Seguridad y Calidad en Aplicaciones Web

Unidad N° 4: Aplicaciones de Seguridad

Referente de Cátedra: Walter R. Ureta
Plantel Docente: Emiliano Zarate, Pablo Pomar,
Walter R. Ureta

Logging

Su finalidad se vincula a los siguientes puntos:

- Identificar incidentes de seguridad
- Monitorear violaciones a las políticas
- Asistir en controles de no-repudio
- Proveer información sobre problemas o situaciones atípicas
- Contribuir con información especifica para la investigación de incidentes que no pueda obtenerse de otras fuentes
- Contribuir con la defensa ante vulnerabilidades y exploits mediante de la detección de ataques

Logging

¿ Donde registrar ?

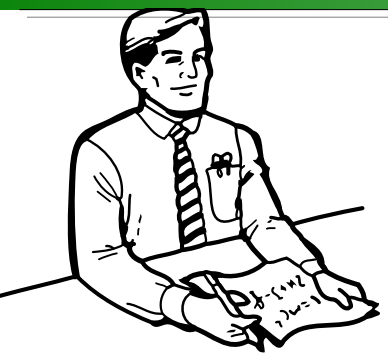
Sistemas de archivos, almacenamiento en la nube, bases de datos SQL y NoSQL

¿ Que registrar ?

Fallos de validación, autenticación, autorización, anomalías, fallas de aplicación, eventos legales...

¿ Que **NO** registrar ?

Código fuente, identificadores de sesión, credenciales y tokens de acceso, claves de cifrado, SPI...



Enmascaramiento de datos

Es el proceso mediante el cual se reemplazan los datos sensibles de un sistema con el objetivo de proteger esta información confidencial ante situaciones que escapen al control sobre la misma.

Segun GDPR, “*seudonimización*”, es el tratamiento de datos personales de manera tal que ya no puedan atribuirse a un interesado sin utilizar información adicional, siempre que dicha información adicional figure por separado y esté sujeta a medidas técnicas y organizativas destinadas a garantizar que los datos personales no se atribuyan a una persona física identificada o identificable.

Logging

Debilidades asociadas:

- CWE-117 Improper Output Neutralization for Logs
- CWE-223 Omission of Security-relevant Information
- CWE-532 Insertion of Sensitive Information into Log File
- CWE-778 Insufficient Logging

Validación de entrada

La debilidad de seguridad más común en aplicaciones web es la falta de validación apropiada de las entradas del cliente o del entorno. Esta debilidad lleva a casi todas las principales vulnerabilidades en las aplicaciones web, tales como inyección a intérprete, ataques locale/Unicode, ataques al sistema de archivos y desbordamientos de memoria.

Nunca se debe confiar en los datos introducidos por el cliente, ya que tiene todas las posibilidades de manipular los datos.

Validación de entrada

La validación debe cubrir los aspectos **semánticos** como **sintácticos** de la información ingresada.

Se sugiere implementar una metodología de validación de tipo **“White List”** para solo validar el ingreso de datos permitidos.

Para este fin se debe describir el tipo válido de datos, se sugiere implementarlo mediante listas de valores o expresiones regulares.

Validación de entrada

Expresiones regulares de validación de entrada por tipos:

URL:^((((https?|ftps?|gopher|telnet|nntp)://)|(mailto:|news:))(%[0-9A-Fa-f]{2}|[-()_.!~*';/?:@&=+$,A-Za-z0-9])+)([).!';/?:,][[:blank:]])?$

IP:^(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)$

EMAIL:^[a-zA-Z0-9+&*-]+(?:\.[a-zA-Z0-9_+&*-]+)*@(?:[a-zA-Z0-9-]+\.)+[a-zA-Z]{2,7}$

NUMEROS:^(cero|uno|dos|tres|cuatro|cinco|seis|siete|ocho|nueve)$

Autenticación en la Web

Su objetivo es proveer servicios de autenticación segura a las aplicaciones Web, mediante:

- Vinculando una unidad del sistema a un usuario individual mediante el uso de una credencial
- Proveyendo controles de autenticación razonables de acuerdo al riesgo de la aplicación.
- Denegando el acceso a atacantes que usan varios métodos para atacar el sistema de autenticación.

Autenticación en la Web

Consideraciones generales

- La autenticación es solo tan fuerte como sus procesos de administración de usuarios
- Use la forma más apropiada de autenticación adecuada para su clasificación de bienes
- Re-autenticar al usuario para transacciones de alto valor y acceso a áreas protegidas
- Autenticar la transacción, no el usuario
- Las contraseñas son trivialmente rotas y no son adecuadas para sistemas de alto valor

Autenticación en la Web

Buenas practicas

1) User IDs: Verificar que NO son case sensitive, para evitar confusiones del tipo "*juan*" o "*Juan*".
2) Fortaleza de contraseñas: su longitud mínima debería ser de al menos 10 caracteres y la máxima no inferior a 20, con recomendación de 128 caracteres. Evaluar su complejidad.
3) Implementar métodos seguros de recuperación: Con canales externos o diversos puntos de información no selectivos (preguntas de seguridad, de texto libre).
4) Almacenar contraseñas de forma segura: con un soporte criptográfico adecuado
5) Transmitir contraseñas solo sobre TLS: la pagina de logueo y todas las posteriores deben operar bajo TLS.

