Access Control for Secure Distributed Data Structures in Distributed Hash Tables

Read Al-Aaridhi, Ahmed Yueksektepe and Kalman Graffi
Department of Computer Science, University of Düsseldorf, Germany
Email: alaaridhi@hhu.de, ahmet.yueksektepe@hhu.de, graffi@hhu.de

***Abstract*—Peer-To-Peer (P2P) networks open up great possibilities for intercommunication, collaborative and social projects like file sharing, communication protocols or social networks while offering advantages over the conventional Client-Server model of computing pattern. Such networks counter the problems of centralized servers such as that P2P networks can scale to millions without additional costs. In previous work, we presented Distributed Data Structure (DDS) which offers a middle-ware scheme for distributed applications. This scheme builds on top of DHT (Distributed Hash Table) based P2P overlays, and offers distributed data storage services as a middle-ware it still needs to address security issues. The main objective of this paper is to investigate possible ways to handle the security problem for DDS, and to develop a possibly reusable security architecture for access control for secure distributed data structures in P2P networks without depending on trusted third parties.**

I. Introduction

Peer-To-Peer (P2P) networks are networks consisting of multiple participating peers which are treated equally. Usually, P2P system does not have a central authority. Unlike the Client- Server Model, the functionality of applications on top of the P2P networks depends on peers to cooperate. While P2P networks eliminate the problems like central authority or censorship, they introduce new challenges concerning the handling of data in such networks. In recent years, the concepts of the P2P technologies are shifting towards sophisticated use cases for example in the field of multimedia streaming as well as distributed online social networks such as in LibreSocial [1], [2], [3] and Safebook [4] or for optimization procedures [5], [6]. In previous works [7], [8] we proposed a Distributed Data Structure (DDS) scheme on Distributed Hash Tables (DHT), which consists of linked data items which contain the following:

- a mandatory object ID under which this item is stored.
- optional pointers to further data items in the DDS graph
- and an optional payload, such as an image or comment.

A DDS supports many functions such as Store DDS, Retrieve DDS, Modify DDS - Modify Values / Add Entry and Modify DDS - Delete Entry. The DDS project offers a distributed storage system over DHTs. In a public P2P system based on a DHT, nodes cannot completely trust each other. Therefore, the assumption that malicious peers can exist is not optional but a necessary. Under these circumstances, the security of distributed data in P2P systems is an essential requirement and it needs to be analyzed extensively in all its aspects. Therefore, we developed in previous work a secure model for DDS in order to_secure_ data which is stored on various DHT nodes. In the secure model, the hybrid cryptography paradigm is to be used as a basis for the security architecture design, and we propose the Granular Encryption and Granular Signature (GEGS) strategy. With the help of this strategy, we secure a Distributed Data Structure as a collection of multiple Secured Data Items. The GEGS strategy describes an idea for maintaining the complex form of a DDS also in its Secured Data form. This strategy combines the Granular Encryption strategy (for the encryption of a DDS) and the Granular Signature strategy (for the signing of a secured DDS). As related work, Hu et al [9] Introduced new access control model, this model uses the voting scheme to aggregate decisions among controllers, each controller announces his privacy concern and his decision for the resources. The final decision made in this model after collecting and aggregating this information. With this context, the intention of this paper is to develop an access control for the DDS project, which ensures the security of the stored data through DDS. In this paper, the access control designed for the DDS scheme is being presented. For this purpose, different aspects of access control are discussed and initial assumptions concerning the security of the DDS project and its data environment are declared. The proposed access control architecture allows to securely access to distributed data objects, securing their confidentiality and integrity. We introduce an access control architecture for secure Distributed Data Structures based on strictly defined security assumptions in detail and we introduce an access control mechanism working completely without trusted nodes. The remainder of the paper is organized as follows. In Section II, we introduce the main assumptions for this work. In Section III, the access control architecture for the Distributed Data Structure is presented and highlight its strengths. Section IV concludes the paper.

II. Assumptions

In the following, the expected assumptions from the access control architecture of DDS are enlisted. With these assumptions, we determine how we understand the term secure access control and what features the architecture should offer.

A. Public Key Infrastructure

Within the scope of this work, we assume that there is an already established public key infrastructure, so users are

assumed to have the public keys of their friends and other known users.

B. Provable Ownership

For users to be able to claim any ownership over a Distributed Data in the DHT, we create the OverlayKeys for Distributed Data based on the public key of the Owner. The simple formula we use for generating OverlayKeys is given below:

OverlayKey= PublicKey(ObjectName)

C. Malicious Nodes

In the DHT environment, we assume that each participant is potentially a malicious node.

III. Distributed Data Structure with access control model

The access control model for DDS, is being explained in the following paper in detail. The access model in this work has been shaped based on the previously defined assumptions.

