Enhanced and Sustainable WS-Security Using the Participant Doman Name Token

Chi Po Cheong
University of Sussex, Brighton, United Kingdom
Email: webster@macau.ctm.net

Chris Chatwin and Rupert Young
University of Sussex, Brighton, United Kingdom
Email: {c.r.chatwin, r.c.d.young}@sussex.ac.uk

***Abstract*—This research proposes a new secure token profile for improving the existing Web Services security standards. It provides a new authentication mechanism. This additional level of security is important for the Service-Oriented Architecture (SOA), which is an architectural style that uses a set of principles and design rules to shape interacting applications and maintain interoperability. Web Services is one of the technologies to implement SOA and it can be implemented using Simple Object Access Protocol (SOAP). A SOAP-based Web Service relies on XML for its message format and common application layer protocols for message negotiation and transmission. However, it is a security challenge when a message is transmitted over the network, especially on the Internet. The Organization for Advancement of Structured Information Standards (OASIS) announced a set of Web Services Security standards that focus on two major areas. "Who" can use the Web Service and "What" are the permissions. However, the location or domain of the message sender is not authenticated. Therefore, a new secure token profile is proposed for enhancing existing Web Service security standards and illustrates its performance advantage over existing WS-Security standards.**

***Index Terms*—Web Services Security standard; OASIS Standard 1.1; Service Oriented Architecture; SOAP**

I. Introduction

The Service-Oriented Architecture (SOA) is a preferred design and architecture style, especially since integrated computing has become ubiquitous. "SOA is one of the preferred approaches for system design, development, and integration" [1]. One survey has shown that, "47.4% of respondents work in organizations where SOA projects are underway, and 30.9% have multiple SOA projects underway" [2]. Cloud computing, or Service-Oriented Architecture (SOA), is implemented by Web services and by the Simple Object Access Protocol (SOAP), which is widely used to implement Web services. A Web service is an XML-based platform-independent protocol. It is also programming language independent, and can be implemented using different programming languages. Therefore, in the Web services, XML Messaging is not only used for data exchange, but also in method discovery and invocation between Web service applications.

Service-orientation is a design paradigm comprised of a specific set of design principles. An SOA-based system is a kind of distributed system and similar to the Internet-based system. One of the major differences is the level of dependence on business logics in the design phase, which splits and groups the business logics into a loosely coupled set of services. The potential benefits of SOA are the services reuse, integration improvement, leveraging the legacy investment - that delivers the best of breed integration [3]. The classification of SOA-based systems is discussed in [4] and the examples of a Web service-based system are shown in [5], [6], and [7]. Security is one of the critical factors for the adopting Web service-based systems, especially for the requirement of sensitive data exchanges over the Internet between systems. The challenges of Web services security has been discussed in [8], the security concerns are: message alteration, confidentiality, man in the middle attacks, replays of message, etc. Therefore, a set of Web Services Security Specifications, which provide SOAP-based message integrity and message confidentiality, were approved by the OASIS Web Services Security (WS-Security) Technical Committee in 2006.

XML encryption and XML signature are defined in the WS-Security Core Specifications [9] and five other separate secure token profile specifications have been published by the WSS Technical Committee and all are collected in the WS-Security 1.1 OASIS standard. The five secure token profiles include the Username Token Profile [10], the X.509 Token Profile [11], the SAML Token Profile [12], the Kerberos Token Profile [13], and the Rights Expression Language (REL) Token Profile [14]. Each token profile has defined a standard set of Web Services Security extensions to provide particular security features that can coexist and cooperate with each other.

A. Motivation

The Simple Object Access Protocol (SOAP) is one of the methodologies to implement Service-Oriented Architecture (SOA). SOAP relies on Extensible Markup Language (XML) for its message format. A structured

doi:10.4304/jetwi.6.3.305-317

message is used to exchange data between the service consumer and the service provider in Web service architecture. Therefore, a secure Web service system not only focuses on the security of the system (e.g., hardware and software), but also on the confidentiality and integrity of the message exchange between the participants through the Internet, which is an unsafe public network. Both of which are two components of the CIA (Confidentiality, Integrity and Availability) Triad in the information security field.

SOAP-based Web Services use the WS-Security 1.1 OASIS standard, which is approved and published by Advancing Open standards for the Information Society (OASIS) to fulfill the security requirements in SOAP message exchanges among participants. OASIS WS-Security Core Specification 1.1 adopts the W3C XML standards, XML Encryption [15] and XML Signature [16] to provide message confidentially and message integrity. However, they are complex and produce a lot of overhead, especially the X.509-based encryption and signature. If many SOAP messages are exchanged between the service consumer and service provider, the overhead will be increased significantly. The OASIS WSS Technical Committee also provides security extensions for the Web service protocol stack to provide end-to-end message security. Five secure token profiles are the security extensions that are based on different existing security mechanisms and open standards, including PKI, X.509, Kerberos, or other algorithms, and these produce additional overheads for parsing and handling SOAP messages. The performance modeling of Web Services Security standards is discussed in [17] and other performance evaluation factors are discussed in [18] and [19].

According to the OASIS Web Services Security Specifications, "Who" can use the services, "What" is the authorization is tackled. However, the "Where" is not handled or supported in the OASIS standards. For instance, a user is allowed to use or invoke a web service and has been granted appropriate user rights. As a result, the user can use or invoke the services everywhere. It is a security hole if the services should only be provided for a particular enterprise, domain or location. Therefore, a new secure token is required to solve this issue and integrated with WS-Security standard. The location of service consumer and provider is taken into consideration for message exchange, parsing and handling. It can be used to reject invalid requests or responses, which come from unknown or fake domains before processing the OASIS Web Services secure token profiles. Therefore, a message receiver can save on processing resources and handle more valid Web service requests or responses.

B. Contributions of This Paper

The OASIS Web Services Security specifications and five secure token profiles have been presented in this paper. Although there are several security standards currently available and adopted in a SOAP-based message, they will produce a lot of overhead if one or more secure token profiles are used in the SOAP message. In this research, The newly designed secure token profile incorporates one of the most widely used Internet service information identity, the Domain Name Service (DNS).

A new secure token profile, named the Participant Domain Name Token (PDNT) [20] has been proposed and developed to enhance Web services security. The system architecture of the proposed token profile is also discussed. Moreover, the syntax of the proposed token, details of the processing rules and processing flow for adopting proposed token are described. A performance comparison between the proposed token and WSS specifications is presented in section 4. The "latency" or "response time" performance metric is used for performance comparison. Using the proposed token profile, the location or domain of the sender can be authenticated, which ultimately reduces the demand on system resources by blocking fake requests.

II. Related Work

The Web service architecture basically consists of three roles, which are as follows: service requestor, service provider, and service registry. The relationship between these three roles is that a service provider defines and publishes a service description to a service requestor or service registry; the service requestor finds a desired service and retrieves the service description locally or from the service registry; and the service requestor uses the service description to bind with and invoke the service implementation from the service provider directly. Two major classes of Web services are defined by W3C. These are the Representation State Transfer (REST)-compliant Web services and the arbitrary Web services, which are the techonology for "Big" Web services. REST is neither a standard nor a protocol. It is an architectural style, like client/server architecture. The arbitrary, or "Big" Web services, is a web service that is implemented by Simple Object Access Protocol (SOAP). However, this paper will focus on the SOAP-based Web services and security.

