GSJ: Volume 11, Issue 4, April 2023, Online: ISSN 2320-9186
www.globalscientificjournal.com

SHARING WEB SERVICES BETWEEN ENTERPRISES: "TRUSTED NETWORK"

Dr. James Mwikya[a], Dr. Josphat Karani[b]
Kirinyaga University, P.O.Box: 143-10300, Kerugoya, Kenya
[a]jmwikya@kyu.ac.ke, [b]jkarani@kyu.ac.ke

Abstract
Web services are self-contained, modular, distributed, and dynamic applications that can be created, published, searched, or invoked over the network to create products, processes, and supply chains. These applications can be local, distributed, or web-based. Web services are based on open standards such as Transmission Control Protocol/Internet Protocol (TCP/IP), Hypertext Transfer Protocol (HTTP), HyperText Markup Language (HTML), Extensible Markup Language (XML), and Java. Web services enable software applications written in different programming languages and running on different platforms to exchange data over computer networks such as the Internet. This is similar to communication between processes on a single computer. Web services allow companies to host their services in a larger market and connect with other companies. But as the business grows, trust issues arise. A trust relationship is required to share web services in a business partner relationship. The main principles of trust in web services are security and privacy. Reliance may be directly enforced or brokered by a third party. Therefore, to build web services as a "trust network," an organization must understand the threats that web services pose and the technical solutions to mitigate those threats. Then, either directly or through a third party, establish and follow a defined engineering process that manages security from the beginning and throughout the lifecycle of the web service.

Key words: web services, network of trust, business enterprises

I. Introduction

Web services are loosely coupled, self-contained, self-describing, and modular applications that can be described, published, retrieved, and invoked over a network (Rostami, Kheirkhah & Jalali, 2013). Web services can be deployed on any platform and written in any programming language. "Using C#, you can create new web services on Windows, which can be called from web applications running on Linux based on Java Server Pages (JSP)" (Pautasso, Zimmermann , and Leymann, 2008). Web services are the incarnation of middleware for distributed computing and, unlike all previous middleware, are simpler, standards-based, loosely coupled technologies for connecting data, systems and organizations (Oh, Ivezic & Niemann, 2019). A service-oriented architecture (SOA) is a collection of services that communicate with each other. SOA describes interactions between software agents as message exchanges between service requesters and service providers. SOA describes the interactions between software agents as message exchanges between service requestors and service providers (Karande, Chunekar, & Meshram, 2011).

II. Web Services as SOA

Web services basically cover three roles in service-oriented architecture. Service Provider (Publisher), Service Requester (Consumer), and Service Broker (Register) (Rajendran & Balasubramanie, 2010). Service Provider (Publisher) A provider of web services. A platform that hosts services. It can be an industry, business, or company that you can serve. Service providers implement services and make them available on the Internet or intranets. Service Requester (Consumer) This is any consumer of a web service. A requester can be a company or a company that needs services. A service requester is an application that looks up services and invokes or initiates interactions with them. A browser acts as a requester, leveraging a consumer or program with no user interface. Requesters leverage existing web services by opening network connections and sending XML requests. Service Broker (registry) This is a logically centralized directory of services. A broker is a location, entity, or system that helps both service providers and service requesters find each other. As such, it acts as a central clearinghouse for companies and their services (Rajendran & Balasubramanie, 2010).

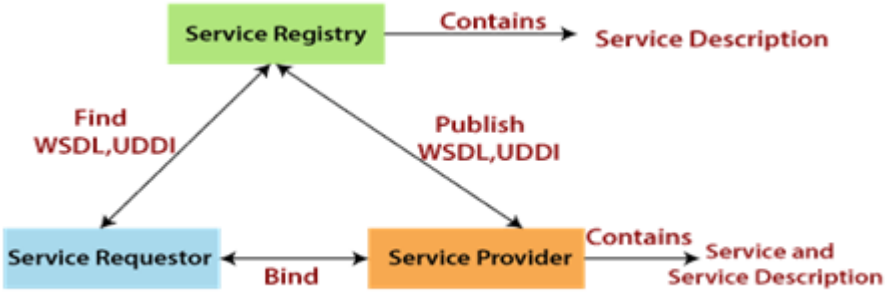
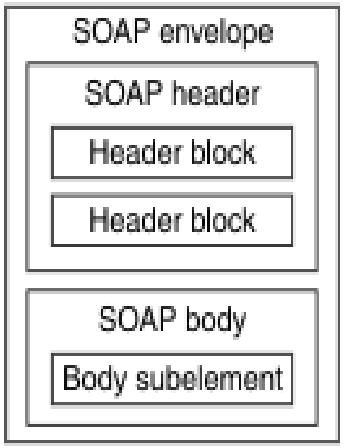


Figure 1: web services roles and technologies

According to Hamid, Abdalrahman, Lafta, and Barazanchi (2019), four technologies form the basis of web services:Extensible Markup Language (XML); Simple Object Access Protocol (SOAP); Web Services Description Language (WSDL); Universal Description, Discovery, and Integration (UDDI).

