Secure Web Services Using Two-way Authentication and Three-party Key Establishment for Service Delivery

Song Han, Tharam Dillon, Elizabeth Chang, Biming Tian

DEBII, Curtin University of Technology
GPO Box U1987 Perth, WA 6845, Australia

Abstract

With the advance of web technologies, a large quantity of transactions have been processed through web services. Service Provider needs encryption via public communication channel in order that web services can be delivered to Service Requester. Such encryptions can be realized using secure session keys. Traditional approaches which can enable such transactions are based on peer-to-peer architecture or hierarchical group architecture. The former method resides on two-party communications while the latter resides on hierarchical group communications. In this paper, we will use three-party key establishment to enable secure communications for Service Requester and Service Provider. The proposed protocol supports Service Requester, Service Broker, and Service Provider with a shared secret key established among them. Compared with peer-to-peer architecture and hierarchical group architecture, our method aims at reducing communication and computation overheads.

Key words: Web Service, key establishment, non-commutative monoid, encryption, shared secret key, transactions, security, service requester, service provider, service delivery

* Corresponding. DEBII, CBS, Curtin University of Technology
GPO Box U1987 Perth, WA 6845, Australia
Email addresses: song.han@cbs.curtin.edu.au (Song Han),
tharam.dillon@cbs.curtin.edu.au (Tharam Dillon),
elizabeth.chang@cbs.curtin.edu.au (Elizabeth Chang),
biming.tian@cbs.curtin.edu.au (Biming Tian).

Preprint submitted to Journal of Systems Architecture

1 Introduction

Resource delivery can be any activity through networks involving service request and provision. It may be for individuals, companies or government agencies. Because of the various possible attacks through heterogeneous networks [1, 5, 7, 13, 45, 28, 34, 35], resource delivery requires some security mechanisms to protect data privacy associated with the requested resource. Service requester and service provider are distributed over different networks and domains. Therefore mechanisms for preventing eavesdropping of data from unauthorized parties have to be considered in any resource delivery which may be sensitive [1 - 7, 9, 19, 48, 22, 47, 39, 40].

Consider the following scenario for an e-government: Suppose the Taxation Office is a service requester. The Taxation Office (**TO**) may request an individual visa and employment history. The service provider can be the Department of Education and Employment(**DEEW**). The Department of Immigration and Citizenship (**DIC**) can act as a broker. Thus, **DEEW** can provide **TO** with the requested resource via **DIC**. Since **DIC** is a trusted party, the communications between **TO** and **DEEW** can be delivered via **DIC**. Another example is: You may use your mobile phone to direct your home computer to transfer your meeting schedule to your office printer and print out this schedule for you. Your mobile phone will act as a service requester and your office printer is a service provider while your home computer is a broker in this scenario.

In the above scenarios, the three parties in each situation can share one same session key for their resource delivery service. In this paper, we will develop a secure service protocol for this kind of resource delivery. Our method is based on non-commutative monids and one-way hash function. We will propose a two-way authenticated and three-party key establishment protocol which can address the above two scenarios.

A key establishment protocol is used to derive a shared secret by two or more parties as a function of information contributed by, or associated with, each of these, but no single party can predetermine the resulting value. The security of most key establishment protocols were based on large integer factoring infeasibility or discrete logarithm infeasibility [41]. Several key establishment protocols based on group theory have been proposed [8, 49, 12, 30]. However, all these protocols are for two parties to establish a secret key. These schemes cannot be transferred to the three-party scenario in their present forms. Therefore, it is interesting to propose three-party key establishment based on non-commutative monoids. In addition, the proposed key establishment should be immune from the existing attacks on algebraic method based cryptographic primitives.

It is true that the WS-world was not initially expected to serve in low-computation devices or similar scenarios that may have particular restrictions in their overhead, but in servers providing these WS-enabled services. However, we notice that the increasing popularity of the mobile devices such as laptop, PDA, and tablet type devices allow new opportunities to develop solutions beyond mobile email including other activities of web services. Therefore, it also opens a new window for delivering the mobile infrastructure which can support mobile web services. Apparently, low communication and computation overheads are positive for mobile web services. That is a sound motivation behind our proposal for the presented two-way authentication and three-party key establishment. There are of course many other motivations. One of them could be reducing time associated with secure web services arising from high overheads.

Some basic security requirements for resource delivery where three parties (service requester, broker and service provider) are involved are identified as follows:

- Requirement 1. Authentication: Before the resource delivery is executed, the participated parties need to be authenticated. That is, Service requester needs to be authenticated to Broker and Service provider. Broker needs to be authenticated to Service requester and provider. Service provider needs to authenticated to Broker and Service requester. All the authentications are required to be two-way authentication [5, 7, 11, 16, 17, 31, 40, 42].
- Requirement 2. Data Privacy: In the whole course of resource delivery, data privacy should be protected. It has two requirements: on the one hand, data associated with the resource should not modified while being delivered; on the other hand, data associated with the resource should be kept unknown to any unauthorized party [18, 21, 46, 32, 42].

Requirement 1 can be attained through password-based authentication which is adopted in our proposed two-way authentication in section 4. Requirement 2 can be attained through session key establishment and secure resource delivery using the established session key.

We need to clarify that other security requirements such as confidentiality, authorization, integrity and non-repudiation (see [41] for the details of these definitions) are also needed for full and secure deployment of the above two scenarios. However, our proposal only focuses on data authentication and privacy. Other techniques such as encryption and digital signature can solve confidentiality and authorization or non-repudiation requirements [41]. Access control techniques and policies can address the authorization requirement [41]. Therefore, we did not address them in the proposed protocol.

The use of web services can pose serious risk if security is not properly ad-

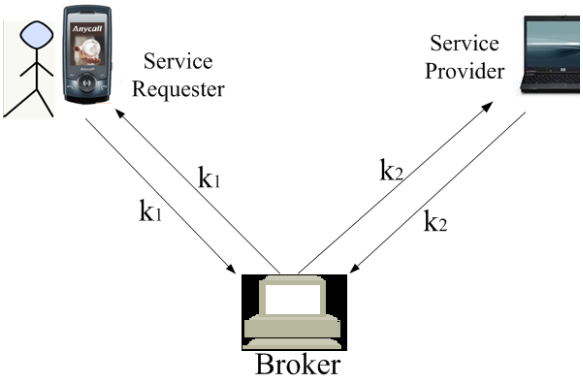

Fig. 1. Two different shared keys are established in the peer-to-peer model.

dressed from the beginning. The purpose of our proposal is not to examine current web service security standards but to facilitate secure web service transactions within a service oriented architecture by using our three-party schema which does not rely on a static and pre-built secure communication channel. Web service networks are dynamic and grow organically, which increases the cost of management and administration. Because of the dynamic nature, it is not realistic to rely on a static and pre-built secure communication channel for some concrete applications such as those two scenarios mentioned above. Our proposal can be combined with some of current web service security standards such as Security Assertion Markup Language (SAML) [5], XML Encryption [3], and XML Signature [2] to enable authorization information, end-to-end encryption, and message integrity.

