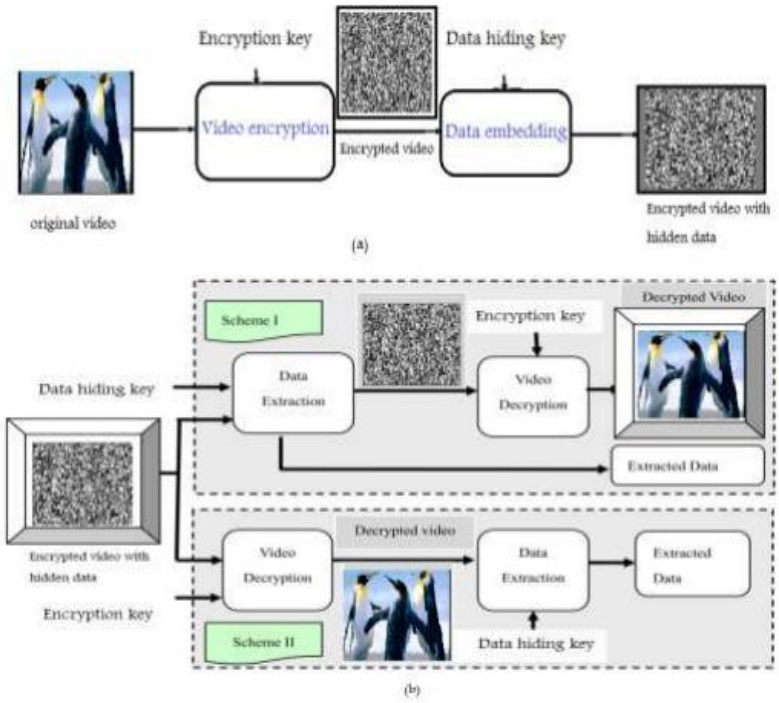

Fig.1 Diagram of proposed scheme. (a) Video encryption and data embedding at the sender end. (b) Data extraction and video display at the receiver end in two scenarios

1) Intra-Prediction Mode (IPM) Encryption: According to H.264/AVC standard, the following four types of intra coding are supported, which are denoted as Intra_4 × 4, Intra_16×16, Intra_chroma, and I_PCM [12]. Here, IPMs in the Intra_4×4 and Intra_16 × 16 blocks are chosen to encrypt[17].

2) Motion Vector Difference (MVD) Encryption: Further to protect both texture information and motion information, not only the IPMs but also the motion vectors should be encrypted. In H.264/AVC, motion vector prediction is further carried out on the motion vectors, which yields MVD. In H.264/AVC baseline profile, ExpGolomb entropy coding is used to encode MVD encryption may change the sign of MVD, but does not affect the length of the codeword and satisfies the format compliance.

3) Residual Data Encryption: In order to keep high security, another type of sensitive data, i.e., the residual data in both I-frames and P-frames should be encrypted. In this region, a novel method for encrypting the residual data based on the characteristics of codeword.In H.264/AVC baseline profile, CAVLC entropy coding is used to encode the quantized

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 2, pp: (323-331), Month: April - June 2015, Available at: www.researchpublish.com

coefficients of a residual block. Each CAVLC codeword can be expressed as the following format: {Coeff_token,Sign_of_Trailing Ones, Level, Total_zeros, Run_before}

B. Data Embedding:

In the encrypted bitstream of H.264/AVC, the proposed data embedding is accomplished by substituting eligible codewords. On the other hand, the codewords substitution should fulfill the following three limitations. First, the bitstream after codeword substituting must remain syntax compliance so that it can be decoded by standard decoder. Second, to keep the bit-rate unchanged, the substituted codeword should have the same size as the original codeword. The codewords of Levels which suffix Length is 2 or 3 would be divided into two opposite codespaces denoted as C0 and C1 as shown in Figure. 3.2. The codewords assigned in C0 and C1are associated with binary hidden information "0" and "1". Suppose the additional data that we want to embed is a binary sequence denoted sB={b(i)|i =1,2,⋯,L,b(i)∈ {0,1}}. Data hiding is performed directly in encrypted bit-stream through the following steps

