

Application Note:

Using the EDS5000(8/16) SDK

Intellectual Property

© 2025 Lantronix, Inc. All rights reserved. No part of the contents of this publication may be transmitted or reproduced in any form or by any means without the written permission of Lantronix.

Lantronix is a registered trademark of Lantronix, Inc. in the United States and other countries. Patented: <http://patents.lantronix.com>; additional patents pending.

All trademarks and trade names are the property of their respective holders.

Contacts

Lantronix, Inc.
48 Discovery, Suite 250
Irvine, CA 92618, USA
Toll Free: 800-526-8766
Phone: 949-453-3990
Fax: 949-453-3995

Technical Support

Online: www.lantronix.com/support

Sales Offices

For a current list of our domestic and international sales offices, go to the Lantronix web site at www.lantronix.com/about/contact

Disclaimer

All information contained herein is provided “AS IS.” Lantronix undertakes no obligation to update the information in this publication. Lantronix does not make, and specifically disclaims, all warranties of any kind (express, implied or otherwise) regarding title, non-infringement, fitness, quality, accuracy, completeness, usefulness, suitability or performance of the information provided herein. Lantronix shall have no liability whatsoever to any user for any damages, losses and causes of action (whether in contract or in tort or otherwise) in connection with the user’s access or usage of any of the information or content contained herein. The information and specifications contained in this document are subject to change without notice.

Revision History

Date	Rev.	Comments
September 2025	A	Initial release.

For the latest revision of this product document, please check our online documentation at www.lantronix.com/support/documentation.

Contents

1. Introduction	4
Additional Documentation	4
Download the SDK Files	4
OS Requirements and Prerequisites.....	4
2. Creating a Custom Package.....	5
Sample Application	5
Extract the SDK.....	5
Add Your Package Feeds	5
Make the Custom Package.....	6
Integrate the Custom Package into an Image	6
3. Building an Image using Image Builder.....	8
Extract Image Builder	8
Usage	8
Select the Profile	8
Select Packages.....	8
Included Packages	8
Excluded Packages.....	9
Including Custom Packages	9
Add Custom Files.....	9
Disable Services.....	9
Make the Image	9
UCI Defaults	10
Device Configuration	11
4. Customizing the UI Using Image Builder.....	12
Firmware File Name and Configuration Version	12
Customize UI Elements	12

1. Introduction

This document describes how to use the Lantronix EDS5000(8/16) SDK and Image Builder to create custom packages, build custom firmware images and customize UI elements.

Note: *The SDK applies to the 8-port and 16-port models only. It does not apply to the 32-port model.*

This document assumes knowledge of OpenWrt build system and familiarity with the EDS5000 web interface for configuring EDS5000 devices. It requires Linux distribution or VM.

The command examples use the Bash command interpreter in a development environment.

Additional Documentation

The following documentation resources provide background and reference information.

1. [OpenWrt online documentation](#) (refer to developer guide)
2. Lantronix EDS5000 Device Server User Guide

Download the SDK Files

To download the Lantronix EDS5000(8/16) SDK, contact [Lantronix Technical Support](#). The SDK includes the following components:

1. Lantronix EDS5000(8/16) SDK
2. Lantronix Image Builder

The SDK is a pre-compiled environment suitable for creating custom packages to install on the device or build into the firmware. It's provided as a tar.xz archive.

The Lantronix Image Builder is a pre-compiled environment suitable for creating custom images for Lantronix EDS5000(8/16) devices. It's also provided as a tar.xz archive.

OS Requirements and Prerequisites

The SDK and Image Builder run in 64-bit Linux. It is recommended that you use a Linux environment, standalone or virtual machine.

The prerequisites are the same as for the OpenWrt build system. Please refer to the [OpenWrt Developer guide](#) for build system setup, build prerequisites in different Linux distributions, and package prerequisites.

2. Creating a Custom Package

Use the SDK to create a custom package. The custom package can then be included into a firmware image using Image Builder, or alternatively, it can be installed using the OPKG package manager.