2 Related Work

In this section, we will review some existing schemes which provide secure web services for service discovery and low-cost devices.

If there are three parties in the service delivery process, then peer-to-peer based model leads to two different shared keys which would be established. See the example as in Figure 1.

Helander and Xiong proposed a secure web service framework based on peer-to-peer model [23, 25]. That framework provided two-way authentication and peer-to-peer key exchange. Their method were based on RSA and AES [41]. The two-way authentication and peer-to-peer key exchange were done with RSA. The transmission and communications were encrypted with an AES peer-to-peer key. RSA-based method is well-known energy-consumed due to its large number involvement (e.g. 2048 bit-length number may be needed for maintaining some security level) [41].

Campo et al.'s security model is based on hierarchical group communication [48]. The authors developed a multiprotocol service discovery platform. They studied mechanisms to enhance availability of services in the network. They did not present a concrete scheme for their framework.

The protocol in [51] was the service discovery chosen by the Bluetooth SIG to search for services in Bluetooth networks. Clients connect to all the prior discovered devices in the proximity and check if they are offering the desired service. This protocol uses a very simple scheme where security relies on a secure pairing between devices and scalability is limited to a low number of devices. It only works for Bluetooth networks and there is no way to run it on any other network technology. These features are not enough in terms of security and scalability.

The protocol in [36] supported multicast mechanism, by which the client multicasts a template of the service and any server receiving the message compares the pattern with its own services. If any service matches, the server answers to the client with the address of the service. The drawback of this protocol is: every client needs to maintain the public key of every server in the network. Another issue is sending the address in clear makes IP spoofing possible [41, 42].

Authentication and key establishment can be realized using either the symmetric key method or the public key method [41]. The first symmetric key protocol involving online trusted third parties was proposed by Needham and Schroeder in [29]. This protocol was shown to be insecure by Denning and Sacco in [37]. The insecurity is because of one party being not able to ensure messages are fresh. Denning and Sacco suggested fixing the freshness flaw in [29] by the use of timestamps.

Otway and Rees in [43] proposed another symmetric key based protocol which involves an online server. This protocol was compromised by Burrows et al in [14] by fooling one party to believe a public message is a new established key.

Woo and Lam in [27] proposed another two-way authentication and key establishment protocol. This protocol was compromised by Clark et al. in [24] using a parallel session attack. The parallel session attack can enable party A to fool party B to accept a previously established key as a new one. This attack is not a particularly strong attack but indicates that the protocol does not provide the authentication property. Lowe in [33] developed a more serious attack using message component symmetry.

Needham and Schroeder in [29] proposed a public key based protocol which involves an online trusted third party. The public key based method needs the deployment of Public Key Infrastructure (PKI). Therefore, it is expensive from a system cost point of view. This protocol was broken by Lowe in [33].

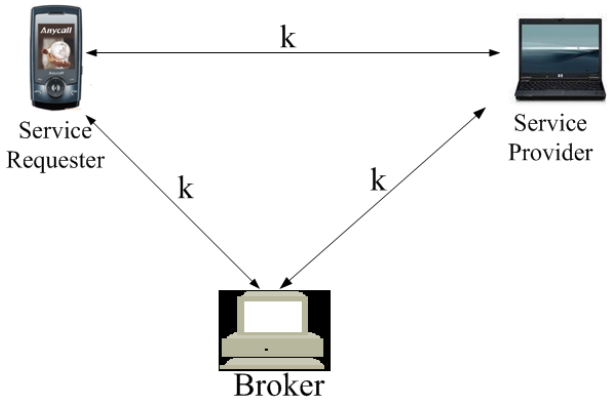

Fig. 2. One shared key is established in our model.

Neuman and Stubblebine presented another protocol in [20] which was broken by Hwang et al. in [10].

Compared to all the above protocols, the ISO/IEC protocol [50] is an efficient one which is based on the symmetric key method without an online trusted third party. Our proposal will use the method of symmetric key protocol without an online trusted third party.

In order to address the above issues, we will propose a new web service protocol which uses two-way authentication and three-party key establishment for resource delivery. Our protocol is targeting at three-party resource delivery scenario, i.e. there are three participants, Service requester, Broker, and Service provider. They only need to establish one shared key for fulfilling the resource delivery securely. The figure 2 gives an overview for this scenario as above:

3 Preliminary

In our scheme, we use non-commutative monoids for the session key establishment. In this section, we therefore review the mathematical definition for monoids, generators, submonids, commutative monoids, and non-commutative monoids.

Monoids have not been much utilized and studies in cryptography although there are several key establishment protocols based on monoids were developed [8, 49, 12]. To the best of our knowledge, our paper is the first one which introduces monoids for web service protocols. Therefore, it is necessary to give a background regarding monoids in this section.

Definition In abstract algebra, a monoid is an algebraic structure with a

single, associative binary operation and an identity element. A monoid is a set $\mathbf{M}$ with binary operation $* : \mathbf{M} \times \mathbf{M} \rightarrow \mathbf{M}$, obeying the following axioms:

- Associativity: for all $a, b, c \in \mathbf{M}$, $(a * b) * c = a * (b * c)$.
- Identity element: there exists an element $e \in \mathbf{M}$, such that for all $a \in \mathbf{M}$, $a * e = e * a = a$.
- Closure: for all $a, b \in \mathbf{M}$, $a*b$ is in $\mathbf{M}$. Alternatively, a monoid is a semigroup with an identity element.

A monoid satisfies all the axioms of a group with the exception of having inverses. A monoid with inverses is the same thing as a group [26, 38].

Definition Generators and Submonoid [26, 38]: A submonoid of a monoid $\mathbf{M}$, is a subset $\mathbf{N}$ of $\mathbf{M}$ containing the unit element, and such that, if $x, y \in \mathbf{N}$, then $x * y \in \mathbf{N}$. It is then clear that N is itself a monoid, under the binary operation induced by that of M. Equivalently, a submonoid is a subset N such that N=N*, where the superscript * is the Kleene star. For any subset N of M, the monoid N* is the smallest monoid that contains N.

A subset N is said to be a generator of M if and only if M=N*. If N is finite, then M is said to be finitely generated.

Definition Commutative monoid: A monoid whose operation is commutative is called a commutative monoid (or, less commonly, an abelian monoid). Commutative monoids are often written additively. Any commutative monoid is endowed with its algebraic preordering , defined by xy if and only if there exists z such that $x + z = y$. An order-unit of a commutative monoid M is an element u of M such that for any element $x \in M$, there exists a positive integer n such that x nu. This is often used in case M is the positive cone of a partially ordered abelian group G, in which case we say that u is an order-unit of G. There is an algebraic construction that will take any commutative monoid, and turn it into a full-fledged abelian group; this construction is known as the Grothendieck group [26, 38].

Definition Partially commutative monoid: A monoid for which the operation is commutative for some, but not all elements is a trace monoid; trace monoids commonly occur in the theory of concurrent computation.