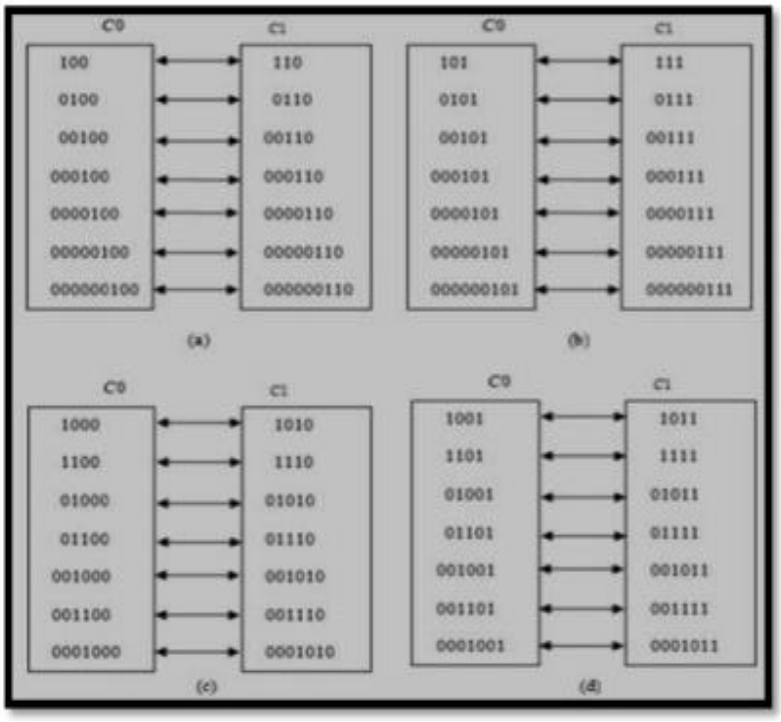

Fig.2 CAVLC codeword mapping

Step1: In order to enhance the security, the additional data is encrypted with the chaotic pseudo-random sequence P={p(i)|i =1,2,⋯,L,p(i)∈{0,1}}[22] to generate the to-be-embedded sequence W={w(i)|i =1,2,⋯,L,w(i)∈{0,1}}. The sequence Pis generated by using logistic map with an initial value, i.e., the data hiding key. It is very difficult for anyone who does not retain the data hiding key to recover the additional data.

```
Procedure
If(databit==0){
        if(the codword belongs to C0)
         The codeword is unmodified;
         Else if (the codeword belongs to C1)
         The codeword is replaced with the corresponding codeword in C0
           }
If(data bit==1){
        if(the codword belongs to C1)
         The codeword is unmodified;
         Else if (the codeword belongs to C0)
         The codeword is replaced with the corresponding codeword in C1
          }
```

Fig.3 Procedure of codeword mapping

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 2, pp: (323-331), Month: April - June 2015, Available at: www.researchpublish.com

Step2: The codewords of Levels are obtained by parsing the encrypted H.264/AVC bitstream

Step3: If current codeword belongs to codespaces C0 or C1, the to-be-embedded data bit can be embedded by codeword substituting. Otherwise, the codeword is left unchanged. The detailed procedure of codeword substituting for data hiding is shown in Figure. 3.3. For example, when Level is positive 1 and its suffix Length is 3, then its corresponding codeword is "1000" which belongs to C0 as shown in Figure.3.2. If the data bit "1" needs to be embedded, the codeword "1000" should be replaced with "1010". Otherwise, if the data bit "0" needs to embedded, the codeword "1000" will keep unchanged.

Step4: Choose the next codeword and then go to Step3 for data hiding. If there are no more data bits to be embedded, the embedding process is stopped. Suppose the to-be-embedded data is "1001", the CAVLC codeword of Level parsing from H.264/AVC bitstream is Figure. 3.3. The procedure of codeword mapping

