Flexible Layered Authentication Graph for Multimedia Streaming

Xinglei Zhu[1], Zhishou Zhang[1], Zhi Li [2] and Qibin Sun[1]
[1]Insititute for Infocomm Research, A*STAR, Singapore
[2]Department of ECE, National Univerisity of Singapore
[1]**{xzhu, zszhang, qibin}@i2r.a-star.edu.sg,** [2]**lizhi@nus.edu.sg**

Abstract*—In this paper, a new flexible layered authentication graph (FLAG) algorithm is proposed for multimedia streaming authentication. While maximizing the verification probability by avoiding authentication path overlapping, this algorithm allows flexible communication overhead in terms of the number of hash links, as well as flexible authentication group size. These flexibilities make FLAG an excellent candidate for multimedia streaming authentication, in that i) in the sender buffering mode, it allows elastic sending delay required by multimedia streaming congestion control; ii) in the receiver buffering mode, it facilitates adaptation to effective network bandwidth; iii) it also has the potential to provide unequal authentication protection (UAP), which is a natural solution for multimedia codestream. Our analysis and experiment results further confirm the validity of our algorithm.*

Keywords—Streaming authentication; hash chaining, layered structrue; flexible authentication graph

Topic area—security and multimedia.

I. INTRODUCTION

With the increasing demand on multimedia streaming in more and more applications, security issues such as integrity and nonrepudiation are becoming increasingly important. Digital signature provides a natural solution to address such issues. However, it is not practical to directly apply general signature scheme on each packet in a multimedia stream, due to the high computation complexity and the communication overhead. Furthermore, in streaming authentication with time concern, there are two conflicting requirements to be balanced - the sender delay and the receiver delay. To address these problems, graph based authentication schemes [1-7] are proposed. In *authentication graph* (AG), packets are connected as *directed acyclic graph* (DAG) as shown in Fig 1. A node corresponds to a packet and a directed edge corresponds to a hash link from its source to its destination. Each packet has at least one directed path to the signature packet. At the receiver side, lost packets are dropped from the graph and a packet is *verifiable* if it has a path to the signature packet. Therefore, a packet need more redundant path to be robust against loss, but this also increases the overhead.

Simple hash chain [1] has low communication overhead because each packet has its hash appended to previous packet and only one packet is signed. Any packet loss will break the chain and all subsequent packets become not verifiable. EMSS [4] makes a great improvement by building multiple hash links for each packet, randomly or deterministically. Expander graph [6] also requires high communication overhead. Recently a butterfly graph based algorithm [3] is proposed. It achieves good authentication probability with low communication overhead and also has good delay property. However, the structure of the butterfly graph [3] is restrictive and it limits its applications.

Notably the graph based authentication schemes can operate in two different modes – sender buffering mode and receiver buffering mode. In the former mode, packets are buffered at the sender side, waiting for the signature packet to be generated, which is subsequently sent first; in the latter mode, packets are generated on-the-fly, but buffered at the receiver side, waiting for the signature packet to arrive and subsequently being verified.

In this paper, a new graph based streaming authentication algorithm - flexible layered authentication graph (FLAG) is proposed. Similar to the butterfly graph [3], it pursues a layered structure and allows only hash links between adjacent layers. Butterfly graph can be seen as a special case of FLAG. In both cases, the construction manages to maximize the verification probability by avoiding authentication path overlapping. In addition, FLAG overcomes the structure limitations of butterfly graph in that it allows flexible number of packets in one group sharing one signature. This property enables the elastic sending delay in the sender buffering mode, making it suitable for multimedia streaming congestion control. Moreover, in the receiver buffering mode, since the authentication graph is not finalized during construction, FLAG allows controllable hash links to be appended to packets, thus it is potentially adaptive to network conditions. Furthermore, the flexible construction of FLAG also enables the potential to provide unequal authentication protection (UAP) [5], which is a natural solution for protecting multimedia codestream. The UAP could be pursued by assigning different number of hash links to each packet according to its importance, similar to the work on JPEG2000 in [2].

The paper is organized as follows. Section 2 analyzes the authentication path overlapping and characters of the layered structure. A detailed description of FLAG is also given in this section. Section 3 compares FLAG with two state-of-art algorithms EMSS and butterfly graph. Section 4 draws the conclusions.

