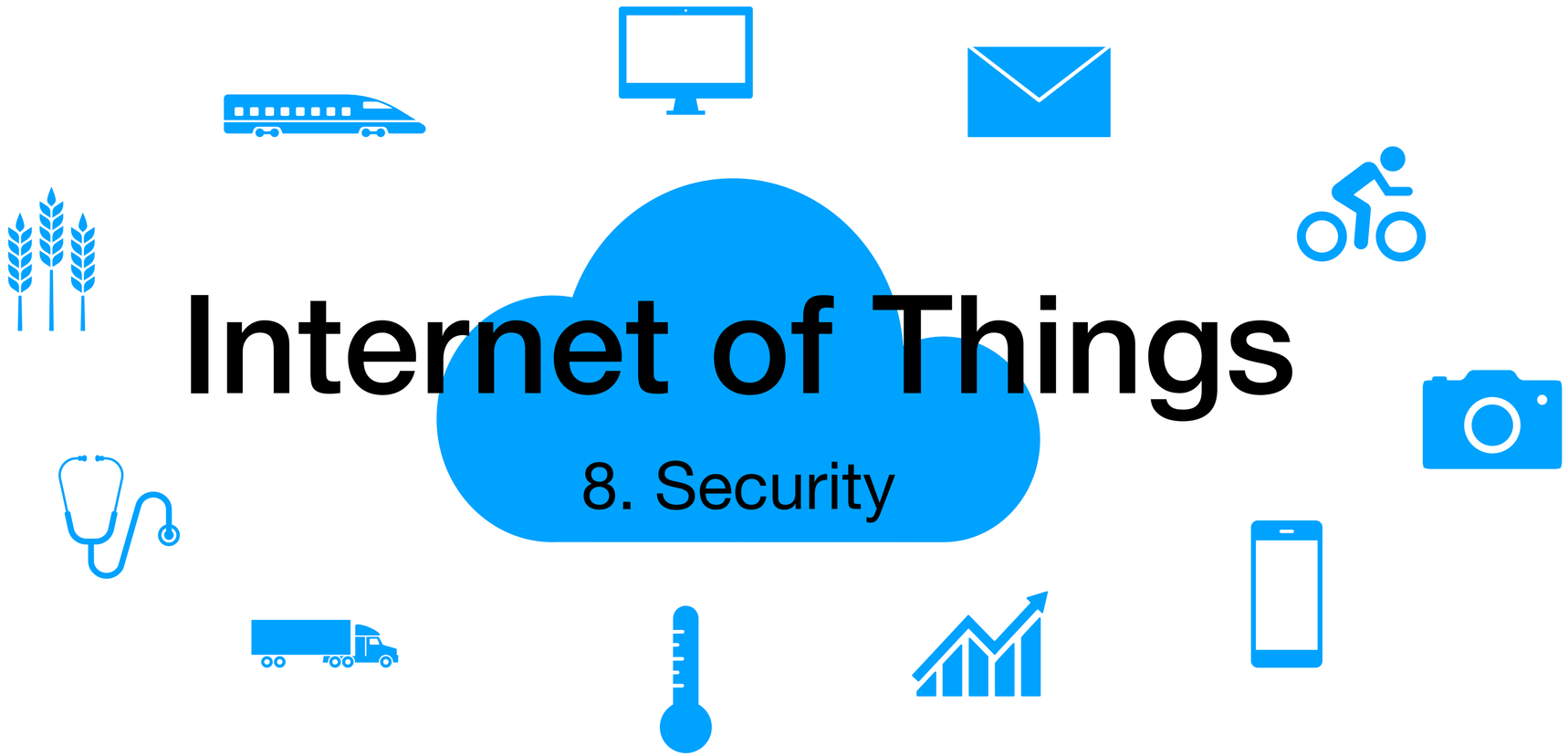


Internet of Things

8. Security

M.Sc. Computer Science 2024-2025

Viviana Arrigoni

Mirai Botnet Attack

- August 2016, the Mirai Botnet attack was launched on Dyn, a company controlling much of the internet's domain name system (DNS) infrastructure.

- The attack was designed to exploit IoT devices such as cameras and DVRs by scanning the internet for vulnerable devices with open ports, default factory settings and preinstalled usernames and passwords.

- The infected devices joined the Mirai botnet, allowing the attackers to send commands from a central command and control server.

Mirai Botnet Attack

- The corrupted IoT devices were flooding the internet
- 1.2 terabits per second of traffic, webpages like Twitter, Reddit, Netflix and Airbnb were down across North America and Europe.
- One of the biggest DDoS attacks of all times.
  - Read this-> https://www.justice.gov/usao-ak/pr/hackers-cooperation-fbi-leads-substantial-assistance-other-complex-cybercrime
    If you want to know about the guys who made this.

