Key Management Schemes for Peer-to-Peer Multimedia Streaming Overlay Networks

J.A.M. Naranjo[1], J.A. López-Ramos[2], and L.G. Casado[1]

[1] Dpto. de Arquitectura de Computadores y Electrónica
Universidad de Almería, Spain
jmn843@alboran.ual.es, leo@ual.es
[2] Dpto. de Álgebra y Análisis Matemático
Universidad de Almería, Spain
jlopez@ual.es

Abstract. Key distribution for multimedia live streaming peer-to-peer overlay networks is a field still in its childhood stage. A scheme designed for networks of this kind must seek security and efficiency while keeping in mind the following restrictions: limited bandwidth, continuous playing, great audience size and clients churn. This paper introduces two novel schemes that allow a trade-off between security and efficiency by allowing to dynamically vary the number of levels used in the key hierarchy. These changes are motivated by great variations in audience size, and initiated by decision of the Key Server. Additionally, a comparative study of both is presented, focusing on security and audience size. Results show that larger key hierarchies can supply bigger audiences, but offer less security against statistical attacks. The opposite happens for shorter key hierarchies.

Keywords: Conditional access, peer-to-peer networks, key distribution, secret sharing, digital rights management.

1 Introduction

Key distribution schemes designed for live streaming peer-to-peer (P2P) networks can exploit the fact that for a TV channel every peer can share encryption keys while playing the content. This reduces the Key Server workload dramatically. Still, several problems must be addressed:

(a) Peer tracking.
(b) Clients churn (clients joining and leaving frequently) and flash crowds.
(c) Large audience sizes.
(d) Keeping keys private all the way from the Key Server to the last peer.
(e) Avoiding illegal key sharing (traitors).

In relation to *(a)*, possible solutions are Gossip Protocols [6] and Distributed Hash Tables [20]. Centralized tracking is not viable for large audience sizes.

Regarding *(b)*, churn directly affects *quality of service* (QoS) in schemes that are event-driven. This kind of schemes perform a rekeying operation whenever

O. Markowitch et al. (Eds.): WISTP 2009, LNCS 5746, pp. 128–142, 2009.

a client joins/leaves the channel, to preserve *backward* and *forward* privacy (a client should not be able to decrypt the content before and after its subscription period, respectively). A scheme of this kind is presented in [14]. Churn effects depend also on the topology chosen, which can be mesh-like or multi-tree [15][16].

Flash crowds occur at the beginning of high interest events. The Key Server may be overwhelmed by the large amount of join requests. Dealing with this problem is still a challenge. Making the join process as simple as possible helps to reduce flash crowds effects.

Audience size, *(c)*, should not be limited by the key distribution method used. The rekeying process usually sets an upper bound for the number of clients that can be served. The most common solution is to divide the audience into groups of users [7][13].

Regarding *(d)*, some solutions have already been proposed. In [14] key distribution is performed on a tree fashion. In this model, each node shares a secret key (KEK: *Key Encryption Key*) with its children. This key is used to encrypt communications between node and childs. The arrival of a new peer implies setting new KEKs all the way from the peer to the root node. The same process must be carried out when leaving. This may lead to *quality of service* (QoS) degradation in high churn situations.

In [20] key exchange requires the establishment of trust relationships among peers, which implies mutual authentications under a secure channel. This consumes time and bandwidth. Furthermore, the *leave* operation is complex and involves several communications among the leaving peer, its neighbors and the Key Server.

Finally, traitors *(e)* are an ever-present problem in the industry. Several traitor-tracing solutions exist, such as watermarking [5][23], but still the heart of the matter is unsolved. Probably the solution will come from Trusted Computing techniques [3][1][2].

This paper introduces two key distribution schemes for multimedia streaming peer-to-peer networks that address problems *(c)* and *(d)*. The first one, *Complete Key Distribution Scheme*, is an extension to the traditional solution that suits (and takes advantage of) the features of peer-to-peer networks. The second one, *Share Based Distribution Scheme*, makes use of Secret Sharing techniques and is simpler in the sense that the key distribution process throughout the peer-to-peer network does not require any encryption/decryption. Both schemes employ a key hierarchy that can vary dynamically from 1 to 3 levels in the case of *Complete Key* and from *1 to 2*, in the case of *Share Based*. The goal of this dynamic variation is to offer a trade-off between security and audience size. Each key hierarchy length suits better a different scenario, depending on the size of the audience. Both schemes also avoid the necessity of key agreements among peers.