Autenticación en la Web

6) Solicitar re-autenticación: Para evitar ataques de CSRF y Hijacking de sesión se debe solicitar autenticación nuevamente antes de realizar operaciones sensibles.

7) Utilizar sistemas de autenticación de factor múltiple: Son aquellos que se comprenden como autenticación fuerte, estando compuestos por más de un elemento de los siguientes tipos

- Algo que se sabe...
- Algo que se tiene...
- Algo que se es…
- Ubicación...

8) Manejo de mensajes de error: los mensajes de error deben ser genéricos a fin de no facilitar ningún indicio o información de los datos de autenticación del sistema.

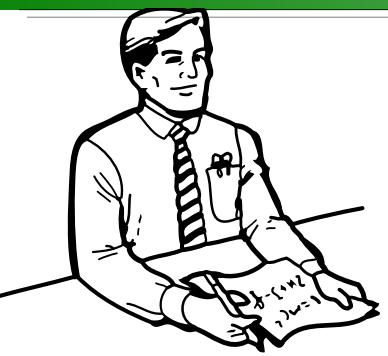
9) Prevenir ataques por fuerza bruta: mediante captchas y controles de comportamiento (tiempo entre intentos, cantidad, etc)

Autenticación en la Web

Métodos de protección ante ataques de automatización:

- MFA o Autenticación de factor múltiple
- Bloqueo de cuenta
- CAPTCHA
- Preguntas de seguridad o palabras a memorables

***Brute Force**: Probar múltiples valores de contraseñas de un diccionario contra para una única cuenta.*
***Credential Stuffing**: Probar combinaciones usuario-contraseña obtenidas de otras brechas de seguridad.*
***Password Spraying**: Probar una contraseña débil sobre un gran numero de cuentas.*



Contraseñas de un solo uso

OTP

"One Time Password" o *"Contraseña de un solo uso"*, corresponde a un valor confidencial que no puede ser reutilizado.

HOTP

Algoritmo de generación de "Contraseñas de un solo uso basadas en HMAC". *RFC-4226*

TOTP

"Time-based One Time Password" o "Contraseña de un solo uso basada en tiempo", es un valor confidencial que no puede ser reutilizado y que ademas cuenta con un tiempo de vida acotado. *RFC-6238*

Técnicas de autenticación de Usuarios

1)Autenticación básica y segura *(HTTP-Basic, HTTP-Digest)*

2)Autenticación basada en formularios *(Usuario-Contraseña)*

3)Autenticación integrada *(ISS, ASP.NET, Active Directory)*

4)Autenticación basada en certificado *(x509)*

5)Autenticación fuerte o de factor múltiple *(Algo que sabes, algo que tienes, algo que eres, tu ubicación)*

Autenticación en la Web

- Re-uso de contraseñas
- Autenticación sin contraseñas

	OAuth2	***OpenID***	***SAML***
Formato del token	JSON o SAML2	JSON	XML
Autorización	Si	No	Si
Autenticación	Si	Si	Si
Version	Oauth2 (2005)	OpenClient (2006)	SAML 2.0 (2001)
Notas	Basado en TLS	-	-
Aplicación	Autorización de API	SSO (Consumidor)	SSO (Empresarial)

Autenticación en la Web (Contraseña olvidada)

- Utilizar un mensaje único ya sea que la cuenta exista o no.
- Asegurar que el tiempo de respuesta al usuario sea uniforme.
- Utilizar otro canal para informar el método de re-seteo de la contraseña.
- Utilizar tokens con URLs para una implementación simple y rápida
- Asegurar que los tokens sean aleatorios, de longitud que provea resistencia a fuerza-bruta, almacenamiento seguro, con tiempo de expiración y de un solo uso.
- No alterar la cuenta del usuario hasta la presentación del token.

Autorización en la Web

Sus objetivos son

- Asegurar que únicamente usuarios autorizados puedan realizar acciones permitidas con su correspondiente nivel de privilegio.
- Controlar el acceso a recursos protegidos mediante decisiones basadas en el rol o el nivel de privilegio.
- Prevenir ataques de escalada de privilegios, como por ejemplo utilizar funciones de administrativas siendo un usuario anónimo o incluso un usuario autenticado.

