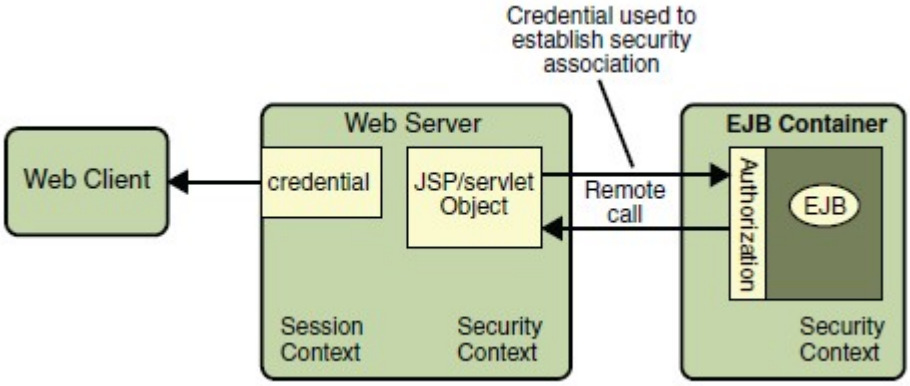
LIYANA ARACHCHIGE RANIL

SHORT NOTES / SECURITY – CONTAINERS and OTHERS

1. DECLARATIVE security expresses an APPLICATION COMPONENTS security requirements using DEPLOYMENT DESCRIPTORS
2. Annotation (METADATA) are used to specify information about security within the CLASS file. When the application is deployed , this information can either be used by or overridden by the application deployment descriptor
3. PROGRAMMATIC security is EMBEDDED in an application and is used to make security decisions. PROGRAMMATIC SECURITY is useful when DECLARATIVE security alone is not SUFFICIENT to express the security model of an application
4. Invoking Enterprise bean Business Method
5. 

6. Characteristics of well defined security
    a. Authentication , Authorization or Access Control, Data Integrity , Confidentiality or Data Privacy , Non-Repudiation ,Quality of Service , Auditing
7. JAAS – Java Authentication and Authorization Service: A set of API that enable services to Authenticate and Enforce access controls upon users. JAAS provides pluggable and extensible framework for PROGRAMMETIC and USER AUTHENTICATION

and AUTHORIZATION. JAAS is a core java SE API an underlying technology for java EE security mechanisms

8. Java GSS-API: Java Generic Security Services: JAVA GSS-API is a token based API used to securely exchange messages between communicating applications.
9. JAVA GENERIC SECURITY SERVICE contains KERBEROS V5 as a MANDATORY support. JSSE does not support Kerberos since KERBEROS is not standardized for TLS protocol
10. JSSE supports SOCKET based API , if you application is communicating using SOCKETS then JSSE more appropriate. JAVA-GSS is on the other hand users TOCKEN BASED APPROACH. This means that the application can use TCP sockets , UDP datagram or any other CHANNEL that will allow it to transport JAVA GSS TOCKENS. If you application has varying communication protocol needs then JAVA-GSS is good for that
11. Since JAVA-GSS is token based you can use selective ENCRYPTION for certain messages
12. JCE – Java Cryptographic Extension – JCE provides a framework and implementations for ENCRYTION, Key generation and key agreement, and Message Authentication Codes (MAC). Supports for encryption included asymmetric , symmetric , block , stream ciphers
13. JSSE – Java Secure Socket Extension : JSSE provides a framework and an implementation for a JAVA version of SSL and TLS protocol and including functionality of Data Encryption , Server Authentication , Message Integrity ,and optional Client authentication to enable secure internet communication