Briefly, the Complete Key approach consists of the distribution of messages which contain CWs (the key which encrypts the multimedia stream). These messages are named $CW - messages$. Each CW-message is encrypted with the key in its upper level and distributed throughout the network. On the other hand, in the Share Based approach different CW-messages are generated and

distributed for each CW. Each message contains a different piece of information called "share" which is not encrypted. The CW can be reconstructed with a given number of different shares and a token provided by the Key Server. Peers themselves distribute the shares.

Problem *(b)* and *(e)* are part of our future work lines. Problem *(a)* is out of the scope of this work.

Sections 2 and 3 introduce some preliminary considerations. Sections 4, 5 and 6 present the two schemes and compare them, respectively. Finally, Section 7 discusses the conclusions and future work lines.

2 Scenario

The proposed schemes have been designed for a scenario that is introduced next. The multimedia stream is delivered to the clients by using a peer-to-peer network. A Contents Server injects the stream into the network by connecting to a small set of clients (preferably those with higher bandwidth and lower latency) that start the distribution chain. A Key Server distributes the keys. Distribution of given key is done via the peer-to-peer network or directly via the TLS protocol, depending on the hierarchy level the key belongs to.

Constraints regarding overlay topology are very relaxed: it can be either mesh-like or multi-tree. The former is better suited by the Complete Key approach. The Share Based approach fits better multi-tree structures.

Keys are not multiplexed into the multimedia stream, and there is only one restriction for the stream format. Every data packet is encrypted with a given key, so the key identifier must be included in the packet. This is done to avoid multiplexing of key change messages into the stream at a given rate. Since there is no guarantee that the whole stream will reach every peer, a peer might miss a key change message and decrypt content packets with the wrong key. An integer data type (32 bits) can be used as key identifier: a negligible portion of bandwidth is consumed and quality of service is not affected.

Both key distribution schemes run separately from the content network, so they can be deployed easily and on the top of an already operating multimedia peer-to-peer platform without peer tracking or topology restrictions.

Audience size measurement can be performed by several means [11], such as *probabilistic polling* [10], *use of epidemic algorithms* [9] and *random walks* [17]. How measures are taken is out of the scope of this paper.

3 A Dynamic Key Hierarchy

Key hierarchies are not new: they have been extensively used in the Digital Video Broadcast business for near two decades now [8][12][22]. TV conditional access systems usually rely on a static hierarchy. The schemes proposed in this paper introduce the possibility of increasing/decreasing the number of levels employed in the hierarchy, depending on the audience size. The smaller the audience is, the shorter the hierarchy is, and vice versa.

Adding a new key on top of the hierarchy at a given moment allows refreshing those below more frequently. This is due to the fact that the highest key must be refreshed against the Key Server, but the others can be renewed via the peer-to-peer network, as will be shown next. Thus, for high interest events a new key can be introduced on the top. The frequency of the refreshment decreases as we climb on the hierarchy. In addition to this, enlarging the hierarchy allows to supply bigger audiences, since the number of communications against the Key Server decrease dramatically.

The problem with large key hierarchies is that the top-level key stands for long periods of time: an attacker has then more time to guess it. Dynamic changes of the key hierarchy seek avoiding this risk when audience size does not require the use of so many key levels. By shortening the hierarchy the top level key is refreshed at a higher rate.

The following sections explain how both schemes work, and how key hierarchies are changed dynamically.

4 The Complete Key Distribution Scheme

As mentioned in section 1, the idea behind this approach is to deliver the key that encrypts the stream (named *CW* for *Control Word*) in a CW-message by using the peer-to-peer network itself. This scheme suits mesh-like networks gracefully, introducing very little overhead and achieving a good security level, as will be shown later.

The key hierarchy length can vary from one to three levels. Next, the three flavors are presented. How to change among them will be explained hereafter.

The One-Level flavor is the simplest of the three, and requires contacting the Key Server at a high frequency (within minutes). The Two-Level scheme increases complexity by adding a second level, allowing avoiding contact with the Key Server for hours. Finally, the Three-Level flavor permits a client to play the content for even days without connecting to the Key Server, by adding a third key over the two others.

4.1 One-Level Flavor

The One Level Complete Key approach employs a centralized client-server architecture for distribution of the key that encrypts the stream. This key, *CW*, is refreshed periodically within seconds or minutes.

