Hardware Hazards: Identifying and Defending against Attacks on LoRaWAN Devices

Technical Brief

The first part of this series[1] introduced weak points in LoRa and LoRaWAN technologies, as well as the scope of available security mechanisms. In the second part,[2] we presented effective techniques and original tools that use software-defined radio to spot attacks on these technologies in the wild.

In this last article, we will detail dangerous hardware attacks that can critically affect LoRaWAN devices. Low-powered LoRaWAN devices are most commonly used by large organizations that deploy them across wide-spread areas, or smart cities that place these devices in public locations across metropolitan zones. Hardware attacks could be real game changers if malicious actors attack the devices deployed in unsecure locations.

The following data will show that the cornerstone of LoRaWAN security resides in the use of default encryptions and the strength of encryption keys. In the sections below, we will introduce the varied hardware attacks that could compromise valuable internal data from an organization, as well as the attacks' specific mechanisms. We will also outline security procedures that could mitigate these types of attacks.

LoRa device components

Vital components of LoRa end-devices include transceivers, microcontrollers (sometimes called MCUs), and sensors. The microcontroller communicates with the LoRa transceiver[3] to set the configuration and retrieve or send packets. All the code and logic reside in the external or internal flash memory of the microcontroller.

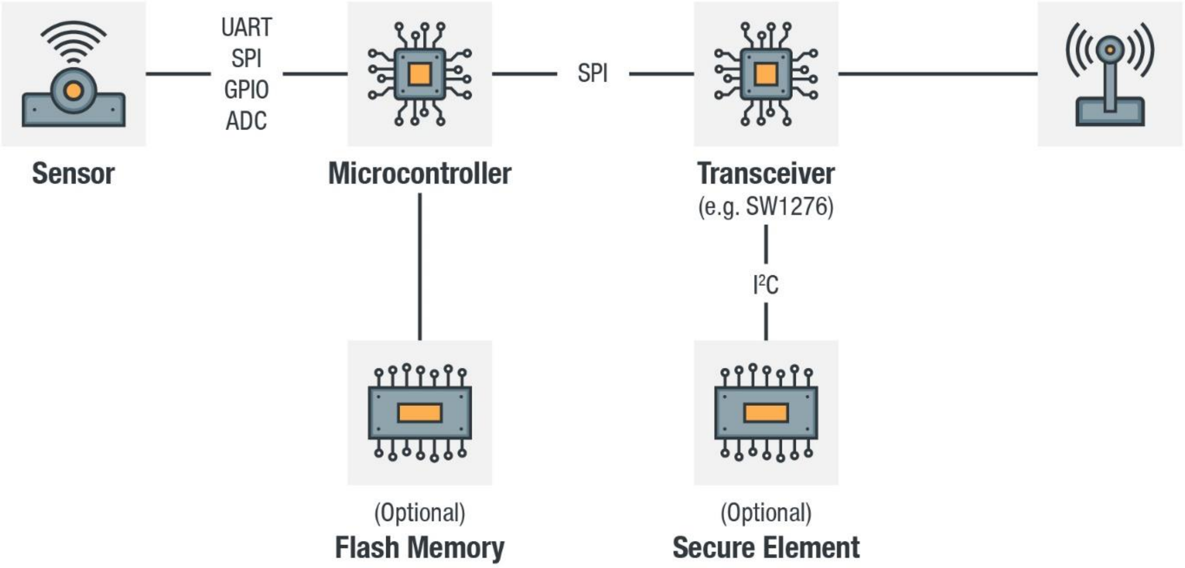


Figure 1. LoRa device components

Typically, a sensor is also connected to the microcontroller. Figure 2 shows a LoRa end-device with a magnetic door sensor (boxed in purple) and the ASR6502 microcontroller (boxed in pink):

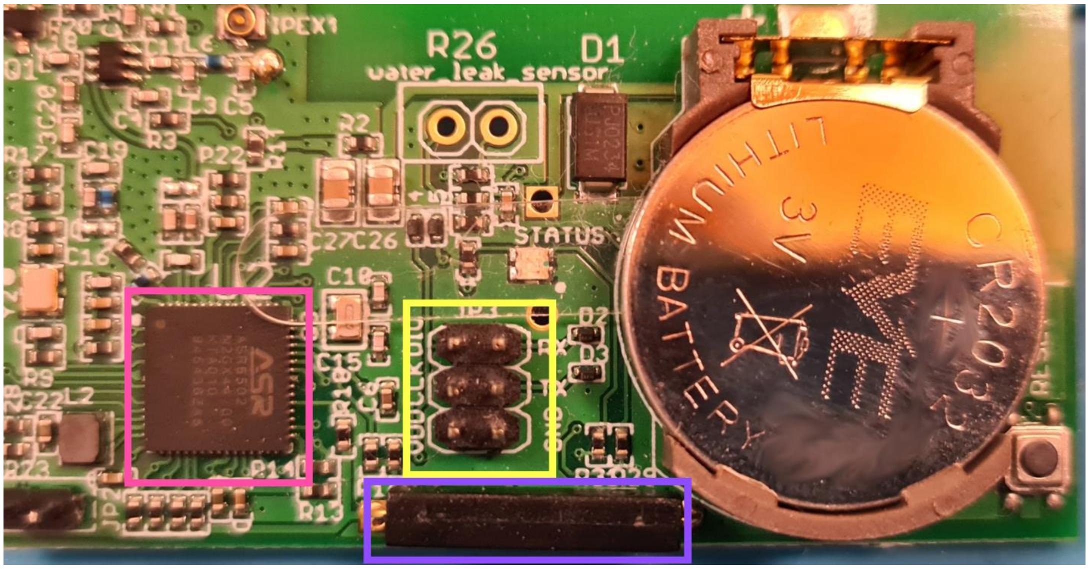


Figure 2. LoRa end-device with a magnetic door sensor

The yellow box in Figure 2 clearly shows PINs, indicating a universal asynchronous receiver-transmitter (UART) that we can interface with.