A. SOAP Message

Web Services is a kind of distributed system and it provides APIs that can be invoked by other applications or systems on computer network or Internet. Currently, SOAP is the most common way to implement Web Services and it is a core component of the Web Services architecture. The SOAP is a lightweight protocol intended for exchanging structured information in a decentralized, distributed environment. On other words, it is a specification for exchanging structured messages between two computer systems via common communication protocols. It uses XML technologies to define an extensible messaging framework by providing a message construct that can be exchanged over a variety of underlying protocols. A SOAP message is encoded using XML and uses SOAP namespace and a SOAP encoding style. Equation (1) shows the components of SOAP.

$$\text{SOAP} = \text{XML Message} + \text{Communication Protocol} \quad (1)$$

In order to pass through the firewall to support interoperable machine-to-machine interaction over the Internet, usually Hyper Text Transfer Protocol (HTTP) or Simple Mail Transfer Protocol (SMTP) is adopted as the communication protocol. There are two different layers to secure the SOAP, which are message level security and transport level security. The transport level security uses the layer 3 or layer 4 protocol to protect the data, such as IPSec [21], SSL, etc. However, the layer 3 or 4 security mechanisms will be broken if there is an intermediary SOAP node between two end points. The message level security works in layer 7, which is the application layer. Therefore, the message level security is a proper method to protect the message from end to end and the WSS is adopted to protect SOAP messages.

B. Web Services Security Standards

OASIS (Organization for the Advancement of Structured Information Standards) is a not-for-profit consortium that drives the development, convergence, and adoption of open standards for the global information society. OASIS announced a set of Web Services Security Standards, named the WS-Security 1.1 OASIS standard, which were approved by the Web Service Security technical committee on November 28, 2006. The WS-Security 1.1 OASIS standard consists of one specification and six token profiles.

The WS-Security core specification 1.1 is also known as the, "Web Services Security: SOAP Message Security 1.1," which was written by Lawrence K. et al. It utilizes two open W3C-approved standards, XML Encryption, and an XML Signature to provide message integrity and confidentiality. Fig. 1 shows the conceptual representation of the WS-Security core specification. It adds a security header <wsse:Security> into the SOAP message header <s11:Header> to define a security framework and it includes extensibility mechanisms. It also defines different tags to contain security information for an intended recipient. For instance, <wsse:Security> is a tag used to define a security header block, which attaches security-related information. A recipient will parse the message security information and will obtain the details of the processing rules based on the WS-Security core specification.

Moreover, the WS-Security core specification is designed to be extensible. It describes a mechanism for associating different types of security tokens with message content. Therefore, a range of security protocols or user defined security protocols can work with a SOAP message using this specification, instead of fixed or limited security protocols.

C. OASIS WS-Security Token Profiles

Since April 2002, the Advancing Open Standards for the Information Society (OASIS) - Web Services Security Technical Committee (WSS TC) continues to form the necessary technical foundation for the higher-level security services, which are to be defined in other specifications. The OASIS Web Services Security TC collects existing XML-based security standards or other open standards, such as XML encryption, XML signatures, X.509 certificates, Security Assertion Markup Language (SAML) [22], the Kerberos protocol [23], and the Right Expression Language [24], to provide SOAP message integrity and confidentiality. It does so in order to fulfill participants' authentication, authorization, and permission for message exchanges. All are defined in five

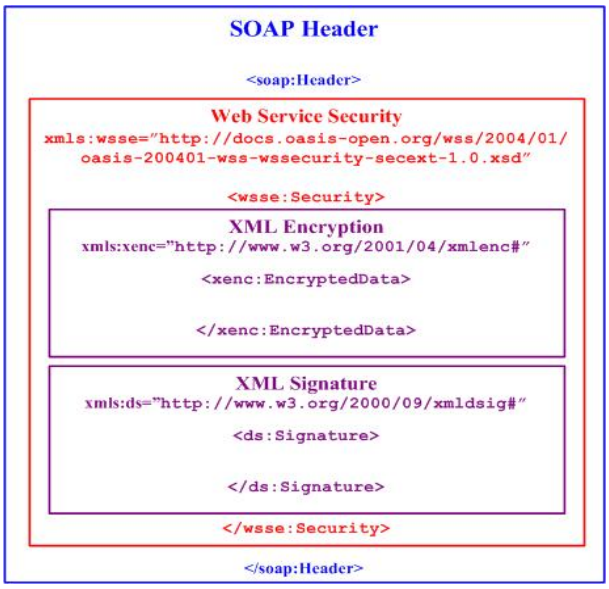

Figure 1. The conceptual representation of the WSS core specification.

separate secure token profile specifications and are collected in the WS-Security 1.1 OASIS.

Because the WS-Security core specification is designed to be extensible, other security mechanisms can be applied to a SOAP message and defined in the security header. Five security-related token profiles have been approved by OASIS and work using the WS-Security core specification. These are as follows: (1) the Username Token Profile specifies how a service consumer supplies a user name and a corresponding password to a service provider. (2) The X.509 Token Profile describes how to use the X.509 authentication framework with the WS-Security core specification. (3) the security information sharing mechanism is described in the SAML token profile, such as authentication and authorization data between participants using SAML through the XML document (4) The Kerberos Token Profile shows how to use the Kerberos protocol in the WS-Security core specification. By adopting the Kerberos protocol in a Web service application a client can use the shared secret to obtain the Ticket Granting Service (TGS) session key for further communication with the TGS. (5) The Rights Expression Language (REL) Token Profile describes the use of ISO/IEC 21000-5 Rights Expression with the WS-Security core specification. The REL token profile describes a permission to define who can perform specific actions with digital information under certain conditions. The permission can be granted to a user, an organization, or other entities.

The processing model for WS-Security with all five token profiles is no different from other security tokens defined in the WS-Security core specification. The message processor or handler must do the token validation and follow the processing rules, which are defined in related protocol specifications and these are not presented in each token profile. All OASIS WSS token profiles involve cryptology, encryption, signature mechanisms, or a Public Key Infrastructure (PKI) to provide authentication and authorization features. The five secure token profiles focus on two major areas. Who can use the Web services and what Web Services and methods that they can use and invoke. The classification of five WSS token profiles is shown in Table 1. However, the location or domain of Web service participants is not handled and is not verified in existing secure token profiles. XML security standards also do not rely on location. Therefore, a new secure token profile is proposed in this paper to provide additional security that can be used by Web services.

D. Web Services Security Technologies

Although the WS-Security 1.1 OASIS standard has been published to protect the SOAP message between two end-points, there is still significant research that needs to be conducted in regards to Web Services security. Authentication and authorization are the mechanisms to determine "who you are" and "what you are authorized to do." Both mechanisms are adopted in many information systems and the difference between old and new technology is the implementation mechanism. The authentication process may be as simple as providing a username and password to an authenticating system, or as complicated as using PKI authentication or a Kerberos authentication system. Research that has been reported in [25], proposes a new secure token to extend the WS-security for implementing existing secure protocols such as ISO9798, Kerberos, etc. Other research [26] designs and implements an Authorization Filter to provide authentication and authorization features between a client and a SOAP gateway for all the SOAP requests. It also describes new tokens or tags to provide the security features such as, "userid," "passwordhash," "role," etc. However, it is not a new security mechanism that can protect a SOAP message. Authorization is the next process after authentication to determine which web services and methods can be invoked by an authenticating user or a participant. Group-based access control and role-based access control [27] are mostly used to control the level of access rights within a system. Other similar models include task-based access [28] and provision-based access control [29]. These models are static and require a pre-defined access level for each participant.

