README file for TICKpy (CogSys) Container v0.9.4

- Container: TICKpy (CogSys)
- Container-Version: 0.9.4
- Interface-Version: 2.0.0
- Build-date: Wed Jun 27 12:09:08 UTC 2018
- Maintainer: Oliver Beyer
- Support: micasupport@HARTING.com

Quickstart:

Connect your MICA to a webbrowser of your choice. Using Windows, you can access your MICA via

```
https://devicename
```

If your host is running an mdns application, you can access your MICA also via

```
https://devicename.local
```

You have to accept a Security Certificate. After Container installation, you have to configure network settings via Firmware -> Settings -> Network in order to access the Container via SSH.

SSH:

Under Windows, you can use putty to SSH into container by

```
containername-devicename
```

If your host is running an mdns application, you can access your container also via

```
containername-devicename.local
```

If you are using Linux and intend to connect to container by name via IPv6 make sure to resolve the link-local address first and append the network identifier. So eg. if using avahi as mdns client, run:

```
avahi-resolve -n -6 containername-devicename.local
```

in order to get the container IPv6 link-local address. Then connect via SSH:

```
ssh root@linklocal%zone-id
```

You need to append the zone-id / network interface, eg. eth0. If you are using ipv4, then using

```
ssh root@containername-devicename.local
```

is sufficient.

By Default login as user

```
root
```

with password

```
root
```

It is recommended to change your password after first login via

```
passwd
```

Running Services:

- SSH-Server running on port 22 (dropbear)
- mDNS Client (avahi-daemon)
- Web-Server (nodeserver)
- Nodejs-Application (nodeapp)
- https://devicename/containername/API shows this README file
- https://devicename/containername/ABOUT shows container Meta information in JSON

Note: Running a HTTP webserver under port 80 will only be made available by MICA Basesystem via https://devicename(.local)/containername/. The webserver is redirected to port 443, hence encrypted, regardless of hosting a webserver with or without encryption within the container. Similar port redirection applies for websocket services running under port 8080.

Web Icons:

Web Icons *App.png* and *Header.png* can be found under */META/web_icons*. In case new Web Container Icons are desired one can simply replace these files.

IPv6:

Each MICA container is configured with two fixed IPv6 addresses. The Link Local address constitutes like

```
fe80::a:ed<mica-entity>:<mica-base/container>
```

with prefix

```
64
```

The first 11 Bytes are fixed for the HARTING domain, whereas the following 3 Bytes define the MICA entity. You can get this information from the MICA label on the back of your device. The second half of the MAC address represent these 3 Bytes. So if your MAC address was

```
00:0a:ed:49:92:ff
```

your IPv6 Link-local address of your mica would be

```
fe80::a:ed49:92ff:0
```

The last 2 Bytes is showing you the MICA base system / container identifier, whereas *0x0000* stands for MICA base system. Containers are iterated from number *0x8001*. So the first installed container will get *0x8001*, the second *0x8002* and so on.

The unique local address (ULA) can be derived quite similarly:

```
fd96:8d76:d432:0:a:ed<mica-entity>:<mica-base/container>
```

with prefix

```
64
```

Note: Make sure to assign a ULA with same prefix for you network interface in order to reach your mica/container via ULA.

Description:

The TICKpy Container provides the TICK stack, Modern Time Series Platform, designed from the ground up to handle metrics and events. The TICK stack is based on an open source core. This open source core consists of the projects—**Telegraf**, **InfluxDB**, **Chronograf**, and **Kapacitor**. The Open Source Time Series Platform provides services and functionality to accumulate, analyze, and act on time series data.

Telegraf

Telegraf is a plugin-driven server agent for collecting & reporting metrics, and is the first piece of the TICK stack. Telegraf has plugins to source a variety of metrics directly from the system it's running on, pull metrics from third party APIs, or even listen for metrics via a statsd and Kafka consumer services. It also has output plugins to send metrics to a variety of other datastores, services, and message queues, including InfluxDB, Graphite, OpenTSDB, Datadog, Librato, Kafka, MQTT, NSQ, and many others.

