nixu cybersecurity.

It WISN't me, attacking industrial wireless mesh networks

DEF CON 26

Introduction

- Erwin Paternotte
- Lead security consultant
- @stokedsecurity

- Mattijs van Ommeren
- Principal security consultant
- @alcyonsecurity

Previous research

- WirelessHART A Security Analysis, Max Duijsens, Master (2015) - https://pure.tue.nl/ws/files/47038470/800499-1.pdf
- Attacking the plant through WirelessHART, Mattijs & Erwin, S4 Miami (2016) -https://www.youtube.com/watch?v=AIEpgutwZvc
- Denial of service attacks on ICS wireless protocols, Blake Johnson, S4 Miami (2018) – slides/video no longer available

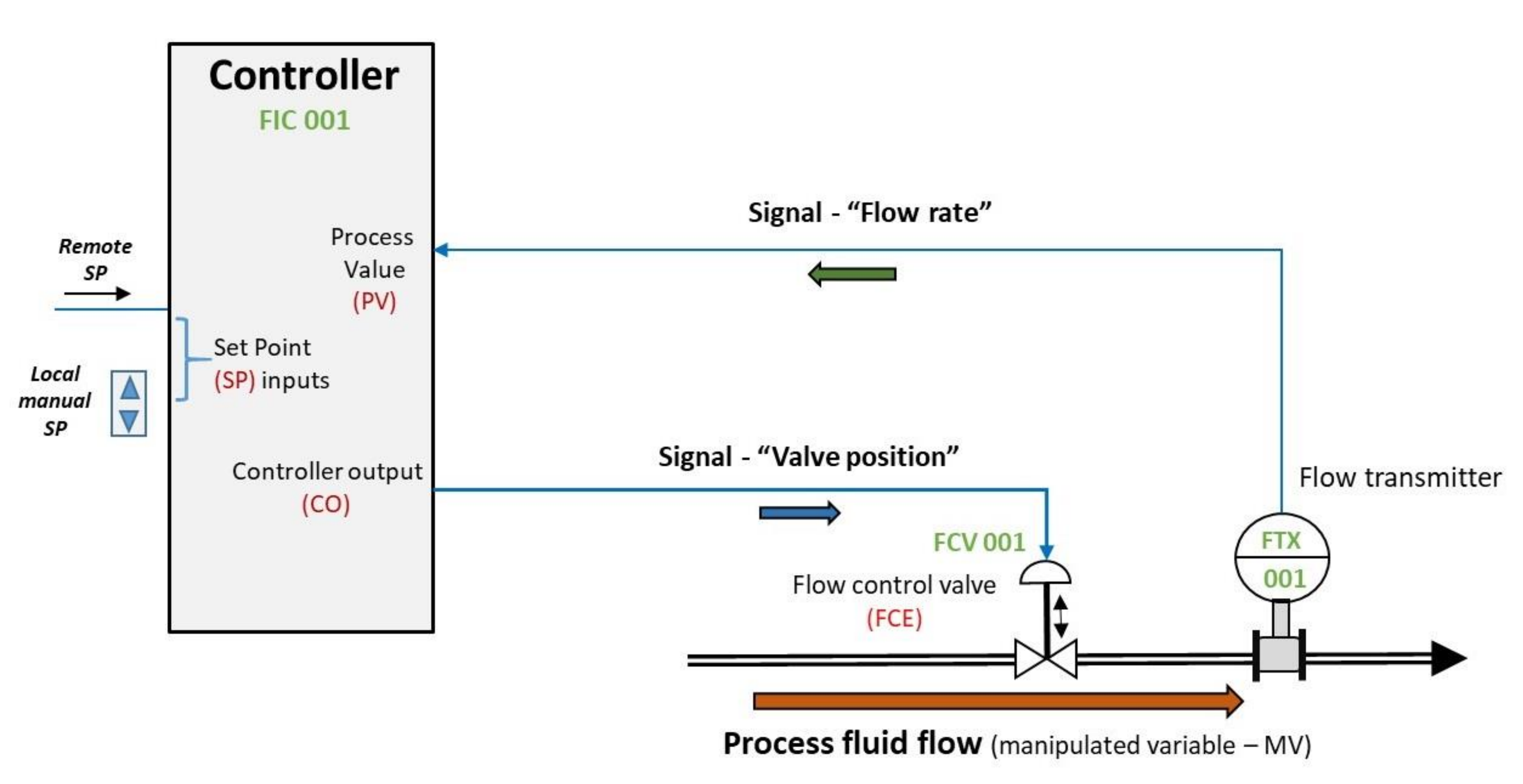
Wright’s principle: “Security does not improve until practical tools for exploration of the attack surface are made available.”