TABLE I.
CLASSIFICATION OF WS-SECURITY TOKEN PROFILES

Area	Secure Token Profile
Web Service Consumer Authentication (Who)	Username Token Profile X.509 Certificate Profile Kerberos Token Profile Security Assertion Markup Language (SAML) Token Profile
Web Service Consumer Authorization (What)	Rights Expression Language Token Profile Security Assertion Markup Language (SAML) Token Profile

In order to tackle this issue, some dynamic approaches have been proposed, an example of which is reported in [30]. Another approach is using data mining to predict Web attacks, an example of which is given in [31]. In the dynamic approach, the policies are dynamically assigned by a reasoner, which is based on rules and ontology. The data mining approach uses different SOAP message attributes, such as, message size and parsing time, to predict whether a SOAP message is valid. However, both approaches are complex and may be false positive and false negative. Moreover, there are no location-based mechanisms for the location authentication. Therefore, a new secure token profile and mechanism is proposed in this paper.

III. PARTICIPANT DOMAIN NAME TOKEN PROFILE (PDNT)

A domain, which is a hierarchical distributed naming system that works like a tree structure, is used to uniquely identify Internet resources such as a website, an email system, etc. Each node in the domain naming tree has one or more zone files, which are used to store resource records, and is managed by a domain name server. The domain name is composed of human readable characters that can be translated into the numerical identifiers, such as the domain name can be translated to an Internet Protocol address by the Domain Name System (DNS). The DNS is implemented in the client-server model and is maintained by a distributed database system. The IP address is used by a machine and it can be used to locate an Internet service ub the worldwide web. A DNS client issues a DNS query to a DNS server or a DNS resolver by using the User Datagram Protocol (UDP). The DNS resolver listens on port number 53 to handle the DNS query and returns a Resource Record (RR) to the client. For example, an Address Record (A Record) [32] is used to resolve an IP address by a given host name.

A Web service consumer, which is an application or software module, sends a service request or invokes a service method by using HTTP protocol. An example is of which is shown in Fig. 2. The HTTP request contains not only the requested information but also the necessary information that is used for responding back to the requester (i.e., the requester's IP address and port number, "192.168.1.1" and "8888" in this example). Therefore, the requester IP address cannot be faked as the requester wants the result to be sent back. The mapping between the IP address and domain name is stored in the DNS, which is hosted by the Internet Service Provider (ISP). Usually, the ISP is a trusted local telecommunications company. Therefore, the DNS records hosted by the ISP are trusted and reliable. The proposed token profile uses one of the DNS resource records to validate the location or domain of participants.

```
Content-Type: application/soap+xml;charset=UTF-8
Content-Length: 44501
Host: 192.168.1.1:8888
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.2 (java 1.5)

<SOAP MESSAGE>
```

Figure 2. An example of a SOAP message using HTTP

A. Service Record (SRV Record)

There are two major namespaces on the Internet. The domain name and the Internet Protocol address spaces. The mapping between these two namespaces is stored in the zone file or database of Domain Name System (DNS), which is address record. Moreover, there are many types of record resources stored in the DNS. Each type of resource record is a defined RFC and has a particular task. For instance, for the "A Record," the type code is "1," which is used to return a corresponding 32-bit IPv4 address by giving a hostname.

A service resource record (SRV record) [33] is one of the DNS resource records and its type code is 33. It is used to define the location of the servers for specified services. For instance, the Session Initiation Protocol (SIP) uses the SRV record, which is stored in the DNS to find or point to a SIP server, which is listening on TCP port 5060 for SIP services. The format of the SRV record is:

_Service._Proto.Name TTL Class SRV Priority Weight Port Target (2)

Where:

- Service is service symbolic name.
- Proto is the symbolic name of the desired protocol.
- Name is the domain this RR refers to.
- TTL is the maximum amount of time that a DNS server should take to cache the record.
- Class is a standard DNS class field.
- Priority is the priority of this target host.
- Weight is a server selection mechanism.
- Port is the port on this target host of this service. The port number rage is 1-65535
- Target is the domain name of the target host. There must be one or more addresses (A Record) for this name.

The SRV record works with the "A Record" because the target field defined in the SRV record points to an address record. Therefore, the corresponding hostname record must be defined in the DNS. The following DNS record is an example of an SRV record to define an ftp service in mydomain.com:

_ftp._tcp.mydomain.com 300 IN SRV 0 1 23 ftpservice.mydomain.com (3)

According to the above SRV record, a user or a system can obtain a corresponding address record, which is pointing to a FTP service. A domain can use the SRV record to declare which services they are providing and where it can use it. For example: the type of protocol, port number, and corresponding IP address.

The proposed token profile utilizes the SRV record and makes a little change to the meaning of the target field to be a self-defined host instead of the remote host. Although the definition of the target field of RFC 2782 has a minor change, it will not affect the existing usage because of different service names. The proposed token profile can be implemented by using the following SRV record defined in the DNS Server.

_pdn._tcp.sussex.ac.uk 300 IN SRV 1 1 8080 soap.sussex.ac.uk (4)

Where:

- _pdn : service name for proposed token.
- _tcp : using TCP protocol.
- .sussex.ac.uk : domain of the owner.
- 300 : time to live, 300 seconds.
- IN : Internet class.
- SRV : Service record.
- : (0-65535). Lowest value represents the highest priority.
- : weight.
- 8080 : port number, self defined.
- soap.sussex.ac.uk: the name of the host that will provide this service. However, it has changed the original definition of RFC 2782 to message sender.

In this example, the Web service participants use the following rule or format to obtain SRV resource records. If the result is not saved in local DNS cache before, a DNS client will query a DNS server to obtain the result and keep it in the local DNS cache to improve the performance the next time.

QNAME = _pdn_tcp.sussex.ac.uk, QCLASS=IN, QTYPE=SRV (5)

Based on SRV resource records, a service provider can validate a message sent by a service consumer or vice versa. However, the message receiver must ask the message sender to register the SRV resource records with the local ISP when the proposed token is implemented.

B. Participant Domain Name Element

The Participant Domain Name Token Profile (PDNT) is used with WSS: SOAP Message Security specification (WSS). It describes how a participant supplies a domain name token as a means of identifying the participant by domain name to authenticate the participant location. In order to use PDNT in a SOAP message, a new

<wsse:ParticipantDominNameToken> element is proposed and introduced in the WSS core specification. The syntax for this is shown in Fig. 3. Within the <wsse:ParticipantDominNameToken> element, a <wsse:DomainName> element is specified. It contains a participant or a message sender domain name with the required format and it will be translated to the IP address through the SRV resource record, which is stored in the

```
<soap:Header>
    ...
    <wsse:Security>
        <pdn:ParticipantDomainName>
            <pdn:DomainName>
                _pdn_tcp.sussex.ac.uk
            </pdn:DomainName>
        </pdn:ParticipantDomainName >
    </wsse:Security>
    ...
</soap:Header>
```

