Customize Installation using Golden ISO

Table 1: Feature History Table

Feature Name	Release Information	Description
Automatic Install of Bridging Bug Fix RPMs	Release 7.5.1	This feature enables an easy one-step, no prompt upgrade, or downgrade, based on GISO. This removes the dependency on having to manually install the bridging bug fix RPMs before performing an upgrade or a downgrade.

Golden ISO (GISO) is a customized ISO that a user can build to suit the installation requirement. The user can customize the installable image to include the standard base image with the basic functional components, and add additional RPMs, SMUs and configuration files based on requirement.

The ease of installation and the time taken to seamlessly install or upgrade a system plays a vital role in a cloud-scale network. An installation process that is time-consuming and complex affects the resiliency and scale of the network. The GISO simplifies the installation process, automates the installation workflow, and manages the dependencies in RPMs and SMUs automatically.

GISO is built using a build script `gisobuild.py` available on the github location Github location.

From Cisco IOS XR Release 7.5.1, you can use the Automatic Install of Bridging Bug Fix RPMs feature to install the bridging bug fix RPMs that are prerequisite for a system upgrade or a downgrade. You need to add the required Bridging Bug Fix RPMs into the customized ISO built using Cisco Golden ISO (GISO) build script **gisobuild.py**. The GISO can include bridging Bug Fix RPMs for multiple releases, and installs only the specific bridging Bug Fix RPMs required for the target release. The bridging bug fix RPMs can be used in the following scenarios:

- To resolve a bug that might stop upgrade.
- The latest version has new prerequisite requirements that are not met by the earlier version.

When a system boots with GISO, additional SMUs and RPMs in GISO are installed automatically, and the router is pre-configured with the XR configuration in GISO. For more information about downloading and installing GISO, see Install Golden ISO, on page 13.

The capabilities of GISO can be used in the following scenarios:

- Migration from IOS XR 32-bit to IOS XR 64-bit

- Initial deployment of the router
- Software disaster recovery
- System upgrade from one base version to another
- System upgrade from same base version but with additional SMUs
- Install update to identify and update dependant packages

- Limitations, on page 4
- Customize Installation using Golden ISO, on page 3
- Golden ISO Workflow, on page 4
- Build Golden ISO, on page 5
- Install Golden ISO, on page 13

Limitations

The following are the known problems and limitations with the customized ISO:

- GISO image size more than 1.8 GB is not supported. The maximum image size for RSP880-LT-SE/TR is 1.599 GB.
- Building and booting GISO for asynchronous package (a package of different release than the ISO) is not supported.
- Verifying the XR configuration is not supported in the GISO build script `gisobuild.py`.
- Renaming a GISO build and then installing from the renamed GISO build is not supported.
- Install operation over IPv6 is not supported.
- Migrating from IOS XR 32-bit to 64-bit OS using GISO involves the following restrictions:
  - The IOS XR 32-bit to 64-bit conversion script does not support file names exceeding 48 characters.
  - The IOS XR 32-bit OS has a maximum file size limit of 2 GB. Ensure that GISO does not exceed that limit.

    For more information about migration methods and system requirements, see the Migration Guide for Cisco ASR 9000 Series Routers.

Customize Installation using Golden ISO

Table 2: Feature History Table

Feature Name	Release Information	Description
Automatic Install of Bridging Bug Fix RPMs	Release 7.5.1	This feature enables an easy one-step, no prompt upgrade, or downgrade, based on GISO. This removes the dependency on having to manually install the bridging bug fix RPMs before performing an upgrade or a downgrade.

Golden ISO (GISO) is a customized ISO that a user can build to suit the installation requirement. The user can customize the installable image to include the standard base image with the basic functional components, and add additional RPMs, SMUs and configuration files based on requirement.

The ease of installation and the time taken to seamlessly install or upgrade a system plays a vital role in a cloud-scale network. An installation process that is time-consuming and complex affects the resiliency and scale of the network. The GISO simplifies the installation process, automates the installation workflow, and manages the dependencies in RPMs and SMUs automatically.

GISO is built using a build script `gisobuild.py` available on the github location Github location.

From Cisco IOS XR Release 7.5.1, you can use the Automatic Install of Bridging Bug Fix RPMs feature to install the bridging bug fix RPMs that are prerequisite for a system upgrade or a downgrade. You need to add the required Bridging Bug Fix RPMs into the customized ISO built using Cisco Golden ISO (GISO) build script **gisobuild.py**. The GISO can include bridging Bug Fix RPMs for multiple releases, and installs only the specific bridging Bug Fix RPMs required for the target release. The bridging bug fix RPMs can be used in the following scenarios:

- To resolve a bug that might stop upgrade.
- The latest version has new prerequisite requirements that are not met by the earlier version.

When a system boots with GISO, additional SMUs and RPMs in GISO are installed automatically, and the router is pre-configured with the XR configuration in GISO. For more information about downloading and installing GISO, see Install Golden ISO, on page 13.

