Efficient Content Authentication in Publish/Subscribe Systems

Walter Wong , Maurício F. Magalhães (Advisor)

Department of Computer Engineering and Industrial Automation (DCA)
Faculty of Electrical and Computer Engineering (FEEC)
University of Campinas (UNICAMP)
P.O.Box 6101, 13083-970 – Campinas, SP, Brazil

{wong,mauricio}@dca.fee.unicamp.br

Abstract – The publish/subscribe communication paradigm is an asynchronous communication model aimed at content retrieval regardless of its location. However, the dissociation between contents and their providers open security issues related to their authentication and integrity. This paper presents a security model based on Merkle trees to provide efficient content authentication and data integrity with their original providers with low processing costs. The Merkle tree is a signature amortization technique based on a binary tree of cryptographic hashes, presenting three main advantages for its adoption: low processing costs compared to public key cryptography, multicast communication support and independent data blocks authentication. As a proof of concept, a prototype using Merkle trees was implemented and evaluated, showing 85 and 3.5 faster signature and verification times compared to the RSA public key cryptography.

Keywords – Security, Merkle tree, publish/subscribe paradigm, next-generation Internet architectures.

1. Introduction

The publish/subscribe communication paradigm [1] has recently received increasing attention due to its time and space decoupling nature, focusing the content retrieval regardless of its location or time of subscription. In publish/subscribe systems, publishers advertise content availability in a publish/subscribe overlay and hope subscribers subscribe to that content. This communication model is being increasingly used in Internet-wide services (e.g., RSS, presence, messaging, news, podcasts, social communities) due to its appealing characteristics.

Although the publish/subscribe paradigm privileges the content retrieval and provides inherent security properties if correctly implemented (e.g., DoS mitigation, information-centrism), the location decoupling nature of the paradigm arises some security issues regarding content authenticity and integrity due to the lack of interaction between publishers and subscribers or a binding between the content and its provider. While in the send/receive paradigm the receiver firstly authenticates the sender before fetching the content, there is no information about the server where the content is currently retrieved. As result, security threats such as fake and unauthorized content publication or content data blocks corruption may arise.

In this paper we present a security model to provide content authentication and integrity in the context of publish/subscribe systems using Merkle trees [2]. The Merkle tree is a binary tree composed of cryptographic hash values, where the leaves contain cryptographic hash values of data blocks; the internal nodes contain the hash of the concatenation of the children values and the top node contains the content public key. The benefits of this approach are threefold: standalone data blocks authentication, allowing the immediate disposal of data blocks and parallel subscription to all data blocks of the content; second, the proposed model natively supports multicast in the publish/subscribe system, since the authentication information is not previously shared between a party, and third, the current approach privileges low processing devices by trading storage (list of hashes) for speed (hash functions are much faster that public key cryptography). The Merkle tree implementation shows the performance of the Merkle tree is 85 and 3,5 times faster than RSA signature and verification times respectively. The cost of the tree is a higher authentication data in each data block, but this can be amortized with composite Merkle trees.

Another security model proposed in the context of the publish/subscribe systems is Packet Level Authentication (PLA) [3]. PLA introduces the concept of per packet authentication, resembling a banknote, where each packet has security mechanisms embedded on it. Hence, anyone receiving the packet can immediately authenticate it without any third-party's help. Each packet in PLA has a digital certificate of a trusted third party endorsing the packet's provider and also a digital signature over the packet to ensure its authenticity and integrity. However, the requirement of a digital signature per packet is a constraint for low processing devices.

The organization of this paper is as follows. Section 2 introduces the Merkle tree information. Section 3 presents the security model for the publish/subscribe system. Section 4 describes the implementation and evaluation of the security model. Finally, Section 5 summarizes this paper and presents the future work.

2. Background

The Merkle Tree [2] represents an optimization of the Lamport one-time signature scheme inheriting the benefits of Lamport scheme (security also relies on the difficulty of inverting one-way functions) and presents a compact representation of public keys as a single hash. Message issuers generate a collection of private keys and generate a hash tree of these keys. The top hash, also known as *Root Hash* is the public key, as shown in Fig. 1(a) and the *Authentication paths* are the private keys.

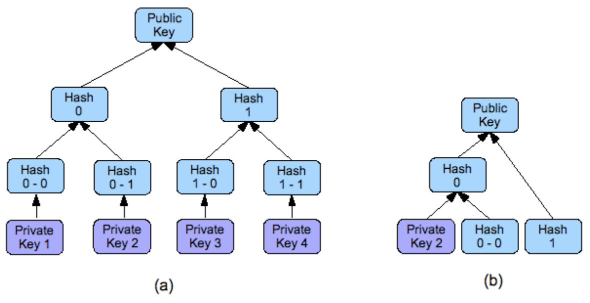

Figure 1. (a) Merkle Tree. (b) The *authentication path* for the private key 2.

