HARTING

Pushing Performance

People | Power | Partnership

White Paper

MICA Security

1. Edition 2018

Contents

Overview

Implementing IoT at an organization adds new network and operational security challenges to a computing environment, by introducing network and cloud connectivity at the shop floor level.

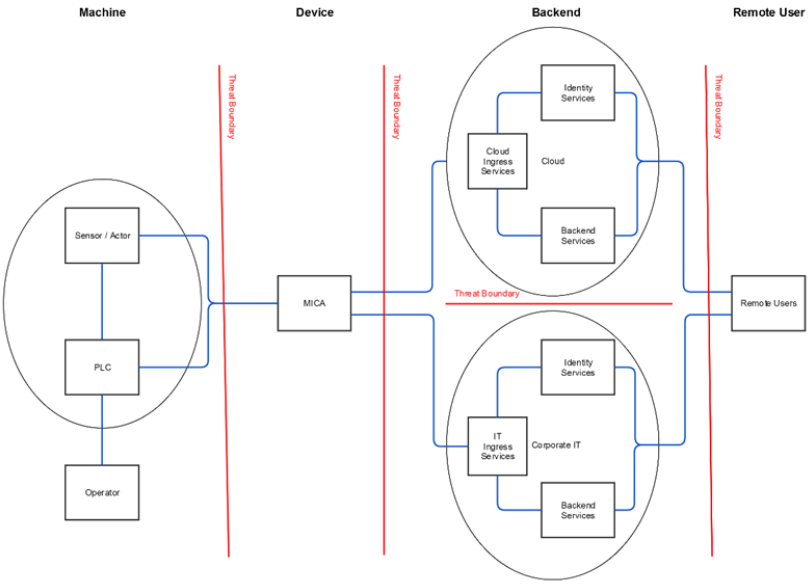
Unfortunately, there are—unlike IT networks, which can look back on 25 years of security challenges and security research— very few specific instructions how to implement IoT securely at the device, network, and system levels. This white paper tries to explain how the MICA can be securely integrated into these environments.

Thread assessment in an IoT Environment

Rolling out IoT at your organization securely requires taking a thorough look at your environment and identifying and mitigating potential threat vectors.

A very useful way of tackling this is to identify thread boundaries—usually contact or handoff points between different systems—and securing them and the zones between the boundaries.

In the case of IoT one could for example define the following threat zones and thread boundaries. This document will mostly cover the Device zone in which the MICA resides.



The Backend thread zone is usually the normal corporate network, the public internet, or private and public clouds. Security in this zone is covered by standard IT security practices. You should consider how to secure the data coming from and going to the MICA. Examples include firewalls, traffic monitors, and in some cases even intrusive methods like deep package inspection.

Conversely, you also might want to tunnel data from the MICA through portions of the Backend zone using techniques like virtual private networks, ssh tunnels, or other forms of encryption.

In many MICA applications like condition monitoring, the traffic from the Machine zone to the MICA is largely unidirectional, or the sensors might be limited enough that no thread emerges. But keep in mind that PLCs and sensors are getting more sophisticated rapidly, so adequate care should be taken by container developers to safeguard against sneak attacks from compromised machines or sensors.

In some MICA applications, though, the MICA is being used to send instructions or data to machines and other actors. This traffic needs to be secured by the container developer.
One example are the HARTING Euromap solution packages which let authenticated users send instructions

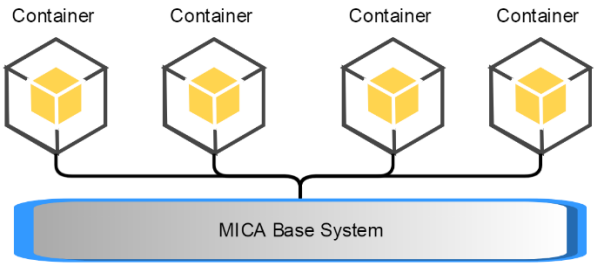
and programs to injection molding machines. Here great care must be taken to make sure only authorized operators can send instructions to the machine.

Physical Security

While the MICA uses uncommon connectors and requires tools to open, it is not intrinsically hardened against malicious access by attackers. If you have to consider this scenario, make sure that the MICA is installed in a secure hardware cabinet.

Software and Network Security

MICA is an ARM-based edge computer with network connectivity running a Linux-based operating system and a virtualized application environment build around Linux containers (LXC)[1], a common and well understood virtualization technology.



While the operating system, called MICA Base, is provided by HARTING, containers may also be developed by third party developers and system integrators. Therefore it's important to distinguish between vulnerabilities of the MICA Base and vulnerabilities of containers.

This document covers the security features of the MICA Base and the container environment. Any security assessment of third party containers needs to be done in conjunction with the container developer or publisher.