The capabilities of GISO can be used in the following scenarios:

- Migration from IOS XR 32-bit to IOS XR 64-bit
- Initial deployment of the router
- Software disaster recovery
- System upgrade from one base version to another
- System upgrade from same base version but with additional SMUs
- Install update to identify and update dependant packages

Limitations

The following are the known problems and limitations with the customized ISO:

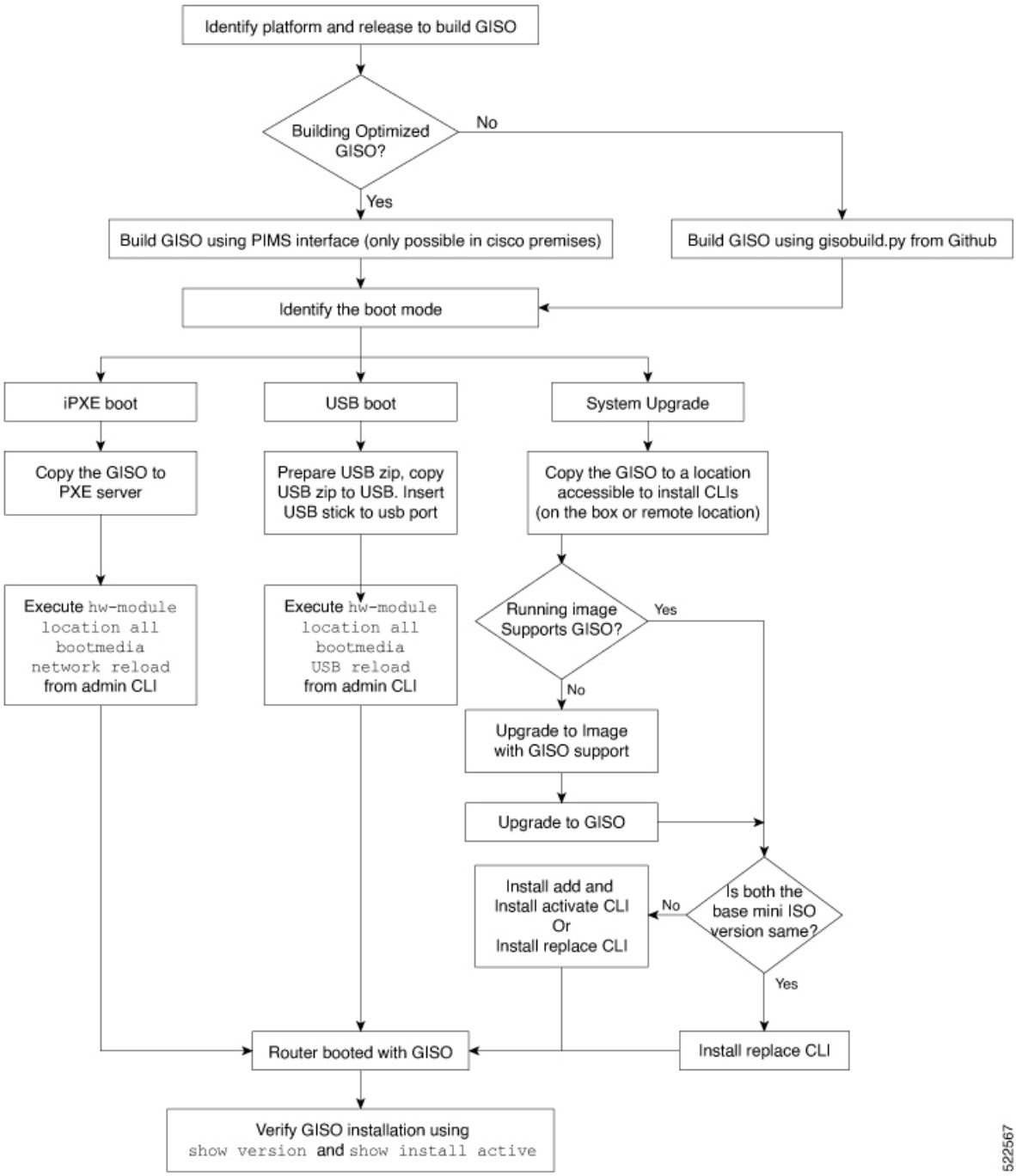
- GISO image size more than 1.8 GB is not supported. The maximum image size for RSP880-LT-SE/TR is 1.599 GB.
- Building and booting GISO for asynchronous package (a package of different release than the ISO) is not supported.
- Verifying the XR configuration is not supported in the GISO build script `gisobuild.py`.
- Renaming a GISO build and then installing from the renamed GISO build is not supported.
- Install operation over IPv6 is not supported.
- Migrating from IOS XR 32-bit to 64-bit OS using GISO involves the following restrictions:
  - The IOS XR 32-bit to 64-bit conversion script does not support file names exceeding 48 characters.
  - The IOS XR 32-bit OS has a maximum file size limit of 2 GB. Ensure that GISO does not exceed that limit.

