Oracle® Hospitality Simphony

Import / Export API Guide

Release 2.10

E93162-01

February 2018

ORACLE

Contents

Tables

Figures

Preface

This guide describes how to use the programming interfaces of the Import/ Export web service in a secure fashion. It introduces the REST application programming interface (API), reviews security threats and mitigations that are built into the web service, and reviews some secure coding principles.

Audience

This guide is intended for users that are familiar with web services, JSON, and common programming language constructs.

Customer Support

To contact Oracle Customer Support, access My Oracle Support at the following URL:

https://support.oracle.com

When contacting Customer Support, please provide the following:

- Product version and program/ module name
- Functional and technical description of the problem (include business impact)
- Detailed step-by-step instructions to re-create
- Exact error message received and any associated log files
- Screen shots of each step you take

Documentation

Oracle Hospitality product documentation is available on the Oracle Help Center at http://docs.oracle.com/en/industries/hospitality/

The following references provide additional background information that are needed for secure usage of the Import/ Export web service.

- Simphony 2.9 Security Guide
- WebClient Class
- Introduction to JSON
- XML Tutorial
- JSON framework for .NET

Revision History

Date	Description of Change
February 2018	• Initial publication

1 Simphony Import/Export API

Background

The Simphony Data Import/ Export API is a JSON-based, REST web service that helps you to create and submit data import and export job requests. The web service can be used to integrate with third-party applications and programmatically export menu items from one system and importing them to another, among many other features, thereby saving time for administrators or managers. Furthermore, you can efficiently create new software applications by leveraging the web service capabilities of Import/ Export API. The web service allows you to:

- Import and export data programmatically
- Export data automatically using the scheduler
- Review the status of completed import and export jobs

Import/Export API Prerequisites

- Simphony version 2.8 and higher is required.
- Obtain Simphony credentials to access the web service.
- You must have the privileges to access the database and run the Simphony Import/ Export web service.
- In order to use the web service, you must have a web client infrastructure. For example, Microsoft .NET framework offers the WebClient class that can be used to make web service requests. Other programming languages and libraries can also be used to create a web client to use the Import/ Export web service.

Security Service API

In order to use the Import/ Export web service, you must authenticate using valid Simphony credentials. The security service API must be called first to obtain an authentication token. The other APIs of the web service can only be used if you have a valid authentication token.

In your web browser, enter the following URL:

`https://<home>/ImportExportAPI/SecurityService.svc/web/help`, in order to view the methods of the security service. It should show a table of methods, similar to Table 1-1.

Table 1-1 Security Service Operations

URI	Method	Description
`/GetUserAuthenticationToken`	`POST`	Authenticates user and returns a token
`/SignOffToken`	`POST`	Invalidates the valid token

GetUserAuthenticationToken

Description: Authenticates a user and returns a token.

URL:
`https://<home>/ImportExportAPI/SecurityService.svc/web/GetUserAuthenticationToken`

HTTP Method: POST

Request JSON body:

```
{
	"orgCode":"String content",
	"password":"String content",
	"userName":"String content"
}
```

When using this web service request, you must replace the **String content** with valid Simphony credentials.

Response JSON body:

```
"token"
```

With valid Simphony credentials, the web service responds with an authentication token. You need this token in order to successfully use the methods of the data service API which are explained in the next section.

SignOffToken

Description: Invalidates the valid token.

URL: `https://<home>/ImportExportAPI/SecurityService.svc/web/SignOffToken`

HTTP Method: POST

Request JSON body:

```
"String content"
```

When using this web service request, you must replace the **String content** with a valid authentication token that should be signed off.

Data Service API

The data service API provides the core functionality of the Import/ Export web service. The methods of the data service are shown in Table 1-2.

In your web browser, enter the following URL: `https://<home>/ImportExportAPI/DataService.svc/web/help`, in order to view detailed descriptions of the methods of the data service. orgCode is needed in a multi-tenant context.