XML: The eXtensible Markup Language (XML) was developed as a structured, self-describing way to represent data completely independent of applications, protocols, vocabularies, operating systems, and even programming languages. XML was originally developed to overcome the limitations of HTML. HTML is good at describing how things should be displayed, but bad at describing what data to display (Kuyoro, Ibikunle, Awodele & Okolie, 2012).

SOAP: Simple Object Access Protocol (SOAP) is used for communication between different web services. SOAP is XML-based. In other words, it is platform and language independent. SOAP is designed to transfer XML from one computer to another over a set of standard transport protocols (Tihomirovs & Grabis, 2016). HTTP is the most common and widely used transport used by the web itself. A SOAP message flows from the sender to the final recipient through the SOAP message path. A SOAP message consists of a SOAP envelope containing a SOAP body element and optional SOAP header elements. A soap header element can contain a number of child elements that describe the message handling that the sender expects the recipient to do. A typical SOAP list is shown below.




```
<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.kyu.ac.ke/2023/05/soap-envelope/"
soap:encodingStyle="www.kyu.ac.ke/2023/05/soap-encoding">
<soap:Header>
</soap:Header>
<soap:Body>
</soap:Body>
</soap:Envelope>
```

Figure 2: A Simple SOAP message

SOAP envelopes are used to encapsulate SOAP messages. SOAP headers are an optional part of the SOAP protocol. The header contains information about the SOAP node, the SOAP message's processor, and how to process the SOAP message. This could be authentication, routing, etc. The SOAP body contains the message intended for the recipient of the SOAP message (Tihomirovs & Grabis, 2016).

WSDL: Web Services Description Language (WSDL) is used to describe the functionality provided by web services. WSDL files have the extension .wsdl. It is an XML language that defines what the input/output structure of a web service should look like and what should be expected in the XML payload message. It provides a machine-readable description of how to call the service, the parameters the service expects, and the data structures it returns. WSDL tells how one service interacts with another, where it is located, what it can do, and how it can be invoked (Fokaefs, Mikhaiel, Tsantalis, Stroulia, and Lau ,2011). Once the requester receives the WSDL document for the candidate Web service, it should validate it. The easiest way to do this is to provide a digital signature for the requester's WSDL document to use. Requesters cannot connect to most providers without some form of authentication.

UDDI: Universal Description Discovery and Integration (UDDI) is used as an information registry for web services. H. To post and retrieve information. UDDI Services is an industry-wide effort to establish a common standard for business-to-business (B2B) integration (Hasmy, 2016). It defines a set of standard interfaces for accessing web service databases. The purpose of UDDI is to allow users to discover and dynamically interact with available Web services. UDDI provides services and the technical interfaces through which these services can be accessed. This process can be divided into three phases: Search (Discover), Bind, and Execute (Hasmy, 2016).

III. Web Service Trust Concerns

Web services are an attractive, powerful and preferred technology not only for application integration, but also for developing distributed applications. Web services provide interoperability between security policy domains. Web services also present daunting privacy and security challenges.The decentralized and heterogeneous nature of web services simply makes them a portal for users to access data. It makes security implementation difficult because it does not provide a variety of entry points that can be exploited. For illegal purposes (Ylianttila, et al. , 2020). Widespread acceptance by developers and customers in business-to-business (B2B) and business-to-consumer (B2C) environments requires that trust be assured among users of Web services. Trust plays a key role in facilitating interactions in such an uncertain environment. A trust provider is an entity that trusts a trusted subject entity, the trustee (Zainab & Mark, 2018). According to Mayer, Davis, and Schoorman (2010), trust is the willingness of a party to be vulnerable to the actions of another based on the expectation that the other will perform specific actions that are important to the trust giver. . The ability to monitor or control its opponent." One of the most important principles of trust is security and data protection. Security and privacy are important factors to consider when building trust (Ylianttila, et al., 2020).

According to Vacca (2013), web-based applications came along with the advent of e-commerce, so secure access to web servers was very important. There are now many mature perimeter security technologies, such as Secure Sockets Layer (SSL), firewalls,

and web authentication/authorization servers that provide additional security between browser clients and corporate web servers (Vacca, 2013). Figure 3 shows new and existing security mechanisms for protecting web services at different security levels.