The refreshment process requires the client to authenticate against the Key Server. *CW* is sent under a secure protocol such as TLS. Connections are client-server, thus not exploiting the benefits of the peer-to-peer network. For performance enhancement, several successive *CW*s can be sent in every communication, thus dividing the number of necessary connections to the Key Server by the number of *CW*s in each message.

The simplicity of the refreshment process has a dramatic impact on the audience size for this flavor: only a very small set of clients can be maintained,

since every client asks the Key Server directly at a very high frequency. Moreover, secure protocols as TLS impose an overhead on communications (time and number of messages).

On the contrary, the security level is very high, since high rate refreshment avoids statistical attacks on CW: the impact of an attacker guessing the key at a given time (by brute force or a weakness of the symmetric encryption) is very low. Furthermore, no sensitive information is stored for a long time in the client side. The main risk is the client distributing CW voluntarily.

4.2 Two-Level Flavor

The multimedia stream is encrypted with CW, which is renewed at a high frequency (similar to the One-Level case). CW is no longer distributed via client-server connections with the Key Server: it is encrypted with SK (*Service Key*), the second level key in the hierarchy. The result is encapsulated in a CW-message, which is released into the peer-to-peer network. Peers themselves distribute the CW-message as needed. The message may contain several successive refreshments of CW, if desired. This, as mentioned above, enhances performance of the network, but weakens scheme's security.

SK lifetime is in the order of hours: that means that after this time clients must ask the Key Server for a new SK. The request is done under TLS.

Due to the introduction of a second key, clients can receive refreshments of CW and decrypt the stream for hours without communicating with the Key Server (there is a small set of clients that communicate constantly with the Key Server, actually: those are the ones which introduce the CW-messages into the peer-to-peer network). Thus, a larger number of clients can be served. What's more, communications among peers do not suffer from overhead since they are done under a non-encrypted channel (CW-messages are encrypted themselves).

Privacy analysis for CW is similar to that shown for the One Level Complete Key approach. The risk of SK being compromised is low, since it is sent to clients under TLS. As in One Level Share Based, an attacker would have to guess the key a reasonable time before it is renewed. Guessing it later would be useless.

4.3 Three-Level Flavor

The Two Level Complete Key approach encrypts the stream with CW, and CW with SK. The present flavor, as can be guessed, employs a new key level to distribute SK. The new key is called T (*authorization Token*). The Key Server encrypts SK and includes it into a message named *SK-message*, which is injected into the peer-to-peer network. T is renewed upon days or even weeks via client-server TLS communications against the Key Server.

T's large lifetime allows to dramatically increase the period for which a client can decrypt the content without communicating with the Key Server. Hence, very large audiences can be reached by using this flavor, say, tenths of thousands. Security, on the other hand, decreases because of the same reason: an attacker would have several days or even weeks to decrypt T, which would lead to being

able to play the content. T's lifetime can be adjusted as necessary to make attackers' life harder (losing audience size, though).

4.4 Changing among Flavors

The Key Server initiates change from one flavor to another when the audience size surpasses a given threshold (audience size can be measured by any of the methods exposed in Section 2). When exceeding, say, several hundreds of clients, the Key Server may decide to change from One-Level flavor to Two-Level flavor. If the expected audience is very large, then the change may be from One-Level to Three-Level directly. The opposite is also valid: when audience decreases sufficiently the Key Server may decide to shorten the key hierarchy. This section explains how these changes are done.

The necessary information for a change is included in the CW-messages. That information consists of three fields: the mode in which the system currently run (field *Current Level*) how many messages of this kind are left before the next hierarchy change (field *Left*), and which mode will be used then (field *Next Level*). Hence, a CW-message contains:

- One or more successive CWs (not encrypted for One-Level, encrypted with SK for Two and Three-Level).
- "Current Level" field.
- "Left" field.
- "Next Level" field.

Joining peers need the field "Current Level" so they can know the mode the system is running on.

Figure 1 shows an example: the system operates in One-Level mode and plans to change to Two-Level mode. The figure depicts the CW-messages generated by the Key Server. The example starts at moment $t = 0$. Change is finished at moment $t = f$. Each CW-message contains three CWs. At moment $t = 0$ the Key Server generates the first CW-message, and an SK-message which contains a brand new SK. When receiving the CW-message under TLS, a client knows that a mode change is happening soon, because of the values in "Left" and "Next Level". Then, the client obtains the SK-message from the Key Server under TLS before $t = f$. The CW-message at $t = m$ is the last one served under TLS by the Key Server. At moment $t = f$ the client retrieves the CW-message from the peer-to-peer network and decrypts its content by using SK. From that moment on the system runs in Two-Level mode, and fields "Left" and "Next Level" indicate 0, until a new mode change is started.