    For more information about migration methods and system requirements, see the Migration Guide for Cisco ASR 9000 Series Routers.

Golden ISO Workflow

The following image shows the workflow for building and installing golden ISO.

Figure 1: Golden ISO Workflow



Build Golden ISO

The customized ISO is built using Cisco Golden ISO (GISO) build script `gisobuild.py` available on the Github location.

The GISO build script supports automatic dependency management, and provides these functionalities:

- Builds RPM database of all the packages present in package repository.
- Scans the repositories and selects the relevant Cisco RPMs that matches the input iso.
- Skips and removes third-party RPMs that are not SMUs of already existing third-party base package in mini-x.iso.
- Displays an error and exits build process if there are multiple base RPMs of same release but different versions.
- Performs compatibility check and dependency check for all the RPMs. For example, the child RPM asr9k-mpls-te-rsvp is dependent on the parent RPM asr9k-mpls. If only the child RPM is included, the Golden ISO build fails.

Build Golden ISO Using Script

Table 3: Feature History Table

Feature Name	Release Information	Description
Enhanced Golden ISO Build Tool	Release 7.5.1	This enhancement provides you with the flexibility to use the `gisobuild.py` tool to build GISO images using Cisco IOS XR software commands, YAML-based template file, or docker capability to suit your customized install requirements. When you build a GISO, you can also specify Zero Touch Provisioning (ZTP) initialization file, script initialization file, Cisco IOS XR configuration file, and SMUs in addition to using the base image and optional RPMs to automatically provision the router.

To build GISO, provide the following input parameters to the script:

- Base mini-x.iso (mandatory)
- XR configuration file (optional)
- one or more Cisco-specific SMUs for host, XR and System admin (optional)
- one or more third-party SMUs for host, XR and System admin (optional)
- Label for golden ISO (optional)
- Optional RPMs
- ZTP initialization `ztp.ini` file (optional)

- Script initialization `script.ini` file (optional)

The GISO script does not support verification of XR configuration.

Note To successfully add k9sec RPM to GISO, change the permission of the file to `644` using the **chmod** command.

```
chmod 644 [k9 sec rpm]
```

Cisco IOS XR, Release 7.5.1 introduces enhancements to the `gisobuild.py` GISO build tool. You can also add a `ztp.ini` ZTP initialization and `script.ini` Script initialization file. The ZTP configuration is applied on the router when the current software version is replaced or rolled back to a version with GISO image, and is used whenever ZTP is run to automatically provision the router. The tool supports more than one repository. You can use CLI command, docker, or a YAML file to build GISO.

Note

- Set the `migration` option to `true` when migrating from IOS XR 32-bit to IOS XR 64-bit software for Cisco ASR 9000 series routers.
- Set the `clean` option to `true` if you use the same build directory after the first GISO is created. Ensure that you set the option to `true` for every successive GISO build.
- Set the `docker` option to `true` if you are building GISO using docker.
- Ensure that the format and syntax of the YAML file is intact to avoid errors when building a GISO. For example, if the `:` symbol is missing, or if an unsupported symbol is used in the template, the GISO build displays errors.

The `gisobuild.py` tool can be run either natively or on systems where docker service is enabled and has the ability to pull published docker images. Prefer building the image using the docker as it does not require additional privileges:.

Note The `full-iso` option is used to build a full ISO image `xrv9k-full-x-7.5.1.iso` specific to Cisco IOS XRv 9000 routers. Starting Cisco IOS XR, Release 7.8.1, the full ISO image must not be used to build GISO.

To build GISO, perform the following steps:

Before you begin

- The system where GISO is built must meet the following requirements:
  - System must have Python version 3.6 and later.
  - System must have free disk space of minimum 12 GB.
  - Verify that the Linux utilities `mount`, `rm`, `cp`, `umount`, `zcat`, `chroot`, `mkisofs` are present in the system. These utilities will be used by the script. Ensure privileges are available to execute all of these Linux commands. However, if you are using docker, these utilities are not required.
  - Kernel version of the system must be later than 3.16 or later than the version of kernel of Cisco ISO.

- Verify that a `libyaml rpm` supported by the Linux kernel is available to successfully `import yaml` in the tool.
- User should have proper permission for security rpm(k9sec-rpm) in rpm repository, else security rpm would be ignored for Golden ISO creation.

- The system from where the `gisobuild.py` script is executed must have root credentials. This is not mandatory if you are building the image within a docker container.
- We recommend that you perform a `git pull` operation before you use the `gisobuild.py` script to ensure you obtain the latest version of the script for the Python version.

Procedure

Step 1 Copy the script `gisobuild.py` from the Github repository to an offline system or external server where the GISO will be built. Ensure that this system meets the pre-requisites described above in the *Before You Begin* section.