Figure 3. An example of the PDNT

DNS. The details of how to use the proposed token are illustrated in the next section. The proposed token can be defined in a new namespace URI or it can use the same namespace as the WSS core specification if the schema file includes the proposed elements. The schema file for a proposed token whose prefix is "pdn" and whose namespace is http://schema.sussex.au.uk/participant-domain-name-token-profile.xsd has been designed and is shown in Fig. 4.

```
<xsd:complexType
name="ParticipantDomainNameTokenType">
    <xsd:annotation>
        <xsd:documentation>
            This type represents a participant domain name token
        </xsd:documentation>
    </xsd:annotation>
    <xsd:complexContent>
        <xsd:sequence>
            <xsd:element name="DomainName"
            type="xsd:string" minOccurs="1" maxOccurs="1" />
            <xsd:any processContents="lax" minOccurs="0"
            maxOccurs="unbounded" />
        </xsd:sequence>
    </xsd:complexContent>
</xsd:complexType>

<xsd:element name="ParticipantDomainNameToken"
    type="wsse:ParticipantDomainNameTokenType">
    <xsd:annotation>
        <xsd:documentation>
            ……..
        </xsd:documentation>
    </xsd:annotation>
</xsd:element>
```

Figure 4. A schema file PDNT

C. The Processing Rules of PDNT

The PDNT works with DNS records, which are stored in the DNS server. Fig. 5 shows the processing flow of the proposed token profile. A service consumer sends an HTTP request, which contains a SOAP message for the service provider. The service provider receives the HTTP request and parses the SOAP message. A participant domain name can be obtained from the <wsse:ParticipantDomainName> element. The service provider sends a SRV resource record query with the domain name to a local DNS server. If the corresponding resource record cannot be found in the local ISP, then the local ISP will do the recursive query worldwide. A host name that is defined in the target field is returned and a second DNS query is executed to lookup the IP address. A corresponding IP address can be resolved by the local ISP and is returned to the service provider. The location of a service consumer or message sender can be authenticated if the equation 6 is valid. The results will be returned to the service consumer if PDNT and other secure token profiles are validated.

$$[\text{IP Address}]_{\text{Container}} = [\text{IP Address}]_{\text{SA}} \quad (6)$$

Where:

- Container is a Web service container or traditional Web server container and in JavaServer Pages (JSP) language it can obtain the message sender IP address by executing request.getRemoteAddr().
- SA is a two-step procedure. First, it looks up a SRV resource record by giving a standard SRV query string. Second, based on the SRV record, it can obtain a canonical hostname and acquire an IP address by querying the address record in DNS.

To eliminate the overhead for processing an unknown or a fake request, the PDNT is processed before other secure token profiles or other security standards such as X.509 Token Profile, Kerberos Token Profile, etc. In order to support high availability, load balancing and services backup, the DNS server may return more than one SRV resource record. The token processor uses priority and weight fields to determine the precedence for use of the record's data. One of the objectives of PDNT is to reject a fake request as fast as possible, which will reduce the processing resources required to handle illegal requests.

One of the objectives of the proposed token is to reject a fake request as fast as possible, which will reduce the processing resources required to handle illegal requests. Therefore, the sequence of processing the proposed token with other secure token profiles is:

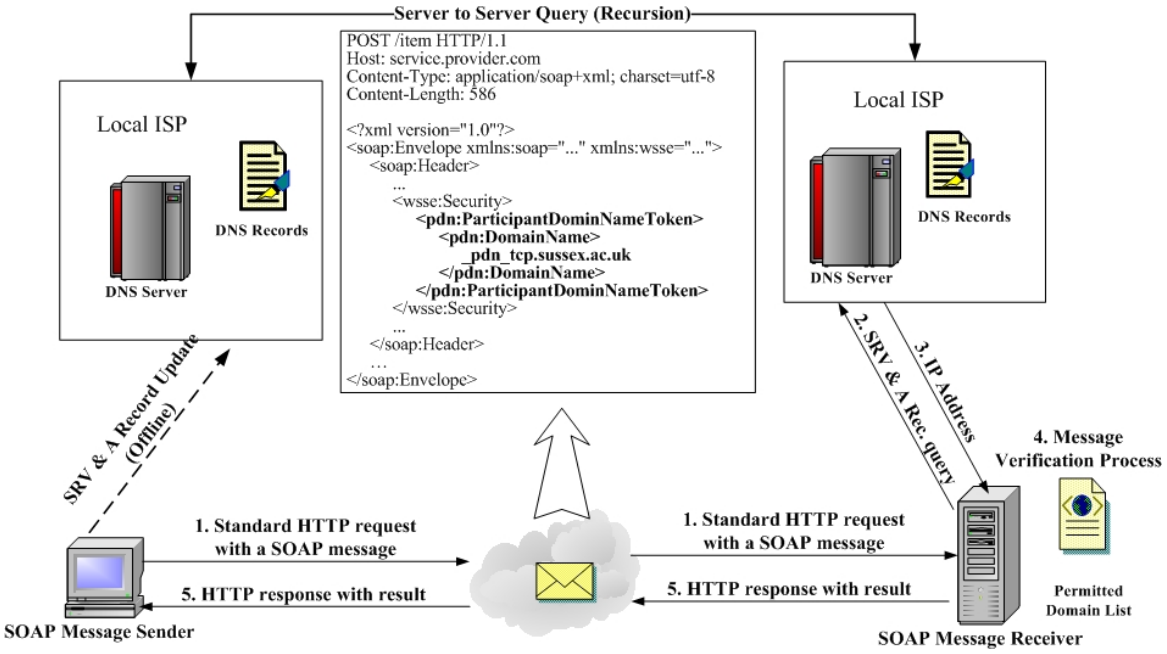

Figure 5. The processing flow of the proposed token

A HTTP container receives a SOAP message, and then validates and parses the message.

1. A SOAP message parser obtains a Web services security element <wsse:Security> from the SOAP message header <soap:Header>. If the message does not contain a <pdn:ParticipantDomainName> element inside the <wsse:Security> header, the HTTP container should reject the message.
2. A SOAP message handler processes the <pdn:ParticipantDomainName> element block and obtains the sender domain name with the service name contained in the <pdn:DomainName> element. A DNS query will be issued to the local DNS server. A corresponding IP address will be returned from the DNS server to the HTTP container. The DNS may do the recursion lookup in the worldwide DNS system for acquiring the corresponding address record.
3. The HTTP container system compares the sender IP address with the address record stored in the DNS by following the proposed token processing rules. If it is valid, it processes other secure token profiles if they exist. Otherwise, it rejects the message.
4. If the message passed all secure token profiles, a corresponding result will be sent back to the message sender.

In order to handle a SOAP message, which does not use or implement the proposed token, a permitted domain list feature is included in the system. In the other words, a permitted domain list is a kind of whitelist, which is used to bypass the processing rule of the PDNT. The whitelist can use the IP address or an address record (A Record), which is stored in the DNS server to control who is allowed to use the Web services. If the IP address is adopted, it will use it to compare the remote IP address directly. Otherwise, the IP resolve step is processed before the comparison. The whitelist will be stored in a secure location, such as in a private database or a file, in a private folder.

The location-based validation process utilizes one of the existing well-known Internet infrastructures, such as the Domain Name System. Therefore, it can be trusted and is reliable. In order to parse and verify the proposed token before any other security specifications, arbitrary data encryption cannot be applied to the proposed token profile. This means that the proposed token should be shown in plain text or be Base64 encoded. The proposed token profile must also be processed before other secure token profiles because it uses less processing time to validate a message. The performance evaluation of token profiles is illustrated in the next sections.