- In 2017, a research group found some serious security holes in some pacemakers and defibrillators made by St. Jude Medical.


Feature | EP Lab | January 09, 2017 | Dave Fornell

FDA Confirms Cybersecurity Vulnerabilities of St. Jude's Implantable Cardiac Devices, Merlin Transmitter

- No authentication, an attacker to get access to devices implanted in human bodies, making it possible extract data and overwhelm the devices with messages to drain their battery faster.
- Similar security issues were found in Owlet, a sensor that babies wear in a sock that monitors their heartbeat and relays that data to a nearby hub through Wi-Fi.


The Register®

This article is more than **1 year old**

Wi-Fi baby heart monitor may have the worst IoT security of 2016

Gaping security holes, but a fix may be coming for Owlet

- The whole interface was not authenticated. An attacker could have disconnected the Wi-Fi altogether, and broken the alerts.

Jeep Cherokees hack

- In 2016, Charlie Miller and Chris Valasek, had discovered and exploited a replicable security vulnerability in Jeep Cherokees
- The hack impacted 1.4 million vehicles and required a product recall
- YouTube video
- Paper: Miller, Charlie, and Chris Valasek. "Remote exploitation of an unaltered passenger vehicle." *Black Hat USA* 2015.S 91 (2015): 1-91.

Jeep Cherokees hack

- Scanned the Sprint network and found thousands of exposed vehicles online.
- Allowed them to connect to the Uconnect (an in-vehicle connectivity platform) head unit, where they found a remote code execution vulnerability, which let them run arbitrary commands.
- From there, they were able to pivot into the vehicle's internal network — the CAN (Controller Area Network) bus — which controls vehicle functions.
- Once there, they could send messages to Electronic Control Units (ECUs) that controlling steering, breaking, transmission, speed.

IoT environments

- Tens of billions devices connected to the Internet.
- Smart objects are becoming ever more ubiquitous and pervasive.
  - Applications span from smart cities, smart homes, healthcare, surveillance, etc.
- Moreover, many of such smart services require users to intentionally **reveal some personal** (and, some times, **private**) **information in change for advanced and more personalized services**.
- Security and privacy should be of primary importance in the design of IoT technologies and services.

- The reality is that many IoT commercial products that are provided with inadequate, incomplete, or ill-designed security mechanisms.

- Many IoT device manufacturers come from the market of low-cost sensors and actuators (e.g., home automation, lights control, video surveillance).

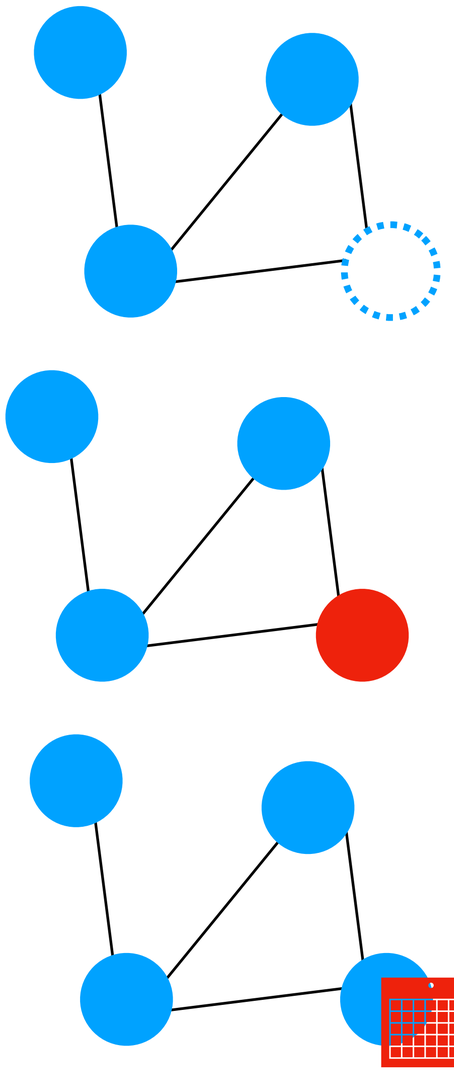
  - Devices originally designed to work in isolated systems, for which the security is not a threat.

- Users are not educated in terms of security practices .

