



This PDF is generated from authoritative online content, and is provided for convenience only. This PDF cannot be used for legal purposes. For authoritative understanding of what is and is not supported, always use the online content. To copy code samples, always use the online content.

Web Services and Applications Deployment Guide

Configuring GWS API Service Settings

Configuring GWS API Service Settings

Contents

- **1 Configuring GWS API Service Settings**
 - 1.1 Logging settings
 - 1.2 Jetty settings
 - 1.3 Optional: Postgres/MSSQL Cluster settings
 - 1.4 Redis settings
 - 1.5 Platform settings
 - 1.6 Server settings
 - 1.7 Partitioned cookie settings
 - 1.8 On-premises settings
 - 1.9 Tuning the Web Services host performance
 - 1.10 SameSite cookies
 - 1.11 Configuring Statistics indexing
 - 1.12 Configuring Configuration Objects indexing
 - 1.13 Next step

As part of [Deploying the web application](#), you created the **application.yaml** file (or Web Services created it for you). To configure basic Web Services and Applications settings, you need to update the **application.yaml** file on each of your Web Services nodes. In later topics, you'll learn more about modifying this file to configure additional [features](#) and [security](#). For now, review the contents below for details about each section in the **application.yaml** configuration file.

Logging settings

The purpose of the **logging** section is to tell Web Services where to find the **logback.xml** file you created (or Web Services created for you) as part of [Deploying the web application](#) and where to save logs.

The **application.yaml.sample** file includes the following default **logging** section:

```
logging:
  file: [ToBeChanged: Log file name, will be stored in ${LOG_FILE} which may be used in
logback.xml]
  path: [ToBeChanged: Log folder, will be stored in ${LOG_PATH} which may be used in
logback.xml]
```

See [logging](#) for details about all supported configuration settings for this section.

Jetty settings

You use the **jetty** section of the **application.yaml.sample** file to tell Web Services how Jetty should behave. The **application.yaml.sample** file includes the following default **jetty** section:

```
jetty:
  port: 8080
```

See [jetty](#) for details about supported configuration settings for this section.

Optional: Postgres/MSSQL Cluster settings

```
# DB Settings for local database
db:
  host: [ToBeChanged: DB_HostName|DB_IP_Address]
  port: [ToBeChanged: DB_Port]
  dbType: [ToBeChanged: "postgres"| "mssql"] <dbType is mandatory for MS-SQL db and optional
for Postgres, Use dbtype as "mssql" for MS-SQL db>
  dbName: [ToBeChanged: DB_NAME]
  schema: [ToBeChanged: SCHEMA_NAME]
  username: [ToBeChanged: USER_NAME]
  password: [ToBeChanged: USER_PASSWORD]

# DB Settings for connectivity to primary database (for write operations initiated from
secondary Data Center)
db-write-only:
```

Configuring GWS API Service Settings

```
host: [ToBeChanged: DB_HostName|DB_IP_Address]
port: [ToBeChanged: DB_Port]
dbType: [ToBeChanged: "postgres" | "mssql"] <dbType is mandatory for MS-SQL db and optional
for Postgres, Use dbtype as "mssql" for MS-SQL db>
dbName: [ToBeChanged: DB_NAME]
schema: [ToBeChanged: SCHEMA_NAME]
username: [ToBeChanged: USER_NAME]
password: [ToBeChanged: USER_PASSWORD]
```

Redis settings

```
# Redis Settings
spring:
  data:
    redis:
      cluster:
        nodes: host1:port1, host2:port2 [ToBeChanged: comma separated list of
REDIS_HOST:REDIS_PORT]
      ssl:
        enabled: ${REDIS.TLS:false}
        password: ${REDIS.PASSWORD:}
        username: <password>
      tls:
        trustStorePath: ${REDIS.TRUSTSTORE.PATH:}
        trustStorePassword: ${REDIS.TRUSTSTORE.PASSWORD:}
        verifyPeer: ${REDIS.VERIFY.PEER:true}
```

or set environment variables: REDIS_TLS REDIS_PASSWORD REDIS_TRUSTSTORE_PATH
REDIS_TRUSTSTORE_PASSWORD REDIS_VERIFY_PEER

Platform settings

```
# Platform Settings
platformSettings:
  platformServiceUrl: http://localhost:8092[ToBeChanged:GWS_SERVICE_PLATFORM_ADDRESS]
```

8092 is the default port but could also be Load Balancer, if used.

