Roadside ITS Station - RIS-9x60 Platform

RIS-9x60-Platform Administrators Manual

Doc No. 47000001770

Version 01-00

Author:	J.Büger; R. Baily; H. Liebhart
DocReviewer:	H. Liebhart / Rob Baily
Release:	R. Tugrul Güner

Overview of changes.

No.	Version	Status	Date	Contributor	Type of the change
1	00-01	Draft	2018-11-13	J.Büger	Initial version
2	00-02	Draft	2018-11-15	T. Güner	Review, corrections and improvements
3	00-03	Draft	2019-02-96	T. Güner	Final Draft for review
4	00-04	Draft	2019-11-19	R. Baily	Edits and improvements for Review
5	00-05	Draft	2020-01-15	H. Liebhart	Updated chapters OS/SW install
6	01-00	Release	2020-02-10	T. Güner	1st Release

Table 1 Overview of changes

Reference to the status, versions and data classification.

Status:	
Draft	the document is being processed
Released	the document has been checked and released, it can only be modified if the version number is updated.
Obsolete	the document is not valid anymore
Versions:	
00-01, 00-02, etc.	draft versions
01	**first released version with the status "Released"**
01-01, 01-02, etc.	draft versions, that supplement version A
02	**second released version with the status "Released"**
Data classification	
Public	No restriction
Internal	Restricted to internal and external Kapsch employees
Confidential	**Restricted to selected active directory and/or sharepoint groups (default)**
Secret	Restricted to selected employees, server encryption needed

Kapsch TrafficCom

RIS-9x60-Platform Administrators Manual

Doc No. 47000001770
Version 01-00

Document number:	47000001770
Document type:	User Manual
Document issue:	01-00
Document status:	Released
Date of issue:	2020-02-10

Table of Contents.

Page

List of Tables.

1 Introduction.

This document is applicable to Roadside ITS Station RIS-9x60 Platform and related features.

1.1 Purpose of the document.

V2X communication is a core solution in state of the art “connected vehicle” environments. Road operators, infrastructure and road users must cooperate to deliver the most efficient, safe, secure and comfortable journey.

Kapsch TrafficCom specified, designed and developed a new generation Roadside Unit (RSU), the Roadside ITS Station 9x60 platform, or RIS-9x60, as a response to North American, European and Asia-Pacific market demand.

This document is part of V2X roadside equipment user manuals to operate RIS-9x60 platform.

This document shall explain how network configuration, user account management, operating system updates, etc. is possible by administrator users.

1.2 Abbreviations.

The following table contains a list of most important abbreviations used within this document to enable an easy reading.

Abbreviation	Description
CIDR	Classless Inter-Domain Routing
DB	Data Base
EFI	Extensible Firmware Interface (Intel™ Corp.)
LEO	Linux Embedded Operating system
OBU	Onboard unit
OS	Operating System (e.g. Linux)
RSU	Road Side Unit
V2X	Vehicle to everything communication

Table 2 List of used abbreviations

1.3 Privileged User Access

All device configuration activities require root access so you will use the sudo command to run any scripts as the root user. See also section 7 Device Management.

2 *RIS-9x60 Applied Concepts*

2.1 OS image concept

RIS-9x60 is using a Debian based Linux operating system (OS). The OS images are stored in a read only file partition therefore no persistent changes can be made to the root file system and thus a reboot of the system will always result in a well-known state. The complete root file system is versioned and provided by Kapsch TrafficCom.

Any kind of additional customer needs (e.g. additional device drivers, application packages, etc.) must therefore be requested by the customer and will be analyzed and/or reviewed by Kapsch Traffic Com and on agreement between the parties implemented by Kapsch TrafficCom OS core team.

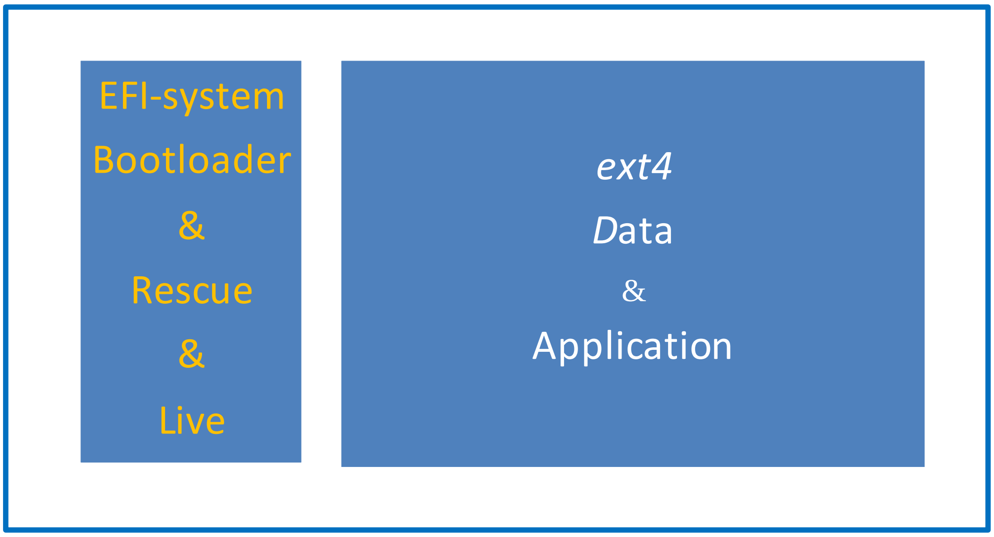
One of the most important design principles for Kapsch roadside equipment is reliability and a possibility for device recovery in almost all situations.

