An Oracle White Paper
May 2013

Oracle Mobile and Social Access Management

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

Mobile computing gradually allows us to make the elusive “anytime, anywhere access” mantra a reality. Online social identities allow us to access web sites using existing identities from leading social networks such as Facebook, LinkedIn, or Twitter.

Mobile computing is blurring the difference between personal and business use. Many companies recognize the importance of personal mobile devices for business use: users can access corporate resources from their mobile devices at their convenience to improve productivity, and companies can enable access to corporate resources through native mobile applications to improve user experience.

However, introducing mobile devices in the enterprise presents additional security challenges. Mobile devices need to blend seamlessly into the corporate computing landscape in order to preserve security without disrupting the workflow of the enterprise. Typically, applications running natively on mobile devices need to integrate with the enterprise-wide identity governance and access control infrastructure for security and compliance reasons.

Oracle’s *Mobile and Social Access Management* is a single, integrated solution addressing both mobile computing and social networks security requirements in order to allow organizations to fully benefit from these disruptive technologies without risk.

The Oracle Mobile and Social Access Management solution includes a new component, the Oracle Mobile and Social server, designed to secure mobile applications leveraging the enterprise's existing back-end identity management infrastructure in terms of single sign-on between browser-based and native mobile applications, strong authentication, device fingerprinting, and device-context-based authorization. Oracle’s Mobile and Social service also provides client software development kits used by developers to weave security into native mobile applications for tight integration with identity management. In addition, the Mobile and Social service enables enterprises to securely leverage social identities for personalization and federated sign-on to help organizations grow their business through social networks.

Introducing Oracle Mobile and Social Access Management

Mobile computing provides new opportunities for accessing corporate resources, but existing infrastructures limit remote or mobile access to browser-based functionality. Mobile users need broader access to more applications and data, from any wireless device. However, accessing corporate resources from and storing data on devices that are often owned by their users presents security challenges to the enterprise. Indeed, a smart phone can easily be lost or stolen, with important data exposed to non-authorized parties.

With more and more organizations establishing a presence on social networks, IT departments require support for social identities, which rely on lighter weight security standards than enterprise identities but are better adapted to the requirements of social networks. For example, some websites may require users to provide access tokens obtained from Facebook or Google in order to be authenticated to their services.

Oracle's Mobile and Social service includes a server that interfaces with existing backend identity management infrastructures. The Mobile and Social server acts as an intermediary between supported mobile client applications and backend identity services. This approach decouples the client applications from the backend infrastructure so that you can modify your backend infrastructure without having to update your mobile client programs.

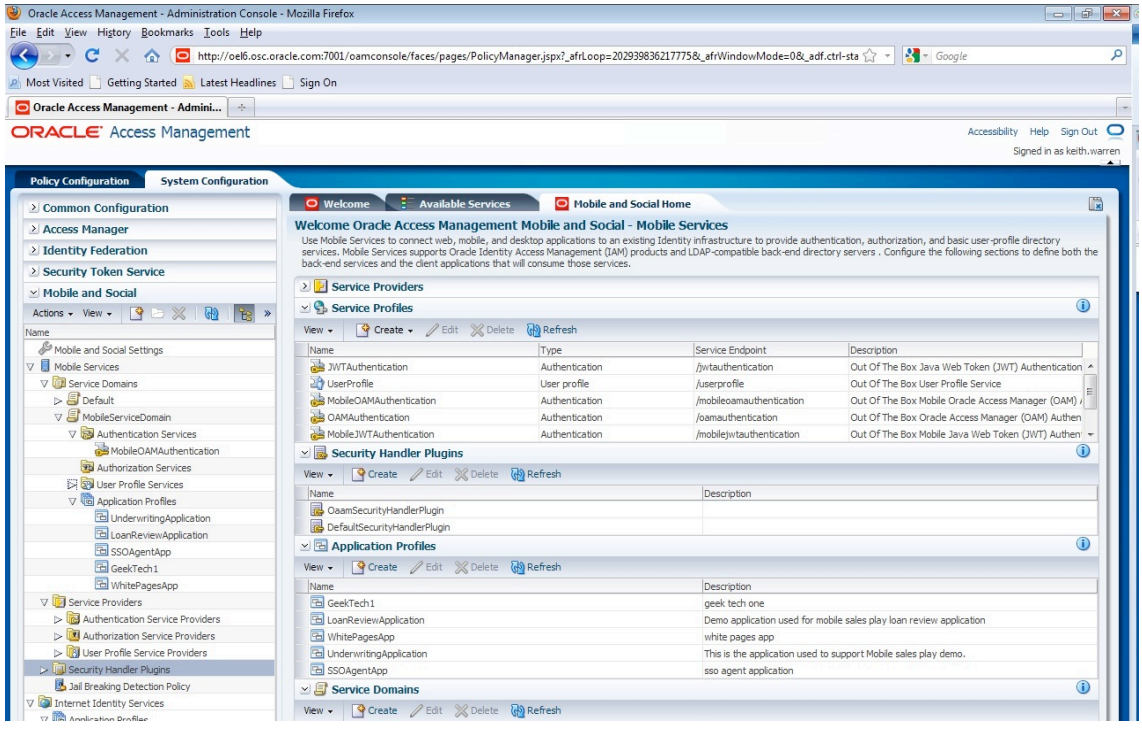

Figure 1: The Mobile and Social service as part of the Oracle Access Management Platform

Oracle's Mobile and Social service includes the following functionality:

- *Mobile Services* connecting browser-based (HTML5) and native mobile applications to the enterprise identity management infrastructure, typically the Oracle Access Management Platform.
- *Internet Identity Services* providing functionality that lets the Mobile and Social solution serve as the relying party when interacting with popular, cloud-based identity authentication and authorization services, such as Google, Yahoo, Facebook, Twitter, or LinkedIn. By deploying Oracle's Mobile and Social service, you provide the user with multiple log-in options without the need to implement access functionality for each identity provider individually.
- *User Profile Services* providing a REST interface for LDAP create, read, update, and delete (CRUD) operations (customers use the same REST interface to build graphical user interfaces for applications), user self-service functions such as self-registration, profile maintenance, password management, and account deletion (see an explanation of REST in the Appendix at the back of this document).
- *Access Management Integration Services* for leveraging Oracle Access Manager (OAM) through a runtime REST interface provided by an agent software development kit.