14. SASL – Simple Authentication and Security Layer : is an internet standard that specifies s protocol for authentication and optional establishment of a security later between client and server applications
15. in java EE there are , APPLICATION LAYER SECURITY , TRANSPORT LATER SECURITY , MESSAGE LAYER SECURITY
16. Application Layer security provides security services for a SPECIFIC APPLICATION
17. In java EE , component containers are responsible for providing Application layer security
18. The application is dependent on security attributes that are not transferable between application types
19. TRANSPORT LATER SECURITY is provided by the transport mechanism used to transmit information over the wire between the client and providers
20. Thus TRANSPORT LAYER SECURITY relies on secure HTTP transport (HTTPS) using SSL
21. TRANSPORT SECURITY is a POINT TO POINT security mechanism that can be used for AUTHENTICATION , MESSAGE INTERGRITY and CONFIDENTIALITY
22. The problem with this is that it is not protected once it gets to its destination, once solution for this is to encrypt the message
23. TRANSPORT LAYER SECURITY is performed in a series of phases,
    a. The client and server AGREES on an appropriate algorithm
    b. A key is exchanged using public key encryption and certificate based authentication

c. A symmetric cipher is used during the information exchange

24. DIGITAL CERITIFICATES are NECESSARY when running secure HTTP (HTTPS) using SSL
25. The security provided by this is only POINT TO POINT , it is NOT and END TO END solution
26. MESSAGE LATYER SECURITY, the security information is contained in the SOAP message itself. This allows security information to travel along with the message or attachment. Message layer security is also known as END TO END SECURITY
27. Message security is independent from application environment and transport protocol
28. The disadvantage is it is relatively COMPLEX and ass some OVERHEAD in PROCESSING
29. DECLARATIVE SECURITY is expressed using Deployment Descriptors. Different types of COMPONENTS have different formats or schemas for their deployment descriptors.
30. EJB – deployment descriptor must be named **ejb-jar.xml** inside META-INF folder
31. WS – uses JAX-RPC mapping info file. This deployment descriptor provides deployment time mapping functionality between Java and WSDL. JAX-WS implements this functionality with development time ANNOTATIONS
32. WEB COMPONENT – uses web.xml file as deployment descriptor.

```
<security-constraint>
    <web-resource-collection>
        <web-resource-name>view dept data</web-resource-name>
        <url-pattern>/hr/employee/*</url-pattern>
        <http-method>GET</http-method>
        <http-method>POST</http-method>
    </web-resource-collection>
    <auth-constraint>
        <role-name>DEPT_ADMIN</role-name>
    </auth-constraint>
    <user-data-constraint>
        <transport-guarantee>CONFIDENTIAL</transport-guarantee>
    </user-data-constraint>
</security-constraint>

<security-constraint>
    <web-resource-collection>
        <web-resource-name>change dept data</web-resource-name>
        <url-pattern>/hr/employee/*</url-pattern>
        <http-method>GET</http-method>
        <http-method>PUT</http-method>
    </web-resource-collection>
    <auth-constraint>
        <role-name>DIRECTOR</role-name>
    </auth-constraint>
    <user-data-constraint>
        <transport-guarantee>CONFIDENTIAL</transport-guarantee>
    </user-data-constraint>
</security-constraint>
```

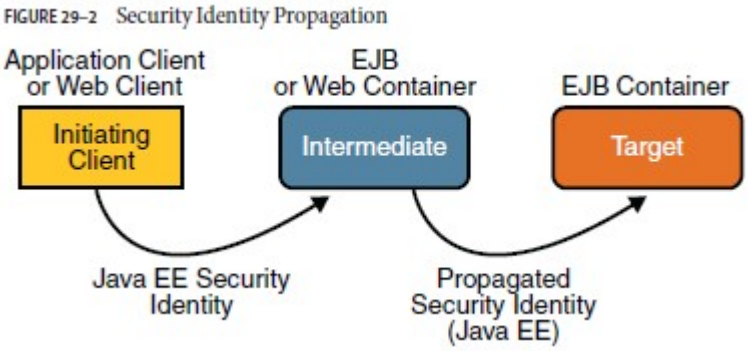
33.

34. Security ROLES can be defined as above in the XML file
35. Usually these SECURITY ROLES are mapped to USERS or GROUPS
36. In J2EE platform ,it provides a way to MAP ROLES defined in the APPLICATION to USERS or GROUPS in the RUNTIME environment

```
<sun-web-app>
    <security-role-mapping>
        <role-name>DIRECTOR</role-name>
        <principal-name>mcneely</principal-name>
    </security-role-mapping>
    <security-role-mapping>
        <role-name>MANAGER</role-name>
        <group-name>manager</group-name>
    </security-role-mapping>
</sun-web-app>
```

37.