  - Often fail to implement even the most basic procedures to protect their devices as, e.g., changing the preinstalled password

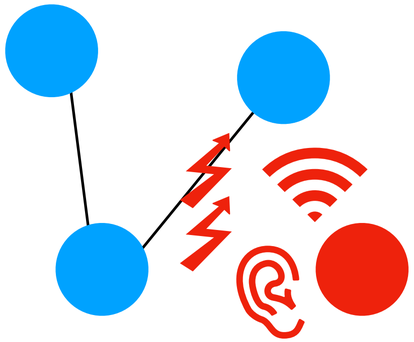
Vulnerabilities of the Thing Layer

- **Node capturing**
  - An attacker can steal or replace a node in a IoT system with a malicious one.
- **Malicious code injecting**
  - An attacker can inject some malicious code in the memory of the node. Firmware and software updates are done on the air. If the process is not encrypted an attacker can intercept the communication and inject malicious firmware.
- **False Data Injection Attack**
  - The attacker may use a malicious node to inject erroneous data onto the IoT system. This may lead to false results and may result in malfunctioning of IoT applications.

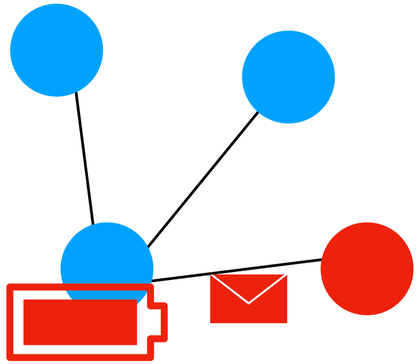


Vulnerabilities of the Thing Layer

- **Eavesdropping and Interference**
  - IoT nodes are often deployed in open environments, easily exposed to eavesdropping. The attackers may eavesdrop and capture the data during transmission and/or authentication.

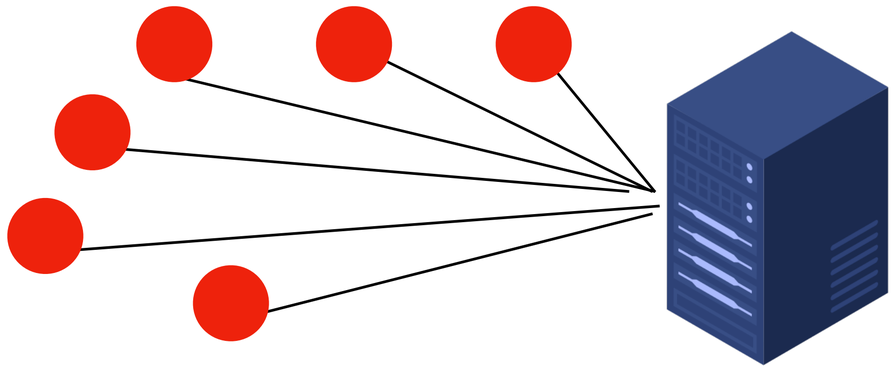


- **Sleep Deprivation Attacks**
  - The attacker tries to drain the battery of the low-powered IoT devices. This leads to a denial of service from the nodes in the IoT application.



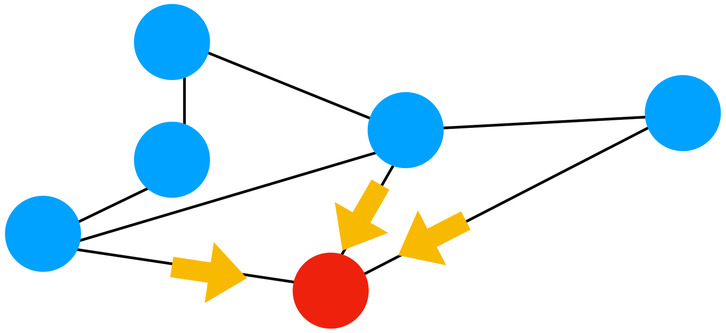
Vulnerabilities of the Network Layer

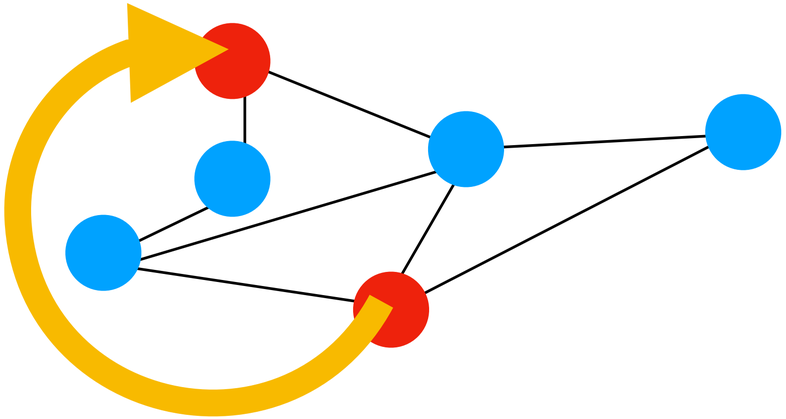
- **DoS/DDoS attacks**
  - the attacker floods the target servers with a large number of unwanted requests. This incapacitates the target server, thereby disrupting services to genuine users.
  - IoT applications are very susceptible to Distributed DoS attacks, if an attacker is able to control many IoT devices.