Definition Acts and transition systems: An operator monoid is a monoid M which acts upon a set X. That is, there is an operation $\cdot : M \times X \longmapsto X$ which is compatible with the monoid operation.

* For all $x \in X$: $e \cdot x = x$. * For all $a, b \in M$ and $x \in X$: $a \cdot (b \cdot x) = (a \cdot b) \cdot x$.

Operator monoids are also known as acts (since they resemble a group action), transition systems, semiautomata or transformation semigroups [38].

Definition Non-commutative monoid: A monoid whose operation is non-commutative is called a non-commutative monoid (or, an non-abelian monoid).

4 Web Service Protocol for Service Requester and Service Provider

In this section, we will provide the security protocol for web service delivery designed for service requester and service provider. An overview of the new protocol will be first presented. A concrete construction will be developed after the overview.

4.1 Overview of the protocol

The proposed resource delivery protocol is composed of four procedures:

Step 1. System Setup This procedure is used to set up system parameters. Two computable and non-commutative monoids $\mathbf{S}$ and $\mathbf{T}$ are needed in the protocol. Three maps f, β and θ are operations over $\mathbf{S}$ and $\mathbf{T}$. $H()$ is a one-way hash function, which maps elements in $\mathbf{T}$ to $\mathbf{T}$. $E_k()$ is a symmetric encryption algorithm which uses k as a secret key. There is a system initiator which helps Service requester, Broker and Service provider setup those parameters.

Step 2. Two-way authentication and Session Key Establishment In order to find the resource for Service requester and to securely delivery this resource from the right Service provider to the right Service requester, security mechanisms need to be designed to satisfy the purposes. Here we use two-way authentication and session key establishment to obtain the security mechanisms. Two-way authentication [31] means Service requester, Broker and Service provider are authenticated with each other. The session key establishment is used to set up a secret session key between Service requester, Broker and Service provider.

Step 3. Session Key Confirmation After Service requester, Broker and Service provider execute the Session Key Establishment procedure, they need to run the Session Key Confirmation procedure. The Session Key Confirmation procedure is used to confirm Service requester, Broker and Service provider with the fact that they really share a secret key.

Step 4. Transaction Encryption and Service Delivery After the Session Key Confirmation procedure returns '*true*', Service provider can use the shared secret key to encrypt the resource (service) and sends it to Service requester. If any dispute is raised, Broker can use the shared key to decrypt the ciphertext and resolve the dispute.

4.2 Construction of the protocol

Assume three participants Alice, Broker and Service provider will be involved in the following protocol. Their unique means of communications is through public channels. Here *one-time* three-party key agreement indicates that the participants re-choose their secret keys for every time protocol run. This can help to prevent attacks from compromising possible long-term secret keys.

Step 1. System Setup

Consider a 5-tuple: $(\mathbf{S}, \mathbf{T}, f, f_1, f_2)$, where $\mathbf{S}$ and $\mathbf{T}$ are computable and non-commutative monoids. The three maps f, f_1 and f_2 are operations over $\mathbf{S}$ and $\mathbf{T}$ and defined as follows:

$$f : \mathbf{S} \times \mathbf{S} \mapsto \mathbf{T}$$
$$\beta : \mathbf{S} \times \mathbf{T} \mapsto \mathbf{T}$$
$$\theta : \mathbf{S} \times \mathbf{T} \mapsto \mathbf{T}$$

They adhere to three axioms [30]:

- Axiom 1: For all g, g_1, and $g_2 \in \mathbf{S}$, $f(g, g_1 \cdot g_2) = f(g, g_1) \cdot f(g, g_2)$;
- Axiom 2: For any g, $h \in \mathbf{S}$, $\beta(g, f(h, g)) = \theta(h, f(g, h))$;
- Axiom 3: Given public elements g_1, g_2, ..., $g_n \in \mathbf{S}$, $h \in \mathbf{S}$ is a secret element, while
  $$f(h, g_1), f(h, g_2), ..., f(h, g_n)$$
  are publicly known. Then, to determine h is not computable in polynomial time (i.e. it is infeasible in polynomial time to obtain h by solving $f(h, g_1), f(h, g_2), ..., f(h, g_n)$).

The system also needs a one-way hash function $H()$, which maps elements in $\mathbf{T}$ to $\mathbf{T}$. The system initiator further selects a symmetric encryption algorithm $E_k()$, which uses k as a secret key. The system initiator helps Service requester, Broker and Service provider setup those parameters.

Here, Service requester is an entity who makes request for resource delivery. Broker is an entity who helps service requester find the required resource from an available service provider. Service provider is an entity who can supply the resource required by the Broker on behalf of service requester.

Suppose n_1, n_2, and n_3 are three positive integers. We assume

(1) $\mathbf{S_A} \neq \mathbf{S_B}$,
(2) $\mathbf{S_B} \neq \mathbf{S_C}$
(3) $\mathbf{S_C} \neq \mathbf{S_A}$

for the following three monoids $\mathbf{S_A}$, $\mathbf{S_B}$, and $\mathbf{S_C}$.

(1) Service requester is assigned a public monoid $\mathbf{S_A} \subsetneqq \mathbf{S}$. Suppose S_A is generated by the elements

$$a_1, a_2, \cdots, a_{n_1}.$$

That is, for any element $x \in \mathbf{S_A}$, x can be represented as $x = \prod_{i=1}^{n_1} a_i^{k(i)}$, where k_i $(1 \leq i \leq n_1)$ are non-negative integers .

(2) Broker is assigned a public monoid $\mathbf{S_B} \subsetneqq \mathbf{S}$. Suppose S_B is generated by the elements

$$b_1, b_2, \cdots, b_{n_2}.$$

(3) Service provider is assigned a public monoid $\mathbf{S_C} \subsetneqq \mathbf{S}$. Suppose S_C is generated by the elements

$$c_1, c_2, \cdots, c_{n_3}.$$

(4) Service requester randomly chooses n_1 non-negative integers $e_1(1), e_1(2), \cdots, e_1(n_1)$ and computes

$$a = \prod_{i=1}^{n_1} a_i^{e_1(i)}.$$

Then $a \in \mathbf{S_A}$ (Service requester keeps $e_1(i)(1 \leq i \leq n_1)$ privately). She then computes

$$f(a, b_1), f(a, b_2), \cdots, f(a, b_{n_2})$$

and

$$f(a, c_1), f(a, c_2), \cdots, f(a, c_{n_3})$$

Service requester's secret key is a while the public key includes $\{f(a, b_1), f(a, b_2), \cdots, f(a, b_{n_2})\}$ and $\{f(a, c_1), f(a, c_2), \cdots, f(a, c_{n_3})\}$.

(5) Service requester then registers $\{f(a, b_1), f(a, b_2), \cdots, f(a, b_{n_2})\}$ and $\{f(a, c_1), f(a, c_2), \cdots, f(a, c_{n_3})\}$ to the system initiator. The system initiator generates a pseudo ID PID_1 and a certificate φ_1 for Service requester.

