Regular Paper

NAL Level Stream Authentication for H.264/AVC

Shintaro Ueda,[†] Hiroshi Shigeno[††] and Ken-ichi Okada[††]

The new video coding standard H.264/AVC offers major improvements in the coding efficiency and flexible mapping to transport layers. It consists of a video coding layer (VCL) and a network abstraction layer (NAL). The VCL carries out the coding, and the NAL encapsulates data from the VCL in a manner where transmission over a broad variety of transport layers is readily enabled. Since no security features are offered, an authentication scheme to authenticate the sender and data integrity is needed. In this paper we propose SANAL, a stream authentication scheme for H.264/AVC. Unlike existing schemes that carry out authentication procedures at the packet level, authentication procedures in SANAL are carried out at the NAL level. This makes it possible to set priorities to H.264/AVC-specific data without interfering with the H.264/AVC features. We implemented a SANAL prototype and carried out comparative evaluations on playout rate, communication overhead, and process load. The evaluation results show that the playout rate is improved by 40% compared to existing schemes.

1. Introduction

Multicast streaming applications, as well as the need for security guarantees for such applications are increasing by the day. Data integrity and sender authentication must be ensured as countermeasures to data tampering, spoofing, and repudiation for cases such as multicast streaming of video news and financial stock quotes.

In stream authentication each packet must be authenticated. Consecutive authentication becomes challenging in cases of packet loss. This is especially true when streaming using real-time transmission protocols[1)] on top of connection less best effort services such as User Datagram Protocol, since packet loss is frequently seen[2),3)]. It is easy to solve the packet loss issue by signing each packet with the sender's digital signature but this approach is inefficient in terms of its high computation cost.

Video coding standards have been developed to efficiently code video to reduce the data size. Well-known video coding standards include MPEG-2[4)] used for digital TV and DVD and MPEG-4[5)] used for streaming. A new video coding standard called H.264/AVC[6),7)] has been developed by the ITU-T Video Coding Experts Group and the ISO/IEC Moving Picture Experts Group to improve the coding efficiency well beyond that of MPEG-4.

A number of error-resilience tools to tolerate error have been included in H.264/AVC[8),9)]. However, none have been included that consider security while streaming media over networks. Ensuring data integrity and sender authentication for H.264/AVC streams is still an open issue.

In addition, H.264/AVC includes not only coded video data but also important parameter data that applies to many other data. The importance of data differs, and some data is dependent on other data. The parameter data have high priority since if these data are lost, all coded video data that apply to them cannot be decoded even if received correctly. High priority data must therefore have strong robustness to error. However, in existing stream authentication schemes, only authentication at the packet level is enabled, and it is thus not able to handle the data features of H.264/AVC since all data are assumed to have the same priority.

In this paper we propose a stream authentication scheme that takes the characteristics of H.264/AVC into account by making high priority data robust to data loss while maintaining the flexibility to facilitate mapping to various transport layers. The authentication procedure is carried out at the NAL level, which is between the video coding process and the mapping-to-transport-layer process. The goal of our scheme is to enable efficient and versatile stream authentication for H.264/AVC streams.

The rest of the paper is organized as follows. Section 2 gives an overview of H.264/AVC. Existing stream authentication schemes are dis-

[†] Graduate School of Science and Technology, Keio University

[††] Faculty of Science and Technology, Keio University

cussed in Section 3. Our stream authentication scheme for H.264/AVC is proposed in Section 4. Evaluation results are presented in Section 5. Finally concluding remarks are given in Section 6.

2. Overview of H.264/AVC

A brief description of the H.264/AVC standard is given in this section.

The ITU-T Recommendation H.264 video coding and the ISO/IEC International Standard 14496-10 Advanced Video Coding together developed H.264/AVC, a new video coding standard. H.264/AVC is a generic coding standard designed for broadcast, storage and transmission of a wide range of multimedia applications. A particular focus was improving the coding efficiency, and the new standard therefore enables the bit rate of MPEG-4 to be halved with the same level of fidelity. However, the methods implemented in H.264/AVC to improve the coding efficiency are not important to our proposal in the terms of stream authentication.

2.1 NAL

One characteristic feature of H.264/AVC is that it is separated into a video coding layer (VCL) and a network abstraction layer (NAL). The VCL carries out the encoding tasks. The NAL encapsulates the data from the VCL to enable transmission over packet networks or multiplex environments. Data such as picture slices and parameter sets are sent from the VCL to the NAL and encapsulated into units called NAL units. These NAL units are used in transport layer mapping. This structure of H.264/AVC allows flexibility for operation over a variety of network environments.