There are also other possible interfaces that can be used to gain this type of access. Among them are:

- JTAG (named after the Joint Test Action Group)
- I^2C (Inter-Integrated Circuit)
- Serial Peripheral Interface (SPI)

The following sections list the different components and the possible attack vectors for each one.

Components and associated attacks

LoRa transceiver

In the following example, the LoRa transceiver communicates with the microcontroller through SPI.[4] The microcontroller uses this SPI access to provide the configuration, but also uses the SPI to get uplink packets or send downlink packets to the gateway.

Using a HydraBus[5] device allows us to interface with this transceiver:

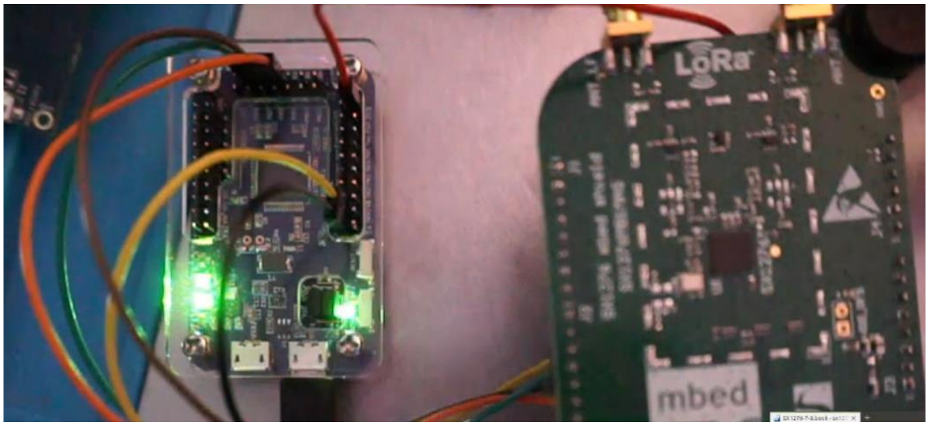
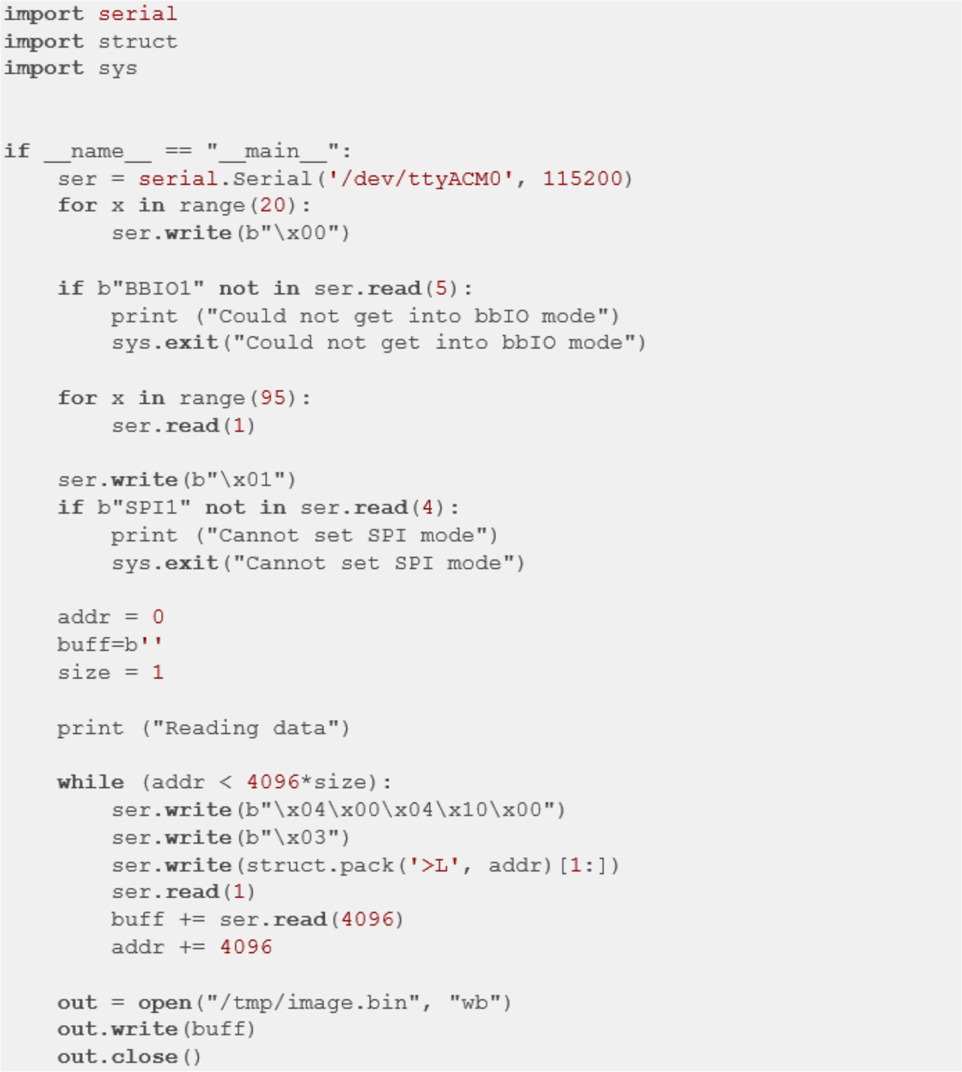


