Security Implementation

Web services security, or to be more precise, SOAP message security identifies and provides solutions for general computer security threats as well as threats unique to Web services. WSO2 supports WS Security, WS-Policy and WS-Security Policy specifications. These specifications define a behavioral model for Web services. Since a requirement for one Web service may not be valid for another, the WSO2 Enterprise Integrator (WSO2 EI) also helps define service-specific security.

It provides 16 predefined, commonly-used security scenarios. All you have to do is apply the required security scenario into your service through the service's dashboard. You can also define a custom security policy. Understanding the exact security requirements is the first step in planning to secure Web services. Consider what security aspects are important to your service, whether it is the integrity, confidentiality, or both.

The default security scenarios

The topics below explain the 16 default security scenarios supported by WSO2.

- UsernameToken
- Non-repudiation
- Integrity
- Confidentiality
- Sign and Encrypt - X509 Authentication
- Sign and Encrypt - Anonymous clients
- Encrypt Only - Username Token Authentication
- Sign and Encrypt - Username Token Authentication
- Secure Conversation - Sign only - Service as STS - Bootstrap policy - Sign and Encrypt , X509 Authentication
- Secure Conversation - Encrypt Only - Service as STS - Sign and Encrypt , X509 Authentication
- Secure Conversation - Sign and Encrypt - Service as STS - Bootstrap policy - Sign and Encrypt , X509 Authentication
- Secure Conversation - Sign Only - Service as STS - Bootstrap policy - Sign and Encrypt , Anonymous clients
- Secure Conversation - Encrypt Only - Service as STS - Bootstrap policy - Sign and Encrypt , Anonymous clients
- Secure Conversation - Encrypt Only - Service as STS - Bootstrap policy - Sign and Encrypt , Username Token Authentication
- Secure Conversation - Sign and Encrypt - Service as STS - Bootstrap policy - Sign and Encrypt , Username Token Authentication
- Kerberos Token-based Security

UsernameToken
Username Token over HTTPS
Message is secured at the Transport level; Username Token is used for authentication. Client sends the Username Token inside the message which is validated against the entries in user store of the service’s end.

Non-repudiation Scenario
Clients should have X.509 certificates and Messages are signed using the private key of the sender and verified using the public key of the signing party.
Confidentiality

**Integrity Scenario**

Clients do not have X.509 certificates; message is signed using a symmetric key derived by Client, which is encrypted using the service’s public key and sent as an encrypted key to the service inside the SOAP message. Service can decrypt the encrypted key using its private key and verify the signature using the symmetric key.
Confidentiality Scenario

Clients do not have X.509 certificates; message is encrypted using a symmetric key derived by Client, which is encrypted using the service’s public key and sent as an encrypted key to the service inside the SOAP message. Service can decrypt the encrypted key using its private key and decrypt the message using the symmetric key.

Sign and Encrypt - X509 Authentication
"Sign and Encrypt – X.509 Authentication" Scenario

Client’s should have X.509 certificates; messages are encrypted using recipient’s public key and signed using sender’s private key. At the recipient’s end, messages are decrypted using recipient’s private key and message signatures are validated using sender’s public key. Each party should have the public certificate of the other party in its key store.

Certificate alias and private key password of the client’s certificate are used to authenticate the client. Clients should provide that information in order to sign the message.

Sign and Encrypt - Anonymous clients
Encrypt Only - Username Token Authentication

"Sign and Encrypt – Anonymous Clients” Scenario
Clients do not have X.509 certificates; message is signed and encrypted using a symmetric key derived by Client, which is encrypted using the service's public key and sent as an encrypted key to the service inside the SCAP message. Service can decrypt the encrypted key using its private key and get the symmetric key. Then service can use that symmetric key to decrypt the message and validate the signature.
Sign and Encrypt - Username Token Authentication

"Encrypt Only – Username Token Authentication" Scenario

Clients do not have X.509 certificates; message is encrypted using a symmetric key derived by Client, which is encrypted using the service’s public key and sent as an encrypted key to the service inside the SOAP message. Service can decrypt the encrypted key using its private key and get the symmetric key. Then service can use that symmetric key to decrypt the message.

