A Light-Weight Encrypting For Real Time Video Transmission

Salah Aly
aly@cs.depaul.edu

Abstract

In the digital world nowadays, multimedia security is becoming more and more important with the continuous increase in the use of digital communications on the Internet. In addition, special and reliable security in storage and transmission of digital images and videos is needed in many digital applications, such as pay-TV, confidential video conferencing and medical imaging systems, etc. Unfortunately, the classical techniques for data security are not appropriate for the current multimedia usage. As a result, we need to develop new security protocols or adapt the available security protocols to be applicable for securing the multimedia applications. Generally speaking, the well-developed modern cryptography should be the perfect solution for this task. In this paper, we show that AES algorithm can be used for securing real time video transmission with little processing overhead over the Internet. We support our study with exhaustive comparison between AES and XOR encryption algorithms with normal transmission.

1 Introduction

The advent of networked multimedia systems will make continuous media streams, such as real time audio and video, increasingly pervasive in future computing and communications environments. It is thus important to secure networked continuous media from potential threats such as hackers, eavesdroppers, etc [1]. This work was originally done at [2], [3]. Consequently, we study the methods of encrypting video packets using different security techniques. Also, we study performance of encryption and decryption algorithms such as AES for real time video streams. We adapt AES and XOR algorithms to be used with JPEG, H261, CellB, and MPEG video encoders and decoders. We present our results in diagrams, which show the encryption performance between different video encoders. In short, the foremost goal of this research is to cover these points:

- Determining how encryption and decryption can be implemented for the real time video applications.
- Comparing our results with the previous results in video encryption.
- Computing performance and overhead of multimedia security.

2 Problem Description

The nature of playing video streams over a network in a real time requires that the transmitted frames are sent in a bounded delay. Also, video frames need to be displayed at a certain rate; therefore, sending and receiving encrypted packets must be sent in a certain amount of time in order to utilize the admissible delay.

As shown in Figure 1, There are three main components for securing multimedia transmissions:

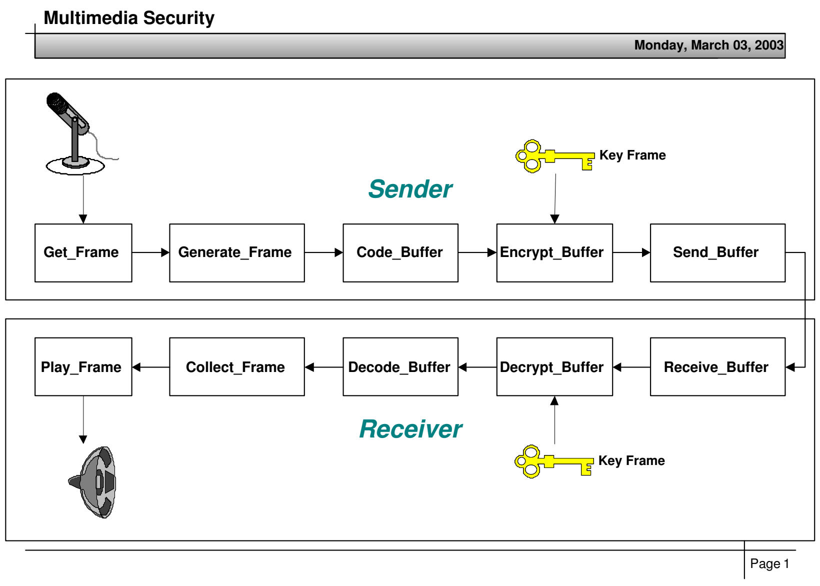

Figure 1: Framework for secure video transmission

1. Key management techniques: the most important issues of key management are key generation, agreement, transport, storage, and refreshment. The well-known key management protocols are D.H., SKEA, SSH, SSS, etc.

2. Encryption and decryption algorithms: the current encryption algorithms that are applicable for video encryption are DES, AES, RSA, RC4, RC6, IDEA, PGP, PEM, Kerberos, etc.

3. Encoder and Decoders mechanisms: discussed in detail in section 4.

For example: Video on-Demand (VOD) requires that the video stream needs to be played whenever the receiver asks for it. So, there are no buffer or playback concepts for the video stream (i.e. it runs in real time). There are many challenges for multimedia security such as:

- The natural size of multimedia data after compression is usually very large, even if using the best available compression techniques. So, the size of a two-hour MPEG-1 video is about 1 GB [4].

