Efficient Methods for Implementation of Generalized Access Structures

James Smith
james.o.smith732@gmail.com

Abstract

The recent advent of cloud computing and IoT has made it imperative to store huge amount of data in the cloud servers. Enormous amount of data is also stored in the servers of big organizations. In organizations, it is not desirable for every member to have equal privileges to access the stored data. Threshold secret sharing schemes are widely used for implementing such access control mechanisms. The access privileges of the members also vary from one data packet to another. While implementing such dynamic access structures using threshold secret sharing schemes, the key management becomes too complex to be implemented in real time. Furthermore, the storage of the users' access privileges requires exponential space ($O(2^n)$) and its implementation requires exponential time. In this paper, the algorithms proposed to tackle the problems of priority based access control and authentication require a space complexity of $O(n^2)$ and a time complexity of $O(n)$, where n is the number of users. In the practical scenario, such space can easily be provided on the servers and the algorithms can run smoothly in real time.

Keywords: Generlized Access Control, Access Structures, Generalized Secret Sharing

1 Introduction

The threshold secret sharing schemes proposed in [1] and [2] are now being widely used. For the past ten years it is also being used to combat side channel attacks and authentication challenges alongside hardware primitives [3, 4]. Unfortunately, with the advent of cloud computing, IoT and big data many desired secret sharing schemes cannot be implemented by threshold implementations. Many works on generalized secret sharing schemes exist in the literature but most of these are infeasible in the practical scenario due to the enormous number of shares required to be generated for these schemes. As proposed in [5], monotone access structures can be implemented but in the worst case, each of the n users has to hold on the order of 2^n shares.

For a given secret sharing scheme or access control scheme, the access structure γ is defined as the sets of members allowed to gain access to the secret. γ

is said to be *monotone* if

$$\alpha \in \gamma \ , \alpha \subseteq \beta \Longrightarrow \beta \in \gamma$$

[6, 7, 8] proves that there exists no threshold secret sharing schemes which for arbitrary monotone functions. It has been proved that a fully generalized secret sharing scheme for any arbitrary γ does not exist without 2^n space and time complexity. In [9, 10, 11, 12], multi-level and compartmented secret sharing schemes are implemented which are more general than threshold secret sharing schemes. Secret sharing schemes have also been generated from lattices [13, 14, 15]. Each of these schemes are suitable for specific applications but none of them is fully generalized such that any arbitrary γ can be implemented.

In this paper, we propose two algorithms. The first one successfully implements access control for arbitrary monotone access structures while the second one implements access control for any arbitrary access structure. The second algorithm consumes more space than the first but both of them consume storage space in the order of n^2 in the worst case, where n is the number of participants. The time taken by both the algorithms is in the order of $poly(n)$ in the worst case. The difference between access control schemes and secret sharing schemes is that the access control scheme is a decision making algorithm which grants access only to the sets present in γ. Whereas in a secret sharing scheme, the secret can be directly computed from the shares of the qualified sets of participants. Since it is proven that practical secret sharing schemes for arbitrary γ does not exist [16, 17, 18], we can implement the same functionality using access control algorithms. The secret can be safely stored in the main server and it will only be revealed to a qualified subset of participants. To check if the subset of participants belongs to γ, we can use the access control algorithms.

For example, there are seven participants namely A, B, C, D, E, F, G. The qualified sets to gain access are (A, B, C), (B, C, D), (D, E), (D, F) and (F, G). Therefore,

$$\gamma = \{(A, B, C), (B, C, D), (D, E), (D, F), (F, G)\}$$

Only these sets are allowed to gain access. All other possible sets are denied access. If γ is a monotone access structure, then all the supersets of the qualifiable sets present in γ will also be able to gain access.

2 Generalized Secret Sharing Schemes

In this section, we define the basic threshold secret sharing scheme, discuss the methods to implement generalized secret sharing using threshold implementation and explain the challenges faced in doing so.

2.1 Threshold Secret Sharing Scheme

The (k, n) threshold secret sharing scheme states that the secret S is divided into n shares and each of the n participants receive one share each.

- The secret S can be reconstructed if at least k participants come together and use their shares.
- The secret S cannot be reconstructed if less than k participants come together and combine their shares.