The format of a NAL unit is shown in **Fig. 1**. A NAL unit consists of a 1-byte NAL header and a variable byte length raw byte sequence payload (RBSP). Data such as picture slices (coded video data) and parameter sets are stored in the RBSP. The NAL header consists of one forbidden bit, two bits (nal_ref_idc) indicating wether or not the NAL unit is used for prediction, and five bits (nal_unit_type) to indicate the type of the NAL unit. Details on nal_unit_type are given in the next section.

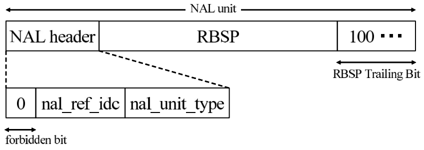


Fig. 1 NAL unit format.

The payload trailing bits are used to adjust the payload to become a multiple of bytes. The trailing bits start with a "1" and are followed by multiple "0s". The end of the payload data is indicated by this "1", the start of the trailing bits.

2.2 NAL Unit Types

The types of NAL units are listed in **Table 1**. nal_unit_type 1-12 are currently defined. nal_unit_type 1-5, and 12 are coded video data called VCL NAL units. The rest of the nal_unit_types are called non-VCL NAL units and contain information such as parameter sets and supplemental enhancement information. Of these NAL units, IDR Pictures, SPS, and PPS are important, and additional descriptions are given below.

An instantaneous decoding refresh (IDR) picture is a picture placed at the beginning of a coded video sequence. When the decoder receives an IDR picture, all information is refreshed, which indicates a new coded video sequence. Therefore, pictures prior to this IDR picture are not needed for this new sequence.

A sequence parameter set (SPS) contains important header information that applies to all NAL units in the coded video sequence. A picture parameter set (PPS) contains header information that applies to the decoding of one or more pictures within the coded video sequence.

H.264/AVC enables handling of multiple sequences in one bitstream, and a sequence contains multiple pictures. Therefore, SPS and PPS are numerated to identify each sequence and picture. Each PPS contains an identifier of which SPS to refer to, and each VCL NAL

Table 1 NAL unit types.

Type	Name
0	[Unspecified]
1	Coded Slice
2	Data Partition A
3	Data Partition B
4	Data Partition C
5	IDR (Instantaneous Decoding Refresh) Picture
6	SEI (Supplemental Enhancement Information)
7	SPS (Sequence Parameter Set)
8	PPS (Picture Parameter Set)
9	Access Unit Delimiter
10	EoS (End of Sequence)
11	EoS (End of Stream)
12	Filler Data
13-23	[Extended]
24-31	[Undefined]

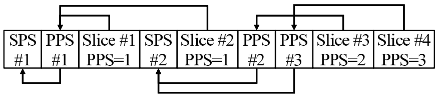


Fig. 2 Relationship between parameter sets and slices.

unit contains an identifier of which PPS to refer to. For example, each coded slice data (coded video data) has a slice header, which includes the PPS identifier. Therefore, by checking the PPS and SPS it is possible to identify which picture and sequence a coded slice data refers to.

The transmission order of parameter sets and slices is restricted; that is, a parameter set must be sent to the decoder before the slice data that refer to that parameter set arrives at the decoder. The relationships of the parameter sets and slices are shown in **Fig. 2**.

3. Related Works: Existing Stream Authentication Schemes

Several approaches to stream authentication have been proposed in order to address the security issues of streaming media[10)~12)].

Gennaro, et al. proposed a scheme that reduces the overhead of the authentication information by amortizing a single digital signature over multiple packets[13)]. A stream is divided into blocks of multiple packets. Each packet contains the hash value of the next packet, and only the first packet in the block is signed. The hash values appended to each packet acts as a chain between the packets. This scheme is very efficient in terms of overhead, but is not robust to packet loss since a loss in a packet will break the chain.

Wong, et al. proposed a scheme where streams are signed using Merkle's signature trees[14)~16)]. In order to tolerate packet loss, each packet is made individually verifiable. The signature of the root node and all hash values of the leaf nodes necessary to compute the root are appended to each packet. However, since each packet carries the signature of the root node the overhead becomes large.

