Ein Bild, das Schrift, weiß, Typografie, Design enthält.

Automatisch generierte Beschreibung

Documentation

**Content:**

1. Introduction 3
2. Preparation of the Mikrotik-device 4
3. Installation of the mikrotik.upgrade.server 6
4. Usage of the mikrotik.upgrade.server 8
5. Technical background of the project 10

**Disclaimer:**

Mikrotik®, WinBox®, RouterOS, ROS®, hap x2, hap x3, RB3011 and others are or may be trademarks or registered names of SIA Mikrotīkls.  
This project is not affiliated with SIA Mikrotīkls and SIA Mikrotīkls is not responsible for this project. Link: <https://mikrotik.com/aboutus>  
All names, trademarks or other techniques are only used to illustrate this project.  
There is no responsibility for any faults, errors, defects and so on regarding using this images and/or this software.  
This is a private project and all information stated here are given you as it is and with no responsibility for any defects, errors and harm using this software.  
Alpine Linux is copyrighted by the Alpine Linux Development Team with all rights reserved.  
Also all names and symbols from Alpine Linux are used for illustration purposes only with no responsibility of the Alpine Linux Development Team. Link: <https://www.alpinelinux.org/>

This software is released under the MIT license. The complete text of this license can be found on the following page.

**MIT-License:**

Copyright © 2024 Detlef Lampart / DL7DET

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Copyright © 2024 Detlef Lampart / DL7DET

Jedem, der eine Kopie dieser Software und der zugehörigen Dokumentationsdateien (die „Software“) erhält, wird hiermit kostenlos die Erlaubnis erteilt, ohne Einschränkung mit der Software zu handeln, einschließlich und ohne Einschränkung der Rechte zur Nutzung, zum Kopieren, Ändern, Zusammenführen, Veröffentlichen, Verteilen, Unterlizenzieren und/oder Verkaufen von Kopien der Software, und Personen, denen die Software zur Verfügung gestellt wird, dies unter den folgenden Bedingungen zu gestatten:

Der obige Urheberrechtshinweis und dieser Genehmigungshinweis müssen in allen Kopien oder wesentlichen Teilen der Software enthalten sein.

DIE SOFTWARE WIRD OHNE MÄNGELGEWÄHR UND OHNE JEGLICHE AUSDRÜCKLICHE ODER STILLSCHWEIGENDE GEWÄHRLEISTUNG, EINSCHLIEẞLICH, ABER NICHT BESCHRÄNKT AUF DIE GEWÄHRLEISTUNG DER MARKTGÄNGIGKEIT, DER EIGNUNG FÜR EINEN BESTIMMTEN ZWECK UND DER NICHTVERLETZUNG VON RECHTEN DRITTER, ZUR VERFÜGUNG GESTELLT. DIE AUTOREN ODER URHEBERRECHTSINHABER SIND IN KEINEM FALL HAFTBAR FÜR ANSPRÜCHE, SCHÄDEN ODER ANDERE VERPFLICHTUNGEN, OB IN EINER VERTRAGS- ODER HAFTUNGSKLAGE, EINER UNERLAUBTEN HANDLUNG ODER ANDERWEITIG, DIE SICH AUS ODER IN VERBINDUNG MIT DER SOFTWARE ODER DER NUTZUNG ODER ANDEREN GESCHÄFTEN MIT DER SOFTWARE ERGEBEN.

**1.Introduction:**

Using the popular devices from Mikrotik® gives at some point the need to update/upgrade this devices. This is usually done by clicking on UPGRADE PACKAGES in the Web-Frontend or in the WINBOX®-client. After that the newest version will be checked and the update/upgrade will be performed. Due to the huge popularity of these devices, an update of one or two devices will be done quit fast, but if someone is in the situation to update a larger amount of devices, this will take some time.

This is on one hand due to the complete download of the packages through the internet and also that, when someone has a large amount of devices, suddenly some kind of fairness-download-speed-limit comes in action. Personally I observed that after the fifth (or more) device-updates: the speed of the download will be decreased. Also we must keep in mind that when a new release becomes available, everyone will be try to download it. Fairly we must understand that this peak in update-sessions could not be handled in a very fast way.

Also we must accept, that Mikrotik® could not prepare and hold additional download-capacity to handle this peak in a affordable manner for all the customers. This is definitely no fault or error by Mikrotik®. In that situation you cannot solve this problem in a reasonable way and with no extra (and exploding) costs.

So I decided that it will be fine to run a local copy of some sort of download/update/upgrade-service in my network(s). Also it would be nice to have all packages needed for an extension of the function of the device (LTE/Wireless/LoRa/ROSE-Storage etc.) on a local repository. This brought me to the development of this software.

There are several approaches to hold all needed packages for an installation/update/upgrade locally on a device directly, but in my point of view they are complicated. I thought of a simple installation of a local server, that is updating itself in the background (nightly) and gives the ability to download the actual packages from a web-frontend and also to be available inside the RouterOS® for doing the packages updates/upgrades. But also it would be nice to not have to have an additional server-installation in the network, furthermore to integrate it directly in some powerful Mikrotik®-device. So I decided to use the CONTAINER-function in RouterOS® available roughly from release 7.5 on.