Sample Application

You will need a sample application to create the custom package. If you don't already have an application, you can create a simple "Hello, world" application with a structure like the following:

```
<packagename>/  
|  
+-- Makefile  
+-- src/  
    |  
    +-- source and include files
```

Note: *<packagename> must not contain spaces.*

Extract the SDK

To extract the SDK archive, use the following command.

```
tar -xvf <sdk-filename.tar.xz>
```

replacing `sdk-filename` with the actual file name of the tar archive.

Add Your Package Feeds

The package feeds are in the `feeds.conf.default` file of the extracted SDK directory. You should modify the package feeds if the following cases apply:

- If the package requires additional dependency packages supplied by OpenWrt, add the repository or change to the proper repository.
- If the package requires additional dependency packages supplied by your own (custom) repository, add your own repository.

Make the Custom Package

Note: All Linux commands should be run as user. Do not use root or sudo.

1. In the package/ directory, create a directory for the package. The directory name must include no spaces. For this example, we'll call it "packagename".
2. Navigate to the "packagename" directory and create or copy a Makefile.
3. In the "packagename" directory, copy or create the src/ directory.
4. Navigate to the src/ directory and copy or create the Makefile mentioning how to build the package. This can be ignored and can also be done from package/Makefile in the parent directory. (Refer to [Openwrt online documentation](#) for Makefile details.)
5. Still in the src/ directory, copy or create the source files and include files.
6. If the package requires additional dependency packages having no supplier/repository, then you will need to create those packages as described in steps 1-5.
7. If the package requires additional dependency packages supplied by OpenWrt or by your own repository, modify the feeds.conf.default file as described in the previous section, [Add Your Package Feeds](#). If you add or modify feeds, use the following commands to update and install the feeds:

```
./scripts/feeds update -a
./scripts/feeds install -a
```

8. Run make menuconfig to open the SDK's menu and select the package. Find the package that you want to build and select it by pressing "m". This will also select all the dependencies. Save the configuration and exit the menu.
9. To compile the package, run:

```
make package/<packagename>/compile
```

Note: You can use V=s option for detailed logs of the build.

10. After the package is compiled, the generated .ipk file (or files) are created in the bin/targets/ma35d1/som/packages/ directory.

Integrate the Custom Package into an Image

Complete these steps to integrate the package you just compiled into the image. The image build steps are described in detail in the next section, [Building an Image using Image Builder](#).

1. Copy the .ipk (or .ipk files) that you created in the previous procedure and paste it into the packages/ directory of the Image Builder.
2. From the SDK directory, read the tmp/.packageinfo file to find the package and dependencies information. Refer to the following example for the "helloworld" package:

```
*****
Source-Makefile: package/helloworld/Makefile

Package: helloworld
Submenu: Bin Packages Version: 0-1
Depends: +libc +GCC_LIBSSP:libssp +USE_GLIBC:librt +USE_GLIBC:libpthread
Conflicts:
Menu-Depends:
Provides:
Section: utils
```

```
Category: Lantronix
Repository: base
Title: HelloWorld
Maintainer:
Source:
Type: ipkg
Description: This is a sample HelloWorld program for SDK.
```

```
@@
```

```
*****
```

3. Copy all the information for the package and its dependencies as shown above into `.packageinfo` of the Image Builder. Be careful to ensure that the information is correct before proceeding because if this fails then the build will fail.
4. Build the image as described in the next section.

3. Building an Image using Image Builder

The Lantronix Image Builder is a pre-compiled environment suitable for creating custom images for Lantronix EDS5000(8/16) devices.

Extract Image Builder

To extract the Image Builder, use the following command.

```
$ tar -xvf <imagebuilder-filename.tar.xz>
```

replacing imagebuilder-filename with the actual file name of the tar archive.

Usage

To build the image, use the `make image` command. To customize the image, make use of the following variables:

- PROFILE
- PACKAGES
- FILES
- DISABLED_SERVICES

Run `make help` to get detailed help information.

