Handling of the demilitarized zone using service providers in SAP

A Iovan and R Robu

1Politehnica University of Timișoara. Computer and Software Engineering Department, Timișoara, Romania
2Politehnica University of Timișoara, Automation and Applied Informatics Department, Timișoara, Romania

E-mail: adrian_iovan@ymail.com

Abstract. External collaboration needs to allow data access from the Internet. In a trusted Internet collaboration scenario where the external user works on the same data like the internal user direct access to the data in the Intranet is required. The paper presents a solution to get access to certain data in the Enterprise Resource Planning system, having the User Interface on a system in the Demilitarized Zone and the database on a system which is located in the trusted area. Using the Service Provider Interface framework, connections between separate systems can be created in different areas of the network. The paper demonstrates how to connect the two systems, one in the Demilitarized Zone and one in the trusted area, using SAP ERP 6.0 with Enhancement Package 7. In order to use the Service Provider Interface SAP Business Suite Foundation component must be installed in both systems. The advantage of using the Service Provider Interface framework is that the external user works on the same data like the internal user (and not on copies). This assures data consistency and less overhead for backup and security systems.

1. Introduction

An ERP system is an integrated, configurable, and tailorable information system which plans and manages all the resources and their use in the enterprise, and streamlines and incorporates the business processes within and across the functional or technical boundaries in the organization. With ERP, an enterprise can automate its fundamental business applications, reduce the complexity and the cost of the collaboration, force the organization itself to take part in the Business Process Reengineering (BPR) to optimize its operations, and finally result in a successful business [1].

As the complexity of information systems in automation increases, so does its potential to get exploited by unauthorized access. Digital information has become an extremely important asset of organizations [2]. Current generations of factory information and automation systems follow a strictly layered hierarchical architecture. Normally an automation system and its information system is divided into three layers (automation pyramid), which are the following:

• Enterprise layer responsible for strategic decisions
• Manufacturing Execution System (MES) coordinating the factory floor by scheduling tasks, reserving resources and handling the flux of products and information
• Field control layer
SAP ERP is enterprise resource planning software developed by the German company SAP SE. SAP ERP incorporates the key business functions of an organization. The latest version (SAP ERP 6.0) was made available in 2006. The most recent Enhancement Package (EHP7) for SAP ERP 6.0 was released in 2013 [3].

The system is splitted in three layers: Presentation Level, Application Level and Database Level (see Figure 1).

In computer security, a DMZ or demilitarized zone (sometimes referred to as a perimeter network) is a physical or logical subnetwork that contains and exposes an organization's external-facing services to a larger and untrusted network, usually the Internet. The purpose of a DMZ is to add an additional layer of security to an organization's local area network (LAN); an external network node only has direct access to equipment in the DMZ, rather than any other part of the network. The name is derived from the term "demilitarized zone", an area between nation states in which military operations are not permitted [5].

The Service Provider Infrastructure (SPI) is an application and UI technology independent layer for business data exposure which is part of the SAP Business Suite Foundation (SAP_BS_FND). Its main goal is to support the application developer in building timeless software and to decouple the UI from the backend completely.

SPI handles huge amounts of data with a great performance and no additional buffering while being minimal invasive to the underlying implementation (arbitrary data repositories can easily be connected). Also dynamic backends can be accessed via the SPI, which for example do not have a static DDIC representation of their data structures or which even change their data type definitions at runtime. Every application that is using the SPI correctly can – out of the box – either run the UI and the backend on the same system or run the UI on a system in the DMZ and the backend on a second system behind the firewall. Furthermore the SPI provides a central enhancement spot which can be used by customers or partners to enhance the application backend modification free [6].

The paper presents a solution to get access to certain data in the ERP system, having the UI on a system in the DMZ and the database on a system which is located in the trusted area. The solution is based on the Service Provider Interface (SPI) framework provided by SAP AG in SAP 7.31 and Remote Function Call technology to link the systems.

In order to use this framework with the DMZ there should be at least a system with the backend and database in the trusted zone and a system with the user interface in the DMZ.

In [7] the authors evaluate cloud computing in the DMZ scenario and the conclusion is that although cloud computing has security deficiencies, the future looks less cloudy as far as more people are being attracted by this topic and pursuing research to improve on its drawbacks.

This article is structured in five chapters which include the introduction, presentation of the used technologies, the handling of the DMZ, experimental results and conclusions.
2. Service Provider Interface (SPI)

2.1. Structure of an SPI Application
An SPI application is identified by a so called Application Building Block ID (ABBID). This Application Building Block (ABB) can represent a complex application with many objects, a single object or even only a small part of an object [6].