Instructions for every possible flavor change are detailed in Figure 2.

4.5 Security and Efficiency Considerations for the Complete Key Approach

It is clear that as key hierarchy length increases, audience size follows: this is due to the fact that the major part of the clients only contact the Key Server to

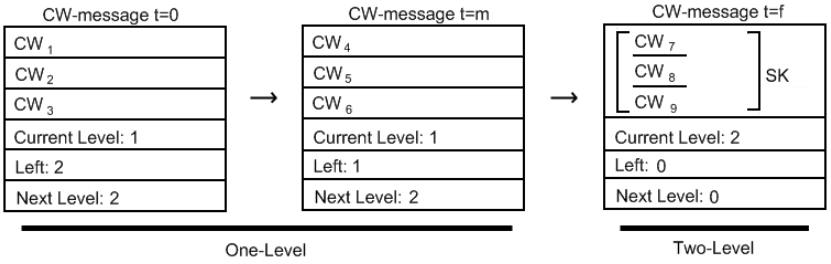

Fig. 1. Mode change from One-Level to Two-Level. A valid SK-message must be obtained by the client between $t = 0$ and $t = f$.

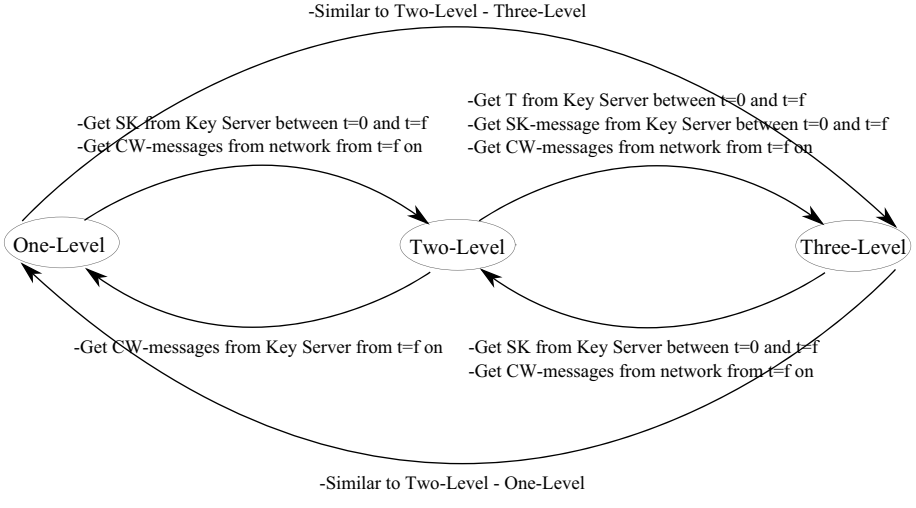

Fig. 2. Mode changes for the Complete Key Approach from clients' view. The process starts at *t=0* and ends at *t=f*.

refresh the highest key (SK in Two-Level and T in Three-Level). The higher the key is, the longer its lifetime, hence less connections are required. Furthermore, to avoid bottlenecks when changing the highest key, requests for it can be made at a random moment before expiration of the current one. If requests from other peers follow a uniform distribution then a homogeneous requests rate will be obtained. This makes workload for the Key Server more bearable.

Security is based in symmetric encryption for key distribution within the peer-to-peer network and in TLS communications with the Key Server. If keys are generated in a secure way then it will be very difficult for an attacker to break a key in a time short enough to get some benefit out of it.

However, having keys with long lifetime has its drawbacks: if the key on top is compromised, then the attacker will gain access to the content for a long time.

What's more, the attacker may share the content and the key with others. That is why it is important (1) to adjust keys' lifetimes as tight as possible and (2) to keep the system running in a mode as low as possible in the three flavors range. The use of a dynamic key hierarchy provides more security to key distribution if flavor changes are planned smartly on the Key Server side.

5 The Share-Based Key Distribution Scheme

The problem with the Complete Key Distribution approach is that its One-Level flavor can only be used for tiny audience sizes. TLS communications consume time and resources and only a little set of clients can be supplied using the One-Level flavor. This is not useful.

