# Oracle® Service Architecture Leveraging Tuxedo (SALT)

## Release Notes

Release 22c (22.1.0.0.0)
F70809-04
August 2024

Oracle Service Architecture Leveraging Tuxedo (SALT) Release Notes, Release 22c (22.1.0.0.0)

F70809-04



Primary Authors: Priya Pathak, Tulika Das

Contributors: Maggie Li

# Contents

# Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

**Access to Oracle Support**

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

1

# Oracle SALT 22c (22.1.0.0.0) New Features

This release note contains new features and enhancements incorporated in Oracle SALT Release 22c (22.1.0.0.0).

- Integrate with Oracle Transaction Manager for Microservices
  In this release, SALT has been enhanced to allow Oracle Tuxedo services to participate in global transactions coordinated by Oracle Transaction Manager for Microservices.
- Support JWT Token Authentication
- Updates to SALT Security
  This release promotes a more secure environment by default. This section describes the default security behavior and the environment variables needed for backward compatibility.

## 1.1 Integrate with Oracle Transaction Manager for Microservices

In this release, SALT has been enhanced to allow Oracle Tuxedo services to participate in global transactions coordinated by Oracle Transaction Manager for Microservices.

Oracle Transaction Manager for Microservices enables enterprise users to adopt and increase use of microservices architecture for mission-critical applications by providing capabilities that make it easier to develop, deploy, and maintain such applications. You do not need any additional configuration to integrate SALT with the Oracle Transaction Manager for Microservices.

> **See Also:**
>
> *Oracle® Transaction Manager for Microservices Developer Guide*

## 1.2 Support JWT Token Authentication

- **JWT Token Authentication**
  You can use the new **TrustedIdpCert** element for JWT token authentication. Add this element in the block of Certificate. Next, set the value of this element to the name of the file that contains a list of PEM formats of X509 certificates.

  When GWWS receives a REST inbound request, it checks the HTTP header. If there is an Authorization: Bearer header present, then GWWS assumes that the request uses the JWT bearer token. GWWS does the credential mapping if the JWT token is valid. By default, GWWS uses the sub claim in JWT as the Tuxedo username.

  > **Note:**
  >
  > If you set the Tuxedo SECURITY to `NONE` in the UBBCONFIG file, then GWWS does not validate the JWT token. In this case, the Authorization header is ignored.

**Example**

```
<System>
        <Certificate>
            <TrustedIdpCert>idp.pem</TrustedIdpCert>
        </Certificate>
    </System>
```

- **JWT Authentication supports the following JWT signing algorithms:**
  - RS256: RSA Signature with SHA-256
  - RS384: RSA Signature with SHA-384
  - RS512: RSA Signature with SHA-512
  - ES256: ECDSA Signature with SHA-256
  - ES384: ECDSA Signature with SHA-384
  - ES512: ECDSA Signature with SHA-512

# 1.3 Updates to SALT Security

This release promotes a more secure environment by default. This section describes the default security behavior and the environment variables needed for backward compatibility.

- *TM_MIN_PUB_KEY_LENGTH:* When you use HTTPS, for RSA, the minimum key length is 2048. When you load the key/certificate, GWWS detects the key length. If the key length is smaller than 2048, it will fail to boot. In case you want to use a shorter key length, then use the environment variable *TM_MIN_PUB_KEY_LENGTH*.
- *TM_TLS_FORCE_VER*: TLS 1.2 is used by default. To use a different version of TLS (for SSL servers), use the environment variable *TM_TLS_FORCE_VER* to specify the forced TLS version.
- Following is the list of cipher suites supported by default:
  - TLS_RSA_WITH_AES_256_CBC_SHA256
  - TLS_RSA_WITH_AES_256_GCM_SHA384
  - TLS_RSA_WITH_AES_128_CBC_SHA256
  - TLS_RSA_WITH_AES_128_GCM_SHA256
    In case you want to use other cipher suites, use the environment variable *TM_CIPHERSUITES* to explicitly specify the ciphers. For example, to interoperate with old versions of Oracle Tuxedo.

2

# Upgrade Considerations

If a previous SALT release is installed, then you must uninstall it before installing Oracle SALT 22c (22.1.0.0.0). This is because the current installation of the software cannot co-exist with a previous SALT installation if the Oracle Tuxedo installation is the same.

> **See Also:**
>
> *Upgrading from Tuxedo Previous Releases to Tuxedo 22c Release*

# 3
# SALT Platform Support

Oracle SALT 22c (22.1.0.0.0) supports the following platforms:

- Oracle Linux 7 (64-bit) on x86-64
- Oracle Linux 8 (64-bit) on x86-64
- Red Hat Enterprise Linux 7 (64-bit) on x86-64
- Red Hat Enterprise Linux 8 (64-bit) on x86-64
- SUSE Linux Enterprise Server 12.5 (64-bit) on x86_64

4

# Interoperability Considerations

This section describes interoperability of Oracle SALT 22c (22.1.0.0.0) with internal and external products.