Vulnerabilities of the Network Layer

- **Routing attacks**
  - Malicious nodes try to redirect the routing paths during data transfer.
  - **Sinkhole attack**: the adversary advertises a better routing path and attracts nodes to route traffic through it.
  - **Warm-hole attack**: An attacker can create a warm-hole (out of band connection) between a compromised node and a device, bypassing the basic security protocols.



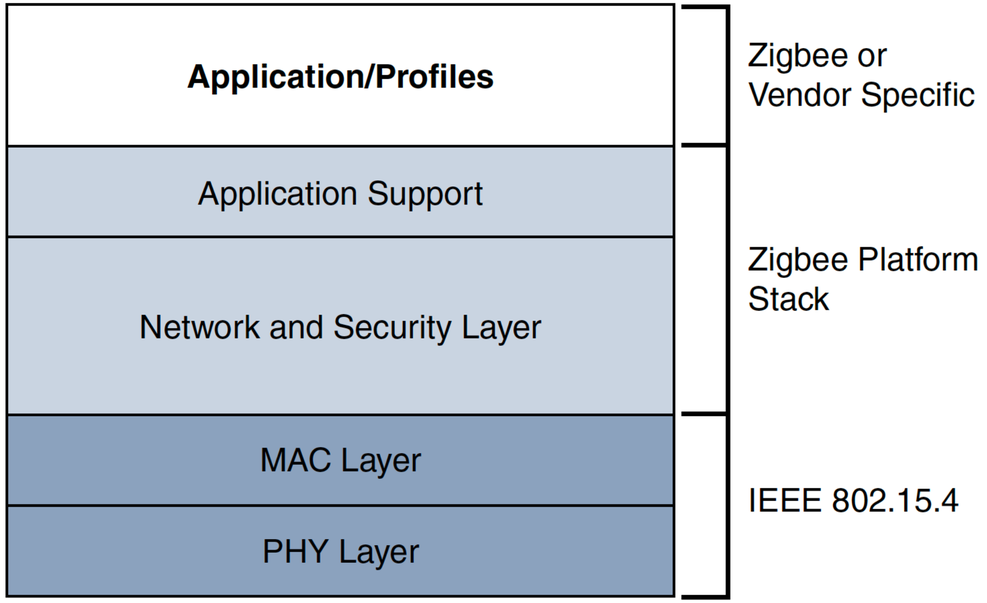


Vulnerabilities of the middleware and gateway layers

- **Man-in-the-Middle Attack**
  - The MQTT protocol uses a publish-subscribe model of communication between clients and subscribers using the MQTT broker. If the attacker can control the broker and become a man-in-the-middle, then he/she can get complete control of all communication without any knowledge of the clients.
- **End-to-end encryption**
  - To ensure the confidentiality of the data, the IoT application should not let anyone other than the unique recipient to decrypt the encrypted messages.
  - To perform protocol translation, gateways are required to decrypt and re-encrypt the messages.
  - Data becomes susceptible to breaches.

Security measures/vulnerabilities/ attacks of popular IoT Protocols 8.1 Zigbee

ZigBee recap (1)



- PHY and MAC layer taken from IEEE 802.15.4 standard.
- The NWK provides functionalities such as routing, security, and configuration of new devices, managed by the **network coordinator** that also acts as **trust center**.
- The trust center is responsible for
  1) authenticating the devices that require to join the network;
  2) deciding whether to accept or deny the join request;
  3) maintaining and distributing network keys;
  4) enabling end-to-end security between devices.

Cryptographic keys

- The cryptographic routines used in ZigBee employ two 128 bit keys:
  - **link key**, used to secure unicast communications between the application and the device.
  - **network key** is needed for broadcast communications: it is shared among the same network.
- Vendors also provide device-specific keys called **master keys**.

Key acquisition

- There are different ways for a device to acquire the required link or network key:
  - Preinstallation: The link or network key is installed in the device during the manufacturing process
  - Key Transport: The link or network key is generated elsewhere (usually by the Trust Center) and then communicated to the device.
  - Key Establishment: Uses asymmetric cryptography (public/private key pairs) to establish a shared secret that both devices can compute independently, which can then be used as a Link Key.
    - Uses a protocol based on Elliptic Curve Diffie-Hellman.