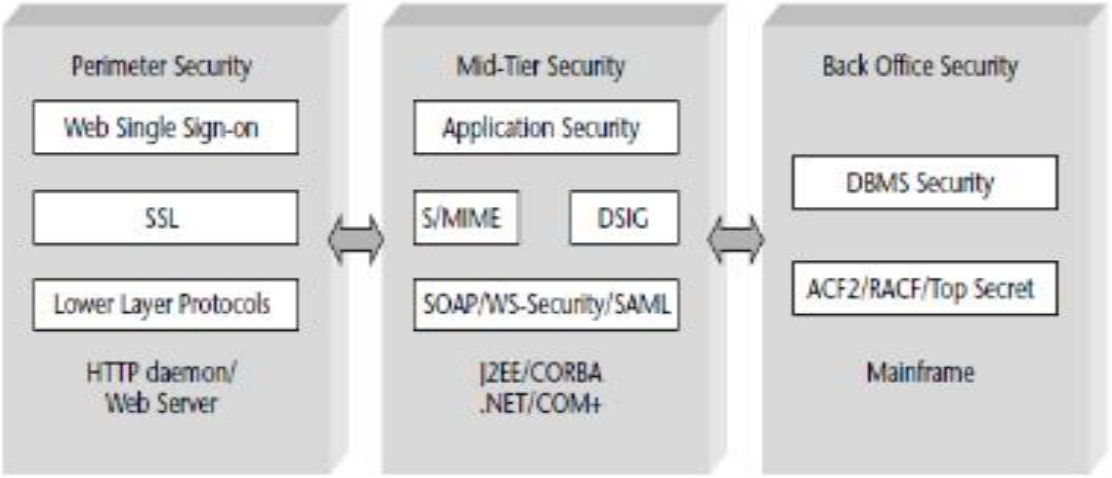


Figure 3: Web Service Security implementation

IV: General Security Framework

Web services, like other distributed applications, require extensive security mechanisms to ensure secure data transfer. As we all know, there are no programming language, architecture, or system dependencies when it comes to sharing existing information and functionality via web services. Because of this interoperability feature and cross-platform access of web services, more attention should be paid to security concepts (Ramalingam & Mohan, 2021). Describes the core security services that underpin end-to-end application security across multi-tier applications.

Authentication: Authentication helps prove the identity of an entity. Entities can be users, services, or processes. The solution to this problem is to authenticate both the requester and the provider and the registrar (Tobin & Reed 2016). All components and units must be certified. This may be done by a trusted third party. Once a user logs into the registry, single sign-on is required for each service that user can access. Once a user is authenticated with a login ID and password, routes are established between various web services based on a trust relationship called federated trust (Tobin & Reed 2016).

Authorization: Grants permissions to resources to principals and forms the basis of access control. This enforces access restrictions and prevents unauthorized use. Access control ensures that only authorized clients can modify resources and that resource content is shared only with authorized clients (Kuyoro, Ibikunle, Awodele, and Okolie, 2012). There are two technologies for determining authorization and authentication for web services, Extensible Access Control Markup Language (XACML) and Security Assertion Markup Language (SAML), which eliminate the overhead of checking authentication and authorization in web services (Kuyoro, Ibikunle , Awodele & Okolie, 2012).

cryptography: Provides cryptographic algorithms and protocols to protect data and messages from disclosure or tampering. Encryption provides confidentiality by encoding data into an unintelligible form using a reversible algorithm that allows the owner of the decryption key to decrypt the encrypted data. Digital signatures provide integrity by applying encryption to ensure that data is authentic and has not been altered during storage or transmission (Deshmukh, Kapse, Thakare, & Kapse, 2023).

Confidentiality: A process whereby only the sender and intended recipients of a message can access it, and unauthorized entities cannot access it (Gugunani, Ghrera, Gupta, Malekian & Maharaj, 2016). To do this, the message is encrypted and decrypted. Encryption is the process of transforming plaintext into ciphertext using a key and an encryption algorithm. Decryption is the reverse of encryption, converting ciphertext into plaintext. XML Encryption is used for encryption and confidentiality in web services and serves as the basis for web service confidentiality. This is done using XML digital signatures (Gugunani, Ghrera, Gupta, Malekian, Maharaj, 2016). Another pervasive approach is to use a virtual private network (VPN).

Non Repudiation: Message senders cannot claim that they did not send a particular message. This proves that a particular transaction was made by the recipient, and neither party can refute or deny that it was subsequently done (Piper, 2019). If the transaction is contested in court, the contract purportedly concluded must be proven by both parties. Each party must have verified a signed contract, traditionally confirming their identity by verification of a wet paper signature, and must have a notary witness verified at the time of signing. XML signatures are used for non-repudiation, but are insufficient for today's toughest security threats (Piper, 2019). That's why you need a trusted third party. Nonrepudiation relies on public key cryptography.

availability: It states that resources and services should be available to qualified parties at all times. A denial of service (DoS) is an attack designed to disrupt the availability of a service (Osanaiye, Alfa & Hancke, 2018).

Integrity: When a message is sent to a user and not received by the recipient exactly as it was sent, this is a loss of message integrity. Messages that require data integrity must explicitly or implicitly include the sender's identity and credentials to enable this type of message her level of security (Baee, Simpson, Boyen, Foo, and Pieprzyk, 2021). Data integrity is achieved using digital signatures. The techniques used to maintain integrity are MD5 and SHA.

Security management: Defines the maintenance lifecycle of security policies and embodies them in user profiles, authentication, authorization, accountability mechanisms, and other data relevant to the security framework (Farris, Taleb, Khettab, & Song, 2018).

Multi Part Multi Signature Document MPMSD: This scenario is used manually in the office, where one employee writes a document and gives it to another employee, the next reviewer reviews the document, comments and signs it, the next reviewer pass it to The resulting document is a compound document with multiple comments and multiple reviews, and the type of resulting compound document is called a Multi Part Multi Signature Document (Borowski, 2020). This process can be used for authentication, authorization, and document integrity in web services that use digital signatures and XML documents (Borowski, 2020). Digital signatures can be used to establish trust in MPMSD. The MPMSD scenario is illustrated in Figure 4.