Step 2 Run the script `gisobuild.py` and provide parameters to build the golden ISO off the router. Ensure that all RPMs and SMUs are present in the same directory or on a repository. The number of RPMs and SMUs that can be used to build the Golden ISO is 64.

```
usage: gisobuild.py [-h] [--iso ISO] [--repo REPO [REPO ...]]
                    [--bridging-fixes BRIDGE_FIXES [BRIDGE_FIXES ...]]
                    [--xrconfig XRCONFIG] [--ztp-ini ZTP_INI] [--label LABEL]
                    [--out-directory OUT_DIRECTORY] [--yamlfile CLI_YAML] [--clean]
                    [--pkglist PKGLIST [PKGLIST ...]] [--script SCRIPT] [--docker]
                    [--x86-only] [--migration]
                    [--remove-packages REMOVE_PACKAGES [REMOVE_PACKAGES ...]]
                    [--skip-usb-image] [--copy-dir COPY_DIRECTORY]
                    [--clear-bridging-fixes] [--verbose-dep-check] [--debug]
                    [--version]

Utility to build Golden ISO for IOS-XR.

optional arguments:
  -h, --help            show this help message and exit
  --iso ISO             Path to Mini.iso/Full.iso file
  --repo REPO [REPO ...]
                        Path to RPM repository. For LNT, user can specify .rpm, .tgz,
                        .tar filenames, or directories. RPMs are only used if already
                        included in the ISO, or specified by the user via the
                        --pkglist option.
  --bridging-fixes BRIDGE_FIXES [BRIDGE_FIXES ...]
                        Bridging rpms to package. For EXR, takes from-release or rpm
                        names; for LNT, the user can specify the same file types as for
                        the --repo option.
  --xrconfig XRCONFIG   Path to XR config file
  --ztp-ini ZTP_INI     Path to user ztp ini file
  --label LABEL, -l LABEL
                        Golden ISO Label
  --out-directory OUT_DIRECTORY
                        Output Directory
  --yamlfile CLI_YAML   Cli arguments via yaml
  --clean               Delete output dir before proceeding
  --pkglist PKGLIST [PKGLIST ...]
                        Packages to be added to the output GISO. For eXR: optional rpm
                        or smu to package. For LNT: either full package filenames or
                        package names for user installable packages can be specified.
```

```
                        Full package filenames can be specified to choose a particular
                        version of a package, the rest of the block that the package is
                        in will be included as well. Package names can be specified to
                        include optional packages in the output GISO.
  --docker, --use-container
                        Build GISO in container environment.Pulls and run pre-built
                        container image to build GISO.
  --version             Print version of this script and exit

EXR only build options:
  --script SCRIPT       Path to user executable script executed as part of bootup post
                        activate.
  --x86-only            Use only x86_64 rpms even if other architectures are
                        applicable.
  --migration           To build Migration tar only for ASR9k

LNT only build options:
  --remove-packages REMOVE_PACKAGES [REMOVE_PACKAGES ...]
                        Remove RPMs, specified in a comma separated list. These are are
                        matched against user installable package names, and must be the
                        whole package name, e.g: xr-bgp
  --skip-usb-image      Do not build the USB image
  --copy-dir COPY_DIRECTORY
                        Copy built artefacts to specified directory if provided. The
                        specified directory must already exist, be writable by the
                        builder and must not contain a previously built artefact with
                        the same name.
  --clear-bridging-fixes
                        Remove all bridging bugfixes from the input ISO
  --verbose-dep-check   Verbose output for the dependency check.
  --debug               Output debug logs to console
```

