



Internet of Things - Security A.A. 2022/2023
Prof. Giuseppe Piro, PhD

Antonio Petrosino, PhD Student
Giancarlo Sciddurlo, PhD Student
Francesco Vista, PhD Student

LoRaWAN™ Security

Indice

1	LoRaWAN™	2
2	Vulnerabilità e attacchi	2
3	Prerequisiti	4
3.1	Hardware necessario	4
3.2	Installazione plugin PyMakr	6
3.3	Applicazione configurata su The Things Network	7
4	Implementazione attacchi fisici	9
4.1	Corruzione della chiave	9
4.2	Impersonificazione	11
5	Conclusioni	13

1 LoRaWAN™

LoRa è una delle tecnologie LPWAN (Low Power Wide Area Network) più promettenti per l'implementazione basata su Internet of Things (IoT) applicazioni. LoRaWAN è il protocollo di controllo dell'accesso al mezzo (MAC) in cima al livello fisico wireless LoRa. Le specifiche del livello MAC di LoRaWAN sono mantenute e pubblicate apertamente dalla LoRa Alliance, cioè un consorzio di produttori di hardware, fornitori di software e operatori di rete.

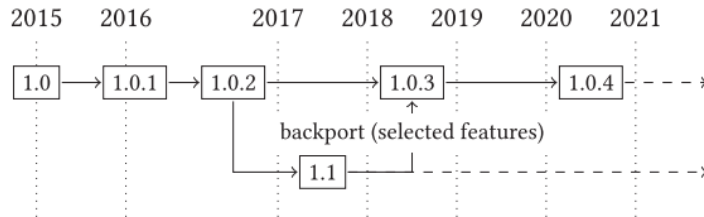


Figura 1: Timeline delle specifiche LoRaWAN.

Pur essendo una tecnologia popolare, diversi lavori in letteratura hanno rivelato vulnerabilità e rischi riguardanti la sicurezza di LoRaWAN 1.0. Dalla sua prima pubblicazione nel 2015, come mostrato in Figura 1, la LoRa Alliance ha ripetutamente rivisto le specifiche LoRaWAN. LoRaWAN 1.0.1 e LoRaWAN 1.0.2 contengono principalmente chiarimenti e correzioni piuttosto che introdurre nuove funzionalità. Oltre a migliorare la sicurezza, LoRaWAN 1.1 finalizza le bozze di funzionalità non specificate della precedente specifica, come le classi aggiuntive di dispositivi, la messaggistica multicast e gli aggiornamenti firmware over-the-air, seguendo le linee guida ETSI [1].

Di seguito verrà riportata una sintesi dei rischi per la sicurezza relativa all'ultima versione di LoRaWAN 1.1, come l'acquisizione fisica del dispositivo finale, il gateway non autorizzato, che richiedono un'attenzione particolare da parte degli sviluppatori e delle organizzazioni che implementano le reti LoRa.

2 Vulnerabilità e attacchi

In questa sezione, discutiamo delle possibili vulnerabilità relative alle entità fisiche e virtuali, distribuzioni delle chiavi e di implementazione.

End-device: La sicurezza fisica degli end-device è essenziale per proteggere le reti da attacchi fisici e per garantire la sicurezza dei dati sensibili. Gli attacchi di cattura dei dispositivi possono compromettere la sicurezza della rete e dei dati in diversi modi, ad esempio attraverso la raccolta di informazioni o l'acquisizione di dati critici. Inoltre, è importante adottare misure di sicurezza fisica per proteggere i dispositivi finali da accessi non autorizzati, ad esempio attraverso l'uso di serrature elettroniche o l'installazione di sensori di allarme. La protezione fisica dei dispositivi finali è un compito continuo e richiede una combinazione di misure hardware e software per garantire la massima sicurezza dei dati.

Implementazione: Queste vulnerabilità si verificano generalmente a causa di errori durante lo sviluppo o l'installazione dell'applicazione, quando i programmatori non seguono

attentamente il documento di specifica a causa della sua complessità o difficoltà nell'applicazione pratica. Prima di procedere con l'accesso ad una rete tramite OTAA, il DevEUI deve essere salvato nell'end-device. Inoltre, gli NS mantengono un registro di tutti i DevEUI utilizzati nella rete. In aggiunta, il JoinEUI deve essere salvato nei dispositivi finali prima di avviare la procedura OTAA. Tuttavia, se il Join Server subisse una modifica che richiederebbe una modifica del JoinEUI, tutti gli end-device precedentemente configurati non funzionerebbero più, poiché non esiste una procedura per il rinnovo del JoinEUI. Di conseguenza, nella versione attuale, il Join Server utilizza lo stesso JoinEUI per tutta la durata della rete.