Customers can install the Mobile and Social solution as a standalone service when not using Oracle Access Manager 11g R2, or when using older versions of Oracle Access Manager.

Standalone mode is used in conjunction with the Oracle API Gateway (OAG) functionality (described later in this document), and User Profile services protected by JSON Web Tokens (JWT). (See the Appendix for a definition and explanation of JSON and JWT.)

Extending Enterprise Security to Mobile Applications

By extending Oracle's unique identity and access management platform approach, customers can securely bring advanced mobile computing into the enterprise. In a typical reference architecture (as shown in Figure 2), Oracle's Mobile and Social service leverages multiple components of Oracle's identity and access management, including:

- Oracle Access Manager (OAM) for web application authentication, authorization, and single sign-on.
- Oracle Adaptive Access Manager (OAAM) for mobile device fingerprinting and registration, risk-based authentication factoring in the mobile device context, and fraud detection. OAAM is tightly integrated with OAM.
- Oracle API Gateway (OAG) for first line of defense supporting multi-protocol and multi-format web services and web application programming interfaces (APIs), security gateway to Cloud services, data redaction (in conjunction with Oracle Entitlements Server), identity propagation, and access to legacy applications.
- Oracle Entitlements Server (OES) for fine-grained authorization policies and access to mobile applications based on the mobile device context.

- Oracle Directory Services for direct access of mobile applications to LDAP-based user directories such as Oracle Internet Directory (OID), Oracle Directory Services Enterprise Edition (ODSEE), and Oracle Unified Directory (OUD).

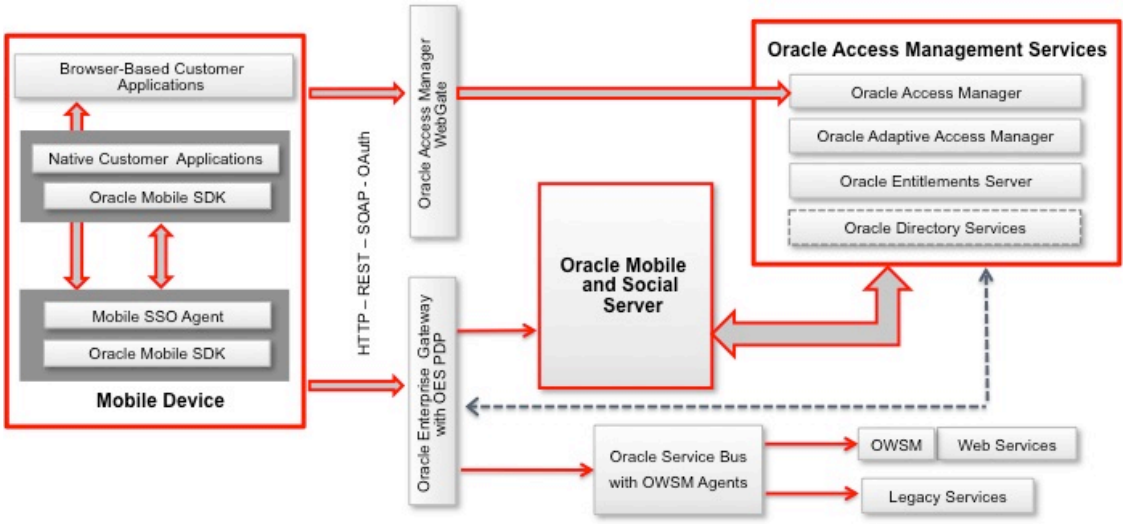

Figure 2: Extending the Oracle Access Management Platform to Mobile Applications

Authenticating Mobile Users

The Mobile and Social solution provides authentication services that let you extend an existing authentication infrastructure to new mobile applications as well as non-mobile applications already in place (authentication is provided by OAM and OAAM when strong authentication is required).

Mobile services support the following common token types:

- A *User Token* granting the token bearer with the permissions associated with the person who has been authenticated.
- An *Access Token* granting access to a specific protected resource, such as an enterprise web application.
- A *Client Token* granting access to a non-mobile device, such as a web application or server application.

Mobile services also use client registration handles, which are similar to client tokens. A client registration handle represents a mobile client application running on a mobile device, such as a device running Apple iOS or Android.

Because mobile devices and non-mobile devices present different security challenges, mobile authentication and non-mobile authentication are managed separately by the Mobile and Social service.

You can configure Mobile services and Social Identity services to work together. For example, you can use Social Identity services to let users authenticate with Google, Facebook, or Twitter, and you can

use Mobile services to provide local authentication functionality, or generate a user token by accepting a user identity assertion from a social Identity Provider (see example scenarios later in this document).

Authorizing Mobile Users

Authorization is handled by Oracle Entitlements Server (OES), a policy-based, fine-grained authorization server designed to externalize authorization from applications and make decisions regarding the access of an authenticated party to protected applications.

Oracle Entitlements Server provides organizations with fine-grained control over mobile users and applications:

- What REST APIs a given client or application can invoke.
- What business transactions a given user or mobile client can submit.
- What data a given user is able to access, and what the user can do with this information.

Authorization policies are defined in accordance with the Attribute Based Access Control (ABAC) and Extensible Access Control Markup Language (XACML) standards, allowing organizations to define policies that include environmental, device, resource, user, and transaction attributes and values. For example, a mobile user should only be able to submit a given type of transactions if the device is trusted and if the transaction amount is less than $1,000.