To achieve this, not only the device partitioning is reflecting this, the device is providing two separate operating systems as well.

- A Live OS image
- A Rescue OS image

2.1.1 Basic partitioning

The following block diagram is a simplified overview about partitioning on the RIS -9x60 platform.



One EFI system partition is used to store the bootloader and OS image files (rescue and live system). The EFI partition is mounted read-only to prevent an accidental modification of the OS images. Another partition with a Linux/ext4 filesystem is used to store applications and data.

2.1.2 EFI

A specification originating from Intel™ Corporation, defining the interface between an operating system and platform firmware, and aiming to reduce OS dependence on details of the firmware and hardware implementation, typically also known as BIOS.

CPU module of RIS-9x60 is using a specialized Kapsch proprietary EFI version and updates are typically done in production facilities only.

2.1.3 Live OS

The LIVE OS Image is intended to provide feature-rich functionality providing everything needed to operate the device in the field. If needed, Kapsch will provide update files for the OS update and will indicate a more recent/updated OS to be used in future for the device. OS updates will reflect fixes for known vulnerabilities, software improvements or fixes for known defects. It is highly recommended to always run the device with the latest variant of OS provided. Any communication about device behavior will assume the most recent version of OS is operated.

The OS is not customer specific. Packages necessary for special application usage must be installed in persistent area of data partition.

2.1.4 Rescue OS

The rescue image is a fully functional Linux operating system (in device delivery state Rescue OS is typically the same version as Live OS) which is only active in case the startup of live image is not possible. It is intended to provide emergency operation capabilities to recover failures from Live OS image and/or data partition to regain normal operation without approaching the device physically.

Network operation mode (eth0) and user account settings will be the same as running live OS image. Depending on installed version the latest patch level or some device drivers may be missing.

Note: Rescue image does not run any application specific start-up scripts! Therefore some application or customer specific settings may be missing.

2.1.5 Persistence area

The persistence area is a partition in the Flash memory used for storage of applications, configurations and logging information in a persistent way. The persistence area is mounted at mounting point `/mnt/c3platpersistent/` at startup. Later in this document the persistence area will be denoted as `<pers>` for simplicity and readability.

2.2 OS / System startup

2.2.1 File System concept

2.2.1.1 File system concept for Debian 8

To meet the expectations of the Debian system forming the basis for the OS several directories are RAM-based tmp-fs behind the scenes.

Among others the most important ones are

- /etc
- /home
- /run
- /tmp
- /var

The OS does know about this and fills in where needed, e.g. the network configuration file `/etc/network/interfaces` is created on the file based on EEPROM parameters found. Therefore, some procedures well known from other systems might work (e.g. adduser) temporarily but will be lost on device reboot.

2.2.1.2 File system concept for Debian 9

The basic concepts described in the previous chapter stays intact for Debian 9 as well, instead of a tmp-fs based solution an overlay file system is used.

2.2.2 Meeting Linux expectations - etc configuration

Linux and a lot of Linux related system components or applications expect having the corresponding configuration available in the `/etc/` directory.

To fulfil this need, configuration is modified at system startup if needed and copied to the /etc directory. After this, the corresponding software can be started reading its configuration from the expected location.

Typically, the startup scripts will take care about this during startup of the system for each software component.

2.2.3 Startup hook

There’s a single point connecting the read-only OS during startup to the persistence area. The OS is checking for a single file after the bare OS has been started: `<pers>/startup.sh`

If this file is found and it’s executable it will be invoked with the parameter `start`.

This startup script now checks for script existing in the directory <pers>/etc/startup.d and executing all found scripts in alphabetical order. All software installed in the persistent area is bringing its own startup scripts which will be started in the correct order.

The startup script has a second meaning as well: It can be invoked manually with the parameter stop, this can be used to gracefully shut down all running application SW, ensuring all network connections or open files will be closed correctly.

2.3 Rescue OS recovery

RIS-9x60 has a built-in boot counter which is increased on boot start and reset on successful OS load. After 3 consecutive failed boot attempts the next boot attempt will start the Rescue OS instead of Live OS. In 99% of all cases the reason for ending up in the Rescue OS is simple – a shaky or interrupted power supply, e.g. during installation which has led to a constant increase of the boot counter. Since OS version 3.4 a reboot from Rescue OS will try again to boot into Live OS so a simple reboot is sufficient when Recue OS got activated accidentally.