Key acquisition

- There are different ways for a device to acquire the required link or network key:
  - Preinstallation: The link or evice during the manufacturing process
  - Key Transport: The link or network key is generated elsewhere (usually by the Trust Center) and then communicated to the device.
  - Key Establishment: Uses asymmetric cryptography (public/private key pairs) to establish a shared secret that both devices can compute independently, which can then be used as a Link Key.
    - Uses a protocol based on Elliptic Curve Diffie-Hellman.

Often sent unencrypted over-the-air

ZigBee Network Key Sniffing Attack

- The attack can be as easy as follows:

Network key

Zigbee coordinator

Zigbee sensor

Run Texas Instruments SmartRF Protocol Packet Sniffer software

Texas Instruments CC2531 USB **dongle**: allows any computer including Raspberry Pis and Mini PC to implement RF transmutation and reception through ZigBee

Parse captured data

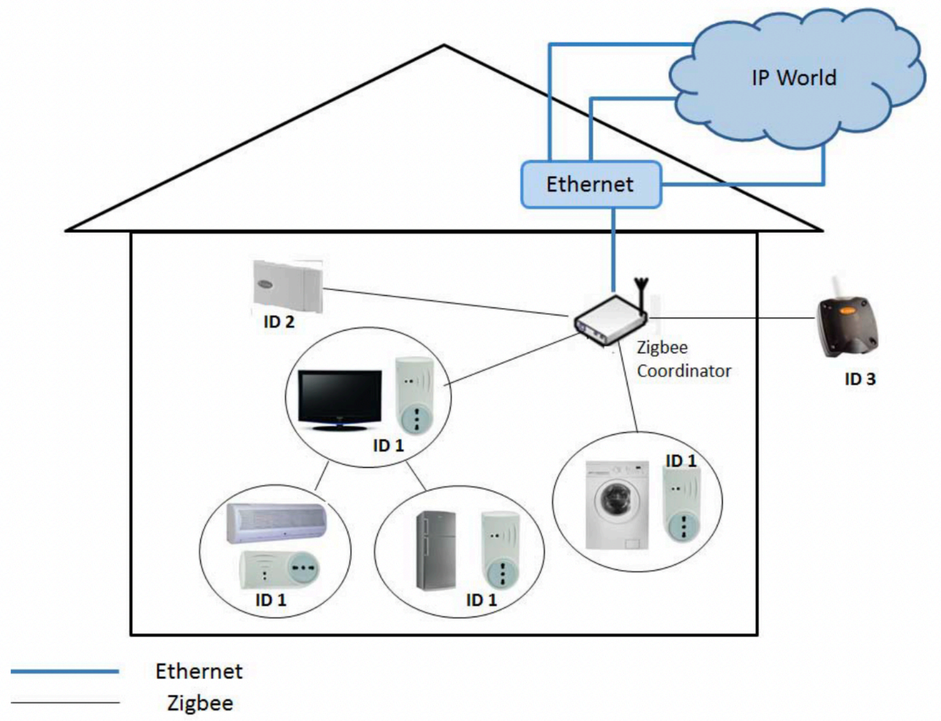
```
nikow@nikow-VirtualBox:~/killerbee-1.0$ zbdsniff generic.pcap
Processing generic.pcap
NETWORK KEY FOUND: 0d:0c:0a:08:06:04:02:00:0f:0d:0b:09:07:05:03:01
  Destination MAC Address: 00:12:4b:00:01:00:37:18
  Source MAC Address:      00:12:4b:00:01:00:17:e8
Processed 1 capture files.
```

Vidgren, Niko, et al. "Security threats in ZigBee-enabled systems: Vulnerability evaluation, practical experiments, countermeasures, and lessons learned." 2013 46th Hawaii International Conference on System Sciences.

Sinkhole attack

- Sinkhole attack: very common attack in Wireless Sensor Networks.
- A compromised node in the network starts spreading false information about his routing capabilities, pretending to have a really good route to the Base Station.
  - All other nodes in the network will start forwarding packets directed to the Base Station through the malicious node.
  - The attacker can drop or simply modify data inside packets and then forward them to the coordinator, making it difficult to detect the attack.

Sinkhole attack - example (1)



ID 1 - ZigBee plugs. Can turn on and off electrical appliances connected to it.
ID 2 - ZigBee Comfort Sensor measuring temperature
ID 3 - ZigBee outdoor sensor. Outdoor sensor measuring temperature humidity and light
ZigBee coordinator is the Access Point

1. Construct a ZigBee network with a single ZigBee coordinator.
2. Multi-purpose device joins the network legally
   - Android Geek Land tablet equipped with CC2530 Zigbee SoC supporting IEEE 802.15.4 and Zigbee applications.

   - **Feature: possibility to set a particular transmit power.**
     Why is this important? …

Coppolino, Luigi, et al. "My smart home is under attack." 2015 IEEE 18th International Conference on Computational Science and Engineering. IEEE, 2015.