Industrial (r)evolution

A brief history of control systems:

- ~1940: Air: Pneumatic logic systems: 3 - 15 psi
- Mid 1950: Analog: Current loop: 4 - 20 mA
- Mid 1980: Digital: HART, Fieldbus, Profibus
- Late 2000: Wireless mesh networks
  - WirelessHART
  - ISA 100.11a

Industrial process control loop



Introduction to WirelessHART

- Supports HART application layer
- Single encryption cipher/key length (AES CCM*)
- Wireless technology based on Time Synced Mesh Protocol developed by Dust Networks
- Radio SoC exclusively provided by Dust Networks

ABB

PEPPERL+FUCHS

SIEMENS

EMERSON.

Endress+Hauser
People for Process Automation

dust
networks™

Introduction to ISA 100.11a

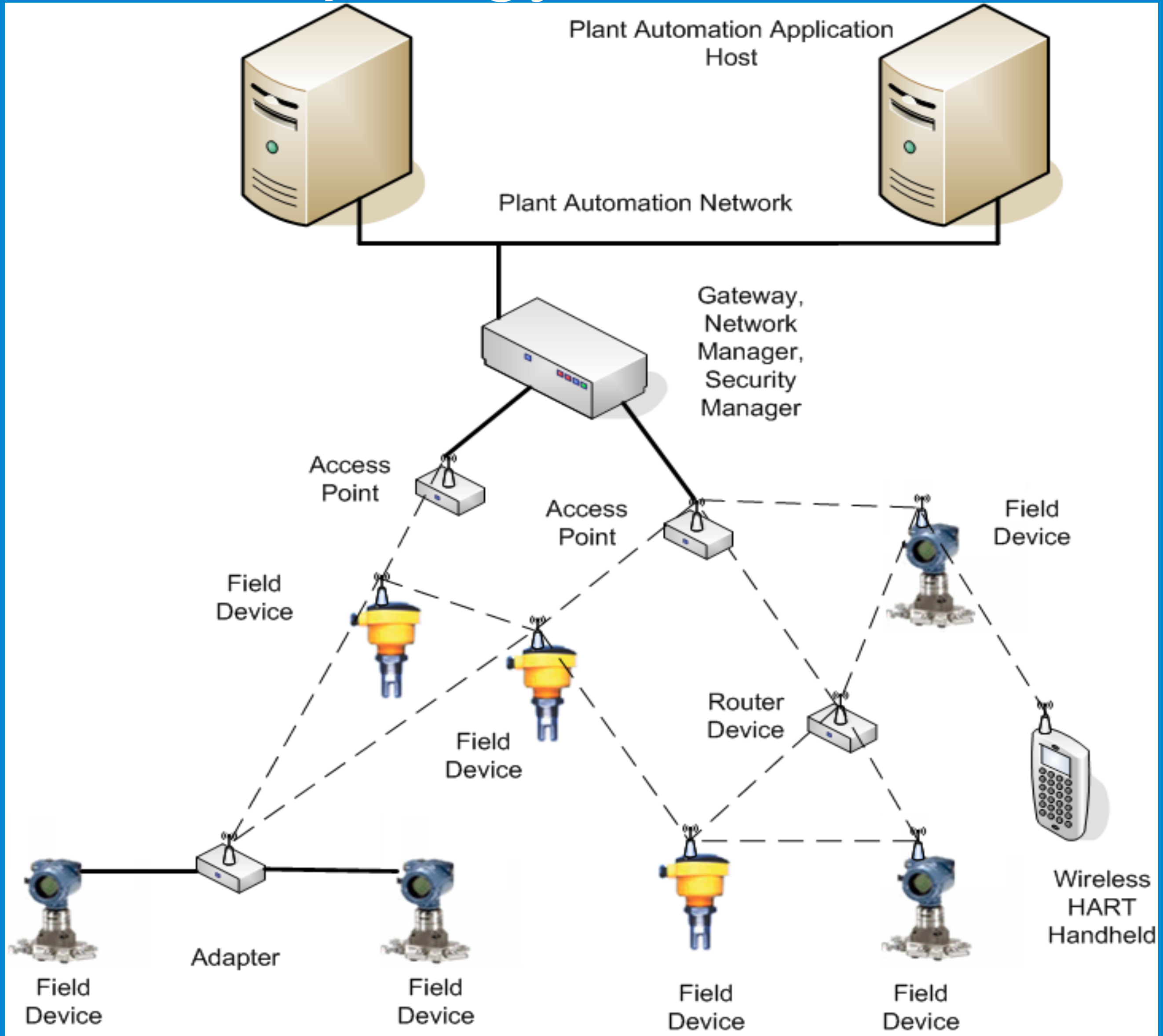
- Relies on several standards: 6LoWPAN/IPv6/UDP
- Ability to tunnel other protocols
- Vendor neutral application layer
- Mainly developed by Nivis
- Generic 802.15.4 chips provided by multiple vendors: STM, NXP, Texas Instruments, OKI