- Future applications of multimedia need to be run in real time on processes such as video on demand.

- Performance of processing multimedia streams should be acceptable (i.e. bounded by certain value of delay).

- Then the encryption techniques should take a short time and require small overhead in comparison to compression techniques.

3 Basic Concepts of Video Encryption

The encryption and decryption of a plain text or a video stream can be done in two ways:

- Secret key encryption: a single secret key can be used to encrypt and decrypt the video streams. Only the sender and the receiver have this key.
- Public key encryption: there are two keys, one for encryption and the other for decryption. The public key, which is known for all senders, is used for encryption. While the private key, which is owned only by the receivers, is used for decryption.

Public key cryptography is not applicable for secure real time video conferencing because it's operations require an amount of time, which is not suitable for video conferencing. However, the idea of public key encryption is applicable for signature and authentication of multimedia security. The four main issues of multimedia security are confidentiality, authentication, signature and watermarking, and copy protection.

As stated in [1], [2] the security systems that have been used for multimedia security include DES, RSA, Kerberos, Privacy Enhanced Mail (PEM), Pretty Good Privacy (PGP), IDEA, etc.

The main issue when encrypting real time video streams whether in video conferencing, personal video transmission, or even public sensitive video data is the performance. Since playing multimedia frames is a real time task, encryption and decryption processes can not take as much time as the compression process; otherwise, the system's performance will suffer and the overhead will be too costly. So, it is important that the encryption and decryption are bounded by a constant time [4]. Therefore, we use secret key encryption to secure real time video transmissions.
The main result of our implementation is that the measured performance using the AES encryption algorithm is 29.3 frames per second(fps) for JPEG, where the original JPEG player performs 23.5 up to 28 frames per second (fps). We also present performance of XOR and AES algorithms in comparison to the performance of video transmission without encryption as shown in the results.

4 Previous Work in Video Encryption

Video encryption techniques can be classified as Naive algorithm, selective algorithm, ZigZag algorithm, Video Encryption Algorithm (VEA), and pure permutation [2], [5]. we will review each one briefly in this section.

4.1 Naive Algorithm

The idea of naive encryption is to deal with the video streams as text data. The simplest way to encrypt video streams is to encrypt every byte. So, Naive algorithm encrypts every byte in the whole video stream. Native algorithms guarantee the most security level. However, it is not an applicable solution if the size of the data is big enough. Because of encryption operations, the delay increases and the overhead will not be satisfactory for the real time video encryptions.

4.2 Selective Algorithm

Tang in [5] suggested encrypting different levels of selective parts of video streams. As video nature is different in its components, he suggested four levels of selective algorithms.

These Four levels are encrypting all headers, encrypting all headers and I frames, encrypting all I frames and all I blocks in P and B frames, and finally encrypting all frames as in Naive algorithm to guarantee the highest security.

4.3 Zig-Zag Algorithm

The idea of ZIG-ZAG algorithm is basically encrypting the video streams before compressing them. Explicitly, when mapping the 8x8 block to a 1x64 vector each time in the same order. We can use a random permutation to map this transformation of the 8x8 block to the 1x64 vector. Therefore, the concept of the encryption key does not exist in the ZIG-ZAG permutation algorithms. Once the permutation list is known, the algorithm will not be secure any longer [3].

4.4 Video Encryption Algorithm

Klara and Quia in [6] suggested a new video encryption algorithm called VEA that depends on dividing the video streams into chunks. These chunks are separated into two different lists (odd and even lists). Afterward, applying an encryption algorithm like DES to the even list and the final ciphertext is a concatenation of output of encryption algorithm XOR with the odd list streams.

4.5 Pure Permutation

The idea of Pure Permutation is simply to apply a permutation technique for the I frames. Both the sender and the receiver have only the correct permutation to encrypt and decrypt the video streams respectively. Later work by Klara [2] proved that it is not secure to use pure permutations.

4.6 Suggested Technique

In our proposed technique, we attempt to select specific frames to encrypt. The encrypted video streams are combinations of I, P, and B frames. In addition, we choose different sizes to encrypt on the same video streams to achieve the outcome of the needed level of security. For example, in H.261 encoder, we try encrypting 16 bytes of 1135 length of the whole video stream as one packet before transmitting it over the network. We found that the video stream is ambiguous and cannot be seen. On the other hand, the same 16 bytes does not guarantee the same level of security when using JPEG encoder. Therefore, we increased consecutively the number of encrypted bytes to 32, 64, 96,128, or 144. We reach the darkness of the displayed video stream when the number of encrypted bytes is 144 using JPEG. This means encrypting only 16 bytes of h.261 is in the same level of security for encrypting 144 the in case of JPEG as we elaborate that in the next section. Using this technique, the video streams can be sent securely using AES encryption algorithm in a real time.