- … because ZigBee supports two possible topologies: hierarchical trees and mesh networks.

- In hierarchical trees, the routing is fixed.

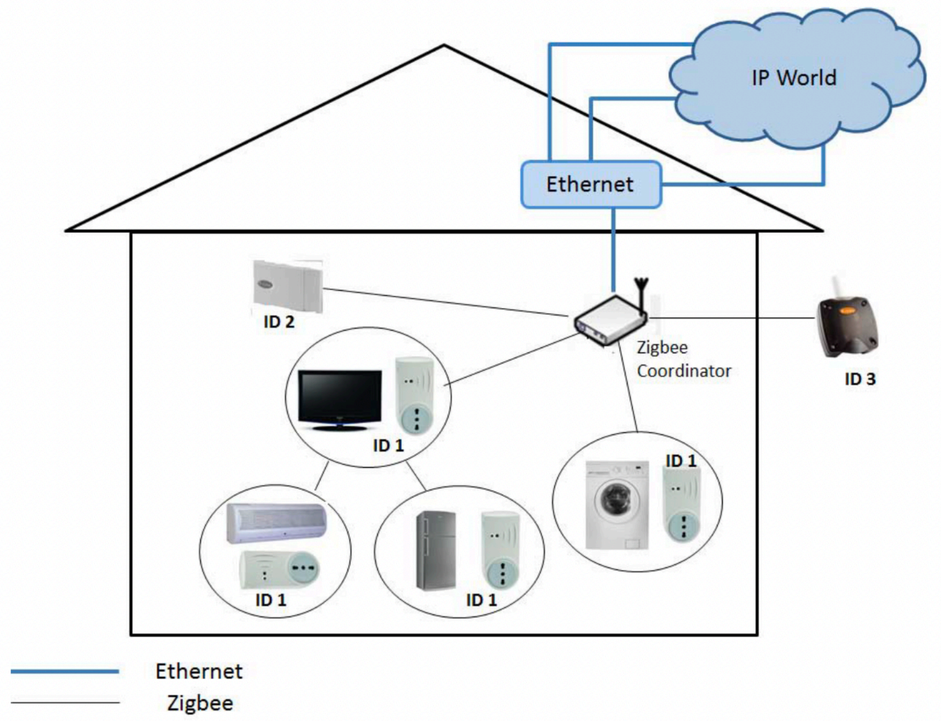
- But in mesh networks, ZigBees uses a weighted AODV routing protocol where link weights are defined as follows:

  $$C(l) = \min\left(7, \left\lceil \frac{1}{p_l^4} \right\rceil\right),$$

  where p_l is the probability of successful packet delivery over link l.
  This is not specified by the ZigBee Alliance, but by individual developers and vendors.

- A reasonable way to define p_l is to make it grow with the transmission power of the devices connected to l

Sinkhole attack - example (2)



ID 1 - ZigBee plugs. Can turn on and off electrical appliances connected to it.
ID 2 - ZigBee Comfort Sensor measuring temperature
ID 3 - ZigBee outdoor sensor. Outdoor sensor measuring temperature humidity and light
ZigBee coordinator is the Base Station

3. Hack the Android tablet by injecting a malware.
   Several attacks that exploit Android's bugs exist, for instance, it has been seen that a Trojan attack can be used to remotely execute commands in the console.

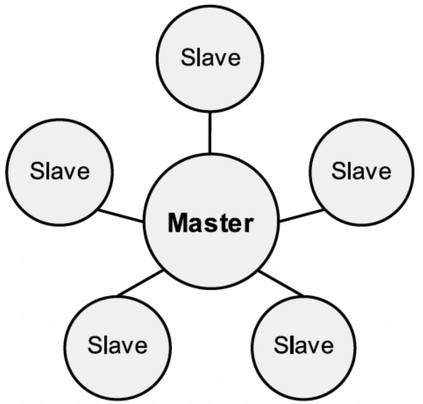
4. Modify ZigBee SoC transmission power parameters in order to perform a sinkhole attack.

5. Packets are indeed routed through the malicious device, which can modify intercepted packets.

Security measures/vulnerabilities/ attacks of popular IoT Protocols 8.2 BLE

BLE - recap (1)

- Central and peripheral nodes. Peripheral broadcast advertisements, central listens to the advertisements of the devices it wants to interact with, and they connect.



- BLE operated in the unlicensed 2.4 GHz ISM band and uses 40 channels with a 2 MHz spacing.