Name of threats		Likelihood			Impact on						Risk at			
		Technical difficulty	Motivation level	Likelihood	Scale	Detectability	Confidentiality	Integrity	Availability	Authentication & access control	Confidentiality	Integrity	Availability	Authentication & access control
False join packets		None	Low	Unlikely	LoRaWAN	Medium	Minimal	Minimal	Significant	Minimal	Minor	Minor	Minor	Minor
MITM Attack	Bit-flipping / message forgery	None	Low	Unlikely	End-device	Low	Minimal	Moderate	Minimal	Minimal	Minor	Minor	Minor	Minor
	Frame payload attack	None	Low	Unlikely	End-device	Low	Minimal	Moderate	Minimal	Minimal	Minor	Minor	Minor	Minor
Network flooding attack		None	Low	Unlikely	LoRaWAN	High	Minimal	Minimal	Significant	Minimal	Minor	Minor	Minor	Minor
Network traffic analysis		None	Low	Unlikely	LoRaWAN-EN	Low	Moderate	Minimal	Minimal	Minimal	Minor	Minor	Minor	Minor
Physical Attacks	Destroy, remove or steal end-device	None	Medium	Likely	End-device	Medium	None	None	Moderate	None	None	None	Major	None
	Device cloning / Firmware replacement	None	High	Likely	End-device	Low	Moderate	Moderate	Minimal	Significant	Major	Major	Minor	Critical
	Security parameter extraction by phy. access	Solvable	Medium	Possible	End-device	Low	Moderate	Minimal	Minimal	Significant	Major	Minor	Minor	Major
Plaintext key capture		None	Medium	Likely	End-device	Low	Significant	Moderate	Minimal	Significant	Major	Major	Minor	Major
RF jamming		None	Low	Unlikely	LoRaWAN	Low	Minimal	Minimal	Significant	Minimal	Minor	Minor	Minor	Minor
Self-Replay attack		Solvable	High	Likely	End-device	Low	Minimal	Minimal	Significant	Minimal	Minor	Minor	Critical	Minor
Rogue end-device		Strong	High	Likely	End-device	Low	Moderate	Moderate	Moderate	Significant	Major	Major	Major	Critical
Rogue Gateway Attack	Beacon synchronization DoS attack	Solvable	Low	Unlikely	LoRaWAN	Medium	Minimal	Moderate	Significant	Minimal	Minor	Minor	Major	Minor
	Impersonation attack	Solvable	Low	Unlikely	LoRaWAN	Low	Minimal	Minimal	Significant	Minimal	Minor	Minor	Major	Minor
Routing Attacks	Selective forwarding attack	None	Low	Unlikely	LoRaWAN	Low	Minimal	Minimal	Significant	Minimal	Minor	Minor	Minor	Minor
	Sinkhole / Blackhole attack	None	Low	Unlikely	LoRaWAN	Low	Minimal	Minimal	Significant	Minimal	Minor	Minor	Minor	Minor

Figura 2: Descrizione della probabilità di accadere, dell'impatto e del rischio di un attacco contro LoRaWAN 1.1 [2].

In Figura 2 è presente una lista completa di tutti gli attacchi al protocollo LoRaWAN. In questo laboratorio ci concentreremo sugli attacchi di tipo fisico che sfruttano le vulnerabilità relative alle entità di rete e all'implementazione di LoRaWAN, come segue:

- **Distruggere, rimuovere o rubare l'end-device:** in una rete LoRaWAN, è importante sapere che ogni dispositivo ha una chiave univoca generata durante la fabbricazione o prima dell'implementazione, dalle quali derivano le chiavi di sessione. Anche se la chiave radice venisse divulgata, questo non comprometterebbe le informazioni di tutta la rete, ma solo i dati archiviati nel dispositivo specifico in questione.
- **Estrazione dei parametri di sicurezza:** la specifica LoRaWAN richiede la protezione dei parametri di sicurezza affinché non vengano riutilizzati in modo improprio, ma è fondamentale anche proteggere i nodi da eventuali modifiche non autorizzate al firmware, che potrebbero comunque portare al riutilizzo dei parametri di sicurezza.
- **Clonazione del dispositivo o sostituzione del firmware:** nel caso in cui un attaccante avesse accesso fisico al dispositivo, potrebbe sostituire il firmware o rubare/riutilizzare le chiavi. Per questo, è importante proteggere i nodi da eventuali modifiche non autorizzate al firmware, che potrebbero indirettamente portare al riutilizzo dei parametri di sicurezza.

3 Prerequisiti

- Sistema Operativo: Windows 10 o superiore
- IDE: Visual Studio Code (o Atom)
- Hardware necessario: Pysense e FiPy
- Plugin da installare: PyMakr
- Applicazione configurata su The Things Network
- GitHub: importare progetti da <https://github.com/telematics-lab/lopy-fw>

3.1 Hardware necessario

Per il seguente laboratorio è necessario avere a disposizione:

- **PySense** (in Figura 3a) è un modulo di espansione (o shield) per schede Pycom, progettato per fornire una vasta gamma di funzionalità di sensoristica e connettività wireless. Il modulo PySense è dotato di diversi sensori, tra cui un sensore di temperatura, un sensore di umidità relativa, un sensore di pressione barometrica, un sensore di luce ambientale e un accelerometro a 3 assi. Inoltre, dispone di diverse opzioni di connettività wireless, come Wi-Fi, Bluetooth Low Energy e LoRaWAN.

PySense utilizza un microcontrollore STM32 integrato e viene fornito con un'interfaccia I2C per la connessione a schede Pycom compatibili come Pycom WiPy, LoPy o FiPy. L'interfaccia I2C consente di comunicare con il modulo PySense utilizzando solo due pin, rendendo facile l'integrazione con altri componenti hardware.