D. Implementation

The proposed token profile is implemented in Java language and uses "java.xml.soap.*" Java Achieve (JAR) to parse a SOAP message. Other libraries such as "org.xbill.DNS.*" are used for SRV and address resource record resolution. These libraries act as a DNS client to issue a query to the DNS system using UDP protocol at port 53. Although the performance of DNS request and response is affected by many factors, such as network layer application layer, the invention of the DNS local cache tackles this issue. A new java class to handle the proposed token, which is named the "ParticipantDomainNameTokenHandler," has been designed and developed. A Java class constructor and a validation method for the proposed token are shown in Fig. 6 and an example of usage is shown in Fig. 7.

```
public static void main(String[] args) {
   MessageFactory factory = MessageFactory.newInstance();
   MimeHeaders mimeHeaders = new MimeHeaders();
   mimeHeaders.addHeader
                        ("Content-Type","text/xml;charset=UTF-8");
   SOAPMessagesoapMessage=factory.createMessage(mimeHeaders
, MessageUtil.readFileToInputStream("soapMessage.xml"));
   ParticipantDomainNameTokenHandler tokenHandler=new
   ParticipantDomainNameTokenHandler(soapMessage);
   tokenHandler.validate(senderIP);
}
```

Figure 6. The class constructor and a validation method of the PDNT

```
message){
   try{
            soapMessage=message;
            this.soapPart = soapMessage.getSOAPPart();
            this.soapEnvelope = soapPart.getEnvelope();
            this.soapHeader = soapEnvelope.getHeader();
            this.soapBody=soapEnvelope.getBody();
            if (soapHeader==null){
                     soapHeader=

   MessageUtil.createSecureHeader(soapEnvelope);                    }
   }catch(Exception e){
            e.printStackTrace();
   }
}

public boolean validate(String senderIP){
   String lookupService=getLookupService();
   List <String>
   ipList=DNSClient.getAddressRecord(lookupService);
   for (int i=0;i<ipList.size();i++){
            if (ipList.get(i).equals(senderIP))
                     return true;
   }
   return false;
}
```

Figure 7. Sample program using the PDNT

An instance, which is created from the class of the ParticipantDomainNameToken, has been tested on different sizes of SOAP messages and it can work with other secure token profiles. All of the related classes and libraries are packed into a JAR file, which can be used and plugged into different Web service containers.

E. Performance Modeling

Security is one of the successful factors for the use of Web Services on the Internet. Many Web Services security standards are designed and proposed, such as OASIS Web Services Security. It provides end-to-end message security properties including integrity, confidentiality and authentication. However, performance is another vital factor for evaluating a security standard. The processing time is a key indicator for measuring the performance of a new or existing security specification. As more security mechanisms are adopted, additional performance overheads will be added to process the Web Services using CPU processing time, large messages will consume channel bandwidth.

In order to compare the performance of the proposed token profile and other secure token profiles, the performance measurements have been defined. The most commonly used performance metrics are the response time (R) and throughput (X) [34]. This paper uses response time or latency time, which is the round-trip time of a message between a sender and a receiver, as a performance metric. The metric can be used to evaluate the proposed token profile and can be compared with other secure token profiles. The latency is defined by the following:

$$\text{Latency} = T_{[\text{Network Layer for SOAP request}]} + T_{[\text{Application Layer}]} + T_{[\text{Network Layer for SOAP response}]} \quad (7)$$

Where:

- $T_{[\text{Network Layer for SOAP request}]}$ and $T_{[\text{Network Layer for SOAP request}]}$ is the total transfer time of a message between the sender and receiver over the network. It consists of many factors, such as the number of network devices involved, the total delay time in each router or switch, the distance between sender and receiver, etc.
- $T_{[\text{Application Layer}]}$ is the total processing time spent in the application level, which includes message encoding / decoding, message parsing, token profiles processing time, total time spent in business logic, database processing time, etc.

However, the total time spent in the network layer is difficult to evaluate over the Internet because it depends on the quality of network transmission and the capacity of the network devices. Therefore, this paper only focuses on the time spent in the application layer, which is defined as:

$$T_{[\text{Application Layer}]} = T_{[\text{parsing}]} + T_{[\text{token profile}]} + T_{[\text{logic}]} + T_{[\text{database}]} \quad (8)$$

Where:

- $T_{[\text{parsing}]}$ is the total time spent in parsing an input stream to an XML document and converts it to a W3C Document Object Model (DOM), which allows a program to access and manipulate the content of the document.
- $T_{[\text{token profile}]}$ is the total time spent in handling different secure token profiles and corresponding mechanisms, such as XML Encryption, XML Signature, etc.
- $T_{[\text{logic}]}$ represents the time spent in business logic.
- $T_{[\text{database}]}$ is the total time spent in Data Manipulation Language (DML). Most Web service applications work with a database to query database tables, in order to get the result back and to return it to the requesters.

The proposed token profile is used by a message receiver to authenticate the location of a sender. Other secure token profiles are also used to validate a SOAP message by different mechanisms. In order to compare the performance between proposed token and other existing token profiles, the T[logic] and T[database]

factors can be ignored in the performance evaluation. It means that only parsing time and security token profile handling time are taken into account.

IV. Experimental Results And Analysis

The latency or response time performance metric is used to compare the proposed token profile with three other WS-Security tokens, which include the Username Token profile, XML Encryption, and XML Signature. All test cases are tested with different message sizes. Moreover, the message size is also compared when adopting the proposed token profile with the three other WSS tokens. A millisecond timescale is used to compute the latency for each round-trip time between the message sender and receiver.

A. Evaluation Method and Assumptions

In the performance evaluation, all of the measurements are made with identical equipment and in the same environment. In order to eliminate the network and other configuration issues, the Web service consumer and provider are running on an Intel Core2 Duo E8500 computer, Windows XP Professional with SP3, with 4GB RAM, and the computer is restarted before each test case to ensure that the starting conditions are the same. Both the Web services consumer and the Web services provider are developed using the Java programming language. The Java 2 Platform Enterprise Edition version 1.6.0_21-b07 and SOAP with API Attachments for JAVA (SAAJ) are used for the development and testing environment. A DNS server is installed and run on the same computer. The DNS local cache is also enabled to improve the performance of the DNS resolution process, which is used by the Participant Domain Name Token. A pure HTTP server has been developed using the Java language and Apache HTTP client [35]. JAR is used to develop an HTTP client, which is used to send a SOAP message to the HTTP server. Five java classes have also been developed, as shown in the following java classes listing:

- SOAPEncyrption.java
- SOAPDecryption.java
- SOAPSignature.java
- UserNameToken.java
- PaticipantDomainNameTokenHander.java

All classes are plugged into the pure HTTP server to handle SOAP messages with different types of WSS. The RSA 2048-bit is used for the asymmetric key algorithm and the AES 128-bit is used for the symmetric key algorithm. SHA is used to create a message digest, which is used in the XML encryption, decryption, signature, and password digest. Latency can be evaluated from the total time spent by the HTTP client when it sends a SOAP request with secure token profiles to an HTTP server, which sends the information back to the HTTP client. In order to evaluate the performance for different sizes of SOAP messages, 100, 200, 300, 400, and 500 employee records are included in the message, respectively. An example of an employee record is shown in Fig. 8. Each SOAP message contains the proposed token and one of the WS-Security tokens. This means that the message size is identical during the performance comparison for the proposed token and one of the WSS tokens. Each test case is repeated 100 times to obtain the average latency. The average latency time is used to make a comparison between the proposed token and WS-Security tokens.

```
<employee>
    <firstName>David</firstName>
    <lastName>Ho</lastName>
    <gender>M</gender>
    <nationality>British</nationality>
    <dateOfBirth>1958-07-01</dateOfBirth>
    <phone>12-543325-8</phone>
    <maritalStatus>Married</maritalStatus>
    <address>No. 7 A Road</address>
    <city>Hong Kong</city>
    <country>China</country>
    <email>david.ho@myService.org</email>
    <empNo>101</empNo>
    <title>Manager</title>
    <department>Accounting</department>
    <hireDate>2005-01-05</hireDate>
</employee>
```

