

License Manager User Guide

MN-003460-03EN Rev.A

Zebra Technologies Corporation



ZEBRA

Table of Contents

Introduction.....	2
Prerequisites	2
Guide to Use Cases.....	2
Familiarize ZLicenseMgr Application	3
Badge ID Configuration	4
License Activation.....	5
Activating a License	5
Displaying Active Licenses	7
Refreshing a License.....	7
Returning a License	9
Settings.....	10
About: The About screen displays:	10
Badge ID Configuration: Users have the flexibility to edit the Badge ID if changes are required.	11
Proxy Configuration for BADGEID Activation	11
License Activation with Zebra's StageNow Tool	12
StageNow Activation via New License Manager	13
License Activation via Zebra ZDNA MyCollection	14
License Management with Third-Party EMMs	16
License Activation from SOTI using ZLicenseMgr as enterprise application	17
License Activation with EMMs Supporting File Sync	18
License Activation Using SOTI FileSync	18
License Activation and Return via Configuration File Deployment	18
License Activation and Return via Zebra OEMConfig Tools	19
License Activation and Return via 42 Gear Tools.....	19
Additional Notes and Usage Guidelines.....	22
Configuration Samples	22
XML SAMPLE :1	22
XML SAMPLE :2	22
XML SAMPLE :3	25
XML SAMPLE :4	26

Introduction

ZLicenseMgr is a software licensing application developed by Zebra to facilitate the efficient management and activation of software licenses for Zebra products. It allows users to activate licenses by using entitlement details provided upon purchase, ensuring a streamlined and effective licensing process.

Key Points

- **Entitlement Details:** Upon purchasing a license from Zebra, you receive entitlement details that include a unique BADGEID and the product name associated with the license.
- **Server Type:** Licenses are specified for either a Production server or a UAT server. Activation generally takes place on a Production server, which is utilized by both customers and partners.
- **Device Association:** A device can consume licenses only from the associated BADGEID. If a device is associated with a different BADGEID and a new license is activated, any previously activated license linked to the former BADGEID will be released.
- **Important Consideration:** Activating a BADGEID-based license using the ZLicenseMgr application will erase licenses that were activated with previous versions of the application, which were part of the device OS or BSPA.

Check the Zebra Support Site for the latest **License Manager** app available at

<https://www.zebra.com/us/en/support-downloads/software/mobile-computer-software/license-manager.html>

Note: In upcoming LG vehicles, the latest ZLicenseMgr will be pre-installed within the BSPA image, simplifying the process by removing the need for manual upgrades.

Prerequisites

Before using the app, ensure that:

- system clock is set appropriately to the current time zone and time.
- there is a working network connection for on-line activation of the licenses.

Guide to Use Cases

Refer to the following use cases to quickly navigate to the sections of this guide that provide detailed instructions tailored to your specific needs

- 1- *As a device operator, I need to activate licenses directly through a license manager application UI on the device (Android Lollipop to Android 14), bypassing the use of EMM or Zebra StageNow tools. This approach allows for easy and immediate activation, ensuring that necessary functionalities are available without additional management overhead.*
[Learn More](#)
- 2- *As a device operator or IT administrator, I need to activate a license on a Zebra device (A13 to A14) by using Zebra StageNow to generate and scan a configuration JS barcode. This process streamlines license application by leveraging StageNow's capabilities to encode necessary settings into a scannable format, enabling quick and accurate activation on devices running MX 14.0.*
[Learn More](#)
- 3- *As an IT administrator, I need to upgrade the License Manager APK on Zebra devices (Android 11 and above) and activate licenses using Zebra DNA Cloud. This involves downloading the necessary APK from the Zebra Support Portal, uploading it to MyCollection in Zebra DNA Cloud, and creating an AppSetup configuration profile to deploy the APK and activate licenses on managed devices.* [Learn More](#)

- 4- As an IT administrator, I need to upgrade the ZLicenseMgr application on Zebra devices (Android 7 and above) and activate licenses using managed configurations provided ZLicenseMgr from a third-party EMM like SOTI or 42Gears. This process involves downloading the application from the Zebra Support Portal, adding it to the managed apps list in the EMM, and deploying it with appropriate configurations. please refer to the following resources: [Learn More about SOTI](#) | [Learn More about 42Gears](#)
- 5- As an IT administrator, I need to ensure that the ZLicenseMgr application is upgraded on Zebra devices (A13 and above) and licenses are activated using Zebra MX XML submissions. By utilizing SOTI EMM's File Sync option, I can host the necessary APK and XML files on a server and deploy them efficiently to managed devices, ensuring consistent and updated configurations across all devices running MX 14.0. [Learn More](#)
- 6- As an IT administrator, I need to upgrade the ZLicenseMgr application and activate licenses on Zebra devices (A5 and above) with an MX version lower than 14.0. Since my EMM does not support managed apps for APK uploads, I will use Zebra StageNow to generate a license activation XML and host it on a server. Using my EMM's File Sync feature (A5 and above) or Zebra OEMConfig passthrough (A8 and above), I can transfer the XML file from the server to the device, upgrade ZLicenseMgr, and activate the license. [Learn More](#)

*Note: For the EMM use case where managed application support is not provided, StageNow XML can be used for APK upgrades and activation on devices running MX versions lower than 14.0. Sample XMLs are included in the document [XML SAMPLE :3](#)
For OEMPassthrough, additional reference: [Learn More](#)*
- 7- As an IT administrator, I need to activate licenses on Zebra devices that already have ZLicenseMgr version 14.0.16 or above installed or as part of the BSPA. Using the Zebra OEMConfig tool, I can configure and deploy the necessary settings to activate licenses directly, leveraging the existing infrastructure and application version on the devices. [Learn More](#)

Familiarize ZlicenseMgr Application

To start the app:


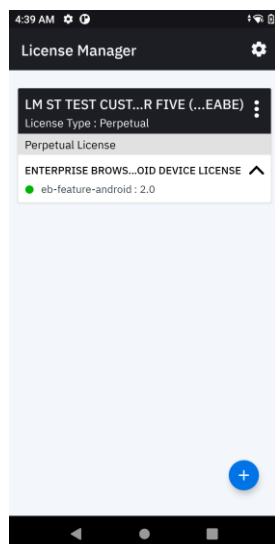
1. On the Home screen, swipe up from the bottom of the screen.
2. Touch  . The **License Manager** app appears.

Figure 1 License Manage

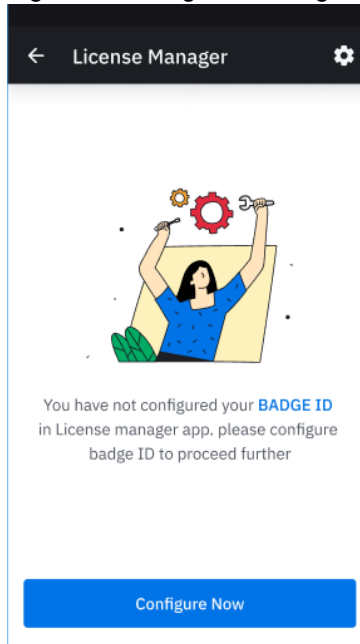


The app displays the list of licenses associated with the device.

Badge ID Configuration

Prior to any license related operation, need to configure the device with Badge ID. This innovation consolidates all licensing entitlements for multiple products under a single Badge ID, unique to each customer.

Figure 2 Badge ID Configuration



- **Badge ID** acts as a singular label for activating applications, eliminating the need for multiple activation IDs.
- Activation is completed using the Badge ID in conjunction with the product name, removing the complexity of handling multiple license keys.

To configure the Badge ID, you need to select the License Source. This determines the source from which the license associated with a software product on the device will be procured.

This involves each device reaching out to the Zebra licensing back-end directly to perform a licensing related operation like activate, return or refresh. Thales supports the following license source types:

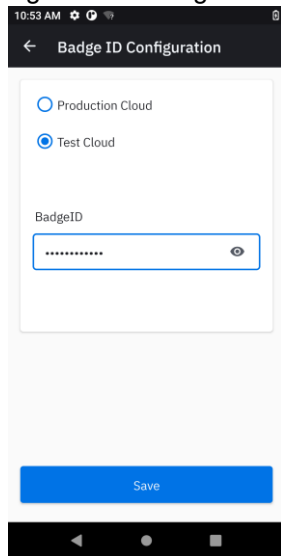
- **Production Cloud** - For directly activating the licenses from the default Zebra Licensing Server on the Cloud. This maps to a built-in URL, so the user is not required to configure anything.

URL: <https://zebratechnologies.prod.sentinelcloud.com/>

Port: 443

- **Test Cloud** - For testing purposes. Only for Zebra internal usage.