II. FLEXIBLE LAYERED AUTHENTICATION GRAPH

We concern two key points in this section when constructing an authentication graph: 1) How to avoid ill-designed structure and 2) how to efficiently utilize the communication overhead. In section II.A we address the first concern by analyzing authentication path

redundancy to find out how to maximize the authentication probability in the layered structure. Then FLAG is designed to deal with the both concerns in Section II.B.

A. Authentication Path Redundancy

In the authentication graph, an *authentication path* of a packet P_i is defined as a path starting from P_i and ending at the signature packet S/P_0, such as $P_{12}\rightarrow P_6 \rightarrow P_2\rightarrow S$ in Fig. 1. At the receiver, lost packets are removed from the graph and a packet is unverifiable unless it has an authentication path. For example in Fig. 1 if packet P_5 is lost then P_9 is unverifiable. *Verification probability* is defined as the ratio of the number of received and authenticable packets over the number of received packets, i.e. the authentication graph's robustness against packet loss. The verification probability of a packet depends on the number of its authentication paths and its distance to the signature packet S in each path. Generally more authentication paths and shorter distance to S lead to higher authentication probability.

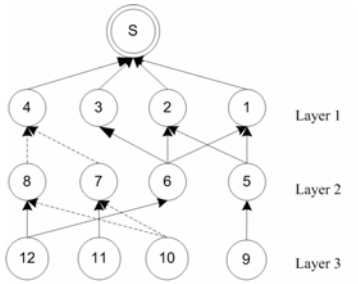

Fig. 1. Layered authentication graph. The dashed edges represent two intersected authentication paths, both starting from P_{10}. .

Any authentication graph can be considered as a layered structure with intra-layer hash links and inter-layer hash links. Consider the set of packets connected to the signature packet as the first layer, the set of packets that is not in the first layer but connected to a packet in the first layer as the second layer. The process goes on till every packet is assigned a layer. In such a layered view, there may be intra-layer hash links and two kinds of inter-layer hash links that from former layer to latter layer (forward inter-layer hash link) and that from latter layer to former layer (backward inter-layer hash link). Because of the way the layers are formed, forward inter-layer hash links are only between adjacent layers.

Authentication path overlapping refers to that two authentication paths of P_i intersect at another packet P_j before they reach the signature packet. If P_j is unverifiable due to the transmission loss, P_j is unable to be authenticated from either path. If P_i does not have other authentication path, it is also unverifiable. In other words, from P_i's perspective the authentication path is not fully utilized. For example, in Fig.1 two authentication paths $P_{10}\rightarrow P_8 \rightarrow P_4\rightarrow S$ and $P_{10} \rightarrow P_7 \rightarrow P_4\rightarrow S$ intersect at P_4. If P_4 is lost, P_{10} is unverifiable although it has two authentication paths. In EMSS [4] it is reported that some settings such as 1-2-3-4-5-6 (referred to the original notation in [4]) are significantly worse than some other settings such as 5-11-17-24-36-39. Authentication path overlapping is the reason for the bad performance of the ill settings. In [5] it is also proved that when the packets on individual authentication paths of P_i are authentication independent, authentication probability of P_i is maximized. Thus to maximize the authentication probability of the latter layers, authentication path overlapping should be avoided if possible. Obviously removing any hash links in an arbitrary AG does not introduce new authentication path overlapping. Thus we remove all the inner-layer hash links and the backward inter-layer hash links and study only the case with only forward inter-layer hash links. Whether an AG could be path overlapping-free depends on the number of nodes N and the number of packets directly linked to the signature packet, i.e. the number of nodes in the first layer.

Signature packet is special among all the packets. If the signature packet is lost, the whole segment is not verifiable. To verify the stream packets, generally the signature packet has to be received, which can be realized by automatic repeat request (ARQ). Connecting all the stream packets to the signature packet is good for authentication purpose but it greatly increases the size of the signature packet. In transmission data packet is further split into transmission units if its size exceeds the size of maximum transmission unit (MTU). For example, if the size of MTU is 1500 Bytes, size of each hash is 16 bytes and size of the signature is 128 bytes, then 68 hash links are affordable for the signature packet to be confined in a single transmission unit. Since a data packet is considered lost unless all the transmission units are received, further increasing the size of signature packet will increase its lost rate or requires more retransmission rate if the lost rate is to be kept below some level. As a result, the number of hash links contained in the signature packet is limited in practice.