PySense è ideale per progetti IoT (Internet of Things) che richiedono la rilevazione di dati ambientali come temperatura, umidità, pressione e luminosità, oltre alla trasmissione di dati attraverso diverse tecnologie wireless.

- **FiPy** (in Figura 3b) è una scheda di sviluppo IoT prodotta dalla società Pycom. La scheda FiPy è basata sul microcontrollore ESP32 di Espressif Systems ed è compatibile con la piattaforma di sviluppo MicroPython.

FiPy è dotata di diverse funzionalità di connettività, tra cui Wi-Fi, Bluetooth Low Energy, LoRaWAN, Sigfox e LTE CAT M1/NB-IoT. La scheda supporta anche una vasta gamma di protocolli di comunicazione, tra cui MQTT, HTTP e WebSocket. FiPy dispone di un'antenna integrata per la connessione Wi-Fi, Bluetooth e Sigfox, nonché di un connettore SMA per l'antenna esterna LoRaWAN e LTE.

La scheda FiPy dispone di un'ampia gamma di porte di ingresso/uscita (I/O) per collegare sensori e dispositivi esterni. Tra queste, ci sono 24 pin di I/O digitali, 3 pin di I/O analogici e diverse interfacce seriali e di comunicazione, tra cui SPI, I2C e UART.

Inoltre, FiPy è dotata di una memoria flash di 4 MB e di una memoria RAM di 512 KB, consentendo di gestire e archiviare grandi quantità di dati. La scheda è alimentata da una batteria al litio ricaricabile e supporta la ricarica tramite un connettore micro USB.

Grazie alla sua flessibilità, connettività e potenza, FiPy è una scheda ideale per la prototipazione e lo sviluppo di progetti IoT avanzati che richiedono la trasmissione wireless di dati tramite diversi standard di comunicazione e l'interazione con il mondo fisico attraverso sensori e attuatori.

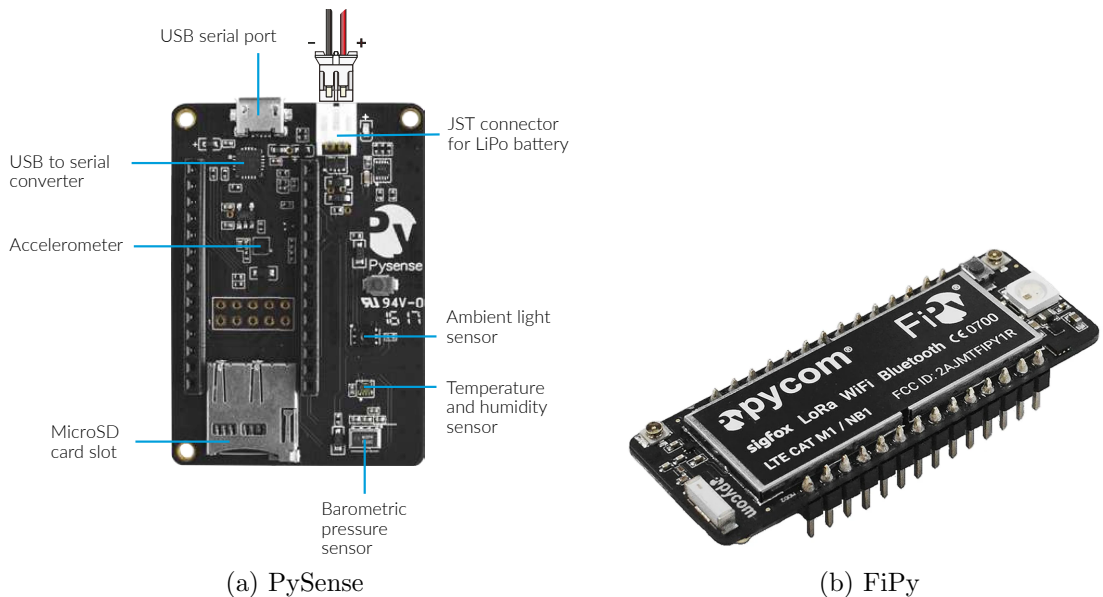


Figura 3: Hardware necessario.

3.2 Installazione plugin PyMakr

Pymakr è un plug-in o un'estensione per l'IDE (Integrated Development Environment), ad esempio VS Code, che è un editor di testo molto popolare tra gli sviluppatori. Pymakr è specificamente progettato per sviluppatori che lavorano con schede di sviluppo basate su microcontrollori come Arduino e Pycom.

Pymakr fornisce un ambiente di sviluppo integrato per scrivere, caricare e debuggare il codice Python su queste schede. Offre funzionalità come il caricamento del codice sulla scheda direttamente dall'IDE, il monitoraggio della console seriale, il debug remoto e la gestione dei file sul dispositivo.

Questo strumento è particolarmente utile per gli sviluppatori che lavorano con microcontrollori Pycom, che sono schede di sviluppo che combinano la potenza del linguaggio di programmazione Python con le funzionalità di connettività come il Wi-Fi, il Bluetooth e la rete cellulare. Pymakr semplifica il processo di sviluppo per questi dispositivi, consentendo agli sviluppatori di scrivere e testare il codice in modo efficiente.