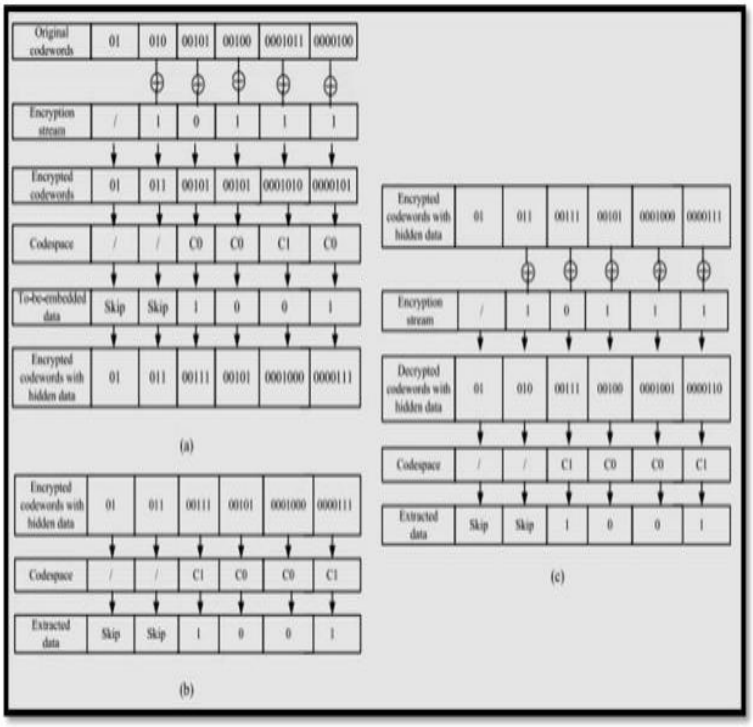

Fig.4 Example of data embedding & extraction. (a)Data embedding. (b)Data extraction in encrypted domain. (c)Data extraction in decrypted domain.

C. Data Extraction:

In this scheme, the hidden data can be extracted either in encrypted or decrypted domain. Data extraction process is fast and simple.

Scheme I: Encrypted Domain Extraction.

Scheme II: Decrypted Domain Extraction.

Scheme I: Encrypted Domain Extraction.

To protect privacy, a database manager (e.g., cloud server) may only get access to the data hiding key and have to manipulate data in encrypted domain. Data extraction in encrypted domain guarantees the feasibility in this case. In encrypted domain, as shown in Figure.1(b), encrypted video with hidden data is directly sent to the data extraction module. An example of data extraction in encrypted domain is shown in Figure.4(b).

Scheme II: Decrypted Domain Extraction.

In scheme I, both embedding and extraction of the data are performed in encrypted domain. However, in some cases, users want to decrypt the video first and extract the hidden data from the decrypted video. For example, an authorized user, which owned the encryption key, received the encrypted video with hidden data. The received video can be

decrypted using the encryption key. That is, the decrypted video still includes the hidden data, which can be used to trace the source of the data. In Figure 1(b), the received encrypted video with hidden data is first pass through the decryption module.

The whole process of decryption and data extraction is given as follows.

Step1: Generate encryption streams with the encryption keys as given in encryption process.

Step2: The codewords of IPMs, MVDs, Sign_of_TrailingOnes and Levels are identified by parsing the encrypted bit stream.

Step3: The decryption process is identical to the encryption process, since XOR operation is symmetric. The encrypted codewords can be decrypted by performing XOR operation with generated encryption streams, and then two XOR operations cancel each other out, which renders the original plain-text. Since the encryption streams depend on the encryption keys, the decryption is possible only for the authorized users. After generating the decrypted codewords with hidden data, the content owner can further extract the hidden information.

Step4: the last bit encryption may change the sign of Level. The encrypted codeword and the original codeword are still in the same code spaces.

Step5: Generate the same pseudo-random sequence that was used in embedding process according to the data hiding key. The extracted bit sequence should be decrypted to get the original additional information. An example of data extraction in decrypted domain is shown in Figure. 4(c)

III. OBJECTIVES

- To provide computerized cloud storage.
- Easy retrieval of crime and criminal records.
- Based on top cloud platform Salesforce.com cloud.
- Sharable data and provides actual statistics. To provide web based and easy analysis of data can be easily available using cloud technology.
- Search tool provided to recognize criminals and crimes.
- Police are only liable to access investigation tool.

IV. SYSTEM DESIGN

4.1 System Architecture:

IAC can be divided in to several components in order to describe the system clearly. Fig. 1 shows those components and how they are interconnected within the system.

A. User interface:

The user interface is one of the most important parts of any program because it determines how easily a user can interact with the program. The goal of user interface design is to make the user's interaction as simple and efficient as possible, in terms of accomplishing user goals. As a powerful program with a poorly designed user interface has very little value, the process of designing user interface was done very carefully. User interface, which is one of the components in ICIS, consists of all the web pages of the system that are displayed to users through a web browser.