Many threshold schemes have been stated in [Sham], [Blakeley], [Asmuth-Bloom], [Kothari]. A simple model is discussed over here for the purpose of introduction. A polynomial

$$y(x) = a_{k-1}x^{k-1} + a_{k-2}x^{k-2} + a_{k-3}x^{k-3} + ... + a_2x^2 + a_1x + S,\ a_i\ \epsilon\ \mathbb{R}$$

is constructed where S is the secret. The shares of the secret given to each user are the points (x, y) which lie on the aforementioned polynomial curve in the real plane. To reconstruct the secret from the shares, the coefficients of the polynomial have to be calculated. At least k such shares are required to do this. Hence, this is an efficient implementation of the (k, n) threshold secret sharing scheme. Algorithms of polynomial evaluation and interpolation mentioned in [Sham cit1] and [Sham cit2] show that this computation can be done in $O(nlog^2n)$ complexity.

2.2 Generalized Secret Sharing using Threshold Implementation

It has been shown in [Benaloh] that generalized secret sharing cannot be implemented using threshold secret sharing schemes. Here, we mention a couple of algorithms which use threshold implementations to implement generalized secret sharing schemes. This is followed by the problems which may be encountered with threshold implementations if the number of participants increases.

2.2.1 Proof of security

If a qualified set of participants contribute their correct keys, they can easily reconstruct the polynomial and obtain **S**. Otherwise, the incorrect shares combine to give the equation of an incorrect curve, thus revealing an error value as the secret. This is an extrapolation of the (t, n) threshold secret sharing scheme proposed by Shamir.

2.2.2 Challenges

- In case of large access structures, there is a high probability that a single participant is involved in many qualified sets. This implies that he will hold many shares. Even if the participants belonging to a qualified set come together but one of the participants contribute an incorrect share, they will be denied access. Hence, share management becomes inconvenient and very difficult in the practical scenario.
- This algorithm is inefficient in the worst case scenario. Considering that there are minimum two members in a qualified set,

Algorithm 1 Using a unique polynomial for every set in Access Structure

1. Let there be m qualified sets of participants in the access structure and let k_i denote the number of participants in the i^{th} set of the access structure.
2. Every qualified set is assigned a unique polynomial in the following way:

$$y_1(x) = a_{1,k_1-1}x^{k_1-1} + a_{1,k_1-2}x^{k_1-2} + a_{1,k_1-3}x^{k_1-3} + ... + a_{1,2}x^2 + a_{1,1}x + \mathbf{S}$$
$$y_2(x) = a_{2,k_2-1}x^{k_2-1} + a_{2,k_2-2}x^{k_2-2} + a_{2,k_2-3}x^{k_2-3} + ... + a_{2,2}x^2 + a_{2,1}x + \mathbf{S}$$
$$.....$$
$$.....$$
$$y_m(x) =$$
$$a_{m,k_m-1}x^{k_m-1} + a_{m,k_m-2}x^{k_m-2} + a_{m,k_m-3}x^{k_m-3} + ... + a_{m,2}x^2 + a_{m,1}x + \mathbf{S}$$

3. The polynomial of every qualified set represents a curve in the two dimensional space. Every participant is given multiple pairs of (y, x), where each pair represents a point lying on each curve in which the paticipant is a legal member. *Note: It should be taken care that the points of intersections of the curves should not be distributed as shares.*
4. When a set of participants come together, they contribute the share corresponding to that group and reconstruct the polynomial to obtain **S**.

Algorithm 2 Using a single big polynomial

1. Let there be m qualified sets of participants in the access structure and let u_i denote the number of qualified sets in which the i^{th} user is a legitimate member.
2. Each qualified set of the access structure is assigned a characteristic prime number. Let the prime for set i be denoted by p_i.
3. The share given to user i is denoted by

$$s_i = p_1 \times p_2 \times ... \times p_{u_i}$$

$$n(\gamma) = \binom{n}{2} + \binom{n}{3} + ... + \binom{n}{n}$$
$$n(\gamma) = 2^n - n - 1$$

This shows that the space required in the worst case is of the order 2^n, which is highly undesirable.