Table 1-2 Data Service Operations

URI	Method	Description
/AddSchedule	POST	Adds a new schedule. Returns positive schedule Id if successful, otherwise returns -1.
/GetBuildVersion	GET	Returns the Build version of API.
/GetExportableTypes/{userName}/{*orgCode}	GET	Returns the type of objects exportable from this service.
/GetFileData/{requestId}/{userName}/{*orgCode}	GET	Returns the file bytes for a given export request, or an error file if an error occurs
/GetHierarchyStructure/{userName}/{*orgCode}	GET	Returns all the Hierarchy objects (Enterprise, Properties, RVC's and Zones) available for the current user in a given organization.
/GetImportableTypes/{userName}/{*orgCode}	GET	Returns the type of objects importable from this service.
/GetObjectInfo/{userName}/{*orgCode}	GET	Help Info: Returns Object Information.
/GetPropertiesForType/{fullyQualifiedTypeName}/{userName}/{*orgCode}	GET	Returns the attributes (Columns) of objects available on a given object.
/GetRequests/{days}/{userName}/{*orgCode}	GET	Returns all the requests made by this users org for specified up to 30 days with latest status. Includes originally scheduled active requests.
/GetRequestStatus/{requestId}/{userName}/{*orgCode}	GET	Returns the request object with latest status.
/GetSchedules/{userName}/{*orgCode}	GET	Returns all the schedules available in this organization
/GetServerTimeWithZone	GET	Returns server current time and time zone.

URI	Method	Description
/GetSnapShotBeforeImport/{requestId}/{userName}/{*orgCode}	GET	Returns the original snapshot of backed up data in bytes before the given import is processed.
/GetSupportedLanguages/{userName}/{*orgCode}	GET	Returns all the Simphony configured languages for a given organization.
/SubmitImportExportRequest	POST	Submit a request for Import or Export.
/UpdateRequestStatus	POST	Updates the status of a request enabling it to be cancelled.
/UpdateScheduleStatus	POST	Enables or disables a schedule.

AddSchedule

Description: Adds a new schedule. Returns positive schedule Id if successful, otherwise returns -1.

URL: `https://<home>/ImportExportAPI/DataService.svc/web/AddSchedule`

HTTP Method: POST

Request JSON body:

```
{
        "Active":true,
        "DaysOfMonth":[2147483647],
        "Friday":true,
        "Id":9223372036854775807,
        "Monday":true,
        "Name":"String content",

        "OrgCode":"String content",
        "Saturday":true,
        "StartDate":"\/Date(928167600000-0500)\/",
        "Sunday":true,
        "Thursday":true,
        "TimeOfDay":"\/Date(928167600000-0500)\/",
        "Tuesday":true,
        "UserName":"String content",
        "Wednesday":true
}
```

Response JSON body:

```
9223372036854775807
```

GetBuildVersion

Description: Returns the Build version of API.

URL: `https://<home>/ImportExportAPI/DataService.svc/web/GetBuildVersion`

HTTP Method: GET

Response JSON body:

```
["String content"]
```

GetExportableTypes

Description: Returns the type of objects exportable from this service.

URL:
`https://<home>/ImportExportAPI/DataService.svc/web/GetExportableTypes/{USERNAME}/{ORGCODE}`

HTTP Method: GET

Response JSON body:

```
["String content"]
```

GetFileData

Description: Returns the file bytes for a given export request, or an error file if an error occurs.

URL:
`https://<home>/ImportExportAPI/DataService.svc/web/GetFileData/{REQUESTID}/{USERNAME}/{ORGCODE}`

HTTP Method: GET

Response JSON body:

```
[81,
109,
70,
122,
90,
83,
65,
50,
78,
67,
66,
84,
100,
72,
74,
108,
89,
87,
48,
61]
```

GetHierarchyStructure

Description: Returns all the Hierarchy objects (Enterprise, Properties, RVC's and Zones) available for the current user in a given organization.