Figure 8. Example of an employee record for test cases

B. The Design of the Test Cases

According to the WS-Security 1.1 OASIS standard, there are five secure token profiles and one core specification. Because some of the secure token profiles are using a similar mechanism and algorithm, not all of the secure token profiles are evaluated. For instance, the X.509 token profile uses a digital signature to verify an X.509 certificate. This is the same as with the XML signature token. Therefore, four test cases have been selected, designed, and evaluated as described by the following test case list:

Test Case 1:
Proposed Token vs. Username Token

Test Case 2:
Proposed Token vs. XML Decryption Token

Test Case 3:
Proposed Token vs. XML Signature Token

Test Case 4:
Proposed Token vs. XML Encryption with Signature Token

Each test case has been divided into three sub-test cases, which contain (a) the proposed token only; (b) one of the above WSS tokens; and (c) the proposed token with one of the above WSS tokens. Each sub-test case is processed a hundred times to acquire the average response time for each sub-test case of each record size.

C. Message Size of Each Secure Token

The size of a SOAP message depends on which secure tokens are used. Different secure token profiles have different element blocks, types, attributes, and syntax. For instance, the Username Token profile has less than 10 elements, which is less than other WS-Security secure token profiles. Therefore, the message size of a SOAP

message header using the Username Token profile is less than when using other WS-Security token profiles. A large SOAP message size will increase the streaming time, network transfer time, and parsing time for the message sender and receiver. Table 2 shows the message size and the percentage size increase for each secure token profile. These are the minimum elements requirement for using each token profile, which means that only compulsory elements are used in each test case and optional elements are not considered.

Based on the results in Table 2, the message size of the Participant Domain Name Token (PDNT) profile has the minimum number of elements and the Encryption Token Profile has the maximum number of elements of the WS-Security secure token profiles. By using WS-Security Token Profiles, the size of a SOAP message is increased by 1.22% - 39.17% when the message contains 100 employee records. However, it only increases by 0.37% if only the PDNT is adopted. The message size overhead for using PDNT with each WS-Security Token Profile is only increased by 0.34%. This is a major advantage of adopting the PDNT because the message size is much less than adopting other WS-Security token profiles and the overhead adds only a very small increase in message size. With PDNT, the more employee records that are included in a SOAP message, the smaller the relative percentage increase of message size and the overhead is decreased compared to other WS-security token profiles.

TABLE II.
PERCENTAGE INCREASE OF MESSAGE SIZE BETWEEN NON-WSS AND EACH SECURE TOKEN PROFILE.

Number of Employee Records	100	200	300	500
PDNT	0.37%	0.19%	0.12%	0.07%
Username Token	1.22%	0.61%	0.41%	0.25%
PDNT with Username Token	1.56%	0.78%	0.52%	0.31%
Signature Token	2.26%	1.13%	0.76%	0.46%
PDNT with Signature Token	2.59%	1.30%	0.87%	0.52%
Encryption Token	39.17%	38.01%	37.62%	37.31%
PDNT with Encryption Token	39.51%	38.18%	37.73%	37.38%
Encryption with Signature Token	41.39%	39.13%	38.37%	37.76%
PDNT, Encryption with Signature Token	41.73%	39.29%	38.48%	37.83%

D. The Results of Test Case 1

Fig. 9 shows the results of comparing the proposed token and the Username Token. A message receiver only uses 7.53 milliseconds to process the Participant Domain Name Token (PDNT), which is less than the message receiver, which uses 8.48 milliseconds to process the Username Tokens when the message contains 100 employee records. This shows that PDNT is 12.74% faster in rejecting a Web service request if it comes from an invalid location or domain. The third sub-test case is to evaluate the performance when both tokens need to be processed. When a message passes the PDNT validation, it also requires other tokens to be pricessed (the Username Tokens in this case). There is a 0.37% processing overhead to process both tokens, which is a very small increase given the additional security that is provided.

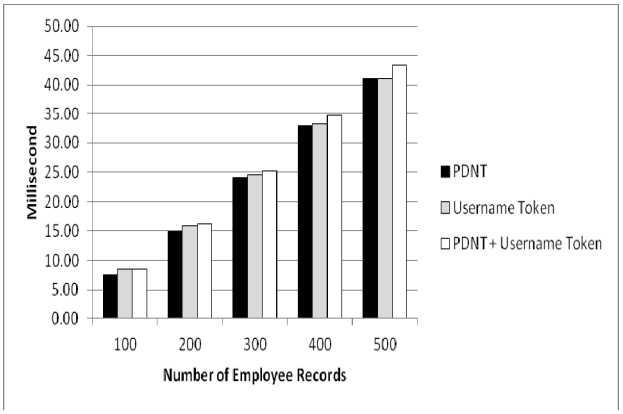

Figure 9. Latency in milliseconds for Test Case 1

E. The Results of Test Case 2

In Test Case 2, the paper assumes that the decryption mechanism defined in the XML encryption standard is used to validate a SOAP message if there are no other authentication methods adopted. As shown in Fig. 10, significant performance gains can be realized by using the proposed token. The processing time of adopting PDNT is more than 8 times faster than the decryption process when a hundred employee records are contained in a SOAP message. A message receiver can reject a message that is sent from an unauthorized location as fast as the XML decryption processing can be completed. The processing time is increased by only 0.49% to process both tokens when a SOAP message contains 100 employee records. Therefore, the overhead is not significant if both tokens are processed.

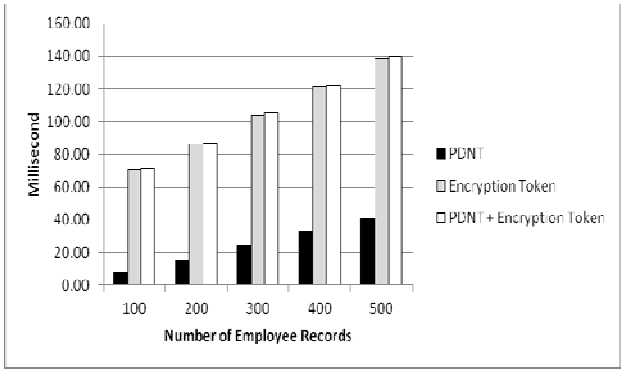

Figure10. Latency in milliseconds for Test Case 2

F. The Results of Test Case 3

An XML signature is used to ensure a SOAP message that is sent from a known sender. Fig. 11 shows the performance results of adopting the PDNT, signature token, and a PDNT with a signature token. According to the results of Test Case 3, it is 50.39% faster on average if PDNT is adopted, as compared to the signature token. The overhead of adopting both tokens is 0.43% on average. Therefore, adopting the proposed token has a performance advantage and it can refuse an invalid SOAP message faster.