3 Conclusion

We propose two algorithms. The first one successfully implements access control for arbitrary monotone access structures while the second one implements access control for any arbitrary access structure. The second algorithm consumes more space than the first but both of them consume storage space in the order of n^2 in the worst case, where n is the number of participants. The time taken by both the algorithms is in the order of $poly(n)$ in the worst case. The difference

between access control schemes and secret sharing schemes is that the access control scheme is a decision making algorithm which grants access only to the sets present in γ. Whereas in a secret sharing scheme, the secret can be directly computed from the shares of the qualified sets of participants. Since it is proven that practical secret sharing schemes for arbitrary γ does not exist, we can implement the same functionality using access control algorithms. The secret can be safely stored in the main server and it will only be revealed to a qualified subset of participants. To check if the subset of participants belongs to γ, we can use the access control algorithms.

References

[1] L.-J. Pang and Y.-M. Wang, "A new (t, n) multi-secret sharing scheme based on shamirs secret sharing," *Applied Mathematics and Computation*, vol. 167, no. 2, pp. 840–848, 2005.

[2] I. N. Bozkurt, K. Kaya, A. A. Selcuk, and A. M. Güloglu, "Threshold cryptography based on blakley secret sharing," *Information Sciences*, pp. 1–4, 2008.

[3] V. Sehwag and T. Saha, "Tv-puf: a fast lightweight analog physical unclonable function," in *2016 IEEE International Symposium on Nanoelectronic and Information Systems (iNIS)*. IEEE, 2016, pp. 182–186.

[4] T. Saha and V. Sehwag, "Tv-puf: A fast lightweight aging-resistant threshold voltage puf," *Cryptology ePrint Archive*, 2016.

[5] G. Dirr, H. Ito, A. Rantzer, and B. S. Rüffer, "Separable lyapunov functions for monotone systems: Constructions and limitations," *Discrete & Continuous Dynamical Systems-B*, vol. 20, no. 8, p. 2497, 2015.

[6] J. Benaloh, "Dense probabilistic encryption," in *Proceedings of the workshop on selected areas of cryptography*, 1994, pp. 120–128.

[7] M. Abadi, "Logic in access control," in *18th Annual IEEE Symposium of Logic in Computer Science, 2003. Proceedings.* IEEE, 2003, pp. 228–233.

[8] W. Tolone, G.-J. Ahn, T. Pai, and S.-P. Hong, "Access control in collaborative systems," *ACM Computing Surveys (CSUR)*, vol. 37, no. 1, pp. 29–41, 2005.

[9] T. Saha, N. Aaraj, N. Ajjarapu, and N. K. Jha, "Sharks: Smart hacking approaches for risk scanning in internet-of-things and cyber-physical systems based on machine learning," *IEEE Transactions on Emerging Topics in Computing*, 2021.

[10] T. Saha *et al.*, "Machine learning-based efficient and generalizable cybersecurity frameworks," 2022.

[11] T. Saha, N. Aaraj, and N. K. Jha, "Machine learning assisted security analysis of 5g-network-connected systems," *IEEE Transactions on Emerging Topics in Computing*, 2022.

[12] J. Brown, T. Saha, and N. K. Jha, "Gravitas: Graphical reticulated attack vectors for internet-of-things aggregate security," *IEEE Transactions on Emerging Topics in Computing*, 2021.

[13] T. Saha, N. Aaraj, and N. K. Jha, "System and method for security in internet-of-things and cyber-physical systems based on machine learning," Jun. 23 2022, uS Patent App. 17/603,453.

[14] S. Oberman, L. Sousa, B. Pasca, A. Nannarelli, D. Mallasen, R. Murillo, A. Barrio, G. Botella, L. Pinuel, M. Prieto-Matias *et al.*, "Emerging and impacting trends on computer arithmetic."

[15] R. S. Sandhu and P. Samarati, "Access control: principle and practice," *IEEE communications magazine*, vol. 32, no. 9, pp. 40–48, 1994.

[16] A. A. E. Kalam, R. E. Baida, P. Balbiani, S. Benferhat, F. Cuppens, Y. Deswarte, A. Miege, C. Saurel, and G. Trouessin, "Organization based access control," in *Proceedings POLICY 2003. IEEE 4th International Workshop on Policies for Distributed Systems and Networks.* IEEE, 2003, pp. 120–131.

[17] V. C. Hu, D. Ferraiolo, D. R. Kuhn *et al.*, *Assessment of access control systems.* US Department of Commerce, National Institute of Standards and Technology , 2006.

[18] P. Samarati and S. C. d. Vimercati, "Access control: Policies, models, and mechanisms," in *International School on Foundations of Security Analysis and Design.* Springer, 2000, pp. 137–196.