URL:
`https://<home>/ImportExportAPI/DataService.svc/web/GetHierarchyStructure/{USERNAME}/{ORGCODE}`

HTTP Method: GET

Response JSON body:

```
[{
        "HierarchyId":9223372036854775807,
        "Id":9223372036854775807,
        "ObjectNumber":2147483647,
        "HierarchyName":[{
                "LangId":2147483647,
                "Text":"String content"
        }],
        "OrganizationId":2147483647,
        "ParentHierarchyId":9223372036854775807,
        "UnitType":0
}]
```

GetImportableTypes

Description: Returns the type of objects importable from this service.

URL:
`https://<home>/ImportExportAPI/DataService.svc/web/GetImportableTypes/{USERNAME}/{ORGCODE}`

HTTP Method: GET

Response JSON body:
```
["String content"]
```

GetObjectInfo

Description: Returns Object Information.

URL: `https://<home>/ImportExportAPI/DataService.svc/web/GetObjectInfo/{USERNAME}/{ORGCODE}`

HTTP Method: GET

Response JSON body:

```
[{
        "ColumnInfo":[{
                "AllowNullsOnImport":"String content",
                "AlternateColumnType":"String content",
                "ColumnName":"String content",
                "ColumnType":"String content",
                "DefaultValue":"String content",
                "ExportOnly":true,
                "KeyColumn":true
        }],
        "Exportable":true,
        "FullTypeName":"String content",
        "Importable":true,
        "ShortName":"String content"
}]
```

GetPropertiesForType

Description: Returns the attributes (Columns) of objects available on a given object.

URL:
`https://<home>/ImportExportAPI/DataService.svc/web/GetPropertiesForType/{FULLYQUALIFIEDTYPENAME}/{USERNAME}/{ORGCODE}`

HTTP Method: GET

Response JSON body:

```
["String content"]
```

GetRequests

Description: Returns all the requests made by this user's org for specified up to 30 days with latest status. Includes originally scheduled active requests.

URL:
`https://<home>/ImportExportAPI/DataService.svc/web/GetRequests/{DAYS}/{USERNAME}/{ORGCODE}`

HTTP Method: GET

Response JSON body:

```
[{
        "DataForImport":[81,
        109,
        70,
        122,
        90,
        83,
        65,
        50,
        78,
        67,
        66,
        84,
        100,
        72,
        74,
        108,
        89,
        87,
        48,
        61],
        "DataSince":"\/Date(928167600000-0500)\/",
        "HierStrucId":9223372036854775807,
        "LanguageId":2147483647,
        "Level":0,
        "ObjectType":"String content",
        "OrgCode":"String content",
        "Origin":0,
        "OriginalRequestId":9223372036854775807,
        "RequestDate":"\/Date(928167600000-0500)\/",
        "RequestId":9223372036854775807,
        "RequestName":"String content",
        "ScheduleId":9223372036854775807,
        "SelectedFormat":0,
        "SelectedObjectMembers":["String content"],
        "SelectedOperation":0,
        "SortExpressions":[{
                "m_Item1":"String content",
                "m_Item2":"String content"
        }],
        "Status":0,
        "UserName":"String content"
}]
```

GetRequestStatus

Description: Returns the request object with latest status.

URL:
`https://<home>/ImportExportAPI/DataService.svc/web/GetRequestStatus/{REQUESTID}/{USERNAME}/{ORGCODE}`

HTTP Method: GET

Response JSON body:


```
{
        "DataForImport":[81,
        109,
        70,
        122,
        90,
        83,
        65,
        50,
        78,
        67,
        66,
        84,
        100,
        72,
        74,
        108,
        89,
        87,
        48,
        61],
        "DataSince":"\/Date(928167600000-0500)\/",
        "HierStrucId":9223372036854775807,
        "LanguageId":2147483647,
        "Level":0,
        "ObjectType":"String content",
        "OrgCode":"String content",
        "Origin":0,
        "OriginalRequestId":9223372036854775807,
        "RequestDate":"\/Date(928167600000-0500)\/",
        "RequestId":9223372036854775807,
        "RequestName":"String content",
        "ScheduleId":9223372036854775807,
        "SelectedFormat":0,
        "SelectedObjectMembers":["String content"],
        "SelectedOperation":0,
        "SortExpressions":[{
                "m_Item1":"String content",
                "m_Item2":"String content"
        }],
        "Status":0,
        "UserName":"String content"
}
```

