Oracle Utilities Testing Accelerator

Security Guide

Release 7.0.0.0

F59688-01

August 2022

ORACLE

Oracle Utilities Testing Accelerator Security Guide, Release 7.0.0.0

F59688-01

Contents

Preface

Welcome to the Oracle Utilities Testing Accelerator Security Guide. This guide describes how to configure security for Oracle Utilities Testing Accelerator using the default features.

The preface includes:

Audience

This guide is intended for product administrators, security administrators, application developers, and others tasked with performing the following operations securely and efficiently:

- Designing and implementing security policies to protect the data of an organization, users, and applications from accidental, inappropriate, or unauthorized actions
- Creating and enforcing policies and practices of auditing and accountability for inappropriate or unauthorized actions
- Creating, maintaining, and terminating user accounts, passwords, roles, and privileges
- Developing interfaces that provide desired services securely in a variety of computational models, leveraging product and directory services to maximize both efficiency and ease of use

Prerequisite Knowledge

You must have a basic understanding of how the product works, and familiarity with the security aspects of the web applications.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle's Accessibility Program website.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. If you are hearing impaired, visit the Oracle Accessibility Learning and Support website for more information.

Related Documents

For more information, refer to the following Oracle resources.

Release Notes

- Oracle Utilities Testing Accelerator Release Notes

Installation and Administration Guide

- Oracle Utilities Testing Accelerator Installation and Administration Guide

User and Reference Guides

- Oracle Utilities Testing Accelerator Security Guide
- Oracle Utilities Testing Accelerator User's Guide
- Oracle Utilities Testing Accelerator Upgrade Guide
- Oracle Utilities Testing Accelerator Licensing Information User Manual

Additional Documents

For more security-related information, refer to the following Oracle resources.

- Oracle Utilities Testing Accelerator User Guide
- Oracle Utilities Application Framework Advanced Security (Doc ID: 1375615.1)
- Technical Best Practices for Oracle Utilities Application Framework Based Products (Doc ID: 560367.1)
- Oracle Identity Management Suite Integration with Oracle Utilities Application Framework Based Products (Doc ID: 1375600.1)
- Web Services Best Practices (Doc ID: 2214375.1)

These documents are available on My Oracle Support and/or Oracle Software Delivery Cloud.

Abbreviations

The following terms are used in this document:

Term	Expanded Form
OUTA/UTA	Oracle Utilities Testing Accelerator

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Note:

- Screen images in this document are for illustrative purposes only.
- Menu options assume the use of Alphabetic sorting. If alternatives are used, adjust the advice accordingly.

Critical Patches

Oracle recommends that customers get all their security vulnerability information from *Oracle Critical Patch Update Advisories* available from links at Critical Patches, Security Alerts and Bulletins. It is strongly recommended that all critical patches be applied in a timely manner.

Additional details are available on Oracle Software Security Assurance website.

Chapter 1

Introducing Security

One of the key aspects of the product is security which not only confirms the identity of an individual user, but also confirms the data and functions that users can access within the product.

Security Features

Security is one of the key features of the product architecture protecting access to the product, its functionality and the underlying data stored and managed via the product.

From an architecture point of view, the following summarizes the approach to security:

- **Web Based Authentication**: The product provides a default method, using a traditional challenge and response mechanism, to authenticate users.
- **Support for Java EE Web Application Server Security**: The supported Java EE Web Application Servers can integrate into several internal and external security stores to provide authentication services. The product can use those configurations, to liaise via Apache Tomcat, to authenticate users for online and Web Services based security.
- **Operating System Security**: For non-online and non-web service-based channels, the product utilizes the operating system security (including any additional products used to enhance the base operating system security).
- **Non-Cookie Based Security**: After authentication, the user's credentials form part of each transaction call to correctly identify the user to the internal authorization model to ensure the user is only performing permitted actions. This support is not browser cookie based.
- **Secure Transport Support**: Transmission of data across the network can utilize the secure encryption methods supported for the infrastructure.
- **Inter-component Security**: Calls within the product and across the tiers are subject to security controls to ensure only valid authenticated and authorized users using Java Authentication and Authorization Services (JAAS).
- **Inbuilt Authorization Model**: Once a user is authenticated, the internal authorization model is used to determine the functions and data the user has access to within the product.