A. Security Context

We propose a security model for DDS and the hybrid cryptography has been preferred, we have used hybrid cryptography as it offers significant advantages over the plain asymmetric cryptography method. The continuous replication in the DHT over the lifetime of Distributed Data can result in significant network load and it is fundamental to reduce this load for a P2P system to be able to scale Distributed Data Structure. The idea of hybrid cryptography is very suitable to minimize to storage space needed for the meta-data of the security model. This concept has been adapted for the DDS project by encrypting Distributed Data with an encryption key deterministically generated by the creator of the Distributed Data, called **DDEKey (Distributed Data Encryption Key)**. The DDEKey is being saved asymmetrically encrypted for each privileged user. Yet, in hybrid cryptography a Secured Data can not be read without being decrypted first. A user getting a Secured Data from a DHT also needs a symmetric key to read the data. And in addition to the symmetric key, we also need additional security meta-data to mark the Secured Data readable for specific users. The **Security Context** module serves exactly to this purpose and is a sink for encrypted and signed data. DDS has a different structure. In Figure 1 the composition of the concrete **Simple Security Context** of the Distributed Data Tree is being illustrated. So the Security Context is actually the main construct embodying the security model this work defines. A Security Context is defined as the module of a Distributed Data responsible for the complete security. For each Distributed Data, a Security Context handles the security logic and access control while holding the secured Distributed Data in form of a Secured Data (or multiple granular Secured Data's). As the naming suggests, the Security Context contains the necessary context information to interpret encrypted data as the corresponding DDS. A concrete implementation of a Security Context shall

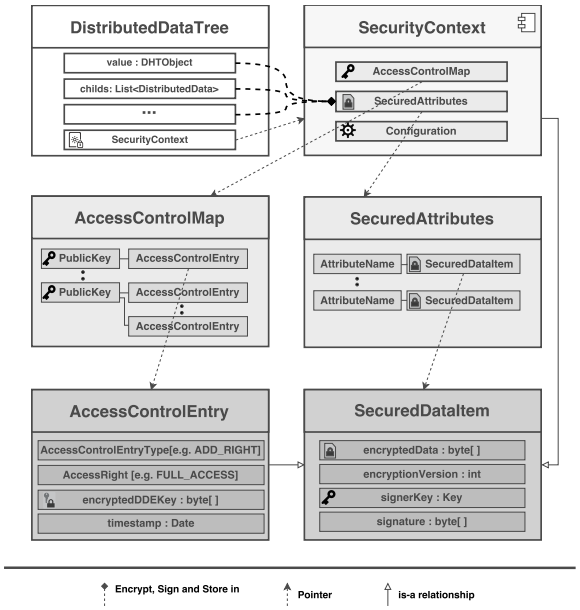


Fig. 1. DDS Security - Structure of the Security Context

enable users to attach desired security needs to a Distributed Data. So the way how a DDS object needs to be secured can be chosen freely, with the requirement of implementing the security interfaces correctly for a new Security Context. Allowing different Security Context implementations allows context sensitive security use cases. A Security Context offers basic functions for validating existing access control map entries, generating new entries for granting access right, revoking access rights, generating new DDS-Key's and generally for encrypting and decrypting the DDS data. This work defines the **Simple Security Context** class as a basis Security Context. This class implements the simple security logic of securing all attributes of a Distributed Data Structure and it is generically usable for all Distributed Data Structure's in the DDS project as a basis for their specific Security Context. As it can been seen in Figure 1, a Security Context is consisting of the three sub-modules: **Access Control Map**, **Secured Data Items** and **Configuration**. The implementation of these operations by a Simple Security Context defines at the same time the core security protocol.

1) Access Control Map: The Security Context module realizes the hybrid cryptography concept with the help of the Access Control Map module. An Access Control Map is the collection of access control information. It is implementing an **Access Control List (ACL) Pattern** [10]. So an Access Control Map contains per a privileged user's public key, a list of **Access Control Entries**. An Access Control Entry is a data record containing multiple information. It indicates which **Access Right** has been granted to the corresponding user concerning the Distributed Data the Security Context belongs to. An Access Right is a publicly known information which is being used to decide if and how a user has the right to access a Distributed Data (or a DDS Operation on a Distributed Data to be more concrete). While declaring an Access Right,

the Access Control Entry also contains a Distributed Data Encryption Key (DDEKey), in asymmetrically encrypted form, only readable for the specific user. An Access Control Entry is at the same time a Signed Data. Therefore, its authenticity can be validated and with the help of Access Control Entry's everyone is publicly able to see which access right a user has on a Distributed Data. The first Access Control Entry in an Access Control Map is the entry created by the creator of the Distributed Data, to which the Security Context and the Access Control Map belongs to. With this first entry, the owner gives himself the Access Right to have full control over the Distributed Data. At the same time, with the existence of this entry, it is proven towards to all DHT nodes that the owner created this Distributed Data with his own intention. A chain of Access Control Entry's (**access chain**) leads to the existence of each Access Control Entry.

2) Secured Data Items: The Secured Data Items module of the Security Context is a map containing DDS Attribute-Name and Secured Data Item (which is a concrete Secured Data) pairs. This map is being used to store the secured form of a DDS in the securing procedure of a DDS. Based on their names, DDS Attribute's of the DDS are secured and stored in the Secured Data Items of the Security Context.