GetSchedules

Description: Returns all the schedules available in this organization

URL:
`https://<home>/ImportExportAPI/DataService.svc/web/GetSchedules/{USERNAME}/{ORGCODE}`

HTTP Method: GET

Response JSON body:

```
[{
        "Active":true,
        "DaysOfMonth":[2147483647],
        "Friday":true,
        "Id":9223372036854775807,
        "Monday":true,
        "Name":"String content",
        "OrgCode":"String content",
        "Saturday":true,
        "StartDate":"\/Date(928167600000-0500)\/",
        "Sunday":true,
        "Thursday":true,
        "TimeOfDay":"\/Date(928167600000-0500)\/",
        "Tuesday":true,
        "UserName":"String content",
        "Wednesday":true
}]
```

GetServerTimeWithZone

Description: Returns server current time and time zone.

URL:
`https://<home>/ImportExportAPI/DataService.svc/web/GetServerTimeWithZone`

HTTP Method: GET

Response JSON body:

```
["String content"]
```

GetSnapShotBeforeImport

Description: Returns the original snap-shot of backed up data in bytes before the given import is processed.

URL:
https://<*home*>/ImportExportAPI/DataService.svc/web/GetSnapShotBeforeImport/{REQUESTID}/{USERNAME}/{ORGCODE}

HTTP Method: GET

Response JSON body:

```
[81,
109,
70,
122,
90,
83,
65,
50,
78,
67,
66,
84,
100,
72,
74,
108,
89,
87,
48,
61]
```

GetSupportedLanguages

Description: Returns all the Simphony configured languages for a given organization.

URL:
`https://<home>/ImportExportAPI/DataService.svc/web/GetSupportedLanguages/{USERNAME}/{ORGCODE}`

HTTP Method: GET

Response JSON body:

```
[{
        "HierarchyId":9223372036854775807,
        "Id":9223372036854775807,
        "ObjectNumber":2147483647,
        "LanguageName":[{
                "LangId":2147483647,
                "Text":"String content"
        }]
}]
```

SubmitImportExportRequest

Description: Submit a request for Import or Export.

URL:
`https://<home>/ImportExportAPI/DataService.svc/web/SubmitImportExportRequest`

HTTP Method: POST

Request JSON body:


```
{
        "DataForImport":[81,
        109,
        70,
        122,
        90,
        83,
        65,
        50,
        78,
        67,
        66,
        84,
        100,
        72,
        74,
        108,
        89,
        87,
        48,
        61],
        "DataSince":"\/Date(928167600000-0500)\/",
        "HierStrucId":9223372036854775807,
        "LanguageId":2147483647,
        "Level":0,
        "ObjectType":"String content",
        "OrgCode":"String content",
        "Origin":0,
        "OriginalRequestId":9223372036854775807,
        "RequestDate":"\/Date(928167600000-0500)\/",
        "RequestId":9223372036854775807,
        "RequestName":"String content",
        "ScheduleId":9223372036854775807,
        "SelectedFormat":0,
        "SelectedObjectMembers":["String content"],
        "SelectedOperation":0,
        "SortExpressions":[{
                "m_Item1":"String content",
                "m_Item2":"String content"
        }],
        "Status":0,
        "UserName":"String content"
}
```


Response JSON body:


```
{
        "RequestId":9223372036854775807,
        "Status":0
}
```

UpdateRequestStatus

Description: Updates the status of a request enabling it to be cancelled.

URL:
`https://<home>/ImportExportAPI/DataService.svc/web/UpdateRequestStatus`

HTTP Method: POST

Request JSON body:

```
{
        "OrgCode":"String content",
        "RequestId":2147483647,
        "RequestStatus":0,
        "UserName":"String content"
}
```

Response JSON body:

```
["String content"]
```

UpdateScheduleStatus

Description: Enables or disables a schedule.