- **Native Web Services Security**: The web services available from the product are natively available from Apache Tomcat. A wide range of security policies are available.
- **Keystore Support**: Keys for encryption can be externalized in JCEKS based keystore.

Chapter 2

Authentication

This chapter focuses on the product authentication information including:

- About Authentication
- Online Authentication
- Configuring Authentication for REST Services

About Authentication

From a security point of view, authentication is about identification of the user. It is the first line of defense in any security solution. It can be as simple as the challenge-response mechanism we know as userid and password. It can be also as complex as using digital certificates as the identification mechanism and numerous other schemes for user identification.

The authentication aspect of security for the product is delegated to the infrastructure used to run the product. This is due to several reasons:

- **Authentication Scheme Support**: The Java EE Web Application Server supports industry standard security repositories and authentication methods. These are native to the application server.

Online Authentication

The product authenticates the online users using native implementation via basic authentication. There are no authentication specific configurations that are required at the container level. The web application server, Apache Tomcat that hosts the utilities testing accelerator can support one or more of the following:

- **Inbuilt Security**: The Java EE Web Application Server typically supplies a default basic security store and associated security management capability that can be used if no other security repository exists.
- **DBMS Based Security**: The Java EE Web Application Server can store, manage and retrieve security information directly from a database.
- **Operating System Based Security**: The Java EE Web Application Server can store, manage and retrieve security information directly from the underlying operating system.

After the validity of the user is established, an authorized user is issued an access token that is used for transaction security. After every successful login, Spring Boot uses a native OAuth 2.0 compliant authorization rules to generate a JWT based which is saved and exchanged although the user session.

REST based webservices also require an authentication token during invocation. These tokens are generated based on user credentials by hitting the oauth specific end point exposed natively by spring boot. For more details see the Configuring Authentication for REST Services section.

Configuring Authentication for REST Services

Oracle Utilities Testing Accelerator provides the following REST services used to integrate with any compatible third-party application. Example: test management applications, Jenkins, etc.

- Prerequisites
- Flow Execution
- Flow Set Execution

- Flow Execution Analytics
- Flow Set Execution Analytics
- Flow Summary
- Flow Set Summary

All the REST services are secured by mandating an authentication token as one of the parameters. The application facilities the generation of these JWT tokens by exposing an endpoint, **/token/generate-token**. These tokens are generated using native JWT Support and once they are obtained, they need to be a part of any incoming requests to these rest services. Below is a sample command to generate an authentication token based on user credentials.

```
curl -X POST -k -F 'username=<login username>' -F 'password=<login
password>' 'https:/
/<hostname>:<port>/token/generate-token -H 'authorization: Basic
dXRhOnV0YXBhc3M= ' -H 'cache-control: no-cache' -H 'content-type:
multipart/formdata'
```

Parameters

```
<login username>: Valid OUTA application user's username
<login password>: OUTA application user's password
<hostname>: OUTA Application host name
<port>: OUTA application port
```

Sample Response

```
{"token":
"eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJsb2NhbGhvc3QiLCJzY29wZXMiOiJST0xF
X0FkbWluaXN0cmF0b3IiLCJpYXQiOjE2MjAwMzQ1NjMsImV4cCI6MTYyMDA1MjU2M3
0.2SrKiZPslj6Ba2S8zqJm8YakjaFdZfhIhfuXcEsSE-Q"}
```

Chapter 3

Authorization

Once users are identified, they must be authorized to specific functions and data within the product. This chapter provides information about authorization, including:

- About Authorization
- Authorization Model
- Flow Lifecycle
- Role-Based Access

About Authorization

Oracle Utilities Testing Accelerator uses an inbuilt security model for authorization. This model contains all the data necessary for the definition of authorizations to function and data.

For information about the security authorization model, see the Authorization Model section.

Authorization Model

Oracle Utilities Testing Accelerator uses an inbuilt security model for authorization. This model contains all the data necessary for the definition of authorizations to function and data. The authorization model is role based and the role defines the level of access an user has within the application. The role is tied to the features of the applications and restrictions are enforced with appropriate role-based checks.