Figure 3 Configure Badge ID with Test Cloud Connectivity Options



NOTE:



- When activating a new license from another BadgeID, previously activated licenses are not returned. It is recommended to return licenses before performing FR/ER unless activated with a persist flag

License Activation

Activating a License



IMPORTANT: Ensure the device clock and network are set appropriately.

To activate a license on-line:



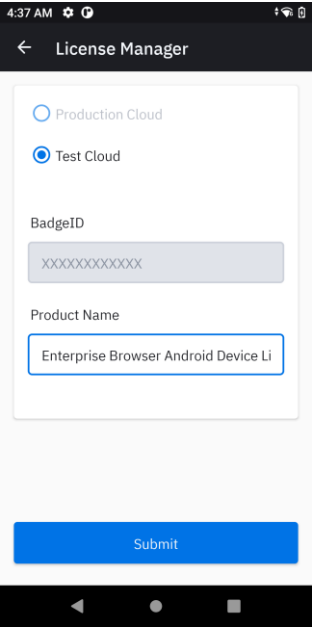
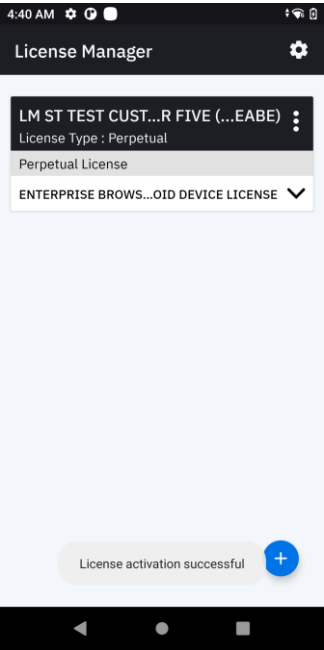
1. On the Home screen, swipe up from the bottom of the screen.
2. Touch . The **ZLicenseMgr** app appears.
3. Touch .
4. Enter the product name associated with the Badge ID

Figure 4 Activate License Screen



- 5. Touch **Submit**. The **Activate License** screen appears.
- 6. The app validates the information and connects to the Server and if successful, **License activation successful** message displays.

Figure 5 Activate Activation successful



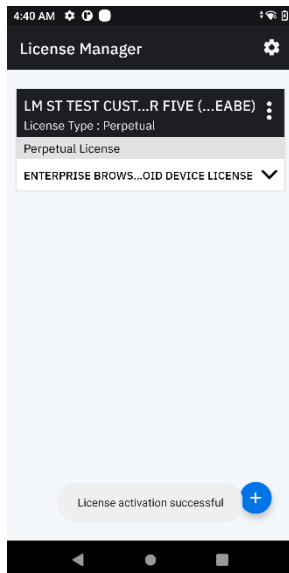
The activated license information displays on a card in the Home screen.

All the license rights displayed in the Home Screen are available for acquisition on the device.

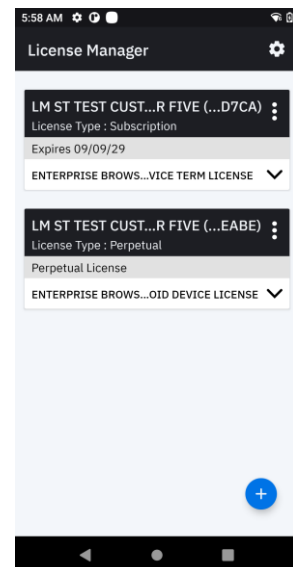
Displaying Active Licenses

All currently active licenses on the device display in the form of cards on the Home screen.

Figure 6 Home Screen



One License



Two Licenses

Each card corresponds to an active license.

Each license card displays the following information:


- The first row in the card displays the **Sold To** entity that purchased the license along with the last four latter of the Activation ID.
- The second row in the card indicates the License Type: Perpetual, Trial License, Subscription (Term), etc.
- The third row corresponds to the expiration date of the license in the format MM/DD/YYYY. License expiring within 30 days from the current date is highlighted in red.
- The fourth row in the card displays (if the license has expired) the number of days left in the grace period.

The rows below the expiration date list the products entitled on that device. The arrow next to the product name toggles a collapsible view of the list of features associated with that product.

Each license can have any number of products associated with it. Each product has a list of features listed under it.

Refreshing a License

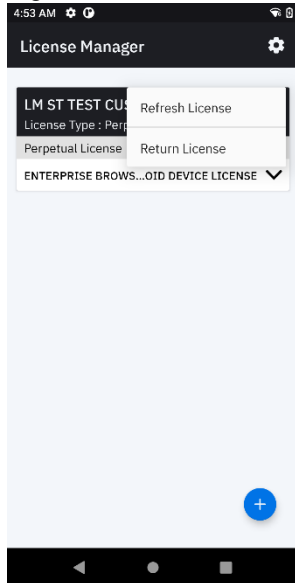
To refresh a currently active license:

1. On the Home screen, swipe up from the bottom of the screen.
2. Touch . The **License Manager** app appears.

- Once one or more licenses are currently active on the device, the user will be able to see the Home screen with a list of cards. Choose the license that needs to be refreshed and select the overflow menu at the right top corner of the license (card) which displays the below two options:

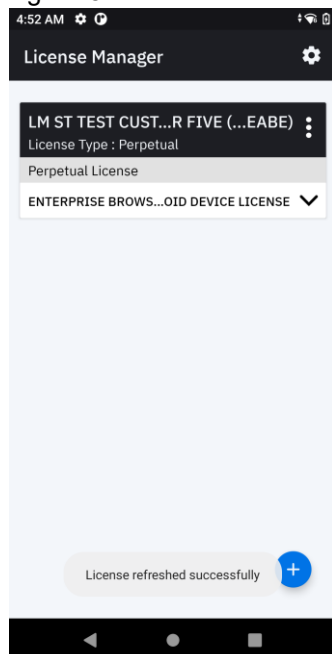
- Refresh License**
- Return License**

Figure 7 Refresh License



- Touch **:** > **Refresh License** to send a request to the server to refresh the selected license.
Once the license is successfully refreshed, **License refreshed successfully** message displays along with all the updated information in the card.

Figure 8 Refreshed License



NOTE:





- If the system date changes from a future date to the current date, licenses may only activate after a refresh from the UI.

Returning a License

Return License enables the user to return an active license from a device back to the source it was served from. Use this option if only a particular license needs to be returned or if a license source serves both restricted and non-restricted licenses to that device

To return a license which is currently active on the device:

1. On the Home screen, swipe up from the bottom of the screen.
2. Touch . The **License Manager** app appears.
3. Touch  > **Refeturn License** to send a request to the server to return the selected license.

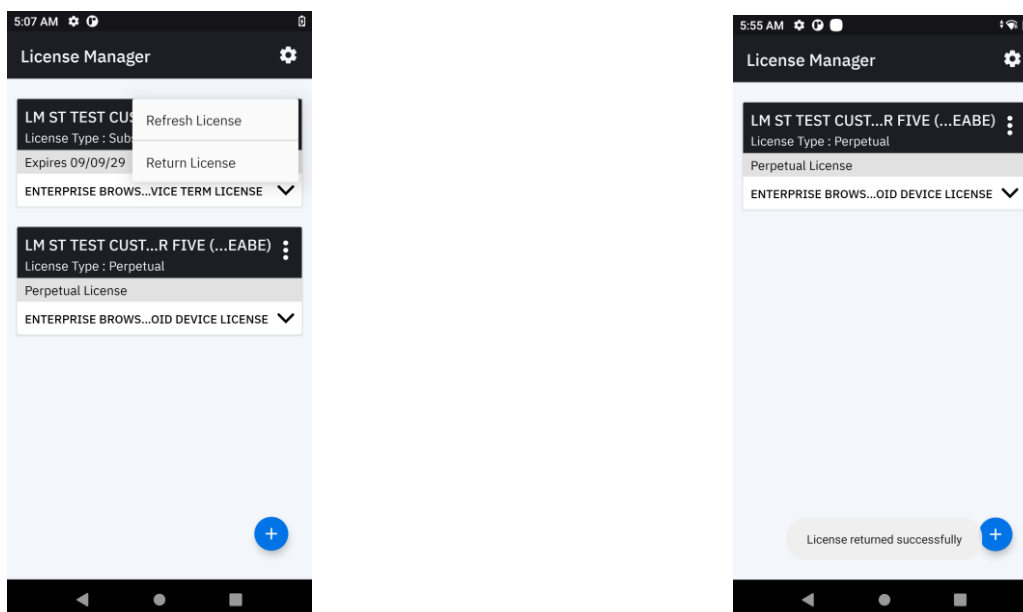
Once the license is successfully returned, **License** returned **successfully** message displays along with all the updated information in the card.