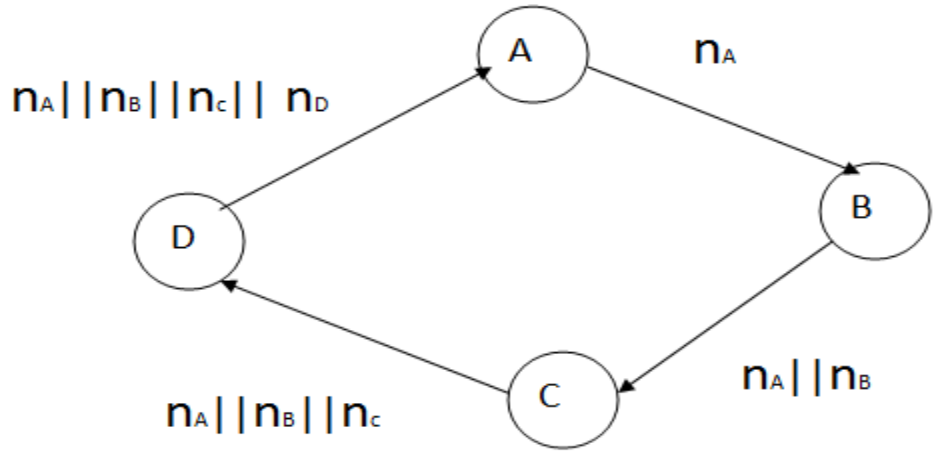


Figure 4: MPMSD Scenario

V: Web Services Security(Trust)Implementation

Authentication technology: Basic authentication is used in almost all applications. Users are prompted for a username and password before they can get any functionality of the application. Both are verified. The main drawback of the implementation is that credentials are easily passed from client to server. Any sniffer can read packets sent over the network.

- *Security Assertion Markup Language (SAML);* to provide strong authentication and authorization tokens. Developed by OASIS, SAML is an open framework for exchanging security information over the Internet via XML documents (Naik & Jenkins, 2017). A user requests a service from a service provider. A service provider requests (to establish trust) and receives an identity assertion from an identity provider. Based on this assertion, the service provider can make access control decisions about whether to run the service for the connected user. Before sending an Identity Assertion to a Service Provider, an Identity Provider may request information from you such as: Usernames and passwords (Naik & Jenkins, 2017).

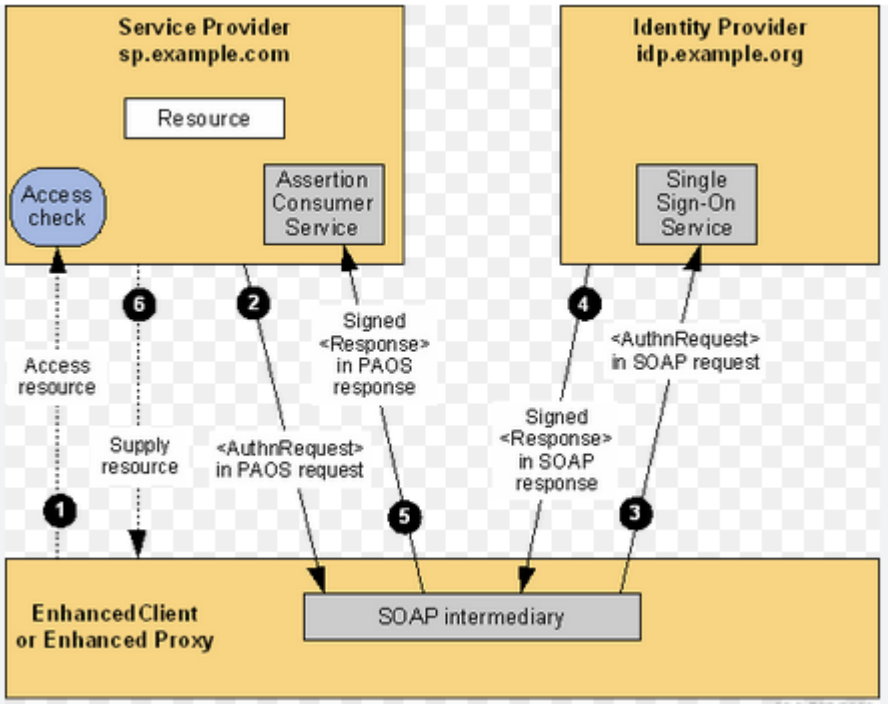


Figure 5: SAML Architecture

- *Digest authentication* encrypts the password entered by the user before sending it to the server. This is another authentication type specified in HTTP1.1. Digest authentication does not require sending a password. Instead, the client takes the username and password, creates a hash using the MD5 hashing algorithm, and sends it to the SQL server. This can be used to verify a user's identity before sending sensitive information such as online banking transactions (Tsai & Su, 2021). An advantage of digest authentication is that it is resistant to replay attacks.