2.4 Parameter store

It has been essential to have a storage possibility fully decoupled from the flash storage. This is considered being the only chance to have a parameter store being capable of withstanding any manipulation of flash file systems on RIS-9x60.

The parameter store is established as a separate EEPROM chip connected to the main system.

2.4.1 Parameter handling during system startup

The OS is aware of the parameter store, on startup the content from the storage is read and extracted in form of single parameters to the directory `/etc/eeprom-cb/` (the extension *cb* is indicating it's the parameter store on the carrier-board since an available option board may have its separate storage along as well).

The dedicated parameter store in combination with the read-only OS and its tmpfs-approach is the reason why some of the well-established procedures known from a standard Linux system cannot be used.

Examples:

- Modifying any file under `/etc/network/` could be done but is lost the moment the system is rebooted. On reboot parameters are read from the parameter store and files in the /etc/ directory (being a mounted tmpfs) are created
- The adduser available on OS level does not know about the decoupled parameter storage, therefore users would be created only temporarily (reflected in modified files in the `/etc/` directory) and won't be available the moment the system is rebooted.

Therefore, dedicated helper scripts / application exists to reflect the need to persist the information in the parameter storage to be available after a system reboot.

3 *Default Configuration*

New units come configured with DHCP unless otherwise noted during your delivery process.

All units come with an admin user and default password that is the same.

Note: See the Device Management section below for information on changing this password. It should be the first thing done to secure the unit.

4 *OS Update*

4.1 General

Both the Live and the Rescue OS can be updated via remote console (SSH). Update of OS images does not influence any kind of already installed customer application nor operation parameter of applications. During the update of OS typically no application software shall run and the actual update step might require one or several device restarts which will be performed automatically.

The following chapters will describe a regular OS update, i.e. an update within an OS generation. Please refer to chapter 4.1.4 first if the OS update is indicated being a major update, e.g. when stepping the major OS version number.

Update of OS images must be done by an user with admin user access rights or using the sudo command.

The update procedures typically consist of following steps:

1. Transfer of new image and extract if necessary
2. Stop all running applications
3. Perform OS update
4. Restart

The OS update image includes everything a RIS-9x60 related operating image needs, hence the file size (compared to the application software running on top) is quite big. Typical file sizes for both OS images are:

- Approx. 400 MB for the Live OS image
- Approx. 180 MB for the Rescue OS image

File sizes must be considered both for bandwidth and amount of transferred data over the back office-connection used, especially in case LTE connectivity is used. Please ensure that enough disk space is available at the target device (especially using *tmpfs* user home directories).

In case both Live and Rescue OS images shall be updated, the update steps must be done in two separate update cycles. For usage of new/additional application functionality an OS update performed in advance before a software update might be required.

Update of rescue image in the field is typically not necessary since basic functionality of units will be given before the devices are delivered.

In case of rescue image update in the field is needed (based on indication by Kapsch TrafficCom only) precautions shall be taken to avoid interruption during the update process. Updating the Rescue OS means manipulating the sheet anchor of the device.

4.1.1 File (OS image) transfer

Transfer new OS image towards target by means of your preferred application program (*scp, WinSCP, …*). This step is identical for both the Live and the Rescue OS image.

Transfer typically is performed into a RAM-FS section on the target which will be cleared automatically when performing the reboot. By default, a location like `/home/<youruser>/` shall be used.

Note: The file transfer can take place while the device is in full operation, the transfer is a preparation step before performing the update.

Sample:

```
My@HOST:~$ scp CB_9160_live-3.1.0.721.198.zip admin@192.168.100.164:/home/admin/

admin@192.168.100.164's password: <admin-password>
```

When the OS update file is successfully transferred to the device the next steps can be executed to actually perform the update.

4.1.2 Stop running all applications

Prior to the OS update itself all applications shall be gracefully shut down. This will close all network connections, open files and similar and will terminate all running software in a controlled manner.

Tear down of RIS-9x60 is forced by issuing the following command (admin user rights needed):

```
admin@PMP00519:~$ sudo /mnt/c3platpersistent/startup.sh stop
```

You will get a lot of printouts showing the shutdown procedure of application and devices, at the end you shall read something like:

```
+ sleep 0.1
+ [[ false == true ]]
+ [[ true == true ]]
+ mkdir -p /var/log/startup.d
+ /mnt/c3platpersistent/etc/startup.d//000_begin.sh stop
+ sleep 0.1
+ exit
```

This printout is (and the fact you're back on the command prompt) indicates the shutdown process has completed and you may do the next step needed to perform the OS update.

4.1.3 Start OS image updates

Now you should be able to issue one of the following commands depending on which image update shall be done.

Note: During OS updates any restart or power cycle must be prevented!

