ISSN: 2321-7782 (Online)
Volume 2, Issue 7, July 2014
International Journal of Advance Research in Computer Science and Management Studies
Research Article / Survey Paper / Case Study
Available online at: www.ijarcsms.com

SOA Based Multiparty Session Authentication

Ryakala Deepika[1]
Student of M.Tech
Department of Information technology
Hyderabad – India

K. Bhima[2]
Associate Professor
Department of Information technology
Hyderabad – India

Abstract: ***In a dynamic and distributed environment, it is often difficult for a complex business process to follow a static business specification. The applications and services involved in a complex business process are typically heterogeneous, provided by different organizations. Since each organization has its own security mechanisms and policies to protect its local resources, the business process has to operate among multiple, heterogeneous security realms. For these applications, security procedures are often omitted in the interest of performance. Collaborating services in a system with a Service-Oriented-Architecture (SOA) may belong to different security realms but often need to be engaged dynamically at runtime. If their security realms do not have a direct cross-realm authentication relationship, it is technically difficult to enable any secure collaboration between the multiple services. A possible solution to this problem is to provide authentication path between the intermediate realms at runtime. However, the process of generating an authentication path for two distributed services can be highly complicated. It could involve a large number of extra operations for credential conversion and require a long chain of invocations to intermediate services. In this paper, we address this problem by designing and implementing a new cross-realm authentication protocol for dynamic service interactions, based on the notion of service-oriented multiparty business sessions.***

Keywords: SOA, Multiparty sessions, Authentication, cross- realm authentication.

I. INTRODUCTION

Modern distributed applications are embedding an increasing degree of dynamism, from dynamic supply chain management, enterprise federations, and virtual collaborations to dynamic service interactions across organizations. Such dynamism leads to new security challenges. Collaborating services may belong to different security realms but often have to be engaged dynamically at run time. If their security realms do not have in place a direct cross-realm authentication [6] relationship, it is technically difficult to enable any secure collaboration between the services. Because organizations and services can join a collaborative process in a highly dynamic and flexible way, it cannot be expected that every two of the collaborating security realms always have a direct cross-realm authentication relationship. A possible solution to this problem is to locate some intermediate realms that serve as an authentication-path between the two separate realms that are to collaborate. However, the overhead of generating an authentication-path for two distributed realms is not trivial. The process could involve a large number of extra operations for credential conversion and require a long chain of invocations to intermediate services. This problem is addressed by presenting a new cross realm authentication method: It offers the ability to identify individual service instances within a business session even if some instances in fact belong to the same service. Although the amount of communications between the partners of a session and the Session Authority is limited, the performance overhead imposed by it is indeed of some practical concern.

Existing cross-realm authentication mechanisms require either federated authentication by maintaining direct cross realm authentication relationships between any pair of security realms or additional credential conversion from one realm to another. We believe what is needed is a “dynamic” scheme for multiparty authentication. Unlike traditional “static” authentication

protocols which assume preexisting and fixed authentication relationships, a new scheme should be able to establish instantly at runtime an authentication relationship between any pair of principals that wish to collaborate. In this paper we provide dynamic multiparty authentication based on SOA business processes [1].

II. LITERATURE SURVEY

A service-oriented architecture is an information technology approach or strategy in which applications make use of services available in a network such as the World Wide Web. Implementing a service-oriented architecture can involve developing applications that use services, making applications available as services so that other applications can use those services, or both. In software engineering, a Service-Oriented Architecture (SOA) is a set of principles and methodologies for designing and developing software in the form of interoperable services. These services are well-defined business functionalities that are built as software components (discrete pieces of code and/or data structures) that can be reused for different purposes. SOA design principles [1]are used during the phases of systems development and integration. With the development of business processes, collaboration amongst organizations has become increasingly common. The emergence of web services further promotes this tendency. Such cross organizational collaboration imposes new security challenges on authentication systems. Most of existing authentication methods [2] (e.g. Kerberos) utilizes two-party sessions to enforce their security management, which are obviously different from the method that is presented here. Generally there are two potential solutions to address the dynamic cross-realm authentication issue raised in the integration of heterogeneous security realms.

