

IT-Security: WebLogic Server and Oracle Platform Security Services (OPSS)

IT security is popular in a way never known before! I love it!

If I discussed e.g. in a WebLogic Server workshop about that, I heard normally from administrators: That's not my thing, forget it! But newly, everybody wants to know "how can we secure our data and our information?!" To be honest, you need to detect your application server that you are using, and if you are not able to use WebLogic Server security features, then this could be your problem.

WebLogic Server uses a security architecture that provides a unique and secure foundation for applications that are available via the Web. It is designed for a flexible security infrastructure and enabled to response the security challenges on the Intra- and Internet. We are able to use security capacity of WebLogic Server as a standalone feature to secure WebLogic Server and/or as part of a corporation-wide, security management system.

Overview

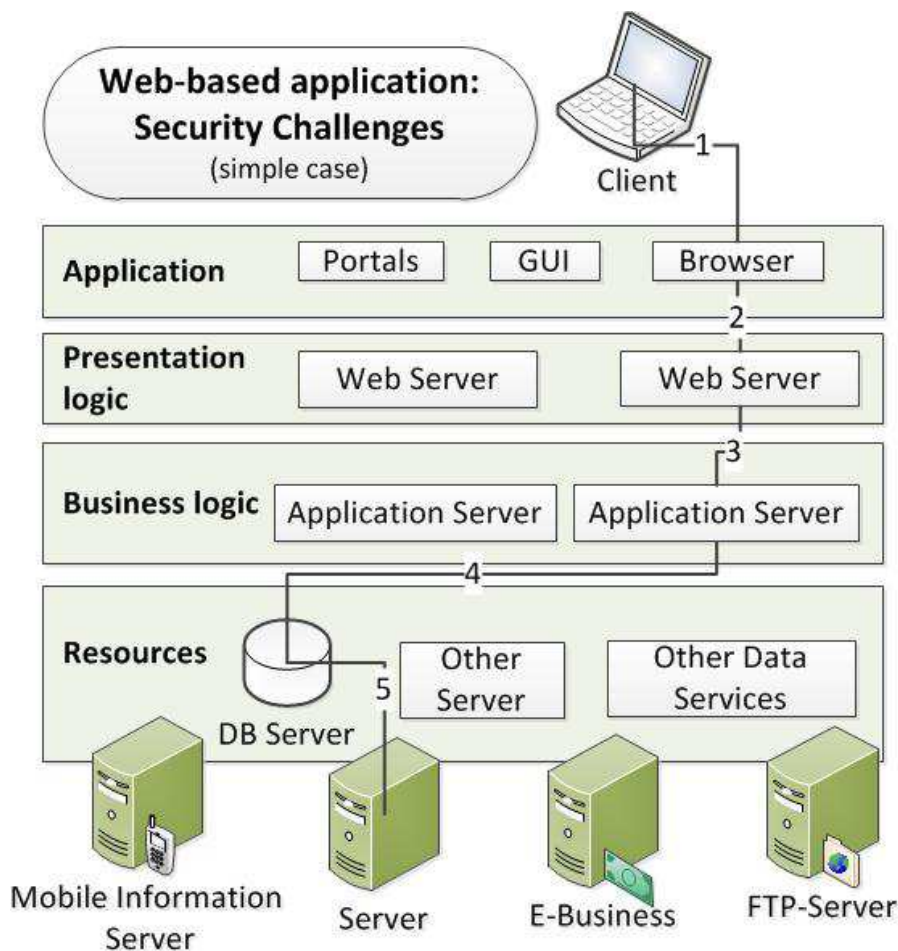
In order to achieve a satisfactory level of security, we have to design an integrated security policy: from lack of resources till the increasing complexity of IT systems. The elementary principles in IT security are Confidentiality and/or privacy, availability and integrity. Confidentiality and/or privacy mean information that has to be protected against unauthorized disclosure. Availability means services; IT system functions and information must be available to users when they need it. Integrity means data must be complete and unaltered. Therefore, we understand security policy as a policy that it covers protection objectives and broad-spectrum security measures in the sense of the acknowledged requirements of an organization.

Simple to say, security is the protection of information that needs to be protected, from unauthorized access. IT security could be helped us through technology, processes, policies and training, so that we can be sure that data stored and secured in a computer or passed between computers is not compromised. Therefore data encryption is the first step in the direction IT-Security. In order to access to specific resources, user needs to provide (normally) his user name and password. Data encryption is the transformation of data into a form that cannot be understood without decryption key(s).

Security Challenges

In a world that we used to work with distributed IT-landscape, we face to with different challenges, e.g. network-based Attacks, heterogeneity on application layer from user interface till to application. It is really difficult to stay on a standard security level for all of team members of development team. We cannot awaiting all of application developers to be able develop solve the security challenges such as privacy, identity management, compliance, audit too. Another area is interfaces between application server and backend database.

A simple case is presented on the following diagram: most applications are multi-tiered and distributed over several systems. A client invokes an application or sends a request to server. This case presents how many systems are in transaction involve. We have to check all of critical points and interfaces: network-based attacks, user interface, application Server and so on.