4.1.3.1 Start Rescue OS Image Update

For updating the Rescue OS image the device must run the Live-OS Image. Go to the directory where you've transferred the rescue OS to and call the update procedure like:

```
admin@PMP00519:~$ cd /home/admin/
admin@PMP00519:~$ ls -la CB*
-rw-r--r-- 1 admin admin 169037761 Nov 12 12:10 CB_9160_rescue-3.1.0.768.209.zip

admin@PMP00519:~$ sudo /bin/update-rescue-os-image.sh -f CB_9160_rescue-3.1.0.768.209.zip
```

During the update procedure you will see following output on the command shell at the end of the process:

```
gpg: key C5D78192: public key (temporary OS image signing key)
gpg: Total number processed: 1
gpg:               imported: 1  (RSA: 1)
Update /home/admin/CB_9160_rescue-3.1.0.768.209.zip -> /mnt/c3platimages//rescue
Validating ...
Extracting ...
Validating ...
admin@PMP00519:~$
```

During rescue image update no automatic restart will be performed.

Restart the device per command manually and check the revision information

```
admin@PMP00519:~$ sudo reboot
```

4.1.3.2 Start Live OS Image Update

Having transferred the OS image to the unit the update process for the Live OS update can be invoked.

```
admin@PMP00519:~$ cd /home/admin/
admin@PMP00519:~$ ls -lah CB*
-rw-r--r-- 1 admin admin 229405553 Nov 12 12:24 CB_9160_live-3.1.0.768.209.zip

admin@PMP00519:~$ sudo /bin/update-live-os-image.sh -f CB_9160_live-3.1.0.768.209.zip
```

During the update procedure of OS live image the RIS-9x60 will restart automatically twice. The update of the Live OS images takes approx. 2-3 minutes in total before the device is back online with the Live OS image. After the device is up and running again you are able to validate that the update is successfully executed.

Update of Live OS image can be invoked both from the Rescue OS and from the Live OS. The device is trying to return to the version previously running – if the update had been invoked from the rescue OS it will stay in the rescue OS, with the update invoked from the Live OS the device is trying to start the updated Live OS. After the device is up and running again (assuming the update step has succeeded and the banner is showing LIVE OS has been booted) the new OS can be validated.

4.1.4 Order of “update all” procedure

Assuming a RIS-9x60 device shall get an update for all parts involved it is important to follow a specific order performing these updates. This is e.g. applicable for a major OS update (e.g. when stepping from OS versions 3.x to a 4.x reflecting the change from the base Debian 8 to Debian 9):

Important: If you’re going to apply a major OS update you need to have three different packages at hand: The Live OS, the Rescue OS and the SW application package.
If you’re missing one of those do NOT perform the update at all!

1. Ensure the device is running on Live OS – check indication when logging in via SSH
2. Update Rescue OS – please refer to chapter 4.1.3.1 for more details
3. Update Live OS – please refer to chapter 4.1.3.2 for more details
4. Update the application software – please refer to chapter 5.4

4.2 OS and HW Version Information

4.2.1 CPU module, EFI

Basically there should be no need to validate or provide information about CPU module as these parameters are only programmed in production facilities and stored in a common production evidence DB.

CPU module version information:

```
admin@PMP00519:~$ sudo cat /etc/eeprom-cb/p.cpu.mod.serial; echo;

NKDF70117
```

Additional information about the EFI version used can be displayed by following query

```
admin@PMP00519:~$ sudo dmidecode | grep "Version:"

        Version: MVV1R976 X64
        Version: N/A
        Version: 1.0.0
        Version: N/A
        Version: Intel(R) Atom(TM) CPU  E3825  @ 1.33GHz
        SBDS Version: N/A
```

4.2.2 OS Version

Information about the installed Live OS version can be retrieved in different ways.

4.2.2.1 Live OS version at welcome screen when logged in

Following information will typically be printed every time a *ssh* connection towards a RIS-9x60 is established:

```
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
  _  __                   _    ______
 | |/ /__ _ _ __  ___  ___| |__ \ \ \ \
 | ' // _\ | '_ \/ __|/ __| '_ \ \ \ \ \
 | . \ (_| | |_) \__ \ (__| | | |/ / / /
 |_|\_\__,_| .__/|___/\___|_| |_/_/_/_/
           |_|

 Copyright (c) Kapsch TrafficCom AG 2016

 LEO version:        3.1.0.721.198
 Image build date:   2018-10-23
 Controller HW type: CB_9160
 OS image type:      LIVE

Last login: Mon Nov 12 07:06:39 2018 from 192.168.100.101
```

The version string nearby “LEO version” does give information about major/minor revision as well as internal version and build information.