3) Configuration of a Security Context: The configuration module of a Security Context contains configuration parameters concerning the usage of the Security Context. The most important parameter in the Configuration module is the Encryption Version parameter which determines which DDEKey must be used for new modifications to the DDS.

B. Access Revocation

Through hybrid cryptography we encrypt a DDS with a symmetric key named DDEKey. Further, we share the DDEKey through asymmetric encryption with other users. The advantage of hybrid cryptography here is that we only need to encrypt the actual data once with the symmetric encryption, which is more suitable for efficiently encrypt larger data blocks, and we store only one copy of the encrypted data. For each user we want to share the encrypted data with, we simply share the DDEKey. A critical issue occurs with this proceeding is that once we want to revoke the reading access right of a user, the user potentially still has the DDEKey and can read the data without authorization. Once a DDEKey is shared with a user, this user save the given DDEKey, and even if the Access Control Entry is removed from the Access Control Map of the DDS, the user is still able to read the encrypted DDS. The strategy we use to solve this problem is to simply ignore the older versions of the data which the user could have read anyways. This strategy is also called **Lazy Revocation** [11]. With the Lazy Revocation strategy we only change the DDEKey and donnot need to re-encrypt any data.

C. Protecting Identities (Anonymous identities)

To overcome the requirement to protect the identity of privileged users has been especially challenging as it is not trivial to publicly give special access rights to public identities with keeping at the same time the identity of the privileged user hidden. Because of two simple reasons, the identity of privileged users need to be known: replication nodes need to check actively the identity of the requesting users and with the public keys of the users, new encryption keys are encrypted and stored in case of an encryption key change. So the answer we found to this requirement was to leave the privileged identity public indeed. With an **anonymous identity** we are able to hide the actual identity of the user. We define at this point an anonymous identity as a public-private key pair specially created for having anonymous access to a certain Distributed Data.

IV. Conclusions

P2P networks currently are shifting to use in more sophisticated applications, they have been used to not only in the form of file sharing applications but also in further application domains, such as for streaming of for P2P-based online social networks. In previous work, we presented a Distributed Data Structure on top of a Distributed Hash Table overlay that offers an object-oriented, distributed storage layer. We developed in previous work a security architecture for the DDS project, which ensures the security of the stored data through DDS. With this context, in this paper, we introduce an access control architecture for the secured DDS working in DHTs without depending on trusted third parties, many access problems have been carefully analyzed such as the revocation problem of shared access to secured distributed system.

References

[1] K. Graffi, S. Podrajanski, P. Mukherjee, A. Kovacevic, and R. Steinmetz, "A Distributed Platform for Multimedia Communities," in *Proc. of IEEE Int. Symposium on Multimedia (ISM)*, 2008.

[2] K. Graffi, C. Gross, P. Mukherjee, A. Kovacevic, and R. Steinmetz, "LifeSocial.KOM: A P2P-Based Platform for Secure Online Social Networks," in *Proc. of IEEE Int. Conf. on Peer-to-Peer Computing (P2P)*, 2010.

[3] K. Graffi, C. Gross, D. Stingl, D. Hartung, A. Kovacevic, and R. Steinmetz, "LifeSocial.KOM: A Secure and P2P-based Solution for Online Social Networks," in *Proc. of IEEE Int. Consumer Communications and Networking Conf. (CCNC)*, 2011.

[4] L. A. Cutillo, R. Molva, and M. Önen, "Safebook: A Distributed Privacy Preserving Online Social Network," in *Proc. of Int. Symp. on World of Wireless, Mobile and Multimedia Networks (WOWMOM)*, 2011.

[5] K. Graffi, K. Pussep, S. Kaune, A. Kovacevic, N. Liebau, and R. Steinmetz, "Overlay Bandwidth Management: Scheduling and Active Queue Management of Overlay Flows," in *IEEE LCN '07: Proceedings of the International Conference on Local Computer Networks*, 2007.

[6] K. Graffi, "Monitoring and management of peer-to-peer systems," Ph.D. dissertation, 2010.

[7] R. Al-Aaridhi and K. Graffi, "Sets, Lists and Trees: Distributed Data Structures on Distributed Hash Tables," in *IEEE IPCCC: Proceedings of the 35th IEEE International Performance Computing and Communications Conference IPCCC*, 2016.

[8] R. Al-Aaridhi, A. Yueksektepe, T. Amft, and K. Graffi, "Distributed Data Structures Improvement for Collective Retrieval Time," in *IEEE WPMC: Proceedings of the 19th International Symposium on Wireless Personal Multimedia Communications*, 2016.

[9] H. Hu and G.-J. Ahn, "Multiparty authorization framework for data sharing in online social networks," in *IFIP Annual Conference on Data and Applications Security and Privacy*. Springer, 2011, pp. 29–43.

[10] N. RFC4949, "Internet Security Glossary, Version 2," in *Recuperado el 24 de febrero de 2010*, 2007.

[11] M. Backes, C. Cachin, and A. Oprea, "Secure key-updating for lazy revocation," in *Pearson Deutschland GmbH*, 2006.