YOKOGAWA

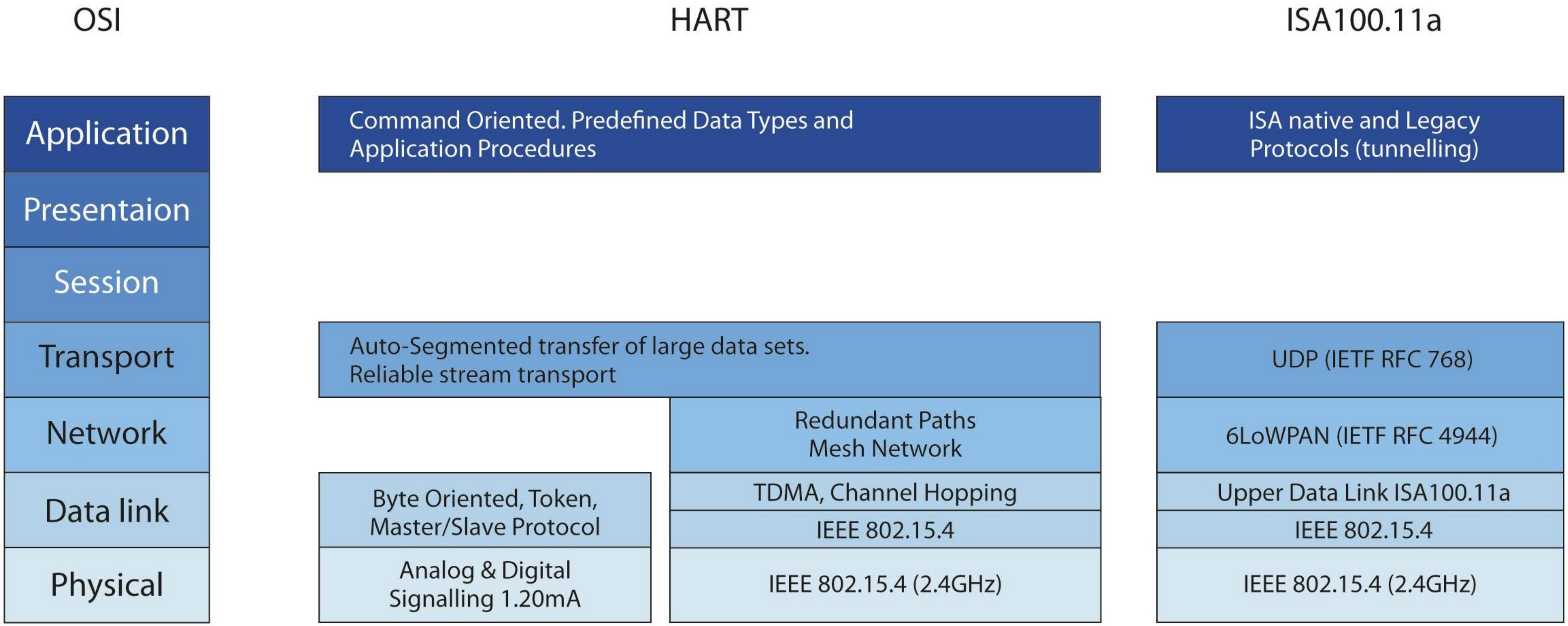
Honeywell

THE POWER OF CONNECTED

WISN topology



Protocol stacks



OSI	HART		ISA100.11a
Application	Command Oriented. Predefined Data Types and Application Procedures		ISA native and Legacy Protocols (tunnelling)
Presentaion			
Session			
Transport	Auto-Segmented transfer of large data sets. Reliable stream transport		UDP (IETF RFC 768)
Network		Redundant Paths Mesh Network	6LoWPAN (IETF RFC 4944)
Data link	Byte Oriented, Token, Master/Slave Protocol	TDMA, Channel Hopping IEEE 802.15.4	Upper Data Link ISA100.11a IEEE 802.15.4
Physical	Analog & Digital Signalling 1.20mA	IEEE 802.15.4 (2.4GHz)	IEEE 802.15.4 (2.4GHz)

Common denominators

- 802.15.4 MAC layer at 2.4 Ghz
- Time Slotted Channel Hopping in order to:
  - Minimize interference with other radio signals
  - Mitigate multipath fading
- Centralized network & security manager orchestrates communication between nodes
- Concluded that developing a common sniffer for both protocols should be possible

WirelessHART & ISA100.11a Security

- AES CCM* (CBC-MAC with counter mode)
  - Network Layer (integrity only)
  - Transport Layer (encryption)
- Join process
  - Handshake with Network Manager
    - Shared secrets
    - Certificates (ISA100.11.a only)

Keys galore

- ISA100.11a