- You can also use certificates to provide certificate authentication. When a client attempts to access a protected web service resource, it presents a certificate containing credentials and a unique public/private key pair to the server. The server verifies the same (Buchelt, 2017). This mechanism must use HTTPS as there is no secure channel for communication. A Kerberos token is a cross-platform authentication and single sign-on system(Buchelt, 2017). The Kerberos protocol provides mutual authentication between two entities based on a shared secret (symmetric key).

Authorization method: XACML (eXtensible Access Control Markup and SAML Language) are two technologies used to detect authorization information. When using SAML, authorization decision assertions involve determining whether a principal can access (authorize) a particular resource given authentication and attribute assertions (Seitz, Selander, & Gehrmann, 2013). . SAML provides a standard way to represent security tokens that can traverse various steps in a business process or transaction, from browsers to portals to networks of web services. This is also a feature supported by

OWSM. In XACML, Policy Decision Point (PDP) and Policy Enforcement Point (PEP) entities respectively make and enforce authorization decisions according to policy. Authorization essentially includes role-based access control (RBAC) and context-based access control (CBAC). RBAC maps security controls to organizational structures (Seitz, Selander & Gehrmann, 2013). RBAC allows you to assign permissions to each user based on their role within your organization.

Confidentiality technology: XML Encryption is the technology used to implement this security control feature. Used when information in a SOAP message needs to be kept confidential while being sent over a multi-hop SOAP transaction. Additionally, XML encryption is useful when the information in a SOAP message needs to remain encrypted after the SOAP message has been processed by a web service. XML Encryption is a World Wide Web Consortium (W3C) recommendation. Encrypted data can be expressed using XML, or portions of the XML document can be selectively encrypted. Algorithms such as Triple-DES and AES are commonly used.

Integrity Techniques: 1. The WSDL file describes the functionality of the web service. The WSDL forms the basis for web service selection, especially during assembly, when a service requester communicates with a UDDI registry. If tampered with, it destroys the integrity of the web service. 2. XML digital signatures can also be used for integrity and non-repudiation of WSDL files, allowing you to publish web service definitions and trust that they have not been tampered with later (Kumar, Singh, Singh, 2016). Developed by W3C and IETF. In web service communication, data integrity may also exist in the lower layers of the OSI stack. However, if integrity is implemented only for SOAP communication, it cannot be said to be persistent (SOAP works with OSI applications and relies on HTTP for transport). Persistent consistency is useful when a document consists of parts created by multiple entities. This is a workflow scenario like Multiple Parties Multiple Signature Document (MPMSD) (Kumar, Singh & Singh, 2016). XML signatures guarantee the integrity of certain parts of the document, but allow participants to edit other unsigned parts of the document.

Non-repudiation technology: WS-Security uses digital signatures to ensure non-repudiation. The specification allows for different signature formats, encryption algorithms, and multiple domains of trust, and is open to different security token models such as X.509 certificates, Kerberos tickets, SAML assertions, and custom tokens. (Meduri, 2019). It injects security features into SOAP headers and works at the application layer. The X.509 specification is fully extensible and is used to format information in digital certificates. A digital certificate typically contains the corresponding private key, serial number, expiration date, and identity information about the entity that owns the public key.

VI: Threats to Web Services Trust

Threats facing web services can occur at the service level or the message level. Service-level threats can include threats faced by UDDI, WSDL, and XML.

Key Service Level Threats:

WSDL and UDDI attacks: Service level information is available in WSDL files and UDDI registries. An attacker can access and manipulate publicly available WSDL files. Attacks can take the form of WSDL scanning or WSDL manipulation. The former scans her WSDL file and reveals operational information, ports, etc. The latter can manipulate data and even access sensitive information. Protecting against these threats using common methods such as authentication and authorization is not trivial (Tabrizchi & Kuchaki, 2020).

Denial of service attack: A denial of service (DoS) attack is launched, compromising system availability. There are two ways to carry out a DoS attack. First, an attacker could consume web application resources to the point that other legitimate users cannot access or use the application. This can be achieved by submitting queries for large amounts of data. The second approach can occur if an attacker locks a user out of an account or overloads the service with too many requests to bring down the entire application. Attackers can combine these two of her approaches with her web service-specific attacks to maximize damage (Watanabe, Shioji, Akiyama, Sasaoka, Yagi & Mori, 2018).

Malicious code injection and spoofing: These attacks are primarily done with XML files. Attackers can inject malicious code and affect the functionality of the service. An identity spoofing attack occurs when a hacker steals the identity of a service requester or service provider. In the first case, an attacker can create a well-formed XML request message of her and send it to the service provider. This allows the service provider to assume that the response is being sent to a valid service requestor. In the second case, an attacker tricks a genuine service requester into sending a message to a bogus service provider. Such attacks are not immediately detectable (Gupta, Tewari, Jain & Agrawal, 2017).