5 Performance Study

This section presents the main results of our experiment and shows the relationship between JPEG, H.261, MPEG, and CellB encoders when the video streams are encrypted using AES and XOR algorithms. Although there are basic differences between these encoders in the way they work, the overhead of encryption and decryption is acceptable when we send encrypted streams over a network. We aim in our experiment to do the following :

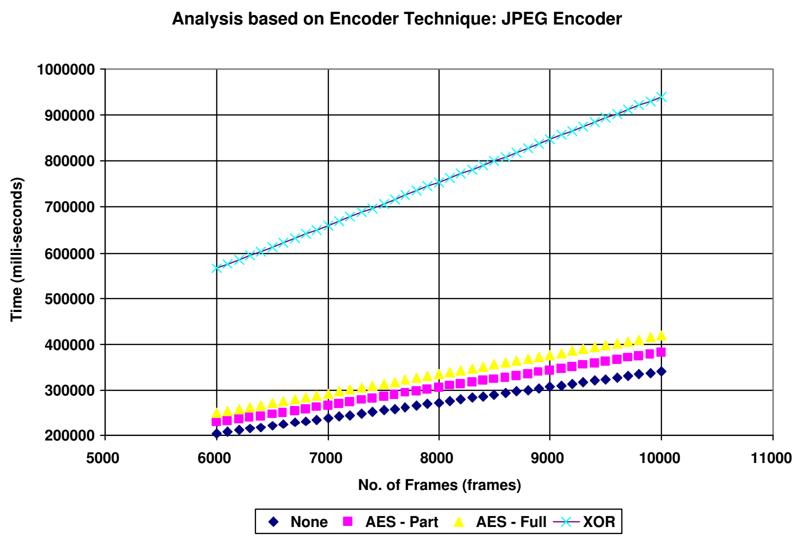

Figure 2: JPEG Encoder

- Use AES algorithm as a widely standard encryption algorithm to secure video transmission.
- Compare our results using AES and XOR algorithms, and previous results by , [2], [5], [6] for encrypting video transmission.
- Observe the difference in performance between partial encryption and full (Naive) encryption.
- Compute the overhead that is added to video compression upon our experiment as a result of encryption processes.

5.1 Implementation

We used the SUN Solaris machines in the multimedia and telecommunication laboratory. In addition, we used Sun XIL1.3 video library to capture the video frames and display them. For the video transmission, we used UDP transmission protocol to send and receive the video packets through the network channel. C programming language has been used since it works perfectly with the SUN Solaris systems and has many advantages with the network programming. In addition, we modified the standard AES and XOR codes to encrypt different lengths of video streams. We developed our final code in some functions that capture the video streams in JPEG, CELLB, H.,261 encoders, and handle the encryption operations.

We selected a fixed key length for AES and XOR encryption algorithms. Klara [2] and Bhargava [9] used DES and IDEA with a key length of 64 bits to encrypt a video stream block of 64 bits. Fortunately, AES helps us to encrypt directly 128 bits of a video stream, which makes the computation fast in comparison to their work. We also tested many different partial encryption sizes of plain video streams. In testing CellB, H.261, and JPEG, the various lengths of encryption were 16, 32, 64, 96, 128, and 144 bytes. And finally, we examined the effect of encrypting the whole length of video packets.