  - **Global Key** – well-known
  - **K_open** – well-known
  - **K_global** – well-known
  - **Master Key** – derived during provisioning
  - **D-Key** – Hop-by-hop integrity
  - **T-KEY** – End-to-end encryption

- WirelessHART
  - **Well-known Key** – Advertisements
  - **Network Key** – Hop-by-hop integrity
  - **Join Key** – Join process
  - **Broadcast Session Key** – End-to-end
  - **Unicast Session Key** – End-to-end

How to obtain key material

- Default keys
  - Documented, more or less
- Sniffing
  - During OTA provisioning (ISA100.11a)
- Keys stored in device NVRAM
  - Recoverable through JTAG/SPI (as demonstrated by our previous research)

WirelessHART default keys

- **445553544E4554574F524B53524F434B** – Multiple vendors
  - DUSTNETWORKSROCK
- **E090D6E2DADACE94C7E9C8D1E781D5ED** – Pepperl+Fuchs
- **24924760000000000000000000000000** – Emerson
- **456E6472657373202B20486175736572** – Endress+Hauser
  - Endress + Hauser

Sniffer hardware selection

- BeamLogic 802.15.4 Site Analyzer
  - 16 channels simultaneously, no injection support, Basic Wireshark dissector, Expensive (~ $1300)
- Atmel RZ Raven
  - Single channel 802.15.4 with standard firmware, no free IDE (Atmel Studio n/a), reached EOL
- NXP BeeKit
  - Single channel 802.15.4 with standard firmware (not open source), reached EOL

