An Oracle Technical White Paper
July 2010

Oracle Platform Security Services 11gR1

ORACLE

Disclaimer

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Introduction

With the proliferation of network-centric, web-based applications, application development has become more complex and increasingly more costly.

Most web applications are multi-tiered and distributed over several systems. For example, a client invokes an application from a browser, a router or a web server redirects the client's request to an application server that processes the request, and a database system stores the information related to that application.

Web-based applications have ushered in new types of security vulnerabilities inherent in the nature of the web (loosely-coupled connections over (mainly) HyperText Transport Protocol). Most network-based attacks are well under control thanks to the use of time-tested technologies such as firewalls and other intrusion detection mechanisms. However, application security has become more crucial because applications are more vulnerable to security breaches than the network they rely on.

Application Security Challenges

Until recently, security was an integral part of the application; business logic and security code were hardly discernible. Today's web-based applications demand much more agility to face changing customer needs, often leading to many application component modifications and redeployments.

Typically, multi-tiered web applications are defined in "patterns" that isolate user interface from business logic and data storage. Application security is different at each tier involved in the overall process. For example, a user interface must provide a way to authenticate incoming requests, and application servers must access backend database systems securely.

Organizations understand the necessity of including security as part of the development process, but they face challenges in implementing security in the various layers of multi-tiered web applications.

Application developers must make sure that the application's business logic and user interface meet the business requirements. Most developers are not familiar with security-aware technologies such as privacy, identity management, compliance, and audit. For example, developers need to understand the details of container-based security to secure Enterprise JavaBeans (EJBs) and web applications deployed as web archives (WAR). Developers also need to be familiar with the single sign-on solution used by the enterprise to "wire" the authentication

of EJBs and web applications correctly, and they need to understand how authorization is done to make sure these components are accessed securely. Authorization may vary depending on the resource that needs to be secured. For example, web applications may rely on an access management product such as Oracle Access Manager for single sign-on and coarse grained access, or they may rely on a controller model that dispatches resource requests to the appropriate web pages. Likewise, EJBs may be secured through their container (provided by the application server), or EJB security may be done programmatically, which requires a thorough understanding of the Java security model.

Audit presents another set of challenges. Audit is typically handled through homegrown frameworks for logging and auditing, which produces a voluminous amount of raw information that needs to be processed and consolidated by yet another tool.

Application Security Requirements

Application security challenges raised with siloed and embedded security are addressed by moving security out of applications. Externalized security can be centrally managed using a consistent set of tools and can be modified without costly application changes or redeployments. Externalized security and a consistent methodology and interfaces for enterprise IDM integration removes the need for costly point solutions and frees business developers to focus on solving business problems instead of inventing their own security modules and IDM glue code.

To build secure applications in a cost effective and efficient way one needs a framework that provides a comprehensive set of security services and, if required, easily allows the application to integrate with various enterprise IDM systems.

What about Java, does it not provide sufficient toolkits to meet these needs? Unlike most development frameworks, Java was designed from the ground up to support network-centric environments with a focus on security.

Java Platform, Standard Edition (Java SE) provides various security application programming interfaces (API) and libraries such as Java Cryptography Architecture (JCA), Java Cryptography Extensions (JCE), Java Authentication and Authorization Services (JAAS), and Java Secure Socket Extension (JSSE) that support basic security features.

Java Platform, Enterprise Edition (Java EE) provides additional security features such as container-managed authentication and coarse-grained authorization (both container-managed and programmatic) but lacks many other security features such as audit, single sign-on, fine-grained authorization, which are required by today's enterprise-grade applications.

The security APIs and libraries provided by Java SE and Java EE are often too low level for business application developers, lack management tooling, and do not cover the full spectrum of security requirements an application needs. Finally, Java EE security does not address the life

cycle of an application. For example, Java EE does not specify how an application's security data is moved from test to production.

What is needed is a framework that allows application developers to pick and choose from a full set of reusable and standards based security services that allow security, privacy, audit, and identity management integration to be externalized from applications, is designed for and can run in heterogeneous environments across different platforms with IDM components from different vendors, and scales and performs to meet the demands of the largest enterprise applications. The framework needs to support the full application lifecycle, allows developers to spend their time where it is most needed - on the business logic of the application, and provides administrators with consistent tooling for lifecycle management, security configuration and runtime metrics.