Example

Example: Build Docker-Based GISO Image

In this example, a GISO image is built using docker.

```
[root@xr src]# ./gisobuild.py --docker --iso /auto/asr9kgiso/asr9k-mini-x-7.5.1.iso
--repo /auto/asr9kgiso --pkglist asr9k-7.5.1.CSCsb88888 asr9k-bgp-2.0.0.0-r751.x86_64.rpm
asr9k-eigrp-1.0.0.0-r751.x86_64.rpm asr9k-isis-2.1.0.0-r751.x86_64.rpm
asr9k-k9sec-3.1.0.0-r751.x86_64.rpm
asr9k-li-1.0.0.0-r751.x86_64.rpm asr9k-mcast-3.0.0.0-r751.x86_64.rpm
asr9k-mgbl-3.0.0.0-r751.x86_64.rpm
asr9k-mpls-2.1.0.0-r751.x86_64.rpm asr9k-mpls-te-rsvp-3.1.0.0-r751.x86_64.rpm
asr9k-ospf-2.0.0.0-r751.x86_64.rpm
asr9k-parser-2.0.0.0-r751.x86_64.rpm --label dockerbasedgiso

Local System requirements check [PASS]
Pulling gisobuild image from hub. Please wait...
\
Done...
System requirements check [PASS]

Platform: asr9000 Version: 7.5.1

Scanning repository [/auto/asr9000giso]...

Building RPM Database...

Total 11 RPM(s) present in the repository path provided in CLI
```

```
[ 1] asr9k-mpls-2.1.0.0-r751.x86_64.rpm
[ 2] asr9k-mgbl-3.0.0.0-r751.x86_64.rpm
[ 3] asr9k-bgp-2.0.0.0-r751.x86_64.rpm
[ 4] asr9k-parser-2.0.0.0-r751.x86_64.rpm
[ 5] asr9k-isis-2.1.0.0-r751.x86_64.rpm
[ 6] asr9k-mcast-3.0.0.0-r751.x86_64.rpm
[ 7] asr9k-mpls-te-rsvp-3.1.0.0-r751.x86_64.rpm
[ 8] asr9k-ospf-2.0.0.0-r751.x86_64.rpm
[ 9] asr9k-li-1.0.0.0-r751.x86_64.rpm
[10] asr9k-eigrp-1.0.0.0-r751.x86_64.rpm
[11] asr9k-k9sec-3.1.0.0-r751.x86_64.rpm
Following XR x86_64 rpm(s) will be used for building Golden ISO:

        (+) asr9k-ospf-2.0.0.0-r751.x86_64.rpm
        (+) asr9k-bgp-2.0.0.0-r751.x86_64.rpm
        (+) asr9k-parser-2.0.0.0-r751.x86_64.rpm
        (+) asr9k-mcast-3.0.0.0-r751.x86_64.rpm
        (+) asr9k-li-1.0.0.0-r751.x86_64.rpm
        (+) asr9k-eigrp-1.0.0.0-r751.x86_64.rpm
        (+) asr9k-mgbl-3.0.0.0-r751.x86_64.rpm
        (+) asr9k-mpls-2.1.0.0-r75114I.x86_64.rpm
        (+) asr9k-mpls-te-rsvp-3.1.0.0-r751.x86_64.rpm
        (+) asr9k-isis-2.1.0.0-r751.x86_64.rpm
        (+) asr9k-k9sec-3.1.0.0-r751.x86_64.rpm

        ...RPM signature check [PASS]

Skipping following rpms from repository since they are already present in base ISO:

        (-) asr9k-parser-2.0.0.0-r751.x86_64.rpm
        (-) asr9k-bgp-2.0.0.0-r751.x86_64.rpm

        ...RPM compatibility check [PASS]

Building Golden ISO...
Summary .....

XR rpms:
        asr9k-mcast-3.0.0.0-r751.x86_64.rpm
        asr9k-mgbl-3.0.0.0-r751.x86_64.rpm
        asr9k-isis-2.1.0.0-r751.x86_64.rpm
        asr9k-mpls-te-rsvp-3.1.0.0-r751.x86_64.rpm
        asr9k-eigrp-1.0.0.0-r751.x86_64.rpm
        asr9k-mpls-2.1.0.0-r751.x86_64.rpm
        asr9k-ospf-2.0.0.0-r751.x86_64.rpm
        asr9k-li-1.0.0.0-r751.x86_64.rpm
        asr9k-k9sec-3.1.0.0-r751.x86_64.rpm

        ...Golden ISO creation SUCCESS.

Golden ISO Image Location: /var/tmp/giso/gisobuild-toolkit-master/src/output_gisobuild/
                           asr9k-golden-x-7.5.1-dockerbasedgiso.iso
```

View that the GISO file is created succesfully.

```
[root@xr src]# ls
exrmod  gisobuild.py  lntmod  output_gisobuild  utils

[root@xr src]# cd output_gisobuild/
[root@xr output_gisobuild]# ls
img_built_name.txt logs asr9k-golden-x-7.5.1-dockerbasedgiso.iso
rpms_packaged_in_giso.txt
```

Example: Build YAML-Based GISO Image

YAML is a markup file that serves as a template to provide the package list and manage the build options.