B. Managed bean:

Technically, a Managed Bean is a POJO that's constructed and injected by the JSF framework. It's commonly referenced by JSF pages. Thus managed beans are used as model for UI components. They contain the data and underlying logic of the application. Managed beans are capable of querying the system for entities requested by the user interface. The system encapsulates the requested entity into an object and returns it to the bean. The bean exposes the object properties through a set of bean properties. This architecture simplifies development and shields the presentation layer from changes to the enterprise system implementation.

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 2, pp: (323-331), Month: April - June 2015, Available at: www.researchpublish.com

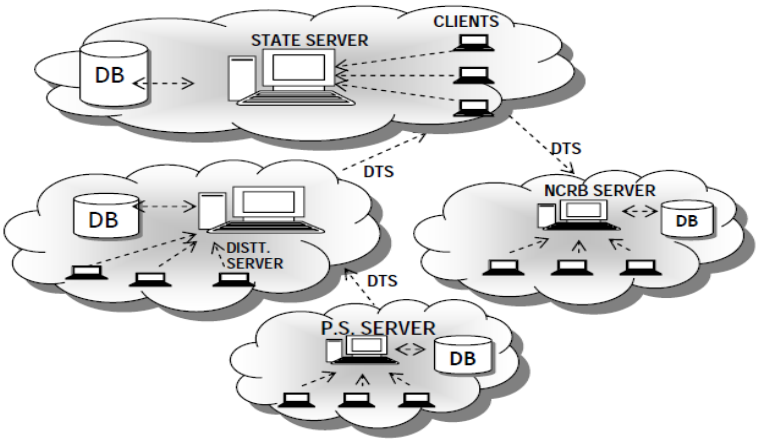

Fig. 1 Architecture of IAC

C. Multi agent system:

This component is used for analyzing existing information of criminals on attempting crimes and identifying a pattern of committing crimes by them. When user inputs relevant crime information for finding a possible criminal, the system starts identifying a pattern of committing that crime based on the evidences available. The major task that was to be done when designing this module is the proper identification of agents and ontology. As I mentioned earlier five types of agents were identified when designing the system namely, message space agent, gateway agent, prisoner agent, criminal agents and evidence agents. Fig shows depicts the prime interactions between each of these agents.

Message Space Agent: responsible for handling messages over the system. This works as the communicator between all agents of the system except Prison agent.

Gateway Agent: acts as a gateway connecting the web application and underlying multi-agent system.

Prison Agent: check whether the suspect is currently a prison inmate.

Evidence Agents: responsible for identifying crime patterns according to the evidences available in a crime scene and for identifying criminal according to those patterns.

Criminal Agents: responsible for calculating the probability of committing a crime. Uses Naïve

Bayesian classification:

X = User x committing a crime

Ci = Set of evidences

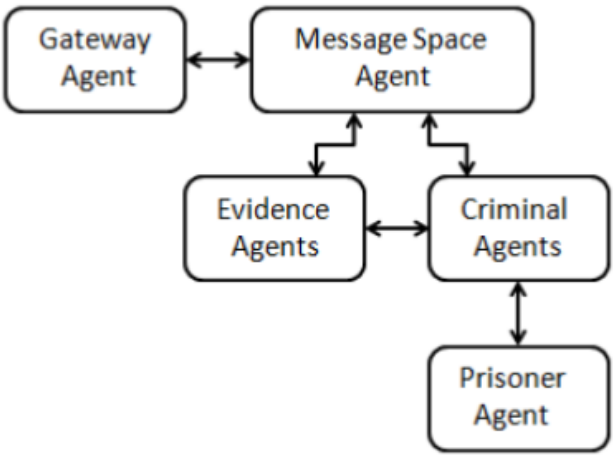

Fig. 2: Interaction between Agents

D. Database:

Database stores all the crime and criminal information as well as the patterns or the clusters of crime incidents. Moreover it stores the details of ICAT users as well.