Now let us configure the conditions for an AG containing only forward inter-layer hash links to be authentication path overlapping free. Suppose each data packet contains no more than k incoming hash-links and the signature packet contains no more than m incoming hash-links. To maximize the authentication probability for the data packets, the first layer should contain m data packets and for data packets in other layers they should have k outgoing hash links each. Since the first layer maximally contains $m*k$ incoming hash links, the second layer should contain no more than m packets. Due to the same reason all the packets in the latter layers should contain no more than m packets. Now consider a packet P_l in the last layer L, it has k outgoing hash links connected to k packets in layer L-1. Since there is no authentication path redundancy, these k packets altogether have $k*k$ outgoing hash links connected to $k*k$ packets in layer L-2, and so on. Thus in layer L-r the number of P_l's authentication ancestors is k^r, especially in the first layer it is k^{L-1}. Since we have maximally m packets in the first layer, there must be $k^{L-1}\leqslant m$ to avoid authentication path redundancy. Thus we have $L \le \lfloor \log_k m \rfloor + 1$ and the total number of packets $N \le m * (\lfloor \log_k m \rfloor + 1)$ (1).

There are two possible situations for any given N, m, k, as below.

1) $N \le m * (\lfloor \log_k m \rfloor + 1)$. An redundancy-free graph could be built. The algorithm is given in Section II.B. There is no authentication path redundancy in this case.

2) $N > m * (\lfloor \log_k m \rfloor + 1)$. We find out the maximal j<k that matches $N \le m * (\lfloor \log_j m \rfloor + 1)$ and build an authentication graph according to the algorithm in Section II.B. There must be such a j existing because when j=1, $m * (\lfloor \log_j m \rfloor + 1)$ is infinity. Then the redundant hash links are added into the authentication graph to further improve the authentication probability of some packets. Here the redundancy acts as the additional authentication probability benefits. The authentication graph could also be constructed by modifying the algorithm in Section II.B, as shown later.

B. *Flexible Layered Authentication Graph (FLAG)*

In this section FLAG is presented. As discussed in the previous section, we assume that the packet number restriction in (1) is fulfilled and each data packet contains k incoming hash links. The maximal number of incoming hash links packet P_j takes is called the *incoming hash capacity* of P_j, noted as $C[P_j]$. The difference between $C[P_j]$ and the number of incoming hash links of P_j is called the *incoming hash vacation* of P_j, denoted as $V[P_j]$. We also denote the *number of outgoing hash links* of P_j as $H[P_j]$. Each packet has an authentication descendance set (denoted as $D[P_j]$) containing the index of its all authentication descendances. The process of the FLAG algorithm is:

Step 0: Initialize $D[P_j]$ of all the packets as empty.

Step 1: Sequentially apply following process for the packets $P_{R,m,\ldots,}P_{R,1}$ in layer R (We start from the last layer $L = \lfloor \log_k m \rfloor + 1$.)

For packet $P_{R,j}$, first check the number of outgoing hash links $k_{R,j}$. Find out the packet $P_{R-1,a}$ in the layer R-1 with maximal hash vacation among the packets in layer R-1 and $D[P_{R,j}] \cap D[P_{R-1,a}] = \varnothing$. Connect $P_{R,j}$ to $P_{R-1,a}$ and revise $D[P_{R-1,a}] = D[P_{R,j}] \cup D[P_{R-1,a}]$. Repeat the process for $P_{R,j}$ until the outgoing hash links of $P_{R,j}$ reaches $k_{R,j}$.

Step 2: Repeat step 1 for layer R-1 and go on until the packets in the 2nd layer are all processed.

Step 3. Connect the packets in the first layer $P_{1,1},\ldots,P_{1,m}$ to the signature packet using hash links.

Since no additional assumption are made of the value of m and k in the above process, they can be any value as long as $m>k$ (say, m is 5 and k is 2). Thus the layered structure is much more flexible than the butterfly graph. Specially, when m is 2's power and k is 2, FLAG generates an authentication graph isomorphic to the butterfly graph. This merit makes the number of packets in the same authentication group flexible while keeping high verification probability and thus FLAG allows elastic sending delay which is required by multimedia streaming congestion control.

As an example, a layered authentication graph is shown in Fig. 2, with m=5, k=2 (same for all packets) and N=15.

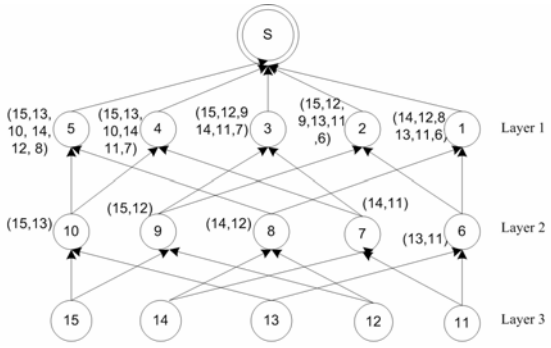

Fig. 2. Layered authentication graph with m=5, k=2 and N=15. The parentheses represent the set of authentication ancestors of each packet.