OES and the Oracle Access Management platform provide a unique end-to-end solution that enables context-aware security policy management based on Identity Context, a service that is part of Oracle's identity and access management offering.

Identity Context is made up of attributes known to the multiple identity and access management components involved in a transaction. Identity Context attributes include user profile (typically stored in a user directory), application and enterprise roles, authentication type (weak, strong), device status (known, managed, trusted), device context such as location and configuration information, federation (partners' attributes), and risk assessment (pattern analysis), as well as network and other devices' information.

Identity Context is shared across Oracle's identity and access management components, and Identity Context attributes, especially device context for mobile access, are made available to the security components designed to make access decisions, in particular OES for authorization.

Oracle's Mobile Access Management solution with Mobile and Social, Identity Context, Oracle Entitlements Server, and Oracle API Gateway allows organizations to enforce fine-grained access control without making any changes to their existing backend systems and applications. Access control is enforced in the Oracle API Gateway layer through which all REST traffic between mobile applications and an organization's backend systems is routed.

Integrating with User Directories

User Profile services let web, mobile, and desktop applications perform a variety of directory lookup and update tasks. User Profile services make it possible to build an application that lets users in your organization access user profiles from mobile devices through REST calls to the Mobile and Social server (see Figure 4 later in this document).

Providing Single Sign-On Across Mobile Applications

Mobile single sign-on (SSO) allows a user to run multiple mobile applications on the same device without having to provide credentials for each application. Both native and browser-based applications can participate in mobile SSO.

For mobile SSO to work, a customer application installed on the mobile device needs to be designated as a *mobile SSO agent*. With Oracle Access Management Platform 11gR2, mobile SSO agents are supported for Apple iOS and Android devices. Users build and brand their own mobile SSO agent applications using the Mobile and Social Client SDK for iOS and Android (described later in this document).

The mobile SSO agent application serves as a proxy between the remote Mobile and Social server and the other applications on the device that need to authenticate with the back-end identity services. The agent can either be a dedicated agent (that is, an application that serves no other purpose), or the agent can be a business application that also provides SSO agent functionality (see Figure 2).

Mobile SSO agents and mobile SSO client applications using Oracle's client SDKs are configured on the Mobile and Social server. Typically, the client application sends the device registration request, the application registration request, and the user token request to the SSO agent, and the SSO agent makes the necessary acquisitions on behalf of the client application (device registration is handled by Oracle's client SDK). The application uses the client SDK authentication API. After authentication, the client application can then request any access tokens it needs because it has the registration handle and user token necessary to do so.

Mobile application developers benefit from using the mobile SSO agent because it handles device registration and advanced authentication schemes (including multi-factor and one-time password authentication), which means that this functionality does not have to be built into each mobile application. The mobile SSO agent also centralizes the task of collecting local device attributes to be passed to the server for risk-based authentication and Identity Context. When the mobile SSO agent is present, user credentials are never exposed to the mobile business application.

A browser-based business application can be configured to use a mobile SSO agent for authentication. When the browser-based request is intercepted by an OAM WebGate, the WebGate defers to the OAM server, which detects the mobile browser and sees that the authentication scheme is set to Mobile and Social. OAM calls the SSO service on the Mobile and Social server, which then redirects to the mobile SSO agent on the user's device. The SSO agent then requests an access token for the resource (on behalf of the business application) and redirects the browser to the URL of the business application with the access token included in the HTTP header.

Similarly, REST web service calls from native or browser-based applications are intercepted by Oracle API Gateway, which brokers SSO for these REST service invocations by validating the tokens against Oracle Mobile and Social (or Oracle Access Manager), before a request to target backend services is granted.

Native and browser-based applications can be opened on the device without asking the user to provide credentials. A business application will fail if configured at the server for SSO, with the SSO application missing in the device. A business application can only directly collect and send credentials if the server-side is configured to allow that.

The mobile SSO agent can additionally time-out idle sessions, manage global logout for all applications, and help in selective device wipeouts. The SSO agent one-way encrypts and locally stores user passwords.

Using Mobile Services with OAM

Enterprise resources may be secured today by a web access solution such as OAM, or they may be SOAP- or REST-based APIs and web services protected by Oracle Web Services Manager (OWSM) and Oracle API Gateway (OAG) as shown in Figure 2.

The Mobile and Social service supports multiple types of resources by offering two token types to secure the path between mobile applications and resources: OAM tokens (HTTP cookies) and JWTs (see the Appendix for a definition of JWT).

The Mobile and Social client SDK (described later in this document) handles authentication programmatically after the SDK collects user credentials using the credential collection user interface. The SDK then uses the Mobile and Social REST interfaces to authenticate the user with the token service configured for the application.

OAM-generated tokens are delivered as JSON payload by the Mobile and Social solution (see the Appendix for a definition of JSON). The application developer extracts the received token and incorporates it into the resource request. When presented to an OAM interceptor (WebGate or AccessGate) by a mobile application, these tokens are validated by the OAM policy server, and they allow access to any type of resource protected by OAM without the OAM interceptor requesting a browser redirection for authentication.

JWTs are generated by the Oracle Platform Security Services framework (OPSS). (See the Appendix for an explanation of OPSS.) JWTs are issued and validated by the Mobile and Social server. These tokens are OAuth-compliant (see the Appendix for an explanation of OAuth), and they can be enforced by a solution that can accept JWTs (e.g., Oracle API Gateway) and validate those tokens against the Mobile and Social service.