(6) Broker randomly chooses n_2 non-negative integers $e_2(1), e_2(2), \cdots, e_2(n_2)$ and computes

$$b = \prod_{i=1}^{n_2} b_i^{e_2(i)}.$$

Then $b \in \mathbf{S_B}$ (Broker keeps $e_2(i)(1 \leq i \leq n_2)$ privately). Broker then computes

$$f(b, c_1), f(b, c_2), \cdots, f(b, c_{n_3})$$

and

$$f(b, a_1), f(b, a_2), \cdots, f(b, a_{n_1}).$$

Broker's secret key is b while public key includes $\{f(b, a_1), f(b, a_2), \cdots, f(b, a_{n_1})\}$ and $\{f(b, c_1), f(b, c_2), \cdots, f(b, c_{n_3})\}$

(7) Broker then registers $\{f(b, a_1), f(b, a_2), \cdots, f(b, a_{n_1})\}$ and $\{f(b, c_1), f(b, c_2), \cdots, f(b, c_{n_3})\}$ to the system initiator. The system initiator generates a pseudo ID PID_2 and a certificate φ_2 for Broker.

(8) Service provider randomly chooses n_3 non-negative integers $e_3(1), e_3(2), \cdots, e_3(n_3)$ and computes

$$c = \prod_{i=1}^{n_3} c_i^{e_3(i)}.$$

Then $c \in \mathbf{S_C}$ (Service provider keeps $e_3(i)(1 \leq i \leq n_3)$ privately). She then computes

$$f(c, a_1), f(c, a_2), \cdots, f(c, a_{n_1})$$

and

$$f(c, b_1), f(c, b_2), \cdots, f(c, b_{n_2}).$$

Service provider's secret key is c while public key includes $\{f(c, a_1), f(c, a_2), \cdots, f(c, a_{n_1})\}$ and $\{f(c, b_1), f(c, b_2), \cdots, f(c, b_{n_2})\}$.

(9) Service provider then registers $\{f(c, a_1), f(c, a_2), \cdots, f(c, a_{n_1})\}$ and $\{f(c, b_1), f(c, b_2), \cdots, f(c, b_{n_2})\}$ to the system initiator. The system initiator generates a pseudo ID PID_3 and a certificate φ_3 for Service provider.

Step 2. Two-way Authentication and Session Key Establishment

This procedure is composed of two sub-steps. It first enables Service requester, Broker and Service provider to authenticate each other. It then helps them to share a secret key

$$k = H(\beta(a, f(b, a)) \cdot \theta(c, f(b, c)) \cdot \theta(a, f(c, a))).$$

Step 2.1. Two-way Authentication After a potential service provider has been located, it is time for authentication and key exchange. Basically, authentication with PKI is a process to verify if a public key belongs to the right entity [31]. If a public key is verified to belong to an entity, then one can trust that any information encrypted with the public key can only be understood by the entity with the right private key. The attributes are used to check against an Access Control List to do authorization [31]. However, the PKI-based authentication method is expensive for mobile web services as the certificates need the deployment of the Public Key Infrastructure (PKI) [41]. Therefore, our proposal will adopt the password based authentication which is actually pre-shared secret authentication and was also integrated within TLS [41].

The added-value of our proposal compared to SSL/TLS or any of their evolutions to three party communications is that our proposal based on non-commutative monoids does neither need very large number computation in RSA or DSA settings nor need point multiplication in ECDSA setting while the current SSL/TLS authentication and key exchange candidates are RSA, DSA, Diffie-Hellman, ECDSA, ECDH, and SRP.

It is assumed that Service requester, Broker and Service provider share a password γ. The password can be shared in an off-line way. Here we do not spare more space on how the password is distributed to Service requester, Broker and Service provider but focus on the two-way authentication below.

(1) Service requester sends PID_1, and $E(\gamma : N_{sr}, PID_2, PID_3)$ to Broker and Service provider. Here $E(.)$ is a symmetric encryption algorithm (AES can be a candidate). N_{sr} is a nounce which is used to differentiate different authentication copies.
(2) Broker sends PID_2 and $E(\gamma : N_{br}, PID_1, PID_3)$ to Service requester and Service provider. N_{br} is a nounce which is used to differentiate different authentication copies.
(3) Service provider sends PID_3 and $E(\gamma : N_{sp}, PID_2, PID_1)$ to Broker and Service requester. N_{sp} is a nounce which is used to differentiate different authentication copies.
(4) After receiving messages, Service requester first decrypts $E(\gamma : N_{br}, PID_1, PID_3)$ and $E(\gamma : N_{sp}, PID_2, PID_1)$ to get N_{br} and N_{sp}, respectively. It then sends $E(\gamma : N_{br}, N_{sp}, PID_2, PID_3)$ to Broker and Service provider. Broker and Service provider can authenticate Service requester by verifying N_{br} and N_{sp}.
(5) After receiving messages, Broker first decrypts $E(\gamma : N_{sr}, PID_2, PID_3)$ and $E(\gamma : N_{sp}, PID_2, PID_1)$ to get N_{sr} and N_{sp}, respectively. It then sends $E(\gamma : N_{sr}, N_{sp}, PID_1, PID_3)$ to Service requester and Service provider. Service requester and Service provider can authenticate Broker by verifying N_{sr} and N_{sp}.
(6) After receiving messages, Service provider first decrypts $E(\gamma : N_{sr}, PID_2, PID_3)$ and $E(\gamma : N_{br}, PID_1, PID_3)$ to get N_{sr} and N_{br}, respectively. It then sends $E(\gamma : N_{sr}, N_{br}, PID_1, PID_2)$ to Service requester and Broker. Service requester and Broker can authenticate Broker by verifying N_{sr} and N_{br}.

After Service requester, Broker and Service provider authenticate each other, they go to the next step - Session Key Establishment.

Step 2.2. Session Key Establishment

Service requester, Broker and Service provider will establish a shared secret key through running the following protocol. One may wonder why selecting

a three-party schema for it. It is true there are multi-party methods to make secret sharing efficient by encrypting the secret, giving the encrypted secret to one participant and sharing the small encryption key among the others. However, this multi-party encryption method would introduce expensive cost as it will need a pre-built secure communication channel to deliver the small encryption key to other users. Fortunately, our method which adopted three-party schema does not need a pre-built secure communication channel.

(1) With Service requester and Broker's public keys, Service provider can use the secret key to compute

$$f(b,c) = \prod_{i=1}^{n_3} f(b,c_i)^{e_3(i)} \tag{1}$$

and

$$f(a,c) = \prod_{i=1}^{n_3} f(a,c_i)^{e_3(i)}. \tag{2}$$

Service provider then computes

$$d_1 = \theta(c, f(b,c))$$

and

$$d_2 = \beta(c, f(a,c)).$$

Finally, Service provider sends $d_1 \cdot d_2 \in \mathbf{T}$ to Service requester and Broker.