NOTE:



- Administrators are recommended to include the ReturnAllAID as the first profile before applying any Thales license activation. This is not mandatory but recommended if the intention is to maintain a single barcode for all products across devices

Figure 9 Returned License



Settings

To access the Settings screen:



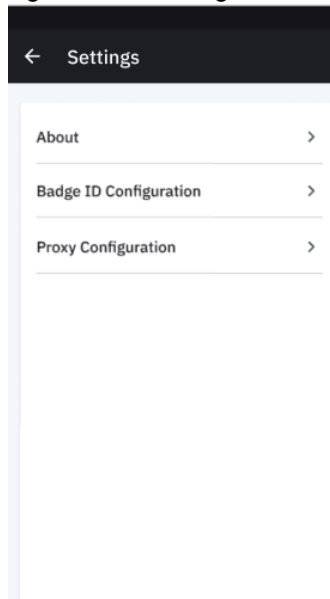
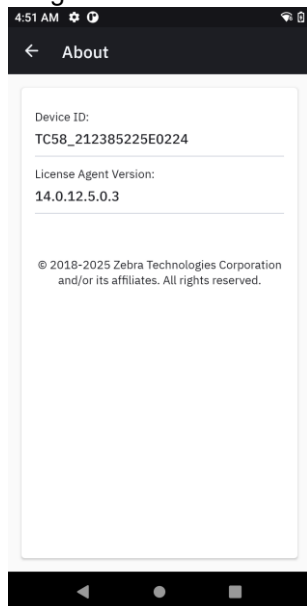
1. On the Home screen, swipe up from the bottom of the screen.
2. Touch . The **License Manager** app appears.
3. Touch .

Figure 10 Settings Screen



About: The About screen displays:

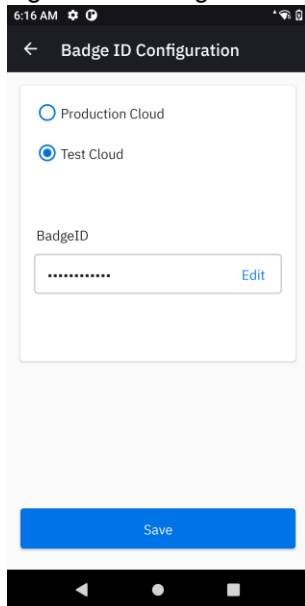
Figure 11 About Screen



- **Device ID** - Displays the unique serial number of the device.
- **License Agent Version** - Displays the License Agent version number.
- **Copyright Information** - Displays Zebra copyright information.

Badge ID Configuration: Users have the flexibility to edit the Badge ID if changes are required.

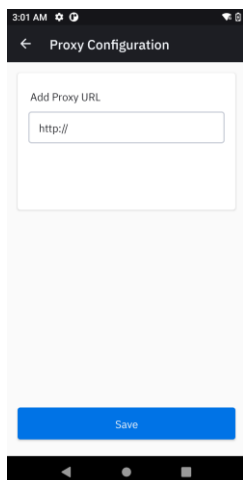
Figure 12 Badge ID Configuration



Proxy Configuration for BADGEID Activation

ZLicenseMgr supports proxy configuration at the application level to facilitate license activation for devices operating behind a network proxy. This ensures that devices can successfully activate licenses associated with their BADGEID even when direct internet access is restricted

Figure 13 Proxy Configuration



License Activation with Zebra's StageNow Tool

Zebra's StageNow Tool offers an alternative method for activating and managing licenses by generating barcodes that can be scanned by devices. This tool simplifies the process of license activation and returns, making it particularly useful for environments where manual entry might be cumbersome.

Key Features:

- **Barcode Generation:** StageNow allows you to generate a barcode that encodes the necessary information for license activation or return. When creating a profile, ensure that MX14.0 is selected from the StageNow drop-down menu to guarantee compatibility with your devices.
- **Easy Activation:** Simply scan the generated barcode with your device to activate the license. This streamlined process ensures quick and efficient license management.
- **License Return:** In addition to activation, StageNow can generate barcodes for returning licenses, facilitating flexible license management as needs change.

Note:

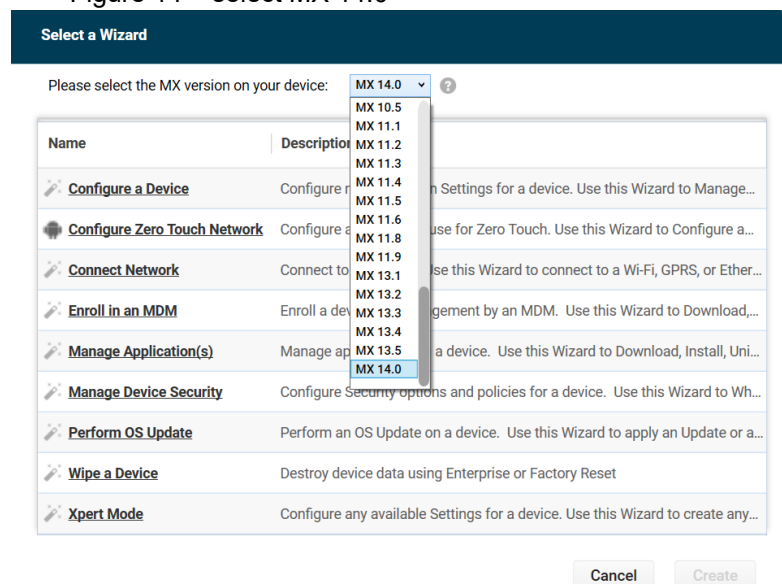
Supported SN version: 5.14.0.1000 or above

A StageNow profile can include multiple license activation commands, allowing comprehensive management of licenses. Administrators should place a "return all" command at the beginning of the profile to release all previously assigned licenses back to the server before processing new activations. This ensures that devices activate only the required licenses, maintaining consistency and preventing conflicts.

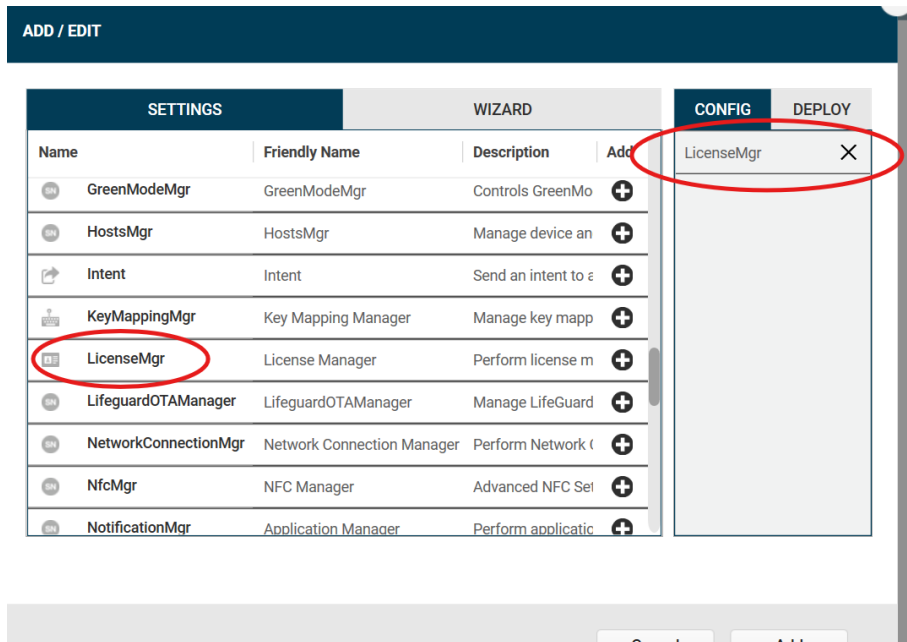
StageNow can generate barcodes from XML and JS profiles, but BADGEID licensing supports only JS-based barcodes.

When exporting StageNow XML profiles, ensure encryption is disabled, as older MX versions do not support URL encryption.

Figure 14 select MX 14.0



License Manager User Guide



The screenshot shows the configuration screen for the License Manager. It includes fields for 'License action type', 'License-server Proxy URL', 'License action', 'Cloud Server Type', 'Badge ID', 'Product Name', and 'Zebra License Persistence'. The 'License action' is set to 'Activate License using Badge ID'. The 'Cloud Server Type' is set to 'Test Cloud'. The 'Badge ID' is 'BDG-*****93U'. The 'Product Name' is 'Enterprise Browser Android Device Term License'. The 'Zebra License Persistence' is set to 'Do not change'.