On these grounds, we need to use an enterprise security framework that allows application developers to pick and choose from a full set of reusable and standards based security services that allow security, privacy, and audit. Oracle Platform Security Services (OPSS) is a security framework that runs on WebLogic Server and is available as part of WebLogic Server. It combines the security features of BEA's internal security (WLS + Oracle Entitlement Server (OES)) and the OAS (Hava Platform Security (JPS) - earlier JAZN) to provide application developers, system integrators, security administrators, and independent SW vendors with a comprehensive security platform framework for Java SE and Java EE applications. In this form, Oracle is able to suggest a uniform enterprise security policy and a self-contained and independent framework with Identity management and audit services across the enterprise. The heart of whole system beats on WebLogic Server.

WebLogic Server provides authentication, authorization, and encryption services with which you can guard these resources. These services cannot provide protection, however, from an intruder who gains access by discovering and exploiting a weakness in your deployment environment. Therefore, whether you deploy WebLogic Server on the Internet or on an intranet, it is a good idea to contact an independent security expert to go over your security plan and procedures, audit your installed systems, and recommend improvements.

OPSS Architecture

As we discussed (http://thecattlecrew.wordpress.com/2014/02/17/it-security-weblogic-server_1/), OPSS is Oracle proposals regarding enterprise security services. It is as a framework that provides a comprehensive set of security services. These services based on Java technologies and have a consistent approach for design and apply security policies to Java EE and resources. We look at OPSS architecture from two different perspectives, which are connected to each other very closely. I try to review the advantages of OPSS for developers and administrators from Application's perspective and present the cooperating of technology components such as LDAP, Application Server and Oracle Fusion Middleware from Component's perspective. Thereby, we can determine the main OPSS's benefits that Oracle says:

- Allows developers to focus on application and domain problems
- Supports enterprise deployments
- Supports several LDAP servers and SSO systems
- Is certified on the Oracle WebLogic Server
- Pre-integrates with Oracle products and technologies

Application's point of view

Oracle Platform Security Services (OPSS) is both a security framework exposing security services and APIs, and a platform offering concrete implementation of security services. It includes these elements:

- Common Security Services (CSS), the internal security framework on which Oracle WebLogic Server is based
- Oracle Platform Services
- User and Role APIs
- Oracle Fusion Middleware Audit Framework

Figure 1 Application's perspective illustrations OPSS's architecture from application point of view. Such architecture allows OPSS to support different security and identity systems without changing the APIs. OPSS is integrated with Oracle Fusion Middleware's management tools to administrate and monitor the security policies implemented in the underlying identity management infrastructure. Therefore, OFM technologies such as Oracle SOA, Oracle WebCenter Suite, Oracle Application Development Framework (ADF), Oracle Web Services Manager (OWSM) and... could use OPSS capacities.

OPSS offers abstraction layer APIs those isolate developers from security and identity management implementation details. In this way, developer can invoke the services provided by OPSS directly from the development environment (e.g. JDeveloper) using wizards. Admin can configure the services of OPSS into the WLS. As you see in Figure, the uppermost layer consists of Oracle WebLogic Server and the components and Java applications running on the server; below this is the API layer consisting of Authentication, Authorization, CSF (Credential Store Framework), and User and Role APIs, followed by the Service Provider Interface (SPI) layer and the service providers for authentication, authorization, and others. The final and bottom layer consists of repositories including LDAP and database servers.