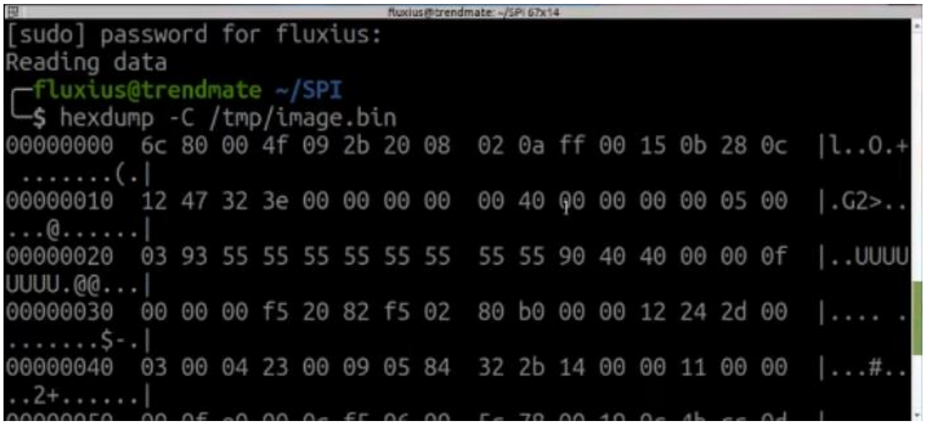
Figure 3. Reading registers of a LoRa transceiver via SPI

We can then adapt and read the SPI example script for HydraBus as follows:



```
import serial
import struct
import sys


if __name__ == "__main__":
    ser = serial.Serial('/dev/ttyACM0', 115200)
    for x in range(20):
        ser.write(b"\x00")

    if b"BBIO1" not in ser.read(5):
        print ("Could not get into bbIO mode")
        sys.exit("Could not get into bbIO mode")

    for x in range(95):
        ser.read(1)

    ser.write(b"\x01")
    if b"SPI1" not in ser.read(4):
        print ("Cannot set SPI mode")
        sys.exit("Cannot set SPI mode")

    addr = 0
    buff=b''
    size = 1

    print ("Reading data")

    while (addr < 4096*size):
        ser.write(b"\x04\x00\x04\x10\x00")
        ser.write(b"\x03")
        ser.write(struct.pack('>L', addr)[1:])
        ser.read(1)
        buff += ser.read(4096)
        addr += 4096

    out = open("/tmp/image.bin", "wb")
    out.write(buff)
    out.close()
```

And get the content as follows:

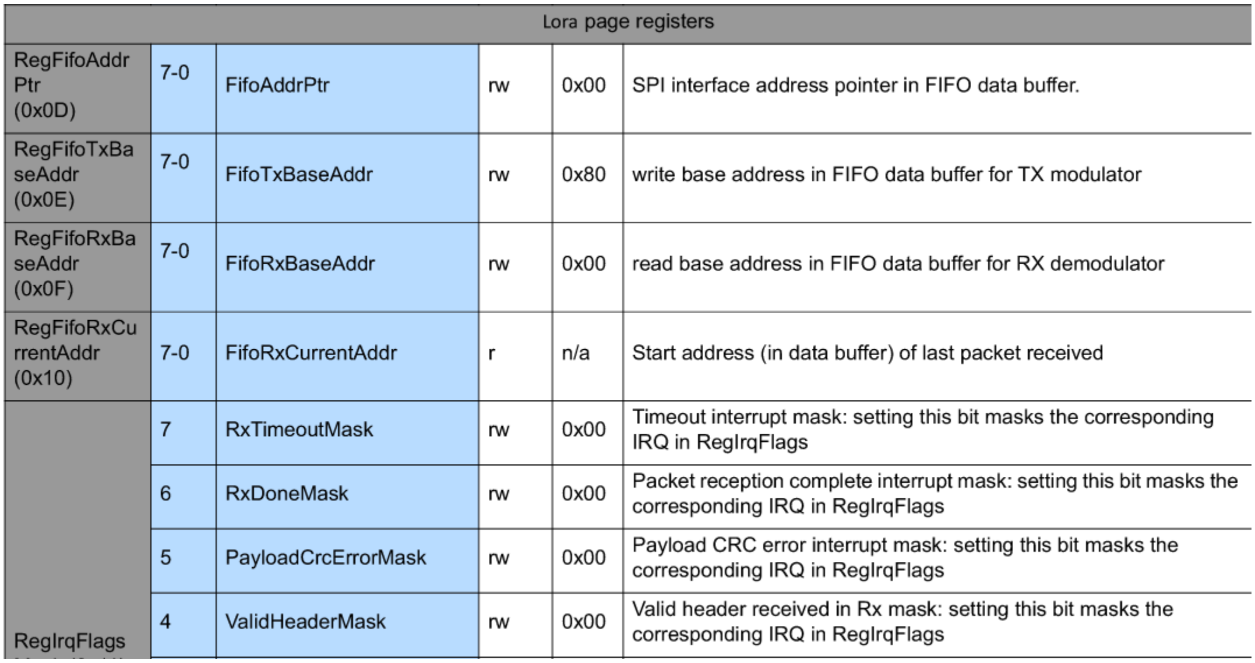


```
[sudo] password for fluxius:
Reading data
┌─fluxius@trendmate ~/SPI
└─$ hexdump -C /tmp/image.bin
00000000  6c 80 00 4f 09 2b 20 08  02 0a ff 00 15 0b 28 0c  |l..O.+
 .......(.|
00000010  12 47 32 3e 00 00 00 00  00 40 00 00 00 00 05 00  |.G2>..
...@......|
00000020  03 93 55 55 55 55 55 55  55 55 90 40 40 00 00 0f  |..UUUU
UUUU.@@...|
00000030  00 00 00 f5 20 82 f5 02  80 b0 00 00 12 24 2d 00  |.... .
.......$-.|
00000040  03 00 04 23 00 09 05 84  32 2b 14 00 00 11 00 00  |...#..
..2+......|
00000050  00 0f a0 00 0c ff 06 00  5c 78 00 10 0c 4b cc 0d  |
```

The transceiver does not handle sensitive information, it only has information on some registers[6] that allow it to configure and work with different modes. The following tables show the information:

Name (Address)	Bits	Variable Name	Mode	Reset	LoRa Description
RegFifo (0x00)	7-0	Fifo	rw	0x00	LoRa base-band FIFO data input/output. FIFO is cleared an not accessible when device is in SLEEP mode
Common Register Settings					
RegOpMode (0x01)	7	LongRangeMode	rw	0x0	0 → FSK/OOK Mode 1 → LoRa Mode This bit can be modified only in Sleep mode. A write operation on other device modes is ignored.
	6	AccessSharedReg	rw	0x0	This bit operates when device is in Lora mode; if set it allows access to FSK registers page located in address space (0x0D:0x3F) while in LoRa mode 0 → Access LoRa registers page 0x0D: 0x3F 1 → Access FSK registers page (in mode LoRa) 0x0D: 0x3F
	5-4	reserved	r	0x00	reserved
	3	LowFrequencyModeOn	rw	0x01	Access Low Frequency Mode registers 0 → High Frequency Mode (access to HF test registers) 1 → Low Frequency Mode (access to LF test registers)
	2-0	Mode	rwt	0x01	Device modes 000 → SLEEP 001 → STDBY 010 → Frequency synthesis TX (FSTX) 011 → Transmit (TX) 100 → Frequency synthesis RX (FSRX) 101 → Receive continuous (RXCONTINUOUS) 110 → receive single (RXSINGLE) 111 → Channel activity detection (CAD)
(0x02)	7-0	reserved	r	0x00	-
(0x03)	7-0	reserved	r	0x00	-
(0x04)	7-0	reserved	rw	0x00	-

Accesses to radio data can be seen through the following page registers:



Lora page registers

RegFifoAddrPtr (0x0D)	7-0	FifoAddrPtr	rw	0x00	SPI interface address pointer in FIFO data buffer.
RegFifoTxBaseAddr (0x0E)	7-0	FifoTxBaseAddr	rw	0x80	write base address in FIFO data buffer for TX modulator
RegFifoRxBaseAddr (0x0F)	7-0	FifoRxBaseAddr	rw	0x00	read base address in FIFO data buffer for RX demodulator
RegFifoRxCurrentAddr (0x10)	7-0	FifoRxCurrentAddr	r	n/a	Start address (in data buffer) of last packet received
RegIrqFlags	7	RxTimeoutMask	rw	0x00	Timeout interrupt mask: setting this bit masks the corresponding IRQ in RegIrqFlags
	6	RxDoneMask	rw	0x00	Packet reception complete interrupt mask: setting this bit masks the corresponding IRQ in RegIrqFlags
	5	PayloadCrcErrorMask	rw	0x00	Payload CRC error interrupt mask: setting this bit masks the corresponding IRQ in RegIrqFlags
	4	ValidHeaderMask	rw	0x00	Valid header received in Rx mask: setting this bit masks the corresponding IRQ in RegIrqFlags

Although this is not sensitive information, it shows that a transceiver can be set up for various purposes.

Microcontroller

The microcontroller operates as it implements the LoRaWAN stack.[7] All the keys and calculations that will encrypt and send packets or decrypt received packets from the transceiver are performed on this component.