MICA Base Security

The MICA uses a busybox[2] operating system based on a recent Linux kernel (at the time of this writing 4.4.96), so it already offers a lot of tried-and-true IT security controls that have evolved over the past 25 years and can be just as effective for IoT. Additionally, its containerized software architecture mitigates many threats that endanger other IoT devices.

Due to very lean nature of the MICA Base, many conventional attack vectors exploiting vulnerabilities in UNIX tools and other core applications are simply not available. For example, the MICA Base does not include package managers, email clients, or other services that are often used by hackers to gain control of devices.

The MICA Base is also not accessible to users or administrators, which also means that it is not possible for users or admins to load additional kernel modules.

[1] For more information on LXC, see https://linuxcontainers.org/
[2] For more information on busybox, see https://www.busybox.net/

User Management

Every MICA comes with three user privilege levels—*user, containeradmin,* and *admin*—to provide protection against accidental or deliberate misconfiguration or deletion/installation of containers. Typically, operators should only be given access at the lowest level they need to fulfill their jobs.

The MICA API

The MICA uses a very small JSON RPC API to communicate between containers and the MICA Base. Except for some trivial functions, executing a RPC command requires authentication.

Starting with firmware 2 all administrative functions and containers are using websockets for communication, so tokens do not need to be passed for each RPC. Instead only a one-time login RPC is needed, which returns a token to the process. After logging in to the MICA Single Sign On (SSO) service one can use this token to pass it to other applications, which need to make sure this token is valid.
If the token has been invalidated, for example when a wss session gets disconnected, the application gets notified by the SSO service.

Virtualisation

All MICA applications run in individual LXC containers, which run in separate kernel namespaces. This automatically provides a number of containment tools, which prevent one process to gain access to another one or the MICA Base.

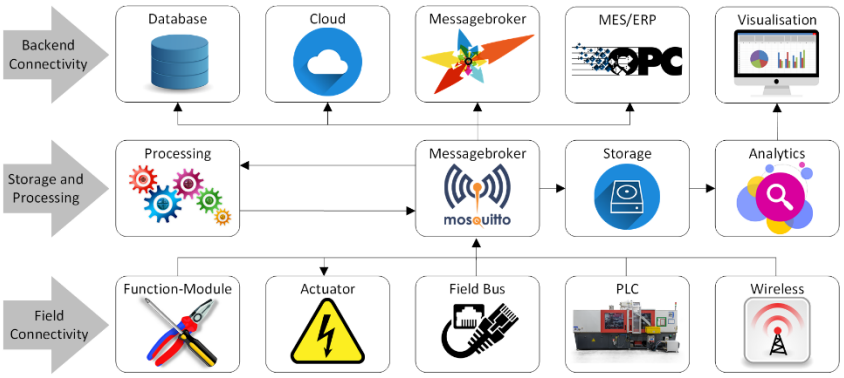
Communication between containers only happen using IP protocols with webservices or wss being the lowest level communication protocol. In many cases, container developers choose higher level protocols like MQTT or OPC-UA which are human readable and also provide some isolation against attacks.

User-generated data is exclusively stored in the overlay file system of each container, which is fully contained inside the container, and not accessible from other containers. Provisions for securing or encrypting the data in the container fall to the container developer.

While this does not offer complete protection, these technologies combined will typically prevent any accidental damage of the host, where damage is defined as things like reconfiguring MICA hardware, reconfiguring the kernel or accessing the MICA Base file system.

Micro services

One important concept for MICA developers is its emphasis micro services: small apps that perform a very specific task. This allows rapid development and reuse of MICA applications, as well as a way to adjust existing systems quickly, by just swapping out containers in an application.
A typical MICA application might contain many containers written in many programming languages, each with a very specific job and all communicating via IP protocols.



From a security standpoint this has two major advantages: functionality and data are isolated between containers, and frequently used components—like the MQTT broker container—are well tested and proven in the field.

Network Security

Secure Protocols

The MICA Base enforces use of secure protocols like HTTPS or Web Socket Secure (wss) communication with the outside world. While not all third party containers have to follow this approach, it is considered good practice for container developers to use secure protocols. Please check the container documentation of the containers running on your MICA for details.

Redirection by the MICA Base

Communication to containers over the most common ports like 80, 8080 (HTTP), 443 (SSL) get rerouted through the MICA Base, so there is no way to access container web servers and the container GUI over unencrypted connections.

Signing and Certification

Signed Firmware

MICA firmware is signed by HARTING and unsigned firmware cannot be installed on MICAs that have not been specifically unlocked for developers.

Container Hashes

HARTING publishes the hash values for all containers we distribute. This prevents third parties from manipulating or inserting malicious code into containers. We recommend that you always verify the hash value of a container archive before you install containers.