- The BLE MAC layer is split into two parts:
  - advertising, using 3 channels to broadcast device information and establish connections
  - data transfer, using 37 of the available channels are used during the transmission of data

BLE - recap (2)

- In the data communication phase, data is normally sent in bursts to save energy.
  -> In this way, peripherals can remain in sleep mode for long periods, waking up periodically to listen to the channel for possible messages from the master.

- The central decides the rendezvous instants with the peripherals, according to a time division multiple access (TDMA) scheme with frequency hopping (different from CDMA, cos each device uses the channel with FH, but one at the time following TDMA).

BLE addresses

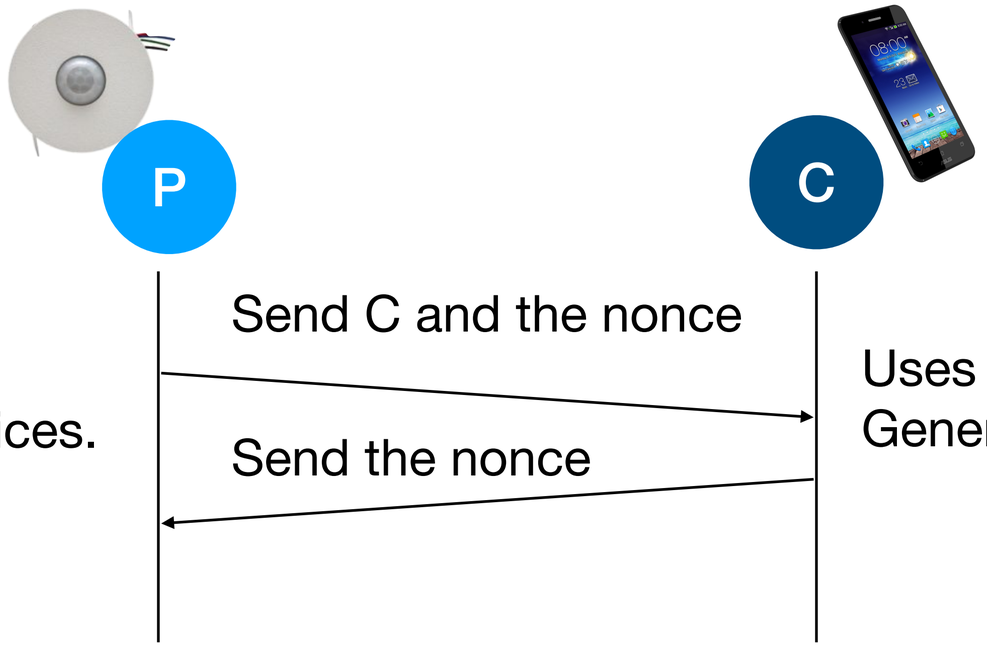
- BLE MAC address is hidden, whereas a **random address** which changes frequently is advertised.
- BLE can choose one of three following types of random addresses:
- **Random Static Address** - Generate a random address per power cycle. The address cannot be regenerated at any other time.
- **Resolvable Private Address (RPA)** - Generate a random address within a given time interval. Generated using a Identity Resolving Key (IRK) and a random number.
  - Trusted devices that know the random key can resolve it, and recover the real identity of the device.
- **Non-resolvable Private Address** - Randomly generated address with a given time interval.
  - No device can resolve it and recover the true identity. Used to achieve full anonymity.

BLE encryption and authentication

BLE encryption and authentication processes are based on AES with 128 bit keys. The symmetric key for link is generated during the **pairing procedure**, which is executed as follows.