Influxdb

InfluxDB is a time series database built from the ground up to handle high write and query loads. It is the second piece of the TICK stack. InfluxDB is meant to be used as a backing store for any use case involving large amounts of timestamped data, including DevOps monitoring, application metrics, IoT sensor data, and real-time analytics.

Chronograf

Chronograf is InfluxData's open source web application. Use Chronograf with the other components of the TICK stack to visualize your monitoring data and easily create alerting and automation rules.

Kapacitor

Kapacitor is an open source data processing framework that makes it easy to create alerts, run ETL jobs and detect anomalies. Kapacitor is the final piece of the TICK stack. Data processing pipelines are written in the script language TICKscript and can be managed using Chronograf. In addition to TICKscripts the container allows to seamlessly extend the capabilities of Kapacitor using Python scripts as User Defined Functions (UDF), in order to provide more complex stream analytics functions utilizing Python modules such as **numpy**, **scipy**, **pandas**, **sklearn**, etc.

The TICKpy Container is currently based on Telegraf v1.5.2, Influxdb v1.4.2, Chronograf v1.4.1, Kapacitor v1.4.0 and Python v2.7.13. For more information about the TICK stack visit https://docs.influxdata.com/.

User-interface:

The user-interface of the TICKpy Container consists of a two section "Dashboard" and "Configuration". The section "Dashboard" gives you access to Chronograf which allows you to dynamically build dashboards, create alerts and TICKscripts and to explore the data being stored in the Influxdb database. The "Configuration" section including the service drop down menus for every service of the TICK stack. These sections can be used to import/export a configuration files (as well as Python scripts) to/from the container and to show log files. Furthermore each service can be started/stopped by using the status button to the right.

The "Configuration" section, as well as the involved HTTP-requests and JSON-RPC-methods are only accessable by the MICA users *containeradmin* and *admin*. The section "Dashboard" is also accessable by the MICA user *user*.

The structure of the drop down menu can be described as follows:

1. **Configuration/Script file text field**: The text field can be used to set the configuration/script you want to upload to the container.
2. **Upload**: The upload button engages the upload of the configuration/script file that has been specified in 1. If an warning message appear on the screen you can check the logs by pressing the "Show Log File" Button.
3. **Download**: The download button engages the download of the current active configuration/script to your file system.
4. **Show Log File**: The log file button lets you enter the log window

Remote Access:

The following **HTTP-requests** are provided by the container:

1. **Upload** : Uploading a service configuration/script.
   Method : POST
   URL : SERVICEup?token=XXX
2. **Download** : Receiving the content of the current active configuration/script.
   Method : GET
   URL : SERVICE?token=XXX
3. **Log file** : Receiving the content of the service log file.
   Method : GET
   URL : SERVICElog?token=XXX

The following **JSON-RPC-methods** are provided by the container:

URL : *wss://devicename/containername/*

1. **Service Status** : Receiving the process status of a service.
   Method : SERVICE_status
   Parameter : [token]
   Return : ["active" : true/false]
2. **Start Service** : Starting the service.
   Method : SERVICE_start
   Parameter : [token]
   Return : ["started" : true/false]
3. **Stop Service** : Stopping the service.
   Method : SERVICE_stop
   Parameter : [token]
   Return : ["stopped" : true/false]

(with SERVICE = telegraf | influxdb | chronograf | kapacitor | pyscript)

Additional Software & Updates:

It is possible to install and update Python modules when logging into the container using SSH.

Install additional Python modules using *pip*.

```
pip install <module>
```

In order to update your PIP-modules use:

```
pip install --upgrade <module>
```

Note: Some PIP-modules require GCC during the build process. If required, install the standard build tools.

```
apt update && apt install build-essential
```

You can also update the TICK stack and Python itself using APT.

```
apt update && apt upgrade telegraf influxdb chronograf kapacitor python
```

Note: You should clean the PIP and APT cache and remove the packages lists after preparation.

```
rm -rf ~/.cache/pip && apt clean && rm -rf /var/lib/apt/lists/*
```