The following example shows a sample YAML template:

```
# Options below correspond to the tool input options.
# --iso ISO             Path to Mini.iso/golden.iso file
# --repo REPO [REPO ...]
#                       Path to list of RPM repositories. RPMs are only used if already
#                       included in the ISO, or specified by the user via the --pkglist
option.
# --pkglist PKGLIST [PKGLIST ...]
#                       Optional list of rpm or smu to add to the ISO.
# --remove-packages REMOVE_PACKAGES [REMOVE_PACKAGES ...]
#                      Remove named RPMs, specified in a space separated list. Valid build
#                       option for LNT only. eXR builds simply ignores this option.
# --bridging-fixes BRIDGE_FIXES [BRIDGE_FIXES ...]
#                       Bridging rpms to package. Takes from-release (supported for eXR)
#                       or rpm names.
# --xrconfig XRCONFIG   Path to XR config file
# --ztp-ini ZTP_INI     Path to user ztp ini file
# --script SCRIPT       Path to user executable script executed as part of
#                       bootup post activate. Valid build option for eXR only.
#                       LNT builds simply ignores.
# --label LABEL         Golden ISO Label
# --out-directory OUT_DIRECTORY
#                       Output Directory. Built GISO and logs will be available post
gisobuild.
# --copy-directory COPY_DIRECTORY
#                      Copy built artefacts to specified directory if provided. Valid build
#                       option for LNT only. eXR build ignores this option.
# --yamlfile CLI_YAML   Cli arguments via yaml.
# --clean               Delete output dir before proceeding.
# --migration           To build Migration tar only for ASR9k. Valid build option for eXR
only.
#                       LNT builds simply ignore this option.
# --docker              Load and run pre-built docker image. Valid build option for eXR
only.
#                       LNT builds simply ignore this option.
# --x86-only            Use only x86_64 rpms even if other architectures are applicable.
Valid build
#                       option for eXR only. LNT builds simply ignore this option.
# --version             Print version of this script and exit

packages:
    iso: <path-to-iso>
    repo:
        - <path-to-repo1>
        - <path-to-repo2>
    pkglist:
        - <pkg1>
        - <pkg2>
    bridge-fixes:
        upgrade-from-release:
            - <dotted-release-1>
            - <dotted-release-2>
        rpms:
            - <pkg1>
            - <pkg2>
    remove_packages:
        - <pkg1>
        - <pkg2>

user-content:
```

```
    script: <path-to-script-sh>
    xrconfig: <path-to-router.cfg>
    ztp-ini: <path-to-ztp.ini>

output:
    label: <giso-label>
    out-directory: <path-to-output-directory>
    clean: <true/false>

options:
    docker: <true/false>
    migration: <true/false>
    x86-only: <true/false>
```

In this example, you configure a YAML file with the required files:

```
packages:
    iso: /auto/751_repo/asr9k-mini-x-7.5.1.iso
    repo:
        - /auto/751_repo/
    pkglist:
        - asr9k-bgp-2.0.0.0-r751.x86_64.rpm
        - asr9k-eigrp-1.0.0.0-r751.x86_64.rpm
        - asr9k-isis-2.1.0.0-r751.x86_64.rpm
        - asr9k-li-1.0.0.0-r751.x86_64.rpm
        - asr9k-mcast-3.0.0.0-r751.x86_64.rpm
        - asr9k-mgbl-3.0.0.0-r751.x86_64.rpm
        - asr9k-mpls-2.1.0.0-r751.x86_64.rpm
        - asr9k-mpls-te-rsvp-3.1.0.0-r751.x86_64.rpm
        - asr9k-ospf-2.0.0.0-r751.x86_64.rpm
        - asr9k-parser-2.0.0.0-r751.x86_64.rpm
        - asr9k-k9sec-3.1.0.0-r751.x86_64.rpm
        - asr9k-mcast-3.0.0.1-r751.CSCxr33333.x86_64.rpm
        - asr9k-os-5.0.0.1-r751.CSCxr11111.x86_64.rpm
        - asr9k-sysadmin-hostos-7.5.1-r751.CSCho99999.admin.x86_64.rpm
        - asr9k-sysadmin-hostos-7.5.1-r751.CSCho99999.host.x86_64.rpm
        - asr9k-sysadmin-topo-7.5.1-r751.CSCcv55555.x86_64.rpm
        - asr9k-sysadmin-system-7.5.1-r751.CSCcv44444.x86_64.rpm
        - openssh-scp-6.6p1.p1-r0.5.0.r751.CSCtp11111.xr.x86_64.rpm
        - cisco-klm-zermatt-0.1.p1-r0.0.r751.CSCtp11111.xr.x86_64.rpm

    remove_rpms: []

user-content:
    script: script.sh
    xrconfig: /auto/751_repo/gisoxrconfig.cfg
    ztp-ini:  /auto/751_repo/ztp.ini

output:
    label: 751_yaml_install
    out-directory: /auto/751_repo/
    clean: true

options:
    docker: false
    full-iso: false
    migration: false
    x86-only: false
```

If you do not want to specify the list of packages and parameters via CLI, you can use the YAML file template.

```
[directory-path]$ ./src/gisobuild.py --yamlfile <input-yaml-cfg>
```

To override any input in the YAML configuration file, use the corresponding CLI options.

```
[directory-path]$ ./src/gisobuild.py --yamlfile <input-yaml-cfg> --label <new-label>
```

This new label overrides the label specified in the YAML file.

When the host machine does not have its package dependencies met, but allows pulling and running docker images, enable the docker option in YAML file to `true` and run the command:

```
[directory-path]$ ./src/gisobuild.py --yamlfile <input-yaml-cfg>
```

where, the `input-yaml-cfg` has the docker option set to `true`.

What to do next

Install the GISO image on the router.

Install Golden ISO

Golden ISO (GISO) automatically performs the following actions:

- Installs host and system admin RPMs.
- Partitions repository and TFTP boot on RP.
- Creates software profile in system admin and XR modes.
- Installs XR RPMs. Use **show install active** command to see the list of RPMs.
- Applies XR configuration. Use **show running-config** command in XR mode to verify.