Select the Profile

The PROFILE variable specifies the target image to build.

```
PROFILE="profile-name"
```

For the EDS5000 image, the profile name should be "eds5000". Run `make info` to get the list of available profiles.

Select Packages

The PACKAGES variable lets you include or exclude packages in the firmware image.

```
PACKAGES="pkg1 pkg2 pkg3 -pkg4 -pkg5 -pkg6"
```

Package names are space separated. Excluded packages are preceded by a minus sign "-".

In the usage example above, pkg1, pkg2, and pkg3 are included in the firmware image and pkg4, pkg5, and pkg6 are excluded from the firmware image.

Included Packages

To get the list of installed packages and IPK packages:

1. To get the list of installed packages, enter the following command.

```
$ cat .config | grep -v '=m' | grep -v 'is not set' | grep CONFIG | awk -F'^CONFIG_' '{print$2}' | sed -E 's/PACKAGE_//' | grep -v '^DEFAULT' | awk -F=' ' '{print$1}' > installed_packages.txt
```

2. To get the list of IPK packages, enter the following command.

```
$ cat .config | grep -v '=y' | grep -v 'is not set' | grep CONFIG | awk -F'^CONFIG_' '{print$2}' | sed -E 's/PACKAGE_//' | grep -v '^DEFAULT' | awk -F'=' '{print$1}' > ipk_packages.txt
```

3. To separate the above new line values into space separated values and remove all CAPITAL strings, use the following command.

```
$ variable=`cat <installed_packages.txt or ipk_packages.txt> | grep -v '[!A-Z]`
```

Note: Run this command once for `installed_packages.txt` and once for `ipk_packages.txt`.

List the `<InstalledPackages>` and `<IPK Packages>` in the `PACKAGE` variable in the `make` command.

Excluded Packages

The following packages will be excluded when the image is built.

```
Ignored Packages == -arm -odhcp6c_ext_cer_id -odhcpd-ipv6only -odhcpd_ipv6only_ext_cer_id -luci-lib-nixio_notls
```

List the ignored packages in the `PACKAGE` variable in the `make` command.

Any other packages to be excluded should be listed in the `PACKAGES` variable, preceded by a minus sign "-".

Note: If the excluded package is a dependency for another package, the build may fail.

Including Custom Packages

To include custom packages with the image, follow the instructions in [Creating a Custom Package](#) to compile and integrate the package files (IPKs) and then list them in the `PACKAGES` variable.

Add Custom Files

The `FILES` variable lets you include custom configuration files into the firmware as default values.

```
FILES= "files"
```

To add custom files, create a "files" directory in the Image Builder root directory where you issue the `make` command. Create a source tree that matches the source tree where the files need to be copied or replaced in the device.

Use the `FILES` variable in the `make` command.

Disable Services

The `DISABLED_SERVICES` variable lists the names of services from `/etc/init.d` to be disabled. Multiple services should be space separated.

```
DISABLED_SERVICES="svc1"
```

where "svc1" is the name of a service from `/etc/init.d`

Use the `DISABLED_SERVICES` variable in the `make` command.

Make the Image

Note: Use the `make help` command to show help information.

1. To build the image, use the `make image` command.

```
make image PROFILE="<Profile name>" PACKAGES="<InstalledPackages> +  
<Installed IPKs> + <IgnoredPackages>"
```

2. To customize the image build:

- To add custom packages, list them in the PACKAGES variable.
- To exclude packages, list them in the PACKAGES variable (preceded by a minus sign "-").
- To add custom files into the firmware as default values, append the FILES variable.
- To disable inbuilt services from `init.d`, append the `DISABLED_SERVICES` variable.

```
make image / PROFILE="<Profile name>" /  
PACKAGES="<InstalledPackages> + <Installed IPKs> + <IgnoredPackages>  
+ <customPackages> + -<unwantedPackages>" /  
FILES="files" / DISABLED_SERVICES="<nameOfServices>"
```

(If copying this example, remove the forward slashes.)