However, some interfaces can be exposed, as seen with the UART interface in the magnet door sensor. We can directly interact with it and dump secrets if the interface is enabled and no authentication mechanisms are applied to the access portal:

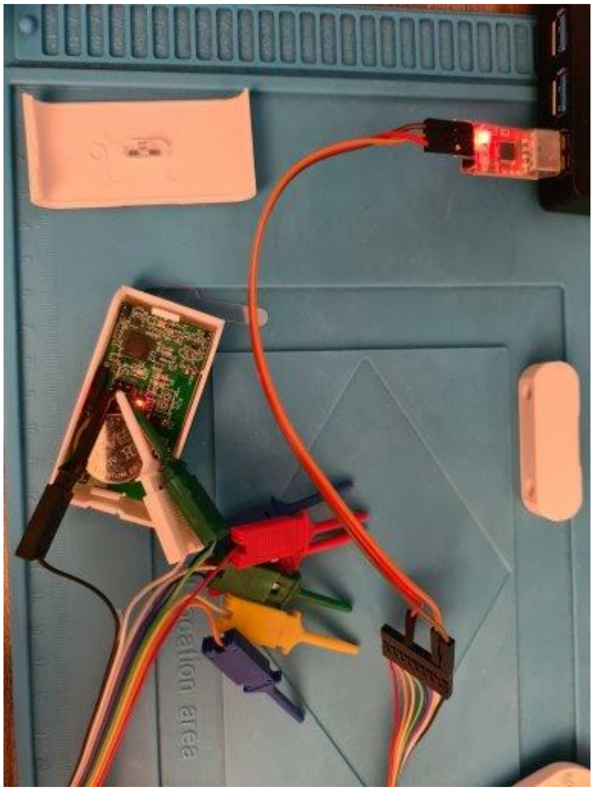


Figure 4. Interfacing with the UART of a magnetic door LoRaWAN sensor

In other contexts, we would have to find other vectors, such as JTAG, SPI, I^2C, in-circuit serial programming (ICSP), or other access methods.

To find access to JTAG or an ICSP interface, we must check if the programming interface is enabled and if the memory is protected somehow. This requires debugging probes that are associated with the device. There are generic debugging adapters that are associated with OpenOCD and can support various communication protocols. Some examples of these generic adapters are: HydraBUS,[8] BUS Pirate,[9] BusVoodoo,[10] J-Link,[11] and others. From an attacker's point of view, glitching attacks can be engaged to get partial or full access to the sensitive data inside the internal flash memory if memory protections are used.

External flash memory

In case the microcontroller uses an external flash, it would be possible for an attacker to interface with this memory through the exposed accesses ports easily (SPI, I^2C, etc.), depending on the surface mount technology (SMT) package.

In certain cases, the attacker can actually chip-off or remove the flash memory from the printed circuit board (PCB) and dump the firmware (dumping is the process of copying or extracting the firmware image):

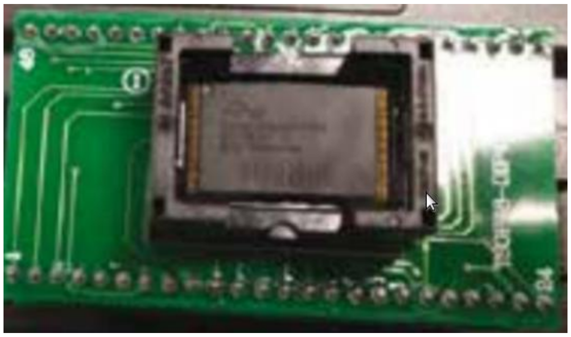


Figure 5. Example of a TSOP package flash to be read using an adapted socket and a programmer

But when it comes to dumping flashes using BGA (Ball Grid Array), the process can be more time consuming for attackers. As seen in Figure 6, the balls of the chip must be aligned with the PCB interface for the attacker to be able to dump the firmware successfully:

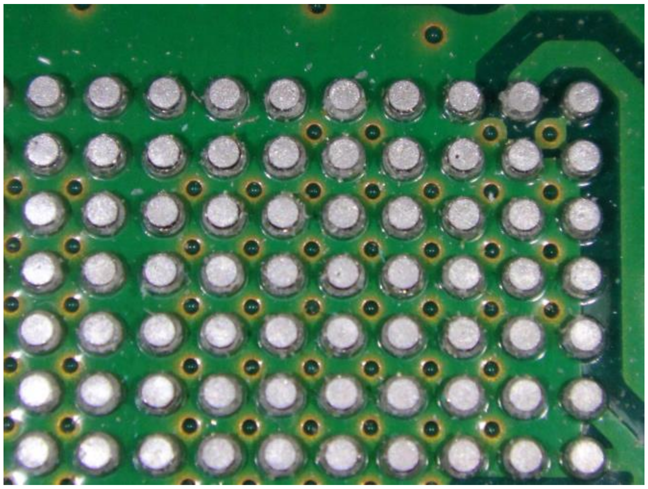


Figure 6. A grid array of solder balls on a printed circuit[12]

The use of a Secure Element

There are many different types of attacks that can be launched against microcontrollers, but the use of Secure Elements (SE) can derail attack attempts to some degree. In the case of LoRaWAN, the SE can "safely" store the master keys that can be derived to encrypt communication and protect message integrity.