Federated Authentication: Federated authentication offers a method for implementing a direct authentication relationship between different organizations and business partners. For example, By using federated authentication, now when the employee clicks on a link to their benefits supplier from within the enterprise, your enterprise single sign on or federated identity system generates a security assertion token, which is passed to the benefits supplier. The benefits supplier receives the token, validates it and then determines if it will accept the authentication for the employee. The employee is then allowed immediate access to their information without having to re-authenticate using the benefit supplier's id and password. As a result, the user is increasingly frustrated with:1)Having to remember multiple user id and passwords.2)Providing more identity information than they would otherwise chose to each entity.

Authentication path of credential conversion: authentication path of credential conversion between the two separate realms that are to collaborate. However, the overhead of generating an authentication path for two distributed realms is not trivial, which requires the cooperation from intermediate security realms. The process may involve a large number of extra operations for credential conversion as well as a long chain of invocations to intermediate services. Moreover, such an authentication path of credential conversion between two security realms may not even exist.

MULTIPARTY SESSION: Informally, a *session* is a series of interactions which serve as a unit of conversation. A session is established among multiple parties via a *shared name*, which represents a public interaction point. Then fresh *session channels* are generated and shared through which a series of communication actions are performed. The foregoing studies on session types have focused on two-party (binary) sessions. While many conversation patterns can be captured through a composition of two party sessions, there are cases where binary session types are not powerful enough for describing and validating interactions which involve more than two parties.

A multiparty session may have two or more session partners for an intended collaboration. A partner can search for and invoke new services at runtime. Before a service (instance) is accepted as a new partner, a Heterogeneous Cross-Realm Authentication process [5] is normally needed. However, unlike a two-party session, authentication for the existing partners of a multiparty session could be simplified significantly without requiring credential conversion and the establishment of any authentication path. This is because the session partners can make use of their shared session secrets and memberships to

authenticate each other even if they belong to different security realms. A shared session key or individual secret keys may be used to enforce a secure collaboration among session partners.