3. After completion of the image build, the build files will be output to the `bin/targets/ma35d1/som/` directory of the Image Builder.
4. To list packages that get installed into the image. (Use the `make help` command to get more details). Two examples are provided below:

```
make manifest PROFILE="<Profile name>"  
//To list all packages that get installed in default firmware  
  
make manifest PROFILE="<Profile name>" PACKAGES="<addCustomPackages>"  
//To list all packages that get installed in firmware after additional custom packages
```

UCI Defaults

OpenWrt relies on UCI, the Unified Configuration Interface, to configure its core services. UCI defaults provides a way to preconfigure your images, using UCI.

To set some system defaults when the device is built, it is possible to create a script in the

`<imagebuilder>/files/etc/uci-defaults/` directory. Scripts in that folder are copied to the device and executed by `/etc/init.d/boot` the first time the device boots. If the scripts exit with code 0 they are deleted afterwards. Scripts that exit with non-zero exit code are not deleted and will be re-executed at the next boot until they also successfully exit.

To update the runtime configuration using a `uci-defaults` script:

1. Under the `<imagebuilder>/files/etc/` directory, create a folder called "uci-defaults".
2. Under the "uci-defaults" folder, create a script file called "99-default-settings".
3. Use SSH to access your EDS5000 device and check for the parameters that you want to change. Use the `uci show <config>` command to get the exact parameter name.

Example (show configuration parameters from network file):

```
uci show network
```

Example output:

```
network.loopback=interface
network.loopback.ifname='lo'
network.loopback.proto='static'
network.loopback.netmask='255.0.0.0'
network.wan=interface
network.wan.ifname='eth0'
network.wan.proto='dhcp'
```

A screenshot of a terminal window titled "COM9 - Tera Term VT". The menu bar includes "File", "Edit", "Setup", "Control", "Window", and "Help". The terminal shows a series of commands entered at a prompt:
`# /bin/sh`
`luci help for more details.`
`luci set network.watchdog.enable=1`
`luci commit`
`# remove LuCI cache`
`rm -rf /tmp/luci-indexcache /tmp/luci-modulecache`
`exit 0`
The output consists of several blank lines. At the bottom, there is a partial command: `- default_settings.sh 1/? 14%`.

4. Customizing the UI Using Image Builder

You can use the Image Builder to customize the UI and other modifications.

Firmware File Name and Configuration Version

Change the firmware file name and the configuration version by modifying the `.config` file in the `<imagebuilder>` directory.

To change the firmware file and configuration version:

1. Change the following options in the `.config` file:
 - `CONFIG_VERSION_DIST=<Lantronix>`
 - `CONFIG_VERSION_NUMBER=<version-number>`

Important: Do not change any other parameters in the `.config` file, as doing so may cause the image build to fail.

2. To view that the file name has been modified, build the image and check the output in the `<imagebuilder>/bin/targets/ma35d1/som/` directory.

Customize UI Elements

Image Builder can be used to customize the items described in this section.

Note: The SDK does not support replacing the UI theme.

To customize UI elements:

1. View the directory structure of the files on the device by navigating to the following directory in Image Builder and drilling down to find the desired UI theme file and other parameter.
`ROOT=<imagebuilder>/build_dir/target-aarch64_generic_musl/root.orig-ma35d1/`
2. Go to the `<imagebuilder>/files` directory (or create "files" directory if it doesn't exist) and create a source tree that matches the source tree where the files need to be copied or replaced.
3. Copy the file from the ROOT and paste it under the "files" directory. Modify or replace the file according to your requirement.
4. Build and install the image to view that the UI related files have been replaced in the device.

The following table describes the customizable UI elements. Refer to the labels in the following figures for examples of the elements listed in the table.