An SPI application is identified by a so called Application Building Block ID (ABBID). This Application Building Block (ABB) can represent a complex application with many objects, a single object or even only a small part of an object [6].

Figure 2. Application Building Block (ABB)

2.2. Consumption of an SPI Application
One central class of the SPI allows the consumption of all SPI application – the so called Connector. It implements the same access interface as the Service Providers, so it feels like calling the SP directly, when using the Connector. Internally the Connector e.g. takes care of the system switch, when running in a DMZ scenario [6].

Almost all SP methods return a direct result via export parameters, for example a table with requested node data. However there is also an indirect channel of communication between the SP and its consumer via the so called Collector. This object stores additional information like messages or field control information and thereby reduces the complexity of the SP's interface.

The complete node model of the application is described by the Metadata Provider.

Figure 3. Consumption of an SPI Application [6]
2.3. Application Building Block
An Application Building Block (ABB) can either be a stand-alone application (e.g. material maintenance) or a smaller part of an application (e.g. long text or classification maintenance) which is integrated into the Service Provider Infrastructure (SPI). To create a dedicated Application Building Block for a smaller part of an application especially makes sense if that part can be reused by multiple applications. Otherwise it is advisable not to split an application into several Application Building Blocks. For every Application Building Block there needs to be a Service Provider and a Metadata Provider. Whenever a SPI based application is accessed, either by instantiating a Connector or by using the FPM SPI Integration (FSI) the Application Building Block ID (ABBID) is a mandatory import parameter which identifies the application that is to be instrumented.

2.4. Metadata Provider, Service Provider
A Metadata Provider (MP) describes the complete node model of a Service Provider Infrastructure (SPI) application. This includes for example the data type of nodes as well as available actions and queries that can be executed on those nodes. Beyond the definitions that are required from a technical perspective by application consumers, attributes that have a declarative/explaining character are also contained; i.e. to allow a consumer to understand what is possible with this application [6] (see Figure 3).

The main purpose of the metadata is the description/definition of:
• The application's behavior and capabilities
• The hierarchical node model and its dependencies
• The signature of generic Service Provider interface methods

The SPI offers huge flexibility in the definition of metadata:
• The metadata is provided via coding and therefore allows e.g. also the consumption (the SPI provides two interfaces for metadata consumption. One is the output interface /PLMB/IF_SPI_METADATA_OUTPUT which serves the purpose of reading and evaluating the defined metadata of an Application Building Block (ABB). This interface is returned as export parameter of the Connector instantiation method CREATE_INSTANCE and can be used by the consumer of the SP. The second interface /PLMB/IF_SPI_METADATA is handed over to the Service Provider during its instantiation. This interface includes the output interface and allows in addition manipulating the metadata at runtime, so it is supposed to be used by the SP only) of an already existing metadata model. Also the correction of coding via note is more convenient than the adjustment of customizing entries.
• The complete data type definition can be done dynamically i.e. without a DDIC representation. This allows for example to build up the node's attributes based on customizing entries or maybe even a combination of different DDIC structures.
• The metadata of a node can be modified at runtime (invalidation and re-reading), (e.g. side-effects or data type of a node).

The Service Provider (SP) functions as interface between the Service Provider Infrastructure (SPI) and the application's backend. Its main responsibility is providing access to the application data and handling requests for creating, changing or deleting the data [6] (see Figure 3).

Technically a SP is an ABAP OO class that implements the application access interface /PLMB/IF_SPI_APPL_ACCESS that consists of six node specific methods and three transactional (node-independent) methods. Additional information that is not returned via the access interface methods such as messages and field properties are handed over by the SP to the Collector.

Furthermore the SP needs to implement the initialization interface /PLMB/IF_SPI_APPL_ACCESS_INIT.
3. Handling Demilitarized Zone Using Service Provider Interface

External collaboration needs to allow data access from the internet. In a trusted internet collaboration scenario where the external users work on the same data like internal users (and not on copies) direct access to the data in the intranet is required.

This requires a special and strong security mechanism:

- No direct access from the internet to the system in the intranet is allowed
- Provide a protocol change between request and data access

A solution for this requirement is to separate UI and data in a way that there is no need to offer direct access to intranet (see Figure 4).