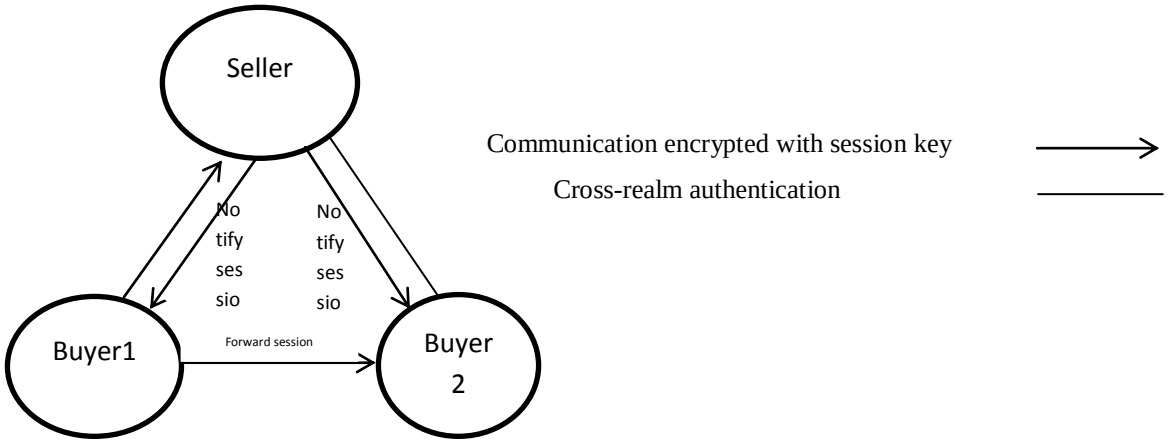

Fig 1: multiparty session scenario

Figure1 shows as an example for multiparty session; let us consider two participating services: consider two buyers, Buyer1 and Buyer2, wish to buy an expensive book from Seller by combining their money. Buyer1 sends the title of the book to Seller, Seller sends to both Buyer1 and Buyer2 its quote, Buyer1 tells Buyer2 how much she can pay, and Buyer2 either accepts the quote or receives the quote by notifying Seller. It is extremely awkward (if logically possible) to decompose this scenario into three binary sessions, between Buyer1 and Seller, between Buyer2 and Seller, and between Buyer1 and Buyer2. Abstracting this protocol as three separate session types also means that our type abstraction loses essential sequencing information in this interaction scenario. For validating this conversation scenario as a whole, therefore, the conversation needs heterogeneous cross realm authentication. This HCRA process is both costly and complex. In some cases, it is practically impossible to perform HCRA in an automatic and on-the fly manner, and human intervention will be needed. For a business session involved with n heterogeneous security realms, the HCRA process would have to be repeated n * n-1/2 times to allow all possible partner interactions with the session. There are different ways to start a multiparty business session, e.g., one or a group of services may serve as a starter to initiate a session. Any existing session partner of the session may introduce new members into the session. If HCRA between the existing partner and the potential new member has been performed based on the established trust relationship between them, the authentication process for the session will be further simplified. In addition, there is some recent progress in developing protocols and mechanisms for HCRA based on automatic negotiation at runtime. The work could be combined with SCRA to reduce further the cost and complexity of dynamic cross-realm authentication. It is important to notice that multiparty business sessions are also a useful design abstraction for controlling and coordinating multiparty interactions and collaborations. While it is convenient to use this abstraction to develop our proposed authentication system for automatic SOA based business processes, our system could be combined with other models for multiparty interactions such as atomic actions and process groups. However, regardless of the model used, a multiparty authentication system[6][7] needs to address the issues with message routing and secret keys for communications. A Session Authority (SA)[6] is also required to provide reliable real-time information (e.g., memberships) about session partners.

The Session Authority is responsible for allowing participating Web Services to join a session and enforcing some rules for behavior in a session. Specifically, Maruyama and Hada state that the Session Authority is responsible for "assigning session ids, creating session secrets, maintaining the status information for each session" as well as keeping all session participants informed of the status.

III. Model Analysis

Here in this section we will define two authentication protocols [5] in our multiparty authentication system[7] using the well-known Logic of Authentication[2]. Protocol 1 is concerned with the introduction of a new session partner, and Protocol 2

performs authentication between two existing session partners. Notice that all messages of the protocols are enveloped in the XML documents and transported by the WS-protocols (e.g., SOAP). It implies that types of the messages are tagged.

Fig.2 illustrates Protocol 1: Accepting a new session partner. Our protocol conforms to the WS-Resource Framework (WSRF) specification, where a service is associated with a factory service F that generates service instances. The details of the messages transported within Fig. 4 are presented as follows, where "A → B" means that A sends a message to B:

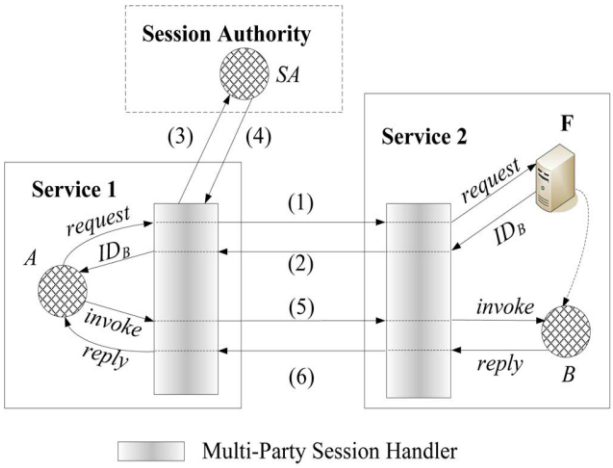

Fig 2: Protocol 1: Accepting a new session partner.

1: A→F: Secure (Request; ID_S, ID_A)

2: F→A: Secure (ID_B, ID_S)

3:A→SA:Valid (SP(B,S),ID_B, ID_A, ID_{SA},ID_S,N)$_{K(A,SA)}$

4: SA→A: Valid (Confirm, N +1) $_{K(SA,A)}$

5: A→B: Valid (Invoke, ID_A, ID_B, ID_S, N1)$_{K(A,B)}$

6: B→A : Valid(Reply; ID_B; ID_A; ID_S,N_1+1)$_{K(B,A)}$ where N and N1 are fresh nonce.

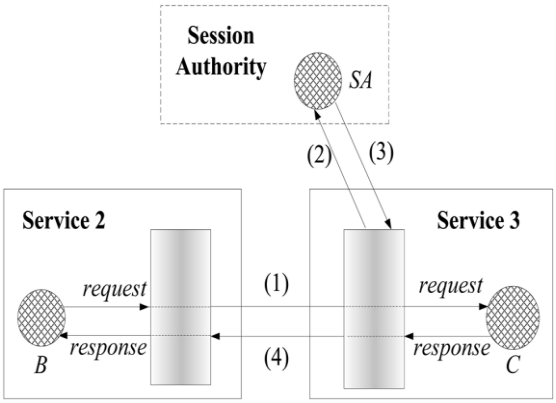

Fig.3. Protocol 2: Authenticating a session partner.