The Share-Based is similar to the Complete Key scheme mentioned above but differs in the fact that it makes use of *secret sharing* techniques to perform key distribution. Due to this, the scheme shows some desirable features:

- The dynamic key hierarchy presents two flavors, instead of three, but can reach similar audience sizes than the Complete Key approach.
- The key distribution process requires very little computational effort, at the price of very little overhead. This is very convenient for small devices, such as smart phones or PDAs.
- It naturally suits multi-tree overlay topologies.

Secret sharing can be defined as follows:

> *To distribute a secret S in n shares, needing $t \leq n$ shares to reconstruct it, and being this impossible with a number of shares below t (threshold).*

The *Shamir Threshold Secret Sharing Scheme* is the most popular among the several existing secret sharing methods. A detailed description can be found in [19][21].

The present scheme uses Secret Sharing to refresh CW. Here, one of the shares is given a higher weight than the others, and CW cannot be reconstructed without it. We have named this "important share" MS (Master Secret) and the "regular shares", SS (Secondary Shares). MS stands for several hours, and CW is refreshed by renewing the different SSs. How MS is refreshed depends on the flavor. More details are given next.

5.1 One-Level Flavor

The One Level Share Based approach employs more information pieces than its Complete Key counterpart, but is still one-level in the sense that only a key is distributed.

The stream is encrypted with a key called CW, which is refreshed at a frequency within seconds or minutes. The distribution of CW is accomplished by making use of a secret sharing method as follows. For each refreshment of CW

the Key Server generates as many shares as desired and includes them (without encryption) in separate messages, which are injected into the peer-to-peer network. The shares are named SSs: *Secondary Shares*. We call the messages that contain them SS-messages. Peers (clients) ask the network for SS-messages and resend them as needed. By collecting a minimum number of SSs (this number is called *threshold*) a subkey can be generated by using secret sharing. This subkey receives the name of PS *Partial Share*. The Key Server may generate and inject as many SSs as desired. Note that SS-messages and hence SSs are not encrypted and can be known publicly.

On the other hand, every peer authenticates against the Key Server and receives via TLS a piece of information named MS (*Master Share*), which must remain secret to outsiders. CW can be computed by means of a one-way function (we name it F) that takes as input a combination of MS and PS. F is not defined in this paper, and can be designed as desired. It should be complex enough, though, to prevent brute force attacks against MS. Refreshment rate of MS is within hours. This reduces the number of requests the Key Server must attend, hence allowing audience size to rise, say to thousands of clients.

Privacy of CW is achieved by keeping secret MS and refreshing it. Security against brute force attacks against CW is achieved by refreshing SS and, therefore PS. The global security of the system depends on the robustness of the F function. If an attacker cannot guess MS a reasonable time before it is refreshed then this scheme can be considered secure enough. MS lifetime can be adjusted as necessary for this purpose. Risk and impact of directly compromising CW is low due to the high refreshment rate.

Figure 3 shows the composition and origin of every piece of information involved.

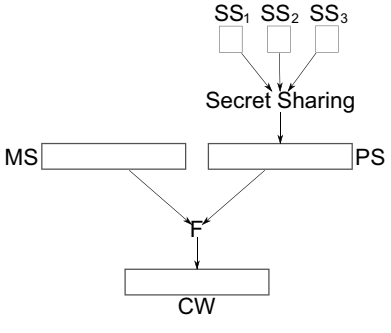

Fig. 3. Secret sharing for $t = 3$, channel i

5.2 Two-Level Flavor

This flavor allows the Share Based approach to reach a larger audience. CW encrypts the stream. It is distributed in the same manner as in the One-Level

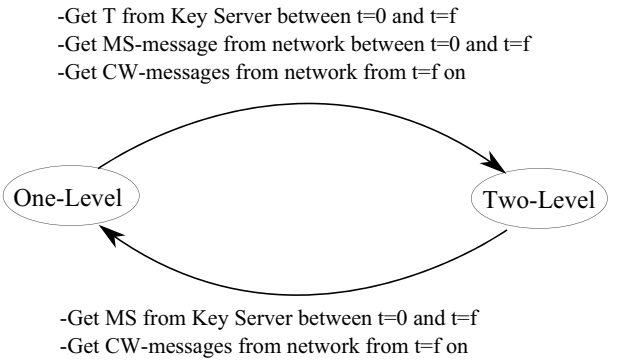

Fig. 4. Mode changes for the Share-Based Approach from clients' view. The process starts at *t=0* and ends at *t=f*.

flavor. The novelty consists of distributing MS throughout the peer-to-peer network: the Key Server encrypts MS with a new level key, named T (*authorization Token*), and the result is injected into the network inside a new type of message, namely MS-message. Peers distribute it as necessary. MS's lifetime is still of several hours. T is renewed under TLS against the Key Server. Its lifetime is in the order of days.