Introducing Oracle Platform Security Services

Oracle Platform Security Services (OPSS) provides enterprise product development teams, systems integrators, and independent software vendors with a standards-based, portable, integrated, enterprise-grade security framework for Java SE and Java EE applications.

OPSS builds upon Java SE and Java EE security and provides an abstraction layer in the form of standards-based APIs that insulate developers from security and identity management implementation details. With OPSS developers don't need to know the nitty-gritty of cryptographic key management or interfaces with user repositories and other identity management infrastructures.

Thanks to OPSS, in-house developed applications, third-party applications, and integrated applications benefit from the same, uniform security, identity management, and audit services across the enterprise.

OPSS is a self-contained, independent framework that provides security to Oracle Fusion Middleware (OFM). OPSS is designed to be portable to third-party application servers. As a result, developers can use OPSS as the single security framework for both Oracle and third-party environments, thus decreasing application development, administration, and maintenance costs.

At development time, OPSS is integrated with JDeveloper's security wizard and authorization editor that automatically creates the required OPSS configuration. When the application is deployed to the runtime environment, systems and security administrators can access OPSS services to configure, manage, monitor, and audit applications using Oracle Enterprise Manager (EM).

OPSS is the common security platform used by OFM components, from the underlying application server (Oracle WebLogic Server) to OFM components such as Oracle SOA, Oracle WebCenter and Oracle Application Development Framework (ADF), as well as Oracle Fusion applications.

OPSS offers standards based APIs and provider based architecture, such that applications are not doing point integrations with IDM products directly but are using OPSS instead. OPSS handles IDM integration automatically through configuration wizards and files that administrators leverage to integrate OPSS enabled applications with pre-certified Oracle or 3rd party IDM products without the need for code changes. This frees application developers from writing and testing glue code and allows them to focus on business logic instead.

With Oracle Fusion Middleware 11gR1, OPSS provides the same, unified security platform for Oracle products, commercial off-the-shelf (COTS) products, and in-house-developed customer applications.

Oracle Platform Security Services Evolution

Oracle WebLogic Server is the *de facto* application server for Oracle Fusion Middleware 11g. With the release of Oracle Fusion Middleware 11gR1, Oracle combined BEA' s internal security framework used in products like WebLogic Server and Oracle Entitlements Server (OES), with Oracle Fusion Middleware's security platform, Java Platform Security (JPS), into one complete security framework known as Oracle Platform Security Services (OPSS).

JPS, earlier known as JAZN, was first released with Oracle Application Server 9.0.4 as an authentication and authorization service that has evolved into a full-fledged security platform over the last few years.

Oracle Platform Security Services Architecture

OPSS provides a layered architecture. Java EE and Java SE applications use APIs to access security features and functionality. These APIs abstract the details of underlying identity management and security systems, allowing application developers to focus on business logic and interfaces. Security is removed from an application and externalized into security systems where administrators can manage it.

OPSS is integrated with Oracle Fusion Middleware's management tools to administer and monitor the security policies implemented in the underlying identity management infrastructure.

Figure 1 shows OPSS's application-centric architecture. OPSS's layered architecture allows it to support different security and identity systems without changing the APIs provided to applications.