Server settings

This section provides the core settings Web Services needs to run your node.

The **application.yaml.sample** file includes the following default **serverSettings** section:

```
serverSettings:
  # OPS account
  opsUserName: [ToBeChanged: <OPS_USER_NAME>]
  opsUserPassword: [ToBeChanged: <OPS_USER_PASSWORD>]
  # Configuration Server credentials
```

```
applicationName: Cloud
applicationType: CFGGenericClient
cmeUserName: [MatchFromEnvironment.yaml: username]
cmePassword: [MatchFromEnvironment.yaml: password]

# Elastic Search
enableElasticSearchIndexing: [ToBeChanged: "true"|"false"]
elasticSearchSettings:
  enableScheduledIndexVerification: false
  enableIndexVerificationAtStartup: false
  transportClient:
    nodes:
      - host: [ToBeChanged: ES_HostName|ES_IP_Address]
        port: [ToBeChanged: ES_Port]

# Multi regional supporting
nodePath: [ToBeChanged: node position in cluster, example: /<REGION>/HOST]
nodeId: [ToBeChangedOrRemoved: unique value in cluster <NODE_ID>]

# SSL and CA (optional, can be disabled or removed if not used)
caCertificate: [ToBeChangedOrRemoved: <CA_CERTIFICATE>]
jksPassword: [ToBeChangedOrRemoved: <JKS_PASSWORD>]

# SAML (optional, can be disabled or removed if not used)
samlSettings:
  encryptionKeyName: [ToBeChangedOrRemoved: <SAML_ENCRYPTION_KEY_NAME>]
  signingKeyName: [ToBeChangedOrRemoved: <SAML_SIGNING_KEY_NAME>]
  identityProviderMetadata: [ToBeChangedOrRemoved: <SAML_IDENTITY_PROVIDER_METADATA>]
  serviceProviderEntityId: [ToBeChangedOrRemoved: <SAML_SERVICE_PROVIDER_ENTITY_ID>]
  encryptionKeyPassword: [ToBeChangedOrRemoved: <SAML_ENCRYPTION_KEY_PASSWORD>]
  signingKeyPassword: [ToBeChangedOrRemoved: <SAML_SIGNING_KEY_PASSWORD>]
  tlsKeyName: [ToBeChangedOrRemoved: <SAML_TLS_KEY_NAME>]
  tlsKeyPassword: [ToBeChangedOrRemoved: <SAML_TLS_KEY_PASSWORD>]
  responseSkewTime: [ToBeChangedOrRemoved: <SAML_RESPONSE_SKEW_TIME>]
  wantAssertionSigned: [ToBeChanged: "true"|"false"]

# CORS
crossOriginSettings:
  allowedOrigins: [ToBeChangedOrRemoved: <CROSS_ALLOWED_ORIGINS>]

# Auth Settings
auth:
  secret: [Must be configured and must match secret from environment variable]

# Multimedia Disaster Recovery
drMonitoringDelay: [ToBeChangedOrRemoved: <DR_MONITORING_DELAY>]
```

Note: The secret length must be at least 32 characters.

Make sure that you update all settings marked as [ToBeChanged]. You must also do the following:

- Set the **applicationName** to the name of the application you created in [Configuring the Web Services applications](#) — for example, `WS_Node`.

See [serverSettings](#) for details about supported configuration settings for this section.