Fig. 3 illustrates Protocol 2: Authenticating a session partner. B and C are session partners of S, but B has not yet communicated with C before. First, B sends a request message (1) to C. C then sends message (2) to SA in order to check the identity of B. SA will send back a confirmation in message (3), confirming that B is a session partner of S. After receiving the confirmation, B will handle the request from C and send the result back. All the messages transferred during this process are encrypted by the secret key generated with the Diffie-Hellman algorithm. The details of the messages passed in Fig.3 are presented as follows:

1: B→C: Valid (Request, ID_B, ID_C, ID_S, N_1) $_{K(B,C)}$

2: C→SA: Valid (Query, IDB,IDC,IDSA,IDS,N2) $_{K(C,SA)}$

3:SA→C:Valid (SP(B,S),ID_{SA},ID_C,ID_S, N_2 + 1)$_{K(SA,C)}$

4: C→B:Valid (Response, ID_C, ID_B, ID_S, N_1 + 1)$_{K(C,B)}$

Where N_1 and N_2 are fresh nonce.

In Protocols 1 and 2, MACs are used to protect the integrity of the messages transported within a business session, and fresh nonce is used to guarantee that a message is not replayed.

Notations:

A, B, C are session partners.

SA :session authority.

ID_A : identifier of A.

S : multiparty session with identifier ID_S.

K(A,B) : secret key generated with Pri(A) and Pub(B)

SP(A,S) : statement that A is a session partner of S. Particularly, SP(SA, S) is always true.

Besides analytic assessments, the feasibility of the proposed authentication system in real-world applications also needs to be examined. Consequently, a series of experiments has been conducted using two production-quality Grid middleware systems in order to assess:1. Scalability of our multiparty authentication system, 2. Compatibility of the system with other common message-level security protocols, and 3. runtime overheads of the mechanism under different conditions.

IV. CONCLUSION

The dynamic authentication process between organizations could be highly complex and time consuming since intermediate authentication paths need to be created at runtime to dynamically covert credentials from different security realms. If there is no existing authentication relationship in place between two organizations, it will be practically difficult for a system to enable any secure collaboration between services from the two organizations in a just-intime fashion. many conversation patterns can be captured through a composition of binary sessions, there are cases where binary session types are not powerful enough for describing and validating interactions which involve more than two parties.

Here in this application we provide a multiparty authentication system. That does not require credential conversion and establishment of authentication paths between collaborative session partners. The system also offers the ability to identify individual service instances within a business session even if some instances in fact belong to the same service. Although the amount of communications between the partners of a session and the Session Authority is limited.

References

1. Service-Oriented Architecture: A Field Guide toIntegrating XML and Web Services (The PrenticeHall Service-Oriented Computing Series from ThomasErl).
2. M. Burrows, M. Abadi, and R. Needham, “A Logic of Authentication,” ACM Trans. on Computer Systems, Scheme (EDAS), Information Systems Frontiers, v.16 n.1, p.113-127, March 2014.
3. F. Cabrera, G. Copeland, T. Freund, J. Klein, D. Langworthy, D. Orchard, J. Shewchuk, and T. Storey, “Web Services Coordination (WS-Coordination),” available from http:// www. ibm. com/ developerworks/ library/ws-coor/ .
4. A. Perrig, “Efficient and Secure Source Authentication for Multicast,” Network and Distributed System Security Symp., Feb. 2005.
5. S.Hada and Maruyana,”Session Authentication protocol for web services”.
6. Dynamic cross realm authentication, IEEE TRANSACTIONS ON SERVICES COMPUTING, VOL. 5, NO. 1. ByJie Xu, Member, IEEE Computer Society, Dacheng Zhang, Lu Liu, Member, IEEE, and Xianxian Li.
7. D.Zhang, “Dynamic authentication for multipartyservice interactions”.
8. An analysis of Hada and Maruyama's proposed multi-party authentication mechanism for Web Service technologies by Kevin Berkowitz.