URL:
`https://<home>/ImportExportAPI/DataService.svc/web/UpdateScheduleStatus`

HTTP Method: POST

Request JSON body:

```
{
        "Active":true,
        "DaysOfMonth":[2147483647],
        "Friday":true,
        "Id":9223372036854775807,
        "Monday":true,
        "Name":"String content",
        "OrgCode":"String content",
        "Saturday":true,
        "StartDate":"\/Date(928167600000-0500)\/",
        "Sunday":true,
        "Thursday":true,
        "TimeOfDay":"\/Date(928167600000-0500)\/",
        "Tuesday":true,
        "UserName":"String content",
        "Wednesday":true
}
```

2 HTTP Headers

In order to use the web service, the web client sends the necessary HTTP headers as part of the request.

Content-type: application/json

The MIME media type for JSON text is application/ json. The default encoding is UTF-8. The web service client shall send the content-type as application/ json. Figure 2-1 shows an example to obtain the authentication token using the Telerik Fiddler Web Debugger.

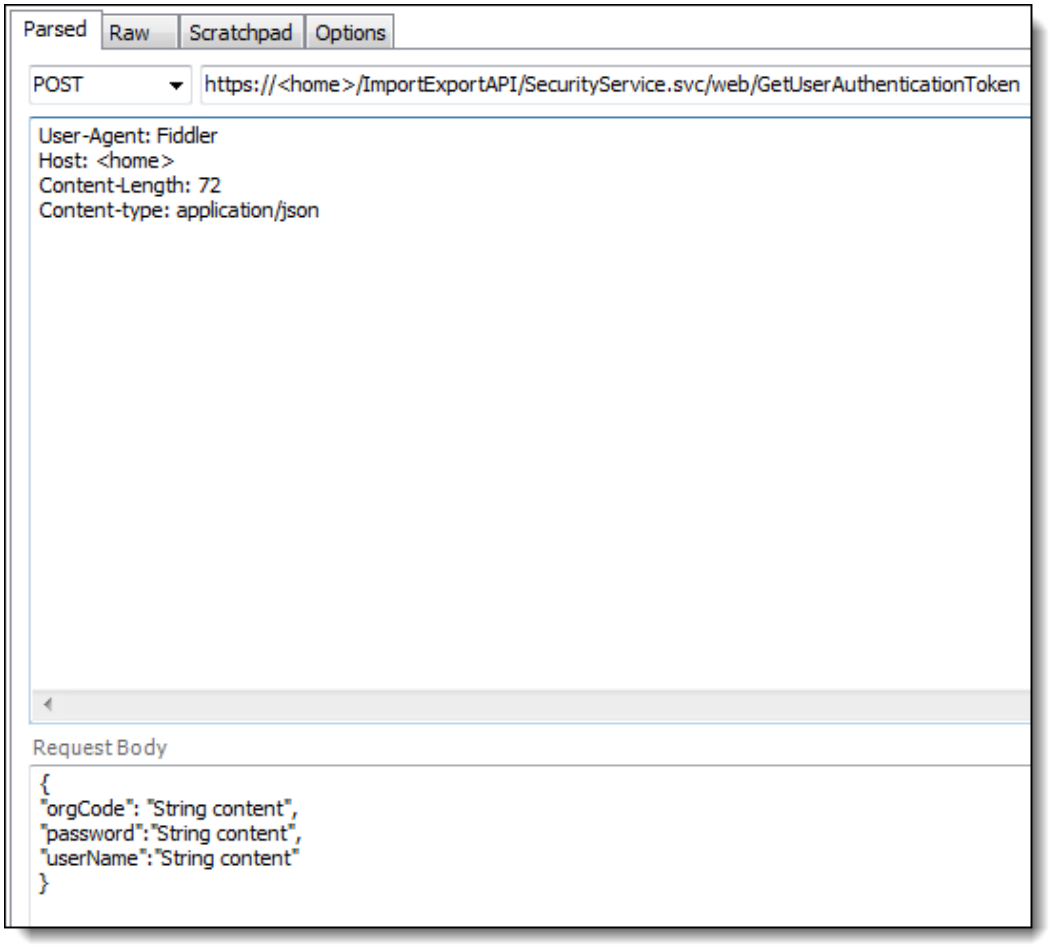

Figure 2-1 Content-type header

Authentication Token Header

All web requests include the authentication token header, which is the latest token that was returned by the get user authentication method, see Figure 2-2 for an example usage.