StageNow Activation via New License Manager

https://www.zebra.com/content/dam/support-dam/en/documentation/unrestricted/video/0001/ZLicenseManager_SN_E2E.mp4



Note:

- BadgeID licensing are only supported from JS barcode staging (MX > 13.5). The new Zebra licensing with MX 13.5 supports only JS-based barcode scanning

License Activation via Zebra ZDNA MyCollection

Zebra ZDNA MyCollection provides an efficient approach for license activation, allowing users to manage licenses through app configuration profiles.

Process Overview:

- **Upload ZLicenseMgr:** Download ZLicenseMgr from the Zebra Support Portal and upload it to ZDNA MyCollection. This integration enables seamless license management within the MyCollection platform.
- **App Configuration Profile:** After uploading ZLicenseMgr, create an app configuration profile to activate or return licenses. This profile includes all necessary settings for the licensing process.
- **Mandatory Transaction ID Change:** Change the transaction ID for each app configuration profile pushed to a device. This change is essential for ZLicenseMgr to detect and apply profile changes, ensuring correct license activation or return.
- **Device Reboot Requirement:** A device reboot is mandatory if an app upgrade is performed. However, no reboot is required for subsequent activation or license return-related app configuration deployments.
- **Returning Licenses to Server Pool:** If an app configuration profile is pushed to a device with empty bundle data and only the transaction ID filled, all activated licenses will be returned to the license pool. This allows for flexible license management and redistribution as needed.
- **Transaction ID Requirement:** Administrators must enter a 5-digit positive number to set the transaction ID.
- **Profile with BADGEID and Product Names:** When a profile is pushed with a BADGEID and a list of product names to be activated, previously allocated licenses on the devices will be released. This ensures that licenses are activated according to the updated profile.

To add ZLicenseMgr application to ZDNA apps collection and to create app configuration profile , please refer below link

<https://techdocs.zebra.com/zebradna/5-1/usage/#addappstocollection>


App Details

×

Hosting Location

☐ Zebra Storage


Linked File Details

 Browse File

Zebra Storage File Link URL

☐ Google Drive

Linked File Details

 Browse File

Google Drive File Link URL

☒ Specified Server

https://firebasestorage.googleapis.com/v0/b/emc-scdemoapp-mindteck-uat2-t.apps

Upload App

Uploading allows zDNA Cloud to automatically populate fields for app and package name, version number, configuration parameters, etc. Apps uploaded here are not hosted by Zebra for distribution purposes.

Upload

No, Enter Manually

License Manager User Guide

Create New App Setup

×

1 Select App

2 Configure App

3 Name and Description

4 Review and Apply

Select an App from

☐ Zebra Collection

App Name

App Version

For Android version compatibility info, visit Zebra Collection.

☒ My Collection

1 App Name

LicenceMgr

2 App Version

14.0.50

If the desired "My Collection" app is not listed above, you can add it.

Add an App to My Collection

Discard

3 Next >

Configuration Details

Search Configuration

Setting Category

Transaction ID

Enter your unique Transaction ID and update it whenever the profile is edited or redeployed to ensure correct processing.

Enterprise Reset Persistence

Select whether all Zebra licenses should persist locally on device following an Enterprise Reset.

License-server Proxy URL

Enter the URL of the proxy server in use for controlling licensing requests (optional).

Badge ID Licensing

Open to configure license activation using Badge ID and Product Name.

☒ Cloud Server Type

Test Cloud

☒ Badge ID

BDG-B****3U

☒ Product Licenses

Add Product Name

☒ Product Name

Enterprise Browser Android Dev

Discard

< Previous

Next >

Select App

Configure App

Name and Description

Review and Apply

Configuration Details

Edit

Transaction ID

Transaction ID

123

Badge ID Licensing

Cloud Server Type

Cloud Server Type

Test Cloud

Badge ID

Badge ID

BDG-B****3U

Product Licenses

Product Name

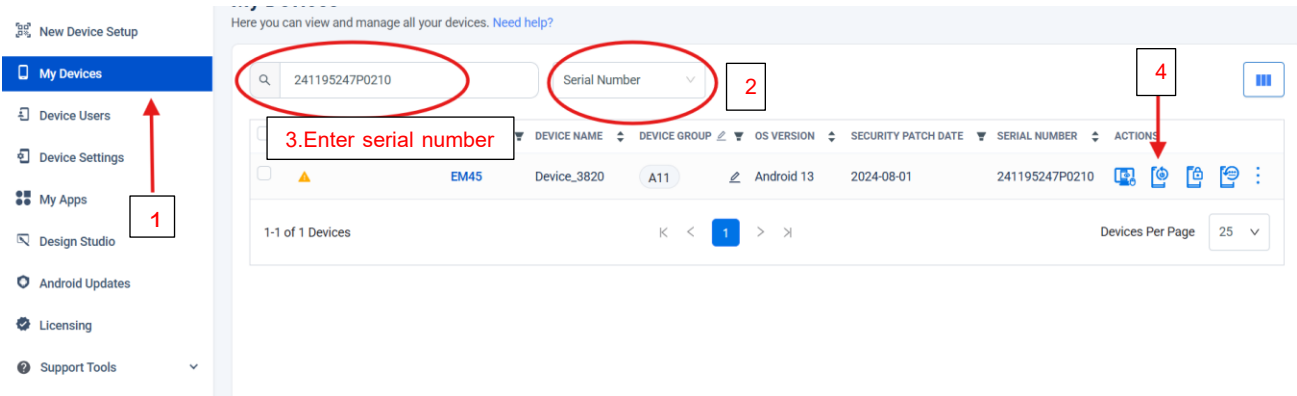
Product Name

Enterprise Browser Android Device Term License

Save

< Previous

Apply Now



Note:

- The ZSL app must be running in the background to process XML files present in the ZSL sandbox path. This applies to both app restriction use cases and XML direct deployment to License Manager use cases.
- A reboot is mandatory post-installation of the new license manager APK before activating any licenses.

License Management with Third-Party EMMs

For EMM solutions that support APK uploads, such as SOTI and AirWatch, licenses can be activated or returned using app configuration files. This integration provides seamless license management within existing EMM frameworks.

Process Overview:

- **Upload ZLicenseMgr:** Download ZLicenseMgr from the Zebra Support Portal and upload it as an enterprise app to your EMM collection. This integration facilitates centralized license management.
- **App Configuration Profile:** Deploy the ZLicenseMgr APK to devices alongside an app configuration profile for license activation or return. This profile includes the necessary settings and parameters for the licensing process.
- **Device Reboot Requirement:** A device reboot is mandatory if an app upgrade is performed. However, no reboot is required for subsequent activation or license return-related app configuration deployments.
- **Transaction ID Modification:** Ensure the transaction ID is modified for every app configuration profile change pushed to the device. This change is essential for detecting and applying profile updates.
- **Returning Licenses to Pool:** Deploying an app configuration profile with an empty bundle array and only the transaction ID filled will result in all activated licenses being returned to the server pool.
- **Transaction ID Requirement:** Administrators must enter a 5-digit positive number to set the transaction ID.
- **Profile with BADGEID and Product Names:** When a profile is pushed with a BADGEID and a list of product names to be activated, previously allocated licenses on the devices will be released. This ensures that licenses are activated according to the updated profile.



Note:

- The ZSL app must be running in the background to process XML files present in the ZSL sandbox path. This applies to both app restriction use cases and XML direct deployment to License Manager use cases.
- A reboot is mandatory post-installation of the new license manager APK before activating any licenses.
- Administrators are recommended to include the ReturnAllAID as the first profile before applying any Thales license activation. This is not mandatory but recommended if the intention is to maintain a single barcode for all products across devices

License Manager User Guide

SELECT APPS

?

Select the apps that you want to install on devices using the App Policy

Apps

App Source

Enterprise

Source

Import

Enterprise URL

Upload APK File *

LicenseMgrService.apk

Name

ZLicenseMgr

Version

14.0.50

Seller

Seller

Description

Advanced Configurations

→

CONFIGURE

ADVANCED CONFIGURATIONS

?