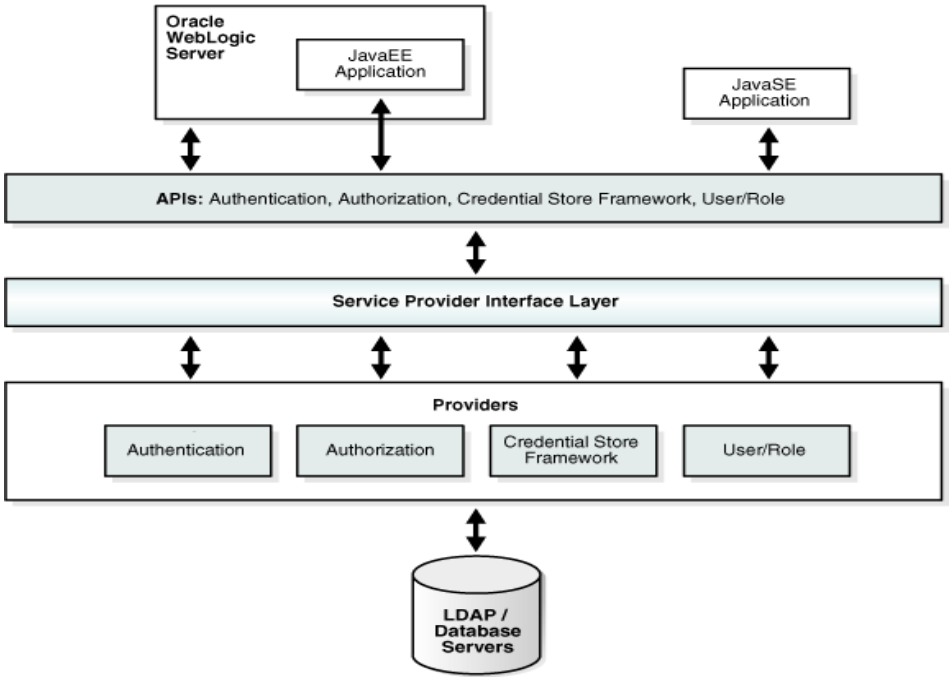

Figure 1: OPSS Architecture - Applications view

Application Life Cycle Support

OPSS provides support for all the phases of an application's life cycle. OPSS is integrated with Oracle JDeveloper, which allows an application designer to model security into the application when building Oracle ADF task flows. Oracle JDeveloper provides a security wizard that creates

the required configuration. JDeveloper also provides an authorization editor that allows developers to create authorization policies for ADF taskflows and pages without writing a single line of code. For more complex security requirements, application developers can use JDeveloper's expression language editor or use programmatic APIs that allow further customization of security policies.

During application development a developer typically deploys the application to a WebLogic Server domain embedded in JDeveloper. The developer can then deploy the application to a remote WebLogic Server domain using Oracle Enterprise Manager Fusion Middleware Control (FMWControl). OPSS is integrated with FMWControl to allow application security policies and credentials migration to be configured during application deployment.

Figure 2 shows an application packaged as an enterprise archive (EAR) auto deployed to an embedded WebLogic Server domain during application development, followed by the deployment of the application to a staging WebLogic Server domain.

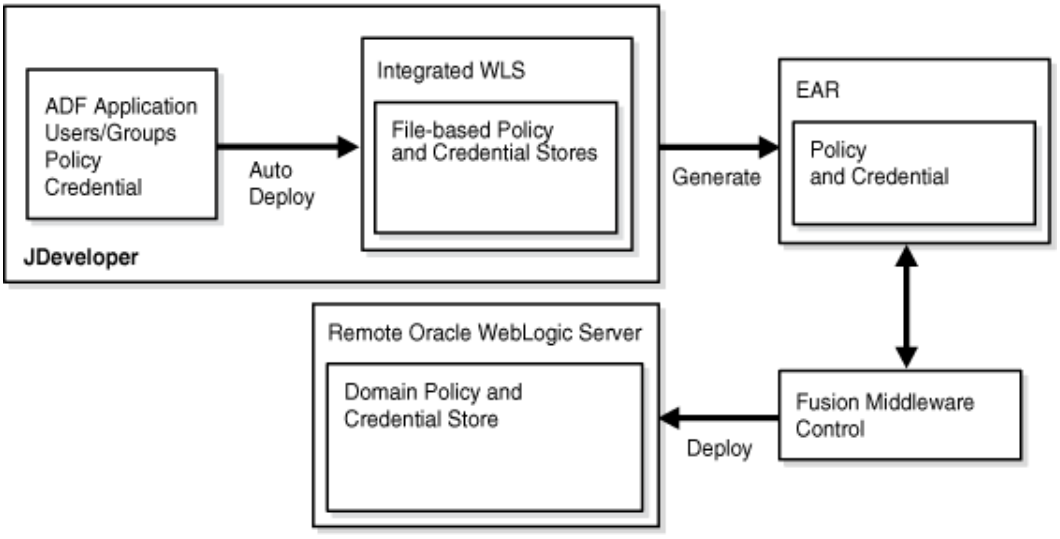

Figure 2 Application Life Cycle Support

Post deployment, an administrator uses FMWControl to manage the application's security policies, e.g., edit authorization policies, or change audit policies. All such changes are transparent to applications and do not require any application code change.