(2) With Broker and Service provider's public keys, Service requester can use the secret key to compute

$$f(b,a) = \prod_{j=1}^{n_1} f(b,a_j)^{e_1(j)} \tag{3}$$

and

$$f(c,a) = \prod_{j=1}^{n_1} f(c,a_j)^{e_1(j)}. \tag{4}$$

Service requester then computes

$$d_3 = \beta(a, f(b,a))$$

and

$$d_4 = \theta(a, f(c,a)).$$

Finally, Service requester sends $d_3 \cdot d_4 \in \mathbf{T}$ to Service provider and Broker.

(3) With Service requester and Service provider's public keys, Broker can use his secret key to compute

$$f(a,b) = \prod_{k=1}^{n_2} f(a,b_k)^{e_2(k)} \tag{5}$$

and

$$f(c,b) = \prod_{k=1}^{n_2} f(c,b_k)^{e_2(k)}. \tag{6}$$

Broker then computes

$$d_5 = \theta(b, f(a,b))$$

and

$$d_6 = \beta(b, f(c,b)).$$

Finally, Broker sends $d_5 \cdot d_6 \in \mathbf{T}$ to Service requester and Service provider.

(4) Service requester computes $K_1 = H(\beta(a, f(b,a)) \cdot (\theta(c, f(b,c)) \cdot \beta(c, f(a,c))))$.

(5) Broker computes $K_2 = H(\theta(b, f(a,b)) \cdot (\theta(c, f(b,c)) \cdot \beta(c, f(a,c))))$.

(6) Service provider computes $K_3 = H((\theta(b, f(a,b)) \cdot \beta(b, f(c,b))) \cdot \beta(c, f(a,c)))$.

The shared secret key of Service requester, Broker and Service provider is $k = K_1 = K_2 = K_3 = H(\beta(a, f(b,a)) \cdot \theta(c, f(b,c)) \cdot \theta(a, f(c,a))) \in \mathbf{T}$.

Step 3. Session Key Confirmation

In order to confirm the session has been successfully established, Service requester, Broker and Service provider execute the following steps.

(1) Service requester sends $\{A = H(K_1, PID_1, tmp_1), PID_1, stmp_1\}$ to Broker and Service provider. $stmp_1$ is the current time-stamp of Service requester.

(2) Broker sends $\{B = H(K_2, PID_2, tmp_2), PID_2, tmp_2\}$ to Service provider and Service requester. tmp_2 is the current time-stamp of Broker.

(3) Service provider sends $\{C = H(K_3, PID_3, tmp_3), PID_3, tmp_3\}$ to Service requester and Broker. tmp_3 is the current time-stamp of Service provider.

(4) Suppose the current time-stamps are respectively tmp_{21} and tmp_{31} when Service requester receives the messages from Broker and Service provider, respectively. Service requester first checks $|tmp_{21} - tmp_2|$ and $|tmp_{31} - tmp_3|$ are both less than an acceptable time threshold. It then checks whether $B = H(K_1, PID_2, tmp_2)$ and $C = H(K_1, PID_3, tmp_3)$. If both checks are *yes*, then Service requester believes that the secret key is successfully established.

(5) Suppose the current time-stamps are respectively tmp_{12} and tmp_{32} when Broker receives the messages from Service requester and Service provider, respectively. Broker first checks $|tmp_{12} - tmp_1|$ and $|tmp_{32} - tmp_3|$ are both less than an acceptable time threshold. Broker checks whether $A =$

$H(K_2, PID_1, tmp_1)$ and $C = H(K_2, PID_3, tmp_3)$. If both checks are *yes*, then Broker believes that the secret key is successfully established.

(6) Suppose the current time-stamps are respectively tmp_{13} and tmp_{23} when Service provider receives the messages from Service requester and Broker, respectively. Service provider first checks $|tmp_{13} - tmp_1|$ and $|tmp_{23} - tmp_2|$ are both less than an acceptable time threshold. Service provider checks whether $A = H(K_3, PID_1, tmp_1)$ and $B = H(K_3, PID_2, tmp_2)$. If both checks are *yes*, then Service provider believes that the secret key is successfully established.

If Service requester, Broker and Service provider all get ' *yes* ' in the above steps, then the Session Key Confirmation procedure returns ' *true* '.

Notice that $\{A = H(K_1, PID_1, tmp_1), PID_1, tmp_1\}$, $\{B = H(K_2, PID_2, tmp_2), PID_2, tmp_2\}$ and $\{C = H(K_3, PID_3, tmp_3), PID_3, tmp_3\}$ are transferred in clear. In order to maintain the privacy of all the three parties, we require PID_1, PID_2, and PID_3 are the pseudo ID of Service requester, Broker, and Service provider, respectively. Here, pseudo ID means that PID_1 is not the real identity of Service requester, but it can be used to identify Service requester uniquely by the involved parties, e.g. the Broker and the Service provider.

Step 4. Transaction Encryption and Service Delivery

Once the shared key is created, an encrypted connection can be set up between Service requester, Broker and Service provider. Encryption is applied to SOAP messages instead of transport layer packets like SSL does. Here, we do not spend space on the SOAP messages construction but focus on the encryption and service delivery. Please reference [1, 5, 7, 42] for the details of SOAP messages.

(1) By the resource requirement set by Service requester, Service provider prepares the service ϖ. ϖ is then encrypted with Service provider's secret key K_3 along with PID_1, PID_2, and PID_3 to get

$$\varepsilon = E_{K3}(\varpi, PID_1, PID_2, H(K_3), PID_3).$$

Service provider then sends ε, PID_3 and PID_1 to Service requester.

(2) After receiving ε, PID_3 and PID_1, Service requester first verifies PID_3 and PID_1 are correct. He then uses his secret key K_1 to decrypt ε and checks whether $H(K_1) = H(K_3)$. If yes, then Service requester believes the service is originally from Service provider and accepts this service. Otherwise, he sends request to Broker for getting correct service.

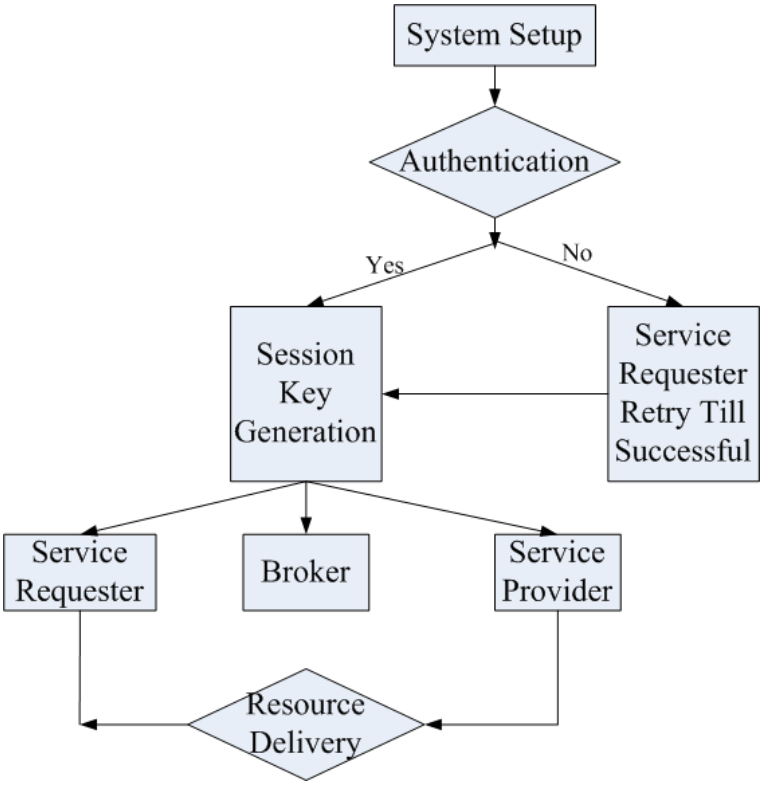

Fig. 3. The flowchart of the proposed secure resource delivery.