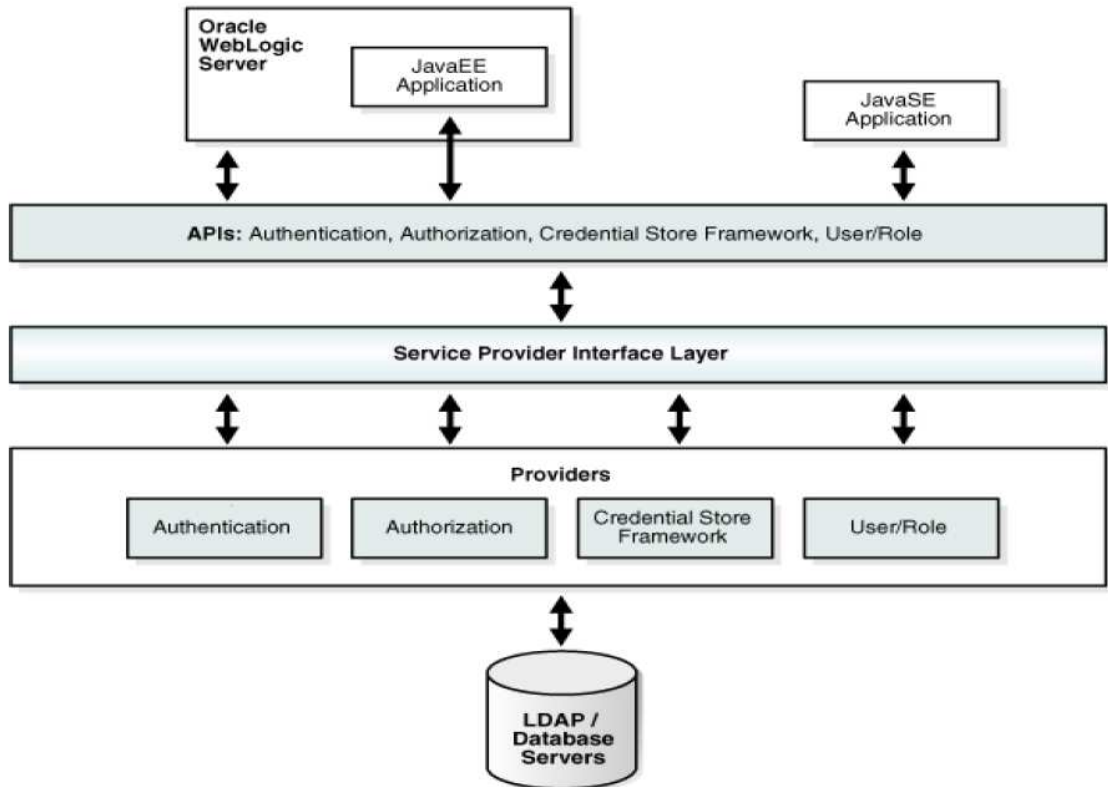


Figure 1 Application's perspective

OFM-Component's point of view

Figure 2 OFM-Component's perspective shows the various security components as layers. The top layer includes the OPSS security services; the next layer includes the service providers, and the bottom layer includes the OPSS security store with a repository of one of three kinds. OPSS provides auditing capabilities for components too.

The second layer [Security Services Provider Interface (SSPI)] has the capability that works with Java EE container security – named Java Authorization Contract for Containers (JACC) mode and in resource-based (non-JACC) mode, and resource-based authorization for the environment.

SSPI is a set of APIs for implementing pluggable security providers. A module implementing any of these interfaces can be plugged into SSPI to provide a particular type of security service. Therefore, OPSS has a consistent structure and is able to meet the requirements for integrating JEE Applications generally and specially OFM-Components and Oracle Security technologies, such as OAM, OID and so on.

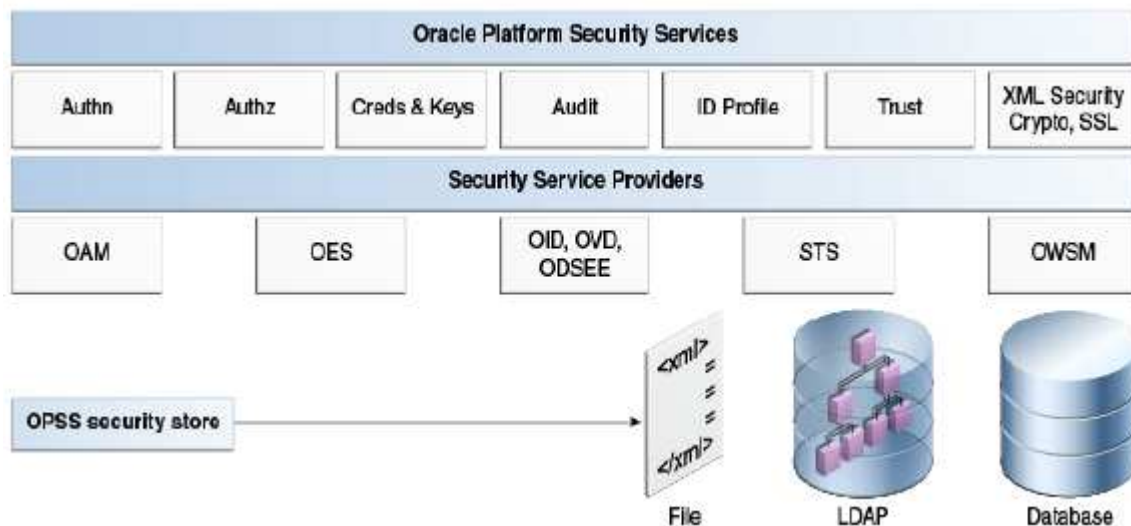


Figure 2 OFM-Component's perspective

IT-Security (Part 3): WebLogic Server and Java Security Features¹

WebLogic Server supports the Java SE and Java EE Security to protect the resources of whole system. The resources could be Web applications, Uniform Resource Locator (URL), Enterprise JavaBeans (EJBs), and Connector components.

Java SE capabilities: Security APIs

Java uses APIs to access security features and functionality and its architecture contains a large set of application programming interfaces (APIs), tools, and implementations of commonly-used security algorithms, and protocols. This delivers the developer a complete security framework for writing applications and enables them to extend the platform with new security mechanisms.²

Java Authentication and Authorization Services (JAAS)

WebLogic Server uses the Java Authentication and Authorization Service (JAAS) classes to consistently and securely authenticate to the client. JAAS is a part of Java SE Security APIs and a set of Java packages that enable services to authenticate and enforce access controls upon users and /or fat-client authentication for applications, applets, Enterprise JavaBeans (EJB), or servlets.

JAAS uses a Pluggable Authentication Module (PAM) framework, and permits the use of new or updated authentication technologies without requiring modifications to the application. Therefore, only developers of **custom** Authentication providers and developers of remote fat client applications need to be involved with JAAS directly. Users of thin clients or developers of within-container fat client applications do not require the direct use or knowledge of JAAS.