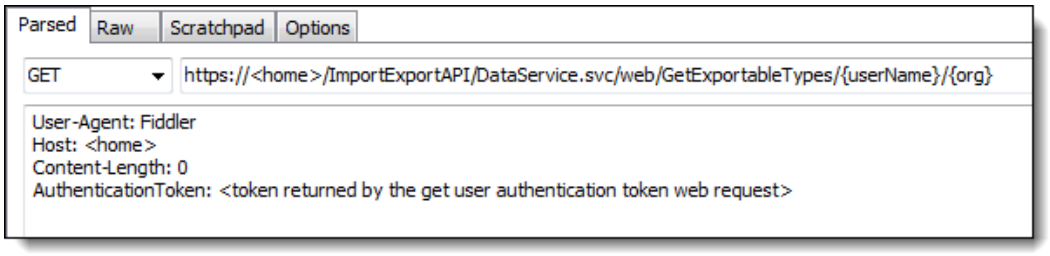

Figure 2-2 Authentication Token Example

3 Security Considerations

Security Threats and Mitigations

Since the Import/ Export Web Service API is available on the Internet, security threats should be identified and mitigated appropriately. This section discusses important threats and mitigations that are employed to protect the confidentiality, integrity, and availability of the web service. A web service client is any software program that interacts with the web service over the Internet trust boundary.

The high-level design was systematically analyzed using design analysis methods and threat modeling tools to identify the threats to the web service. As discussed here, all of these threats were mitigated with appropriate controls in place.

Interaction: Request

This section explains different types of threats and mitigation strategies for the "Request" interaction between a web service client and the Import/ Export web service.

- **XML DTD and XSLT Processing [State: Mitigation Implemented] [Priority: High]**

  Category: Tampering

  Description: If a dataflow contains XML, XML processing threats (DTD and XSLT code execution) may be exploited.

  Justification: XML DTD and Entity Processing are not enabled.

- **Elevation by Changing the Execution Flow in Import/Export Web Service [State: Mitigation Implemented] [Priority: High]**

  Category: Elevation Of Privilege

  Description: An attacker may pass data into Import/ Export Web Service in order to change the flow of program execution within Import/ Export Web Service to the attacker's choosing.

  Justification: Security testing/ analysis has been performed. It will be difficult to change the flow of program execution.

- **Import/Export Web Service May be Subject to Elevation of Privilege Using Remote Code Execution [State: Mitigation Implemented] [Priority: High]**

  Category: Elevation Of Privilege

  Description: Web Service Client may be able to remotely execute code for Import/ Export Web Service.

  Justification: Security testing/ analysis has been performed. It will be difficult to perform remote code execution.

- **Elevation Using Impersonation [State: Mitigation Implemented] [Priority: High]**

  Category: Elevation Of Privilege

  Description: Import/ Export Web Service may be able to impersonate the context of Web Service Client in order to gain additional privilege.

  Justification: Import/ Export Web Service design does not escalate privilege.

- **Data Flow Request Is Potentially Interrupted [State: Mitigation Implemented] [Priority: High]**

  Category: Denial Of Service

  Description: An external agent interrupts data flowing across a trust boundary in either direction.

  Justification: TLS protocol is used for this purpose.

- **Potential Process Crash or Stop for Import/Export Web Service [State: Mitigation Implemented] [Priority: High]**

  Category: Denial Of Service

  Description: Import/ Export Web Service crashes, halts, stops or runs slowly; in all cases violating an availability metric.

  Justification: Extensive testing and analysis has been performed. The design follows well-established patterns.

- **Potential Data Repudiation by Import/Export Web Service [State: Mitigation Implemented] [Priority: High]**

  Category: Repudiation

  Description: Import/ Export Web Service claims that it did not receive data from a source outside the trust boundary. Consider using logging or auditing to record the source, time, and summary of the received data.