In the above procedure, due to the shared key

$$k = K_1 = K_2 = K_3 = H(\beta(a, f(b, a)) \cdot \theta(c, f(b, c)) \cdot \theta(a, f(c, a))) \in \mathbf{T},$$

we therefore have $H(K_1) = H(K_3)$. In the following section, we will discuss the security of the proposed scheme.

5 Security Discussion

We first explain why Service requester, Broker and Service provider can have the shared secret key $k = H(\beta(a, f(b, a)) \cdot \theta(c, f(b, c)) \cdot \theta(a, f(c, a))) \in \mathbf{T}$. Then we show an adversary cannot work out the shared secret key.

5.0.1 Service requester, Broker and Service provider can share the same key k

Throughout the process of Step 1, 2 and 3 in Section 4, we get the following:

- Service requester has d_3 and $d_1 \cdot d_2$ and can compute $d_7 = d_3 \cdot (d_1 \cdot d_2)$.
- Broker has d_5 and $d_1 \cdot d_2$ and can compute $d_8 = d_5 \cdot (d_1 \cdot d_2)$.

- Service provider has d_2 and $d_5 \cdot d_6$ and can compute $d_9 = (d_5 \cdot d_6) \cdot d_2$.

To explain Service requester, Broker and Service provider share the key

$$k = H(\beta(a, f(b, a)) \cdot \theta(c, f(b, c)) \cdot \theta(a, f(c, a))),$$

it is sufficient to show

$$H(d_7) = H(d_8) = H(d_9) = k.$$

Using the three Axioms specified in Step 1 of Section 4, we can get the following relations:

$$\begin{aligned} H(d_7) &= H(d_3 \cdot (d_1 \cdot d_2)) && (7)\\ &= H(\beta(a, f(b, a)) \cdot (\theta(c, f(b, c)) \cdot \beta(c, f(a, c)))) && (8)\\ &= H(\beta(a, f(b, a)) \cdot \theta(c, f(b, c)) \cdot \beta(c, f(a, c))) && (9)\\ &= H(\beta(a, f(b, a)) \cdot \theta(c, f(b, c)) \cdot \theta(a, f(c, a))) && (10)\\ &= k. && (11) \end{aligned}$$

$$\begin{aligned} H(d_8) &= H(d_5 \cdot (d_1 \cdot d_2)) && (12)\\ &= H(\theta(b, f(a, b)) \cdot (\theta(c, f(b, c)) \cdot \beta(c, f(a, c)))) && (13)\\ &= H(\beta(a, f(b, a)) \cdot (\theta(c, f(b, c)) \cdot \theta(a, f(c, a)))) && (14)\\ &= H(\beta(a, f(b, a)) \cdot \theta(c, f(b, c)) \cdot \theta(a, f(c, a))) && (15)\\ &= k. && (16) \end{aligned}$$

$$\begin{aligned} H(d_9) &= H((d_5 \cdot d_6) \cdot d_2) && (17)\\ &= H((\theta(b, f(a, b)) \cdot \beta(b, f(c, b))) \cdot \beta(c, f(a, c))) && (18)\\ &= H((\beta(a, f(b, a)) \cdot \beta(b, f(c, b))) \cdot \theta(a, f(c, a))) && (19)\\ &= H(\beta(a, f(b, a)) \cdot \theta(c, f(b, c)) \cdot \theta(a, f(c, a))) && (20)\\ &= k. && (21) \end{aligned}$$

5.1 Two-way authentication

As specified in Step 2.1 of Section 4, the following steps are used to provide two-way authentication mechanism.

(1) Service requester sends PID_1, and $E(\gamma : N_{sr}, PID_2, PID_3)$ to Broker and Service provider. Here $E(.)$ is a symmetric encryption algorithm (AES can be a candidate). N_{sr} is a nounce which is used to differentiate different authentication copies.

(2) Broker sends PID_2 and $E(\gamma : N_{br}, PID_1, PID_3)$ to Service requester and Service provider. N_{br} is a nounce which is used to differentiate different authentication copies.
(3) Service provider sends PID_3 and $E(\gamma : N_{sp}, PID_2, PID_1)$ to Broker and Service requester. N_{sp} is a nounce which is used to differentiate different authentication copies.
(4) After receiving messages, Service requester first decrypts $E(\gamma : N_{br}, PID_1, PID_3)$ and $E(\gamma : N_{sp}, PID_2, PID_1)$ to get N_{br} and N_{sp}, respectively. It then sends $E(\gamma : N_{br}, N_{sp}, PID_2, PID_3)$ to Broker and Service provider. Broker and Service provider can respectively authenticate Service requester by verifying N_{br} and N_{sp}, respectively .
(5) After receiving messages, Broker first decrypts $E(\gamma : N_{sr}, PID_2, PID_3)$ and $E(\gamma : N_{sp}, PID_2, PID_1)$ to get N_{sr} and N_{sp}, respectively. It then sends $E(\gamma : N_{sr}, N_{sp}, PID_1, PID_3)$ to Service requester and Service provider. Service requester and Service provider can respectively authenticate Broker by verifying N_{sr} and N_{sp}, respectively.
(6) After receiving messages, Service provider first decrypts $E(\gamma : N_{sr}, PID_2, PID_3)$ and $E(\gamma : N_{br}, PID_1, PID_3)$ to get N_{sr} and N_{br}, respectively. It then sends $E(\gamma : N_{sr}, N_{br}, PID_1, PID_2)$ to Service requester and Broker. Service requester and Broker can respectively authenticate Broker by verifying N_{sr} and N_{br}, respectively .

5.1.1 Security of $f(a,b)$, $f(c,b)$, $f(a,c)$, $f(b,c)$, $f(b,a)$ and $f(c,a)$

Without Service requester's private key, an adversary cannot compute $f(b,a)$ and $f(c,a)$ in polynomial time. This is because

$$f(b,a) = \prod_{i=1}^{n_1} f(b,a_i)^{e_1(i)}$$

and

$$f(c,a) = \prod_{i=1}^{n_1} f(c,a_i)^{e_1(i)}.$$

The security of $f(a,b)$, $f(c,b)$, $f(a,c)$, and $f(b,c)$ can be derived similary.