Protecting Mobile Applications and Web APIs with OAG

Organizations build mobile applications to enable anywhere, anytime access to information stored in databases, content management systems, and, in some cases, mainframes on the corporate network. The information users should be able to access, and the various types of business transactions that

users should be able to submit from mobile devices have in the past often been available to users through applications hosted within the corporate network, accessible through devices issued and managed by the organization. As such, corporate systems often have little, if any, security and compliance controls built in and instead rely on some degree of implicit trust.

Security and access control becomes a critical requirement now that organizations need to expose their internal systems to devices running outside the corporate network, accessed by internal and external users, from unknown locations, and over potentially unsecure networks. As a consequence, organizations must be able to control and audit what kind of business transactions can be submitted, what information leaves the corporate network, under what circumstances.

Mobile applications typically access corporate information through lightweight REST-based APIs as mobile devices lack support for more involved application, web services, and SOA-based infrastructures using the SOAP, Java Message Service (JMS), Message Queue (MQ), or even File Transfer Protocol (FTP) technologies that existing corporate systems often rely on.

Oracle's complete Access Management solution is designed to address these challenges. With Oracle API Gateway (OAG), organizations can expose internal systems and corporate data as fully secure REST-based APIs (using JSON payloads) without the need for any coding; this is achieved by virtualizing the existing backend SOAP or JMS services as REST APIs through OAG. Existing transport protocols and security tokens required for authentication, identity propagation, and user claims (attribute assertions) can also be automatically transformed to address modern mobile requirements without changing existing backend systems. For example, an organization may only want to accept REST-based JWT tokens issued by Oracle's Mobile Access Management solution; once authenticated the tokens can be converted to SAML or any other type of token required by the SOAP-based backend systems.

OAG adds many capabilities to an organization's REST API infrastructure: API access, transactions, and the data requested/returned can be monitored and audited. Requests from mobile clients (or business partners and Cloud applications) can be validated to ensure they are properly formed, are free from any malicious content and threats such as SQL injection attacks, denial-of-service attacks (even based on message payload content), viruses, and a large number of other XML, cryptographic, and other types of threats. Throttling policies can be defined to ensure that certain types of clients – perhaps based on different mobile applications or the users subscription level (gold, silver, bronze) – can only perform a given number of transactions over a specified time interval, allow an organization to charge for API usage if desired, and ensure that rogue clients do not overload the system with excessive or malicious requests.

As a mobile and Cloud access gateway, OAG provides the following control features:

- Validation of HTTP parameters, REST query and POST parameters, XML and JSON schemas.
- Protection against denial-of-service (DoS), SQL injection, and cross-site scripting attacks.
- Creation and exposure of virtual APIs tailored to mobile consumers and mobile applications formats.
- Throttling, rate limiting, and quota controls over web API traffic.

- Full OAuth 2.0 support, acting as an authorization server and resource server for both 2- and 3-legged scenarios.
- Context- and content-based routing.
- Data redaction and fine-grained access control over mobile business transactions leveraging OES.
- Reporting and analytics for tracking and metering API usage.
- Auditing and logging web API usage for each mobile client.
- Mapping of XML to JSON for consumption by mobile devices.
- Response caching for common web API requests, and response aggregation.
- Brokering of calls to Cloud services, and centralized cloud connectivity.

Oracle API Gateway is part of Oracle's complete Mobile Access Management solution and integrates with Oracle Access Manager and the Access Management Mobile and Social solution for authentication, validation of user tokens , fraud detection, and Identity Context propagation; Oracle Entitlements Server for authorization and audit of REST API access, transactions, and selective data redaction of the response payload; and Oracle Directory Services for user lookup and enrichment of the message payload (see Figure 2 and the use case example later in this document).

Validating Device Identities

The Mobile and Social service enforces new layers of authentication by requiring both device and application registration. Each application communicating with the Mobile and Social service downloads configuration parameters, and obtains an application registration handle that is required for all subsequent requests from that application. When SSO is configured, the application registration for SSO serves as a device registration, and that device registration handle is required for all subsequent requests. Device registration is also subject to the policies and risk assessments available in Oracle Adaptive Access Manager (OAAM). These policies can trigger step-up challenges such as knowledge-based authentication (KBA) or one-time passwords (OTP) delivered via email or SMS text. Only post-authentication and challenge policies are supported through OAAM integration in 11gR2 (other policies will be supported in a later release).

OAAM policies can be implemented for first-time access, so new device registrations require KBA, or more sensitive applications can require OTP (see Figure 3). Policies can also be defined for specific users, allowing users with lower levels of access in with a username and password, but requiring an OTP for users with more privileged access.

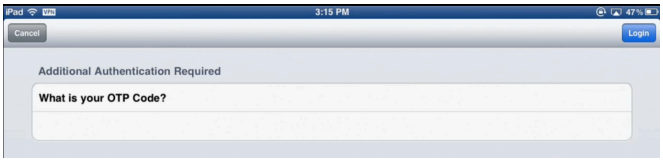

Figure 3: OAAM challenge for one-time password

For organizations wanting stricter control over the devices accessing their systems, devices can be required to be pre-registered in OAAM before their applications can authenticate or obtain tokens. Each request to the Mobile and Social service can be required to provide additional device Identity Context data such as operating system platform and version, Mobile and Social SDK version, "jailbreak" status, VPN status, telephone number, Media Access Control (MAC) address, and location data. This data can be used for device fingerprinting and data reporting, as well as fine-grained authorization to determine access rights (see *Authorizing Mobile Users* section above).

Through OAAM, device Identity Context data can be stored and used to create a more comprehensive fingerprint that can be compared to previously stored fingerprints or device attributes, and policies can use the device data to determine risk and respond accordingly. On its own, the Mobile and Social service contains basic capabilities to detect changes in the device from request to request by using one of the attributes (such as a MAC address) to identify a unique device.

Addressing Device Loss and Theft

Smart phone loss and theft create a high security risk for users and companies, particularly when these devices are used to access corporate resources. The Mobile and Social service working in conjunction with OAAM addresses this risk by providing a way to mark a device lost or stolen, and then implement specific policies that are enforced when a stolen device tries to access enterprise applications.