Park, et al. proposed a scheme called SAIDA[17)] (Signature Amortization using IDA), which uses IDA (information dispersal algorithm)[18)]. First, the hash values of each packet are concatenated. Then, the hash of this concatenated value is computed. This value is called the group hash. In SAIDA, only the group hash is signed. Next, the FEC data of the group hash and the signature is generated using the IDA encoding process. Then the FEC data is distributed to each packet in the group. By using IDA, this scheme raises the robustness to packet loss.

The common characteristic of these existing stream authentication schemes is that authentication is carried out at the packet level. However carrying out authentication procedures for H.264/AVC data at the packet level will disable the flexible mapping to transport layers. Also, in packet level authentication, the type of data encapsulated in the packets are not veiled, and an authentication procedure according to priority is not possible. In H.264/AVC, there are dependencies between parameter sets and slices and the importance of NAL units differs from one another. Carrying out authentication procedures at the NAL level makes it possible to set a priority for each NAL unit, and this will enable a new and efficient stream authentication scheme.

4. SANAL: Stream Authentication at the NAL Level

In this section we propose SANAL, a stream authentication scheme for H.264/AVC.

4.1 Overview of Signature Method of SANAL

In our stream authentication scheme, signing and verification procedures are carried out at the NAL unit level. As mentioned in Section 3, this is to maintain the flexibility offered by H.264/AVC when mapping to the transport layer and to specify priorities of different data types in order to improve efficiency.

Our scheme focuses on the following four NAL unit types: coded slice, IDR, SPS, and PPS, since video sequences are composed mainly of these types. The other NAL unit types can be readily addressed by extending our scheme.

An example of a bitstream and the relationship between these four NAL unit types are shown in **Fig. 3**. In Figs. 3–9, S, P, I and C denote SPS, PPS, IDR picture and coded slice NAL units, respectively. The arrows show the relationships between the parameter sets and slices as mentioned in Section 2.2. It is readily seen that the NAL units referred to have higher priority.

Next, an overview of our scheme is given below. Our scheme uses a combination of hashes and digital signatures, as do the exist-

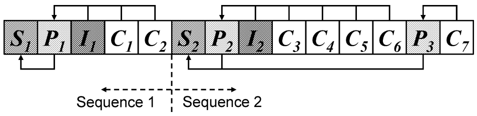

S_1 P_1 I_1 C_1 C_2 S_2 P_2 I_2 C_3 C_4 C_5 C_6 P_3 C_7
Sequence 1
Sequence 2

Fig. 3 Relationship between SPS, PPS, IDR, and coded slices in a bitstream.

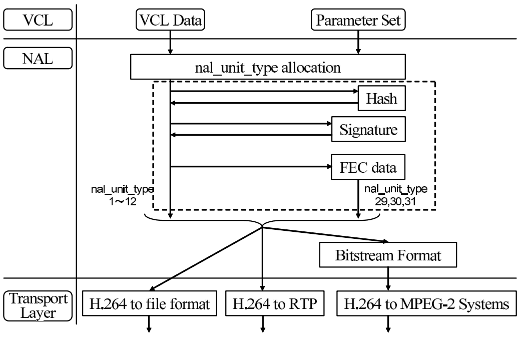


Fig. 4 Procedure flow.

ing schemes. Our scheme also uses a forward error correction (FEC) technique to make high priority NAL units robust to data loss. The nal_unit_types that we use FEC with are SPS, PPS, and IDR. These NAL units are considered high priority, since if they are lost, all NAL units until the next parameter sets and IDR are affected. We use FEC techniques with the following (n, $n-m$) characteristics: when n FEC data packets are generated from the original data, the original data can be reconstructed if m FEC data packets are received. Therefore, $n-m$ packet loss can be tolerated.

Currently, there are several undefined nal_unit_types for further use. We define three new nal_unit_types for authentication, which are used as follows: nal_unit_type 29 indicates the concatenated value of the hash value of each coded slice, nal_unit_type 30 indicates a digital signature, and nal_unit_type 31 indicates FEC data. The flow from the encoding layer to the transport layer including our scheme (inside the dotted-line box) is shown in **Fig. 4**. Our scheme does not take away the flexibility offered by H.264/AVC since the addition is only made in the NAL level.

4.2 Signing Procedure

In this section the signing procedures on the sender side are explained. In our authentication scheme a stream is divided into groups of N NAL units called NAL unit groups. Authentication is carried out in these unit groups. The maximum size of a NAL unit group is defined as N_{max}.