Autorización en la Web

Métodos de control de acceso

1. Role Based Access Control (**RBAC**)
2. Discretionary Access Control (**DAC**)
3. Mandatory Access Control (**MAC**)
4. Attribute Based Access Control (**ABAC**)

Autorización en la Web

En **Role Based Access Control (RBAC)**, las decisiones de acceso se basan en las funciones y responsabilidades de un individuo dentro de la organización o de la base de usuarios. El proceso de definición de las funciones se basa por lo general en el análisis de los objetivos y la estructura de una organización que por lo general está relacionada con la política de seguridad. Por ejemplo, en una organización médica, los diferentes roles de los usuarios pueden incluir aquellos como médico, enfermera, asistente, enfermera, pacientes, etc. Estos miembros requieren diferentes niveles de acceso para llevar a cabo sus funciones, sino también los tipos de las transacciones Web y su contexto permite variar mucho dependiendo de la política de seguridad y los reglamentos pertinentes (HIPAA, Gramm-Leach-Bliley, etc.)

Autorización en la Web

El **Discretionary Access Control (DAC)** es un medio para restringir el acceso a la información sobre la base de la identidad de los usuarios y/o la pertenencia a ciertos grupos. Decisiones de acceso se basan normalmente en las autorizaciones concedidas a un usuario basándose en las credenciales que presentó en el momento de la autenticación (nombre de usuario, contraseña, hardware / identificador de software, etc.) En los modelos más típicos del DAC, el propietario de la información o cualquier recurso es capaz de cambiar los permisos a su discreción (de ahí el nombre). DAC tiene el inconveniente de que los administradores no son capaces de gestionar de forma centralizada los permisos de los archivos / datos almacenados en el servidor web. Por ejemplo, el sistema de archivos de un sistema Unix (rwx).

Autorización en la Web

Mandatory Access Control (MAC) garantiza que la ejecución de la política de seguridad de la organización no se basa en el cumplimiento del usuario en la aplicación web. MAC asegura la información mediante la asignación de etiquetas de sensibilidad en la información y comparando esto con el nivel de sensibilidad de un usuario está operando a. En general, los mecanismos de control de acceso MAC son más seguras que las DAC todavía tener soluciones de compromiso en el rendimiento y la conveniencia de los usuarios. Mecanismos MAC asignan un nivel de seguridad a toda la información, asignar un control de seguridad de cada usuario, y garantizar que todos los usuarios sólo tengan acceso a los datos pertinentes. MAC es generalmente apropiado para sistemas extremadamente seguras como aplicaciones militares seguras multinivel o aplicaciones de datos de misión crítica.

Autorización en la Web

Attribute Based Access Control (ABAC) Es un sistema de control basado en atributos asignados a cualquier componente del sistema (usuarios, recursos, etc). El mismo establece políticas que utilizan la información de dichos atributos para establecer si un acceso esta permitido, de esta forma se establece un sistema de control basado en relaciones. Los atributos son del tipo clave-valor, generalmente se utiliza un lenguaje de soporte como XACML para definir las reglas de acceso. Este modelo es muy utilizado en las plataformas actuales de Cloud.

Ejemplo: *"Los usuarios pueden acceder a los servicios del proyectoA si pertenecen al grupo de DesarrolloA"*. Aquí vemos que la política requiere tres atributos rol:usuario, servicio:proyectoA, grupo:desarrolloA.

IAM (Identity Access Management) es un sistema de políticas y procesos para controlar el acceso a sistemas de información

Autorización en la Web

Buenas prácticas de implementación

- Codificar el control en la actividad objetivo
- Disponer de un Controlador Centralizado (ACL)
- Utilizar un Control Central de Acceso, en las diferentes capas
- Verificar la política del lado del servidor (Server-side)

Autorización en la Web

Ataques de control de acceso

1. **Vertical Access Control Attacks** - Un usuario convencional obtiene accesos superiores o de administrador.
2. **Horizontal Access Control attacks** - Con el mismo rol o nivel el usuario puede acceder a información de otros usuarios.
3. **Business Logic Access Control Attacks** - Abusar de una o más actividades para realizar una operación con un resultado no autorizado para ese usuario.

Administración de Usuarios y privilegios

Objetivos

- Las funciones de nivel de administrador están segregadas apropiadamente de la actividad del usuario
- Los usuarios no pueden acceder o utilizar funcionalidades administrativas
- Proveer la necesaria auditoria y trazabilidad de funcionalidad administrativa

Administración de Usuarios y privilegios

Mejores prácticas

- Cuando se esta diseñando aplicaciones, trazar la funcionalidad administrativa fuera y asegurarse que los controles apropiados de acceso y auditoría están en su lugar
- Considerar procesos – en algunas ocasiones todo lo que se requiere es entender cómo los usuarios pueden ser prevenidos de utilizar una característica con la simple falta de acceso
- Acceso de servicio de asistencia es siempre un término medio – ellos necesitan acceso para ayudar a los clientes, pero no son administradores.
- Diseñar cuidadosamente la funcionalidad de servicio de asistencia / moderador / soporte al cliente alrededor de una capacidad administrativa limitada y aplicación segregada o acceso.