T's large lifetime allows to increase dramatically the period for which a client can decrypt the content without communicating with the Key Server. Hence, very large audiences can be reached by using this flavor. Security, on the other hand, decreases because of the same reason: an attacker would have several days or even weeks to try to compromise T, which would lead to being able to play the content. T's lifetime can be adjusted as necessary to make attackers' life harder (though losing audience size). Additionally, function F must be carefully chosen.

5.3 Changing among Flavors

Flavor change is performed when audiences exceeds (or goes under) a given threshold. The change is carried out in the same way than in Section 4.4: when informing about next change the necessary information is included in SS-message fields "Left" and "Next Level". Next we show how changes are done. Figure 4 depicts the details of flavor changing. We assume that the first notification occurs at moment $t = 0$ and change ends at $t = f$.

5.4 Security and Efficiency Considerations for the Share-Based Approach

Regarding efficiency, this scheme shows several advantages. First, the refreshment process requires very little computational resources: if Reed-Solomon codes are used then PS can be calculated by means of a Fast Fourier Transform, which

is computed efficiently at a very low computational cost [4][18]. Furthermore, the fact that SS-messages are not encrypted helps in that sense too.

If MS has not expired for a peer, gathering each SS only requires a communication: request and response. This speeds communications up. By appropriately adjusting the *threshold* (the minimum number of SS-messages needed to compute CW) and the number of successive updates of CW that can be calculated with t SS-messages (that is, the number of successive shares included in each SS-message), keys for a long playing time can be obtained without communicating with the Key Server.

Updates of MS are far apart in time and peers can ask for it with anticipation for subscription channels, making the requests rate follow a uniform distribution. This allows the Key Server to attend a bigger number of clients, favoring scalability of the net. Additionally, the update does not depend on arrivals or leavings of peers.

Another important advantage is the possibility for peers to be members of many peer-to-peer networks, each one for a different multimedia channel. Peers can even serve SS-messages and MS-messages for channels they are not subscribed to: possessing only the SSs for a channel does not allow to regenerate a valid CW, and MS-messages are encrypted with their corresponding T.

Regarding security, considerations are similar to those of section 4.5. However, there is one important difference: CW refreshment is carried out by means of a secret sharing technique and function F. Sharing techniques are secure enough to not to be worried about [19][21], but F should be chosen very carefully. If a hash function is involved, then its output should be long enough to withstand brute-force attacks for a period of time similar at least to SK's lifetime. Signature of every CW-message and MS-message may be also implemented to avoid denial-of-service attacks.

We believe that security considerations make inappropriate the addition of a third level key to the Share Based approach. Anyway, if desired, a new key could be added to encrypt and distribute T. Its lifetime could be in the order of a month. The DVB industry has battled with the set-top box key theft problem for years, still a reliable and durable solution has not been found. The more time an attacker has to study a key, the easier will be guessing and sharing it.

6 Comparison of Complete Key Distribution Scheme and Share-Based Key Distribution Scheme

Table 1 sums up both schemes, focusing on audience size and security against fraud. Each flavor is assigned a security level based on the risk of key compromise and its consequences. The best suited scenario is also assigned. Figures 5 and 6 compare ideal reachable audience size and security for both schemes.

The main conclusion that can be extracted is that each Share-Based flavor can supply similar audience sizes than the Complete Key approach, employing a shorter key hierarchy. On the other hand, the Complete Key method offers more security. Therefore, Complete Key should be used for small and medium audience