The LoRaWAN-node stack also has examples of implementations containing a few integrated SEs.[13] But, to study these Secure Elements, we have taken the approach documented by the team mbed in a blog post written by researcher Jan Jongboom.[14] Figure 7 shows an example of the setup:

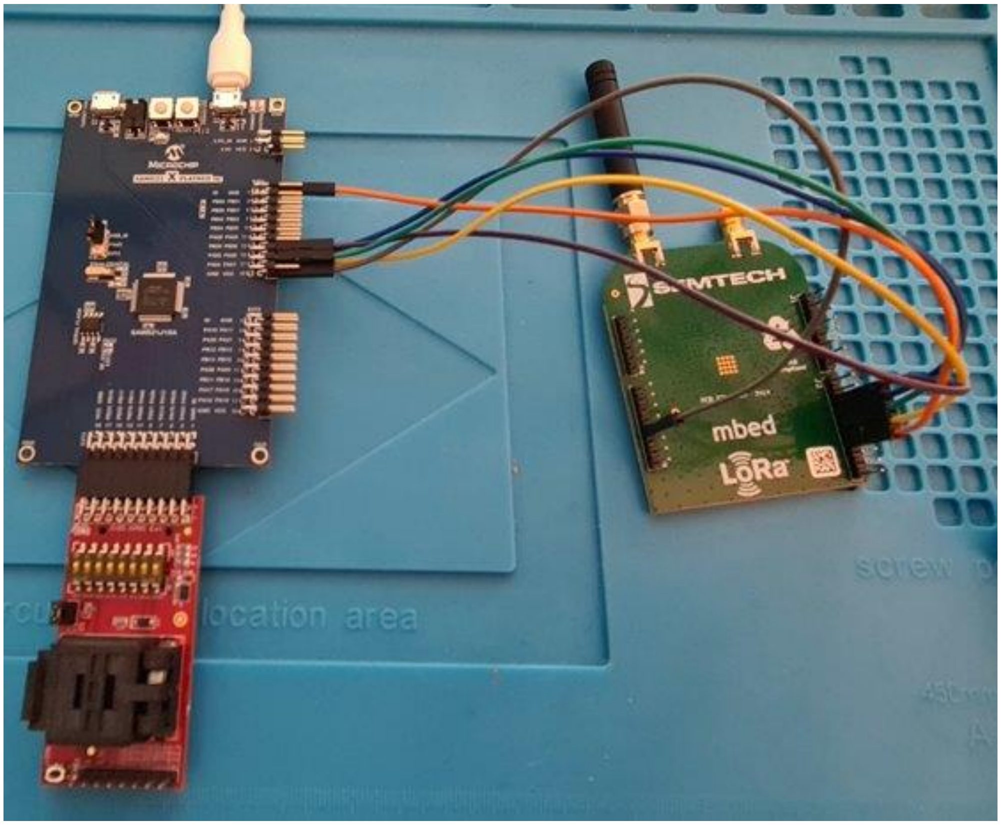


Figure 7. Setup using a LoRa transceiver, a MCU dev kit connected to a UDFN socket kit with a Secure Element Inside

Exactly like in the mbed blog post, we are using the following components:

- 1 x ATSAMD21J1
- 1 x AT88CKSCKTUDFN-XPRO
- 1x SX1276MB1MAS

The Secure Element is a pre-provisioned ATECC608A-MAHTN-T put inside the UDFN socket as follows:

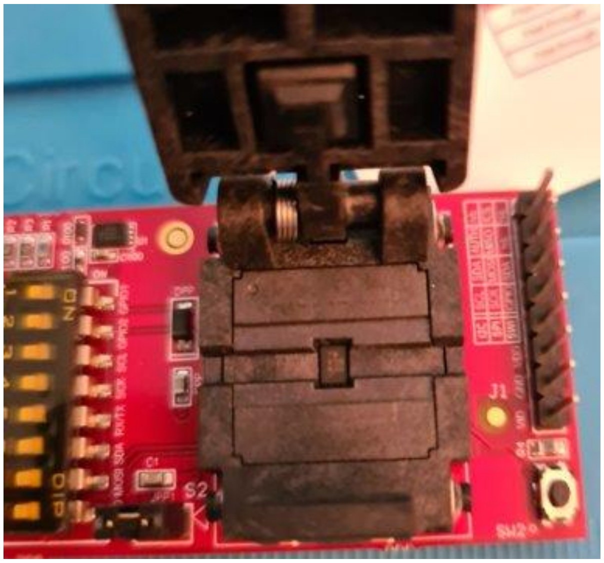


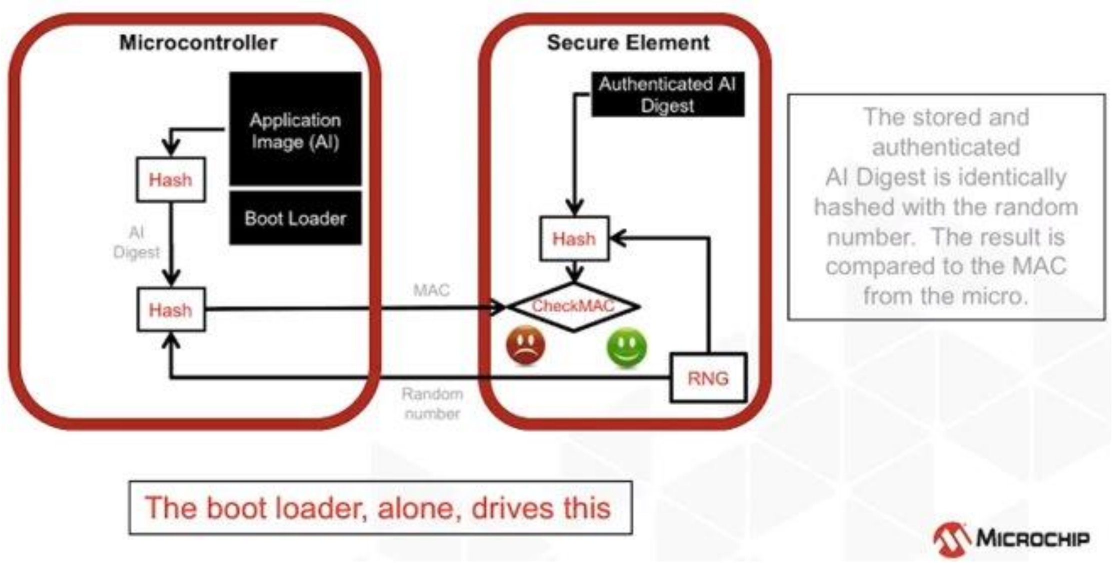
Figure 8. The Secure Element attached

The Github repository of mbed also has a good example of code for using the ATECC608A-MAHTN-T[15] with the mbed-OS and libraries efficiently. Using the provided code, we can securely talk in I2C using the *CryptoLib* library with the UFDN socket:

```
int main(void)
{

    // Setup secure element
    atecc608_i2c_config.iface_type = ATCA_I2C_IFACE;
    atecc608_i2c_config.atcai2c.baud = 100000;
    atecc608_i2c_config.atcai2c.bus = 2;
    atecc608_i2c_config.atcai2c.slave_address = 0xb2;
    atecc608_i2c_config.devtype = ATECC608A;
    atecc608_i2c_config.rx_retries = 20;
    atecc608_i2c_config.wake_delay = 1500;
    //select_device(ATECC608A);
    atcab_init(&atecc608_i2c_config);

    uint8_t serialnum[ATCA_SERIAL_NUM_SIZE];
    ATCA_STATUS serialnum_status =
atcab_read_serial_number(serialnum);
    if (serialnum_status != ATCA_SUCCESS) {
        printf(" Failed to read ATECC608A serial number (%d) \r\n",
serialnum_status);
        return 1;
    }
[…]
```

Unfortunately for the developers, the *Secure Boot* is supported by the ATECC608A-MAHTN-T, but not implemented in the example. So, if a device is using this implementation with a Secure Element, it would be easy for an attacker to use the same Secure Element (with everything stored inside it) in another MCU.

Note that this Secure Element is provided with a *Secure Boot* feature to authenticate the MCU, as shown in one of the Microchip presentations:



The stored and authenticated AI Digest is identically hashed with the random number. The result is compared to the MAC from the micro.

The boot loader, alone, drives this

Figure 9. How to use the Secure Element with the MCU

There are very few documented instances of Secure Elements being used in products. Unfortunately, this means that it is not easy for a developer to be aware of all the beneficial security features that come with using these SEs.

Recommendations

This is the last in our series of articles on LoRaWAN security, and we have compiled some recommendations for enterprises and organizations that use this technology. Following the suggestions below could help prevent attacks against the essential IoT solutions that rely on LoRaWAN:

Attacks	Recommendations
Eavesdropping	• Use strong and randomly generated NwkKeys and AppKeys • Use the ABP method only in experimental environments, but not in production
Bit-Flipping	• Communication between the gateway and the network server should be encrypted; an option would be using MQTT with SSL when using TheThingNetwork solutions • Communication between the network server and the applications server should

	also be encrypted and strong keys must be in use
ACK spoofing	• Add a cryptographic checksum to confirm that the ACK value belongs to the right packet[16]
DoS with counter overflow	• Use version 1.1.x of LoRaWAN that would make this attack less viable with a 32-bit long counter instead of a 16-bit counter (used in v1.0)
LoRa class B attacks	• Change the LoRA PHY cyclic redundancy check (CRC) with a message integrity check (MIC) — this is a complex task usually done by a developer
Root key management	• Do not expose public management interfaces, and use a client certificate • Audit the management interfaces
Hardware attacks	• Choose an MCU resistant to known physical attacks • Configure memory protections (fuse bits) to prevent an attacker from dumping the memory through programming interfaces
Secure Element reuse	• Configure the Secure Boot of the Secure Element to authenticate the MCU that uses it

These recommendations apply to devices using LoRaWAN. However, it should also be mentioned that devices using only the LoRA modulation and a custom stack are highly vulnerable to radio attacks if no confidentiality and integrity mechanisms are used.

Conclusion

Hardware attacks could be complementary to the radio frequency attacks discussed in the previous entries of this series. Indeed, by dumping keys, an attacker could replay them in the wild and find keys that are also used in other devices. The attacker could also eavesdrop on communications by letting the dumped device communicate with a network and then capturing the sensitive information exchanged between the end-device and the network.