Per installare il plug-in, si deve cliccare sull'icona delle estensioni nel menù di navigazione, come mostrato in Fig. 4 e cercare **pymakr**. A questo punto, cliccando sul bottone **Installa**, come mostrato in Fig. 5, apparirà il bottone di **Ricarica l'Ide**. Una volta ricaricato l'IDE, il plug-in è installato con successo.

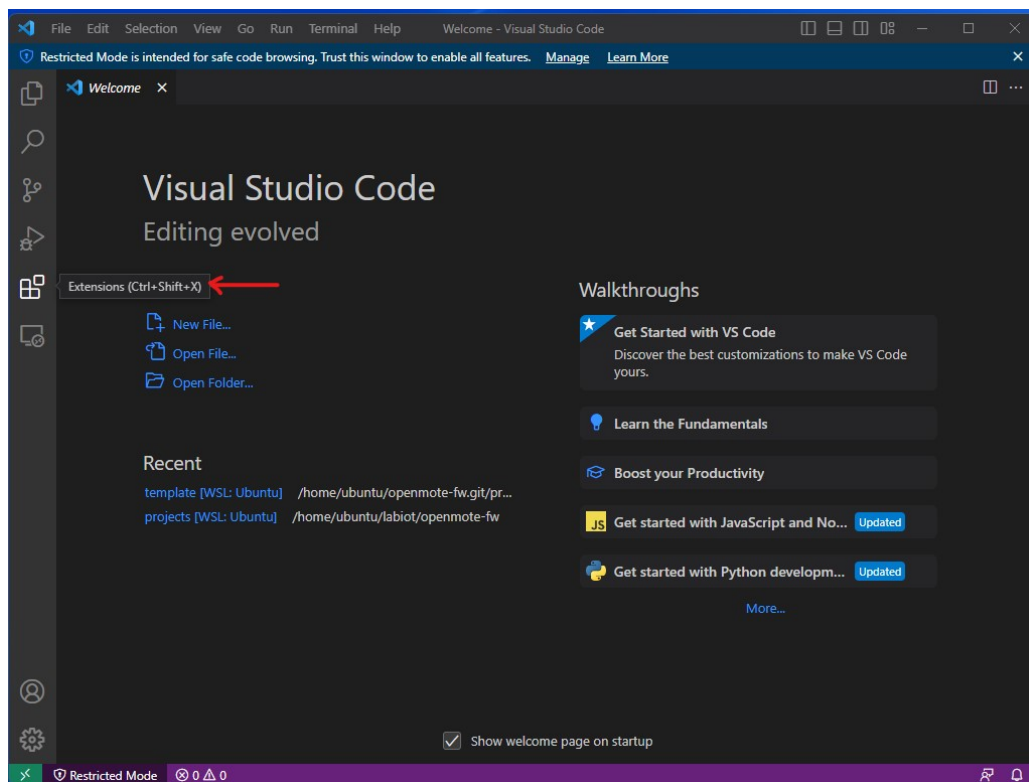


Figura 4: Visual Studio Code Extensions.

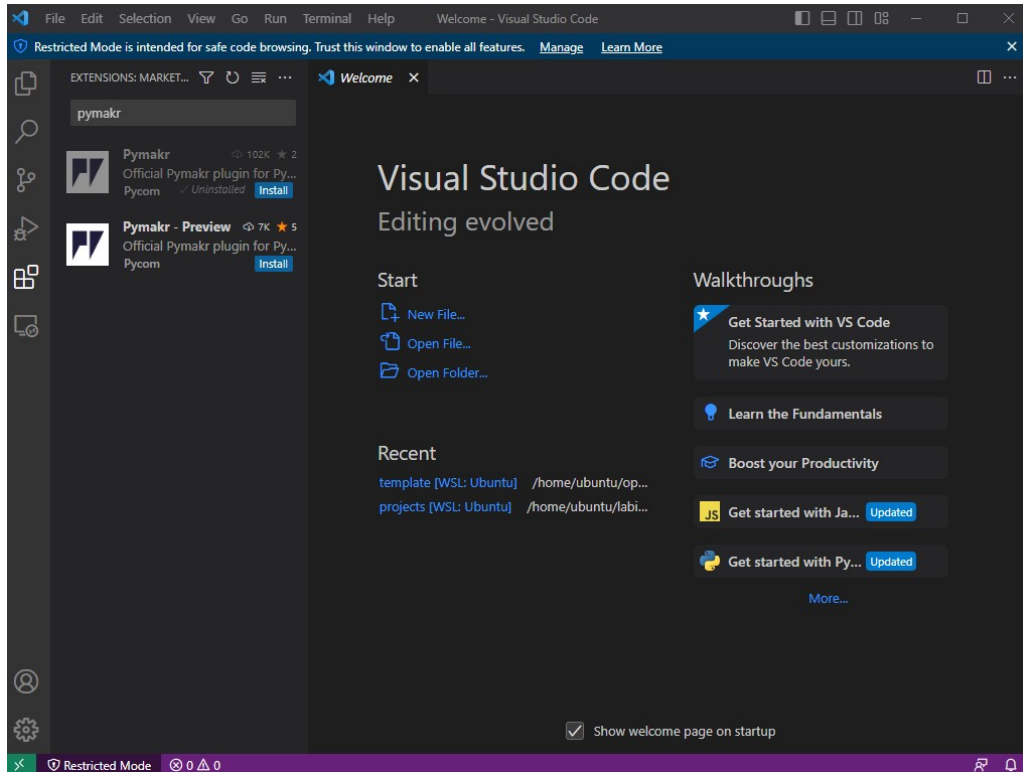


Figura 5: Official Pymakr plugin.