Table 1. Comparison of all flavors

Complete Key	
One-Level	Keys: CW (secs, mins) Dependence on Key Server: very high Audience size: very low ($\approx$ 0.1 K clients?) Security: high CW compromise risk: very low CW compromise impact: very low Scenario: small videoconferences
Two-Level	Keys: CW (secs, mins); SK (hours) Dependence on Key Server: medium Audience size: medium ($\approx$ 1 K clients?) Security: medium-high CW compromise risk: very low CW compromise impact: very low SK compromise risk: medium SK compromise impact: high Scenario: large videoconferences, event broadcasting
Three-Level	Keys: CW (secs, mins); SK (hours); T (days, weeks) Dependence on Key Server: very low Audience size: high ($\approx$ 10K clients?) Security: medium CW compromise risk: very low CW compromise impact: very low SK compromise risk: medium SK compromise impact: high T compromise risk: medium T compromise impact: very high Scenario: great interest event broadcasting, P2PTV
Share-Based	
One-Level	Keys: CW (secs, mins); $\{MS$ (hours), SS (secs, mins)$\}$ Dependence on Key Server: low Audience size: medium ($\approx$ 1 K clients?) Security: medium-high CW compromise risk: low CW compromise impact: low MS compromise risk: medium MS compromise impact: high Scenario: large videoconferences, event broadcasting
Two-Level	Keys: CW (secs, mins); $\{MS$ (hours), SS (secs, mins)$\}$; T (days, weeks) Dependence on Key Server: very low Audience size: high ($\approx$ 10 K clients?) Security: medium CW compromise risk: low CW compromise impact: low MS compromise risk: medium MS compromise impact: high T compromise risk: medium T compromise impact: very high Scenario: great interest event broadcasting, P2PTV

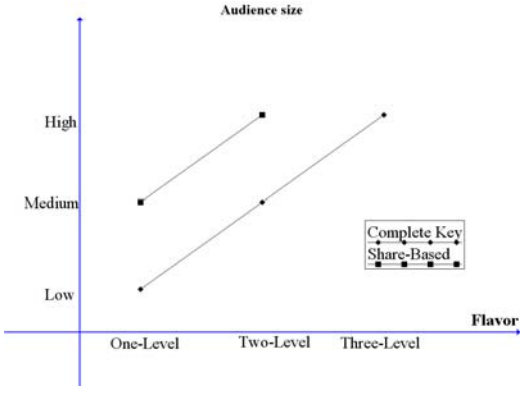

Fig. 5. Comparison of both schemes regarding audience size

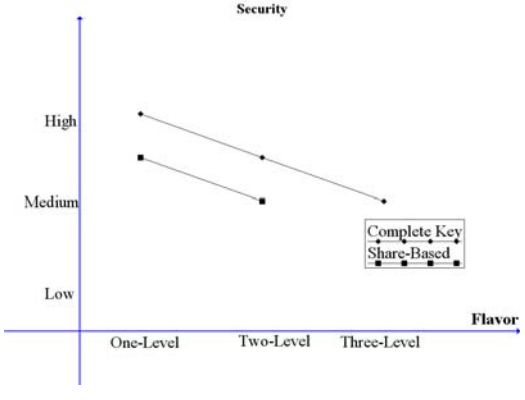

Fig. 6. Comparison of both schemes regarding security

scenarios and Share-Based for very large ones. We believe, also, that the Share-Based is more appropriate for set-top boxes, since these devices incorporate less computational capacities.

7 Conclusions and Future Work

This paper introduces two novel key distribution schemes for live multimedia peer-to-peer overlays, which share a common new feature: the possibility of dynamically changing the number of levels in the key hierarchy. This is motivated by the fact that the larger the key hierarchy is, the more audience can be supplied, but the more time the highest level key is exposed. Hence, the key hierarchy

employed should be shortened whenever possible. However, it can be enlarged if audience size requires it.

The first scheme introduced, the Complete Key Distribution Scheme, is simple and direct. It fits scenarios where a low exposure time is required for keys, preferably on top of mesh-like topologies. The second one, the Share-Based Key Distribution Scheme, is similar to the Complete Key scheme, but employs a Secret Sharing technique to refresh the lowest level key. It suits better multi-tree topologies and high audience scenarios, while needing little computational power. This last feature makes it more appropriate for set-top boxes. Additionally, the Share-Based approach outperforms Complete Key because it requires less hierarchy levels and computational cost. On both schemes, a smart key hierarchy level adjustement will adjust the higher level key exposure time to the minimum possible, depending on audience size.

Our plans for the near future are to implement and test both schemes in a peer-to-peer simulator and to study a more efficient top level key distribution to address the problem of receiving a large number of requests produced by flash crowd situations. Additionally, it is interesting to consider a fusion of both schemes (not covered in this paper): the Complete Key One-Level and Two-Level flavors might be used for small and medium-size audiences, while the Share-Based Two-Level flavor would be employed for larger audiences.

Acknowledgements

This work has been partially funded by grants from the Spanish Ministry of Science and Innovation (TIN2008-01117), Junta de Andalucía (P06-TIC-1426, P08-TIC-3518, FQM 0211 and TEC2006-12211-CO2-02), and the European Regional Development Fund (ERDF).