Configuration Options

App Details

Installation Options

Managed App Config

Search Configurations

Badge ID Licensing > Product Licenses

Enable Managed App Config

Advanced options available for the application

Transaction ID

1234567

Enterprise Reset Persistence

Please Select

License-server Proxy URL

Badge ID Licensing

Cloud Server Type

Test Cloud

Badge ID

B0G-BK****3U

Product Licenses

Add Product Name

NAME

Product Name 1

CANCEL

SAVE

License Activation from SOTI using ZLicenseMgr as enterprise application

https://www.zebra.com/content/dam/support-dam/en/documentation/unrestricted/video/0001/LicenseActivationUsing_SOTI_EnterpriseApp.mp4

License Activation with EMMs Supporting File Sync

For EMM solutions like SOTI that support the file sync feature, administrators can deploy Zebra MX XML configurations efficiently. This approach enables streamlined license activation using StageNow profiles.

Process Overview:

- **XML Export:** Administrators can perform an XML export of a StageNow license activation profile. This export provides the necessary configuration details for license activation tailored to Zebra devices.
- **Profile Configuration:** When creating the StageNow profile, include a “return all” action before the activation request for the required products. Placing the return all licenses action at the top of the profile ensures that any previously allocated licenses are released from the devices before activating the new list from the server.
- **File Sync Deployment:** Use the file sync option in your EMM to deploy the exported XML file to devices. This method ensures that devices supporting MX version 14.0 receive the correct configuration for activating licenses.
- **Compatibility:** Ensure that the deployment targets devices running MX version 14.0 or higher to guarantee compatibility with the XML configurations.

License Activation Using SOTI FileSync

https://www.zebra.com/content/dam/support-dam/en/documentation/unrestricted/video/0001/LicenseActivationUsing_SOTI_FileSync.mp4



Note:

- The ZSL app must be running in the background to process XML files present in the ZSL sandbox path. This applies to both app restriction use cases and XML direct deployment to License Manager use cases.
- Profiles are applied multiple times via SOTI FileSync. Enable “Execute Script Only if Files Transmitted” to ensure the policy is applied only once.

License Activation and Return via Configuration File Deployment

ZLicenseMgr supports license activation and return through the deployment of configuration files. This method allows administrators to manage licenses efficiently by leveraging Zebra MX’s file management capabilities.

Process Overview:

- **Create a License Activation Profile:** Administrators can create a license activation profile using ZLicenseMgr. It is recommended to include a “return all” action at the top of the profile. This ensures that any previously activated licenses are released from the device before activating the list of required product names specified in the profile.
- **Export as XML:** Once the profile is configured, export it as an XML file. This XML contains all necessary instructions for license management. Rename file licenseconf.xml. When exporting StageNow XML profiles, ensure encryption is disabled, as older MX versions do not support URL encryption.
- **Host and Transfer XML:**
 - Host the XML file on a server accessible by the devices.
 - Use Zebra MX’s FileMgr to transfer the XML file to the predefined location on the device.
- **File Transfer Locations:**
 - For devices running Android 10 and below: Transfer the file to /Android/data/com.zebra.licensemgrservice/licenseconf.xml.
 - For devices running Android 11 and above: Use advanced FileMgr configuration to deploy the file through Secure Storage Manager to com.zebra.licensemgrservice/licenseconf.xml.
- **Ensure Application Readiness:** Before issuing a file transfer command for license activation, administrators must ensure that the ZLicenseMgr app is launched. This readiness ensures that the application is prepared to process the configuration file immediately upon receipt.

```
<wap-provisioningdoc>
  <characteristic version="13.5" type="AppMgr">
    <parm name="Action" value="LaunchApplication"/>
    <parm name="ApplicationName" value="ZLicenseMgr"/>
  </characteristic>
</wap-provisioningdoc>
```

StageNow Activation for A10 and below via New License Manager for a device with MX version lower than MX 14.0
https://www.zebra.com/content/dam/support-dam/en/documentation/unrestricted/video/0001/SN_Activation_for_lower_MX_version_A10_and_below_devices.mp4

StageNow Activation for A11 and A13 via New License Manager for a device with MX version lower than MX 14.0
https://www.zebra.com/content/dam/support-dam/en/documentation/unrestricted/video/0001/SN_Activation_for_lower_MX_version_A11_and_A13_devices.mp4

NOTE:



- StageNow profiles created with the encrypted option are not compatible with Lollipop devices. For mixed device use cases, ensure that the encryption option is not used when deploying the latest StageNow tool.
- When attempting MDNA and EB BadgeID activation from a single barcode, a “Failed to connect” error message may appear.

License Activation and Return via Zebra OEMConfig Tools

Zebra OEMConfig Tools provides another method for managing licenses through app configuration profiles. This approach allows for precise control over which licenses are activated or returned, utilizing the capabilities of OEMConfig.

Process Overview:

- **Create App Configuration Profile:**
 - Use the Zebra OEMConfig Tools to create an app configuration profile.
 - The profile should include the BADGE ID and a list of product names that you wish to activate.
- **License Management:**
 - Licenses not listed in the profile will be automatically released from the device, ensuring that only the specified licenses are active.
 - Licenses included in the profile will be activated, allowing for targeted and efficient license management.
- **Empty Bundle Handling:**
 - If the app configuration profile is deployed with an empty bundle, it will result in all licenses being returned to the license server pool. This feature provides flexibility in managing license allocations dynamically.

OEM config for license activation and return

<https://www.zebra.com/content/dam/support-dam/en/documentation/unrestricted/video/0004/Activate-Using-Soti-with-OemConfig-as-an-Enterprise-App.mp4>

NOTE:



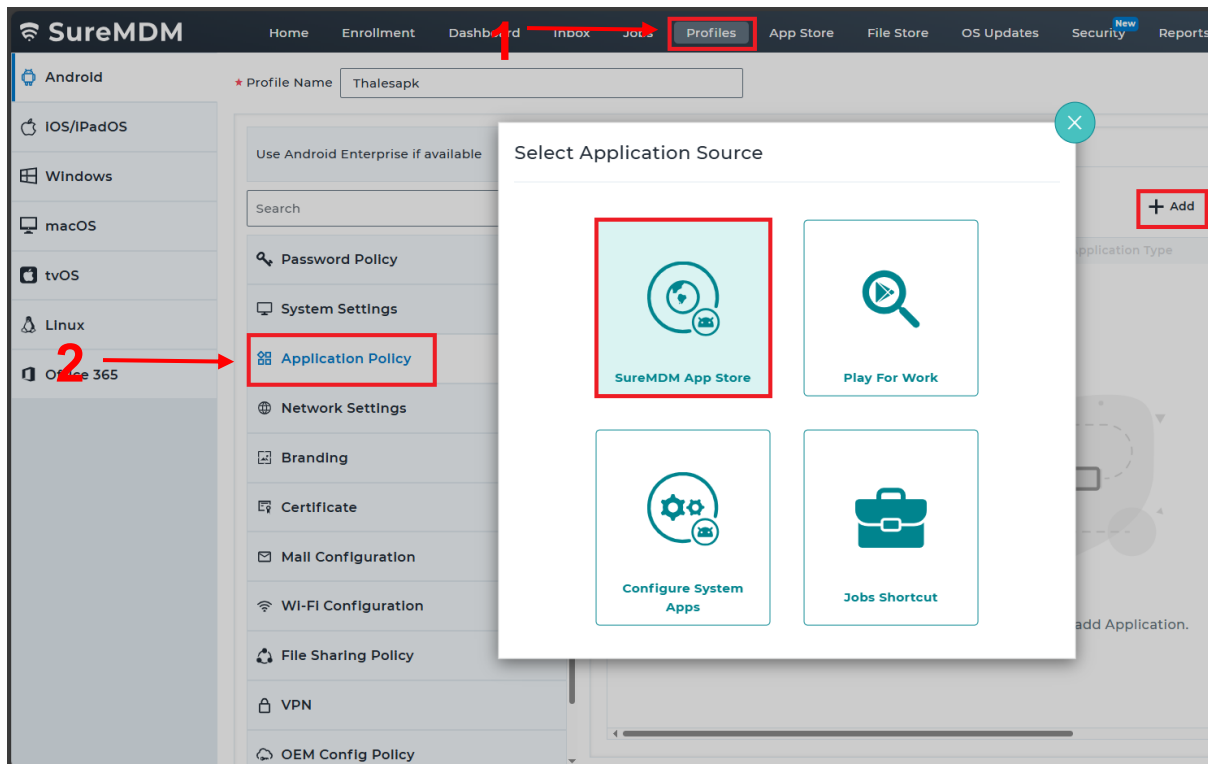
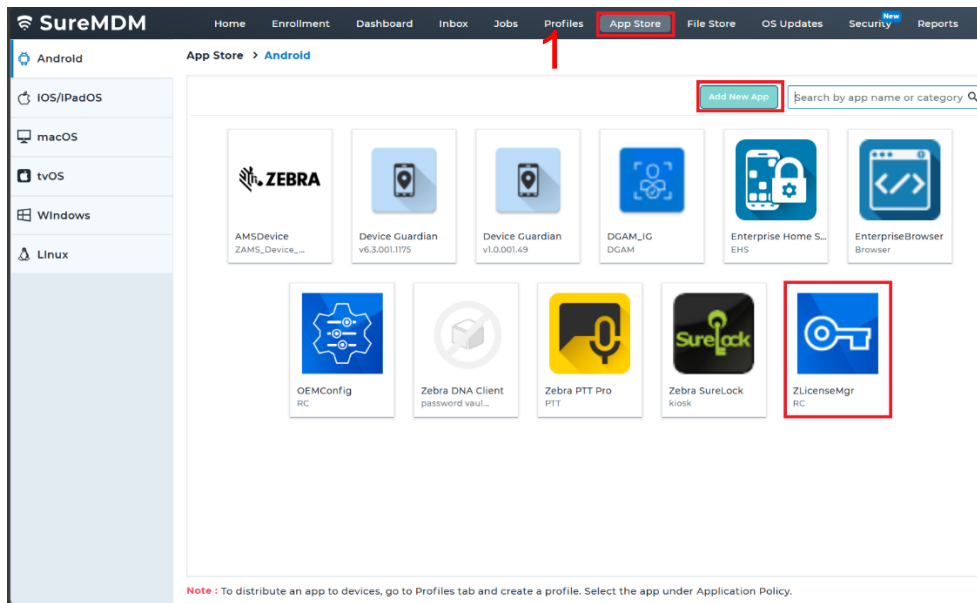
- StageNow profiles created with the encrypted option are not compatible with Lollipop devices. For mixed device use cases, ensure that the encryption option is not used when deploying the latest StageNow tool.
- When attempting MDNA and EB BadgeID activation from a single barcode, a “Failed to connect” error message may appear.