3.3 Applicazione configurata su The Things Network

The Things Stack è un server di rete LoRaWAN sviluppato da The Things Industries. Da gennaio 2021, The Things Network (TTN) gestisce The Things Stack Community Edition, una distribuzione gratuita di The Things Stack basata sulla community. The Things Stack Community Edition offre un modo gratuito e facile per gli sviluppatori di familiarizzare con la tecnologia LoRaWAN, fornendo una rete LoRaWAN pubblica e comunitaria. Gli utenti possono utilizzare la Console con un'interfaccia utente, per gestire gateway, dispositivi, applicazioni, utenti e organizzazioni, nonché per interagire con gli eventi di uplink e downlink in tempo reale.

Per poter connettere i moduli alla rete LoRaWAN è necessario effettuare la registrazione dei dispositivi e del gateway sul server TTN. Prima, tuttavia, bisogna registrarsi al sito thethingsnetwork.org e accedere all'area personale (cliccando su [accesso student](#) e [zona Europe1](#)). Una volta entrati nell'area personale, vi sarà un pulsante apposito per la registrazione di un nuovo gateway (come mostrato in Figura 6). Cliccando su **Register a gateway**, si dovranno inserire i parametri relativi al GATEWAY EUI e GATEWAY ID. Il GATEWAY EUI è un esadecimale univoco (64 bit) che espande l'indirizzo MAC WiFi, mentre il GATEWAY ID è una stringa di caratteri univoci che indentificano il gateway. Di seguito, sono elencati tutti i parametri da inserire necessari per la registrazione del gateway:

- GATEWAY EUI

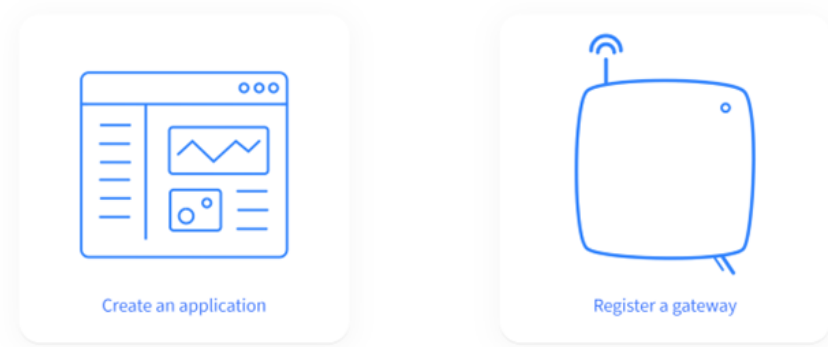


Figura 6: The Things Network Gateway registration.

- GATEWAY ID
- GATEWAY name (a proprio piacimento)
- Frequency plan Europe 863-870 MHz (SF9 for RX2 - recommended)

e procedere con **Register gateway**.

Per poter utilizzare il server TTN è necessario creare una nuova applicazione, in questo modo la rete saprà dove inviare i dati, e registrare i dispositivi. In particolare, per creare l'applicazione bisogna accedere alla sezione di registrazione dell'applicazione, cliccare su **Create Application** e inserire l'**Application ID** univoco e un nome identificativo. Anche in questo caso occorre recuperare dai dispositivi, tramite interfaccia MicroPython, il Dev_EUI. Quindi, aprendo la cartella contenente il codice relativo al dispositivo, digitare su linea di comando:

```
> import machine
> import ubinascii
> WIFI_MAC = ubinascii.hexlify(machine.unique_id()).upper()
> DEV_EUI = WIFI_MAC[:6] + "FFFF" + WIFI_MAC[6:12]
> print("Dev_EUI ", DEV_EUI)
```

Per la registrazione dei nodi sul server bisogna inserire per ognuno dei nodi i seguenti parametri (cliccando su **Enter end device specifics manually**), come in Figura 7:

- Frequency Plan = Europe 863-870 MHz (SF9 for RX2 - recommended);
- Lora version = LoRaWAN Specification 1.0.2;
- Regional Parameters = RP001 Regional Parameters 1.0.2;
- JoinEUI = 0000000000000000; (App_EUI di default tutti zeri)
- DevEUI: identificatore univoco a livello globale per il dispositivo (precedentemente ricavato dalla scheda);

- AppKey è una chiave generata da TTN;
- End Device ID è il nome identificativo univoco del dispositivo e viene precompilato direttamente da TTN.

Provisioning information

JoinEUI ⓘ *

00 00 00 00 00 00 00 00 Reset

This end device can be registered on the network

DevEUI ⓘ *

F0 08 D1 FF FE CC DD 68 Generate 0/50 used

AppKey ⓘ *

97 BB 32 10 09 AE 50 CB AA 76 15 C3 74 3C D1 73 Generate

End device ID ⓘ *

eui-f008d1fffeccdd68

This value is automatically prefilled using the DevEUI

After registration

☒ View registered end device

☐ Register another end device of this type

Register end device

Figura 7: The Things Network device registration.