The leafs of the hash tree contain the hash values of private keys and the intermediate nodes contain the hash of the concatenation of their two children (considering a binary tree). Fig. 1(a) shows the leaf 0-0 containing the hash of private key 1 (hash(privkey1)) and its parent node containing the hash of the concatenated values of its two leafs (0-0 and 0-1, hash (leaf 0-0 || leaf 0-1)), where || represents the concatenation of strings. The *Root Hash* is the public key associated to the group of private keys 1 to 4 and it is published in a public directory.

For an issuer who wishes to send a signed message, he first chooses one of the one-way private keys and signs the message. Then he computes the *authentication path* of the corresponding private key to allow the signature verification with the published public key in the receiver side. The *authentication path* is the list of hash values needed to reach the top hash, or the public key. Considering the example in Fig. 1(b), the *authentication path* for the private key 2 includes the leaf with hash value hash 0-0 and the node with hash 1.

3. Security model

In order to provide content authentication and data integrity in the context of publish/subscribe systems, we adopt the content *metadata* proposed in [4]. The metadata contains all relevant information regarding the content, including the provider identification, data blocks list and authentication information, as shown in Fig. 2.

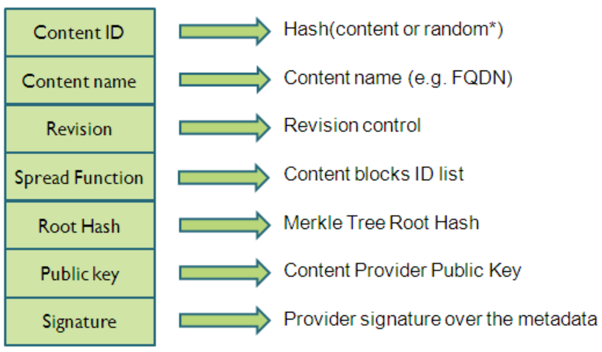

Figure 2. Metadata description.

The *Content ID* field contains a randomly generated 256-bit identifier used to identify the metadata in the security plane. The *Content name* field contains the high-level content description. The *Revision* field is used to control the current content version for update purposes.

The *Spread Function* field contains a mathematical function that lists the sequence of data blocks identifiers forming the content, similar to the one presented in [5]. This approach avoids the insertion of all data blocks IDs in the metadata, leading to prohibitive metadata size costs, or the inclusion of a range of IDs, which may result in collisions in the ID space.

The *Root Hash* field contains the root hash of the Merkle tree used to authenticate the content. For each data block, the content provider attaches the authentication path, allowing the subscriber to efficiently check the integrity of data blocks by applying the chained hashes. Since the entire metadata is signed, the root hash contains the original signature of the content, protecting against any modifications. Therefore, the metadata includes a *Public key* field containing the public key of the content provider and a *Signature* field containing the provider's signature over the metadata to protect against unauthorized modifications of the content descriptor.

The metadata is stored in a security plane where content providers must authenticate themselves before any metadata publication. Thus, subscribers can securely subscribe to the content metadata and verify its authenticity, and later subscribe to all data blocks of the content from the closest mirror available in the Internet.

4. Implementation and Evaluation

The security plane for publish/subscribe systems was implemented over the Pastry Distributed Hash Table (DHT) [6] using Java language and the Merkle tree was implemented in C language due to performance issues. By the time of this article, the prototype was not fully implemented with all proposed features. Hence, this paper focuses the evaluation of the Merkle tree considering the signature and verification times for different content and data block sizes and later comparing with the RSA public key cryptography.

The first experiment evaluated the required time to generate the public and private keys of a Merkle tree for different content sizes. The second experiment evaluated the overhead of the authentication path for different data block sizes for a same content. The third experiment evaluated the performance of the RSA digital signature and verification times and compared to the Merkle tree times.

Tab.1 shows the Merkle tree signature and verification times for different content sizes. Files with different sizes were fragmented in blocks of 128 bytes in order to generate different sizes of Merkle trees and authentication paths. Each test was evaluated 100 times and the average value and standard deviation collected in a Macbook 2.4GHz T8300 Core Duo 2G RAM.

The 68KB and 118KB files shares a special case due to same tree size although the file sizes are different. This case is due to the balanced binary tree requirement of Merkle trees, resulting in trees that may have many leaves with zero values. For this special case, the smallest tree which can hold all hash values for both files has 1024 leaves. The Merkle tree with 512 leaves holds up to 65KB (*data block size * number of leaves = 128 * 512*). All non-used leaves are filled with zeros, which is a sub-optimal condition. One solution for this problem is the composite Merkle tree, which will be part of the future works.