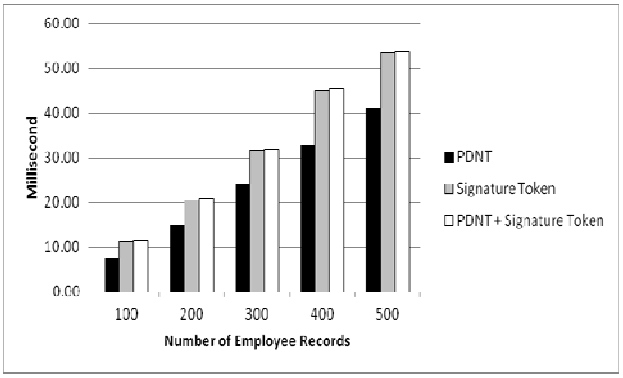

Figure11. Latency in milliseconds for Test Case 3

G. The Results of Test Case 4

Some Web service applications require more security mechanisms to protect the information between sender and receiver. Therefore, more than one secure token profile is adopted in a SOAP message. In Test Case 4, PDNT, XML encryption, and XML signature tokens are used in a single SOAP message. As in the previous test cases, an identical message header is used to eliminate the different message header size issue. The results for Test Case 4 are shown in Fig. 12. The results show that the PDNT is faster by 257.12% to 884.62% on average if only PDNT is adopted, as compared to an encryption with a signature token. The overhead of adopting all three tokens is 0.11% to 0.78% on average. Therefore, with less than 1% overhead, additional security can be gained by using PDNT.

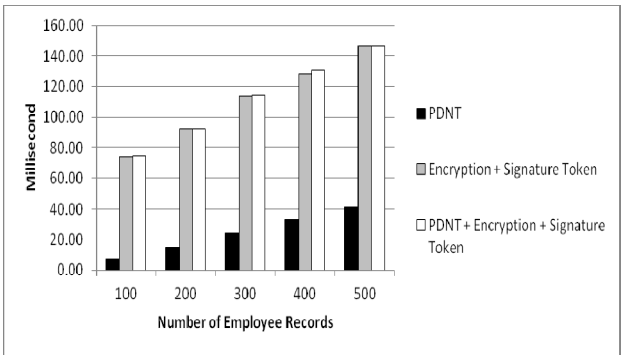

Figure12. Latency in milliseconds for Test Case 4

H. Results Analysis

The latency of each test case includes the time that it takes for a message transfer between the two end-points, for XML parsing, for secure token parsing, for secure token processing, and for the DNS IP address lookup for the PDNT. However, the DNS lookup is not a time consuming process because it always uses the local server DNS cache after its first query. The message transfer time between the client and server is also not a major consideration in this testing environment, because the client and server process are both running on the same computer. According to the results of the four test cases, all results that only adopt the PDNT token are significantly faster than other secure token profiles. Moreover, the overhead of adopting PDNT tokens with other secure token profiles is minor. Therefore, the PDNT has a performance advantage.

V. THE ADVANTAGES OF THE PROPOSED TOKEN

Three advantages accrue from the proposed Participant Domain Name Token (PDNT). First, it provides one more security feature, which the WSS token profiles do not include. The PDNT works with the DNS to provide location or domain validation of a SOAP message. Second, the PDNT has a performance advantage when compared with the WS-Security token profiles. The overheads of the XML signature, XML encryption, and the five secure token profiles are significant. Therefore, by using the proposed token, a service provider can reject an unknown domain or faked SOAP request as soon as possible. The proposed token is parsed and processed before the WS-Security core specification and other secure token profiles are processed. Therefore, it can save server resources such as CPU time, memory, and energy and it can handle more valid SOAP requests. Based on the evaluation of the four test cases, it takes less than 1% overhead to adopt PDNT, which is not a significant cost. Third, a permitted domain list feature is also provided. It acts like a whitelist or approved list to control "Where" the Web services method can be invoked. The PDNT is simple and easy to implement as compared with other Web services security specifications and token profiles.

VI. CONCLUSIONS

The Service-Oriented Architecture (SOA) is a preferred design and architecture style for the integration of ubiquitous computing resources. Traditionally designed ICT systems use an application style, which is designed as a tightly coupled system. This is less effective and suffers from the dependencies of each component. The invention of Service-Oriented Architecture (SOA) makes each component system independent and permits for a loosely coupled system. The SOA can be adopted in a large complex system that includes many independent components. For instance, online shopping applications are composed of different functions like credit card authorization, currency conversion, searching for the best prices, etc. These components can be designed and implemented into

several independent services. The services can then be used in a single application or in other applications. Service reuse is one of the advantages of adopting SOA. Moreover, adopting the SOA can gain benefits in enterprise application integration, service reuse, leveraging the legacy investment, and provides best of breed integration. Therefore, SOA is suitable to design a distributed, Internet-based, dynamically changed, autonomous, and non-point-to-point system.

Web services are one of the technologies that are used to implement SOA. A Web service is a kind of distributed system that provides APIs, which can be used by all systems on the Internet. Currently, the Simple Object Access Protocol is the most common way to implement Web services and it is a core component of the Web service architecture. SOAP is a specification for exchanging structured messages between two computer systems via common communication protocols such as Hypertext Transfer Protocol (HTTP). SOAP relies on XML for its message format and a SOAP message can be protected in the transport and message layer. The message layer security is a proper method for protecting the message format from end to end. Existing message layer security standards are collected and defined in OASIS standard 1.1. However, it does not authenticate the location of a remote client. It only authenticates who is calling the Web services and what their permissions are. This paper presents a brief introduction to XML and Web services security standards and their relationships. Some researchers propose other mechanisms to protect Web Services such as Ontology and data mining techniques. On the other hand, performance is one of the major concerns for Internet-based systems. Therefore, a new secure token profile, which is known as the Participant Domain Name Token (PDNT) profile, has been proposed in this paper.

The PDNT can be used for the authentication of a message sender location. The newly designed secure token profile incorporates one of the most widely used Internet service information identities, which is called the Domain Name Service (DNS). It can verify, control, and monitor the location of the service consumer or message sender. The newly designed token can be used to reject invalid requests or responses, which come from unknown or fake domains, before parsing and processing the other secure token profiles. The performance comparisons of the proposed token with the Username Token, decryption process, and signature have been presented. The results show that the proposed token provides performance advantages and the overhead of adopting the token is not significant for all of the test cases. Therefore, by adopting this token, the service provider or consumer can save on processing resources, they can handle more valid requests for Web services, and they can reject invalid requests that come from unknown or fake domains.

REFERENCES

[1] Davis J., *Open Source SOA*, Manning Publications Co., 2009.

[2] TechTarget / Forrester Research, *"State of SOA 2010"*, http://cdn.ttgtmedia.com/searchSOA/downloads/TTAG-State-of-SOA-2010-execSummary-working-523%5B1%5D.pdf

[3] G. A. Lewis, E. Morris, S. Simanta, L. Wrage, *"Common Misconceptions about Service-Oriented Architecture", Proceedings of 6th International Conference on Commerical-off-the-Shelf (COTS)-Based Software Systems, ICCBSS'07)*, 2007, pp.123-130.

[4] W.T. Tsai, C. Fan, Y. Chen, R. Paul, J. Y. Chung, *"Architecture Classification for SOA-Based Application", Proceedings of 9th International Symposium on Object and Component-Oriented Real-Time Distributed Computing*, 2006, pp.295-302.

[5] C. P. Cheong, S. Fong, P. Lei, C. Chatwin, R. Young, "Designing an Efficient and Secure Credit Card-based Payment System based on ANSI X9.59-2006 with Web Services", *Journal of Information Processing System, Korea, Information Processing Society*, Volume 8, 2012, ISSN: 1976-913X (Print), ISBN: 2092-805X (Online), 2013, pp.495-520.