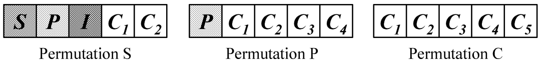


Fig. 5 NAL unit group permutations.

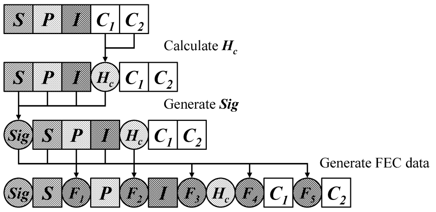

S P I C_1 C_2
Calculate H_c
S P I H_c C_1 C_2
Generate Sig
Sig S P I H_c C_1 C_2
Generate FEC data
Sig S F_1 P F_2 I F_3 H_c F_4 C_1 F_5 C_2

Fig. 6 Signing procedure of Permutation S.

There are three possible permutations of how NAL unit groups can be formed in a stream, and these are shown in **Fig. 5**. For purposes of simplicity, we explain our scheme when N_{max} is set to 5.

(1) Permutation S: Beginning of a sequence
(2) Permutation P: Only the PPS is updated
(3) Permutation C: Contains only coded slices

Permutation S always appears at the beginning of a new sequence. An SPS, PPS, and IDR are followed by coded slices. Permutation P appears when the PPS is updated. A PPS is followed by coded slices. Permutation C is a permutation with only coded slices, and this permutation appears the most frequently.

Each permutation has a different priority, and thus, the signing procedures are carried out accordingly to each permutation. The procedures are explained according to these permutations.

First, the flow of Permutation S is shown in **Fig. 6** and explained as follows.

The hash value of each C is computed and concatenated with each other and expressed as H_c. This H_c is stored in a NAL unit with nal_unit_type 29.

$$H_c = Hash(C_1) \parallel Hash(C_2) \quad (1)$$

This H_c is concatenated with SPS, PPS, and IDR. Then the digital signature Sig as shown in the following equation is generated.

$$Sig = Enc(KEY_s, Hash(S \parallel P \parallel I \parallel H_c)) \quad (2)$$

Here, KEY_s is the private key used in public-key cryptography. The digital signature is stored in a NAL unit with nal_unit_type 30.

Then the FEC data of the concatenation of Sig, S, P, I and H_c is generated. Here, n, the number of NAL units with the FEC data, is set to N, the size of the NAL unit group. The FEC

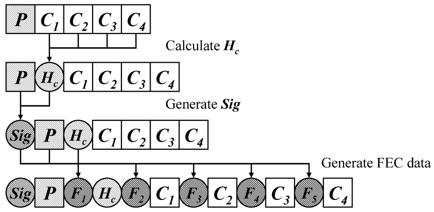


Fig. 7 Signing procedure of Permutation P.

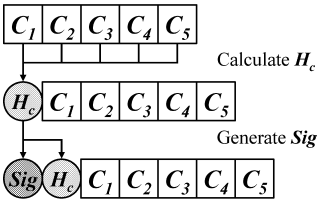


Fig. 8 Signing procedure of Permutation C.

data is stored in NAL units with nal_unit_type 31. Then the FEC data is aligned equally into the NAL unit group. When $n > N$, in the case of data loss, the maximum buffer delay on the receiver side increases; therefore, n is set to N. In Fig. 6, F denotes the FEC data.

Next, the flow of Permutation P is shown in **Fig. 7** and explained as follows.

As in the case of Permutation S, the hash value of each C is computed and concatenated with each other and expressed as H_c. This H_c is concatenated with P. Then the digital signature Sig of the concatenated value is generated. The FEC data of the concatenation of Sig, P, and H_c is generated. The n of the FEC data is set to the size of the NAL unit group. The FEC data is aligned equally into the NAL unit group.

Finally, the flow of Permutation C is shown in **Fig. 8** and explained as follows.

The hash value of each C is computed and concatenated with each other and expressed as H_c. Digital signature Sig is generated. Then Sig and H_c are placed at the beginning of the NAL unit group. In Permutation C, there are no high priority data. Therefore, no FEC data is generated to make data robust to loss.

Figure 9 shows an example of how our signing procedure is applied to a stream of NAL units. The figure is divided into steps that show the original stream, our authentication procedures, and the final NAL units transmitted from the sender side. In this example, N_{max} is set to 5. However it can be seen in the figure, that not all NAL unit groups are groups of five NAL units. This is because the NAL unit groups are divided at the appearance of SPS, PPS and IDR NAL units, respectively.