V. SYSTEM IMPLEMENTATION

- **User Interface:**
- Java GUI
- **Hardware Required:**
  - Processor: At least Dual Core
  - Memory (RAM): Minimum512 MB RAM
  - Hard Disk: Minimum 40 GB
  - Operating System :Minimum Windows XP
- **Software Required:**
  - Development Tool: Eclipse IDE
  - Development Kit: JDK
  - Development platform:JAVA

VI. CONCLUSION

The existing system just addresses the calculation of smoothness and the process of image recovery and lacks in addressing the data encryption & data embedding process. Time consumption rate is also high when compared to the recent methods developed. The block encryption methods are not robust to noise.Data hiding in encrypted media is a new topic that has started to draw attention because of the privacy-preserving requirements from cloud data management. An algorithm is used to embed additional data in encrypted H.264/AVC bit stream, which consists of video encryption, data embedding and data extraction phases.The data-hider can embed additional data into the encrypted bit stream using codeword substituting, even though he does not know the original video content. Since data hiding is completed entirely in the encrypted domain, here we can preserve the confidentiality of the content completely.

REFERENCES

[1] W. J. Lu, A. Varna, and M. Wu, "Secure video processing: Problems and challenges," in Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, Prague, Czech Republic, May 2011, pp. 5856–5859.

[2] B. Zhao, W. D. Kou, and H. Li, "Effective watermarking scheme in the encrypted domain for buyer-seller watermarking protocol," Inf. Sci., vol. 180, no. 23, pp. 4672–4684, 2010.

[3] P. J. Zheng and J. W. Huang, "Walsh-Hadamard transform in the homomorphic encrypted domain and its application in image watermarking," in Proc. 14th Inf. Hiding Conf., Berkeley, CA, USA, 2012, pp. 1–15.

[4] W. Puech, M. Chaumont, and O. Strauss, "A reversible data hiding method for encrypted images," Proc. SPIE, vol. 6819, pp. 68191E-1–68191E-9, Jan. 2008.

[5] X. P. Zhang, "Reversible data hiding in encrypted image," IEEE Signal Process. Lett., vol. 18, no. 4, pp. 255–258, Apr. 2011.

[6] W. Hong, T. S. Chen, and H. Y. Wu, "An improved reversible data hiding in encrypted images using side match," IEEE Signal Process. Lett., vol. 19, no. 4, pp. 199–202, Apr. 2012.

[7] X. P. Zhang, "Separable reversible data hiding in encrypted image," IEEE Trans. Inf. Forensics Security, vol. 7, no. 2, pp. 826–832, Apr. 2012.

[8] K. D. Ma, W. M. Zhang, X. F. Zhao, N. Yu, and F. Li, "Reversible data hiding in encrypted images by reserving room before encryption," IEEE Trans. Inf. Forensics Security, vol. 8, no. 3, pp. 553–562, Mar. 2013.

[9] A. V. Subramanyam, S. Emmanuel, and M. S. Kankanhalli, "Robust watermarking of compressed and encrypted JPEG2000 images," IEEE Trans. Multimedia, vol. 14, no. 3, pp. 703–716, Jun. 2012.

[10] S. G. Lian, Z. X. Liu, and Z. Ren, "Commutative encryption and watermarking in video compression," IEEE Trans. Circuits Syst. Video Technol., vol. 17, no. 6, pp. 774–778, Jun. 2007.

[11] S. W. Park and S. U. Shin, "Combined scheme of encryption and watermarking in H.264/scalable video coding (SVC)," New Directions Intell. Interact. Multimedia, vol. 142, no. 1, pp. 351–361, 2008.

[12] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," IEEE Trans. Circuits Syst Video Technol., vol. 13, no. 7, pp. 560–576, Jul. 2003.

[13] S. G. Lian, Z. X. Liu, Z. Ren, and H. L. Wang, "Secure advanced video coding based on selective encryption algorithms," IEEE Trans. Consumer Electron., vol. 52, no. 2, pp. 621–629, May 2006.

[14] Z. Shahid, M. Chaumont, and W. Puech, "Fast protection of H.264/AVC by selective encryption of CAVLC and CABAC for I and P frames," IEEE Trans. Circuits Syst. Video Technol., vol. 21, no. 5, pp. 565–576, May 2011.

[15] M. N. Asghar and M. Ghanbari, "An efficient security system for CABAC bin-strings of H.264/SVC," IEEE Trans. Circuits Syst. Video Technol., vol. 23, no. 3, pp. 425–437, Mar. 2013.

[16] T. Stutz and A. Uhl, "A survey of H.264 AVC/SVC encryption," IEEE Trans. Circuits Syst. Video Technol., vol. 22, no. 3, pp. 325–339, Mar. 2012.

[17] Advanced Video Coding for Generic Audiovisual Services, ITU, Geneva, Switzerland, Mar. 2005.