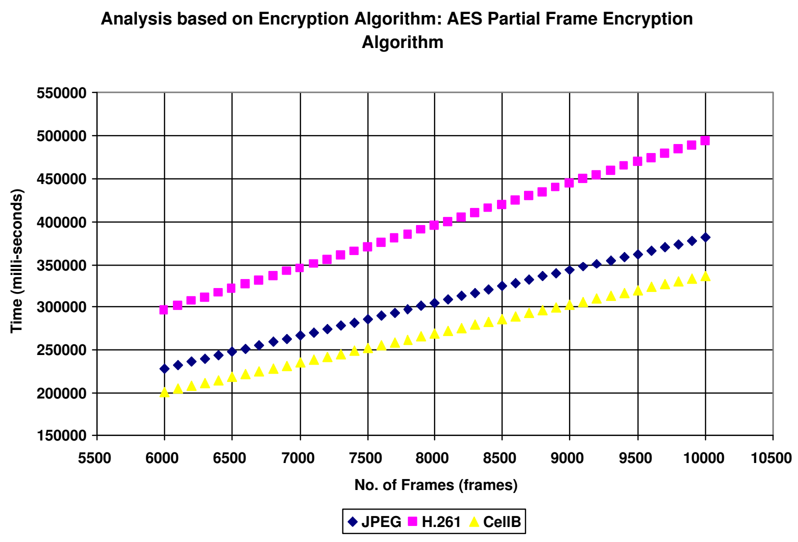

Figure 3: AES Partial Encryption

5.2 Performance of Video Encoders

As in Figure 2, we show a comparison between AES and XOR encryption algorithms when sending video stream over a network. We used partial and full AES algorithm. We notice that the difference between the non-encrypted packets and encrypted packets using AES is not huge. Moreover, the overhead of partial and full AES is very close. The difference between the encrypted packets using XOR and AES is sufficient, which makes using AES a better solution.

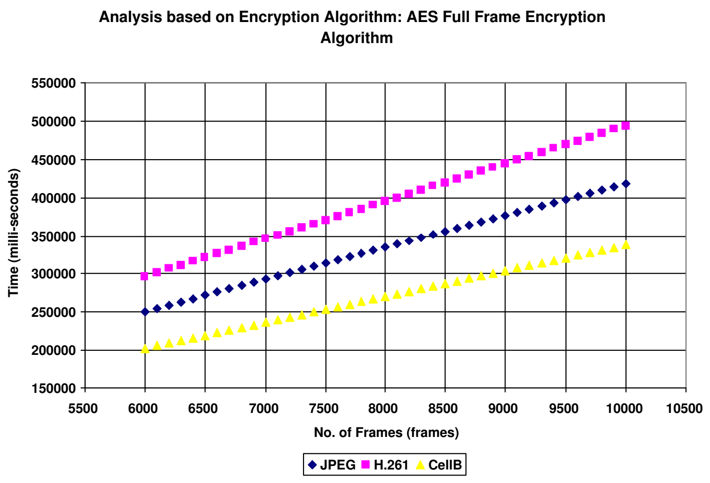

Figure 4: AES Full Encryption

5.3 Performance of Partial and Full AES Encryption

Figures 3 and 4 show encrypted video streams using AES algorithm. We notice that there is no big difference between encrypting only 16-bytes or full-packet since the time for encryption is very short in comparison to the time spent for frames encoding. But, we also noticed that encrypting 16 up to 128 bytes is not enough for security if a JPEG encoder is used. However, encrypting only 16 bytes of CellB and H.261 guarantees enough security as shown in the experiment.

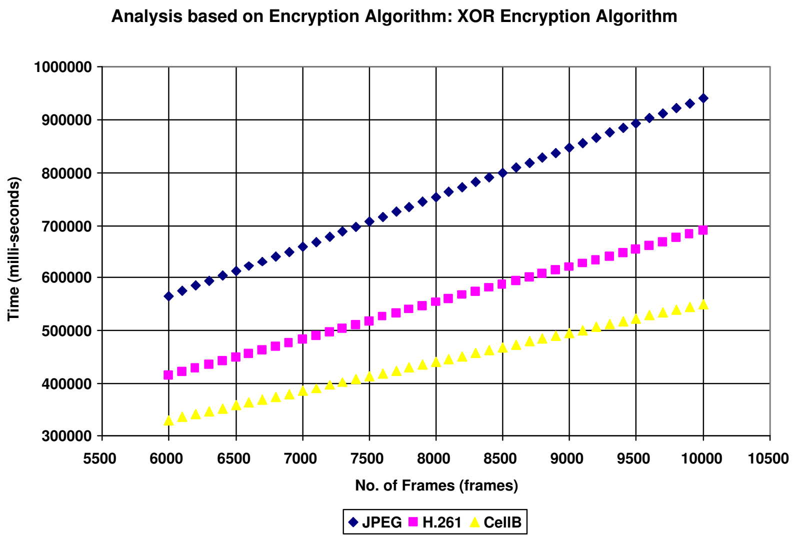

Figure 5: XOR Encryption

5.4 Summary of Results

As Figures 3, 4, and 5 show, The overhead time of encrypted packets using AES is less than the overhead time of the encrypted packet using XOR. From these Figures, the relative time spent for the encryption operation does not negatively affect video stream transmission. As a result, the delay resulted from both encryption and compression is approximately 0.031 second which is acceptable for video transmissions. In conclusion, encrypting video streams using AES is an applicable solution to secure real time video transmission.