JAAS LoginModules

All LoginModules are responsible for authenticating users within the security realm (we are going to discuss about that later) and for populating a subject with the necessary principals (users/groups). LoginModules contains necessary methods for Login Context, Accounts, Credentials, configuration of them, and different ways to exception handling. Each Authentication providers will be configured in a security realm, its LoginModules will store principals within the same subject too. I try to present that

with an example: Via WebLogic Server Admin Console: Home >myDomain > Domain Structure click on Security Realms and then create a new realm “Moh_Realm-0” and then click on “OK”

Create a New Realm

OK | Cancel

Realm Properties

The following properties will be used to identify your new Realm.

* Indicates required fields

What would you like to name your new Realm?

* **Name:**

To avoid overwriting new credential mapping information with old information

Ignore Deploy Credential Mapping

OK | Cancel

Figure 1 create a new Realm

Select new realm and then click on tab “provider”, and then click on “New”, in order to create a new provider:

Realms (Filtered - More Columns Exist)

New Delete Showing 1 to 2 of 2 Previous | Next

<input type="checkbox"/> Name	Default Realm
<input type="checkbox"/> Moh_Realm-0	false
<input type="checkbox"/> myrealm	true

Figure 2 open the new Realm

In this use case, we select type: “WebLogic Authentication Provider” and give a name e.g. “DefAuthN”, then “OK”. The WebLogic Authentication provider is configured in the default security realm (myrealm). The WebLogic Authentication provider allows you to edit, list, and manage users, groups, and group membership. User and group information is stored in the embedded LDAP server.³

Create a New Authentication Provider

|

Create a new Authentication Provider

The following properties will be used to identify your new Authentication Provider.

* Indicates required fields

The name of the authentication provider.

*** Name:**

This is the type of authentication provider you wish to create.

Type:

|

Figure 3 create a new Authentication Provider

After define "Provider", we have to restart Admin Server. Now, we can check and compare users of new realm (Moh_Realm-0) with default realm (myrealm) of WebLogic. For myrealm, I created a new user named "userDOAG" and we see the following list there (Home >Summary of Security Realms >myrealm >Users and Groups)

	Name
<input type="checkbox"/>	OracleSystemUser
<input type="checkbox"/>	userDOAG
<input type="checkbox"/>	weblogic

Figure 4 users of myrealm

But I didn't create same user for Moh_Realm-0 (Home >DefAuthN>Summary of Security Realms >Moh_Realm-0 >Users and Groups):



Figure 5 users of Moh_Realm-0

It shows, that we can use security provider in different gatherings und expand our security realm with additional user, groups, and security providers. We are working on it in next part of this article.

JAAS Control Flags

The JAAS Control Flag attribute determines how the LoginModule for the WebLogic Authentication provider is used in the login sequence. The values for the Control Flag attribute are as follows: Home >Summary of Security Realms > Moh_Realm-0 >Providers > **DefAuthN**

Settings for DefAuthN

Configuration

Performance

Migration

Common

Provider Specific

This page displays basic information about this WebLogic Authentication provi

Name:

DefAuthN

Description:

WebLogic Authentication Provider

Version:

1.0

Control Flag:

OPTIONAL
▾

REQUIRED

REQUISITE

SUFFICIENT

OPTIONAL

Figure 6 Control flags via Admin Console

- **REQUIRED** - This LoginModule must succeed. Even if it fails, authentication proceeds down the list of LoginModules for the configured Authentication providers. This setting is the default.
- **REQUISITE** - This LoginModule must succeed. If other Authentication providers are configured and this LoginModule succeeds, authentication proceeds down the list of LoginModules. Otherwise, return control to the application.
- **SUFFICIENT** - This LoginModule needs not succeed. If it does succeed, return control to the application. If it fails and other Authentication providers are configured, authentication proceeds down the LoginModule list
- **OPTIONAL** - The user is allowed to pass or fail the authentication test of these Authentication providers. However, if all Authentication providers configured in a security realm have the JAAS Control Flag set to OPTIONAL, the user must pass the authentication test of one of the configured providers.⁴

Now, we can focus on two important JAAS-tasks: authentication and authorization of users...⁵

IT-Security: WebLogic Server and Authentication – Part 4

As I mentioned, JAAS is able for two important tasks: authentication and authorization of users. Now, let us see more about them.

Authentication: Who are you?

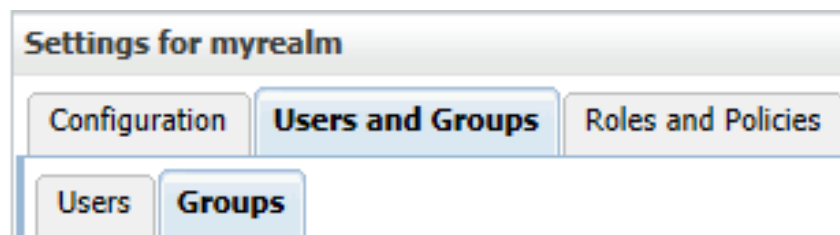
Authentication verifies that the user is who she/he claims to be. But user is also an entity and could be a person, a software entity or other instances of WebLogic Server (so called "resources"). WLS performs proof material typically through a JAAS LoginModule and JAAS authentication is implemented in a pluggable method. A user's identity is confirmed through the credentials presented by that user, such as:

1. something one has, e.g. credentials issued by a trusted authority such as a passport or a smart card
2. something one knows, e.g. a shared secret such as a password,
3. something one is, e.g. biometric information

A combination of several types of credentials is known as "strong" authentication; e.g. using an ATM card (credential 1) with a PIN or password (credential 2).⁶

Types of Authentication


WebLogic Server is able to perform the different types of authentication, because it can use the WebLogic Authentication provider or custom security providers. Administrators are able to define a user and password with WebLogic Authentication provider. The all passwords will be encrypted. Users may be placed into groups or be related with security roles.