  Justification: Logging and auditing is in place.

- **Web Service Client Process Memory Tampered [State: Not Applicable] [Priority: High]**

  Category: Tampering

  Description: If Web Service Client is given access to memory, such as shared memory or pointers, or is given the ability to control what Import/ Export Web Service executes (for example, passing back a function pointer.), then Web Service Client can tamper with Import/ Export Web Service. Consider if the function could work with less access to memory, such as passing data rather than pointers. Copy in data provided, and then validate it.

  Justification: Import/ Export Web Service is agnostic to the web service client.

- **JavaScript Object Notation Processing [State: Mitigation Implemented] [Priority: High]**

  Category: Tampering

  Description: If a dataflow contains JSON, JSON processing and hijacking threats may be exploited.

  Justification: Security testing/ analysis has been performed. It will be difficult to exploit JSON processing vulnerabilities.

  Interaction: Response

This section explains different types of threats and mitigation strategies for the "Response" interaction between the Import/ Export web service and a web service client.

- **Elevation by Changing the Execution Flow in Web Service Client [State: Mitigation Implemented] [Priority: High]**

  Category: Elevation Of Privilege

  Description: An attacker may pass data into Web Service Client in order to change the flow of program execution within Web Service Client to the attacker's choosing.

  Justification: We recommend that the web service client validates all input data.

- **Web Service Client May be Subject to Elevation of Privilege Using Remote Code Execution [State: Mitigation Implemented] [Priority: High]**

  Category: Elevation Of Privilege

  Description: Import/ Export Web Service may be able to remotely execute code for Web Service Client.

  Justification: Security testing/ analysis has been performed. It will be difficult to perform remote code execution.

- **Elevation Using Impersonation [State: Not Applicable] [Priority: High]**

  Category: Elevation Of Privilege

  Description: Web Service Client may be able to impersonate the context of Import/ Export Web Service in order to gain additional privilege.

  Justification: Import/ Export Web Service design does not escalate privilege.

- **Data Flow Response Is Potentially Interrupted [State: Mitigation Implemented] [Priority: High]**

  Category: Denial Of Service

  Description: An external agent interrupts data flowing across a trust boundary in either direction.

  Justification: TLS protocol is used for this purpose.

- **Potential Process Crash or Stop for Web Service Client [State: Mitigation Implemented] [Priority: High]**

  Category: Denial Of Service

  Description: Web Service Client crashes, halts, stops or runs slowly; in all cases violating an availability metric.

  Justification: We recommend that the web service client is analyzed and tested for programming/ design errors.

- **Weak Authentication Scheme [State: Not Applicable] [Priority: High]**

  Category: Information Disclosure

  Description: Custom authentication schemes are susceptible to common weaknesses such as weak credential change management, credential equivalence, easily guessable credentials, null credentials, downgrade authentication or a weak credential change management system. Consider the impact and potential mitigations for your custom authentication scheme.

  Justification: We do not have custom authentication schemes.

- **Potential Data Repudiation by Web Service Client [State: Mitigation Implemented] [Priority: High]**

  Category: Repudiation

  Description: Web Service Client claims that it did not receive data from a source outside the trust boundary. Consider using logging or auditing to record the source, time, and summary of the received data.

  Justification: We recommend that the web service client implements logging or auditing.

- **Collision Attacks [State: Mitigation Implemented] [Priority: High]**

  Category: Tampering

  Description: Attackers who can send a series of packets or messages may be able to overlap data. For example, packet 1 may be 100 bytes starting at offset 0. Packet 2 may be 100 bytes starting at offset 25. Packet 2 will overwrite 75 bytes of packet 1. Ensure you reassemble data before filtering it, and ensure you explicitly handle these sorts of cases.

  Justification: The application-layer does not have any communication protocols. TLS takes care of replay attacks.