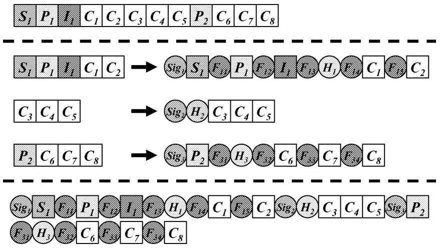


Fig. 9 Example of the signing procedures.

4.3 Verification Procedures

The verification procedures carried out on the receiver side are explained in this section. We focus our explanation on two cases: i) verification when there is no data loss in the high priority NAL units, and ii) verification when data loss occurs in the high priority NAL units. As mentioned above SPS, PPS, and IDR are high priority NAL units. The explanation is carried out for Permutation S (Fig. 6).

When there is no data loss, general verification using digital signature is carried out. After receiving S, P, I, and H_c, the receiver verifies these NAL units using the digital signature. Using KEY_p, the public key of the public-key cryptography, Sig is decrypted, and the hash value of the concatenation of S, P, I, and H_c is computed. The decrypted value and the computed hash value are compared, and if the two are equal, the received S, P, I, and H_c are verified.

When data loss of high priority NAL units occurs, our scheme uses the FEC data to reconstruct the lost NAL units. Reconstruction of the lost data is possible if m out of n FEC data is received on the receiver side. After the data is reconstructed, general verification is carried out, and consecutive authentication is enabled.

In existing schemes all data are handled at the same priority level, and thus, all data have equal robustness to data loss. This becomes a problem at the receiver side in terms of playout. This is because when high priority NAL units are lost, all NAL units referring to it are unplayable. In our scheme, however NAL units with high priority are made robust to loss; therefore, more NAL units are authenticated and played at the receiver side.

5. Evaluation

To evaluate SANAL, we implemented a SANAL prototype and modified the H.264/

Table 2 Performance measurement parameters.

Parameter	Value
maximum size of a NAL unit group: N_{max}	5,. . . ,15
Reconstruction threshold: M	3,. . . ,N_{max}
Packet loss rate (%): p	0,. . . ,40
Expected burst loss length: β	8
Number of frames generated at the encoder: F_n	900
Frame rate (number of frames/sec): F_r	30
SPS insertion interval (msec): S_i	2000,. . . ,5000
PPS insertion interval (msec): P_i	1000,. . . ,2000
Sequence format	CIF

MPEG-4 AVC Reference Software[19] to support authentication. Since the packet loss probability changes over time, it is difficult to evaluate our scheme over real networks. We therefore used the two-state Markov chain loss model to express burst packet losses and ran measurements over virtual networks. We used the two-state Markov chain loss model since it is often used to evaluate stream authentication schemes.

5.1 Implementation Environment

Performance measurements were carried out on a Pentium 4 3.4-GHz CPU, 2.0-GB RAM processor. The implementation of SANAL is written in C/C++. We embedded SANAL to the H.264/MPEG-4 AVC Reference Software JM9.6. Also, 160-bit SHA-1 hash functions and 1024-bit RSA for digital signatures from the OpenSSL library were used, although our authentication scheme is not dependent on any particular type of hash function or digital signature. IDA from Crypto++ library was used as the FEC technique to reconstruct data lost in packet loss. Due to the features of JM9.6, each NAL unit was encapsulated into one RTP packet.

5.1.1 Measurement Parameters

The parameters of the performance measurements are shown in **Table 2**.

The maximum size of a NAL unit group, N_{max} was set to 5,. . . ,15. The reconstruction threshold, M was set to 3,. . . ,N_{max}. The reconstruction threshold is the number of FEC data needed to reconstruct the original data in case of packet loss. The maximum value of the packet loss rate, p was set to 40%. Results from several studies that measured packet loss over the Internet show that packet loss probability via the Internet is much less than 40%[20)~23)]. The expected burst loss length was set to eight packets since the average burst packet loss length over the Internet is less than eight packets.

The number of frames generated at the encoder, F_n, was set to 900 frames, and the frame rate, F_r, was set to 30 frames/sec. The value F_n/F_r is the length of the encoded video sequence measured for evaluation. The SPS insertion interval S_i was set to a random number between 2000,. . . ,5000 msec since it is stated that an IDR is inserted every 2 to 5 seconds[24]. The PPS insertion interval is set to a random number between 1000,. . . ,2000 msec. This is to measure values of Permutation P.

