Introduction to Flexible Packaging

Flexible packaging is a method of breaking down the Cisco IOS XR operating system into modules and providing them as RPMs (packages). Delivering packages as RPMs enables easier and faster system updates.

The lean base software contains only the required packages, while the optional packages are provided separately as installable RPMs. You can select and install the services you want by selecting the required RPMs.

Flexible packaging feature also supports automatic dependency management, whereby, while the user is updating an RPM, the system automatically identifies all relevant dependent packages and updates them. The system uses standard LINUX tools to manage dependency during upgrades.

The base software is the minimum mandatory package (with utilities), required for the basic functioning of the router. This is also called the mini.iso file. This base package contains:

• Operating system (OS)—Kernel, file system, memory management, and other OS utilities
• Base components—Interface manager, system database, checkpoint services, configuration management utilities
• Infrastructure components—rack management, fabric management
• Routing protocols—mandatory routing protocols (such as BGP)
• Forwarding components—FIB, ARP, QoS, ACL
• Line card drivers

Mandatory RPMs (such as, BGP) which are a part of the base software, cannot be removed and can only be upgraded. Optional RPMs such as, EIGRP can be added, upgraded and removed as required.

Figure 1: Granular routing modules

This section also describes the following:
Workflow for Flexible Packaging

This image shows the overall workflow for Flexible Packaging.

Figure 2: Flexible Packaging Workflow

Packaging Filename Format

The format of an RPM is: **name-version-release.architecture.rpm** where,

- **name** - of the platform the software supports
- **version** - the version of the software
- **release** - the number of times this version of the software has been delivered
- **architecture** - the node's processor architecture

Consider the following example:

```
asr9k-mpls-1.0.2.0-r611.x86_64.rpm
```

Platform-Package Name : asr9k-mpls
Version : 1.0.2.0
Release : r611
Architecture : x86_64

Software Maintenance Upgrades (SMUs) are delivered as RPMs. RPMs have a four-digit version number. The first three digits represent major, minor, and build numbers respectively. The fourth digit is incremented with each SMU release.
This table lists the reasons when each digit of the version gets incremented.

<table>
<thead>
<tr>
<th>Version (Digit from left)</th>
<th>Indicates</th>
<th>Incremented When</th>
</tr>
</thead>
<tbody>
<tr>
<td>First</td>
<td>Major</td>
<td>non-backward compatible API(s) change(s)</td>
</tr>
<tr>
<td>Second</td>
<td>Minor</td>
<td>a backward compatible change occurs to a public API</td>
</tr>
<tr>
<td>Third</td>
<td>Build</td>
<td>an RPM is built without any API change</td>
</tr>
<tr>
<td>Fourth</td>
<td>SMU Release</td>
<td>a new SMU is released</td>
</tr>
</tbody>
</table>

**Defect ID**

SMUs are identified with a defect-ID. In this example, note that, for the first SMU release of the package, the fourth digit starts at 1 and for the second SMU release of the package, the fourth digit is incremented to 2.

First SMU of the mpls package: asr9k-mpls-1.0.2.1-r611.CSCub12345.x86_64.rpm
Second SMU of the mpls package: asr9k-mpls-1.0.2.2-r611.CSCub12345.x86_64.rpm