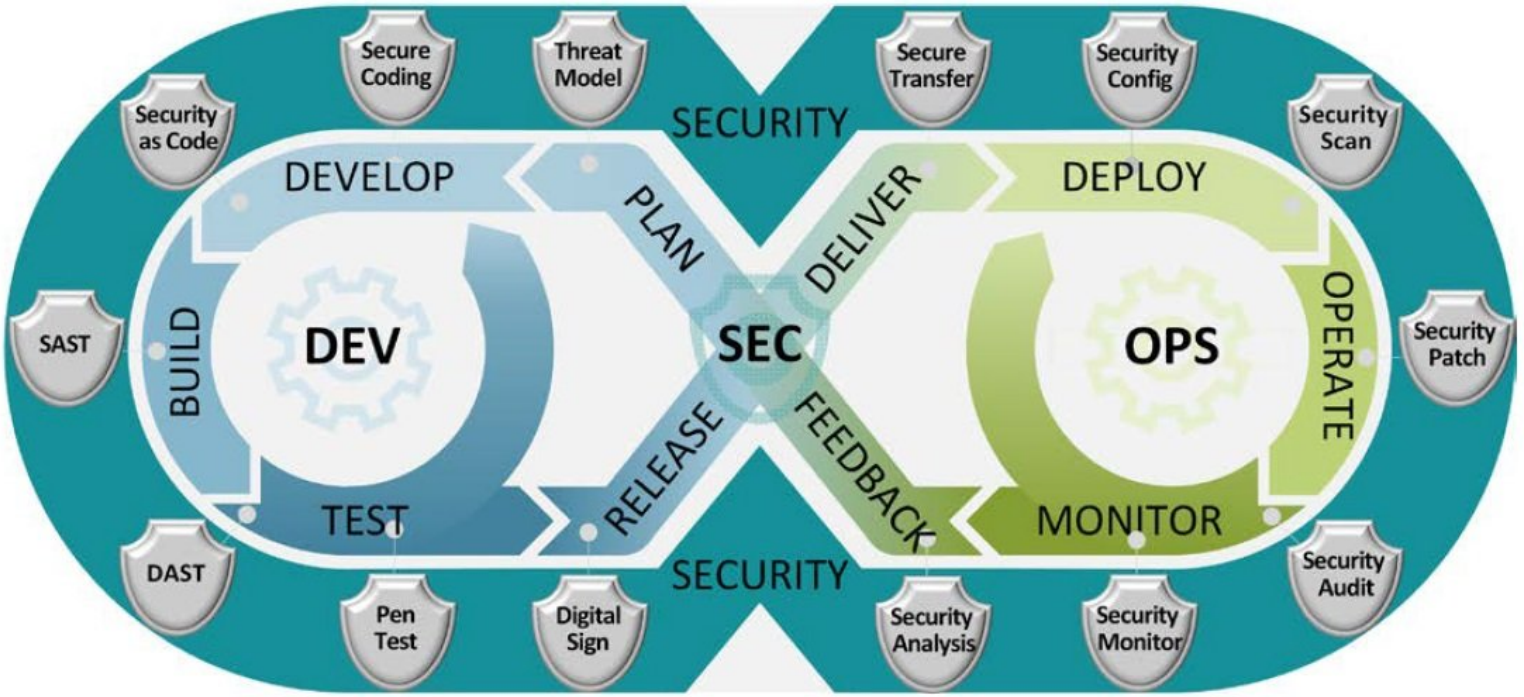
Administración de Usuarios y privilegios

- Todos los sistemas deberían tener aplicaciones separadas del acceso de los usuarios para los administradores.

- Sistemas de alto valor deberían separar estos sistemas en un servidor separado, que tal vez no sea accesible desde el amplio Internet sin acceso para la administración de redes, como a través del uso de una VPN fuertemente autenticada o desde la red de un centro de operaciones de confianza.

DevSecOps

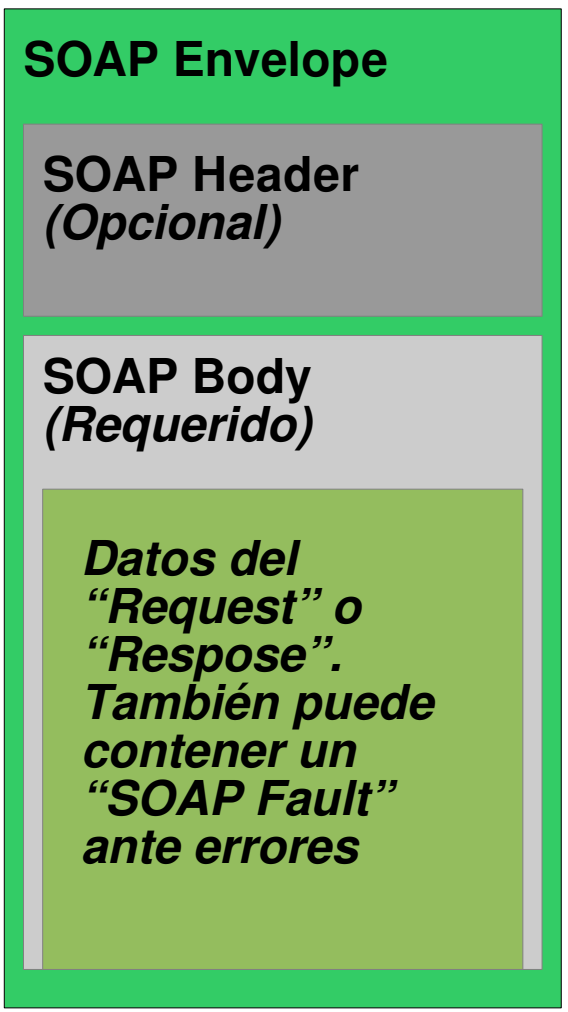
Es un conjunto de practicas que combinan el desarrollo de software (Dev), la seguridad (Sec), y las operaciones de tecnología de la información (Ops) para asegurar y acortar el ciclo de vida del desarrollo de software.