5.1.2 Evaluation Criteria

We evaluated playout rate, communication overhead, and process load. The playout rate is the number of authenticated and playable NAL units on the receiver side divided by the total number of NAL units transmitted by the sender side. In previously proposed schemes, evaluation of the robustness to packet loss is often carried out as authentication rate, that is the total number of authenticated received packets divided by the total number of packets transmitted by the sender side. However, there are cases where authenticated data are unplayable when there are dependencies between data. So the authentication rate is not necessarily equal to the playout rate. Therefore, the playout rate is a more valuable evaluation criteria when dealing with data that carry dependencies. The communication overhead is the amount of the authentication information of SANAL divided by the total amount of H.264/AVC encoder-generated NAL data. Here the authentication information refers to data such as H_c, Sig, and FEC, the data added by applying SANAL to the original H.264/AVC encoder. The H.264/AVC encoder-generated NAL data are the data such as coded slice, IDR, SPS and PPS, the data generated by the original H.264/AVC encoder. The process load is the encoder and decoder process time due to SANAL divided by the process time of the original H.264/AVC encoder. The encoder process time is the total time for hash calculation, signature generation, and IDA encoding. The decoder process time is the total time for hash calculation, signature verification, and IDA decoding.

We will compare SANAL with SAIDA, since SAIDA uses FEC techniques.

5.2 Results

5.2.1 Playout Rate

Figure 10 shows the relationship of the packet loss rate and the playout rate when $N_{max} = 9$ and $M = 5$ and 9.

The playout rate of SANAL is higher than that of SAIDA for both $M = 5$ and $M = 9$. For example, when the packet loss rate is 20% and M=5, the playout rate of SANAL and SAIDA are 0.65 and 0.47 respectively. This shows that SANAL has a 38% better playout rate than SAIDA. When packets are lost, the high priority parameter data are reconstructed in SANAL but not in SAIDA. In SAIDA, only the authentication information is made robust to packet loss. Therefore, when the parameter data are lost, all data referring to the lost parameter data are unplayable even if authenticated. This is the main reason for the difference in playout rate between the two schemes.

In addition, in SANAL, the NAL unit groups are formed according to the appearance of the coded video sequences generated by the encoder, such as permutations S, P and C. In contranst, in SAIDA, a NAL unit group is formed for every N_{max} packets. For example, say a NAL unit group formed by SAIDA is 'C, C, C, C, S, P, I'. This NAL unit group contains data from two different sequences, since S indicates a new sequence. For cases when the coded slices in the first half of the NAL unit group are lost, the parameter data in the last half of the NAL unit group may become unverifiable due to loss of data in a different sequence. Thus, it is inefficient in terms of authentication. Furthermore, the succeeding data that are dependent on these unverifiable parameter data are unplayable and thus inefficient in terms of playout. In SANAL, the parameter data are placed at the start of the NAL unit group, and no data from a different sequence are included. Thus, cases where data are unverifiable and unplayable due to lost data in NAL units of a different sequence do not occur.

5.2.2 Communication Overhead

Figure 11 shows the relationship of the communication overhead and the playout rate.

SANAL has a higher playout rate than SAIDA, as explained in the previous section. However, the overhead of SANAL is also up to 10 times higher than SAIDA. In SANAL, FEC is carried out for IDR slices, which are NAL units that include some of the largest coded video data. On the other hand, in SAIDA, FEC is only carried out for the hashes of packets and digital signatures, and thus the overhead is small. Although the overhead of SANAL is higher compared to SAIDA, it is still less than 10% of the total H.264/AVC encoder-generated bitstream. Also, the FEC data generated for Permutation P are similar in size to the FEC data generated by SAIDA, so the increase in the overhead is mainly due to the FEC data of Permutation S.

5.2.3 Process load

Figure 12 shows the relationship between the reconstruction threshold and the encoder and decoder process load. p is 20 for the decoder process load.

Encoder Process Load

Figure 12 shows that the encoder process

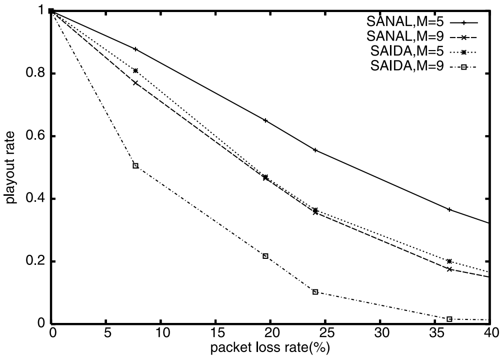


Fig. 10 Relationship between packet loss rate and playout rate.