XML schema manipulation: A schema file uses an XML parser to understand the grammar and structure of XML and contains important preprocessor directives. An attacker could corrupt the XML schema or replace it with a modified schema. This allows parsers to process malicious SOAP messages and specially crafted XML files to inject operating system commands into a server or database (Hong, Nhlabatsi, Kim, Hussein, Fetais & Chan, 2019).

Session hijacking: Session hijacking involves unauthorized control of a legitimate user's session state. This happens when an attacker steals a valid session ID (a valid session cookie) and uses it to gain the permissions of a specific user within your application. By

intercepting or sniffing SOAP messages, an attacker can hijack a user's session in the same manner as a typical web application attack. However, once a hacker authenticates as a valid user, they can perform more dangerous activities (Abdul-Ghani, Konstantas & Mahyoub, 2018). Note:
Don't use public Wi-Fi for banking.

Major message level threats:

SQL Injection: is a high-risk exploit which may be performed using SOAP messages. If a server does not validate data correctly, a SOAP message can easily be used to create XML data which inserts a parameter into an SQL query and have the server execute it with the rights of the Web Service (Gupta & Thilagam, 2013). SQL Injection is only one of the threats a server is exposed to if data is not validated.

Replay of Messages: An attacker captures a valid message and replays it later to gain sensitive information via unauthorized access to the services. Usually this is the initial step in hacking a web service wherein the hacker can hijack the session and tamper with the services. With the right tools, patterns can be detected more easily even if the same or similar payload is being sent across multiple mediums like HTTP, HTTPS, and SMTP or across different interfaces (Oberoi, Raj, Ongsakul & Goyal, 2023)

Message Confidentiality and Eavesdropping: Interception of messages is always a threat to web services. Traditional security mechanisms like VPN or SSL are not sufficient to secure the web services against such threats (Kumar, Srivastava & Singh, 2015).

VII: Current Technology for Managing Web Services Security Threats
WS-Security: WS-Security is a building block intended to be used in conjunction with other web services and application-specific protocols. Support different security models. This protocol is now officially called WSS and was developed by a committee at Oasis-Open (Ruiz, Arjona, Maña & Rudolph, 2017). This standard defines how security tokens are included in SOAP messages, an XML security specification for token encryption and signing. A token is attached to the message and a timestamp is also inserted. Other parts of SOAP messages can also be encrypted, and the method is mentioned in the standard (Ruiz, Arjona, Maña & Rudolph, 2017).

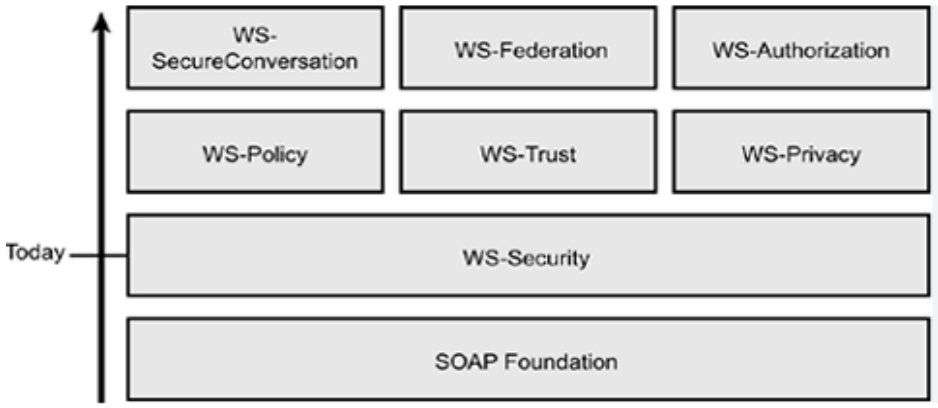


Figure 6: Overview of Web Service Security

Key features of this standard include WS-Policy, WS-Trust, and WS-Privacy. WS-Policy allows organizations to set security requirements for web services. WSDL bindings are used to attach policy information to web services. WS-Trust defines how to build trust relationships. Trust is either direct or intermediary. For mediated trust, the trust proxy is used to read the policy information and request the appropriate security tokens to insert into the SOAP message. WS-Privacy describes how to include privacy requirements in your policy. It also uses WS-Trust to evaluate privacy claims in SOAP messages against user and organizational preferences (Saadaoui, & Scott, 2017).

Secure socket layer: Secure Socket Layer (SSL) is a protocol or technology used to protect organizations from web service security attacks. SSL is used in encryption technology and is used to implement privacy protection. SSL creates a secure tunnel between a source and destination computer based on public key cryptography (Kumar & Deepamalar, 2016). A common safeguard is to send messages over a secure connection using SSL. For simple applications, for example, an SSL connection between two points may be sufficient. Some web services allow encrypting and signing entire messages or individual parts of messages to protect the confidentiality and integrity of web service messages (Kumar & Deepamalar, 2016).

XML encryption: XML Encryption provides end-to-end security for applications that require secure modification of structured data. XML Encryption is primarily a confidentiality guarantee for encrypting XML data. XML-based cryptography is a natural way to meet the security requirements of data exchange applications (Liu, 2022). XML Encryption is not intended to replace or replace Secure Socket Layer (SSL). Rather, it provides a mechanism for security needs not covered by SSL. XML encryption is great for confidentiality. XML Encryption does not introduce any new encryption algorithms or techniques. RSA encryption can still be used for real encryption (Liu, 2022).