Basic Authentication: Username/Password

Basic authentication is defined via The Internet Engineering Task Force (IETF®) so: "The "basic" authentication scheme is based on the model that the client must authenticate itself with a user-ID and a password for each realm. The realm value should be considered an opaque string which can only be compared for equality with other realms on that server. The server will service the request only if it can validate the user-ID and password for the protection space of the Request-URI. There are no optional authentication parameters."⁷

In this type of authentication will be user/password requested. WebLogic scenario looks like this: the user and sent ID/PW to WebLogic Server. It checks them and if it is reliable, gives access to the protected WebLogic resource. In background, WebLogic Server checks the security policy of the WebLogic resource and the principal (that the user has been assigned) to make sure that the user has the obligatory permissions to continue.



In addition, you can use https. User/password will be encrypted between client and server through SSL communication. It is an extra advantage that the transaction between client and server will not be performed in clear text.

Certificate Authentication

We are going to discuss about Secure Sockets Layer (SSL) in the next articles. SSL delivers protected connections. The SSL-communicating authenticate identity of two entity and/or application that communicate through a network connection. In addition, the whole SSL-communication is encrypted. WebLogic Server provides a pure-Java implementation of SSL and supports One-Way- and Two-Way-SSL authentication.

Simple to say, if a WLS to authenticate to a client, then we have a One-Way SSL. If a client to authenticate to a WLS, then we have Two-Way SSL. One-Way SSL is obligatory but Two-Way SSL is optional. During “handshaking” exchange the applications and/ or entities digital certificates. The digital certificate is supplied by an entity, which authenticates the identity of WebLogic Server.

Afterwards, the both sides, also WebLogic Server and client, decide on the encryption algorithms to be used. As third step, SSL-connection generates the encryption keys to be used for the remainder of the session. The encryption keys is a hybrid encryption approach that it uses advantages of asymmetric and symmetric encryption therefore, it is known as a good combination between better performance and security in network communication.

Digest Authentication

We are going back to this topic for deeper discussion. As an introduction, we can start with the definition of The Internet Engineering Task Force (IETF®): “Like Basic Access Authentication, the Digest scheme is based on a simple challenge-response paradigm. The Digest scheme challenges use a nonce value. A valid response contains a checksum (by default, the MD5 checksum) of the username, the password, the given nonce value, the HTTP method, and the requested URI. In this way, the password is never sent in the clear. Just as with the Basic scheme, the username and password must be prearranged in some fashion not addressed by this document.”⁸

Weblogic Server supports digest authentication and is resistant to replay attacks. “The implementation maintains a cache of used nonces/timestamps for a specified period of time. All requests with a timestamp older than the specified timestamp are rejected as well as any requests that use the same timestamp/nonce pair as the most recent timestamp/nonce pair still in the cache. WebLogic Server stores this cache in a database.”⁹

I’m going to continue with Authentication’s topic in next part of IT-Security and WebLogic Server.

IT-Security (Part 5): WebLogic Server, perimeter Authentication and Identity Assertion

I tried to discuss about “perimeter authentication” in one extra part of IT-Security’s blogs, because this authentication’s process is an essential approach in a heterogenous world of systems, applications and technologies that they need to trust and communicate to each other. Generally, we discussed about perimeter authentication, if a remote user requires an asserted identity and some form of proof material to an authentication server that performs the verification and then passes an artifact, or token, to the application server domain.¹⁰

If we want to identify a *remote user outside* of the WebLogic server domain, as an authentication server, then we need to another approach for authenticating’s process instead basic authentication with username and password¹¹. This authentication’s process is called perimeter authentication. It establishes trust via a passphrase, e.g. tokens. Tokens will be generated as part of the authentication process of users or system processes and could have many different types and / or vendors, e.g. Kerberos and Security Assertion Markup Language (SAML). WebLogic Server is able to use the token(s) so that users are not requested to sign on more than once.

This form of authentication operates with authentication agent. It performs an authentication process that outcomes in a token. It contains the authentication information of user and guarantees for the user’s identity. The Figure 7 Perimeter Authentication¹² presents the sequence of events in authenticating process:

Remote User sends a request with passphrase to Authentication Agent. It creates a token and sends to WebLogic Server to access resources and / or application(s). The WebLogic Server perform perimeter authentication via Identity Assertion.