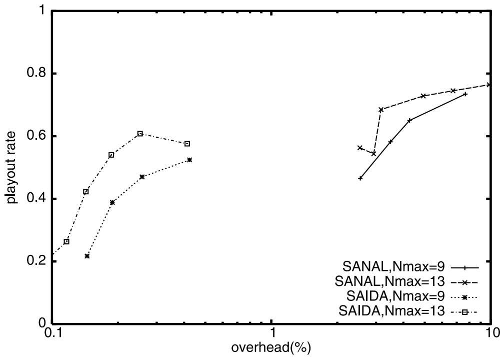


Fig. 11 Relationship between overhead and playout rate.

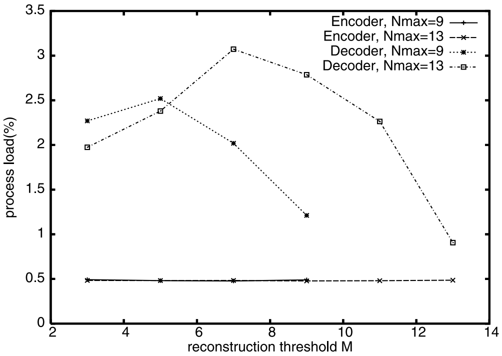


Fig. 12 Relationship between reconstruction threshold and process load.

load is kept below 0.5% and also that the encoder process load has an approximately constant value and is not affected by the value of M or N_{max}. This is due to the fact that the encoding process time of the H.264/AVC encoder is much larger than the time needed by SANAL to generate hashes, digital signatures and FEC data. Also, in SANAL, the number of times procedures such as generation of digital signatures and FEC data which requires a comparatively longer time are kept to a small value.

Decoder Process Load

Figure 12 shows that the decoder process load is kept below 3.5%. The maximum decoder process load is not at the minimum or maximum possible value of M. This is due to the following reasons. When M is a small value, less number of FEC data are needed to reconstruct the lost NAL units. The IDA decoding procedures are inverse matrix calculations, and has characteristics where the larger the number of data becomes the larger the dimension of matrix becomes, which results in a longer procedure time. In other words, the smaller the number of FEC data, the shorter the procedure time. Also, when the value of M becomes large, reconstruction of the lost NAL units becomes difficult since more FEC data are needed, and therefore the IDA decoding procedures are not carried out for the lost NAL units.

Unlike the encoder process load, the bigger the value of N_{max}, the bigger the decoder process load is. Also, the value of M where the decoder process load peaks is a higher value. A higher value of M results in a larger number of FEC data for reconstruction and therefore, a longer process time.

6. Conclusion

We have proposed SANAL, a stream authentication scheme for H.264/AVC. To take account of the features of H.264/AVC, authentication procedures are carried out at the NAL level. We implemented a SANAL prototype, and through our measurement results, we showed the effectiveness of SANAL. The playout rate is improved by 40% compared to existing schemes while the process load is kept below 3.5%.

Acknowledgments This work is supported in part by JSPS Research Fellowships for Young Scientists.

References

1) Schulzrinne, H., Casner, S., Frederick, R. and Jacobson, V.: RTP: A transport protocol for real-time applications, RFC 3550 (2003).

2) Argyriou, A. and Madisetti, V.: Streaming H.264/AVC Video over the Internet, *IEEE Consumer Comm. and Networking Conference*, pp.169–174 (Jan. 2004).

3) Shahbazian, J. and Christensen, K.J.: TSGen: a tool for modeling of frame loss in streaming video, *International Journal of Network Management*, pp.315–327 (2004).

4) ISO/IEC 13818-2: 2000, Information technology-Generic coding of moving pictures and associated audio information (2000).

5) ISO/IEC 14496-2: 2001, Coding of audio-visual objects — Part2: Visual (2001).

6) ITU-T Recommendation H.264. Advanced Video Coding for generic audiovisual services (2003).

7) ISO/IEC International Standard 14496-10 (2003).

8) Wiegand, T., Sullivan, G., Bjontegaard, G. and Lutra, A.: Overview of the H.264/AVC Video Coding Standard, *IEEE Trans. on Circuits and Systems for Video Technology*, Vol.13, No.7, pp.560–576 (July 2003).

9) Wenger, S.: H.264/AVC Over IP, *IEEE Trans. on Circuits and Systems for Video Technology*, Vol.13, No.7, pp.645–656 (July 2003).

