Securing Heterogeneous Systems Using Oracle Web Services Manager

Jens Peters
One Fox
Utrecht, the Nederlands

Ronald van Luttikhuizen
Vennster
Utrecht, the Netherlands

Keywords:

Oracle Service Bus, Oracle Web Services Manager, Microsoft .NET, Microsoft Silverlight, WS-Security, Web Services

Introduction

Remember the time when Web Services were new and exciting? They were taking off by promising to provide "true" interoperability and independence from underlying platforms and toolsets. In those days vendors introduced tooling to create and consume Web Services while several WS-* standards were not yet fully matured. No wonder that true interoperability was not achieved right away. Integrating a plain-old Web Service in which the client used the same toolset as the provider was rarely problematic. But trying to consume a Web Service generated in a completely different technology than the client could be a nightmare, especially when one or more WS-* standards, such as WS-Security, were applied.

In this presentation we will investigate a case-study in which an employee portal application built on the Microsoft .NET framework and Silverlight consumes Web Services that are exposed by the Oracle Service Bus (OSB). Needless to say, this scenario involves Web Service interactions between two completely different toolsets. The demonstration describes how to achieve first contact between these toolsets, how to use Oracle Web Services Manager (OWSM) to secure the exposed Web Services in a declarative way and how the portal application interacts with these secured services.

Case study

The case study is based on a public sector project in the Netherlands. The main goal of this project is to increase flexibility and agility in the organization's IT landscape, thereby simplifying future changes in both the organization and its IT systems. This goal, among others, is achieved by gradually transforming the application landscape from a collection of siloed applications that "do a bit of everything" to a landscape of reusable, easily integrated building blocks (or services), each with a clear and specific purpose. These building blocks can be new commercial off-the-shelve (COTS) components, new custom developed services, or existing components.

A simplified overview of the new application landscape and its components is shown in figure 1. The functionalities of the backend systems (yellow) are exposed as Web Services using Oracle Service Bus (blue), whereby underlying specifics such as proprietary protocols and data formats are "hidden" by the bus. Examples of these generic building blocks include:

- A Document Service, virtualizing two existing Document Management Systems;
- A Case Management Service that exposes the functionality of a Case Management System;
- Financial Services;
- CRM functionality that can be accessed through the Customer Service.

The exposed services are consumed and reused by redesigned business processes, such as the grants and permits processes that are implemented using Oracle SOA Suite (red) and a portal application used by employees to support the organization's business processes. The portal is built in Microsoft .NET and Silverlight (green).