Given the range of these attacks, users should not only be careful about the version of the LoRaWAN protocol and the strength of the keys being used, but also the security mechanisms of the MCU. Using devices proven to be resistant to known and accessible glitching attacks will help prevent attackers from reading the memory. And, if users want to use Secure Elements, they should also use security mechanisms — the Secure Boot used to authenticate the MCU to the SE is an important feature.

This three-part series was created to help users operate LoRaWAN devices securely and safely, so that their processes and communications can continue without complications while ensuring that their data is safely stored. Hopefully, the actionable recommendations in this part, as well as the first and second articles of the series, can serve as an effective guide on security solutions for their LoRaWAN technology.

References

[1] Sébastien Dudek. (Jan. 26, 2021). *Trend Micro*. “Low Powered and High Risk: Possible Attacks on LoRaWAN Devices.” Accessed on March 29, 2021, at https://www.trendmicro.com/en_us/research/21/a/Low-Powered-but-High-Risk-Evaluating-Possible-Attacks-on-LoRaWAN-Devices.html.

[2] Sébastien Dudek. (Feb. 19, 2021). *Trend Micro*. “Gauging LoRaWAN Communication Security with LoraPWN.” Accessed on March 29, 2021, at https://www.trendmicro.com/en_us/research/21/b/gauging-lorawan-communication-security-with-lorapwn.html.

[3] SemTech. (n.d.). *Semtech*. “LoRa® Transceivers.” Accessed on March 29, 2021, at https://www.semtech.com/products/wireless-rf/lora-transceivers.

[4] Piyu Dhaker. (Sep. 2018). *Analog Dialogue*. “Introduction to SPI Interface.” Accessed on March 29, 2021, at https://www.analog.com/en/analog-dialogue/articles/introduction-to-spi-interface.html#.

[5] HydraBus. (n.d.). *HydraBus*. “HydraBus v1.0 Rev1.4 Batch March 2021 Production Assembly Video.” Accessed on March 29, 2021, at https://hydrabus.com/.

[6] SemTech. (May 2020). *Semtech*. “SemTech SX1276-7-8-9 Datasheet.” Accessed on March 29, 2021, at https://semtech.my.salesforce.com/sfc/p/#E0000000JelG/a/2R0000001Rbr/6EfVZUorrpoKFfvaF_Fkpgp5kzjiNyiAbqcpqh9qSjE.

[7] Miguel Luis et al. (March 17, 2021). *GitHub*. “LoRaWAN end-device stack implementation and example projects.” Accessed on March 29, 2021, at https://github.com/Lora-net/LoRaMac-node.

[8] HydraBus. (n.d.). *HydraBus*. “HydraBus v1.0 Rev1.4 Batch March 2021 Production Assembly Video.” Accessed on March 29, 2021, at https://hydrabus.com/.

[9] Dangerous Prototypes. (n.d.) *Dangerous Prototypes*. “Bus Pirate.” Accessed on March 29, 2021, at http://dangerousprototypes.com/docs/Bus_Pirate.

[10] BusVoodoo. (n.d.). *BusVoodoo*. “BusVoodoo.” Accessed on March 29, 2021, at https://bus.cuvoodoo.info/.

[11] Segger. (n.d.). *Segger*. “J-Link Debug Probes: The Number One Choice.” Accessed on March 29, 2021, at https://www.segger.com/products/debug-probes/j-link/.

[12] Electronics Notes. (n.d.). *Electronics Notes*. “Ball Grid Array, BGA.” Accessed on March 29, 2021, at https://www.electronics-notes.com/articles/electronic_components/surface-mount-technology-smd-smt/ball-grid-array-bga.php.

[13] Miguel Luis. (May 26, 2020). *GitHub*. “LoRaMac Secure Element Integration.” Accessed on March 29, 2021, at https://github.com/Lora-net/LoRaMac-node/wiki/secure-element.

[14] Jan Jongboom. (Jan. 31, 2019). *Arm Mbed*. “Introducing hardware crypto for LoRaWAN.” Accessed on March 29, 2021, at https://os.mbed.com/blog/entry/introducing-lorawan-hw-crypto/.

[15] Jan Jongboom. (Jan. 30, 2019). *GitHub*. “LoRaWAN example application for Mbed OS using the ATECC608A-MAHTN-T secure element.” Accessed on March 29, 2021, at https://github.com/armmbed/mbed-os-example-lorawan-atecc608a.

[16] Xueying Yang et al. (2018). *2018 IEEE/ACM Third International Conference on Internet-of-Things Design and Implementation*. “Security Vulnerabilities in LoRaWAN.” Accessed on March 29, 2021, at https://www.cyber-threat-intelligence.com/publications/IoTDI2018-LoraWAN.pdf.

TREND MICRO^{TM} RESEARCH

Trend Micro, a global leader in cybersecurity, helps to make the world safe for exchanging digital information.

Trend Micro Research is powered by experts who are passionate about discovering new threats, sharing key insights, and supporting efforts to stop cybercriminals. Our global team helps identify millions of threats daily, leads the industry in vulnerability disclosures, and publishes innovative research on new threats techniques. We continually work to anticipate new threats and deliver thought-provoking research.

www.trendmicro.com