[6] C. P. Cheong, C. Chatwin, R. Young, *"An SOA-Based Disease Notification System", Proceedings of 7th International Conference on Information, Communcations and Signal Processing*, ICICS2009, 2009, pp.1-4.

[7] C. P. Cheong, C. Chatwin, R. Young, *"A Framework for Consolidating Laboratory Data Using Enterprise Service Bus", Proceesings of 3rd IEEE Internation Conference on Computer Science and Information Technology*, 2010, pp.557-550.

[8] A. Barbir, C. Hobbs, E. Bertino, F. Hirsch, L. Martion, "Challenges of Testing Web Services and Security in SOA Implementations", *Testing Analysis if Web Services*. SpringerLink, 2007, pp. 395-440.

[9] A. Nadalin, C. Kaler, R. Monzillo, P. Hallam-Baker, Web Service Security: SOAP Message Security 1.1. (WS-Security 2004), OASIS Standard Specification, 1 February 2006.

[10] A. Nadalin, C. Kaler, R. Monzillo, P. Hallam-Baker, Web Service Security: Username Token Profile 1.1, OASIS Standard Specification, 1 February 2006.

[11] A. Nadalin, C. Kaler, R. Monzillo, P. Hallam-Baker, Web Service Security: X.509 Certificate Token Profile 1.1, OASIS Standard Specification, 1 February 2006.

[12] R. Monzillo, C. Kaler, A. Nadalin, P. Hallam-Baker, Web Service Security: SAML Token Profile 1.1, OASIS Standard Specification, 1 February 2006.

[13] A. Nadalin, C. Kaler, R. Monzillo, P. Hallam-Baker, Web Service Security: Kerberos Token Profile 1.1, OASIS Standard Specification, 1 February 2006.

[14] T. D. ContentGuard, A. Nadalin, C. Kaler, R. Monzillo, P. Hallam-Baker, Web Service Security: Rights Expression Language (REL) Token Profile 1.1, OASIS Standard Specification, 1 February 2006.

[15] T. Imamura, B. Dillaway, E. Simon, XML Encryption Syntax and Processing, W3C Recommendation, 10 December 2002.

[16] M. Bartel, J. Boyer, B. Fox, B. LaMacchia, E. Simon, XML Signature Syntax and Processing (Second Edition), W3C Recommendation, 10 June 2008.

[17] S. Chen, J. Zic, K. Tang, D. Levy, *"Performance Evaluation and Modeling of Web Service Security", Proceedings of 2007 IEEE International Conference on Web Services (ICWS 2007)*, 2007, pp.431 – 438.

[18] S. Chen, B. Yan, J. Zic, R. Liu, A. Ng, *"Evaluation and Modeling of Web Services Performance", Proceedings of*

International Conference on Web Service (ICWS' 06)", 2006, pp.437 – 444.
[19] K. Xiong, *"Web Services Performance Modeling and Analysis", Proceedings of International Symposium on High Capacity Optical Networks and Enabling Technologies*, 2006, pp.1 – 6.
[20] C. P. Cheong, C. Chatwin, R. Young, *"Performance enhancement of WS-security using Participant Domain Name (PDNT)", Proceedings of International Joint Conference on Computer Science and Software Engineering (JCSSE)*, 2012, pp.213-218.
[21] S. Kent, K. Seo, *RFC 4301:* Security Architecture for the Internet Protocol, Network Working Group on Internet Engineering Task Force, December 2005.
[22] [22]S. Cantor, J. Kemp, R. Philpott, E. Maler, Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0, OASIS Standard, 15 March 2005.
[23] C. Neuman, T. Yu, S. Hartman, K. Raeburn, *RFC 4120:* The Kerberos Network Authentication Service (V5), Network Working Group of Internet Engineering Task Force, July 2005.
[24] ISO/IEC 21000-5:2004, Information technology – Multimedia framework (MPEG-21) – Part 5:Rights Expression Language, International Organization for Standardization, 2004.
[25] B. Genge, P. Haller, *"Extending WS-Security to Implement Security Protocols for Web Services", Proceedings of 1st Internatonal Conference on Recent Achievements in Mechartronics, Automation, Computer-Sciences and Robotics*, 2009, pp.105-112.
[26] E. Damiani, S. De Capitani di Vimercati, S. Paraboschi, P. Samarati, "Secure SOAP E-Services", *International Journal of Information Security (IJIS)*, vol. 1, n. 2, 2002, pp. 100-115.
[27] [27]R.S. Sandhu, E.J. Coyne, H.L. Feinstein, C.E. Youman, "Role-based access control models", *IEEE Computer*, 1996, pp. 38-47.
[28] S. Oh, S. Park, "Task-role-based access control model", *Journal of Information System*, Elsevire, 2003, Vol 28, Issue 6, pp. 533-562.
[29] M. Kudo, "PBAC: Provision-based access control model", *International Journal of Information Security*, Springer Berlin, 2004, pp. 116-130.
[30] J. X. Li, B. Li, L. Li, T. S. Che, *"An Agent-based Policy Aware Framework for Web Services Security", Proceedings of 2007 IFIP International Conference on Network and Parallel Computing*, 2007, pp. 849-854.
[31] H. F. EL Yamany, M. A.M. Capretz. *"Use of Data Mining to Enchance Security for SOA", Proceedings of Third 2008 International Conference on Convergence and Hybrid Information Technology*, 2008, pp. 551-558.
[32] P., Mockapetris, Domain Names – Implementation and Specification, RFC 1035, IETF, 1987.
[33] Gulbrandsen, P. Vixie, L. Esibov, *RFC 2782:* A DNS RR for specifing the location of services, Network Working Group of the Internet Engineering Task Force, February 2000.
[34] Lazowska, J. Zahorjan, S. Graham and K. Sevcik, *Quantitative system performance:computer system analysis using queueing network models*, Prentice Hall, Englewood Cliffs, N. J.,1984,pp.203-300.
[35] (n.d.). *"Jakarta Commons HTTP Client"*, http://hc.apache.org/httpclient-3.x/

Chi Po Cheong Mr. Cheong is currently pursuing a PhD degree from the School of Science and Technology at the University of Sussex in Brighton, UK. In 2007, Mr. Cheong graduated with a Master of Science with a major in E-Commerce Technology from the University of Macau.

Chris Chatwin Professor C. R. Chatwin holds the Chair of Engineering, University of Sussex, UK; where, inter alia, he is Research Director of the "iims Research Centre" and the Laser and Photonics Systems Engineering Group. At Sussex he is a member of the University: Senate, Council and Court. He has published two research monographs: one on numerical methods, the other on hybrid optical/digital computing - and more than two hundred international papers. Professor Chatwin is on the editorial board of the International Journal "Lasers in Engineering." He is also a member of the Institution of Electrical and Electronic Engineers, the British Computer Society, and the Association of Industrial Laser Users. He is a Chartered Engineer, Euro-Engineer, International Professional Engineer, Chartered Physicist, Chartered Scientist, and a Fellow of The Institution of Electrical Engineers, The Institution of Mechanical Engineers, The Institute of Physics, and The Royal Society for Arts, Manufacture, and Commerce.

Rupert Young Dr. Rupert Young currently is serving as a Reader in Engineering (Engineering and Design) and Optical and Medical Imaging (Biomedical Engineering) at the University of Sussex in Brighton, United Kingdom. Dr. Young obtained his BSc degree and PhD degree from the University of Glasgow, UK.