4.2.2.2 Version file on tmpfs created during OS boot

The live OS version is available (e.g. for scripting) also in directory /etc:

```
admin@PMP00519:~$ cat /etc/c3plat_version

3.1.0.721.198
```

4.2.3 LIVE OS version retrieved from partition data

This option may be helpful in case of RIS-9x60 is currently running from rescue image.

```
admin@PMP00519:~$ sudo cat /mnt/c3platimages/live/LIVE | grep "live image"

        MENU LABEL live image (ver 3.1.0.721.198)
```

4.2.4 Rescue OS version at welcome screen when logged in

Similar to Live OS images when the Rescue OS image is active a welcome banner is printed when connection via SSH is established.

```
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
  _  __                    _    ______
 | |/ /__ _ _ __  ___  ___| |__ \ \ \ \
 | ' // _\ | '_ \/ __|/ __| '_ \ \ \ \ \
 | . \ (_| | |_) \__ \ (__| | | |/ / / /
 |_|\_\__,_| .__/|___/\___|_| |_/_/_/_/
           |_|

 Copyright (c) Kapsch TrafficCom AG 2016

 LEO version:         3.5.0.987.244
 Image build date:    2018-10-23
 Controller HW type: CB_9160
 OS image type:       RESCUE

Last login: Mon Nov 12 07:06:39 2018 from 192.168.100.101
```

The version string nearby “LEO version” does give information about major/minor revision as well as internal version and build information.

4.2.5 Rescue OS version retrieved from partition data

The version of the installed rescue image can be retrieved by following command.

```
admin@PMP00519:~$ sudo cat /mnt/c3platimages/rescue/RESCUE | grep "rescue system"

        MENU LABEL rescue system shell (ver 3.1.0.721.198)
```

5 *Application Software*

5.1 General

You may need to install your application software on your RIS-9x60 from scratch and initialize persistent settings related to localization, project or customer for this you need to execute the software installation procedure in 5.3.

You may be given application software updates periodically to add new functionality or to address deficiencies for this you need to execute the software update procedure in 5.4. You should update these on your unit to ensure it is up to date. If you have multiple units then update one and let it run as a “burn in” for several days before updating the remaining units.

5.2 Application Software Version

The Application Software installed on a RIS-9x60 always comes with an indication for version, this information can be retrieved from the file `<pers>/etc/issue`. A typical content will be

```
<SW-Projectname>
<SW version>
built on <buildmachine>
built at <builddate>
built by <build-server>
build number <build-number>
```

Example

```
root@164-PMP00520:/mnt/c3platpersistent# cat etc/issue

ITSG5-GenericCCA
1.10-beta-deb9
built on debian9
built at 2020-02-09T23:21:06+01:00
built by jenkins-a-stack-for-ris-debian9-437
build number 437
```

When updating the software according section 5.4 the project name must match the name of the software package to be installed. If it matches then you are allowed to update the firmware. If not then it must be full installation per the section 5.3 below.

Note: Full Software Installation should only be done with the guidance of Kapsch! This procedure will wipe out all application settings so use with caution!

5.3 Software Installation

As explained in other parts of this document the device has different areas where information is stored. The rational for this is a best stable and robust device which can fulfil the needs of an field-installed and 24/7 operated device.

Application software is stored on the W/R area of the flash file system, called persistence area, in this document referred to as `<pers>`. This means when `<pers>` is used in some path indication this shall be replaced by `/mnt/c3platpersistence/` to get the correct full path usable for a command on the device.

When a device is shipped it is typically provided with a default application software on it. To use the device in a particular project the correct project related software package needs to be installed.

The first full installation of a particular software variant may be necessary for new devices, but could also have other reasons, like starting from a well-known initial state, after the persistence area has been wiped, when changing usage of a device between different projects and similar.

Full Application Software installation of the RIS-9x60 with a new or different software version does not change existing configuration of the device, this means basic communication parameters stored in the EEPROM like all IP settings stay fully intact and will be used further on in operation.

Software delivery is done by providing a full self-contained tar.gz providing anything needed for correct operation (beside previously installed OS). The naming convention for such a file is

```
kapsch-ris-firmware_<release#>-<OSVariant>_<Projectname>_<buildnumber>.tar.gz
```

To perform full application software installation please execute the following steps

1. Copy the file to the device using your credentials with a program like WinSCP. The package should go in your home directory (like /home/admin)

2. Copy the provided SW package on the device, e.g.

```
admin@PMP00519:~$ scp <swpackage.tar.gz> admin:192.168.0.27:/home/admin
```

3. Change to the directory where you’ve uploaded the file

```
admin@PMP00519:~$ cd /home/admin
```

4. Extract the software package

```
admin@PMP00519:~$ tar xzf <swpackage.tar.gz>
```

This will result in a set of DEB package files and two scripts files.

Note: While extracting you may see a warning about file timestamp is in future. This may be ok since it's just indicating the system does not have a correct time set (yet, many times a device will be set up without having sufficient GNSS coverage to retrieve time information).

5. Execute the bootstrapping script file with elevated permissions:

```
admin@PMP00519:~$ sudo ./bootstrap_roadside.sh
```

Note: If existing software is detected the script is asking if overwriting is ok – this needs to be answered as indicated with yes<enter> to continue, otherwise the script stops installation.

6. As a final step basic configurations needs to be set. For this execute the following and set the parameters requested accordingly.

```
admin@PMP00519:~$ sudo /mnt/c3platpersistent//bin/roadside admin configure
```

7. If not done already you may want to setup additional user accounts, please refer to related sections in this document for details e.g. 7.7 Account management.