License Activation and Return via 42 Gear Tools

42Gears provides a streamlined approach to enterprise mobility management, enabling users to efficiently manage and secure devices through comprehensive configuration profiles.

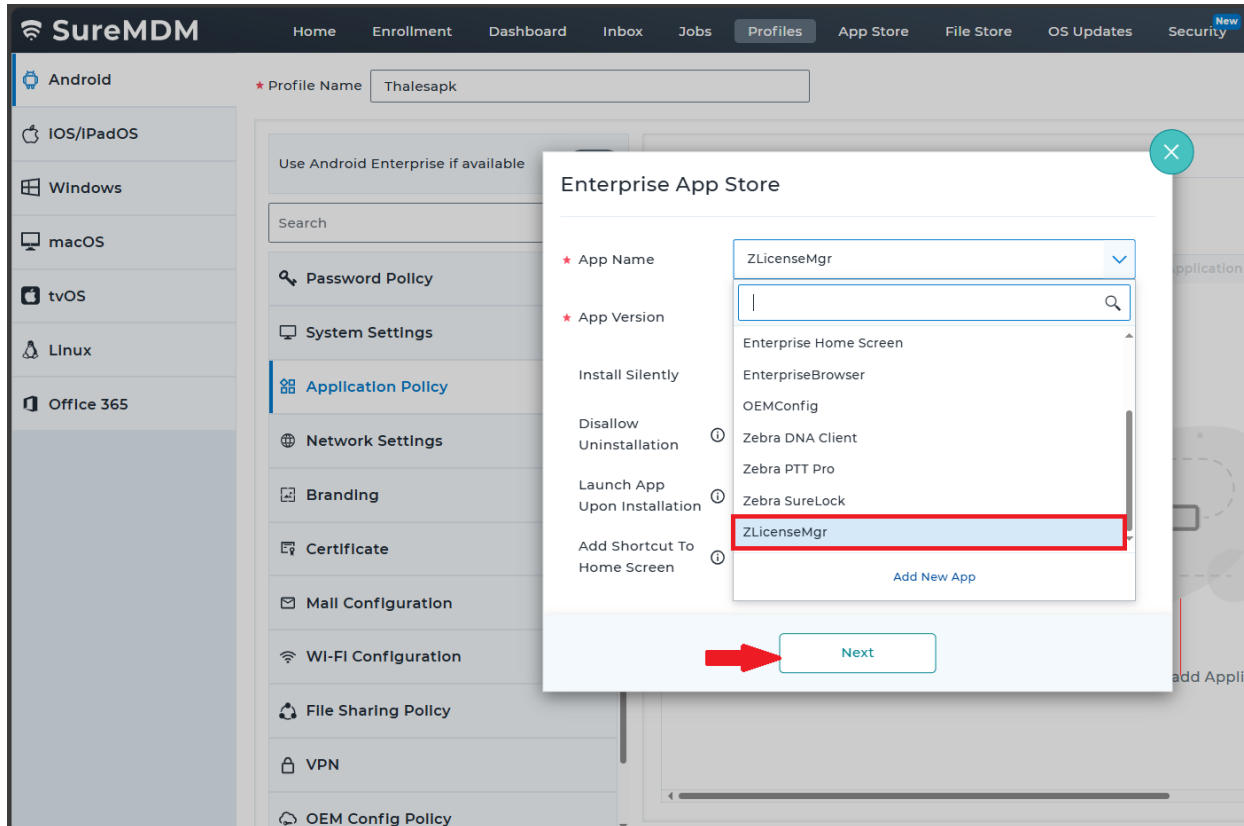
- Navigate to the Apps Store section and upload the ZLicenseMgr APK to the console.

License Manager User Guide

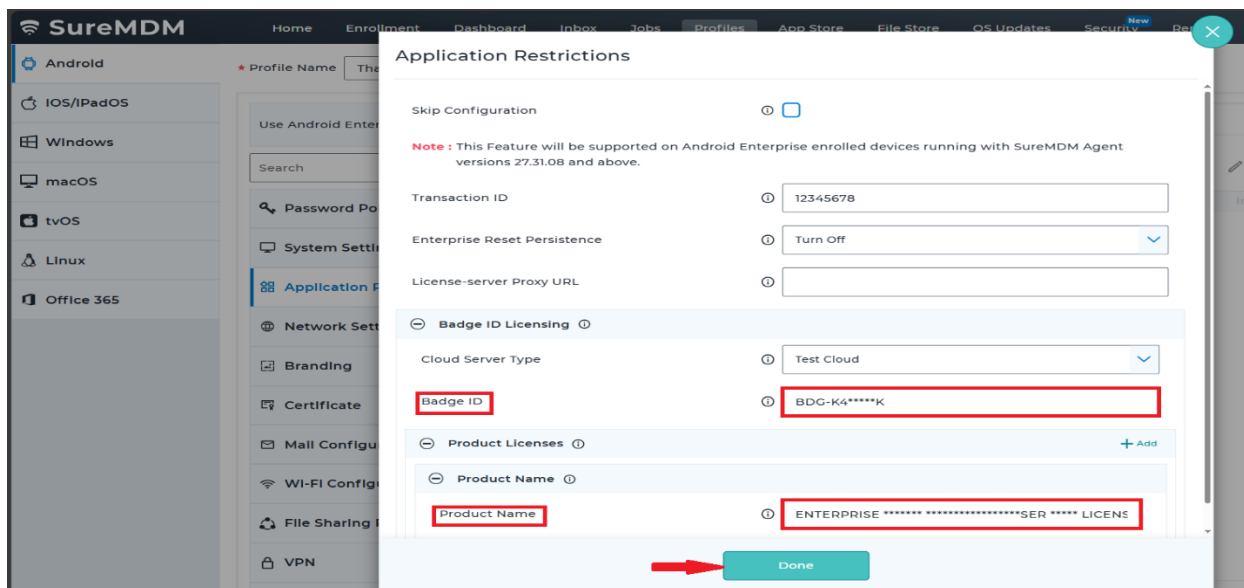


- Select the uploaded ZLicenseMgr app and create a configuration profile.

License Manager User Guide



- Enter the required details such as BADGEID, product names, and server type (Production or Test Cloud).
- Push the configuration profile to target Zebra devices.