The second experiment evaluated the authentication path overhead for different data block sizes for a same file. For this experiment, we choose a 3.8 MB file and divided it in blocks of 128, 256, 512, 1024, 1200, 2048, 4096 and 8192 bytes and compared the authentication path overhead. The overhead is calculated considering the size of a 120-bit SHA-1 cryptographic hash. Tab. 2 shows the authentication path overhead for different data block sizes. We conclude that the current Merkle tree approach is sub-optimal for large file transference over the network. Considering a 1200-byte data block size, the overhead due to the authentication path is 20% of the total packet size (1200 bytes of data and 240 bytes of authentication path). This limitation will be addressed as future work with composite Merkle trees.

Table 2. Merkle tree authentication path overhead

File (bytes)	# of blocks	# of hashes	Auth. (bytes)	Overhead (%)
128	32.768	15	300	234,37
256	16.384	14	280	109,37
512	8.192	13	260	50,78
1024	4.096	12	240	23,42
1200	4.096	12	240	20
2048	2.048	11	220	7,42
4096	1.024	10	200	4,88
8192	512	9	180	2,22

The third experiment evaluated the same scenario of the first experiment, but using RSA public key cryptography. Tab. 3 shows the signature and verification times using a 1024-bit RSA asymmetric key. Each file was fragmented in blocks of 128 bytes and digitally signed and verified using RSA. For each file, 100 samples were collected and the average value and standard deviation computed in

Table 1. Merkle tree signature generation and verification times

Content size (KB)	Number of leaves	Auth. path length	Signature time (ms)	Standard dev. (ms)	Verification time (ms)	Standard dev. (ms)
10	128	7	1,51	0,15	2,39	0,39
18	256	8	3,33	1,31	4,72	1,53
38	512	9	7,25	1,38	10,54	2,81
68	1024	10	15,95	2,45	23,62	4,80
118	1024	10	17,85	2,35	21,62	4,20
182	2048	11	38,23	4,36	48,50	7,66

the Macbook.

Table 3. 1024-bit RSA digital signature and verification times

File size (KB)	# of blocks	Sign. (ms)	Stdev (ms)	Ver. (ms)	Stdev (ms)
10	83	258	65	11	4
18	145	394	86	18	6
38	311	768	205	37	12
68	550	1009	381	59	18
118	949	2007	835	114	36
182	1460	3247	2554	172	104

Comparing the signature and verification times obtained with Merkle tree (Tab. 1) and RSA public key cryptography (Tab. 3), we conclude through experiments that the first approach is as 85 and 3,5 times faster for signature and verification respectively. The Merkle trees have a higher overhead (180 to 300 bytes in the experiment) compared to the RSA scheme (128 bytes) and its efficiency relies on the trade of space for speed.

5. Conclusion and Future Work

This paper presented a security model for the publish/subscribe system to provide content authentication and data integrity. The model is based on a content metadata to securely describe the content, including the provider's identification, the data blocks list and the authentication information based on Merkle tree. The Merkle tree presents three advantages for its adoption: low processing costs compared to public key cryptography, multicast support and independent and out-of-order verification and subscription of data blocks. The experiments show that Merkle tree is an efficient technique to amortize the digital signatures in the security context, presenting a factor of 85 and 3,5 faster signature and verification times compared to the RSA public key cryptography, although it increases the amount of required authentication data.

As future work, we will work on composite Merkle trees to alleviate the authentication path overhead and algorithms to generate balanced trees with small number of leaves with zeros.

References

[1] M. Sarela, T. Rinta-aho, and S. Tarkoma, "RTFM: Publish/Subscribe Internetworking Architecture," *ICT Mobile Summit, Stockholm*, June 2008.

[2] R. C. Merkle, "A Certified Digital Signature," *Advances in Cryptology - CRYPTO'89 Proceedings*, pp. 218–238, 1989.

[3] C. Candolin, J. Lundberg and H. Kari, "Packet Level Authentication in Military Networks.," *Proceedings of the 6th Australian Information Warfare & IT Security Conference, Geelong, Australia*, November 2005.

[4] W. Wong, F. Verdi, and M. F. Magalhaes, "A security plane for publish/subscribe based content oriented networks," *4th ACM International Conference on emerging Networking EXperiments and Technologies (ACM CoNEXT), Student Workshop, Madrid, Spain*, December 2008.

[5] Publish/Subscribe Internet Routing Paradigm, "Conceptual architecture of psirp including subcomponent descriptions. Deliverable d2.2, PSIRP project," , August 2008.

[6] A. Rowstron and P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems," *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), Heidelberg, Germany*, pp. 329–350, November, 2001.