Fuente: DoD, https://public.cyber.mil/devsecops

Web Services

En el nivel más simple, los servicios web pueden ser vistos como aplicaciones web especializadas que difieren principalmente en la capa de presentación. Mientras que las aplicaciones web son típicamente basadas en HTML, los servicios web son basados en XML/SOAP.



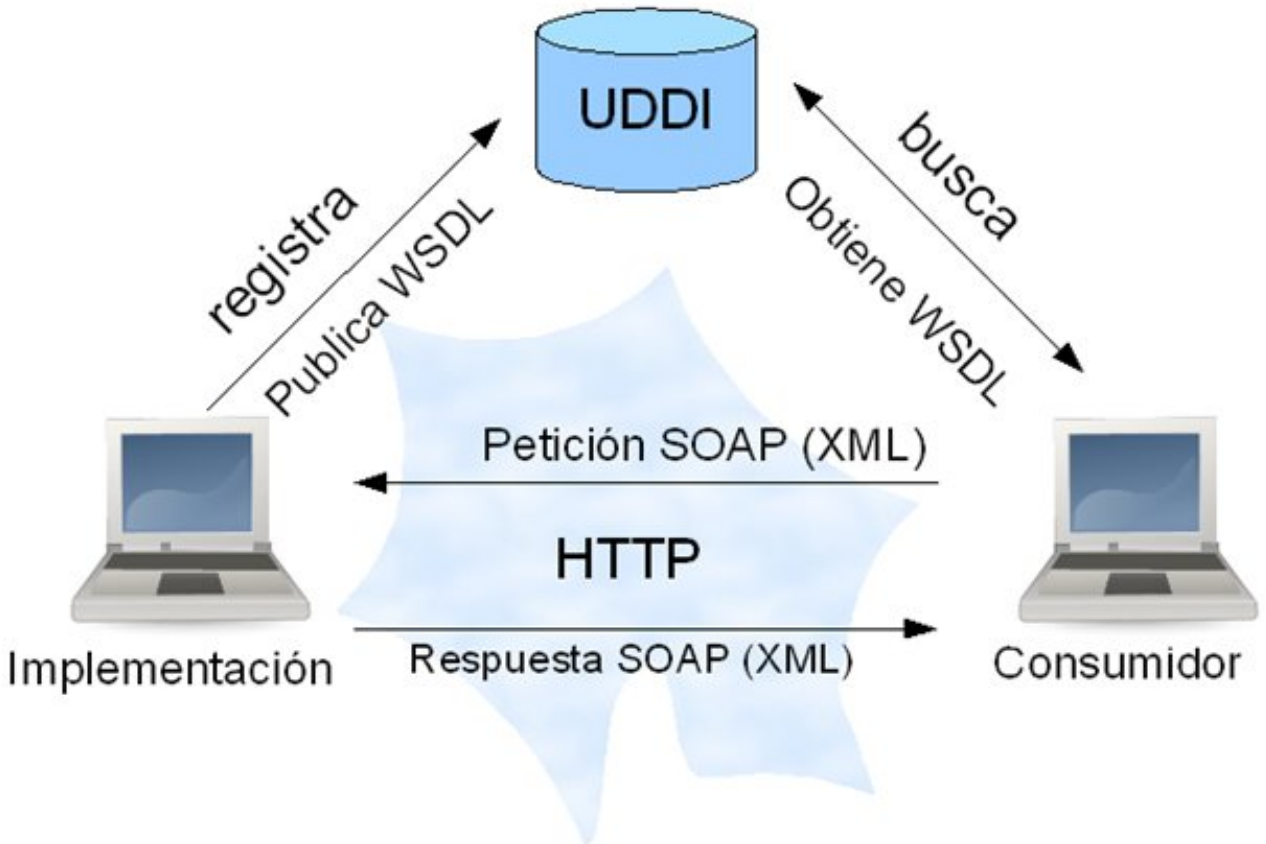
Web Services

Comités de estándares

- W3C (http://www.w3.org), estándares de esquema XML (XML Schema), SOAP (Simple Object Access Protocol), XML-dsig, XML-enc y WSDL
- OASIS (http://www.oasis-open.org), estándares de WS-Security,
- OASIS (http://uddi.xml.org), UDDI (Universal Description, Discovery and Integration)
- OASIS (http://saml.xml.org), SAML (Security Assertion MarkUp Language).
- Grupo de interoperabilidad de servicios web (WS-I - http://www.ws-i.org/)