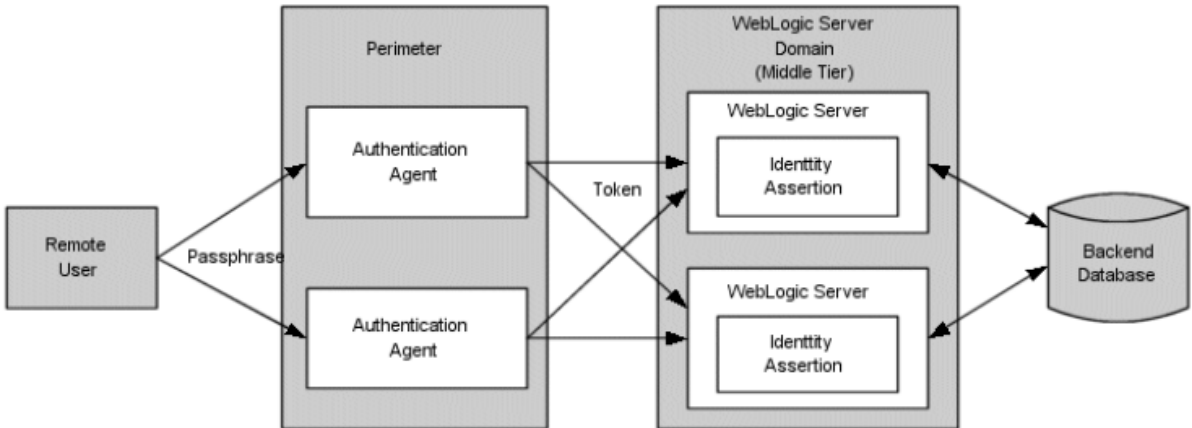


Figure 7 Perimeter Authentication

We can define the Identity Assertion provider, as a *specific form of Authentication provider* that permits users or applications to assert their identity using tokens. With other words, it supports user’s mappers, which map a valid token to a WLS-User. It is possible to develop your own or use a third-party security vendor's Identity Assertion providers. Identity assertion can use perimeter authentication schemes such as the Security Assertion Markup Language (SAML), the Simple and

Protected GSS-API Negotiation Mechanism (SPNEGO), or enhancements to protocols such as Common Secure Interoperability (CSI) v2 and support single sign-on.¹³ The WebLogic Identity Assertion providers support the following token types¹⁴ (here is [a selected list](#) of token types):

- `AU_TYPE`, for a WebLogic `AuthenticatedUser` used as a token.

X509 is an ITU-T standard for a public key infrastructure (PKI) and Privilege Management Infrastructure (PMI) and RFC 4158 provides information and guidance for certification path building.¹⁵

- `X509_TYPE`, for an X509 client certificate used as a token:
- `CSI_X509_CERTCHAIN_TYPE`, for a CSIv2 X509 certificate chain identity used as a token.

“The Negotiate Identity Assertion provider is used for SSO with Microsoft clients that support the SPNEGO protocol. The Negotiate Identity Assertion provider decodes SPNEGO tokens to obtain Kerberos tokens, validates the Kerberos tokens, and maps Kerberos tokens to WebLogic users. The Negotiate Identity Assertion provider utilizes the Java Generic Security Service (GSS) Application Programming Interface (API) to accept the GSS security context via Kerberos. The Negotiate Identity Assertion provider is for Windows NT Integrated Login.”¹⁶

- `AUTHORIZATION_NEGOTIATE`, for a SPNEGO internal token used as a token.
- `WWW_AUTHENTICATE_NEGOTIATE`, for a SPNEGO internal token used as a token.

“The SAML Identity Assertion providers handle SAML assertion tokens when WebLogic Server acts as a SAML destination site. The SAML Identity Assertion providers consume and validate SAML assertion tokens and determines if the assertion is to be trusted (using either the proof material available in the SOAP message, the client certificate, or some other configuration indicator).”¹⁷ I am going back to SAML topic in an additional article(s).

- `SAML_ASSERTION_B64_TYPE`, for a Base64 encoded SAML assertion used as a token.
- `SAML_ASSERTION_DOM_TYPE`, for a SAML DOM element used as a token.
- `SAML_ASSERTION_TYPE`, for a SAML string XML form used as a token.
- `SAML2_ASSERTION_DOM_TYPE`, for a SAML2 DOM element used as a token.
- `SAML2_ASSERTION_TYPE`, for a SAML2 string XML form used as a token.
- `SAML_SSO_CREDENTIAL_TYPE`, for a SAML string consisting of the `TARGET` parameter concatenated with the assertion itself and used as a token.

I introduced about Digest Authentication¹⁸ in previous blog and WebLogic supports für Web Service application the following Digest type:

- `WSSE_PASSWORD_DIGEST_TYPE`, for a username token with a password type of password digest used as a token.

The Authentication and Identity Assertion Process

Now, we can compare Basic authentication Process with Identity Assertion Process. On Figure 8 Authentication Process (Principal Validation Process)¹⁹ shows the authentication process for a fat-client login. A user attempts to log into a system using a username/password combination. WebLogic Server establishes trust by calling the configured Authentication provider's `LoginModule`, which validates the user's username and password and returns a subject that is populated with principals per Java Authentication and Authorization Service (JAAS)²⁰ requirements. In this way, an

authentication context will be established and user can access to certain resource and / or components in WebLogic Domain.