8. As the final step a reboot or power cycle shall be performed to activate the software installed.

5.4 Software Update

A Software update can be performed when the software package to be installed is different -typically newer- version to the already installed version i.e. an update from one release to a more recent release shall be performed. Software update keeps the existing configuration unchanged, therefore for a software update the roadside admin configure step is omitted.

Note: Please check content of file <pers>/etc/issue against the SW package name if the variant is the same. In case of doubt full installation according 5.3 shall be performed.

Follow the steps below to update software from one release to a more recent release within the same software variant:

1. Copy the provided SW package on the device, e.g.

```
admin@PMP00519:~$ scp <swpackage.tar.gz> admin:192.168.0.27:/home/admin
```

2. Change to the directory where you've uploaded the file

```
admin@PMP00519:~$ cd /home/admin
```

3. Extract the software package

```
admin@PMP00519:~$ tar xzf <swpackage.tar.gz>
```

This will result in a set of DEB package files and two scripts files.

Note: While extracting you may see a warning about file timestamp is in future. This may be ok since it's just indicating the system does not have a correct time set (yet, many times a device will be set up without having sufficient GNSS coverage to retrieve time information).

4. Execute the software update script to install the new variant of software

```
admin@PMP00519:~$ sudo ./update_roadside.sh
```

Note: All existing software will be stopped before installation of new software can take place.

Update does not change already configured user accounts, neither any existing network configuration will be modified.

5. As the final step a reboot or power cycle shall be performed to activate the software installed.

6 Network Configuration

6.1 General

RIS-9x60 is using a Debian based Linux operating system (OS). The RIS-9x60 is equipped with an Ethernet interface as the only external interface for system integration e.g. Traffic Management Center, Traffic Light Controller or similar. The Ethernet interface will be configured during startup of the device according to a parameter set stored in the EEPROM of the device.

The default network configuration is DHCP (Dynamic Host Configuration Protocol), therefore a local DHCP server must be available to get connected to the device.

Note: Dynamic addressing simplifies network administration because the software keeps track of IP addresses rather than requiring an administrator to manage the task manually. This means that new devices can be added to a network without the hassle of manually assigning them unique IP addresses and managing address spaces.

Typically the DHCP server does also provide additional information about gateway, net mask and other network configuration settings.

Nevertheless the RIS-9x60 administrator must have access (at least read only) to the respective DHCP server to obtain the IP address which has been assigned to the new RIS-9x60 device in the network.

The MAC addresses of all network devices in the RIS-9x60 are printed on a dedicated MAC address sticker on the enclosure of the device as for Ethernet, IEEE 802.11p radios, etc.

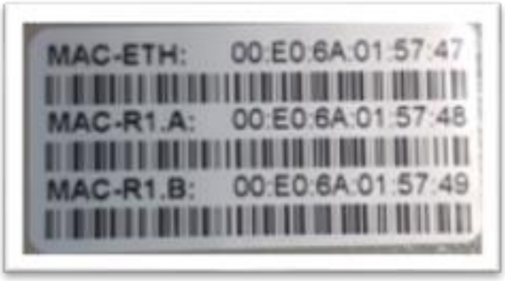

MAC-ETH: 00:E0:6A:01:57:47
MAC-R1.A: 00:E0:6A:01:57:48
MAC-R1.B: 00:E0:6A:01:57:49

Figure 1MAC address label on the RIS-9x60 enclosure.

The permanent MAC address of the specific device in this case MAC-ETH may be used for DHCP server configuration to assign always the same IP address for this device (Detailed configuration information may be part of the product documentation of the server device).

The network configuration may also be set to static IP addressing and a static IP address may be assigned manually by the device admin.

Important:

Modification of the network configuration is allowed for users with privileged user rights only.

Modifications are stored in the EEPROM and are activated during the next restart of the RIS-9x60.

For IP configuration there is a script named ***update_ipconfig.sh*** to handle updating the configurations that are written to EEPROM. If you run the script with no arguments it provides help on what commands are available. Typical configurations used are shown below.

Warning: There is NO recovery tool yet to reset the IP settings to a factory default. A RIS-9x60 with misconfigured ETH cannot be accessed anymore and needs to be returned to production facility for reset to factory defaults. See below for a possible alternative to access a misconfigured device.