Web Services

Los servicios Web típicamente representan una interfaz pública funcional, que se llama de forma programática



Web Services

Definition WSDL - Parte 1

```
<definitions name="HelloService" targetNamespace="http://www.examples.com/wsdl/HelloService.wsdl"
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:tns="http://www.examples.com/wsdl/HelloService.wsdl"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">

  <message name="helloRequest">
      <part name="name" type="xsd:string"/>
  </message>
  <message name="helloResponse">
      <part name="return" type="xsd:string"/>
  </message>

  <portType name="hello_PortType">
      <operation name="hello">
      <input message="tns:helloRequest"/>
      <output message="tns:helloResponse"/>
      </operation>
  </portType>
```

Web Services

Definition WSDL - Parte 2

```
<binding name="hello_Binding" type="tns:hello_PortType">
<soap:binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
<operation name="hello">
    <soap:operation soapAction="hello"/>
    <input>
    <soap:body
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
    namespace="urn:examples:helloservice"
    use="encoded"/>
    </input>
    <output>
    <soap:body
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
    namespace="urn:examples:helloservice"
    use="encoded"/>
    </output>
</operation>
</binding>
```

Web Services

Definition WSDL - Parte 3

```
  <service name="hello_Service">
      <documentation>WSDL for Hello Web Service</documentation>
      <port binding="tns:hello_Binding" name="hello_Port">
      <soap:address
      location="http://server.net/hello/">
      </port>
  </service>
</definitions>
```

Web Services

SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
    <S:Header/>
    <S:Body>
        <ns2:hello xmlns:ns2="http://org/">
            <name>Mundo</name>
        </ns2:hello>
    </S:Body>
</S:Envelope>
```

Web Services

SOAP Response

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
    <S:Body>
        <ns2:helloResponse xmlns:ns2="http://org/">
            <return>Hola Mundo !</return>
        </ns2:helloResponse>
    </S:Body>
</S:Envelope>
```

Web Services

Implementacion en Java con Java EE 5, JAX-WS 2.0

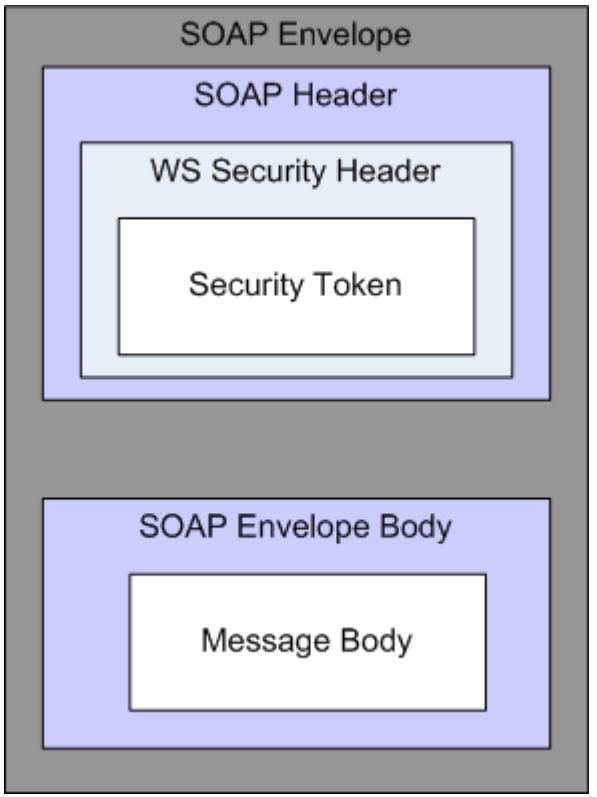
```
package org;
import javax.jws.WebService;
import javax.jws.WebMethod;
import javax.jws.WebParam;

@WebService(serviceName = "NewWebService")
public class NewWebService {
    @WebMethod(operationName = "hello")
    public String hello(@WebParam(name = "name") String txt) {
        return "Hola " + txt + " !";
    }
}
```

Web Services - WS-Security - WSS

El estándar WSS lidia con varias áreas medulares de seguridad, dejando muchos detalles a los llamados documentos perfil. Las áreas principales, ampliamente definidas por el estándar son:

- Maneras de agregar encabezados de seguridad (encabezados WSSE) a los sobres de SOAP
- Adjuntar testigos de seguridad y credenciales al mensaje
- Insertando un estampado de tiempo
- Firmar el mensaje
- Cifrado del mensaje
- Extensibilidad