The predefined roles in the application are:

- Administrator
- Approver
- Developer

Administrator is the super user with all privileges and needs to be created initially. All other user creation is done by the Administrator user. The user with Administrator role can perform the following actions:

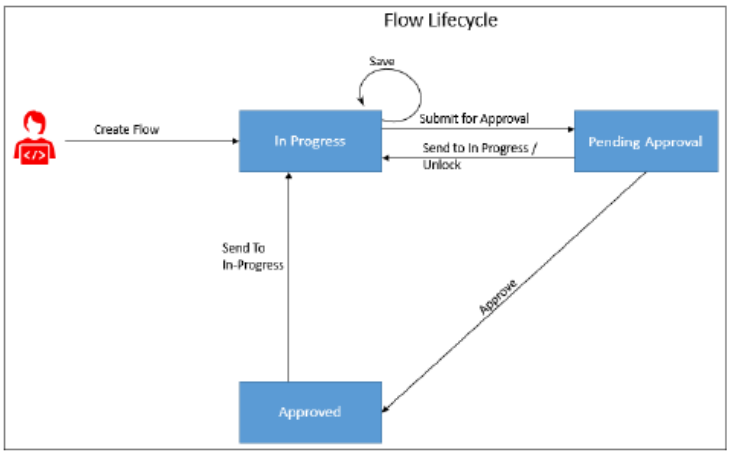
- Managing Releases
- Managing Portfolios
- Managing Products
- Managing Modules
- Managing Users
- User Access Types
- Purging Flow Execution Data

Approver is the role which can perform a subset of functionalities that the Administrator user is capable of. Major functionalities like components creation, flow creation go through an approval life cycle state. Users with Administrator/Approver roles can only approve the actions performed by the users with Developer role, which is the least privileged.

For information about the flow lifecycle and role-based access during the state transitions, see the Flow Lifecycle section.

Flow Lifecycle

The flow lifecycle begins once a flow is created in Oracle Utilities Testing Accelerator. It can exist in one of the several possible lifecycle states as shown in the following diagram.



The state of a flow determines the actions performed on the component. The following table summarizes the component states, and the possible actions and roles that can take the actions.

Flow Lifecycle State	Permitted Actions	Role	Resultant State (after action)
In Progress	Submit for Approval	Developer, Approver, Administrator	Pending Approval
Pending Approval	Send to In Progress	Developer, Approver, Administrator	In Progress
	Unlock	Developer, Approver, Administrator	In Progress
	Approve	Approver, Administrator	Approved
Approved	Send to In Progress	Developer, Approver, Administrator	In Progress

Role-Based Access

Role is tied to the functionalities of the application and restrictions are enforced with appropriate role-based checks. The table below lists the functionalities supported by the application and the role-based access privilege corresponding to them.

Access	Developer	Approver	Administrator
Approve/reject a component		✓	✓

Access	Developer	Approver	Administrator
Approve/reject a flow		✓	✓
Change Password (self)	✓	✓	✓
Component - Create/update/ view/delete	✓	✓	✓
Copy Component	✓	✓	✓
Create Module			✓
Create Portfolio			✓
Create Product			✓
Create Release			✓
Export Component	✓	✓	✓
Export Flow	✓	✓	✓
Flow - Create/update/view/ delete	✓	✓	✓
Generate a component from WSDL	✓	✓	✓
Import Component	✓	✓	✓
Import Flow	✓	✓	✓
Update Module	✓	✓	✓
Update Portfolio	✓	✓	✓
Update Release	✓	✓	✓
User Management • Create/update/view/ delete • Change other user password			✓

Chapter 4

Managing Security

This chapter focuses on managing security, including:

About Managing Security

Once the security definitions are established, they must be managed from the product itself, and the security infrastructure and security repositories used.

Managing Users

Only users with an Administrator role can manage the other users.

To create, search for, upgrade or delete a user, the user should be an Administrator with visibility to the **Administration** page to perform the tasks. The Administrator can also change the password from this page.

The Administration feature in Oracle Utilities Testing Accelerator allows the users with Administrator role to do the following:

- Create/edit release, portfolio, product and modules
- Create/manage Oracle Utilities Testing Accelerator application user accounts
- Allow upgrading CM content from one version of an Oracle Utilities application to a later version

  Example: From Oracle Utilities Customer Care and Billing V2.6.0.0.0 to Oracle Utilities Customer Care and Billing V2.6.0.1.0
- Purging old flow execution data

Configuring Authentication for Secure Outbound Web Service Requests

While connecting to the edge applications that use the HTTPS protocol, before executing the Oracle Utilities Testing Accelerator scripts, the security certificate should be saved in the system from where the Oracle Utilities Testing Accelerator test cases are executed. Register the certificate in the Java security certificates repository.