In any non-trivial application scenario, an application normally goes from development to a staging (or test) environment before being put in full-blown production. OPSS supports this model by providing migration tools that move security policies from a test domain into a

production domain. For example, audit policies configured in a test domain can be exported into the target production domain.

Oracle Platform Security Services Functionality

The following section describes the various services provided by OPSS.

Authentication

OPSS integrates with WLS Security for container-managed authentication and leverages WLS Authenticators. Container managed authentication is a good choice for many applications. OPSS also provides programmatic API for identity and token assertion and user authentication to Java EE applications. On Java SE platform OPSS provides login modules to provide authentication.

Single Sign-On

OPSS provides a single sign-on (SSO) framework that insulates applications from the details on the SSO system configured in a deployment. The SSO framework integrates with SSO systems such as Oracle Access Manager (OAM) and allows applications to dynamically decide when authentication is required. OPSS also provides building block for OAM-WLS integration via Security Service Provider Interface (SSPI).

Authorization

OPSS provides a Java policy provider that supports code-based and subject-based authorization. OPSS allows authorization policies to be partitioned based on the application. OPSS supports logical roles specific to an application (Application Roles). Unlike Java EE's logical role, the OPSS applications role support role inheritance hierarchies. OPSS also provides an advanced policy model that includes elements such as Resource Type & Entitlement allowing complex authorization policies to be conveniently defined and managed. A Resource Type is a type of secured artifact created by the application developer, e.g., a Web URLs, Pages, Methods, Web Services, ADF task flow or a custom application resource. An Entitlement represents a reusable set of named actions on a collection of resource instances required to perform a business function. Using FMWControl or WebLogic Scripting Tool (WLST), the administrator can manage an application's authorization policies, including mapping the application roles to enterprise groups and users, or editing the permissions granted to an application role. OPSS also provides a policy management API allowing programmatic control over authorization policies.

Audit

OPSS provides an internal audit framework used across Oracle Fusion Middleware. Customers using OPSS's API don't need to write a single line of audit-related code. Audit policies are maintained outside of an application, allowing administrators to change audit policies without affecting the application. The audit data generated across Oracle Fusion Middleware is centrally

stored in an audit store. Audit policies can be configured using either FMWControl or WLST commands. OPSS also provide pre-built reports that can be viewed using Oracle Business Intelligence Publisher.

Credential Store Framework

Applications often need to store credentials (e.g., username and password). For example many applications need to store the credentials necessary to access a database, a user directory, a web service, or a web site. Many applications store credentials in plain text files, which often do not meet secure coding standards. OPSS provides Credential Store Framework (CSF) allowing applications to access credentials stored securely outside of an application. FMWControl and WLST commands are used to manage these stored credentials without affecting the application or requiring any restarts. Applications using CSF allow administrator to manage credentials without knowing any application details or without making any application code changes.

Identity Governance Framework (IGF)

Often applications need to access user attributes. For example, many applications have a "My account" page that shows the user's address, email, and phone number. These attributes may be stored across different types of systems (LDAP or RDBMS). OPSS provides an abstraction called the IGF API allowing applications to query user attributes regardless of the underlying identity store; the IGF API provides additional features such as searching for users, groups, and attribute mapping.

Cryptography

Cryptography is provided by a set of Java-based libraries known as Oracle Security Developer Toolkit (OSDT). Typically, OSDT (based on Sun's JCA/JCE provider) allows developers to include security in enterprise applications using standards-based APIs for securing message envelopes, Secure/Multipurpose Internet Mail Extensions (S/MIME), public key infrastructure (PKI), XML security (encryption and signature), Security Assertion Markup Language (SAML), XML Key Management (XKML), WS-Security, and the Liberty Alliance standards.