Authentication is achieved by using Username Tokens. Clients send the Username Tokens inside a message and Service validates them against the entries in its user store.
"Sign and Encrypt - Username Token Authentication" Scenario

Clients do not have X.509 certificates; message is signed and encrypted using a symmetric key derived by Client, which is encrypted using the service's public key and sent as an encrypted key to the service inside the SOAP message. Service can decrypt the encrypted key using its private key and get the symmetric key. Then service can use that symmetric key to decrypt the message and validate the signature.

Authentication is achieved by using Username Tokens. Clients send the Username Tokens inside a message and Service validates them against the entries in its user store.

Secure Conversation - Sign only - Service as STS - Bootstrap policy - Sign and Encrypt, X509 Authentication
"Secure Conversation - Encrypt Only - Service as STS - Sign and Encrypt, X509 Authentication"

Clients should have X.509 certificates; they are establishing a security context with the service by using this certificate. RST and RSTR are signed and encrypted using PKI of service and client.

Once the security context is established, keys derived from the security context are used to sign the messages between service and client.
Secure Conversation - Sign and Encrypt - Service as STS - Bootstrap policy - Sign and Encrypt, X509 Authentication

Clients should have X.509 certificates; they are establishing a security context with the service by using this certificate. RST and RSTR are signed and encrypted using PKI of service and client.

Once the security context is established, keys derived from the security context are used to encrypt the messages between service and client.
"Secure Conversation – Sign and Encrypt – Service as STS – Sign and Encrypt, X509 Authentication"

Clients should have X.509 certificates; they are establishing a security context with the service by using this certificate. RST and RSTR are signed and encrypted using PKI of service and client.

Once the security context is established, keys derived from the security context are used to sign and encrypt the messages between service and client.
"Secure Conversation - Sign Only - Service as STS - Sign and Encrypt, Anonymous Clients"

Clients do not have X.509 certificates; they are establishing a security context with the service by using a symmetric binding.

Once the security context is established, keys derived from the security context are used to sign the messages between service and client.

Secure Conversation - Encrypt Only - Service as STS - Bootstrap policy - Sign and Encrypt, Anonymous clients
Secure Conversation - Encrypt Only - Service as STS - Bootstrap policy - Sign and Encrypt, Username Token Authentication

"Secure Conversation - Encrypt Only - Service as STS - Sign and Encrypt, Anonymous Clients"
Clients do not have X.509 certificates; they are establishing security context with the service by using a symmetric binding.

Once the security context is established, keys derived from the security context are used to encrypt the messages between service and client.
"Secure Conversation - Encrypt Only - Service as STS - Sign and Encrypt, Username Token Authentication"

Clients do not have X.509 certificates; they are establishing a security context with the service by using a symmetric binding. Username Token is used to authenticate the user when establishing the security context.

Once the security context is established, keys derived from the security context are used to encrypt the messages.
Kerberos Token-based Security

If you apply security scenario 16 (Kerberos Token-based Security), you must associate your service with a service principal. Security scenario 16 is only applicable if you have a Key Distribution Center (KDC) and an Authentication Server in your environment. Ideally you can find KDC and an Authentication Server in a LDAP Directory server.

Two configuration files are used to specify Kerberos related parameters as follows.

- **krb5.conf** - Includes KDC server details, encryption/decryption algorithms etc.
- **jaas.conf** - Includes information relevant to authorization.

The above files are located in `<PRODUCT_HOME>/repository/conf/security` folder.

After selecting scenario 16, fill information about the service principal to associate the Web service with. You must specify the service principal name and password. The service principal must be already defined in the LDAP Directory server.