There is an interesting property when the hash capacity and outgoing hash links are the same for all the packets. The algorithm generates such a well distributed hash links between layer L and layer L-1 that they can be directly copied to other layers, as long as (1) is valid, and hence it has lower complexity to construct the graph. The proof is omitted here due to the space limitation.

In the receiver buffering mode, if the multimedia transmission rate changes, say, more communication overhead is allowed, the number of outgoing hash links for the unprocessed packets could be increased to achieve better authentication probability. However the non-overlapping condition in (1) may not be valid. We simply remove all the constrains related to authentication descendance set $D[P_j]$ in the FLAG algorithm Fig. 3 gives an example, where k increases to 3 when processing P_8 and keeping m, N fixed. In the receiver buffering mode, since the authentication graph is not finalized till the signature packet is sent out, the structure of FLAG can be changed during the transmission process and thus it facilitates adaptation to channel bandwidth.

The FLAG algorithm can be further extended to a situation where each packet has different hash capacity and different number of outgoing hash links. The overlapping-free property may be no longer kept in this case. This flexibility makes it a potential algorithm for UAP by setting different hash capacity and different number of outgoing hash links for packets with different importance, similar to the scheme applied on JPEG2000 in [2].

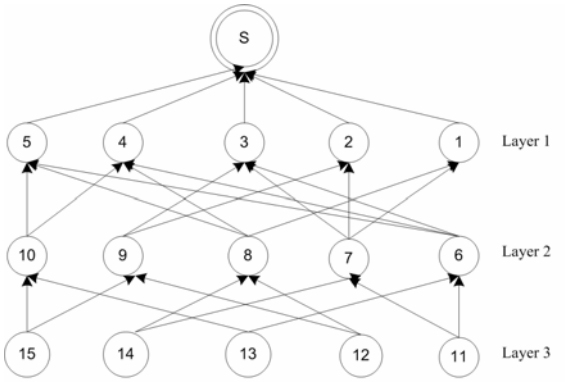

Fig. 3. Layered authentication graph with m=5 and N=15.

III. EXPERIMENTS

In this section we compare the FLAG with EMSS and butterfly graph, which are among the best performing authentication graphs. FLAG provides a more generic graph structure, where butterfly can be considered as a special case.

A. *Authentication probability with different packet loss rate*

Fig. 4 shows the authentication probability change with the packet loss probability. The number of stream packets N is 1024, the number of packets per layer m and the number of hash link per frame k are shown in the legend. For EMSS, we randomly connect a packet P_j to two other packets in the range of P_{j-1}, …, P_{j-128}.

From Fig. 4 we can see that when the average hash link per packet is 2 and the number of packets per layer is 128 for both FLAG and EMSS, FLAG obviously outperforms EMSS, especially when the packet loss rate is high. FLAG generates a graph that is isomorphic to the butterfly graph when m is 2's power and k=2, their performance is indiscriminative in this case. Since in this situation (1) is valid, the path overlapping-free graph exists and there is no

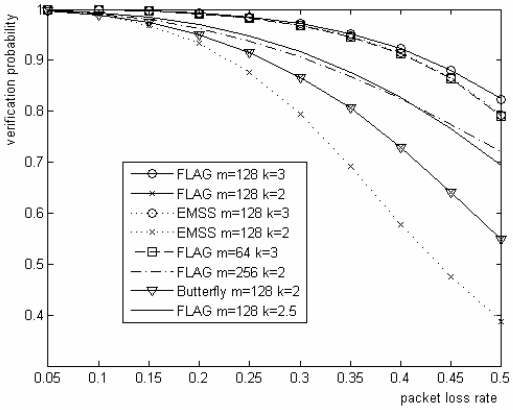

Fig 4. authentication probability of FLAG and EMSS with change of packet loss rate.

redundant link to improve the authentication probability. Compare with FLAG, random connections in EMSS do not explicitly address the authentication redundancy problem. We can also see that further increasing the number of packets per layer to 256 helps to improve the authentication probability, at the expense of nearly double the signature packet's size. Increasing the number of hash links to 3 greatly improves the performance, since there are more hash links to increase the authentication probability. FLAG still outperforms EMSS in this case but the difference is not as large as in the previous setting. Suppose in the receiver buffering mode the effective channel bandwidth increases during the transmission and the average number of hash links adaptively increases from 2 to 3 for half of the stream packets, the authentication probability increases to a level (the line of "FLAG m=128 k=2.5") much better than k=2 case as shown in Fig 4. Finally, when k=3, setting m to 64 in FLAG generates almost the same result as setting m to 128 in random EMSS, which means FLAG achieves the same authentication performance with a much smaller signature packet.