Define the following properties in the configuration file.

System Property	Comments
javax.net.ssl.keyStore	Keystore location
javax.net.ssl.keyStoreType	Default keystore type
javax.net.ssl.keyStorePassword	Default keystore password

To import the security key store into Java key store:

1. Enter the URL (HTTPS) of the application in the browser.

   Example: Internet Explorer
2. Click **Continue to this Website (not recommended)** link on the **Security Certificate** page.
3. Click **Certificate Error** in the address bar.

4. Click the **View Certificates** link on the **Certificate Invalid** pop-up window.
5. On the **Details** tab, click **Copy to File**.
6. Click **Browse** and select the file to export. Click **Next**.
7. Review the settings and click **Finish**.
8. Login to the machine where this certificate should be imported into the Java key store and open the command prompt.
9. If the Java path is not set in the environment variables, navigate to the Java/jdk/bin directory and execute the following command.

   ```
   keytool -import -alias <Alias Name> -file <path of the file
   exported in Step 7> -keystore <Java keystore path>
   ```

10. Enter "changeit" as the Password.
11. Click **Yes** to import the certificate. The property file attributes for HTTPS requests are as follows:

    ```
    ##Handling Https WSDL - Java key Store
    gStrJavaKeyStorePath=C:\\jdk8\\jre\\lib\\security\\
    gStrJavaKeyStorePwd=changeit
    ```

Based on the version of the Oracle Utilities application (and Oracle Utilities Application Framework), the web service requests are also expected to include additional information other than the user credentials. To support this, two new properties are introduced in the configuration.properties file using which users can specify the authentication used by the environment.

For the latest versions of Oracle Utilities applications, a timestamp is expected in the web service requests. For these environments, specify the header type as TIMESTAMP, the other property gStrTimeToLive specifies the validity of the request in seconds.

```
#Header Type
gStrApplicationHeaderType=TIMESTAMP
#Timestamp interval
gStrTimeToLive=120
```

In cases where the configuration.properties contains details of more than one environment, prefix the header property with the application string.

```
#Header Type
gStrUAT_gStrApplicationHeaderType=TIMESTAMP
#Timestamp interval
gStrUAT_gStrTimeToLive=120
```

For the older versions of Oracle Utilities applications, only the user credentials are expected. So specify the header as USERTOKEN.

```
#Header Type
gStrApplicationHeaderType=USERTOKEN
```

In cases where there is a mix of environments that use the new header type and old header type in the same configuration.properties file, specify the properties for individual environments as follows.

```
#Header Type
gStrUAT_gStrApplicationHeaderType=TIMESTAMP
#Timestamp interval
gStrUAT_gStrTimeToLive=120
#Header TypegStrINT_gStrApplicationHeaderType=USERTOKEN
```

Note: The user credentials are sent as digest by default. To send them as plain text, set the property mentioned needs to 'true'.

```
gStrSendPasswordAsText = true
```

Password Encryption Tool

The Password Encryption Tool is provided in the application to encrypt passwords for additional security. All the password fields (gStrApplicationUserPassword, gStrApplicationDBPassword) in the configuration.properties file and security.properties file, if any, must be encrypted. This tool helps in encrypting the passwords used.

To encrypt plain text using the Password Encryption Tool:

1. Double-click the PasswordEncryptor.jar file.
2. Enter the password and click **OK**. A dialog box with the encrypted text is displayed.
3. Copy this text and paste it in the respective password field in the configuration.properties or security.properties file.

Alternatively, perform the following steps in the Console:

1. Navigate to <ECLIPSE-WORKDIR>/tools.
2. Enter the following command:

   ```
   java -jar PasswordEncryptor.jar
   ```

3. When prompted, enter the password.
4. Press **Enter**.
5. Copy this text and paste it in the respective password field in the configuration.properties or security.properties file.

Data Masking Support

If particular data (such as passwords) within the user configuration or flow configuration objects is considered a candidate for data masking, then the masking capabilities provided along with the encryption option with the product can be used to mask the data in an appropriate fashion.

Note: The data is not stored in masked fashion; it is configured to be displayed in masked format for users using the Security Types.