NXP USB-KW41Z

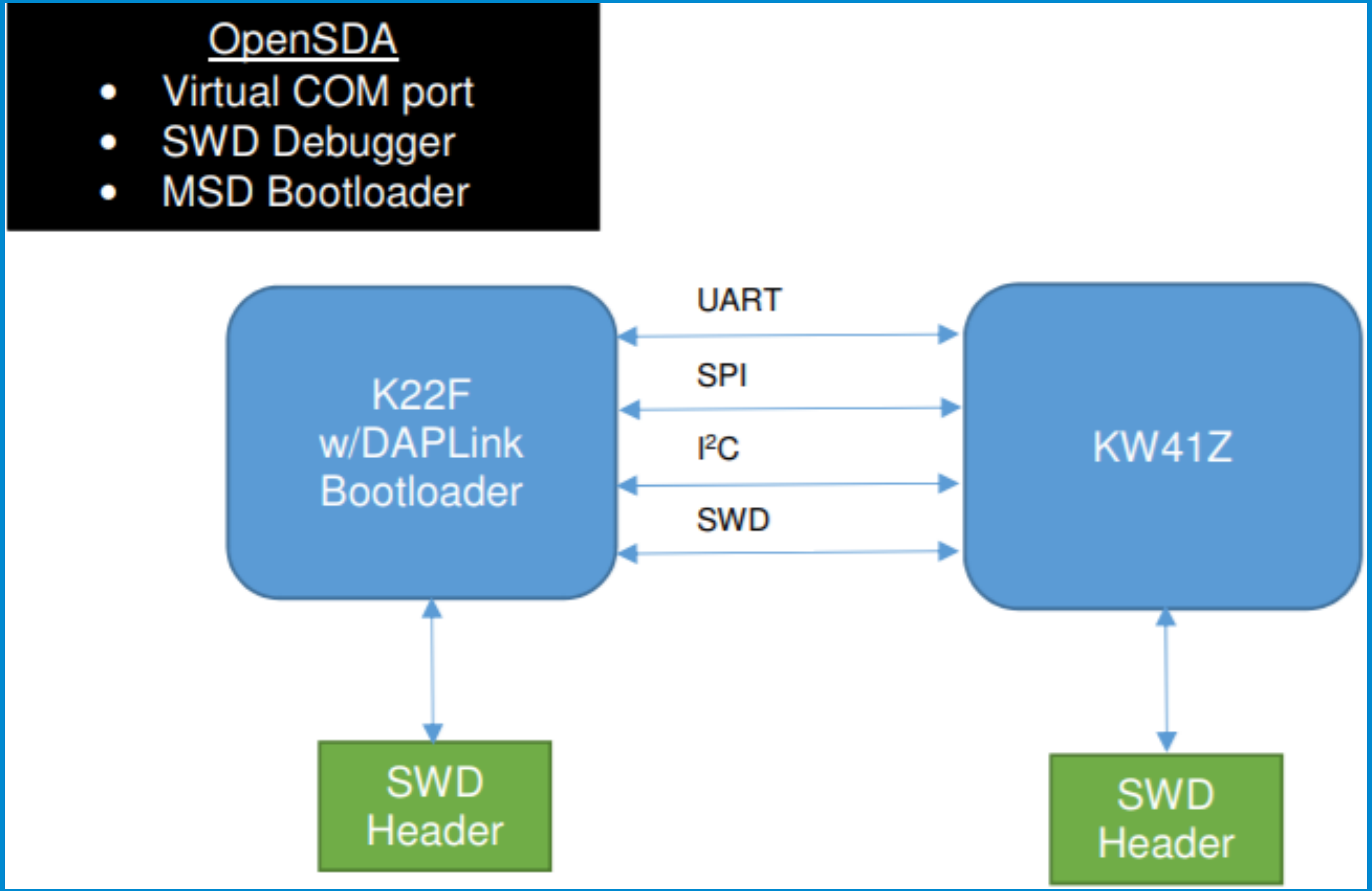
- Single channel 802.15.4 with standard firmware (not open source)
- Actively supported
- Free IDE available
- Powerful microcontroller (Cortex M0+)
- PCB ready for external antenna (Wardriving!)
- Easy firmware flashing via USB mass storage (OpenSDA)
- Documentation and examples, but with a few important omissions

Demo 1: NXP sniffer application

USB-KW41Z <-> host communication

- Hardware is detected as virtual COM/UART port (Windows/Linux)
- Freescale Serial Communication Interface (FSCI) developed by NXP for communication between host and device firmware.
- Host SDK for FSCI is available (with Python bindings)
- FSCI protocol is fairly well documented
- Allowed us to communicate directly with the USB-KW41Z without requiring the SDK to be installed