42Gear for license activation and return

<https://www.zebra.com/content/dam/support-dam/en/documentation/unrestricted/video/0004/Activate-Using-42Gear-with-ZLicenseMgr-as-an-Enterprise-App.mp4>

Additional Notes and Usage Guidelines

- Activating a BADGEID-based license using the ZLicenseMgr application will erase licenses that were activated with previous versions of the application, which were part of the device OS or BSPA.
- Before associating a device with a new BADGEID, it's important to release all licenses from the device to ensure compliance, security, and proper resource management.
- After upgrading ZLicenseMgr on the device, a reboot is recommended to ensure all changes are properly applied and the system functions optimally.
- If ZLicenseMgr is downgraded, there is a possibility of losing licenses, so it's important to redeploy a license reactivation profile that is applicable and supported by the downgraded version.
- In the event of a clock reset leading to an invalid license state, it is necessary to correct the clock settings and perform a license reactivation to update and restore the license state.
- To prevent a profile from being applied multiple times through SOTI FileSync, enable the "Execute Script Only if Files Transmitted" option to ensure scripts are executed only when new files are transmitted.
- When upgrading ZLicenseMgr using the `adb install -r` command, you may encounter the "INSTALL_FAILED_SESSION_INVALID" error; however, the installation will still succeed.
- Third-party EMMs that do not support managed enterprise apps or the FileSync option for deploying MX XML profiles can utilize the OEMConfig Tools' pass-through command feature to upgrade ZLicenseMgr on the device.
- For Android versions A8 to A11, it is recommended to use the Legacy OEMConfig tool, whereas for Android version A13 and above, Zebra's new OEMConfig tool should be used.

Configuration Samples

- Below is a sample MX XML configuration that SOTI users can refer to for upgrading ZLicenseMgr and activating a BADGEID-based license on devices running Android 10 and below using the FileSync option:
- XML uses condition manager to give sufficient time to complete action before executing next command. And it varies based on the network strength and device performance.

XML SAMPLE :1

```
<wap-provisioningdoc>
  <characteristic version="14.0" type="LicenseMgr">
    <parm name="LicenseChoice" value="ZebraT"/>
    <characteristic type="LicenseZebraT">
      <characteristic type="LicenseActionChoice">
        <parm name="SelectedLicenseActionChoice" value="BadgeID"/>
        <parm name="ServerType" value="test"/>
        <parm name="BadgeID" value="BDG-BK****3U"/>
        <parm name="ProductName" value="Enterprise Browser Android Device License"/>
      </characteristic>
    </characteristic>
  </wap-provisioningdoc>
```

XML SAMPLE :2

```
<wap-provisioningdoc>
```

```
<characteristic version="9.1" type="Wi-Fi">
  <parm name="UseRegulatory" value="0" />
  <parm name="UseDiagnosticOptions" value="0" />
  <parm name="UseAdvancedOptions" value="0" />
  <parm name="NetworkAction" value="Add" />
  <characteristic type="network-profile">
    <parm name="SSID" value="IRIS_5G" />
    <parm name="SecurityMode" value="1" />
    <parm name="WPA Mode" value="1" />
    <characteristic type="key-details">
      <parm name="KeyType" value="Passphrase" />
      <parm name="ProtectKey" value="0" />
      <parm name="PassphraseWPAClear" value="*****" />
    </characteristic>
    <parm name="UseDHCP" value="1" />
    <parm name="UseProxy" value="0" />
  </characteristic>
  <parm name="UseHotspotOptions" value="0" />
</characteristic>

<characteristic version="6.0" type="ConditionMgr">
  <parm name="DataType" value="2" />
  <characteristic type="IntegerDetails">
    <parm name="IntegerSourceType" value="2" />
    <parm name="IntegerSystemValue" value="1" />
    <parm name="IntegerConditionTest" value="1" />
    <parm name="IntegerConstantValue" value="0" />
  </characteristic>
  <parm name="ConditionMetAction" value="0" />
  <parm name="ConditionNotMetAction" value="2" />
  <parm name="ConditionRepeatCount" value="5" />
  <parm name="ConditionRepeatInterval" value="4" />
  <parm name="ConditionFailMessage" value="timeout" />
  <parm name="SuppressMessage" value="1" />
  <parm name="ConditionWaitMessage" value="" />
</characteristic>

<characteristic version="7.0" type="AppMgr">
  <parm name="Action" value="Upgrade" />
  <parm name="APK" value="/sdcard/ZLicenseMgr.apk" />
</characteristic>

<characteristic version="6.0" type="ConditionMgr">
  <parm name="DataType" value="2" />
  <characteristic type="IntegerDetails">
    <parm name="IntegerSourceType" value="2" />
    <parm name="IntegerSystemValue" value="1" />
    <parm name="IntegerConditionTest" value="1" />
    <parm name="IntegerConstantValue" value="0" />
  </characteristic>
  <parm name="ConditionMetAction" value="0" />
  <parm name="ConditionNotMetAction" value="2" />
  <parm name="ConditionRepeatCount" value="5" />
  <parm name="ConditionRepeatInterval" value="4" />
  <parm name="ConditionFailMessage" value="timeout" />
  <parm name="SuppressMessage" value="1" />
  <parm name="ConditionWaitMessage" value="" />
</characteristic>
```


License Manager User Guide

```
</characteristic>
<parm name="ConditionMetAction" value="0" />
<parm name="ConditionNotMetAction" value="2" />
<parm name="ConditionRepeatCount" value="5" />
<parm name="ConditionRepeatInterval" value="4" />
<parm name="ConditionFailMessage" value="timeout" />
<parm name="SuppressMessage" value="1" />
<parm name="ConditionWaitMessage" value="" />
</characteristic>
<characteristic version="7.0" type="AppMgr">
  <parm name="Action" value="LaunchApplication" />
  <parm name="ApplicationName" value="ZLicenseMgr" />
</characteristic>
<characteristic version="6.0" type="ConditionMgr">
  <parm name="DataType" value="2" />
  <characteristic type="IntegerDetails">
    <parm name="IntegerSourceType" value="2" />
    <parm name="IntegerSystemValue" value="1" />
    <parm name="IntegerConditionTest" value="1" />
    <parm name="IntegerConstantValue" value="0" />
  </characteristic>
  <parm name="ConditionMetAction" value="0" />
  <parm name="ConditionNotMetAction" value="2" />
  <parm name="ConditionRepeatCount" value="5" />
  <parm name="ConditionRepeatInterval" value="4" />
  <parm name="ConditionFailMessage" value="timeout" />
  <parm name="SuppressMessage" value="1" />
  <parm name="ConditionWaitMessage" value="" />
</characteristic>
<characteristic version="5.0" type="FileMgr">
  <parm name="FileAction" value="1" />
  <characteristic type="file-details">
    <parm name="TargetAccessMethod" value="2" />
    <parm name="TargetPathAndFileName"
value="/sdcard/Android/data/com.zebra.licensemgrservice/licenseconf.xml" />
    <parm name="SourceAccessMethod" value="1" />
    <parm name="SourceURI" value="https://firebasestorage.googleapis.com/v0/b/emc-scdemoapp-mindteck-uat2-
t.appspot.com/o/Other%20Files%20FEB-eval-BDG-Z4MX2Q6V%202.xml?alt=media" />
  </characteristic>
</characteristic>
</wap-provisioningdoc>
```