Label	UI Element and Location	File Location/Description
1	SSH banner on Command Line	"ROOT/etc/banner"
2	Logo image on Web UI Login page	"ROOT/www/luci-static/rosy/logo.png"
3	Logo image on Web UI – other pages	"ROOT/www/luci-static/rosy/logo_t.png"
4	Other icons/logos on Web UI (not shown)	"ROOT/www/luci-static"

Label	UI Element and Location	File Location/Description
5	Default hostname on Web UI Login page and above navigation menu on Web UI	See UCI Defaults or Device Configuration .
6	"Powered by" string on Web UI	"ROOT/etc/openwrt_release" Modify DISTRIB_DESCRIPTION parameter.
7	Quick Setup text on Web UI	"ROOT/usr/lib/lua/luci/view/quicksetup/quicksetup.htm"
8	Stylesheet – fonts, sizes, color, styles	"ROOT/www/luci-static/rosy/cascade.css"

Table 1: UI elements

Example: Customize the SSH login banner

Copy the banner file from the ROOT/etc/ directory. Navigate to the <imagebuilder>/files directory and create the etc/ directory. Paste the banner file here and modify it as per your requirement. Build the image using Image Builder and install the firmware on the device.

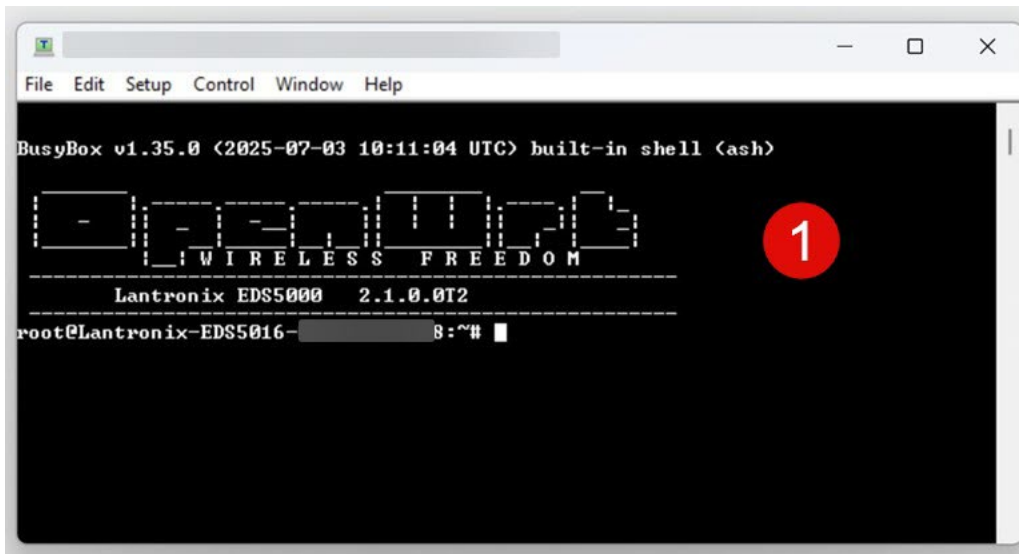
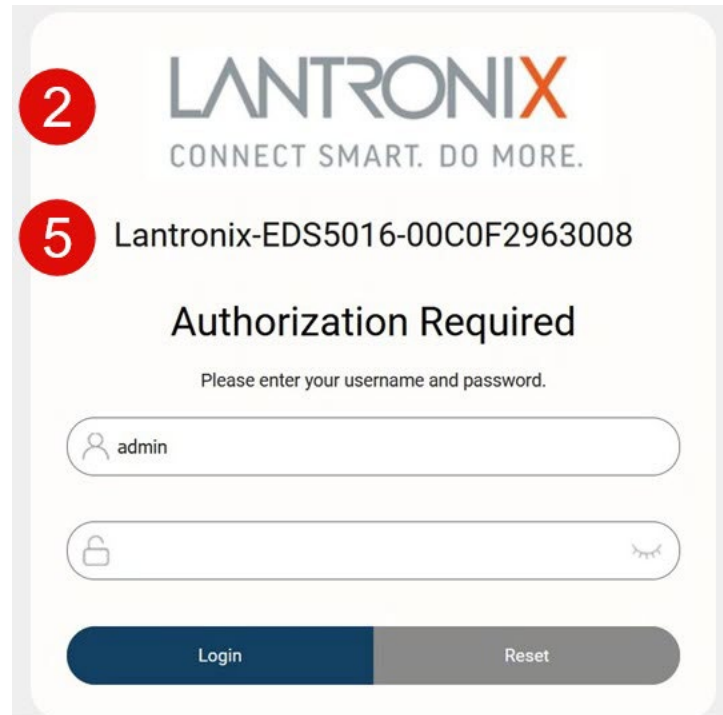
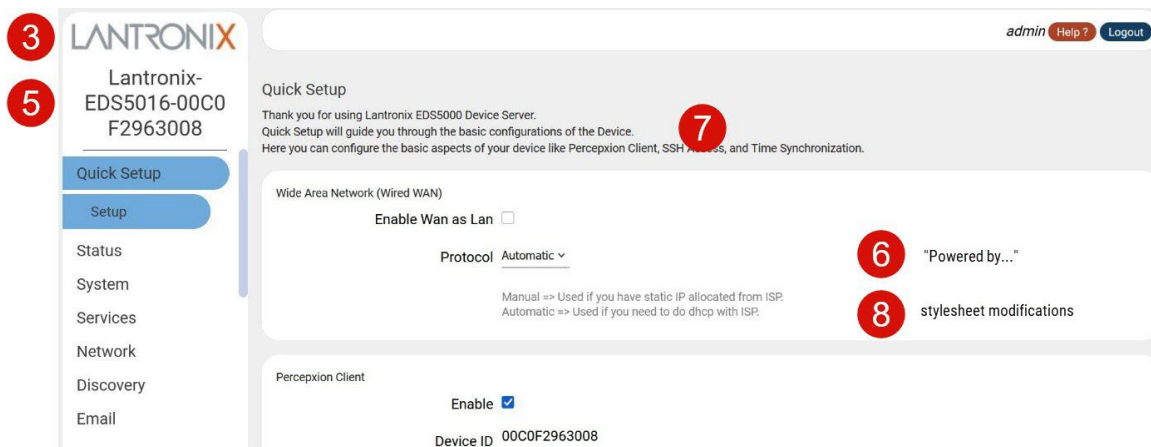