SAML: Security Assertion Markup Language (SAML) is a protocol for asserting authentication and authorization information. It also provides end-user attributes in XML format. Information can be put into a SOAP message. A SAML server can be

accessed for authentication and authorization data to enable single sign-on (SSO) (Chaturvedi, Matheus, Nguyen, & Kolbe, 2019). If the recipient of this SOAP message trusts the sender of her SAML data, the end-user can also authorize to her web service (Chaturvedi, Matheus, Nguyen & Kolbe, 2019).

XACML: The eXtensible Access Control Markup Language or XML-Access Control Markup Language (XACML) was developed to express access control rules in XML format. XACML can be used in combination with SAML, although the two technologies are not explicitly linked. Authorization decisions expressed in SAML assertions can be based on rules expressed in XACML (Kumar, 2020).

VIII: Conclusions

This work demonstrated the concept of using Web services as a "network of trust" by reviewing the operation of Web services, common security frameworks, and security implementations of Web services. Emerging web services security threats and current web services security techniques were also discussed. The main principles of trust in web services are security and privacy. Reliance may be directly enforced or mediated by a third party. Therefore, to build web services as a "trust network," an organization must understand the threats to web services, understand technical solutions to mitigate those threats, and have defined engineering practices to manage security. Must have a process. Must be set and followed. During the web service lifecycle from the beginning, either directly or through a third party. This process can be broken down into four steps:

1. Determine the appropriate security architecture for your web service.
2. Conforms to technical standards.
3. Establish an effective testing process for web services.
4. Create and maintain reusable and re-executable tests.

By following these four steps, the enterprise is almost guaranteed to build a reliable network of web services.

References

Abdul-Ghani, H. A., Konstantas, D., & Mahyoub, M. (2018). A comprehensive IoT attacks survey based on a building-blocked reference model. International Journal of Advanced Computer Science and Applications, 9(3).

Baee, M. A. R., Simpson, L., Boyen, X., Foo, E., & Pieprzyk, J. (2021). A model to evaluate reliability of authentication protocols in C-ITS safety-critical applications. IEEE Transactions on Vehicular Technology, 70(9), 9306-9319.

Beuchelt, G. (2017). Securing Web applications, services, and servers. In Computer and information security handbook (pp. 183-203). Morgan Kaufmann.

Borowski, K. E. (2020). Online Conflict Discourse, Identity, and the Social Imagination of Silesian Minority in Poland (Doctoral dissertation, University of Kansas).

Chaturvedi, K., Matheus, A., Nguyen, S. H., & Kolbe, T. H. (2019). Securing spatial data infrastructures for distributed smart city applications and services. Future Generation Computer Systems, 101, 723-736

Communication and Networking Technologies (pp. 1-7). IEEE.

Deshmukh, P. V., Kapse, A. S., Thakare, V. M., & Kapse, A. S. (2023). High capacity reversible data hiding in encrypted images using multi-MSB data hiding mechanism with elliptic curve cryptography. Multimedia Tools and Applications, 1-29.

Farris, I., Taleb, T., Khettab, Y., & Song, J. (2018). A survey on emerging SDN and NFV security mechanisms for IoT systems. IEEE Communications Surveys & Tutorials, 21(1), 812-837.

Fokaefs, M., Mikhaiel, R., Tsantalis, N., Stroulia, E., & Lau, A. (2011, July). An empirical study on web service evolution. In 2011 IEEE International Conference on Web Services (pp. 49-56). IEEE.

Gugnani, G., Ghrera, S. P., Gupta, P. K., Malekian, R., & Maharaj, B. T. J. (2016). Implementing DNA encryption technique in web services to embed confidentiality in cloud. In Proceedings of the Second International Conference on Computer and Communication Technologies: IC3T 2015, Volume 3 (pp. 407-415). Springer India.

Gupta, A. N., & Thilagam, P. S. (2013). Attacks on web services need to secure XML on web. Computer Science & Engineering, 3(5), 1.

Gupta, B. B., Tewari, A., Jain, A. K., & Agrawal, D. P. (2017). Fighting against phishing attacks: state of the art and future challenges. Neural Computing and Applications, 28, 3629-3654.

Hamid, S. A., Abdalrahman, R. A., Lafta, I. A., & Al Barazanchi, I. (2019). Web Services Architecture Model to Support Distributed Systems. Journal of Southwest Jiaotong University, 54(6).

Hasmy, A. J. M. (2016). Security issues in web services: a review approach with process and applications of multi–part multi-signature document.

Hong, J. B., Nhlabatsi, A., Kim, D. S., Hussein, A., Fetais, N., & Khan, K. M. (2019). Systematic identification of threats in the cloud: A survey. Computer Networks, 150, 46-69.

Karande, A. M., Chunekar, V. N., & Meshram, B. B. (2011). Working of web services using BPEL workflow in SOA. In Advances in Computing, Communication and Control: International Conference, ICAC3 2011, Mumbai, India, January 28-29, 2011. Proceedings (pp. 143-149). Springer Berlin Heidelberg.