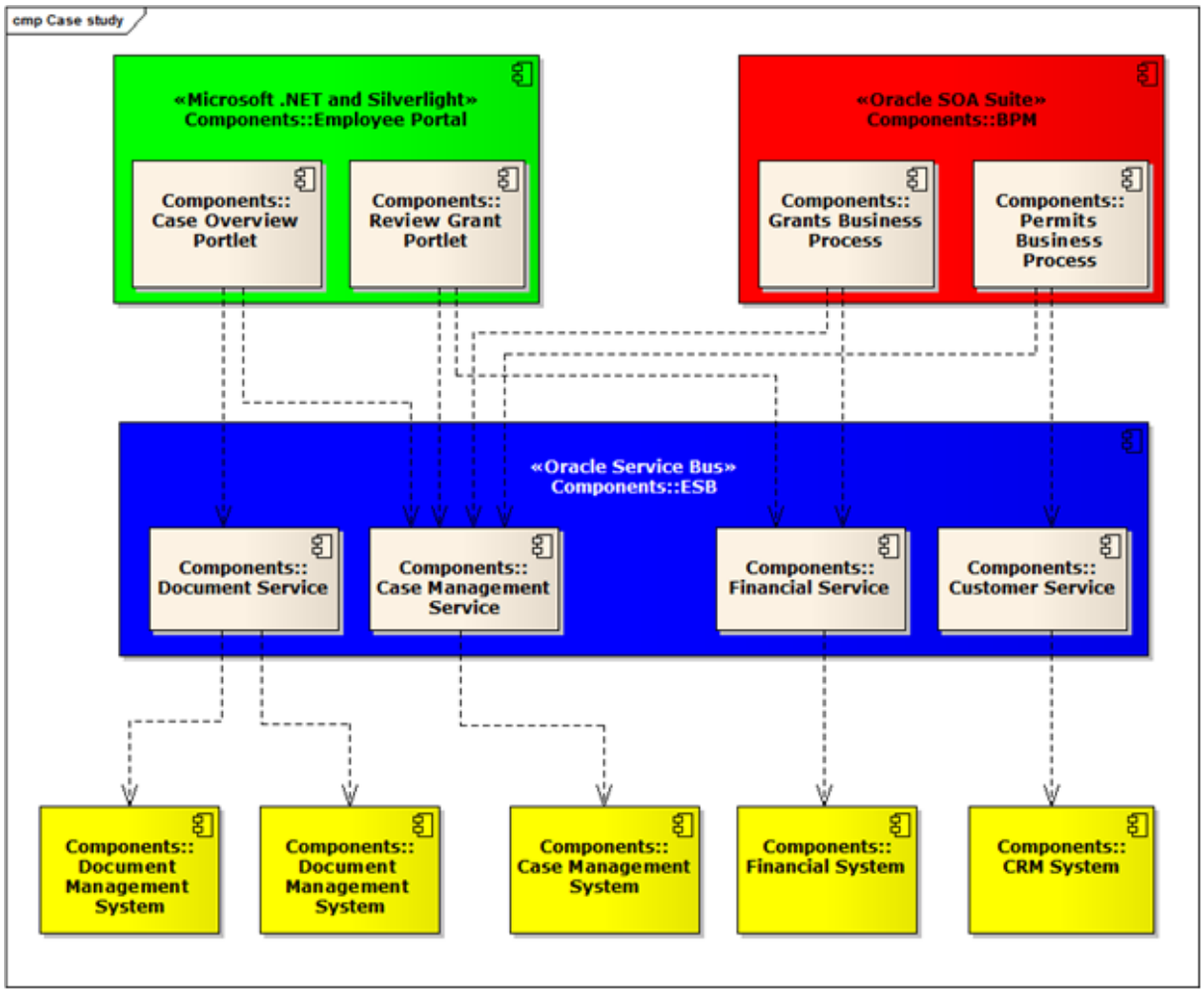

Figure 1

Future systems such as a web based Customer Portal and other business processes should reuse the same services. It is not yet known what the underlying technology of these systems will be, so it is vital that the services are interoperable and provide good quality-of-service, including adequate security.

Requirements and considerations

Many things can be said of siloed and monolithic systems: they can be hard to change, difficult to administer, complex to analyse, and troublesome to integrate and reuse. However, combining all data and functionality into one big monolithic system can be easy from a security viewpoint. Data and functionality are located in one place. If you secure such a system effectively, your data is protected and functionality is secured. In SOA environments data and functionality are divided in different smaller chunks that themselves are simple, flexible and can be easily changed and integrated. As a side-effect, more messages will flow between your services and flow in and out of your organization than is the case for a landscape consisting of only a few monolithic applications. This requires a different view on security. The following concrete aspects of service-orientation can have impact on security:

- Increased amount of machine-to-machine interactions as compared to human-machine interactions. Automated interactions also require security mechanisms such as authentication to be automated;
- There are more intermediate components -- such as ESBs, agents and routing services -- involved in message exchanges between the initial and final recipient of messages. These intermediaries may not operate on these message or may not be allowed to view or change inflight messages. Information needs to be secured "in rest" in addition to being secured solely during transport;
- Reusable services can have lots of consumers, both within and outside your own organization. We need to determine if consumers are allowed access to data and functionality exposed by a service and we must be able to differentiate access control between the various consumers;
- There will be more straight-through-processing in which a whole process or large parts of it are fully automated without any human intervention. Consequences of faults may be detected fairly late because of the higher degree of automation, with significant repercussions due to the higher volume of processed messages, thereby increasing the need for good security and management;
- Increased use of external services in which security is largely outside your immediate control, for example, when consuming Cloud-based services over the Internet. Conversely, your services can be consumed by external organizations. Your services will only be used if consumers have faith in their quality-of-service, including their security;
- Chances are that not all services an organization uses are built in the same technology. Together with the higher degree of integration between services this requires implementation-neutral security standards. Security implementations should adhere to these standards;

Security in IT systems is often implemented on the transport or application layer. Examples of transport security mechanisms include the use of SSL (or TLS) to encrypt and sign messages in transit and to enforce authentication — using 2-way SSL, for example. The downside of using transport-level security alone is that "data in rest" is not secured. This can be the case for messages that are handled in intermediary components such as ESB and BPM platforms. When using message-level security, the messages themselves are encrypted, signed and can contain authentication and authorization information. In these cases data can also be secured while being stored or processed in these intermediaries. Numerous articles and books discuss transport- and message-level security, so we will skip a deep-dive into those mechanisms in this article.