- **Replay Attacks [State: Mitigation Implemented] [Priority: High]**

  Category: Tampering

  Description: Packets or messages without sequence numbers or timestamps can be captured and replayed in a wide variety of ways. Implement or utilize an existing communication protocol that supports anti-replay techniques (investigate sequence numbers before timers) and strong integrity.

  Justification: The application-layer does not have any communication protocols. TLS takes care of replay attacks.

- **Import/Export Web Service Process Memory Tampered [State: Not Applicable] [Priority: High]**

  Category: Tampering

  Description: If Import/ Export Web Service is given access to memory, such as shared memory or pointers, or is given the ability to control what Web Service Client executes (for example, passing back a function pointer.), then Import/ Export Web Service can tamper with Web Service Client. Consider if the function could work with less access to memory, such as passing data rather than pointers. Copy in data provided, and then validate it.

  Justification: The client and the Import/ Export Web Service are two different (remote) processes. The design does not deal with low-level memory, pointers, etc.

- **XML DTD and XSLT Processing [State: Mitigation Implemented] [Priority: High]**

  Category: Tampering

  Description: If a dataflow contains XML, XML processing threats (DTD and XSLT code execution) may be exploited.

  Justification: We recommend that the web service client disables XML DTD and External Entity Expansion.

- **JavaScript Object Notation Processing [State: Mitigation Implemented] [Priority: High]**

  Category: Tampering

  Description: If a dataflow contains JSON, JSON processing and hijacking threats may be exploited.

  Justification: Output encoding is followed in the design of the Import/ Export Web Service.

Secure Usage of Import/Export Web Service API

This section recommends a few important secure coding practices for the developers of the web service client, in order to securely use the Import/ Export Web Service.

- Do not store your Simphony credentials in your web client source code or any other configuration files. Ideally, the web client shall ask the user to type in the password. However, if the user cannot be in the loop, Oracle Hospitality recommends storing the credentials in secure storage, such as DPAPI, Crypt API, or any other credential/ key management system.
- Do not log your Simphony password or authentication tokens returned by the Import/ Export web service. Although the authentication tokens are short-lived, if attackers (including insiders) can access the log file, they will be able to impersonate you and run other important API methods.
- Make sure your web service client catches exceptions and log them securely, instead of showing the stack trace to users of your web service client. Attackers can leverage the leaked information to construct further attacks.
- Check the input length and type of all data received from the users of third-parties of your web service client. For example, before exporting a CSV file to the web service, it is recommended that the size of the uploaded file is in an acceptable range, to avoid denial of service.
- If your web service client saves the CSV files that are uploaded by your users, you must make sure to run virus/ malware scanners in order to detect whether malicious files are uploaded or not.
- If your web service client is publicly accessible, ensure only authenticated users can use your service.
- Check the return code (a.k.a. error code) of the web service methods. Ensure the actual return code matches the expected return code. For example, if a web service method returns 200, then it denotes the operation was successful. It is a good practice to log failures, for example, when the return code is 404 (a.k.a. page not found).
- The web service client should disable Document Type Definition processing as well as External Entity Expansion to avoid exploits such as remote code execution, server-side forgery attacks, denial-of-service, to name a few attacks.
- The web service client should check the digital certificate presented by the Import/ Export web service in order to ensure that the client is interacting with the trustable web service.

4 Sample API Usage Application

To help you use the Import/ Export Web Service, a sample application has been developed. This sample application is written in the C# language. You may download the sample app from the Import/ Export web site if you have valid Simphony credentials.

Resources available for download

`https://<home>/ImportExportApp/Help` contains the link to download the sample application archive, under the "Web API Downloads" section.

`http://<home>/ImportExportAPI/html/index.html` contains the Data Contract specification for each API method of the Import/ Export web service.

Brief summary of the sample application

The main entry point to the sample application is the WebAPIExample.cs, which contains the demo usages of different APIs of the Import/ Export web service. In order to compile the sample app, you would need to install the Newtonsoft JSON C# library, 9.0.1 is the recommended version. Json.NET is a high-performance JSON framework for .NET. We refer the reader to download the same application to understand how to use the web service programmatically from a client.