5.1.2 Security of $\beta(a, f(b,a))$, $\theta(a, f(c,a))$, $\theta(c, f(b,c))$ and $\theta(a, f(c,a))$

To identify the input a and $f(b,a)$ to the function β are both computably infeasible for an adversary in polynomial time. a is a secret key of Service requester while the adversary cannot compute $f(b,a)$ in polynomial time. Therefore, the adversary cannot work out $\beta(a, f(b,a))$. Similarly, an adversary cannot work out $\theta(a, f(c,a))$ and $\theta(c, f(b,c))$ in polynomial time.

5.1.3 Security against cut-and-paste or replaying attacks

An attacker targeting at an old session key may mount a cut-and-paste or replaying attack on the session key confirmation by copying some overdue transaction messages and replaying them with Service requester, Broker or Service provider. Assume the target is Broker to whom the attacker tries to compromise. Suppose the attacker got the i-th message

$$\{B = H(K_2(i), PID_2, tmp_2(i)), PID_2, tmp_2(i)\}$$

which was the copy that Broker sent to Service provider and Service requester in the i-th turn/round of web service transactions, where $K_2(i)$ was the session key established in the i-th turn/round of web service transactions and $tmp_2(i)$ was the time-stamp when Broker sent this message. The session key confirmation in section 4 can resist cut-and-paste or replaying attacks. This is because if the attacker wants to fool Service requester and Service provider successfully into believing the overdue session key $K_2(i)$ is a new session key $K_2(i+j)$ which is established by Service provider, Broker and Service requester in the $(i+j)$-th session key establishment. The attacker by replaying $\{B = H(K_2(i), PID_2, tmp_2(i)), PID_2, tmp_2(i)\}$ needs to convince respectively Service requester and Service provider of the following conditions:

(1) $|tmp_{21}(i+j) - tmp_2(i)|$ is less than the pre-set acceptable time threshold $\hat{\theta}$. $B = H(K_1(i+j), PID_2, tmp_2(i))$ is true.
Here $tmp_{21}(i+j)$ is the time-stamp when Service requester received the message $\{B = H(K_2(i), PID_2, tmp_2(i)), PID_2, tmp_2(i)\}$ from the attacker; $K_1(i+j)$ is the $(i+j)$-th session key Service requester has.

(2) $|tmp_{23}(i+j) - tmp_2(i)|$ is less than the pre-set acceptable time threshold $\hat{\theta}$. $B = H(K_3(i+j), PID_2, tmp_2(i))$ is true.
Here $tmp_{23}(i+j)$ is the time-stamp when Service provider received the message $\{B = H(K_2(i), PID_2, tmp_2(i)), PID_2, tmp_2(i)\}$ from the attacker; $K_3(i+j)$ is the $(i+j)$-th session key Service provider has.

It is obvious that all the above conditions will not be true. This is because the time slot between when Broker sent $\{B = H(K_2(i), PID_2, tmp_2(i)), PID_2, tmp_2(i)\}$ to Service requester and Service provider in the i-th session and when Service requester and Service provider received the replayed message $\{B = H(K_2(i), PID_2, tmp_2(i)), PID_2, tmp_2(i)\}$ from the attack in the $(i+j)$-th session is too big. This will result in $|tmp_{21}(i+j) - tmp_2(i)| > \hat{\theta}$ and $|tmp_{23}(i+j) - tmp_2(i)| > \hat{\theta}$. In addition, it is easy to see that

$$B \neq H(K_1(i+j), PID_2, tmp_2(i))$$

and

$$B \neq H(K_3(i+j), PID_2, tmp_2(i)).$$

Therefore, Service requester and Service provider will both not accept this session key confirmation initiated by the attacker. Hence, the cut-and-paste or replaying attacks will not work.

5.1.4 Security against existing attacks on braid group based cryptographic primitives

Braid group based cryptographic primitives were first proposed by Anshel et al. in [8, 49, 11, 12]. One of those primitives is a commutator key agreement protocol based on braid groups and their colored Burau representation [8]. Lee et al. proposed a summit set attack on Anshel et al.'s protocol [8, 15]. In fact, the protocols in [8] which were broken by Lee et al. were only some instances of the key agreement based on braid groups. That attack could not be applied to the generic construction of Anshel et al.'s protocol [8]. Therefore, that attack could not be applied to our three-party key establishment either. This is because (1) our key agreement is a generic construction; (2) our key agreement is based on non-commutative monoids; (3) the key agreement is one-time per key establishment.

In [44], Vasco et al. proposed two attacks on a public key cryptosystem based on free partially commutative monoids and groups. However, their attacks cannot be applied to our three-party key agreement protocol. This is because: On the one hand, their attacks are ciphertext only attacks and chosen ciphertext attacks while our protocol is key agreement. On the other hand, the monoids in our paper are assumed to be non-commutative.

Based on the above discussion, we can get the adversary cannot work out the shared secret key $k = H(\beta(a, f(b, a)) \cdot \theta(c, f(b, c)) \cdot \theta(a, f(c, a)))$ using existing attacks .

5.1.5 Security against spoofing attack

A spoofing attack is a situation in which one person or program successfully masquerades as another by falsifying data and thereby gaining an illegitimate advantage. In our scheme, spoofing attack is not possible due to the fact that there is no identity information transferred in cleartext [41, 42]. There is only a pseduo ID $PID_i (i = 1, 2, 3)$ which cannot be used by attackers to identify Service requester, Broker or Service provider. For example, if an attacker tries to make spoofing attack on Service requester, she may fake a pseudo ID PID_j. In order to pass the session key confirmation verifying made by Broker or Service provider, the attacker needs to find an element x_0 to calculate an A' so that

$$A' = H(x_0, PID_j, tmp_1) = H(K_1, PID_j, tmp_1).$$

However, the attacker will encounter the one-way hash function intractability. Therefore, she cannot make a successful spoofing attack on Service requester.

6 Conclusions

In this paper we proposed a secure web service scheme for service delivery. This protocol used two-way authentication and three-party key establishment to realize the security requirements for service delivery in network environments. Our method can help to reduce communication and computation overheads. This is because in our model there is only one shared key was established for Service requester, Broker and Service provider. After two-way authentication, they can use this key for the service delivery using symmetric encryption. Symmetric encryption is more efficient than asymmetric encryption with regard to computation cost. We have discussed the shared keys' security, the security against some existing attacks on algebraic method based key establishment and the spoofing resistance.

References

[1] Information on Web Service Security, http://www.oasis-open.org/committees/wss

[2] Information on XML digital signature, http://www.w3.org/Signature

[3] Information on XML Encryption, http://www.w3.org/Encryption

[4] Information on XKMS, http://www.w3.org/TR/xkms

[5] Information on SAML http://oasis-open.org/committees/security

[6] Information on XACML, http://www.oasis-open.org/committees/xacml/

[7] Information on the Liberty Alliance Project, http://www.projectliberty.org

[8] Anshel I., Anshel M., Fisher B., Goldfeld, D. New key agreement protocols in braid group cryptography, in CT-RSA 2001, Lecture Notes in Computer Science, vol. 2020, Springer, 2001.