6.1.1 Static IPV4 Address

The command below sets the main eth0 IP address, the number of bits for the network mask to 24 bits (255.255.255.0) and the default gateway.

```
admin@PMP00519:~$ sudo update_ipconfig.sh ipv4_first static 145.80.17.31 24
145.80.17.1
```

6.1.2 DHCP IPV4 Address

The command below sets the main eth0 interface to DHCP.

```
admin@PMP00519:~$ sudo update_ipconfig.sh ipv4_first dhcp
```

6.1.3 DNS IPV4 Address

The command below sets the DNS server(s) IPV4 address. Up to three DNS servers can be configured.

```
admin@PMP00519:~$ sudo update_ipconfig.sh ipv4_dns 145.80.17.2
```

6.1.4 Static IPV6 Address

The command below sets a static global IPV6 address, the number of bits for the network mask to 64 bits and a default gateway.

```
admin@PMP00519:~$ update_ipconfig.sh ipv6 enable

admin@PMP00519:~$ update_ipconfig.sh ipv6_global static 3001:bb::2 64 3001:bb::1
```

6.1.5 Recovery via IPV6 Link Local Address

Even though the configuration script is checking the values before writing them to EEPROM it might happen, that the IPv4 configuration has been done in a wrong way (e.g. wrong IP address cause of a typo) leading to a non-accessible device.

By default, RIS-9x60 is providing both IPv4 and IPv6 networking capabilities. For IPv6 there are two different addresses – the global address with a configurable prefix/length and a link local address.

This link local address is formed according the rules described in RFC4862. Given this, the link local address of a device is built based on a rule containing information from the devices MAC address.

```
eth0      Link encap:Ethernet  HWaddr 00:e0:6a:aa:bb:cc
          inet6 addr: fe80::2e0:6aff:feaa:bbcc/64 Scope:Link
          inet6 addr: fd91:ce63:7c52:0:2e0:6aff:feaa:bbcc/64 Scope:Global
```

Since a RIS-9x60 will always come with a MAC address belonging to Kapsch the vendor specific part 00:e0:6a can be considered fixed, only the last three bytes of the devices MAC address are of interest.

This means a device can be (if IPv6 hasn't been explicitly disabled on the device) by using its link local address containing device specific MAC details as depicted above. It might be necessary to tell your SSH client which IF to use when connecting, please refer to the manual/man pages of your preferred SSH tool.

7 Device Management

7.1 General

The OS of the RIS-9x60 is located in a read only partition and application specific software is stored in a persistent storage location ("`/mnt/c3platpersistent`").

The persistent storage location may be formatted, deleted, etc. during configuration and setup of the device for specific applications and is therefore not the storage media which is used for account management.

For each customer there is a preconfigured administrative account and any further user management configuration is up to the customer and will not be known by Kapsch TrafficCom or anybody else except the customer, the owner and admin of the device.

Instead on a normal file system partition the user account parameters are stored in the EEPROM of the RIS-9x60 similar to those of the network configuration parameters. During startup of the device this parameter set will be extracted from the EEPROM and made available to the OS for consecutive operation.

In case of user accounts are changed, added, disabled, etc. the RIS-9x60 must be restarted to become the changes in the EEPROM active for operation.

7.2 Secured privacy

Customer specific account information is protected in the same way as this is known from any other Linux based OS.

Therefore Kapsch Traffic Com is not able to read, decode etc. any of this information and data and help you to recover user account settings in case you lock out yourself and don’t have the user information to unlock.

This of course may also apply to any kind of repair or refurbish handling, any device at production facility will be reinitialized to default production settings and all customer specific information and configuration will be lost.

It is not possible to delete user accounts, but only to disable them.

7.3 Predefined administrative account

Each RIS-9x60 device is preconfigured with a customer specific "`admin`" account and an application software. This account is only configured during startup from persistent area as long as no other information is stored on the EEPROM of the device. It is the customers responsibility to define and store updated "admin" information (e.g. password) at the initial device setup (e.g. maybe at the same time with the network configuration, etc.) to prevent unintended access and control.

Note: The customer setup of the user “admin” must be done always at the first configuration of the device as in case of any faulty behaviour or operation any change or new setup of the persistent storage area of the

RIS-9x60 may cause to lock-out the customer of the device because the user admin may be damaged or destroyed!

Note: Configuration of additional users for e.g. installation, logging, etc. will not be sufficient as these user accounts typically would not have administrative rights as the admin user.

7.4 Special Kapsch TrafficCom users

For the production of the RIS-9x60 there are "*root*" and "*kapsch*" user accounts but they are never used in real operation and the credentials are protected.

7.5 "root" access

RIS-9x60 does not have an external serial debug console accessible. The connection using "*root*" user via network connection is disabled.

Initially the "*admin*" user account or any other "*user*" account defined by the customer after delivery must be used instead.

7.6 User home directory

Any user account defined will have its own home directory within a RAM -temporary, volatile- file system section (*tmpfs*), hence after power cycle or restart of the device all user related information will be gone!

In case there is a need to keep information or data (files, applications, etc) permanently this information must be stored in appropriate directories in the persistent storage area ("*/mnt/c3platpersistent*").

Warning: This is a common share partition with limited size!

Note: Bootstrapping of RIS-9x60 will delete this information in the persistent storage area ("*/mnt/c3platpersistent*") too.

To create of new directories or store of files within the persistent storage area the user must have admin rights.

In case a normal user should have the ability to store files in the persistent storage area the admin user must create appropriate directory and apply specific user rights after the normal user has been created.