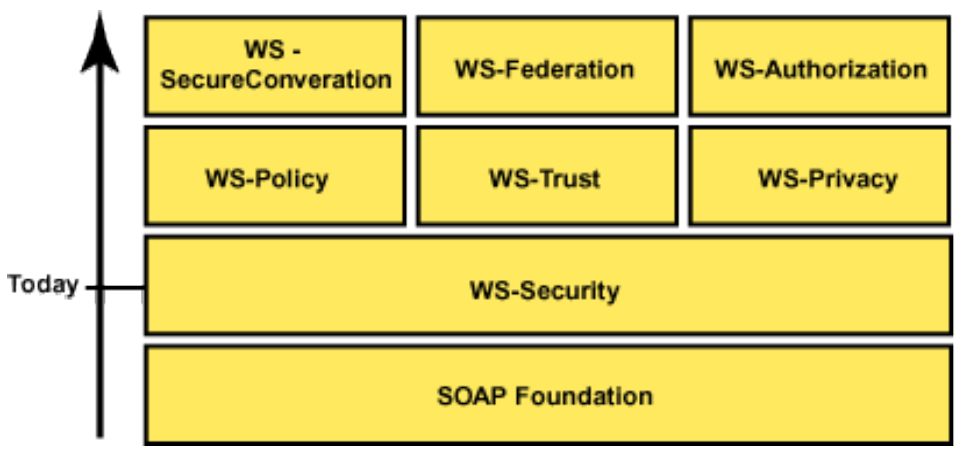
Web Services - WS-Security - WSS

```
<?xml version="1.0" encoding="utf-8"?>
<S11:Envelope xmlns:S11="..." xmlns:wsse="..." xmlns:wsu="..." xmlns:ds="...">
        <S11:Header>
        <wsse:Security xmlns:wsse="...">
        <xxx:CustomToken wsu:Id="MyID" xmlns:xxx="http://fabrikam123/token">
                FHUIORv...
        </xxx:CustomToken>
        <ds:Signature>
                <ds:SignedInfo>
                        <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
                        <ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#hmac-sha1"/>
                        <ds:Reference URI="#MsgBody">
                                <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                                <ds:DigestValue>LyLsF0Pi4wPU...</ds:DigestValue>
                        </ds:Reference>
                </ds:SignedInfo>
                <ds:SignatureValue>DJbchm5gK...</ds:SignatureValue>
                <ds:KeyInfo>
                        <wsse:SecurityTokenReference>
                                <wsse:Reference URI="#MyID"/>
                        </wsse:SecurityTokenReference>
                </ds:KeyInfo>
        </ds:Signature>
        </wsse:Security>
        </S11:Header>
        <S11:Body wsu:Id="MsgBody">
                <tru:StockSymbol xmlns:tru="http://fabrikam123.com/payloads">
                QQQ
                </tru:StockSymbol>
        </S11:Body>
</S11:Envelope>
```

Web Services - WS-Security - WSS

WS-Policy: Describe capacidades y limitaciones de la seguridad, politicas e intermediarios (reglas de seguridad, algoritmos soportados, etc)
WS-Trust: Describe un framework para para facilitar la interoperación de WS en forma segura.
WS-Privacy: Describe el modelo sobre cómo los Web Services manejan las peticiones y preferencias de seguridad
WS-SecureConversation: Describe como manejar y autenticar el intercambio de mensajes, el contexto de seguridad y claves de sesión.

WS-Federation: Describe cómo administrar y manejar la relaciones de confianza entre sistemas federados.
WS-Authorization: Describe cómo administrar la autorización de datos y políticas.