References

1. Aciicmez, O., Seifert, J.-P., Zhang, X.: A secure DVB set-top box via trusting computing technologies. In: Consumer Communications and Networking Conference (2009)
2. Balfe, S., Gallery, E., Mitchel, C., Paterson, K.: Challenges for trusted computing. In: Security & Privacy. IEEE, Los Alamitos (2008)
3. Balfe, S., Lakhani, A., Paterson, K.: Trusted computing: providing security for peer-to-peer networks. In: Fifth IEEE International Conference on Peer-to-Peer Computing (2005)
4. Blahut, R.E.: A universal Reed-Solomon decoder. IBM Journal of Research and Development 28(2), 150–158 (1984)
5. Eskicioglu, A.M.: Multimedia security in group communications: recent progress in key management, authentication, and watermarking. Multimedia Systems 9(3), 239–248 (2003)
6. Ganesh, A.J., Kermarrec, A.-M., Massoulié, L.: Peer-to-peer membership management for gossip-based protocols. IEEE Transactions on Computers 52(2) (2003)

7. Huang, Y.-L., Shieh, S., Ho, F.-S., Wang, J.-C.: Efficient key distribution schemes for secure media delivery in pay-tv systems. IEEE Transactions on Multimedia 6(5), 760–769 (2004)
8. International Teleommunication Union. ITU-R Rec. BT.810. Conditional-Access broadcasting systems (1992)
9. Jelasity, M., Montresor, A.: Epidemic-style proactive aggregation in large overlay networks. In: ICDCS 2004: Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS 2004), pp. 102–109. IEEE Computer Society, Los Alamitos (2004)
10. Kostoulas, D., Psaltoulis, D., Gupta, I., Birman, K., Demers, A.: Decentralized schemes for size estimation in large and dynamic groups. In: NCA 2005: Proceedings of the Fourth IEEE International Symposium on Network Computing and Applications, pp. 41–48. IEEE Computer Society, Los Alamitos (2005)
11. Le Merrer, E., Kermarrec, A.-M., Massoulie, L.: Peer to peer size estimation in large and dynamic networks: A comparative study, pp. 7–17 (2006)
12. Lee, W.: Key distribution and management for conditional access system on DBS. In: Proc. of international conference on cryptology and information security, pp. 82–86 (1996)
13. Liu, B., Zhang, W., Jiang, T.: A scalable key distribution scheme for conditional access system in digital pay-tv system. IEEE Transactions on Consumer Electronics 50(2) (2004)
14. Liu, X., Yin, H., Lin, C., Deng, Y.: Efficient key management and distribution for peer-to-peer live streaming system, pp. 638–641 (2007)
15. Liu, Y., Guo, Y., Liang, C.: A survey on peer-to-peer video streaming systems. Peer-to-Peer Networking and Applications 1(1) (2008)
16. Magharei, N., Rejaie, R., Guo, Y.: Mesh or multiple-tree: A comparative study of live P2P streaming approaches. In: INFOCOM 2007 (2007)
17. Massoulié, L., Le Merrer, E., Kermarrec, A.-M., Ganesh, A.: Peer counting and sampling in overlay networks: random walk methods. In: PODC 2006: Proceedings of the twenty-fifth annual ACM symposium on Principles of distributed computing, pp. 123–132. ACM, New York (2006)
18. McEliece, R.J., Sarwate, D.V.: On sharing secrets and Reed-Solomon codes. Communications of the ACM 24(9) (1981)
19. Menezes, A.J., Oorschot, P.C.V., Vanstone, S.A., Rivest, R.L.: Handbook of applied cryptography (1997)
20. Qiu, F., Lin, C., Yin, H.: EKM: An efficient key management scheme for large-scale peer-to-peer media streaming. In: Zhuang, Y.-t., Yang, S.-Q., Rui, Y., He, Q. (eds.) PCM 2006. LNCS, vol. 4261, pp. 395–404. Springer, Heidelberg (2006)
21. Schneier, B.: Practical Cryptography. Wiley & Sons, Chichester (2003)
22. Tu, F., Laih, C.S., Tung, H.H.: On key distribution management for conditional access system on pay-tv system. IEEE Transactions on Consumer Electronics 45, 151–158 (1999)
23. van der Veen, M., Lemma, A.N.: Electronic content delivery and forensic watermarking. Multimedia Systems 11(2), 174–184 (2005)