Ripetere la stessa procedura per gli altri nodi che si desidera installare nell'environment, in particolare per registrare un altro dispositivo nella sezione dell'Applicazione sul Web server, cliccare su **Register end-device** nella scheda dell'applicazione precedentemente creata.

4 Implementazione attacchi fisici

4.1 Corruzione della chiave

L'obiettivo principale di questa fase è valutare possibili falle nella sicurezza dovute ad attacchi fisici ai dispositivi. Nello specifico, si prova a cambiare un numero della tupla rappresentante la chiave di sicurezza per mostrare che tipo di richiesta arriva all'application server.

A questo scopo, prendiamo un dispositivo (nodo che svolge la funzione di sensing) correttamente registrato all'applicazione desiderata sul cloud e, una volta collegato all'interfaccia

seriale, sfruttiamo le capacità del plugin Pymakr per attuare le modifiche oggetto della prova di corruzione della chiave. L'interfaccia grafica di Visual Studio Code (o Atom) riconosce la porta seriale COM del dispositivo. A questo punto creare il terminale di accesso all'interfaccia MicroPython tramite le scelte che Pymakr mette a disposizione e cliccare su *Open Device* in file explorer. Sul file `main.py` del dispositivo, come mostrato in Figura 8, è necessario individuare i parametri OTA per l'autenticazione del dispositivo relativi alla registrazione dell'applicazione (mostrati in Figura 9).

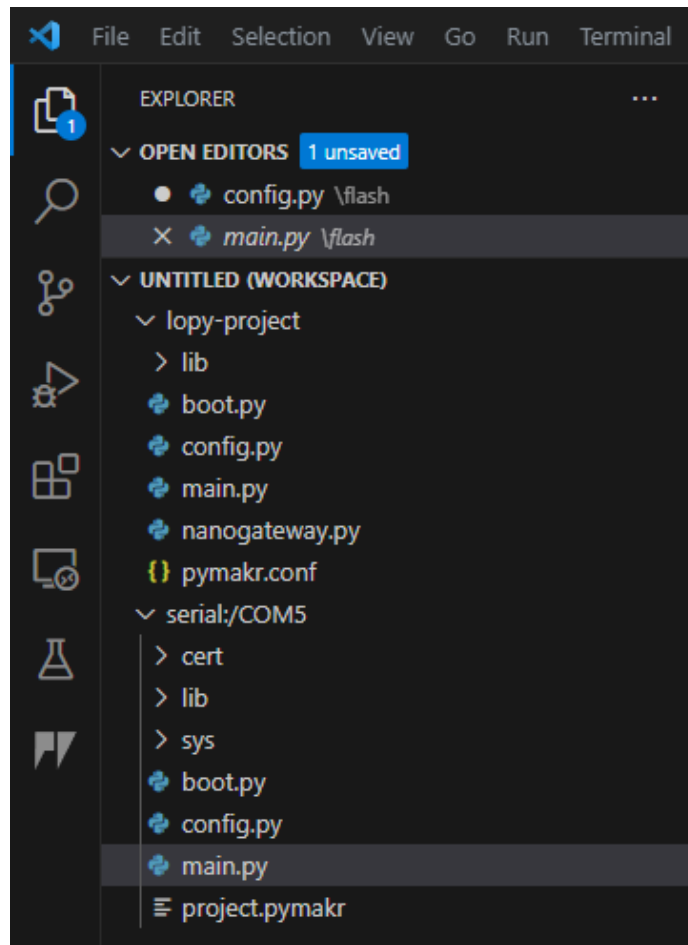


Figura 8: PyMakr.

```
# create an OTA authentication params
dev_eui = binascii.unhexlify('F008D1FFFFCCDD68')
app_eui = binascii.unhexlify('0000000000000000')
app_key = binascii.unhexlify('35ACCFB5A590B8905D503021CCB88B6')
```

Figura 9: PyMakr.

Per la corruzione delle credenziali del nodo è possibile modificare:

- *Dev_Eui*: cambiando l'identificatore univoco a livello globale per il dispositivo precedentemente ricavato dalla scheda, sarà possibile vedere lato server l'arrivo di pacchetti.

Tuttavia l'application server non riuscirà a leggere i pacchetti arrivati poiché riceverà un identificativo non associato a quella relativa applicazione;

- *App_Eui*: modificando la chiave dell'applicazione il pacchetto viene passato dal gateway al server che tenta il decrypt attraverso l'applicazione. Tuttavia il cambio di chiave non consente la lettura corretta del pacchetto ricevuto poiché fallisce l'AES decrypt;
- *App_Key*: cambiando la chiave univoca per dispositivo generata da TTN otteniamo lo stesso risultato della modifica precedente.

Questi esempi di corruzione del dispositivo descritti in questo paragrafo sono esplicativi dell'importanza della chiave. Nel paragrafo successivo si mostra con un ulteriore esempio un eventuale attacco risultante dal furto delle informazioni sensibili, con l'estrazione dei parametri di sicurezza ed conseguente clonazione del dispositivo.