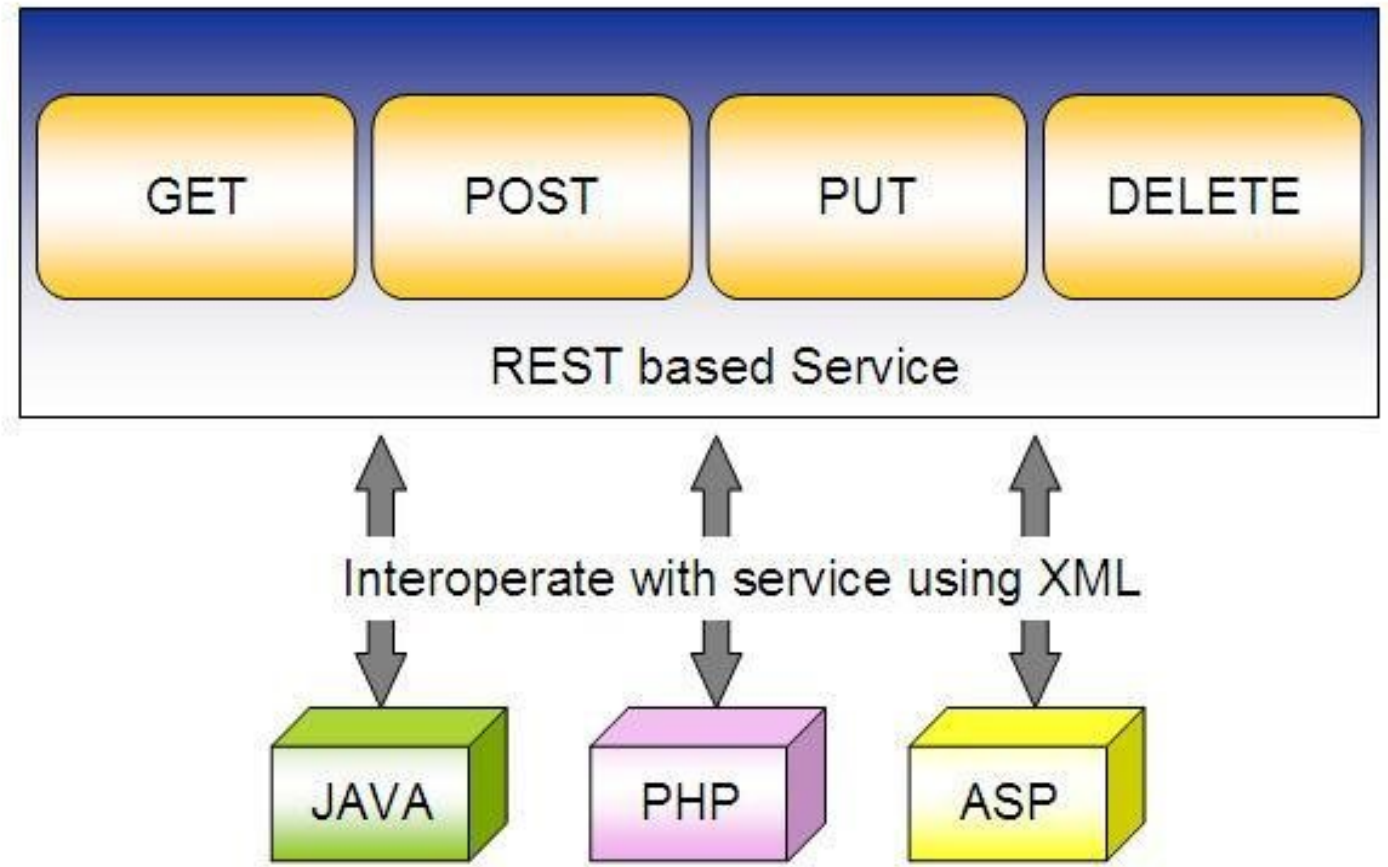


ReST

Es una técnica de arquitectura para sistemas distribuidos, su nombre es un acrónimo que deriva de "*Representational State Transfer*". Su origen data del año 2000 siendo definido en una tesis doctoral de Roy Fielding.

Su objetivo fue evitar el uso de métodos complejos como CORBA, RPC y SOAP para interconexión de sistemas; con este fin las aplicaciones ReSTFul usan llamados **HTTP** para las operaciones **CRUD (Create/Read/Update/Delete)** orientando su diseño a elementos en lugar de operaciones.

ReST



Mensajes implementados en formato Plano, XML o JSON

ReST

Request

http://server.net/hola/Mundo

Plain Response

Hola Mundo!

XML Response

<datos>Hola Mundo!</datos>

ReST

Implementacion en Java con Jersey, JAX-RS

```
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.Produces;
 @Path("/hola/{username}")
public class Hola {
        @GET
        @Produces("text/plain")
        public String sayHola(@PathParam("username") String userName) {
                return "Hola “ + userName + ”!";
        }
        @GET
        @Produces("application/xml")
        public String sayHolaXml(@PathParam("username") String userName) {
                return "<datos>Hola “ + userName+ ”!</datos>";
        }
}
```

ReST

WADL - Web Application Description Language

```
<resources base="http://example.com/widgets">
  <resource path="{widgetId}">
    <param name="customerId" style="query"/>
    <method name="GET">
      <request>
        <param name="verbose" style="query" type="xsd:boolean"/>
        <param name="text" style="query" type="xsd:string"/>
      </request>
       <response status="200">
          <representation mediaType="application/xml"  element="yn:ResultSet"/>
     </response>
   </method>
 </resource>
</resources>
```

http://www.w3.org/Submission/wadl/

Material de Referencia

Logging

- Hoja de referencia:

https://cheatsheetseries.owasp.org/cheatsheets/Logging_Cheat_Sheet.html

- NIST SP800-92 Guide to Computer Security Log Management

http://csrc.nist.gov/publications/nistpubs/800-92/SP800-92.pdf

Validación

- Hoja de referencia:

https://cheatsheetseries.owasp.org/cheatsheets/Input_Validation_Cheat_Sheet.html#Email_Address_Validation

Material de Referencia

Autenticación

- Hoja de referencia general sobre autenticación:
https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html
- NIST SP 800-62b Digital Identity Guidelines, Authentication and Lifecycle Management:
https://pages.nist.gov/800-63-3/sp800-63b.html
- Hoja de referencia para procedimientos de contraseñas olvidadas:
https://cheatsheetseries.owasp.org/cheatsheets/Forgot_Password_Cheat_Sheet.html
- Hoja de referencia para MFA:
https://cheatsheetseries.owasp.org/cheatsheets/Multifactor_Authentication_Cheat_Sheet.html