[9] Doomun R., Implementing and securing web services in context-aware pervasive environment, Proceedings of the 3rd IEEE/IFIP International Conference in Central Asia on Internet, ICI 2007, 26-28 Sept. 2007, 1-6.

[10] Hwang T., Lee NY., Li CM., Ko M. and Chen YH., Two attacks on Neuman-Stubblebine authentication protocols. Information Processing Letters, 53:103-107, 1995.

[11] Czerwinski S. E. , Zhao B. Y., Hodes, T. D., Joseph A. D. , and Katz H., An Architecture for a Secure Service Discovery Service, in the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM 1999), pages 24-35, Seattle, WA USA, August 1999.

[12] Anshel I. , Anshel M. , Goldfeld D. , A method and apparatus for cryptographically secure algebraic key establishment protocols, International Patent Application Number: WO99/44324. US patent allowed. International Application Published Under the Patent Cooperation Treaty.
[13] Blaze M. , Feigenbaum J. , and Lacy J. , Decentralized Trust Management, in Proceedings of the IEEE Conference on Privacy and Security, 1996.
[14] Burrows M., Abadi M. and Needham R., A logic of authentication. Technical Report 39, Digital Systems Research Center, February 1989.
[15] Lee S.J., Lee E., Potential weaknesses of the commutator key agreement protocol based on braid groups, in Eurocrypt2002, Lecture Notes in Computer Science, vol. 2332, Springer, 2002.
[16] Shin S., Secure Web Services, JavaWorld.com, 2003.
[17] Hwang G.H. , Chang Y.H., Chang T.K., An Operational Model and Language Support for Securing Web Services, Proceedings of the IEEE International Conference on Web Services, ICWS 2007. 9-13 July 2007, 463-470.
[18] Thabet D., Lamia H,, Henda B., Toward Situational Secure Web Services Design Methods, in Proceedings of the IEEE International Conference on Web Services, 9-13 July 2007, 1179 - 1180.
[19] Subramoney K.P., Hancke G.P., A Secure Web Service for Electricity Prepayment Vending in South Africa: A Case Study and Industry Specification, Proceedings of the Second International Conference on Internet and Web Applications and Services, ICIW 07, 13-19 May 2007, 66-66.
[20] Neuman B. and Stubblebine S., A note on the use of timestamps as nonces. Operating Systems Review, 27(2):10-14, April 1993.
[21] Thabet D., Has L., Henda G., Situational Secure Web Services Design Methods, International Conference on Software Engineering Advances, 25-31 Aug. 2007, 2-2.
[22] Xu J., Yang E.Y., Bennett K.H., A practical approach to secure Web services, Ninth IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing, 24-26 April 2006 , 8-15.
[23] Mondejar R., Garcia P., Pairot C., Skarmeta A., Enabling Wide-Area Service Oriented Architecture through the p2pWeb Model, Proceedings of the 15th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, June 2006, 89-94.
[24] Clark J. and Jacob J., Freshness is not enough: note on trusted nonce generation and malicious principals.
[25] Helander J., Xiong Y., Secure Web services for low-cost devices, Eighth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing, 18-20 May 2005, 130-139.
[26] Weisstein E.W., Monoid at MathWorld, March 2008.
[27] Woo TY. and Lam S., A lesson on authentication protocol design. Operating Systems Review, pages 24-37, 1994.
[28] Yahalom R., Klein B., and Beth T., A distributed authentication perspective, in Proc. of the 1993 IEEE Symposium on Research in Security and Privacy, pages 150-164, May 1993.

[29] Roger Needham and Michael Schroeder. Using Encryption for Authentication in Large Networks of Computers. Communications of the ACM, 21(12), December 1978.
[30] Monoid at PlanetMath, accessed in Feb 2008.
[31] Zimmermann P., PGP User's Guide, MIT Press, Cambridge, 1994.
[32] Schneier, B., Applied Cryptography - Protocols, Algorithms, and Source Code in C, John Wiley & Sons, 1996.
[33] Lowe G., Some new attacks upon security protocols. In Proceedings of the Computer Security Foundations Workshop VIII. IEEE Computer Society Pres, 1996.
[34] Simmons G.J., Cryptanalysis and protocol failure, Communications of the ACM, 37(11), Nov 1994.
[35] Park C., Kurosawa K., Okamoto T., and Tsujii S., On key distribution and authentication in mobile radio networks, in Advances in Cryptology EuroCrypt '93, Lecture Notes in Computer Science, vol. 765, pp. 461-465, 1993.
[36] Guttmann E., Perkins C., Veizades J., Day M., Service Location Protocol, IETF Internet Draft, RFC 2608.
[37] Dorothy Denning and G. Sacco. Timestamps in Key Distribution Protocols. Communications of the ACM, 24(8), August 1981.
[38] Howie J.M., Fundamentals of Semigroup Theory (1995), Clarendon Press, Oxford ISBN 0-19-851194-9.
[39] Hubaux J. , Buttyan L., and Capkun S. , The quest for security in mobile ad hoc networks, in Proceedings of the ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC) 2001.
[40] Fox A., and Gribble S.D., Security on the move: indirect authentication using Kerberos, in the 2nd Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM 1996), pages 155-164, White Plains, NY USA, November 1996.
[41] Stallings W., Cryptography and Network Security: Principles and Practices, Fourth Edition, Prearson Education, 2006.
[42] Singhal A., Web Services Security: Challenges and Techniques, Proceedings of the Eighth IEEE International Workshop on Policies for Distributed Systems and Networks, 13-15 June 2007, 282-282.
[43] Otway D. and Rees D., Efficient and timely mutual authentication. Operating Systems Review, 21(1):8-10, January 1987.
[44] Vasco M.I., Steinwandt R., Pitfalls in public key cryptosystems based on free partially commutative monoids and groups, Applied Mathematics Letters 19 (2006) 1037-1041.
[45] Wilhelm U. G. , Staamann S., and Buttyan L., On the problem of trust in mobile agent systems, in IEEE Network and Distributed Systems Security Symposium 1998, pages 11- 13, San Diego, CA.
[46] Trabelsi S., Pazzaglia J.C., Roudier Y., Secure Web Service Discovery: Overcoming Challenges of Ubiquitous Computing, Proceedings of the 4th European Conference on Web Services, Dec. 2006, 35-43.

[47] Tatebayashi, M. N. Matsuzaki, and Newman D. B. J. , Key distribution protocol for digital mobile communication systems, in Advances in Cryptology-Crypto '89 Proceedings, Lecture Notes in Computer Science, vol. 435, 1989, pp. 324-334.
[48] Campo J.V., Pegueroles J., Soriano M., Providing Security Services in a Resource Discovery System, Journal of Networks, vol. 2, no. 1, 2007, 48-59.
[49] Anshel I., Anshel M., Goldfeld D., An algegraic method for public key cryptography, Mathematical Research Letters 6 (1999) 287-291.
[50] ISO/IEC. Information Technology - Security techniques - Entity Authentication Mechanisms Part 2: Entity authentication using symmetric techniques, 1993.
[51] www.bluetooth.org, The Official Bluetooth Membership website, accessed in February 2008.