Procedure

Step 1 Download GISO image to the router using one of the following options:

- **PXE boot:** when the router is booted, the boot mode is identified. After detecting PXE as boot mode, all available ethernet interfaces are brought up, and DHClient is run on each interface. DHClient script parses HTTP or TFTP protocol, and GISO is downloaded to the box.

  When you bring up a router using the PXE boot mode, existing configurations are removed. To recover smart licensing configurations like Permanent License Reservation (PLR), enable these configurations after the router comes up.

  ```
  Router#configure
  Router(config)#license smart reservation
  Router(config)#commit
  ```

  The following example shows the logs from installation of GISO using PXE boot:

  ```
  ...
  Fri Dec  2 19:18:03 UTC 2016: ---Starting to prepare host logical volume---
  ...
  Fri Dec 02 19:18:14 UTC 2016: Skipping tp base rpm(openssh-scp-6.6p1-r0.0.host.x86_64.rpm) from
   installation
  Fri Dec 02 19:18:14 UTC 2016: Skipping tp base rpm(kernel-modules-3.14-r0.1.host.x86_64.rpm) from
   installation
  Fri Dec 02 19:18:15 UTC 2016: Installing asr9k-sysadmin-hostos-6.1.3-r613.CSChu77777.host.x86_64
            [SUCCESS]
  ```

```
...

Fri Dec  2 19:18:23 UTC 2016: ---Starting to prepare calvados logical volume---

...

Fri Dec 02 19:18:48 UTC 2016: Skipping tp base rpm(openssh-scp-6.6p1-r0.0.admin.x86_64.rpm) from
 installation
Fri Dec 02 19:18:48 UTC 2016: Skipping tp base rpm(kernel-modules-3.14-r0.1.admin.x86_64.rpm)
from installation
Fri Dec 02 19:18:49 UTC 2016: Installing asr9k-sysadmin-system-6.1.3-r613.CSCcv44444.x86_64
            [SUCCESS]
Fri Dec 02 19:18:50 UTC 2016: Installing asr9k-sysadmin-shared-6.1.3-r613.CSCcv33333.x86_64
            [SUCCESS]
Fri Dec 02 19:18:51 UTC 2016: Installing asr9k-sysadmin-hostos-6.1.3-r613.CSChu77777.admin.x86_64
          [SUCCESS]

...

Fri Dec  2 19:19:07 UTC 2016: ---Starting to prepare repository---
Fri Dec  2 19:19:11 UTC 2016: File system creation on /test took 3 seconds
Fri Dec  2 19:19:11 UTC 2016: Copying /iso/host.iso to repository /iso directory
Fri Dec  2 19:19:11 UTC 2016: Copy Host rpms to repository
Fri Dec  2 19:19:13 UTC 2016: Copying /iso/asr9k-sysadmin.iso to repository /iso directory
Fri Dec  2 19:19:13 UTC 2016: Copy Sysadmin rpms to repository
Fri Dec  2 19:19:16 UTC 2016: Copy HostOs rpms to repository
Fri Dec  2 19:19:16 UTC 2016: Copy XR rpms to repository
Fri Dec  2 19:19:16 UTC 2016: Copy giso_info.txt to repository
Fri Dec  2 19:19:17 UTC 2016: Copying /iso/asr9k-xr.iso to repository /iso directory
Fri Dec  2 19:19:21 UTC 2016: Copying all ISOs to repository took 10 seconds

...
```

- **USB boot or Disk Boot:** when the USB mode is detected during boot, and GISO is identified, the additional RPMs and XR configuration files are extracted and installed.
- **System Upgrade:** when the system is upgraded, GISO can be installed using **install add**, **install activate**, or using **install replace** commands.

Important To replace the current version and packages on the router with the version from GISO, note the change in command and format.

- In versions prior to Cisco IOS XR Release 6.3.3, 6.4.x and 6.5.1, use the **install update** command:

  ```
  install update source <source path> <Golden-ISO-name> replace
  ```

- In Cisco IOS XR Release 6.5.2 and later, use the **install replace** command.

  ```
  install replace <absolute-path-of-Golden-ISO>
  ```

Note To create a Bootable External USB Disk, do the following:

- Ensure that the USB Boot Disk has a minimum storage of 8GB, and that you have root/admin or appropriate permission to create bootable disk on linux machine.

a. Copy and execute usb-install script on the Linux machine to create a bootable external USB.

```
Router#admin

sysadmin-vm:0_RSP0#run chvrf 0 ssh rp0_admin
[sysadmin-vm:0_RSP0:~]$ssh my_host
[host:~]$cd /misc/disk1/
[host:~]$./usb-install-712-or-latest.sh asr9k-goldenk9-x64-7.0.2-dr.iso /dev/sdc EFI

Preparing USB stick for EFI
parted gpt: Failed to create partition - continuing ...
Create filesystem on /dev/sdc1
Mounting source iso at //misc/disk1/cdtmp.CnuKnA
Mounting destination /dev/sdc1 at //misc/disk1/usbdev.SSBb4R
Copying image to USB stick
Initrd path is //misc/disk1/cdtmp.CnuKnA/boot/initrd.img
Getting boot
3749342 blocks
Copying boot
Copying initrd.img
Copying signature.initrd.img
Copying certs
Creating grub files
Copying /misc/disk1/asr9k-goldenk9-x64-7.0.2-dr.iso in USB Stick
USB stick set up for EFI boot!
```

b. Reset the RSP/RP and plug in bootable USB to RSP/RP's front panel. The USB will get detected in ROMMON. Note that when the system is in ROMMON, and if you add a front panel external USB, the USB will not be detected until the RSP/RP is reset.