Management

OPSS is integrated with Oracle Fusion Middleware management tools. OPSS services and providers are managed with FMWControl and WLST. In addition, OPSS also provides Java Management Extensions (JMX) MBeans allowing programmatic management of OPSS. The management model allows security to be abstracted out of the application using the OPSS APIs giving the administrator the ability to control and manage security without making any application code changes.

Security Providers

OPSS provides a rich set of providers that support various identity stores for authentication. In 11gR1, out of the box, OPSS supports authentication and ID profile lookups against Oracle Internet Directory, Oracle Virtual Directory, Microsoft Active Directory, Oracle Directory Server Enterprise Edition, Novell eDirectory, IBM Tivoli Directory, and Open LDAP. If none of the out-of-box providers meets the customer's need, OPSS allows customers to write and plug in their own provider to meet their unique requirements.

Security Stores

OPSS provides the abstraction of Identity, Policy and Credential stores to hide the repository details. The Identity Store is the repository of enterprise users and groups. The Policy Store is the repository of application and system policies. The Credential Store is the repository of domain credentials. OPSS uses one logical store to keep both policies and credentials. This abstraction allows OPSS to support different security stores (XML, LDAP & DB) to be used without any changes to application code. An environment can start with XML based store and as its security needs evolve could change to use LDAP based policy store without application redeployment or code changes.

Oracle Platform Security Services in Action

Figure 3 describes how Java EE application developers can benefits from OPSS's added value.

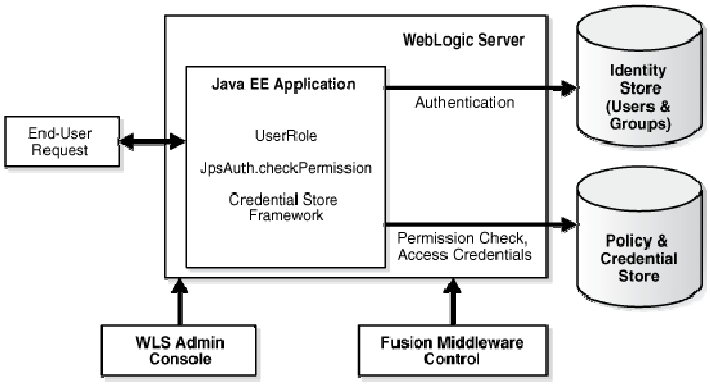

Figure 3 Java EE Application Using OPSS APIs

OPSS offers standards based APIs for authentication, authorization, audit, credentials, policy management, identity access and crypto etc.

OPSS comes pre-integrated with Oracle Application Development Framework (ADF). ADF applications use OPSS authentication, authorization, audit, credential framework and other features in a declarative fashion and are configured through ADF wizards and editors. Most basic

security usecases in an ADF environment do not require coding to OPSS APIs. For advanced usecases developers can also directly invoke OPSS programmatic APIs.

Non – ADF based Java developers can use OPSS APIs in both Java EE and Java SE environments. OPSS integrates with Java EE containers to provide declarative authentication and also provides Login Module based programmatic APIs for authentication & Identity Assertion.

For authorization, OPSS offers checkPermission APIs and will support the OpenAz standard in a future release. OPSS stores all protected application resources in a catalog. OPSS also offers Policy Management API (MAPI) for dynamic and programmatic policy management.

All OPSS based policy modification, authentication, authorization, credential modification and access calls are audited. OPSS provides integration with FMWControl and scripting tools to dynamically enable audit for a selected security service.

OPSS also offers standards based & privacy enabled IGF Aris ID API for accessing Identity attributes stored in a variety of stores. OPSS Credential Store Framework provides APIs to access credentials and symmetric keys in a central keystore.

Java SE applications that need authentication, authorization, and other security features can use the same OPSS APIs available for Java EE applications.

These OPSS security services are also pre-integrated with the management tools used across Oracle Fusion Middleware. This integration allows FMW administrators to use familiar FMW management tools to manage and modify OPSS security policy and configuration.

