

How to Keep Moxa IIoT Gateway Software Up-to-date and Free of Vulnerabilities

Moxa Technical Support Team
support@moxa.com

Contents

- 1 The Importance of Keeping IIoT Gateways Up-to-date..... 2**
- 2 APT Repository 3**
- 3 Updating the Entire System 3**
 - 3.1 Preparing a Staging Environment 3
 - 3.2 Synchronizing With Repositories..... 3
 - 3.3 Updating Your IIoT Gateway Base-system 4
 - 3.4 Updating the Entire System 4
- 4 Setting Up an Onsite Repository 5**
- 5 Other Common and Useful APT Commands 5**
 - 5.1 List all available packages 5
 - 5.2 Check dependencies for specific packages 5
 - 5.3 Install or upgrade specific packages 5
 - 5.4 Install multiple packages..... 5
 - 5.5 Install packages without upgrading 5
 - 5.6 Upgrade only specific packages..... 6
 - 5.7 Specify which version of a package to install..... 6
 - 5.8 Remove an installed package, but not the configuration 6
 - 5.9 Completely remove a package, including the configuration 6
 - 5.10 Clean up disk space..... 6

Copyright © 2021 Moxa Inc.

Released on June 25, 2021

About Moxa

Moxa is a leading provider of edge connectivity, industrial computing, and network infrastructure solutions for enabling connectivity for the Industrial Internet of Things (IIoT). With over 30 years of industry experience, Moxa has connected more than 57 million devices worldwide and has a distribution and service network that reaches customers in more than 70 countries. Moxa delivers lasting business value by empowering industries with reliable networks and sincere service. Information about Moxa’s solutions is available at www.moxa.com.

How to Contact Moxa

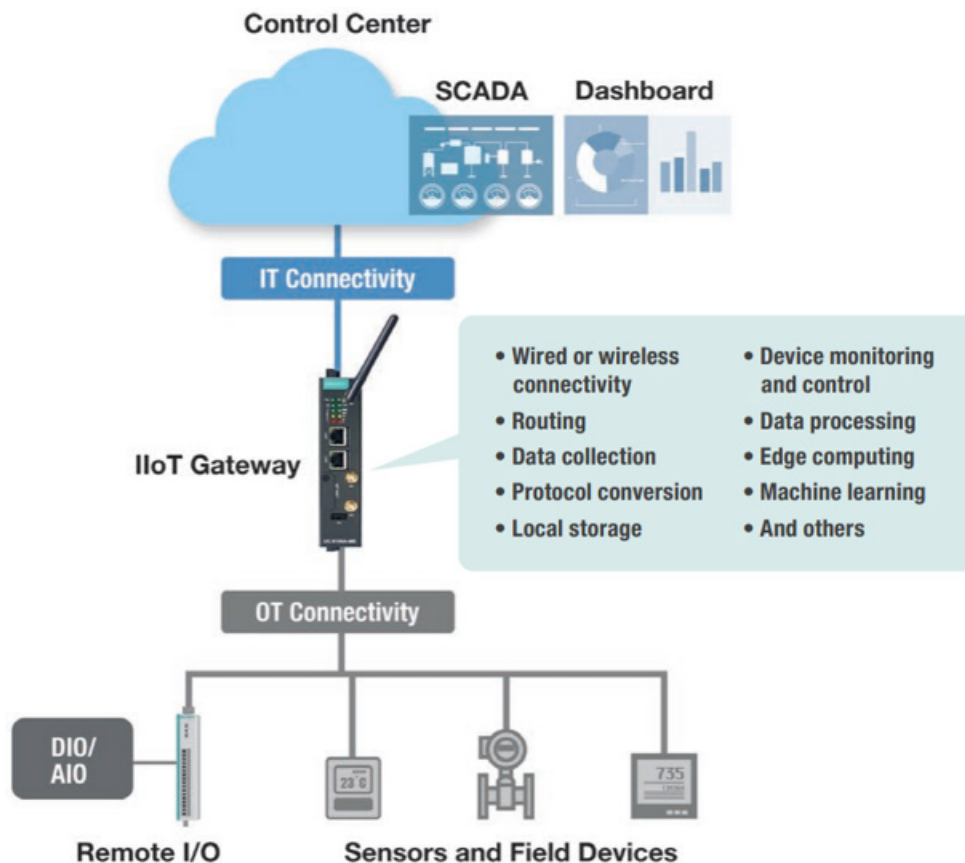
Tel: +886-2-8919-1230
Fax: +886-2-8919-1231



1 The Importance of Keeping IIoT Gateways Up-to-date

IIoT gateways act as a gate or bridge between OT devices and the cloud in IoT and IIoT applications. Since they often need to process the bulk of ingress and egress data, they are more exposed to cybersecurity threats than other devices. For this reason, it is extremely important to keep your IIoT gateway software up-to-date to ensure sustainability, performance, and security.

Moxa IIoT gateways are designed for long-term operation. The gateways come with a rich variety of Moxa proprietary utilities, libraries, and tools to accelerate your IIoT application development. Gateway models that use the Moxa Industrial Linux (Debian 9 based) operating system have a 10-year software life-cycle, which offers five more years of fixes for both bugs and security vulnerabilities than the standard Debian OS.



In this section, we explain how you can quickly update all of the software packages in just a few steps using the **Advanced Package Tool (APT)** in the Debian OS pre-installed in the Moxa IIoT gateway. We also cover common functionalities of APT commands that can be used to install, remove, update, and search for specific software packages.