Partitioned cookie settings

If your browser support Cookies Having Independent Partitioned State (CHIPS) and if you want to opt-in to use CHIPS (adding Partitioned cookie attribute), add the following configuration in the **application.yaml** file.

```
jetty:  
  cookies:  
    partitioned: true
```

Important

This configuration is mandatory for browsers where third-party cookies are disabled but they should be shared across domains. For more details, see [How We're Protecting Your Online Privacy - The Privacy Sandbox](#)

On-premises settings

Example on-premises settings:

```
onPremiseSettings:  
  countryCode: US
```

The **application.yaml.sample** file doesn't include a default **onPremiseSettings** section, so you'll need to add it yourself.

See [onPremiseSettings](#) for details about all supported configuration settings for this section.

Tuning the Web Services host performance

Complete the steps below on each Web Services node to tune the performance of the host environment.

1. To optimize TCP/IP performance, you can run the following commands:

```
sudo sysctl -w net.core.rmem_max=16777216  
sudo sysctl -w net.core.wmem_max=16777216  
sudo sysctl -w net.ipv4.tcp_rmem="4096 87380 16777216"  
sudo sysctl -w net.ipv4.tcp_wmem="4096 16384 16777216"  
sudo sysctl -w net.core.somaxconn=4096  
sudo sysctl -w net.core.netdev_max_backlog=16384  
sudo sysctl -w net.ipv4.tcp_max_syn_backlog=8192  
sudo sysctl -w net.ipv4.tcp_syncookies=1  
sudo sysctl -w net.ipv4.tcp_congestion_control=cubic
```

2. After providing for some means of starting Jetty, determine the user or group that will start Jetty and increase the file descriptors available to that user or group by adding the following to the `/etc/security/limits.conf` file:

```
<user_name>          hard nofile          100000
<user_name>          soft nofile          100000
```

Where `<user_name>` is the name of the user or group that is starting Jetty.

SameSite cookies

To handle sameSite cookie attribute, you must configure options for both [Jetty](#) and [CometD](#).

If the value of **SameSite** is set to None, Chrome browser also checks if the Secure cookie attribute is present, and if not, then warn user.

To mitigate this issue, make the following edits in `application.yaml`:

```
...
jetty:
  ...
  cookies:
    ...
    secure: true
    sameSite: None
  ...
serverSettings:
  ...
  cometDSettings:
    ...
    cookieSecure: true
    cookieSameSite: None
```

Important

If cookies are configured to be secure, the browser applies them to a secure connection only (https); therefore, these options take effect only if **enableSsl** is set to true.

If the value of **SameSite** is set to Lax or Strict, a secured connection is not required, for example:

```
...
jetty:
  ...
  cookies:
    ...
    httpOnly: true
    secure: false
    sameSite: Lax
  ...
serverSettings:
```

```
...
cometDSettings:
  ...
  cookieHttpOnly: true
  cookieSecure: false
  cookieSameSite: Lax
```

However, it is important to note the following:

- If **SameSite** is set to Lax, the cookie is sent only on same-site requests or by top-level navigation with a safe HTTP method. That is, it will not be sent with cross-domain POST requests or when loading the site in a cross-origin frame, but it will be sent when the user navigates to the site via a standard top-level `` link.
- If **SameSite** is set to Strict, the cookie is never sent in cross-site requests. Even if the user clicks a top-level link on a third-party domain to your site, the browser refuses to send the cookie.

Important

You can choose an insecure connection by specifying a different type of SameSite (Lax or Strict), but this means that it will be impossible to embed Workspace Web Edition in an iframe or use it for any other cross-domain integrations. For example, applications like Genesys CRM Workspace/Adapter will not work with this configuration.

Configuring Statistics indexing

```
Add nodeId: /<REGION>/HOST
enableElasticSearchIndexing: true
```

Configuring Configuration Objects indexing

```
enableElasticSearchIndexing: true
enableScheduledIndexVerification: true
enableIndexVerificationAtStartUp: true
```

Next step

- [Configuring features](#)