Since device Identity Context data is delivered to OAAM each time a device attempts to communicate with the Mobile and Social server, OAAM has the ability to challenge a user if the device has been reported lost, or deny any access from a device if the device has been reported stolen.

Additionally, if the device attempts to communicate with the Mobile and Social server after being reported lost or stolen, the Mobile and Social service can reply with an instruction to the device to wipe out all authentication tokens and handles stored in it (that instruction can also be leveraged by mobile applications as a trigger to clear cached data). The wipe instruction executes in one of two ways, based on the policies configured by the customer:

- If OAAM is invoked and the device is marked as stolen, OAAM can deliver an error status of "blocked" with a sub-code of "wipe" which is executed if an application attempts to authenticate against Mobile and Social;
- If the Mobile and Social service is configured with jailbreak detection on, and the policy is set to wipe when the device is jail broken, the wipeout happens as part of the re-authentication process.

Finally, if anomalies are detected in the access pattern, Oracle API Gateway can be configured to completely shut down access to REST APIs and corporate resources.

Exposing Directory Data Through User Profile Services

LDAP directory services are used for many functions, including user self-service, company white pages, or help-desk user account maintenance. The Mobile and Social solution makes directory services available to mobile devices, without a need for building LDAP clients (you still need a mobile client such as a white-list application).

The Mobile and Social service provides REST interfaces to Oracle Directory Services (including OID, ODSEE, and OUD) as well as third-party directory services such as Microsoft Active Directory. The Mobile and Social solution's User Profile services gives users and administrators access to configured directories for many common functions, and provides additional outward-facing security layered on the directory's own security.

Figure 4: Looking up directory services from a native iPad application

User Profile services include the ability to search, view, create, update and delete users, groups and relationships (such as a user's manager), subject to both directory permissions and an optional layer of Mobile and Social permissions. These services are protected by either an OAM token or a JWT, and they can also require device and application registration.

User Profile services are essential for user self-service functions such as self-registration, profile maintenance, password management, and account deletion. Corporate or community white pages are another common application using User Profile services. A user can look up other users (see Figure 4), navigate up and down the management chain, and copy a contact into their mobile device's contacts application. Users can also update attributes on their own record, such as mobile phone number or home office address.

Mobile directory administrative tools can also be created. An authorized administrator can use a native mobile application to create users, set passwords, delete accounts, create and update groups, or change manager relationships.

Logging and Auditing Mobile Transactions

The Mobile and Social service offers two types of auditing: auditing as provided by OAAM, and Mobile and Social audit events and diagnostic logs.

The Mobile and Social service uses post-authentication policies and challenge policies from OAAM. The audit events corresponding to those policies and their evaluation are supported through the integration of OAAM with the Mobile and Social server.

The Mobile and Social server logs each REST transaction to its own log. Diagnostic logs use Oracle Diagnostic Logging (ODL), and Mobile and Social logs go to the standard log files controlled by Oracle Fusion Middleware (log levels are configured through Oracle Fusion Middleware configuration settings).

Mobile and Social Use Case Scenarios

This section provides three typical examples of use case scenarios. We present two examples focusing on the Mobile and Social server, and we show how Oracle API Gateway and Oracle Entitlements Server contribute to the overall Mobile and Social solution.

Authenticating to a Service Provider Resource Through OAuth

In this use case, a mobile user accesses a resource exposed by a Service Provider (SP) using OAuth tokens (the OAuth process flow can be very involved in some cases, please refer to the Appendix for a detailed explanation of OAuth's purpose and process flow).