2 APT Repository

The APT Repository is the network server from which APT downloads packages that are installed on your IIoT gateway. The repository source list can be found here:

`/etc/apt/sources.list`

By default, Moxa IIoT gateways include the following repositories that contain stable and well-tested packages best suited for ensuring the stability of your IIoT project.

Repository URL	Description
<code>http://deb.debian.org/debian stretch</code>	Debian official repository containing the latest stable Debian 9 release (released about every 2 months)
<code>http://deb.debian.org/debian stretch-updates</code>	Debian official repository containing bug fixes that will be included in the upcoming Debian 9 release
<code>http://security.debian.org/ stretch/updates</code>	Debian official repository containing security hotfixes that will be included in the upcoming Debian 9 release
<code>mirror://debian.moxa.com/debian/mirrors stretch</code>	Moxa repository containing Moxa's proprietary library, tools, utilities, and kernel. Moxa will maintain security and bug fixes even after Debian 9 is EOL

3 Updating the Entire System

3.1 Preparing a Staging Environment

Since the Moxa IIoT gateway operating system is an open platform, you are free to install any software that you would like to use. However, we highly recommend that you test new software on a staging platform first before installing the software on your production gateways.

3.2 Synchronizing With Repositories

The first and most important step is to synchronize the package index files in your IIoT gateway with the source repositories specified in the file **`/etc/apt/sources.list`**. When you perform the synchronization, information related to the packages, including versions and dependencies, will also be downloaded from the repositories.

To perform the synchronization, make sure that your network environment can connect to the APT repositories, and then run the **`apt-get update`** command with root permission to synchronize the package index.

```
root@moxa:~# apt-get update
```

3.3 Updating Your IIoT Gateway Base-system

Moxa regularly releases updated firmware versions that include new libraries, kernels, tools, utilities, and other items, to improve the functionality of the IIoT gateways. To ensure that your IIoT gateway will also get any new packages via APT, you first need to install the new firmware base-system package.

1. Use the **apt-cache search base-system** command to display all available Moxa base-system packages.

```
root@Moxa:~# apt-cache search base-system

uc2100-base-system - Base system for UC-2100
uc3100-base-system - Base system for UC-3100
uc5100-base-system - Base system for UC-5100
uc8100-base-system - Base system for UC-8100
uc8100a-me-base-system - Base system for UC-8100A-ME
uc8100me-base-system - Base system for UC-8100-ME
uc8200-base-system - Base system for UC-8200
```

2. Use the following commands to check your IIoT gateway model name and the corresponding base-system package name.

```
# mx-ver -m
# dpkg -l | grep base-system
```

```
root@Moxa:~# mx-ver -m
UC-3111-T-EU-LX

root@Moxa:~# dpkg -l | grep base-system
ii uc3100-base-system          1.10.4          armhf
Base system for UC-3100
```

3. Use the command **apt-get install <product>-base-system** to install the base-system.

```
root@Moxa:~# apt-get install uc3100-base-system
```

4. Reboot your IIoT gateway after performing the base-system installation.

```
root@Moxa:~# reboot
```

3.4 Updating the Entire System

Use the **apt-get upgrade** command to upgrade latest versions of all the packages used by your IIoT gateway.

```
root@Moxa:~$ apt-get upgrade
```

4 Setting Up an Onsite Repository

If your IIoT gateway network environment does not allow connecting to the public repositories defined in `/etc/apt/sources.list`, or network bandwidth is a problem, you may want to set up and maintain a repository of your own and add the URL to source list. Ample information and detailed instructions on how to do this can be found on the Internet.

5 Other Common and Useful APT Commands

In this section, we briefly describe some useful APT commands that you may want to use.

5.1 List all available packages

To get more detailed package information (e.g., version number, check sums, size, installed size, and category), along with a short description of the package, use the **show** sub command, which is illustrated below for the "vim" package:

```
root@Moxa:~# apt-cache show vim
```

5.2 Check dependencies for specific packages

Use the **showpkg** subcommand to check the dependencies for particular software packages, and whether those dependent packages are installed or not. For example, use the **showpkg** command along with the package-name as follows:

```
root@Moxa:~# apt-cache showpkg vim
```

5.3 Install or upgrade specific packages

Use the **install** sub command to install or upgrade a package.

```
root@Moxa:~# apt-get install vim
```

5.4 Install multiple packages

You can add more than one package name along with the install command to install multiple packages at the same time. For example, the following command will install the packages **vim** and **goaccess**.

```
root@Moxa:~# apt-get install vim goaccess
```

5.5 Install packages without upgrading

Use the **--no-upgrade** subcommand to prevent packages that are already installed from getting upgraded.

```
root@Moxa:~# apt-get install packageName --no-upgrade
```

5.6 Upgrade only specific packages

Use the **--only-upgrade** subcommand if you only want to upgrade an already installed package. If the package is not installed, then the package will not be installed.

```
root@Moxa:~# apt-get install packageName --only-upgrade
```

5.7 Specify which version of a package to install

By default, the latest version available in the repository will be installed. If you don't want to install the latest version, you can specify the package version number.

```
root@Moxa:~# apt-get install packageName =version_number
```

5.8 Remove an installed package, but not the configuration

Use the **remove** subcommand to remove the binaries of a package, but keep the configuration file for later reuse.

```
root@Moxa:~# apt-get remove packageName
```

5.9 Completely remove a package, including the configuration

Use the **purge** subcommand to completely remove a package, including the configuration files.

```
root@Moxa:~# sudo apt-get remove --purge packageName
```

5.10 Clean up disk space

Use the **clean** sub command to clean the downloaded package(.deb) file mentioned above to free up some disk space.

```
root@Moxa:~# apt-get clean
```