4.2 Impersonificazione

L'attacco di impersonificazione sfrutta due delle vulnerabilità che sono state introdotte nella Sezione 2: l'estrazione dei parametri di sicurezza e la clonazione del dispositivo.

Per eseguire l'attacco, un potenziale aggressore può accedere al firmware di un dispositivo funzionante (cioè registrato correttamente alla rete) utilizzando l'interfaccia seriale. All'interno del firmware sono presenti i dati sensibili, come i parametri di autenticazione OTAA e le funzionalità per cui il dispositivo è stato programmato, ad esempio i sensori utilizzati e i parametri misurati. Successivamente, l'aggressore può clonare il dispositivo utilizzando un altro dispositivo, in cui viene inserito del codice malevolo insieme alle chiavi recuperate in precedenza. Ad esempio, il codice può essere progettato per registrare e inviare una temperatura falsa. In questo modo, il dispositivo clonato può impersonare il dispositivo originale e utilizzare le credenziali e le funzionalità precedentemente estratte per accedere alla rete o eseguire azioni malevole.

L'attacco di impersonificazione descritto rappresenta una minaccia significativa poiché consente all'aggressore di assumere l'identità del dispositivo originale. Ciò può avere diverse conseguenze negative, tra cui la generazione e la manipolazione dei dati inviati alla rete. Ad esempio, se il dispositivo originale è progettato per monitorare temperature critiche e inviare un avviso in caso di superamento di una soglia predefinita, un aggressore potrebbe inviare dati falsi che superano la soglia, causando una risposta non necessaria e spreco di risorse.

Di seguito sono mostrati i vari passi per procedere all'attacco:

- Attraverso l'uso di Visual Studio Code e l'estensione Pymakr, è possibile accedere all'interfaccia seriale di un dispositivo connesso e di visualizzare, modificare o aggiungere file al firmware, inclusi i parametri di sicurezza e altre funzionalità del dispositivo, come indicato in Figura 10.

```
6 import config
7
8 import pycom
9 from pycproc_2 import Pycproc
10 import machine
11
12
13 from LIS2HH12 import LIS2HH12
14 from SI7006A20 import SI7006A20
15 from LTR329ALS01 import LTR329ALS01
16 from MPL3115A2 import MPL3115A2, ALTITUDE, PRESSURE
17
18 pycom.heartbeat(False)
19
20 py = Pycproc()
21 if py.read_product_id() != Pycproc.USB_PID_PYSENSE:
22     raise Exception('Not a Pysense')
23
24 # initialize LoRa in LORAWAN mode.
25 # Please pick the region that matches where you are using the device:
26 # Asia = LoRa.AS923
27 # Australia = LoRa.AU915
28 # Europe = LoRa.EU868
29 # United States = LoRa.US915
30 lora = LoRa(mode=LoRa.LORAWAN, region=LoRa.EU868)
31
32 # create an OTA authentication params
33 dev_eui = binascii.unhexlify('F008D1FFFCBD3AC')
34 app_eui = binascii.unhexlify('0000000000000000')
35 app_key = binascii.unhexlify('BF91E4F7BAB22474CACA03B58233066')
36
37 # set the 3 default channels to the same frequency (must be before sending the OTAA join r
38 lora.add_channel(0, frequency=config.LORA_FREQUENCY, dr_min=0, dr_max=5)
39 lora.add_channel(1, frequency=config.LORA_FREQUENCY, dr_min=0, dr_max=5)
```

Figura 10: main.py del dispositivo da cui rubare l'identità.

- Copiare i parametri di autenticazione, come quelle evidenziate in Figura 9 e in Figura 10.

```
44
45 # create an OTA authentication params
46 dev_eui = binascii.unhexlify('F008D1FFFCBD3AC')
47 app_eui = binascii.unhexlify('0000000000000000')
48 app_key = binascii.unhexlify('BF91E4F7BAB22474CACA03B58233066')
49
50
51
52
```

```
Pycom MicroPython 1.20.2.rc9 [v1.11-1a257d8] on 2020-06-10; FiPy with ESP32
Pybytes Version: 1.5.0
Type "help()" for more information.
>>> import machine
>>> import ubinascii
>>>
>>> WIFI_MAC = ubinascii.hexlify(machine.unique_id()).upper()
>>> DEV_EUI = WIFI_MAC[:6] + "FFFE" + WIFI_MAC[6:12]
>>> print("DEV_EUI", DEV_EUI)
DEV_EUI b'F008D1FFFECD374'
```