Oracle Products Using OPSS

Over 100 Oracle products use OPSS. Table 1 provides examples of Oracle products consuming OPSS services.

TABLE 1. ORACLE PRODUCTS USING OPSS

PRODUCT NAME	WHAT IT DOES	HOW IT USES OPSS
Oracle ADF / WebCenter	ADF (Application Development Framework) is an end-to-end Java EE environment that provides out-of-the-box infrastructure services and a visual and declarative development experience.	• Uses OPSS for authentication and authorization (JAAS) • Application roles • Anonymous and Authenticated roles • Policy store abstraction • Policy management • Credential store framework
Oracle Web Services Manager (OWSM)	OWSM provides SOA and web services security.	Authentication, identity assertion, authorization, key store services and audit.
Oracle SOA	Oracle SOA provides components designed to deploy SOA environments (BPEL, ESB, etc.).	Authentication, authorization and audit.
Oracle Service Bus (OSB)	OSB connects, mediates, and manages SOA composites interaction.	Authentication, identity assertion, authorization, role mapping, credential mapping, certificate lookup, audit, SSO, and the SSPI framework for third-party integration.
Oracle Entitlements Service (OES)	OES provides externalized fine-grained authorization.	Authentication, identity assertion, authorization, role mapping, credential mapping, certificate lookup, and audit.
WebLogic Server (WLS) Container	Java EE server / container.	Authentication, identity assertion, authorization, role mapping, credential mapping, certificate lookup, audit, SSO, and the SSPI framework for third-party integration.
Oracle Access Manager (OAM)	Web access and single sign on platform.	Identity assertion and integration with WebLogic Server security.
Oracle Fusion Applications	Next generation of Oracle's packaged applications	All services

Delivering Oracle Platform Security Services

OPSS is delivered as part of Oracle Fusion Middleware (OFM). OFM is hot pluggable, which means it can also run on third-party application servers. Oracle SOA, Oracle Web Center and Oracle IDM 11g ship with OPSS libraries. OPSS is also available with Oracle JDeveloper 11g.

Authorization Policy Manager

Oracle's next generation packaged applications, Fusion Applications, are built with OPSS for its security and identity needs. Authorization Policy Manager (APM) is a Graphical User Interface (GUI) tool that provides Fusion Applications security administrators with rich desktop like drag and drop capabilities for centralized authorization policy management across multiple applications and domains. APM is also available for in house built ADF based applications. APM displays policy related information in a human readable fashion with display names and descriptions to make it easier for security administrators to understand and manage the security artifacts. APM allows administrators to manage the OPSS resource catalog, which is a searchable registry of protected application resources.

Looking Forward

Oracle's strategy for OPSS is to provide a single security platform portable to multiple Java EE containers, for example Oracle Weblogic, IBM WebSphere, and JBoss. Oracle's Fusion Middleware, Fusion Applications, and Vertical Applications already are, or are in the process of uptaking OPSS. The next generation of Oracle Entitlement Server will include OPSS and APM will evolve into the OES 11g Admin Console. Future versions of OPSS will be available independently of Oracle Fusion Middleware and will provide support for programming models beyond Java SE & Java EE.

Conclusion

The security features provided by Java SE and Java EE are often insufficient and require too much knowledge of application developers. OPSS provides security APIs that insulate application developers from underlying identity systems and allows them to focus on business logic.

OPSS is an enterprise-grade, standards-based and portable security platform used by a large number of components of Oracle Fusion Middleware and Oracle Fusion applications. OPSS supports the complete application life cycle including application design, development, deployment, runtime, management, monitoring, upgrade and maintenance, and provides Oracle JDeveloper integration, management tooling, and extensive documentation and samples. As the security platform used by Oracle for its own products, OPSS is now also available to customers of Oracle Fusion Middleware for writing custom applications.

Iintroducing Oracle Plaform Security Services
July 2010
Author: Vinay Shukla
Contributing Authors: Roger Wigenstam, Marc Chanliau

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com

Oracle is committed to developing practices and products that help protect the environment

0109