38. To MAP security ROLES defined in the APPLICATION or MODULE to USERS and GROUPS in the SERVER , the above XML can be used (For Sun Server)
39. ROLE name can be MAPPED to EITHER a PRINCIPAL(User) , a GROUP or BOTH
40. PRINCIPALS and GROUP names must be valid one in the CURRENT REALM of the SERVER
41. ANNOTATIONS enable DECLARATIVE style of programming
42. NOT all security information can be specified by ANNOTATIONS
43. It is not POSSIBLE to assign MULTIPLE certificates to a SINGLE IP ADDRESS and this is a design limitation of SSL protocol
44. For ENTERPRISE BEANS you can use USERNAME-PASSWORD authentication mechanism , for WEB SERVICE END POINTS you can use BASIC or CLIENT-CERT
45. It can be specified whether the callers SECURITY IDENTITY should be used for the execution of specified method of an ENTERPRISE BEAN , or whether a SPECIFIC RUN-AS identity should be used



46.

47. The security identity of the FIRST call is the identity of the Caller

48. The security identity in the second call can be

    a. By DEFAULT, the identity of the CALLER of the INTERMEADIATE COMPONENT is PROPAGATED to the TARGET enterprise bean. This technique is used when the target container TRUSTS the intermediate container

```
<security-identity>
    <use-caller-identity />
</security-identity>
```

    b.

    c. A SPECIFIC identity is propagated to the TARGET EJB. This technique is used when the target container EXPECTS access using a specific identity

    d. To propagate an identity to the target EJB , configure RUN AS identity

    e. Establishing RUN AS identity for an enterprise bean DOES NOT AFFECT the identities of its callers, which are the identities tested for permission to access the METHODS of the EJB

```
...
<enterprise-beans>
  ...
  <session>
    <ejb-name>EmployeeService</ejb-name>
    ...
    <security-identity>
      <run-as>
        <role-name>admin</role-name>
      </run-as>
    </security-identity>
    ...
  </session>
  ...
</enterprise-beans>
...
```

f.

g. The RUN AS identity establishes the identity that the EHB will used when it makes CALLS

h. The RUN AS identity applies to the EJB as a WHOLE

49. TRUST BETWEEN CONTAINERS

a. When an EJB is designed so that either the ORIGINAL CALLER identity or a DESIGNATED identity is used to call a target bean, the target bean will RECECIVE the PROPAGATED identity only. It will not RECEIVE any AUTHENTICATION DATA

b. There is NO WAY for the TARGET container to AUTHENTICATE the propagated SECURITY IDENTITY. However because the security identity is used in AUTHORIZATION checks it is VITALLY important that the security IDENTITY BE AUTHENTIC

c. Target would trust that the CALLING container has propagated an AUTHENTICATED security identity

d. By DEFAULT the APPLICATION SERVER is configured to TRUST identities that are propagated from different

CONTAIENRS. There for there are no special steps that you need to take to set up a trust relationship

50. in EIS applications , components request a connection to an EIS resource. As part of this connection , the EIS can REQUIRE a SING ON for the requested to access the resource. The application component provider has two choices for the design of the EIS sign on
    a. IN the container managed SIGN ON approach , the application component lets the container take the responsibility of configuring and managing EIS sign on. The container determine the username and password for establishing a connection to an EIS instance

```
// Invoke factory to obtain a connection. The security
// information is not passed in the getConnection method
javax.resource.cci.Connection cx = cxf.getConnection();
...
```

    b.
    c. In the component managed sign on approach , the application component code manages EIS sign on by including code that performs the sign on process to an EIS

```
// Invoke factory to obtain a connection
properties.setUserName("...");
properties.setPassword("...");
javax.resource.cci.Connection cx =
     cxf.getConnection(properties);
...
```

    d.

51. Configuring RESOURCE ADAPTOR security
    a. Need to edit ra.xml
    b. You can specify the Authentication mechanism supported by the resource adaptor. This support is for the resource adaptor , not for the underlying EIS instance

c. There are two mechanisms supported , **BasicPassword** and **Kerbv5**