Using the small Linux-distribution Alpine Linux and so very basic tools inside, this container is very small at first (about 40 Mbytes). Also there are no special programming-tools or languages that are used inside the container. Only using the basic Linux-environment, a webserver and some small additional tools installable via the package manager of the Linux-distro makes the size of that container also very small.

The container itself contains during and after install no packages from the download-area of the Mikrotik®-servers and is completely self-configuring. After a fresh install of the container it will be starting to download all the packages, that are available currently from the master-server based at Mikrotik®. Then the system will check each night, if there is a change in the releases and will download the fresh versions of the files/packages. The last packages downloaded before will be stay there, as long as the container is not deleted and freshly installed. Also using the persistent storage function in the container, all packages will stay permanently even after an upgrade or fresh install of this software. So someone cloud build an archive of past and currently releases with this software.

Using the container-technology makes this software to be nearly a “one-clicker” to run and serve the files/packages to the Mikrotik®-devices in the network.

All preparation and installation steps with additional information about the usage will be described later in this manual.

So at the end the reader will be asking “what is the name of that project/software ?”.  
Well, the name is “**mikrotik.upgrade.server**” or short “**mus**”.

I hope that this software will be useful for someone and will speed up your updates/upgrades a little bit. If this is true, I am completely satisfied. If not, sorry for taking your time.

Regards, Detlef

**2.Preparation of the Mikrotik®-device:**

Using the “mikrotik.upgrade.server” (or from now on in short form “mus”) needs a Mikrotik®-device with the CONTAINER-function enabled. This makes only x86\_64/ARM/ARM64-based devices usable for the “mus”. Other architectures are currently not capable to run a container-image inside the devices.

Also a affordable size of memory (RAM) and some sort of (relatively fast) disk (with enough disk space on it) is needed. Devices with a amount of >256 Mbytes of RAM and a disk (usb/network/local on CHR) with minimum 16 GB are needed.

Devices that support the ROSE-storage-package will give the availability of running the disk as a iSCSI/SMB etc. share. Especially CHR (virtualized installations) will make use of a local disk in the hypervisor or virtual environment. This is the recommended storage for a fast and convenient installation.

Several details will be described in the later manual as they appear.

Here is a rough overview of the following steps in short:

1. Install CONTAINER-package and enable CONTAINER-function on the device
2. Create one or several VETH-devices for the container(s) and give them IP-addresses
3. Create a bridge (also with IP-address) for these VETH-devices and the container(s) running on
4. Prepare the FIREWALL, NAT and the PORT-FORWARDING to reach the container(s) from the outside networks (from the view outside the container-bridge)
5. Configure the container with the used VETH-device, the image to be downloaded and some other configurations like DNS, start-on-boot, logging etc.
6. Apply the container to the device
7. Start the container
8. Try to reach the web-interface
9. On access, please wait during the self-configuration and the download of the files/packages.
10. Use the “mus” on your network – have fun

Steps 1 to 4 are regarding to preparation.  
All later steps are described under topic 3. “Installation of the mikrotik.upgrade.server”

As additional steps the installation of a persistent container storage will be described also under topic 3.

Preparation in detail:

Citation from the Mikrotik®-Wiki found [here](https://help.mikrotik.com/docs/display/ROS/Container):  
*Device-mode limits container use by default, before granting container mode access - make sure your device is fully secured.*

First of all the container-function must be enabled on the device. If you are using WINBOX® as your configuration client, you will see “Container” in the left sidebar.  
(Using WEBFIG via a browser will be slightly different, but nearly the same,)

If not, you have to enable this mode as following:

Please make sure that the CONTAINER-package is installed on your device !

Open a console using “NEW TERMINAL” on the left sidebar and type in:

**/system/device-mode/update container=yes**

After that you must restart the device in the given time without shutting it down.  
This means that you have to reset the device via the reset-button or reset the device via the hypervisor/virtual environment (cold-reboot/reset). Do not restart it or shut it via the WINBOX®-Client. This means you have to restart the device the “hard way”, which normally should not be done.

After successfully activating the CONTAINER-mode the following printout should be seen on the “NEW TERMINAL”:

**system/device-mode/print**

**mode: enterprise   
container: yes**

Important hint:

Enabling the container mode and running containers on the device could be harmful if the device is not properly secured (strong passwords & correct firewall configuration etc).  
Anyone who gets access to the device could get root-access to the whole system.  
So be warned !

Now you need to add some virtual interfaces to the device which can be used by the containers. It is advisable to define a new network segment for the containers and the needed bridge. The network-bridge is needed if you want to run more than one container on the device, but I strongly advise to use it. Every container needs one unique VETH-interface and also a unique IP-address. This configuration can be done via the CLI (“NEW TERMINAL”) or via the WINBOX®-client.

Here is the syntax for the CLI:

**/interface/veth/add name=veth\_docker01 \ address=10.10.10.11/24 gateway=10.10.10.1***(the \ means to put all content in one line !)*

Now create a bridge for the container-veth’s:

**/interface/bridge/add name=bridge\_containers**

**/ip/address/add address=10.10.10.1/24 interface= bridge\_containers**

**/interface/bridge/port add bridge=containers interface=veth\_docker01**

After that setup a NAT rule to give the containers access to the outside:

**/ip/firewall/nat/add chain=srcnat action=masquerade src-address=10.10.10.0/24**