6 Conclusion

Our study in this paper shows that we can send and receive encrypted real time video streams using AES encryption Algorithm. The performance of AES encryption frames is sufficient to display the received frames on time. As the results show, the encryption delay overhead using AES is less than the overhead using XOR algorithm. In addition, AES can achieve satisfactory encryption results with little overhead. Therefore, we conclude that using AES is a feasible solution to secure real time video transmissions. We recommend to use partial or full AES encryption to secure real time video applications including Video On-Demand (VOD), pay-TV, and video conferencing. This usage depends also on how much security is needed. Our future work of video conferencing includes using AES algorithm with the new compression schemes such as H.264 and MPEG7.

References

[1] Y. Li, Z. Chen, S. Tan, R. Campbell, Security-enhanced MPEG Player, Technical report, UIUC, 1996 (presented by Hao Chu)

[2] L. Qiao, K. Nahrstedt, Comparison of MPEG Encryption Algorithms, International Journal on Computers and Graphics, Special Issue on Data Security in Image Communication and Network, vol. 22, num. 3, Permagon Publisher, 1998. ps

[3] L. Qiao, Multimedia Security and Copyright Protection, Ph.D. Thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, October, 1998.

[4] B. Bhargava, C. Shi, and Y. Wang, MEPG Video Encryption Algorithms, Multimedia Tools and Applications 2003.

[5] L. Tang, Methods for Encrypting and Decrypting MPEG Vedio Data Efficiently, proceeding of fouth ACM international multimedia conference 96, page 219-230, Boston, MA, Nov. 1996

[6] L. Qiao, K. Nahrstedt, A New Algorithm for MPEG Video Encryption, in Proc. of International Conference on Imaging Science, Systems, and Technology (CISST'97), pp. 21-29, Las Vegas, NV, June, 1997.

[7] V. Talwar, S. Kumar Nath, K. Nahrstedt, RSVP-SQoS : A Secure RSVP Protocol, in Proc. of IEEE International Conference on Multimedia and Expo 2001 (ICME2001), Tokyo, Japan, August, 2001.

[8] J. G. Apostolopoulos, W. Tan, S. J. Wee, Vedio Streaming: Concepts, Algorithms, and Systems,HP Laboratories Palo Alto, Hewlett-Packard, sept, 2002

[9] B. Bhargava, C. Shi, and Y. Wang, MPEG Video Encryption in Real-time Using Secret Key Cryptography, 1999

[10] L.A. Rowe, K. Patel, B.C. Smith, K. Liu, MPEG Video in Software: Representation, Transmission and Playback, Proc. of IS/T/SPIE 1994 Int'l Symp. on Elec. Imaging: Science and Technology, San Jose, CA, February 1994.

[11] C. Shi and B. Bhargava. An Fast MPEG Video Conferencing Encryption Algorithm, in Proceedings of the 6th ACM International Multimedia Conference, Sept. 1998, Bristol, UK, pp. 8188.

[12] A. M.Eskicioglu, E. J. Delp, , An Integrated Approach to Encrypting Scalable Video, Proceedings of the 2002 IEEE International Conference on Multimedia and Expo, pp. 573-576, Lausanne, Switzerland, August 26-29, 2002.

[13] I. Agi and L. Gong, An Empirical Study of MPEG Video Transmission. In proceedings of the Internet Society Symposium on Network and Distributed System Security, Pages 137-144, San Diego, CA, Feb. 1996.