7.7 Account management

7.7.1 Application "*manage-users*"

In the persistent binary folder ("*/mnt/c3platpersistent/bin*") a special application is prepared for user account handling. The application syntax is similar to the specific Linux commands (*adduser, passwd, ...*) but the application is also taking care about the EEPROM handling.

Note: Use "*manage-users*" only, DON'T use native commands even though they may be available!

Commands must be issued with sudo permission.

Use "-h" or "--help" to get additional help for usage, see example.

```
admin@OMB01068:/mnt/c3platpersistent/bin# sudo ./manage-users -h

NAME
        manage-users - Create or update EEPROM persisted users.

SYNOPSIS
        manage-users [OPTIONS]... <COMMAND> <USER>

DESCRIPTION
        Manages the user information in EEPROM. Use --help|-h for full usage.

        Users are created on the system and restored during each boot.
        Authentication information can be either a password, a SSH public key or
        both.

COMMANDS
        add       Add user with given information.
        update    Update user, only information given is changed.
        disable   Disable user by disabling password, removing groups and public
                  keys.
        list      Prints information about user. If <USER> is "all" print
                  information about all users.

OPTIONS
        --password <p>  Use password <p>.
        --pubkey <k>    Use public SSH key <k>.
        --keyfile <f>   Use public SSH key from file <f>.
        --groups <g>    <g> is a comma-seperated list of additional groups to add
                        the user to. (The user is added to the default group in any
                        case.)
        -p              Same as --password.
        -k              Same as --pubkey.
        -f              Same as --keyfile.
        -g              Same as --groups.
        -v|--verbose    Print summary of new values before writing to EEPROM.
        -n|--noop       Do not actually update EEPROM.
        -h|--help       This help screen.

admin@OMB01068:/mnt/c3platpersistent/bin#
```

7.7.2 Add new user

Please be careful using this command, once a user has been created it can only be disabled but not removed or deleted, only admin user(s) is allowed to perform this operation.

Typically you should additionally state the given password and group permissions as needed.

In case of no additional group permissions are stated the user will be in her own group.

```
sudo ./manage-users add <username> -p <user-password> -g sudo
```

This command will create the given user-password and add it to the sudo group (== creates an additional account with admin user rights).

In case of the user shall have access rights to even more groups enter the parameter as comma separated list: e.g. `–g kapsch,sudo,something`.

Please ensure that minimum one user created belongs to the group sudo, else you might end up having a system where nobody is allowed to perform actions with elevated permissions needed. This applies also to the admin user when creating a permanent one with customer-owned password.

7.7.3 Update existing user

To change of password and/or group permissions use following syntax.

```
sudo ./manage-users update user -p <new-password> -g sudo,add_group
```

Change can be done by any user having admin user rights, old user password is not needed to change it with a new user password.

7.7.4 Disable existing user

In case of the specific user is not needed anymore it is possible to disable the specific user account.

Note: It is not possible to delete a specific user as we must keep track of UIDs in the system. In case of a new user would get the same UID as a previous user it would automatically have full access to the files created by the previous user which we have to prevent..

Disabling of a specific user account will remove the password setting (and possible key values) and will change the status of the specific user account to "disabled".

```
sudo ./manage-users disable user
```

Change can be done by any user having admin user rights, old user password is not needed for the change.

8 SNMP User Management

8.1.1 User Management

Best practice calls for the users i.e. administrators of an RSU to change the SNMP credentials upon installation.

To create new SNMP users and modify their passwords use the snmpusm command. Note it must be cloned from an original user and you should remove the original user(in our case rwUser) in snmpd.conf after adding the new user.

```
snmpusm -v 3 -u <OriginalUser> -l AuthPriv -a SHA -A <AuthPassword> -x AES -X
<PrivPassword> <IPAddress> create <NewUser> <OriginalUser>
```

You can then change the passwords for the <NewUser> in this example it is assumed <AuthPassword> and <PrivPassword> are the same:

```
snmpusm -v 3 -u <NewUser> -l AuthPriv -a SHA -A <AuthPassword> -x AES -X <PrivPassword>
<IPAddress> password <AuthPassword> <NewAuthPassword>
```

Here only the Authentication Password is changed:

```
snmpusm -v 3 -u <NewUser> -l AuthPriv -a SHA -A <AuthPassword> -x AES -X <PrivPassword>
<IPAddress> -Ca password <AuthPassword> <NewAuthPassword>
```

Here only the Privacy Password is changed:

```
snmpusm -v 3 -u <NewUser> -l AuthPriv -a SHA -A <AuthPassword> -x AES -X <PrivPassword>
<IPAddress> -Cx password <PrivPassword> <NewPrivPassword>
```

If you want to delete a user:

```
snmpusm -v 3 -u <NewUser> -l AuthPriv -a SHA -A <AuthPassword> -x AES -X <PrivPassword>
<IPAddress> delete <NewUser>
```

- END OF DOCUMENT -