Figure 1: SSH Login Banner



The image shows the Lantronix web administration login page. At the top, the Lantronix logo is displayed with the tagline "CONNECT SMART. DO MORE." Below the logo, the device ID "Lantronix-EDS5016-00C0F2963008" is shown. The main heading is "Authorization Required", followed by the instruction "Please enter your username and password." There are two input fields: the first is for the username, containing "admin", and the second is for the password, which is currently empty. Below the input fields are two buttons: "Login" and "Reset".

Figure 2: Web Administration Login page



The image shows the Lantronix web administration "Quick Setup" page after logging in. The top navigation bar includes the Lantronix logo, the device ID "Lantronix-EDS5016-00C0F2963008", and the user "admin" with "Help ?" and "Logout" links. The left sidebar contains a menu with "Quick Setup" (selected), "Setup", "Status", "System", "Services", "Network", "Discovery", and "Email". The main content area is titled "Quick Setup" and includes a welcome message: "Thank you for using Lantronix EDS5000 Device Server. Quick Setup will guide you through the basic configurations of the Device. Here you can configure the basic aspects of your device like Perception Client, SSH Access, and Time Synchronization." Below this, there is a section for "Wide Area Network (Wired WAN)" with an "Enable Wan as Lan" checkbox and a "Protocol" dropdown menu set to "Automatic". A note explains that "Manual" is for static IP and "Automatic" is for DHCP. Another section for "Perception Client" has an "Enable" checkbox checked and shows the "Device ID" as "00C0F2963008".

Figure 3: Quick Setup page (logged in user)