Signed Containers

In the near future, HARTING will offer a certification program for third party containers and the distribution of signed containers.

Web Certificates

MICA ships with a HARTING web certificate that is used for encrypting and securing the communication between MICA and other networked devices. You can either accept this certificate as trusted certificate if required or replace it with your own.

Container Security

While we spent a lot of time and effort to secure the MICA Base from harm and preventing malicious containers from eavesdropping on or harming other containers, we cannot guarantee that 3rd party containers will be free of malicious code or specifically designed to exploit other vulnerabilities in your network. For this reason, we recommend that you only install HARTING certified containers, or containers from another trusted source.

Updates and patches

The MICA Base gets updated every 3 months. HARTING recommends that you check the release notes of each release and install updates that fix newly discovered security problems.

Best practices for system administrators

While no system is completely secure, there are some steps you should take to secure your network and operations.

- Update your firmware and applications to the newest and most secure version if your MICA is connected to any network.
- Change the default password immediately.
- Only install signed HARTING firmware.
- Only install containers from trusted sources.
- Review your network security; especially regarding the threat boundary immediately above and below the planned MICA

MICA Security Consultants and Service Partners

If you have security requirements beyond what MICA and MICA containers provide out of the box, please a MICA.network partner specializing in security, or ask your HARTING contact to recommend a partner.

For general security questions, please contact out support team at micasupport@harting.com.

Advanced Topic: Network Topology of MICA

Due to the virtualization technology in the MICA, a single MICA running containers will show up as a number of network devices with their own MAC and IP addresses.

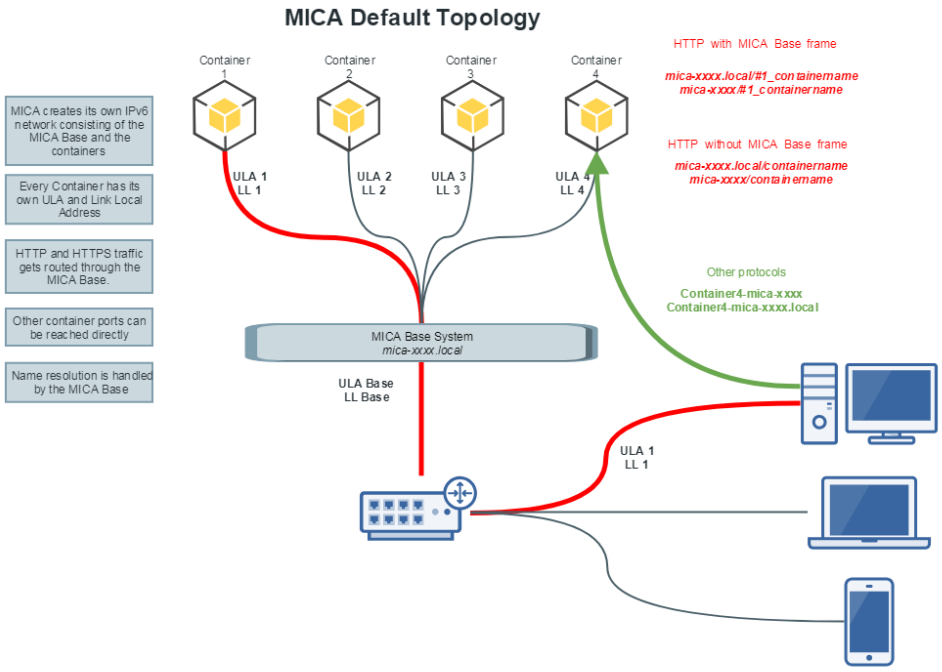
Note: Since containers have their own MAC and IP addresses, any upstream MAC or IP filters need to be updated after a container gets installed on a MICA, or its IP address gets changed manually.

By default, the MICA Base creates its own internal network and assigns each container a IPv6 ULA and Link Local address. Each continer can be reached by these addresses directly, except for port 80, 8080, and 443 which get rerouted through the MICA Base.

The MICA Base also performs name resolution while containers have their own MDNS and LLRP responders.

This means that containers can be addressed via <micaname>/containername. For example, http://mica-test/gpio (or http://mica-test-local/gpio on Linux and Mac OS) will bring up the default web page of the container named gpio running on mica-test if a webserver is runing in the container and a warning message that no web server is running otherwise.

Adding a #1_ to the URL will bringup the default website of the container as an iFrame in the MICA Base web page.



Admins can also enable IPv4 and DHCP on the MICA Base and containers as well as assign additional IPv4 and IPv6 addresses. When DHCP is enabled, the containers and the MICA will get their addresses from the selected DHCP server.