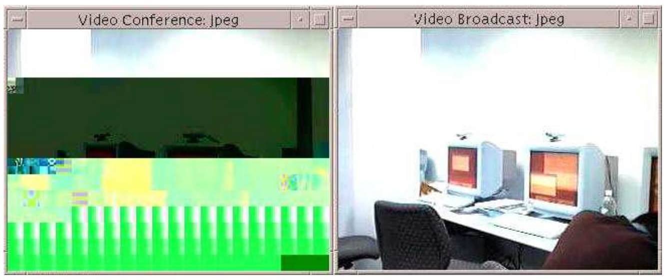

Figure 6: AES: encrypt only 16-bytes using JPEG encoder

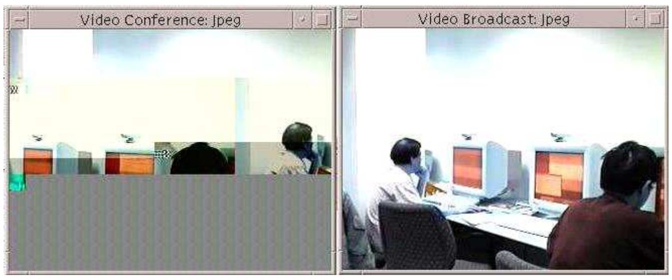

Figure 7: AES: encrypt only 32-bytes using JPEG encoder

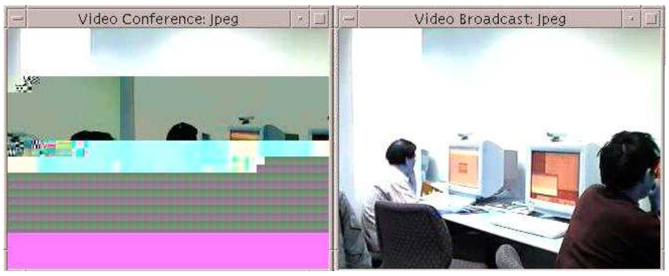

Figure 8: AES: encrypt only 64-bytes using JPEG encoder

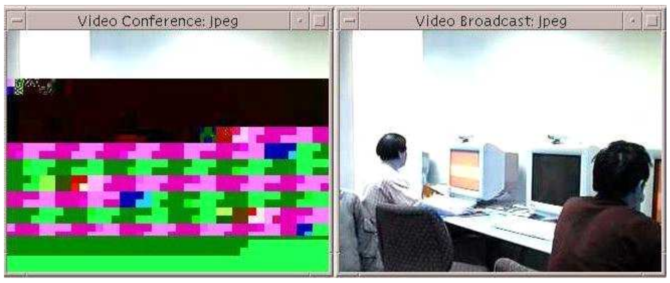

Figure 9: AES: encrypt only 128-bytes using JPEG encoder

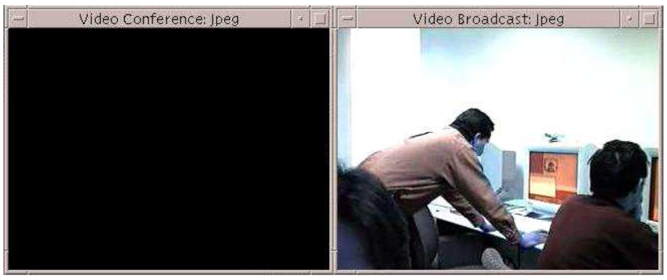

Figure 10: AES: encrypt only 144-bytes using JPEG encoder

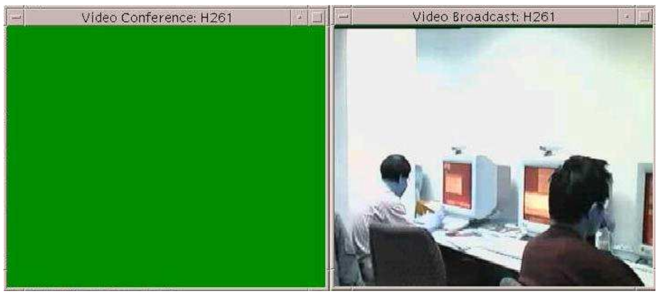

Figure 11: AES: encrypt only 16-bytes using H261 encoder

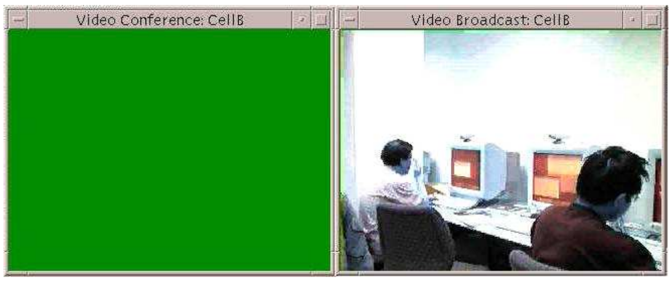

Figure 12: AES: encrypt only 16-bytes using CellB encoder