USB-KW41Z block diagram



OpenSDA

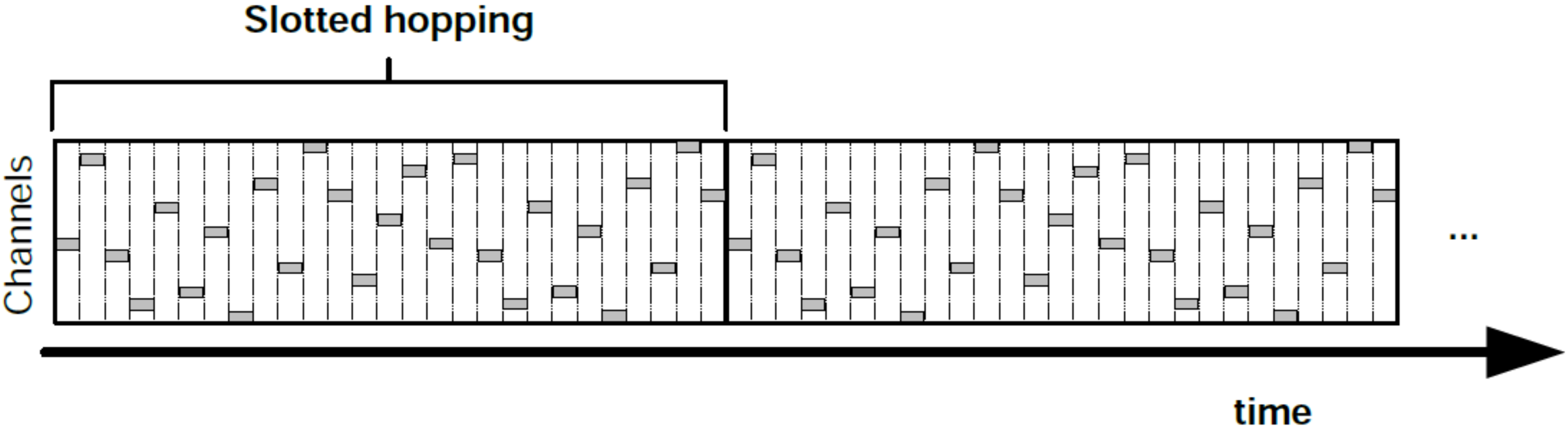
- Virtual COM port
- SWD Debugger
- MSD Bootloader

Building the toolset

- Extended the KillerBee framework with a driver for the USB-KW41Z
  - Allows us to comfortably capture 802.15.4 traffic into PCAP format
- Developed Scapy protocol support
  - Allows us to forge and inject packets
- Developed Wireshark dissectors for WirelessHART and ISA100.11a
  - Bringing WISN packet viewing to the masses
  - Live capture and dissecting of WISN traffic on a single channel at the time

Demo 2: Sniffing traffic with KillerBee and Wireshark

Theory Time Slotted Channel Hopping



Implementing Time Slotted Channel Hopping

- Both protocols require high speed channel hopping via predefined, but different patterns.
- FSCI communication too slow to tune into time slots (10ms)
  - **Solution: implement channel hopping in firmware**
- Two layers of encryption/authentication
  - **Solution: Implement in host software (Killerbee)**
- Ability to inject traffic
  - FSCI supports injection of arbitrary frames
  - **Solution: Implement frame injection in Killerbee, add protocol support to Scapy for crafting packets**

Demo 3: Sniffing with channel hopping

Unauthenticated attacks

- Signal jamming through continuous power emission
- Concurrent packet transmission
  - Join slot jamming
  - Selective jamming transmitter communication
  - Transmitting fake advertisements

Demo 4: Join slot jamming

Demo 5: Capturing the join process

Authenticated attacks

- Nonce exhaustion
  - Both protocols use a semi-predictable nonce counter to feed the AES CCM* algorithm
  - A device will reject a packet if a nonce value is lower than a previously received one
  - Spoofing a packet with a maximum nonce value, causes legitimate packets to drop
- Sending spoofed measurements to influence the process

Conclusions

- Still a large unexplored attack surfaces due to complexity of the protocols
- The released tools and research will fill this gap and enable security researchers to move forward in the field of WISN research
- Using WISN technology for process control and especially functional safety applications is probably not a good idea, and should be reconsidered

Future research

- Expand tool with more theorized attacks
- Research forced rejoin triggers
- Mapping WISN locations (wardriving)
- Implementation specific vulnerabilities (transmitters, gateways)

Questions & thank you

- https://github.com/nixu-corp

nixu

cybersecurity.

www.nixu.com

f /nixuoy

@nixutigerteam

in /company/nixu-oy