10) Ueda, S., Eto, S., Kawaguchi, N., Uda, R., Shigeno, H. and Okada, K.: Real-time Stream Authentication Scheme for IP Telephony, *IPSJ Journal*, Vol.45, No.2, pp.605–613 (Feb. 2004).

11) Ueda, S., Kaneko, S., Kawaguchi, N. Shigeno, H. and Okada, K.: A Real-Time Stream Authentication Scheme for Video Streams, *IPSJ Journal*, Vol.47, No.2, pp.415–425 (Feb. 2006).

12) Challal, Y., Bettahar, H. and Bouabdallah, A.: A Taxonomy of Multicast Data Origin

Authentications: Issues and Solutions, *IEEE Comm. Surveys and Tutorials*, Vol.6, No.3, pp.34–57 (2004).

13) Gennaro, R. and Rohatgi, P.: How to Sign Digital Streams, *CRYPTO 1997*, LNCS1294, pp.180–197 (1997).

14) Merkle, R.: A Certified Digital Signature, *Proc. Conference on Advances in Cryptology*, pp.218–238 (1989).

15) Merkle, R.: A Digital Signature Based on a Conventional Encryption Function., *Proc. Conference on Advances in Cryptology*, pp.369–378 (1987).

16) Wong, C. and Lam, S.: Digital Signature for Flows and Multicasts, *IEEE/ACM Trans. on Networking*, Vol.7, No.4, pp.502–513 (1999).

17) Park, J., Chong, E. and Siegel, H.: Efficient Multicast Stream Authentication Using Erasure Codes, *ACM Trans. Inf. Syst. Security*, pp.258–285 (May 2003).

18) Rabin, M.: Efficient Dispersal of Information for Security, Load balancing, and Fault Tolerance, *J. ACM*, Vol.2, pp.335–348 (1989).

19) http://iphome.hhi.de/suehring/tml/ (Nov. 2005).

20) Loguinov, D. and Radha, H.: Measurement Study of Low-bitrate Internet Video Streaming, *Proc. 1st ACM SIGCOMM Workshop on Internet Measurement*, pp.281–293 (2001).

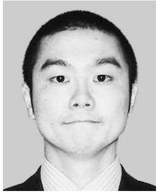
21) Yajnik, M., Moon, S., Kurose, J. and Towsley, D.: Measurement and modeling of the Temporal Dependence in Packet Loss, *Proc. IEEE Conference on Computer Comm.*, pp.345–352 (1999).

22) Paxson, V.: End-to-End Internet Packet Dynamics, *IEEE/ACM Trans. Networking*, Vol.7, No.3, pp.277–292 (June 1999).

23) Boyce, J. and Gaglianello, R.: Packet Loss Effects on MPEG Video Sent Over the Public Internet, *Proc. 6th ACM international conference on Multimedia*, pp.181–190 (1998).

24) Sakaida, S., Iguchi, K., and Gohshi, S.: AVC/H.264 Video Encoder for Mobile Digital Terrestrial Broadcasting, *NHK R&D*, Vol.93, pp.26–31 (2005).

(Received May 12, 2006)
(Accepted September 14, 2006)
(Online version of this article can be found in the IPSJ Digital Courier, Vol.3, pp.55–63.)



Shintaro Ueda received a B.S. degree in information and computer science from Keio University, Japan in 2002, and an M.S. degree in open and environment systems from Keio University in 2005. He is currently working toward a Ph.D. degree in open and environment systems at Keio University. His research interests includes network security.

Hiroshi Shigeno received B.S., M.E. and Ph.D. degrees in instrumentation engineering from Keio University, Japan in 1990, 1992, and 1997. Since then he has been with the Department of Information and Computer Science at Keio University, where he is currently an assistant professor. His current research interests include computer networking architecture and protocols, mobile and ubiquitous computing, and agent computing and communications. He is a member of IPSJ.

Ken-ichi Okada received B.S., M.E. and Ph.D. degrees in instrumentation engineering from Keio University, in 1973, 1975, and 1982. He is currently a professor in the Department of Information and Computer Science at Keio University. His research interests include CSCW, groupware, human computer interaction and mobile computing. He has been a chair of SIGGW, a chief editor of IPSJ Journal, and an editor of IEICE Transactions. Dr. Okada received the IPSJ Best Paper Award in 1995 and 2001 and the IPSJ 40th Anniversary Paper Award in 2000.