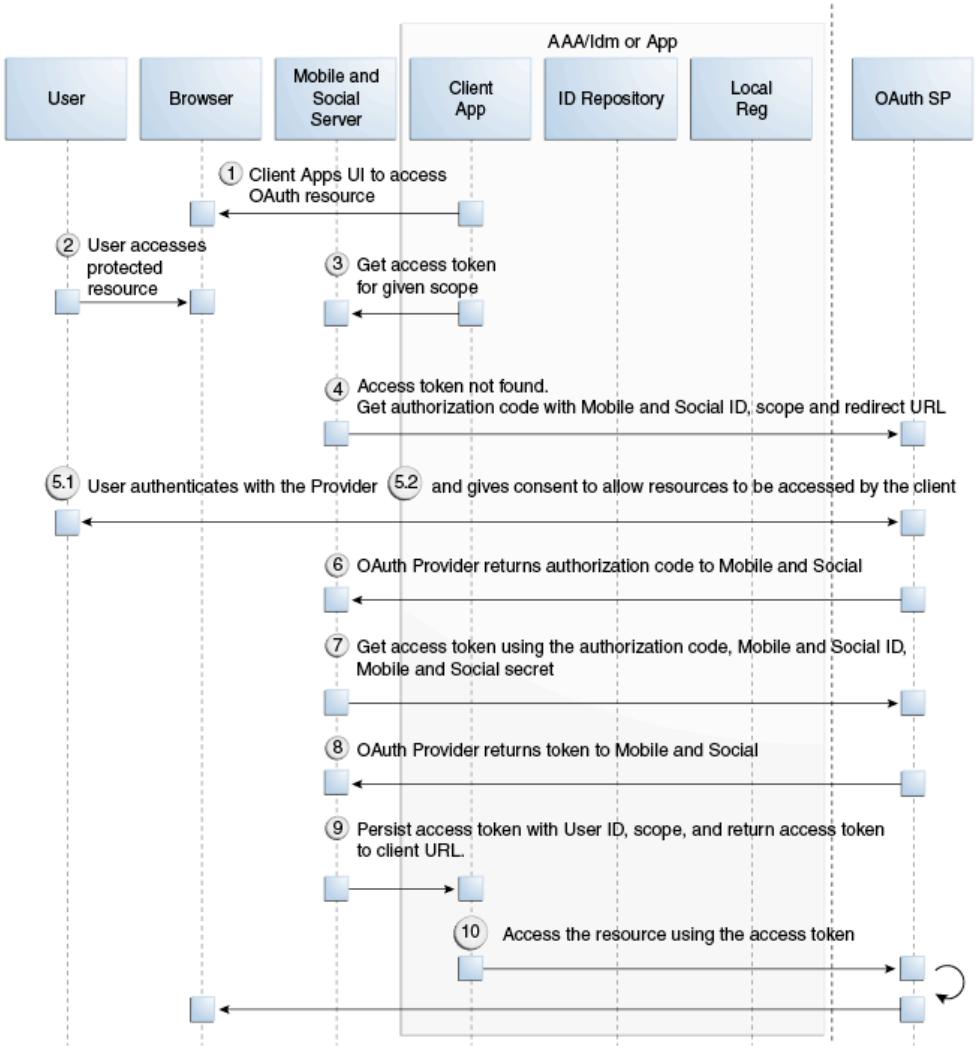

Figure 5: Authenticating to a Service Provider through OAuth

Accessing Resources Leveraging OAM and OAAM

In this use case, a mobile user establishes single sign-on between mobile applications through the SSO agent and leverages the Oracle Access Management Platform for enhanced security.

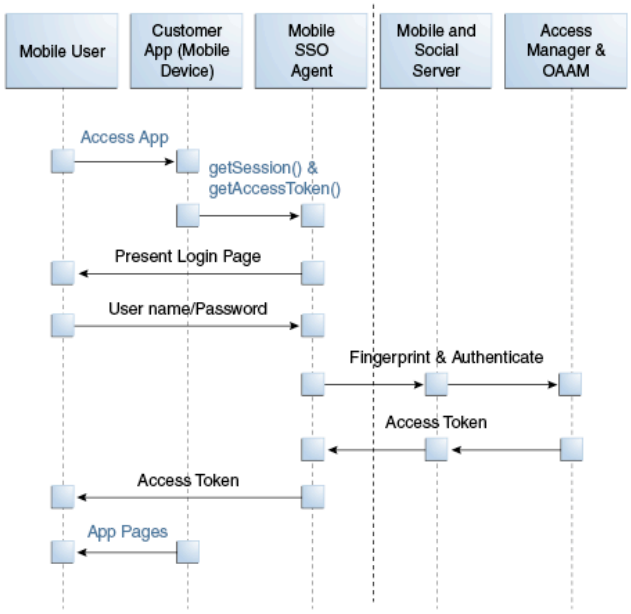

Figure 6: Accessing enterprise resources leveraging OAM and OAAM

Mobile Authorization and Data Redaction

Oracle API Gateway (OAG) provides first line of defense in the corporate DMZ. OAG is used as an access gateway (for external applications), an API gateway (extending new and legacy enterprise application APIs to support mobile and Cloud applications), a Cloud gateway (securing access to Cloud services), a business-to-business (B2B) gateway, and a JSON and XML firewall and accelerator.

Oracle Entitlements Server (OES) is Oracle's strategic fine-grained authorization solution. OES allows users to externalize authorization from applications into a single, centralized point of administration. OES provides a standards-based policy model to represent complex authorization requirements in heterogeneous deployments such as the Java platform, Microsoft .NET, and SOA ecosystems.

OAG natively integrates with OES. OAG ships with an out-of-the-box adapter that calls OES Java Security Service Module (SSM) APIs. OAG automatically becomes a managed component in OES's centralized administration console. This means that all aspects of configuration, provisioning, and management of OAG's authorization policies can be seamlessly integrated with the rest of your enterprise using OES's attribute-based access control (ABAC) and resource management.

This use case scenario shows how OAG and OES combine their specific functionality to provide context-aware authorization of mobile business transactions, authorization for REST APIs, and

selective data redaction in the response payload (the authorization service provided by OES can also be directly exposed to mobile clients).