![Figure 4. DMZ Scenario](image)

In the DMZ scenario the presentation layer (UI) is running on a system in the DMZ (demilitarized zone) network area which is separated from the system in the intranet where the data is stored. Therefore all data (with a few exceptions) needs to be accessed via RFC function modules. This allows a controlled access to the backend data and avoids a duplication of data needed for external collaboration.

Additional security mechanism may be needed (e.g. specialized collaboration UIs; more detailed authorization in the coding; Access control to the DMZ network area (e.g. using IP filter).

3.1. RFC Setup

User data and a few Customizing Settings (Output Formatting, RFC Connections, Language, etc) are needed in the DMZ system. Everything else has to be retrieved from the backend system by RFC.

To test the DMZ scenario internally an additional client representing the DMZ system is sufficient.

For real tests with external partners, a real DMZ system in the real SAP DMZ area is needed:

- A Security Concept which documents all accesses and connections (protocols and ports) crossing the firewalls has to be approved by the network security responsible (Security from SAP Hosting) [8].
- The DMZ system needs to be connected to the development system landscape to get all transport.
- The user ID in the DMZ system must be identical to the user ID in the intranet system.
- The authorizations for the user in the DMZ system can be reduced to a minimum to increase security.

3.1.1. Settings needed in the Backend system. Transaction SM59 would have to be used to set up a Remote Function Call (RFC) connection from the DMZ system to the backend system. A technical user like ALEREMOTE should not be used for this RFC connection. The settings needed in the Backend system are presented in Table 1.
Table 1. Settings for backend system

| Attribute          | Setting                                      |
|--------------------|----------------------------------------------|
| RFC Destination    | <Backend System Name> CLNT <client>          |
| Connection Type    | 3 (Connection to SAP R/3)                    |
| Load Distribution  | Yes                                          |
| Target Host        | <target host name>                          |
| Trusted System     | Yes                                          |
| SNC                | Inactive                                     |
| Language           | Leave this field empty                       |
| Client             | <client> (Client which has the data, e.g. 002)|
| User               | Leave this field empty                       |
| Password           | Enter the following string: “is initial”     |
| Current User       | Select                                       |

Next saving RFC destination and going back to the RFC Overview screen is needed. Next, in the RFC Overview screen it is necessary to choose from the menu bar "Extras" -> "Trusted Systems" (or Transaction SMT1) and to add the system name of the backend system as "Trusted System".

3.1.2. Registering the RFC Destination as Backend for the DMZ System. It is necessary to enter the RFC connection in the Customizing of the DMZ system (table /PLMB/RFC_DEST) IMG-Path: Cross-Application Components -> Processes and Tools for Enterprise Applications -> Settings for BO Framework and Navigation -> BO Framework -> Define RFC Destinations.

3.1.3. Services. All data accesses using the Service Provider Infrastructure are enabled for the DMZ scenario.

Limitations of the DMZ Scenario affect the attributes with references to backend data which can't be handled in a DMZ scenario. These are:

- Object references to instances existing in the backend
- URLs (e.g. for Icons) generated on the backend

The SPI is designed for accessing Business Object (BO) like Data with CRUD services (Create, read, update and delete), but there are some use cases for data accessing which need a different approach:

- DDIC Search help
- Conversion Exits
- Customizing

3.2. DMZ Enabled Scenarios

3.2.1. DMZ Enabled DDIC Search Help. The DMZ-enabled DDIC search helps are in principle a copy of the normal DDIC search helps, which has:

- The search help exit /PLMB/SPI_SHLP_F41F_GEN_EXIT
• An additional Search help parameter "IV_SHLP" which has as default the name of the normal DDIC search help
  • Data elements which carry also DMZ-enabled DDIC Search helps
    When collective search helps are needed for the DMZ they need to be build up from elementary DMZ-enabled search helps.
    The search help exit /PLMB/spi_shlp_f4if_gen_exit should also be added to the collective DMZ-enabled search help.
    Limitations of DMZ Enabled Search Help are the following:
      • DDIC Search helps with programmed UIs can't be DMZ enabled.
      • Packed numbers (e.g. timestamps) are not supported (limitation of internally used FM BALW_SHLP_BAPI_FILL)
      • Help value is limited to char255 (help values returned in structure BAPIF4C from BALW_SHLP_BAPI_FILL)
    The search help exit /PLMB/spi_shlp_f4if_gen_exit allows the call of application own RFC logic (available with SAP_BS_FND 7.31):
      • Own function modules if the source system does not have SAP_BS_FND installed
      • Own RFC determination) instead of the predefined logic

3.2.2. DMZ Enabled Conversion Exit. Conversion Exits in the DMZ system will not work if they need application data. All other Conversion Exits based on algorithm and/or customizing need no change.
  For example application data is needed if an internal key is mapped to a readable key within a Conversion Exit (e.g. MATN2 for long material number)
  The class /PLMB/CL_SPI_CONVEXIT is used to handle this conversion on the backend.