The options to upgrade the system are as follows:

- **system upgrade from a non-GISO (image that does not support GISO) to GISO image:** If a system is running a version1 with an image that does not support GISO, the system cannot be upgraded directly to version2 of an image that supports GISO. Instead, the version1 must be upgraded to version2 mini ISO, and then to version2 GISO.
- **system upgrade in a release from version1 GISO to version2 GISO:** If both the GISO images have the same base version but different labels, **install add** and **install activate** commands does not support same version of two images. Instead, using **install source** command installs only the delta RPMs. System reload is based on restart type of the delta RPMs.

  Using **install replace** command performs a system reload, irrespective of the difference between ISO and the existing version.
- **system upgrade across releases from version1 GISO to version2 GISO:** Both the GISO images have different base versions. Use **install add** and **install activate** commands, or **install replace** command to perform the system upgrade. The router reloads after the upgrade with the version2 GISO image.

Step 2 Run the **show install repository all** command in System Admin mode to view the RPMs and base ISO for host, system admin and XR.

```
sysadmin-vm:0_RP0#show install repository all
Admin repository
---------------------
 asr9k-sysadmin-6.1.1

 asr9k-sysadmin-hostos-6.1.1-r611.CSCcv10001.admin.x86_64
 asr9k-sysadmin-system-6.1.1-r611.CSCcv10005.x86_64
....

  XR repository
------------------
 asr9k-iosxr-mgbl-3.0.0.0-r611.x86_64
 asr9k-xr-6.1.1
....

  Host repository
---------------------
 host-6.1.1
```

Step 3 Run the **show install package <golden-iso>** command to display the list of RPMs, and packages built in GISO.

Note To list RPMs in the GISO, the GISO must be present in the install repository.

```
Router#show install package asr9k-goldenk9-x64-6.1.3

Sun Dec  4 13:52:48.279 UTC
This may take a while ...
    ISO Name: asr9k-goldenk9-x64-6.1.3
    ISO Type: bundle
    ISO Bundled: asr9k-mini-x64-6.1.3
    Golden ISO Label: temp
    ISO Contents:
        ISO Name: asr9k-xr-6.1.3
        ISO Type: xr
        rpms in xr ISO:
            iosxr-os-asr9k-64-5.0.0.0-r613
            iosxr-ce-asr9k-64-3.0.0.0-r613
            iosxr-infra-asr9k-64-4.0.0.0-r613
            iosxr-fwding-asr9k-64-4.0.0.0-r613
            iosxr-routing-asr9k-64-3.1.0.0-r613
        ...

        ISO Name: asr9k-sysadmin-6.1.3
        ISO Type: sysadmin
        rpms in sysadmin ISO:
            asr9k-sysadmin-topo-6.1.3-r613
            asr9k-sysadmin-shared-6.1.3-r613
            asr9k-sysadmin-system-6.1.3-r613
            asr9k-sysadmin-hostos-6.1.3-r613.admin
        ...

        ISO Name: host-6.1.3
        ISO Type: host
        rpms in host ISO:
            asr9k-sysadmin-hostos-6.1.3-r613.host

        Golden ISO Rpms:
            xr rpms in golden ISO:
                asr9k-k9sec-x64-2.2.0.1-r613.CSCxr33333.x86_64.rpm
                openssh-scp-6.6p1.p1-r0.0.CSCtp12345.xr.x86_64.rpm
                openssh-scp-6.6p1-r0.0.xr.x86_64.rpm
                asr9k-mpls-x64-2.1.0.0-r613.x86_64.rpm
                asr9k-k9sec-x64-2.2.0.0-r613.x86_64.rpm

            sysadmin rpms in golden ISO:
```

```
    asr9k-sysadmin-system-6.1.3-r613.CSCcv11111.x86_64.rpm
    openssh-scp-6.6p1-r0.0.admin.x86_64.rpm
    openssh-scp-6.6p1.p1-r0.0.CSCtp12345.admin.x86_64.rpm

host rpms in golden ISO:
    openssh-scp-6.6p1-r0.0.host.x86_64.rpm
    openssh-scp-6.6p1.p1-r0.0.CSCtp12345.host.x86_64.rpm
```

The ISO, SMUs and packages in GISO are installed on the router.