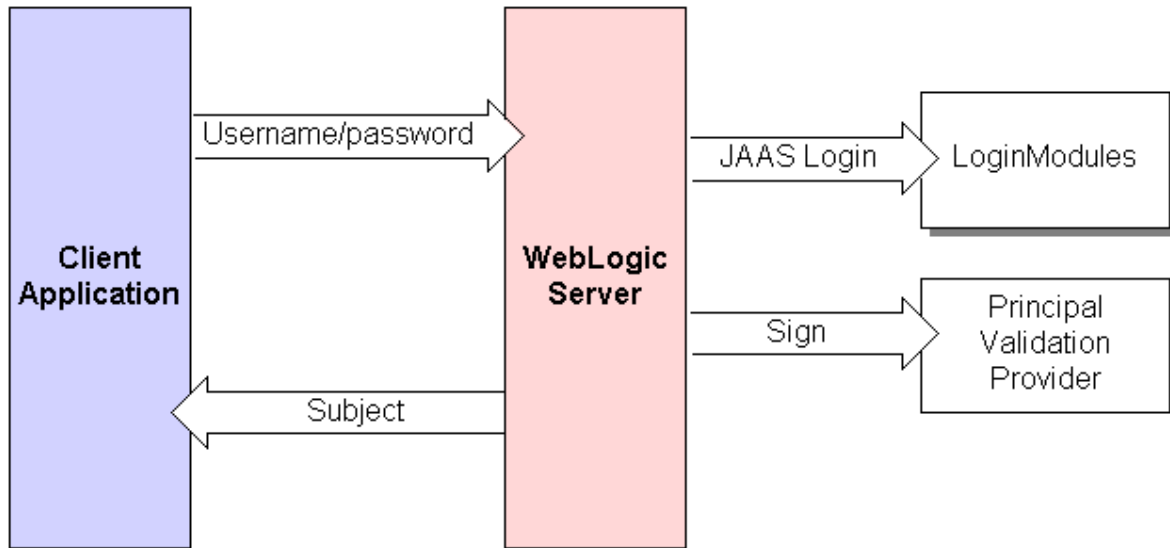


Figure 8 Authentication Process (Principal Validation Process)

Figure 9 Perimeter Authentication presents the perimeter authentication process²¹.

1. A token from *outside* of WebLogic Server is passed to an Identity Assertion provider that is responsible for validating tokens of that type and that is configured as "active".
2. If the token is successfully validated, the Identity Assertion provider maps the token to a WebLogic Server username, and sends that username back to WebLogic Server, which then continues the authentication process as described above. It requires the same components, but also adds an Identity Assertion provider. Specifically, the username is sent via a Java Authentication and Authorization Service (JAAS) `CallbackHandler` and passed to each configured Authentication provider's `LoginModule`, so that the `LoginModule` can populate the subject with the appropriate principals.

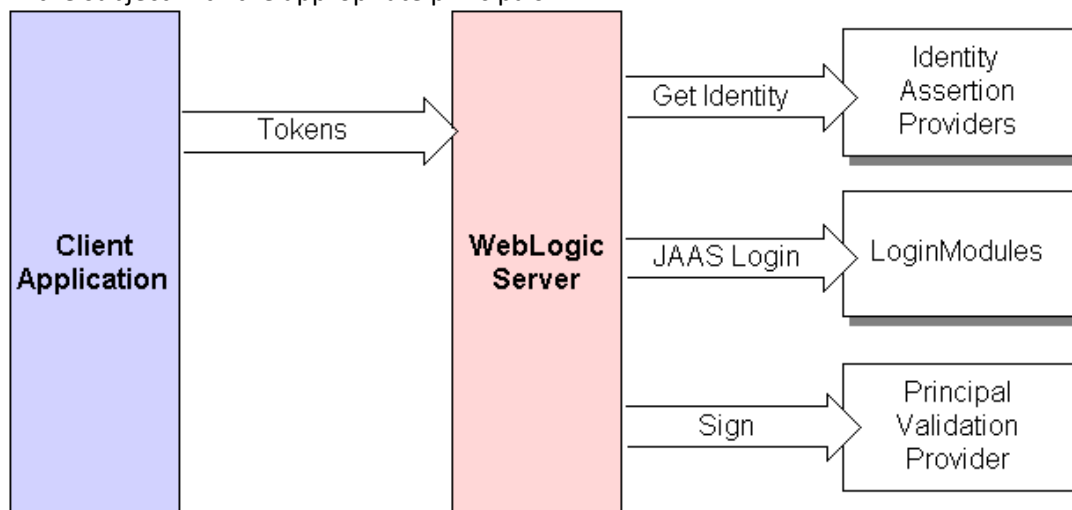


Figure 9 Perimeter Authentication

If you compare the two ways of authentication, then you can find out a core security characteristic of WebLogic Server too. It is mean; WebLogic Server security architecture has a consistence modular

structure and therefore can respond rapidly to new challenges and technologies in security area. This architecture is capable to expand its features and integrate new security components in itself.