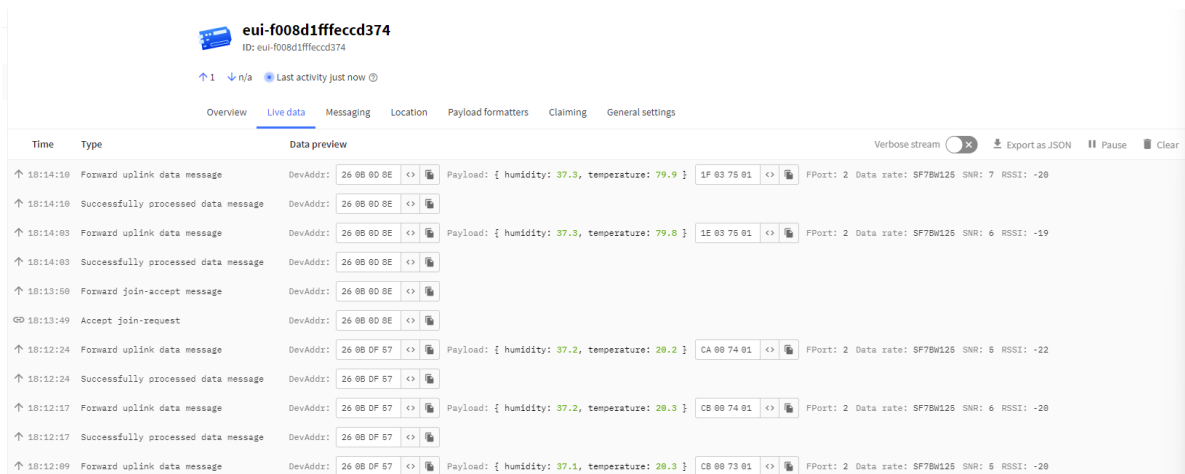
Figura 11: Definizione dei parametri di sicurezza con il DEV_EUI dichiarato in modo errato.

- Nel file main.py si inseriscono i parametri copiati e si modifica la riga relativa alla misura della temperatura, come in Figura 11 e in Figura 12.

```
mp = MPL3115A2(py,mode=ALTITUDE) # Returns height in meters.
# print(mp.temperature())
print("MPL3115A2 temperature: " + str(mp.temperature()))
temp = (mp.temperature()+50)*10
temp= round(temp)
si = SI7006A20(py)
print("Relative Humidity: " + str(si.humidity()) + " %RH")
humidity = si.humidity()*10
humidity = round(humidity)
```

Figura 12: Inserimento di un codice malevolo all'interno del firmware del nuovo dispositivo.

- Vedere su TTN l'arrivo dei pacchetti con le temperature falsificate, come mostrato in figura 13.



Time	Type	DevAddr	Payload	Verbose stream	Export as JSON	Pause	Clear
18:14:10	Forward uplink data message	26 08 00 8E	{ humidity: 37.3, temperature: 79.9 }	1F 83 75 01	FPort: 2 Data rate: SF7BW125 SNR: 7 RSSI: -20		
18:14:10	Successfully processed data message	26 08 00 8E					
18:14:03	Forward uplink data message	26 08 00 8E	{ humidity: 37.3, temperature: 79.8 }	1F 83 75 01	FPort: 2 Data rate: SF7BW125 SNR: 6 RSSI: -19		
18:14:03	Successfully processed data message	26 08 00 8E					
18:13:50	Forward join-accept message	26 08 00 8E					
18:13:49	Accept join-request	26 08 00 8E					
18:12:24	Forward uplink data message	26 08 0F 57	{ humidity: 37.2, temperature: 20.2 }	CA 00 74 01	FPort: 2 Data rate: SF7BW125 SNR: 6 RSSI: -22		
18:12:24	Successfully processed data message	26 08 0F 57					
18:12:17	Forward uplink data message	26 08 0F 57	{ humidity: 37.2, temperature: 20.3 }	CB 00 74 01	FPort: 2 Data rate: SF7BW125 SNR: 6 RSSI: -20		
18:12:17	Successfully processed data message	26 08 0F 57					
18:12:09	Forward uplink data message	26 08 0F 57	{ humidity: 37.1, temperature: 20.3 }	CB 00 73 01	FPort: 2 Data rate: SF7BW125 SNR: 6 RSSI: -20		

Figura 13: Inserimento di un codice malevolo all'interno del firmware del nuovo dispositivo.

5 Conclusioni

Lo scopo di questo laboratorio è quello di dare una descrizione delle vulnerabilità che possono essere sfruttate per attuare degli attacchi all'infrastruttura LoRaWAN. In particolare, sono state sfruttate due vulnerabilità che risultano essere quelle più facili e probabili: l'estrazione dei parametri di sicurezza e la modifica o la clonazione del firmware. Queste vulnerabilità si riflettono nell'implementazione di attacchi fisici.

Entrambi gli attacchi descritti in questo laboratorio possono essere evitati implementando misure di sicurezza adeguate. Queste misure possono includere la crittografia dei dati sensibili, l'implementazione di autenticazione e autorizzazione robuste, la protezione fisica dei dispositivi e dei loro firmware, nonché l'adozione di pratiche di sviluppo sicure per evitare la

presenza di vulnerabilità nel codice del firmware. Inoltre, la sicurezza fisica degli ambienti in cui sono presenti i dispositivi è fondamentale per prevenire l'accesso fisico non autorizzato ai dispositivi e alle loro interfacce.

Riferimenti bibliografici

- [1] F. Hessel, L. Almon, and M. Hollick, "Lorawan security: An evolvable survey on vulnerabilities, attacks and their systematic mitigation," vol. 18, no. 4, 2023. [Online]. Available: <https://doi.org/10.1145/3561973>
- [2] I. Butun, N. Pereira, and M. Gidlund, "Security risk analysis of lorawan and future directions," *Future Internet*, vol. 11, no. 1, p. 3, 2018.