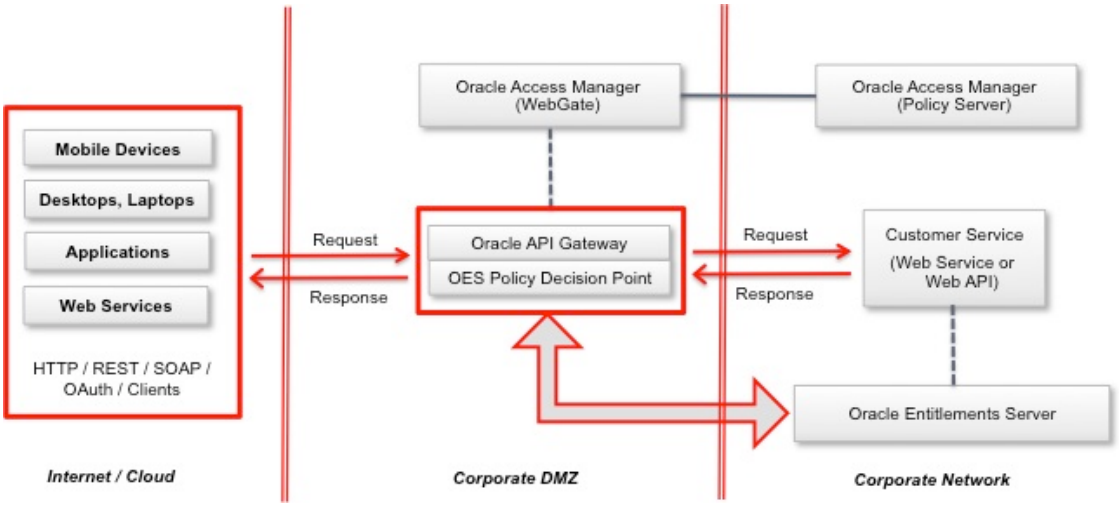

Figure 7: Mobile authorization and data redaction with OAG and OES

Use Case Process Flow

- A mobile device sends a request for accessing a corporate resource (e.g., a business application) over HTTP/REST; Note that the same use case would apply to a request generated from a laptop or desktop computer, an external application, or an external web service over SOAP.
- OAG intercepts the request and authenticates the request either leveraging Oracle Access Management components such as OAM, or directly against a directory service (e.g., Oracle Unified Directory or Microsoft Active Directory).
- Through its tight integration with OES, OAG invokes OES to authorize the request.
- Upon successful authentication and authorization, the user request gets to the endpoint, an enterprise business application in this case.
- The enterprise business application processes the request and sends a response back to the requester.
- The response is intercepted by OAG. OAG makes an authorization request to OES.
- Based on OES's configured policies, some confidential information needs to be removed ("redacted") from the response.
- OES provides the response to OAG (including the redacted information).
- OAG sends the response back to the requester.

Depending on the data format and transport protocol supported by the client application and the targeted resource, OAG supports REST-to-SOAP and SOAP-to-rest conversion. For example, the client may call a REST service and expect a SOAP response. In this case, OAG extracts the REST

request attributes, validates the JSON schema, retrieves the attributes with the JSON path, and converts the JSON format to XML.

Mobile and Social Client SDKs

Oracle's Mobile and Social service acts as a proxy between a mobile user seeking to access protected enterprise resources, and the back-end identity and access management services that protect these resources (typically, the Oracle Access Management Platform).

The Mobile and Social service provides client libraries that allow developers to add feature-rich authentication, authorization, and identity capabilities to registered mobile applications. On the back-end, the Mobile and Social server's pluggable architecture lets system administrators add, modify, and remove identity and access management services without having to update software installed by the user.

The Mobile and Social service provides separate client software development kits (SDKs) for Apple's iOS, Google's Android, and generic Java for desktop applications. These client SDKs are designed to build identity security features into your mobile applications and enable you to use your existing identity infrastructure for authentication, authorization, and directory-access services.

Client SDKs allow developers to get native mobile applications to interact with Oracle's Mobile and Social service through REST calls.

The table below summarizes the functionality provided by the iOS and Android SDKs as well as the Java SDK.

FUNCTIONALITY	IOS / ANDROID	JAVA
Build a mobile application that can acquire a client registration handle, user, and access tokens through the Mobile and Social server.	X	
Build a desktop application that can acquire client, user, and access tokens through the Mobile and Social server.		X
Interact with a user directory server and implement User Profile services.	X	X
Create a mobile single sign-on application.	X	

The Mobile and Social client SDK for Apple iOS can be used in the Xcode development environment on a Mac, the Android SDK can be used in the Eclipse environment (supported on multiple operating systems).

Conclusion

Oracle provides a unique, enterprise-wide, end-to-end Mobile and Social Access Management solution including the new Mobile and Social service tightly integrated with other Oracle Access Management Platform components such as Oracle Access Manager (web single sign-on), Oracle Adaptive Access Manager (strong authentication, device fingerprinting), Oracle API Gateway (REST and SOAP security), and Oracle Entitlements Server (device-context-aware authorization). Oracle's Mobile and Social service is the secure, server-based intermediary between mobile device applications and users, and the enterprise's back-end identity and access management infrastructure.

Oracle Mobile and Social Access Management provides the following encompassing security services: Single sign-on across native and web-browser-based mobile applications, device registration, device context data collection (used across a single transaction for fraud detection and fine-grained authorization), mobile user profile services (for direct access to user directories), the ability to expose existing corporate systems to mobile devices and the Cloud through REST-based APIs without making changes to the organization's existing backend systems, and mobile client development kits to weave security into mobile applications installed on wireless devices.

For more information on the Oracle Access Management components contributing to the Oracle Mobile and Social Access Management solution, please visit Oracle's website at http://www.oracle.com/identity.

Appendix: The New Mobile Computing Paradigm

With the advent of more powerful and functional tablets and smart phone offerings, mobile computing is fast becoming the "new normal." However, because of the (still) limited capabilities of these wireless devices (reduced processing power and network bandwidth), new, more lightweight development and deployment techniques need to be used. This appendix briefly summarizes the new technologies and industry standards that characterize mobile and social computing.

Mobile Application Development Models

There are three ways to develop mobile applications for heterogeneous smart phones and tablets: web browser, native platform software development kit (SDK), and cross-platform development.

- Browser-based development requires HTML5, Cascading Style Sheets (CSS) and JavaScript. This mode of development targets the many web developers already familiar with these technologies, but it does not allow developers to fully leverage the specific features of the mobile devices, such as gestures or voice recognition.
- Native SDKs are specific to each mobile device platform, for example iOS SDK for Apple's iPhone and iPad, or Android SDK for Android. This approach requires developers to be familiar with the device platform SDK and development environment. The benefit of this approach is that developers can leverage all the physical characteristics of the device through application programming interfaces (APIs). This is the most popular (and end-user friendly) form of mobile device application development.
- Cross-platform development: Applications developed in this mode compile to each device's native platform from a common source code. Examples of such development environments include PhoneGap, an HTML5 application platform that allows one to author native applications with web technologies, or ViniSketch, an editor that allows you to design your application's user interface and data flow, and then generate a PhoneGap project supporting generation for Xcode (iOS) and Ant (Android).

Oracle Platform Security Services

Oracle Platform Security Services (OPSS) is Oracle Fusion Middleware's foundation security layer. OPSS is a Java framework that provides security as a service through open application programming interfaces. OPSS is used by developers to weave security into applications, separately from the applications (security information is stored as metadata in XML files). OPSS services (such as authentication, authorization, or session management) are consumed by the various Oracle Fusion Middleware components (such as Oracle WebCenter, Oracle Integration, and of course, Oracle Identity Management).