Securing OSB Services [demonstration]

OSB services can be secured using SSL and WS-Security. The demonstration explains how to enable SSL on the Managed Server on which OSB runs and lists the steps to apply OWSM policies using the Oracle Service Bus Console. These steps include:

- Enable SSL on Oracle WebLogic Server using the Console;
- Apply OWSM policies on OSB Services (proxy services) that provide WS-Security capabilities;
- Configure the authentication provider in WebLogic Server using the Console;
- Test the secured services using soapUI.

Introduction to .NET/Silverlight

Microsoft Silverlight is a platform primarily used for creating Web-based applications. Silverlight requires a browser plugin comparable to Adobe Flash or Java Applets, rather than HTML 5 or server-based frameworks such as ASP.NET, JSP and JSF that render HTML and CSS. Microsoft Silverlight can be a good choice for creating (Intranet) business applications for companies already running on Microsoft platforms.

Silverlight is based on the .NET framework. The Silverlight framework inherits a lot of the familiar classes and namespaces from its big brother. Windows Communication Foundation is the de facto standard for inter process communication in the .NET platform. Subsequently Silverlight uses a "WCF-light" for accessing Web services.

Adding Security Configuration to a Silverlight Client [demonstration]

The following sections describe how to create a Silverlight application that consumes the secured Web Service exposed by Oracle Service Bus. It shows how to add WS-Security headers to the Silverlight application and enable SSL. These steps also apply to "normal" .NET Windows Communication Foundation (WCF) applications. The demonstration includes:

- Gaining Access to a Cross-Domain Web Service from Silverlight;
- SSL Pitfalls;
- Implementing WS Security in Silverlight;
- Test the Silverlight application.

Summary

The added-value of interoperable standards such as WS-Security is most obvious when integrating heterogeneous platforms such as Oracle and Microsoft tool stacks. Oracle Web Services Manager (OWSM) is Oracle's primary tool to secure services and business processes by applying and enforcing standards such as SSL, WS-Security, and SAML. This is done in a declarative fashion that requires no coding and in which policies (or agents) are applied at runtime to provide the necessary security. OWSM provides dozens of predefined policies that can be readily applied to Web Services exposed by Oracle SOA Suite, OSB, and Oracle WebLogic Server (JAX-WS Web Services).

This presentation showed how to apply OWSM policies to a Web Service exposed by Oracle Service Bus and how to consume this Web Services from a Web Application built in Microsoft Silverlight. Silverlight is closely related to the Windows Communication Foundation (WCF) that also supports WS-Security.

Applying security measurements isn't the most trivial task. Error messages related to security can be shielded to prevent sensitive information (such as platform versions, stack traces, required security configurations, and so) from being exposed to the outside world and payloads can be encrypted. This can make debugging in a secured environment somewhat more difficult.

Luckily several helpful tools are available that assist in implementing and testing security. These include:

- Fiddler – for investigating which data (requests and responses including security headers) are sent over the network;
- soapUI – for testing and mocking Web Services including;
- Oracle Web Services Manager – which provides a logging policy for logging inbound and outbound messages.

Standards usually leave some room for interpretation and vendors rarely implement standards completely to the letter. There can be some smoke when the rubber hits the road in the integration of completely different technologies, such as Oracle and Microsoft platforms. See the Interoperability Guide for Oracle Web Services Manager to minimize the smoke.

Sources

http://www.oracle.com/technetwork/articles/soa/osb-silverlight-owsm-1634070.html

Contact address:

Jens Peters
One Fox
Oudegracht 231
3511 NK, Utrecht, the Netherlands

Phone: +31 (0)30 2324350
Fax: +31 (0)30 2324351
Email jens.peters@onefox.nl
Internet: http://www.onefox.nl/

Ronald van Luttikhuizen
Vennster
Postbus 31457
6503 CL, Nijmegen, the Netherlands

Phone: +31 (0)6 52456043
Email ronald.van.luttikhuizen@vennster.nl
Internet: http://www.vennster.nl/
Twitter: rluttikhuizen