Oracle SALT 22c (22.1.0.0.0) certifies the interoperability with WLS 14.1.1. SALT supports most external web service applications that are described using WS-TX standard documents. For more information, see *WS-TX Interoperability Table*

For products that are certified with previous releases, see *Certified Interoperable Web Service Server Toolkits*

5

# Major Enhancements Post Oracle SALT Release 12.2.2

The following section describes the major enhancements made to Oracle SALT 22c (22.1.0.0.0) post the Release 12.2.2.

| BugDB Number | Description |
|---|---|
| Bug 29697086 | Support long REQUEST-URI up to 8192 characters |
| Bug 29123988 | GWWS SSL handshake supports multi-threading. A new variable `GWWS_SSL_HANDSHAKE_MULTITHREADING` has been added. If the value of this variable is set to `Y`, GWWS leverages multiple threads to handle the SSL handshake. |
| Bug 28694696 | WSDL supports `soapenc:string` |
| Bug 28190520 | Allows empty RECORD buffer |
| Bug 28079925 | Outbound support `XSL:NIL` |
| Bug 27768540 | **HTTP response code enhancements**<br>A new environment variable `GWWS_HTTP_RESP_FIELD` has been introduced. Following is the format of GWWS_HTTP_RESP_FIELD:<br>GWWS_HTTP_RESP_FIELD=`FML32_FIELD_NAME[:keep]`<br>**Example**<br>• export GWWS_HTTP_RESP_FIELD=`Test_Resp_Code`<br>• export GWWS_HTTP_RESP_FIELD=`Test_Resp_Code:keep`<br>**Note:**<br>If the replied error buffer contains a `Test_Resp_Code` field, then the HTTP status code is applied to the value of `Test_Resp_Code`. Further, the default behavior removes the `Test_Resp_Code` from the error buffer and it is not converted to a JSON message. However, if you want it to be a part of JSON data, you must specify:`keep`. In addition, `Test_Resp_Code` must be present at the top level of the replied error buffer, else it does not take effect in an embedded FML32 buffer. |
| Bug 27608108 | **Inbound service supports the following two attributes**:<br>• `jsonTopLevelArray`: If the value of this attribute is set to `true` and the top level data of the inbound response is an array, then GWWS returns a no-name json array at the top level. The default value is false; `Required:No`<br>• `enableRplyBuffer`: If the value of this attribute is set to `true`, then GWWS converts the TPFAIL error buffer to a json message. The default value is false; `Required:No` |

| BugDB Number | Description |
|---|---|
| Bug 25735361 | **Inbound service supports default content-type**<br><br>`content-type`: A new attribute has been added to the Inbound->HTTP->Service element in the SALT deployment file. It is the default value for the Inbound request `content type`.<br>If the Inbound request HTTP header contains the `content-type` field, SALT uses the value in HTTP header. Otherwise, SALT uses the default `content-type` attribute set in the Inbound->HTTP->Service element. |
| Bug 25342939 | **Outbound service supports override URI**<br><br>In order to specify the override URI, the Oracle Tuxedo client can add system FML32 field `TA_HTTP_CONNECT_URI` to META_TCM of the request data by invoking `tpsetcallinfo()`. |
| Bug 24818790 | **Support whitespace collapse for `xsd:string`**<br><br>A new parameter level keyword `whitespace` has been added to Oracle Tuxedo Metadata Repository. When converting the Oracle Tuxedo buffer to XML data mapping, if the above keyword is set to `collapse`, then the white space collapses. |
| Bug 24395972 | **Inbound service supports specifying namespace for SOAP responses**<br><br>You can specify `namespace` at three different levels:<br>• **service level**: specifying `namespace` attribute in WSDF<br>`<Service name="operation_1" soapAction="op1" namespace="op_ns" />`<br>• **out buffer level**: `outbufschema` parameter in the MIF file<br>`XSD_E:<element_local_name>@namespaceURI`<br>• **parameter level**: `paramschema` parameter in the MIF file<br>`XSD_E:<element_local_name>@namespaceURI`<br>The element `namespace` uses the value defined in the nearest scope. If none of above is specified, then the element uses `urn:pack.<WSDF-name>_typedef.salt11` as before.<br>If you specify the service level `namespace` or `outbufschema`, then all the elements under `<servicenameResponse>` or`<outbuf>` inherit it. However if the `paramschema` is specified for the matched parameter, then the inbound `namespace` works only for the `document/literal` mode SOAP operation; it does not work for the `rpc/encoded` mode.<br><br>**Note:**<br>`element_local_name` in `outbufschema` and `paramschema` is not used. |
| Bug 23742025 | **Metadata Repository supports `jsonarray` for SALT**<br><br>The Metadata Repository supports a new parameter level keyword `jsonarray`. If this keyword is set to `Y`, then GWWS maps the Oracle Tuxedo buffer to JSON array type, even if there is only one occurrence of a field. |