Kumar, S., Srivastava, S., & Singh, A. (2015). Web Services Security: Threats and Challenges. International Journal of Computer Applications, 975, 8887.

Kumar, S., & Deepamalar, D. M. (2016). A Study on Web Service Security Technology. Globus An International Journal of Management & IT, 8(1), 1-4.

Kumar, S., Singh, S., & Singh, A. (2016). Security issues in web services: A evaluation and advancement perspective concerning research agenda. International Journal of Computer Applications, 135(7), 12-17.

Kumar, P. (2020). Prelude of security dispensation in web technology. Cosmos Journal of Engineering & Technology, 10(1), 1-4.

Kuyoro, S. O., Ibikunle, F., Awodele, O., & Okolie, S. O. (2012). Security issues in web services. IJCSNS International Journal of Computer Science and Network Security, 12(1), 23-27.

Liu, X. (2022, September). Design and implementation of heterogeneous data exchange platform based on web technology. In International Conference on Intelligent Systems, Communications, and Computer Networks (ISCCN 2022) (Vol. 12332, pp. 126-132). SPIE.

Meduri, R. K. K. (2019). Webservice Security. Webservices: Theory and Practice, 119-172.

Naik, N., & Jenkins, P. (2017, May). Securing digital identities in the cloud by selecting an apposite Federated Identity Management from SAML, OAuth and OpenID Connect. In 2017 11th International Conference on Research Challenges in Information Science (RCIS) (pp. 163-174). IEEE.

Oberoi, O., Raj, S., Ongsakul, V., & Goyal, V. (2023). Benefits and Risks of Cloud Computing. In Advancing Computational Intelligence Techniques for Security Systems Design (pp. 105-124). CRC Press.

Oh, H., Ivezic, N., & Nieman, S. T. (2019). Standards-based semantic integration of manufacturing information: Past, present, and future. Journal of manufacturing systems, 52, 184-197.

Osanaiye, O. A., Alfa, A. S., & Hancke, G. P. (2018). Denial of service defence for resource availability in wireless sensor networks. IEEE Access, 6, 6975-7004.

Pautasso, C., Zimmermann, O., & Leymann, F. (2008, April). Restful web services vs." big"'web services: making the right architectural decision. In Proceedings of the 17th international conference on World Wide Web (pp. 805-814).

Piper, F. (2019). Digital signatures for non-repudiation. In Practical Data Security (pp. 93-103). Routledge.

Rajendran, T., & Balasubramanie, P. (2010, July). An optimal agent-based architecture for dynamic web service discovery with qos. In 2010 Second International conference on Computing,

Ramalingam, C., & Mohan, P. (2021). Addressing semantics standards for cloud portability and interoperability in multi cloud environment. Symmetry, 13(2), 317.

Rostami, N. H., Kheirkhah, E., & Jalali, M. (2013). Web services composition methods and techniques: A review. International Journal of Computer Science, Engineering & Information Technology, 3(6), 10-5121.

Ruiz, J. F., Arjona, M., Maña, A., & Rudolph, C. (2017). Security knowledge representation artifacts for creating secure IT systems. Computers & Security, 64, 69-91.

Saadaoui, A., & Scott, L. S. (2017, October). Web services policy generation based on SLA requirements. In 2017 IEEE 3rd International Conference on Collaboration and Internet Computing (CIC) (pp. 146-154). IEEE.

Seitz, L., Selander, G., & Gehrmann, C. (2013, June). Authorization framework for the internet-of-things. In 2013 IEEE 14th International Symposium on" A World of Wireless, Mobile and Multimedia Networks"(WoWMoM) (pp. 1-6). IEEE.

Tabrizchi, H., & Kuchaki Rafsanjani, M. (2020). A survey on security challenges in cloud computing: issues, threats, and solutions. The journal of supercomputing, 76(12), 9493-9532.

Tobin, A., & Reed, D. (2016). The inevitable rise of self-sovereign identity. The Sovrin Foundation, 29(2016), 18.

Tihomirovs, J., & Grabis, J. (2016). Comparison of soap and rest based web services using software evaluation metrics. Information technology and management science, 19(1), 92-97.

Tsai, C. H., & Su, P. C. (2021). The application of multi-server authentication scheme in internet banking transaction environments. Information systems and e-business management, 19(1), 77-105.

Vacca, J. (Ed.). (2013). Network and system security. Elsevier.

Watanabe, T., Shioji, E., Akiyama, M., Sasaoka, K., Yagi, T., & Mori, T. (2018, April). User blocking considered harmful? An attacker-controllable side channel to identify social accounts. In 2018 IEEE European Symposium on Security and Privacy (EuroS&P) (pp. 323-337). IEEE.

Ylianttila, M., Kantola, R., Gurtov, A., Mucchi, L., Oppermann, I., Yan, Z., ... & Röning, J. (2020). 6G white paper: Research challenges for trust, security and privacy. arXiv preprint arXiv:2004.11665.