B. Authentication probability with different communication overhead

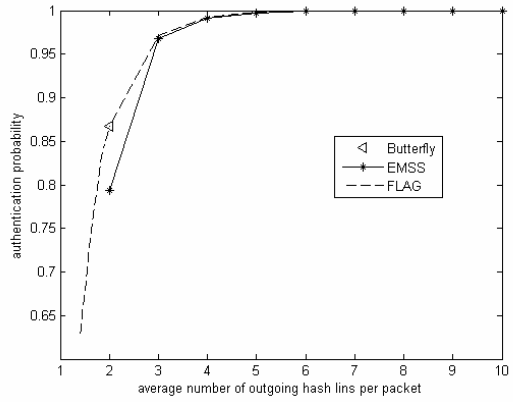

Fig 5. Authentication probability with change of average number of outgoing hash-links per packet

In this subsection we compare the authentication probability of FLAG and EMSS under different communication overhead, which is represented as the average number of incoming hash links per packet. In the original proposal of EMSS the algorithm requires a minimal number of hash links of 2. Fig. 5 shows the authentication performance of FLAG and EMSS with different communication overheads at packet loss rate p=0.3. (As a special case of FLAG, the butterfly algorithm corresponds to a point on the performance curve of FLAG, as shown in Fig. 5.) When the overhead exceeds 3 hashes per packet the performance of these algorithms are very close. When the communication overhead is below 2 hashes per packet, the authentication probability of FLAG drops very quickly since it does not have enough hash links to keep robust against packet loss.

IV. CONCLUSIONS AND FUTURE WORKS

In this paper, a new flexible layered authentication graph (FLAG) algorithm is proposed for multimedia streaming authentication. While maximizing the verification probability by avoiding authentication path overlapping, this algorithm allows flexible communication overhead in terms of the number of hash links, as well as flexible authentication group size. These flexibilities make FLAG an excellent candidate for multimedia streaming authentication, in that i) in the sender buffering mode, it allows elastic sending delay required by multimedia streaming congestion control; ii) in the receiver buffering mode, it facilitates adaptation to effective network bandwidth; iii) it also has the potential to provide unequal authentication protection (UAP), which is a natural solution for multimedia codestream. Analysis and experiment results further confirm the validity of our algorithm. Our future works include developing UAP algorithms based on FLAG for certain media types such as H.264 video stream and further study of the authentication graph structures for media streaming by taking media coding structure into consideration.

REFERENCES

[1] R. Gennaro and P. Rohatgi. "How to Sign Digital Streams", *in Advances in Cryptology-CRYPTO'97*.

[2] Z. Zhang, Q. Sun, W. Wong, J. Apostolopoulos and S. Wee, "An optimized Content-Aware Authenitcation Scheme for Streaming JPEG-2000 Images Over Lossy Networks", *IEEE Trans. on Multimedia*, 2007.

[3] Z. Zhang, Q. Sun and W. Wong, "A proposal of Butterfly-graph Based Stream Authentication over Lossy Networks",*IEEE International Conference on Multimedia and Expo*, 2005.

[4] A. Perrig, R. Canetti, J. Tygar and D.Song, "Efficient Authentication and Signing of Multicast Streams over Lossy Channels", *Proc. Of IEEE Symposium on Security and Privacy*, 2000.

[5] Z. Li, Q. Sun, Y. Lian and C. Chen, "Joint Source-Channel-Authentication Resource Allocation and Unequal Authenticity Protection for Multimedia over Wireless Networks", *IEEE Trans. on Multimedia*, to be published on June, 2007.

[6] D. Song, D. Zuckerman and J. D. Tygar, "Expander Graphs for Digital Stream Authentication and Robust Overlay Networks", *Proc of IEEE Symposium on Research in Security and Privacy*, 2002.

[7] S.Miner and J. Staddon, "Graph-based Authentication of digital Streams", *Proc. Of IEEE Symposium on Research in Security and Privacy*, 2001.