3.2.3. DMZ Enabled Table Provider for Customizing. Customizing which is needed in the UI (e.g. for configuration) can simply be accessed with class /PLMB/CL_FRW_TABLE_PROVIDER.
  The names of tables which should be read by the table provider need to be populated in table /PLMB/FRW_TABPRO.
  The limitations of table provider are that the applications tables (delivery class A) are not allowed to be used with the table provider.

3.2.4. DMZ Enabled Classes. The Table Provider class as well as the Conversion Exit class using the class /PLMB/CL_RFC_CONNECTOR for the RFC call and they are implementing the Interface /PLMB/IF_RFC_CONNECTOR.
  This mechanism can also be used for application specific needs where DMZ enablement is necessary but the additional services of the SPI are not needed.
  An example for this use case is the DMZ enablement of the Business Context Viewer (BCV).
  In order to enable a class, the DMZ has to implement the interface /PLMB/IF_RFC_CONNECTOR which has one Method RFC_CALL_RECEIVE.
  Within this method the application specific input parameters (exported with method RFC_CALL) can be imported. The requested result needs than to be exported and the export data is delivered back from Method RFC_CALL.
  The data is exchanged by using the ABAP commands export and import. The format in which the data is exported and imported has to be defined by the implementing class.
  It is strongly recommended that the caller method and the called method are implemented in the same class, to have the exchange protocol defined in only one class.

4. Experimental Results
For testing the connections two systems are configured: one in the DMZ Area and one in the Intranet.
  The system in the DMZ Area contains only the UI and using the SPI framework and RFC connection is linked to the Intranet system which contains productive backend data (see Figure 5).
Every application is secured using SAP Authorization objects that are assigned to users. When a user tries to access an application, the authorizations objects assigned to the application are compared with the authorizations objects assigned to the user, and if the users do not have the necessary authorization, he will not be allowed to continue.

The DMZ system contains no productive data on the database layer, so the applications are routed to get the data from the Intranet System. Specific applications only have access to specific data. Access to unauthorized data is not permitted since the productive data is not on the same system as the application. The application gets the data through RFC connections which cannot be used by a standard user to access other data that the ones intended for the application.

5. Conclusions
Using the SPI framework to handle the DMZ makes is easy to manage external collaborations that need access to data in the company Intranet. By following the guidelines of the framework there will be no direct access from the internet to the system and a protocol will be provided to request and access data.

Another advantage is that there is no need to create separate databases for the separate systems and there is no need to administrate the replication of data from one system to another. As a result the effort and capacity to administrate and maintain the system will be low.

6. Acknowledgment
This work was partially supported by the strategic grant POSDRU/159/1.5/S/137070 (2014) of the Ministry of National Education, Romania, co-financed by the European Social Fund – Investing in People, within the Sectoral Operational Programme Human Resources Development 2007-2013.

References
[1] Wei S and Thuraisingham B 2007 Security for Enterprise Resource Planning Systems, Information Systems Security 16 152–163
[2] Byres E and Lowe J 2004 The myths and facts behind cyber security risks for industrial control systems, Proceedings of VDE 2004 Kongress, Berlin, Germany, October, pp 213–218
[3] ***SAP Administration 2006 Practical Guide, SAP Press, SAP AG
[4] ***SAP Netweaver 2005 Overview, SAP Press, SAP AG
[5] Schmidt M, Smith M, Fallenbeck N, Picht H and Freisleben 2007 Building a Demilitarized Zone with Data Encryption for Grid Environments, 1st International Conference on Networks for Grid Applications GridNets07, Lyon, France, October 17-19, article no.17
[6] Schneider M and Walter W 2008 Service Provider Infrastructure, SAP Press, SAP AG
[7] Traian Andrei 2009 Cloud Computing Challenges and Related Security Issues, A project report
written under the guidance of Prof. Raj Jain, http://www1.cse.wustl.edu/~jain/cse571-09/ftp/cloud.pdf

[8] Chen B, Hanck J, P Hanck P, Hertel S, Lissarrague A and Médaille P 2015 *SAP Data Services*, SAP AG