References

- Oracle Fusion Middleware 11.1.1.5, Security Guides
http://docs.oracle.com/cd/E21764_01/security.htm
- Oracle® Fusion Middleware Understanding Security for Oracle WebLogic Server 11g Release 1 (10.3.5) http://docs.oracle.com/cd/E21764_01/web.1111/e13710/toc.htm
- Oracle® Fusion Middleware Securing Oracle WebLogic Server
http://docs.oracle.com/cd/E21764_01/web.1111/e13707/toc.htm
- Oracle Platform Security Services 11gR1 (White Paper)
<http://www.oracle.com/technetwork/middleware/id-mgmt/opss-tech-wp-131775.pdf>
- IT Security Guidelines, IT Baseline Protection in brief, Federal Office for Information Security
- See too: Do you forget your WebLogic Server password? No problem!,
<http://thecattlecrew.wordpress.com/2014/01/27/did-you-forget-your-weblogic-server-password-no-problem/>
- Oracle® Fusion Middleware Understanding Security for Oracle WebLogic Server 11g Release 1 (10.3.5) http://docs.oracle.com/cd/E21764_01/web.1111/e13710/toc.htm
- <http://www.oracle.com/technology/tech/standards/idm/igf/index.html>
- Oracle® Fusion Middleware: Understanding Security for Oracle WebLogic Server 12c Release 1, (12.1.1), E24484-02, January 2012:
http://docs.oracle.com/cd/E24329_01/web.1211/e24484.pdf
- For details, see OPSS Architecture Overview in the Oracle Fusion Middleware Application Security Guide: http://docs.oracle.com/cd/E23943_01/core.1111/e10043.pdf
- Oracle® Access Manager Integration Guide:
http://docs.oracle.com/cd/E12530_01/oam.1014/e10356/weblogic.htm

¹ IT-Security (Part 1): http://thecattlecrew.wordpress.com/2014/02/17/it-security-weblogic-server_1/
IT-Security (Part 2): <http://thecattlecrew.wordpress.com/2014/03/05/it-security-part-2-weblogic-server-and-oracle-platform-security-services-opss-2/>

² See: <http://docs.oracle.com/javase/7/docs/technotes/guides/security/overview/jsoverview.html>

³ More detail in: http://docs.oracle.com/cd/E24329_01/apirefs.1211/e24403/core/index.html

⁴ Oracle Fusion Middleware: Understanding Security for Oracle WebLogic Server 12c Release 1, (12.1.1), E24484-02, January 2012: http://docs.oracle.com/cd/E24329_01/web.1211/e24484.pdf

⁵ See too: Do you forget your WebLogic Server password? No problem!,
<http://thecattlecrew.wordpress.com/2014/01/27/did-you-forget-your-weblogic-server-password-no-problem/>

-
- ⁶ See Oracle Fusion Middleware Security Overview
http://docs.oracle.com/cd/E23943_01/core.1111/e12889.pdf
Oracle Fusion Middleware 11.1.1.5, Security Guides http://docs.oracle.com/cd/E21764_01/security.htm
Oracle® Fusion Middleware Securing Oracle WebLogic Server
http://docs.oracle.com/cd/E21764_01/web.1111/e13707/toc.htm
Oracle Platform Security Services 11gR1 (White Paper)
<http://www.oracle.com/technetwork/middleware/id-mgmt/opss-tech-wp-131775.pdf>
- ⁷ Request for Comments: 2617: The Internet Engineering Task Force (IETF®):
<https://datatracker.ietf.org/doc/rfc2617/>
- ⁸ Request for Comments: 2617: The Internet Engineering Task Force (IETF®):
<https://datatracker.ietf.org/doc/rfc2617/>
- ⁹ Oracle® Fusion Middleware Understanding Security for Oracle WebLogic Server 11g Release 1 (10.3.5)
http://docs.oracle.com/cd/E21764_01/web.1111/e13710/toc.htm
- ¹⁰ Oracle® Fusion Middleware: Understanding Security for Oracle WebLogic Server, 11g Release 1 (10.3.6), E13710-06
- ¹¹ For „Basic Authentication: Username/Password“ see: <http://thecattlecrew.wordpress.com/2014/06/05/it-security-weblogic-server-and-authentication-part-4/>
- ¹² Oracle® Fusion Middleware: Understanding Security for Oracle WebLogic Server, 11g Release 1 (10.3.6), E13710-06
- ¹³ Oracle® Fusion Middleware Developing Security Providers for Oracle WebLogic Server, 11g Release 1 (10.3.6), Part Number E13718-05, http://docs.oracle.com/cd/E23943_01/web.1111/e13718/ia.htm
- ¹⁴ Oracle® Fusion Middleware Developing Security Providers for Oracle WebLogic Server, 11g Release 1 (10.3.6), Part Number E13718-05, http://docs.oracle.com/cd/E23943_01/web.1111/e13718/ia.htm
- ¹⁵ See: <http://tools.ietf.org/html/rfc4158>
- ¹⁶ Oracle® Fusion Middleware Developing Security Providers for Oracle WebLogic Server, 11g Release 1 (10.3.6), Part Number E13718-05, http://docs.oracle.com/cd/E23943_01/web.1111/e13718/ia.htm
- ¹⁷ Oracle® Fusion Middleware Developing Security Providers for Oracle WebLogic Server, 11g Release 1 (10.3.6), Part Number E13718-05, http://docs.oracle.com/cd/E23943_01/web.1111/e13718/ia.htm
- ¹⁸ See <http://thecattlecrew.wordpress.com/2014/06/05/it-security-weblogic-server-and-authentication-part-4/>
- ¹⁹ See: http://docs.oracle.com/cd/E23943_01/web.1111/e13718/atn.htm#i1141106
- ²⁰ IT-Security (Part 3): WebLogic Server and Java Security Features:
<http://thecattlecrew.wordpress.com/2014/03/14/it-security-part-3-weblogic-server-and-java-security-features/>
- ²¹ See http://docs.oracle.com/cd/E23943_01/web.1111/e13718/ia.htm