XML SAMPLE :3

Below is a sample MX XML configuration that SOTI users can refer to for upgrading ZLicenseMgr and activating a BADGEID-based license on devices running Android 11 and above with MX version less than 14.0

```
<wap-provisioningdoc>
<characteristic version="13.5" type="Wi-Fi">
  <parm name="UseRegulatory" value="0" />
  <parm name="UseDiagnosticOptions" value="0" />
  <parm name="UseAdvancedOptions" value="0" />
  <parm name="NetworkAction" value="Add" />
  <characteristic type="network-profile">
    <parm name="SSID" value="Candy_5G" />
    <parm name="SecurityMode" value="1" />
    <parm name="WPAModePersonal" value="1" />
    <characteristic type="key-details">
      <parm name="KeyType" value="Passphrase" />
      <parm name="ProtectKey" value="0" />
      <parm name="PassphraseWPAClear" value="*****" />
    </characteristic>
    <parm name="UseDHCP" value="1" />
    <parm name="UseProxy" value="0" />
  </characteristic>
  <parm name="UseHotspotOptions" value="0" />
</characteristic>
<characteristic version="11.3" type="FileMgr">
  <parm name="FileAction" value="1" />
  <characteristic type="file-details">
    <parm name="TargetAccessMethod" value="2" />
    <parm name="TargetPathAndFileName" value="/sdcard/LicenseMgrService.apk" />
    <parm name="IfDuplicate" value="1" />
    <parm name="SourceAccessMethod" value="1" />
    <parm name="SourceURI" value="https://firebasestorage.googleapis.com/v0/b/emc-scdemoapp-mindteck-uat2-t.appspot.com/o/Other%20Files%2FTestTeam_Only%2FLicenseMgrService_6.6.60.apk?alt=media" />
  </characteristic>
</characteristic>
<characteristic version="13.5" type="AppMgr">
  <parm name="Action" value="DisableApplication" />
  <parm name="Package" value="com.android.vending" />
</characteristic>
<characteristic version="13.5" type="AppMgr">
  <parm name="Action" value="Upgrade" />
  <parm name="APK" value="/sdcard/LicenseMgrService.apk" />
</characteristic>
<characteristic version="13.5" type="AppMgr">
  <parm name="Action" value="EnableApplication" />
  <parm name="Package" value="com.android.vending" />
</characteristic>
```

```

</characteristic>
<characteristic version="6.0" type="ConditionMgr">
  <parm name="DataType" value="2" />
  <characteristic type="IntegerDetails">
    <parm name="IntegerSourceType" value="2" />
    <parm name="IntegerSystemValue" value="1" />
    <parm name="IntegerConditionTest" value="1" />
    <parm name="IntegerConstantValue" value="0" />
  </characteristic>
  <parm name="ConditionMetAction" value="0" />
  <parm name="ConditionNotMetAction" value="2" />
  <parm name="ConditionRepeatCount" value="5" />
  <parm name="ConditionRepeatInterval" value="4" />
  <parm name="ConditionFailMessage" value="timeout" />
  <parm name="SuppressMessage" value="1" />
  <parm name="ConditionWaitMessage" value="" />
</characteristic>
<characteristic version="13.5" type="AppMgr">
  <parm name="Action" value="LaunchApplication" />
  <parm name="ApplicationName" value="ZLicenseMgr" />
</characteristic>
<characteristic version="11.3" type="FileMgr">
  <parm name="FileAction" value="10" />
  <characteristic type="file-details">
    <parm name="TargetApplicationAndFileName" value="com.zebra.licensemgrservice/licenseconf.xml" />
    <parm name="TargetApplicationSignature" value="" />
    <parm name="PersistFile" value="0" />
    <parm name="SourceAccessMethod2" value="1" />
    <parm name="SourceURI2" value="https://firebasestorage.googleapis.com/v0/b/emc-scdemoapp-mindteck-uat2-t.appspot.com/o/Other%20Files%2Fnew_MDNA-and-eb_PersistFlagON.xml?alt=media" />
  </characteristic>
</characteristic>
</wap-provisioningdoc>

```

XML SAMPLE :4

Below is a sample MX XML configuration that SOTI users can refer to for upgrading ZLicenseMgr and activating a BADGEID-based license on devices running Android 11 and above **with MX14.0**

```

<wap-provisioningdoc>
  <characteristic version="14.0" type="Wi-Fi">
    <parm name="UseRegulatory" value="0" />
    <parm name="UseDiagnosticOptions" value="0" />
    <parm name="UseAdvancedOptions" value="0" />
    <parm name="NetworkAction" value="Add" />
  </characteristic>
</wap-provisioningdoc>

```

```

<characteristic type="network-profile">
  <parm name="SSID" value="Candy_5G" />
  <parm name="SecurityMode" value="1" />
  <parm name="WPAModePersonal" value="1" />
  <characteristic type="key-details">
    <parm name="KeyType" value="Passphrase" />
    <parm name="ProtectKey" value="0" />
    <parm name="PassphraseWPAClear" value="*****" />
  </characteristic>
  <parm name="UseDHCP" value="1" />
  <parm name="UseProxy" value="0" />
</characteristic>
<parm name="UseHotspotOptions" value="0" />
</characteristic>
<characteristic version="11.3" type="FileMgr">
  <parm name="FileAction" value="1" />
  <characteristic type="file-details">
    <parm name="TargetAccessMethod" value="2" />
    <parm name="TargetPathAndFileName" value="/sdcard/LicenseMgrService.apk" />
    <parm name="IfDuplicate" value="1" />
    <parm name="SourceAccessMethod" value="1" />
    <parm name="SourceURI" value="https://firebasestorage.googleapis.com/v0/b/emc-scdemoapp-mindteck-uat2-
t.appspot.com/o/Other%20Files%2FTestTeam_Only%2FLicenseMgrService_6.6.60.apk?alt=media" />
  </characteristic>
</characteristic>
<characteristic version="14.0" type="AppMgr">
  <parm name="Action" value="DisableApplication" />
  <parm name="Package" value="com.android.vending" />
</characteristic>
<characteristic version="14.0" type="AppMgr">
  <parm name="Action" value="Upgrade" />
  <parm name="APK" value="/sdcard/LicenseMgrService.apk" />
</characteristic>
<characteristic version="14.0" type="AppMgr">
  <parm name="Action" value="EnableApplication" />
  <parm name="Package" value="com.android.vending" />
</characteristic>
<characteristic version="6.0" type="ConditionMgr">
  <parm name="DataType" value="2" />
  <characteristic type="IntegerDetails">
    <parm name="IntegerSourceType" value="2" />
    <parm name="IntegerSystemValue" value="1" />
    <parm name="IntegerConditionTest" value="1" />
    <parm name="IntegerConstantValue" value="0" />
  </characteristic>
  <parm name="ConditionMetAction" value="0" />

```

License Manager User Guide

```
<parm name="ConditionNotMetAction" value="2" />
<parm name="ConditionRepeatCount" value="5" />
<parm name="ConditionRepeatInterval" value="4" />
<parm name="ConditionFailMessage" value="timeout" />
<parm name="SuppressMessage" value="1" />
<parm name="ConditionWaitMessage" value="" />
</characteristic>
<characteristic version="14.0" type="AppMgr">
  <parm name="Action" value="LaunchApplication" />
  <parm name="ApplicationName" value="ZLicenseMgr" />
</characteristic>
<characteristic version="11.3" type="FileMgr">
  <parm name="FileAction" value="10" />
  <characteristic type="file-details">
    <parm name="TargetApplicationAndFileName" value="com.zebra.licensemgrservice/licenseconf.xml" />
    <parm name="TargetApplicationSignature" value="" />
    <parm name="PersistFile" value="0" />
    <parm name="SourceAccessMethod2" value="1" />
    <parm name="SourceURI2" value="https://firebasestorage.googleapis.com/v0/b/emc-scdemoapp-mindteck-uat2-t.appspot.com/o/Other%20Files%20Fnew_MDNA-and-eb_PersistFlagON.xml?alt=media" />
  </characteristic>
</characteristic>
</wap-provisioningdoc>
```

Below is a sample MX XML configuration that EMM users can refer to for upgrading ZLicenseMgr and activating a BADGEID-based license on devices running Android 11 and below (A11 to A8) using the Legacy OEMConfig Tool:

Note : Users must apply [XML SAMPLE :2](#) inside the submit XML as specified below.

ADVANCED CONFIGURATIONS

Configuration Options

App Details

Installation Options

Managed App Config

← Add Transaction Step > Device Administration Configuration

DataWedge Configuration

Device Administration Configuration

Action

Do nothing

Allow Submit XML Package Name

Disallow Submit XML Package Name

Submit XML

Allow Package Update Package Name

Disallow Package Update Package Name

<wap-provisioningdoc> <characteristic

CANCEL

SAVE

License Manager User Guide

Below is a sample MX XML configuration that EMM users can refer to for upgrading ZLicenseMgr and activating a BADGEID-based license on devices running Android 13 and below using the Zebra's new OEMConfig Tool:

Note : Users must apply [XML SAMPLE :3](#) or [XML SAMPLE :4](#)(with **MX14.0**) inside the pass-through as specified below

EDIT APP POLICY | LicenseActication_OEMPassThrough

GENERAL APPS

ADVANCED CONFIGURATIONS

Configuration Options

App Details

Installation Options

Managed App Config

Search Configurations

Displays a path by clicking on a property

Remote Scanner Configurations

Wake-Up Configuration

Pass-Through Command

Logs Configuration

UI Configuration

<wap-provisioningdoc> <characteristic

CANCEL SAVE

Reference video : License Activation Using OEM Passthrough from SOTI

https://www.zebra.com/content/dam/support-dam/en/documentation/unrestricted/video/0001/LicenseActivationUsing_SOTI_OEM_Passthrough.mp4

License Manager User Guide

- Below is a sample MX XML configuration XML for launching ZLicenseMgr application

```
<wap-provisioningdoc>
  <characteristic version="13.5" type="AppMgr">
    <parm name="Action" value="LaunchApplication" />
    <parm name="ApplicationName" value="NewLicenseAgent" />
  </characteristic>
</wap-provisioningdoc>
```

- Below is a sample MX XML configuration XML for rebooting the device.

```
<wap-provisioningdoc>
  <characteristic version="13.1" type="PowerMgr">
    <parm name="ResetAction" value="4" />
  </characteristic>
</wap-provisioningdoc>
```