REST

Representational State Transfer (REST) is a lightweight alternative to SOAP (defined below) for designing and accessing web services. REST is based on the HyperText Transfer Protocol (HTTP). REST uses clients to initiate requests and servers to process client requests and return responses (clients ask for a specific representation through HTTP context negotiation). While SOAP exposes operations that represent logic, REST exposes resources that represent data. REST allows for different resource representations: text, XML, or JSON (defined below).

JSON

JavaScript Object Notation (JSON) is a lightweight data-interchange format based on a subset of the JavaScript language. In terms of functionality and use, JSON is comparable to XML (although the JSON data model is much simpler than XML, and arguably less powerful). JSON's popularity is growing and some very well known social network companies now use JSON to the exclusion of XML.

JSON Web Token

A JSON Web Token (JWT, often pronounced "jot") represents a set of claims encoded as a signed and/or encrypted JSON object and transferred between two parties (examples of JWT claims include "expiration time" or "JWT identifier"). The Oracle Security Developer Tools (OSDT) package (part of Oracle Platform Security Services described above) includes Oracle JSON Web Token, a full Java solution that provides extensive support for JWTs.

OpenID

OpenID (http://openid.net/) is an authentication standard that any web site can leverage without having to develop its own authentication system. As a user, the OpenID standard allows you to log in to multiple OpenID-enabled sites with a single openID (the name of the token (openID) is the same as the name of the standard (OpenID)).

How to get an openID:

1. As a user, an openID is automatically provided to you by web sites supporting the OpenID standard. For example, john.doe@google.com is a valid openID because Google supports OpenID as an authentication system. Other sites supporting OpenID include Yahoo, Flickr, or Facebook. You can also get an openID from dedicated OpenID sites such as www.myopenid.com.
2. As a web site owner, you can become an openID provider by implementing the openId API (http://openid.net/developers/) or you can become a site that accepts openIDs (http://openid.net/add-openid/).

From a user's point of view, the process flow is as follows, assuming you have an openID (e.g., john.doe@google.com), and visit an openID-enabled web site (e.g., www.example.com) from your openID provider (Google in this case).

1. The openID-enabled web site (www.example.com) challenges you with an HTML form asking for your openID (e.g., john.doe@google.com and your password).
2. Upon successful authentication, you're taken to www.example.com.
3. The openID provider receives a message from www.example.com in order to validate your openID.
4. Your openID provider checks that you are who you claim to be (since you started from Google, the check is transparent to you).

OAuth

OAuth (Open Authorization, hosted at http://oauth.net) is an IETF standard (http://datatracker.ietf.org/wg/oauth/charter/) that allows a *User* to transparently share his private data stored on one site (*Service Provider*) with another site (*Consumer*). For example, an OAuth-enabled photo-sharing site (Service Provider) can allow an individual (User) to use an OAuth-enabled printing web site (Consumer) to print the individual's photos without allowing the printing site to know about the individual's identity.

Parties and Components:

1. Service Provider: A web application that allows access to its resources via OAuth.
2. User: An individual that has an account with an OAuth-enabled Service Provider.
3. Consumer: Web application that uses OAuth to access a Service Provider on behalf of a User.
4. Protected Resources: Data controlled by the Service Provider, which the Consumer can access through authentication.
5. Request Token: A value used by the Consumer to obtain authorization from the User, and exchanged for an Access Token.
6. Access Token: A value used by the Consumer to gain access to the Protected Resources on behalf of the User, instead of using the User's Service Provider credentials.
7. Token Secret: A secret used by the Consumer to establish ownership of a given Token.

OAuth authentication is the process by which Users grant access to their Protected Resources without sharing their credentials with the Consumer. OAuth Authentication is done in three steps:

1. The Consumer obtains an unauthorized Request Token.
2. The User authorizes the Request Token.
3. The Consumer exchanges the Request Token for an Access Token.

The OAuth process flow is as follows:

1. The Consumer requests a Request Token
2. The Service Provider grants the Request Token
3. The Consumer directs the User to the Service Provider
4. The Service Provider directs the User to the Consumer
5. The Consumer requests the Access Token
6. The Service Provider grants the Access Token
7. The Consumer accesses the Protected Resources

Oracle's Mobile and Social solution supports the consumption of both OAuth and openId tokens.

SAML

The Security Assertion Markup Language (SAML) is a standard framework for sharing security information on the Internet through encrypted and digitally signed XML documents. The SAML framework includes 4 parts:

- Assertions: How you define authentication information and attribute statements in XML snippets.
- Protocols: How you ask (SAML Request) and get (SAML Response) the assertions you need.
- Bindings: How SAML Protocols ride on industry-standard transport (e.g., HTTP) and messaging frameworks (e.g., SOAP, defined below).
- Profiles: How SAML Protocols and Bindings combine to support specific use cases (e.g., browser profile, artifact profile, etc.).

In the context of WS-Security (explained below), only SAML assertions are used (the WS-Security framework provides the protocol and bindings).

WS-Security and SOAP

WS-Security is an XML framework that specifies SOAP security extensions. (Originally known as Simple Object Access Protocol, SOAP provides an XML envelope that defines how messages must be structured and exchanged in XML-based web services interactions.) WS-Security also includes profiles that specify how to insert different types of binary and XML security tokens in WS-Security headers for authentication and authorization purposes (SAML assertions are the most common example of security tokens used with WS-Security).

Oracle Mobile and Social Access Management
May 2013
Author: Marc Chanliau, Oracle Identity Mgt
Original Contributors: Dan Killmer, Forest Yin

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

SOFTWARE. HARDWARE. COMPLETE.