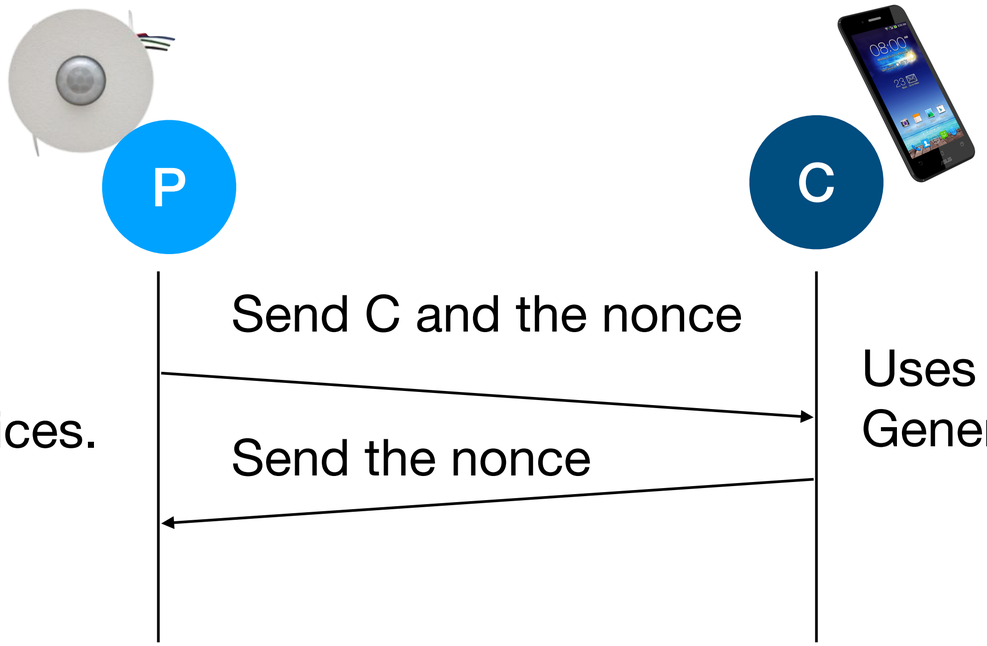
1. **Feature exchange.** Clients and servers exchange IO capabilities, authentication requirements, Long Term Keys (LTK) entropy proposal (KeySize, from 7 to 16 bytes - **negotiation**) etc. None of this is encrypted.
2. **Key establishment**
   - The devices generate or exchange a temporary key (TK) using one of the available **pairing methods** *(next slide)*.
   - After that, a short term key (STK) is generated from the TK.
3. **Key distribution phase**
   Using the STK-encrypted link, both devices send selected long-term keys, used to derive session keys for future encrypted connections
4. **Bonding phase**
   - Devices exchange and store common link keys (LTK) that can be reused when a link between the two devices is re-established at a later time.

Pairing methods in authentication - just works



- The Temporary Key is 0.
- Does not provide any security: an attacker can generate its own nonce and use the public keys to create a valid confirmation value C.

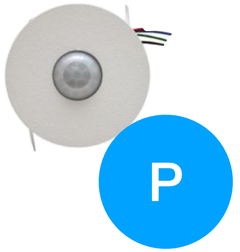
Pairing methods in authentication -numeric comparison



- Just like "just works", but the devices also display a 6-digit key, and the user must manually confirm that the key is the same for both devices.
  - Devices must be equipped with a display.

Pairing methods in authentication - passkey

- One of the two devices generates a k bit long secret **passkey**.
- The user inserts it in the other device.
- Both devices:
  - Generate a nonce and compute a commitment value, which is function the nonce, the passkey and the public keys.
  - Exchange commitments and nonces.
- Each device recalculates the commitments as before, but changing the order of the two public keys and using the nonce of the other device.
- If the passkey is the same, the new commitment value must match the one sent by the other device.



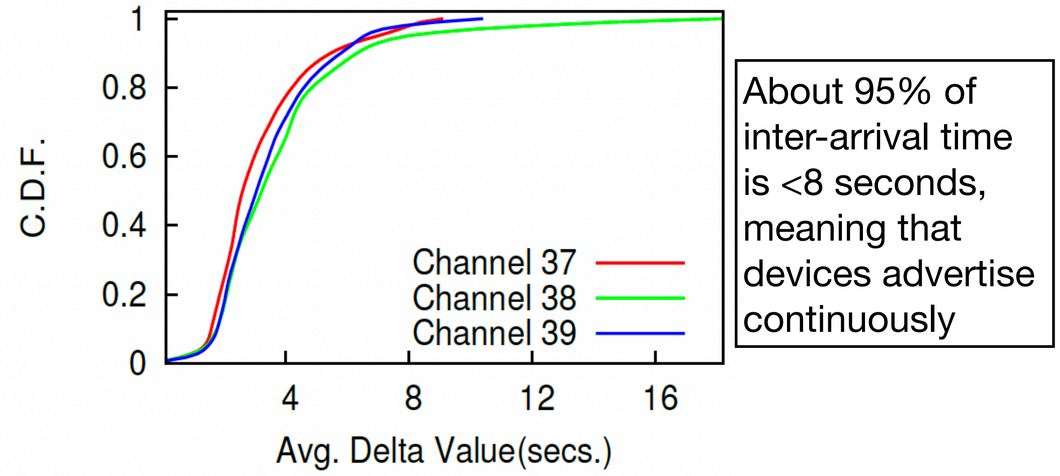
$C_P=f(n_P$,passK,pubKP, pubKC) $C_C=f(n_C$,passK,pubKC, pubKP)

$C'_P=f(n_C$,passK,pubKC, pubKP) $C'_C=f(n_P$,passK,pubKP, pubKC)

$C'_P=C_C$ $C'_C=C_P$

Network traffic sniffing - fitness tracking case study (1)

- Exploited vulnerability: frequent advertisement BLE packets and poor address management
- Listening to BLE traffic in a gym.
- BLE traffic traces collected for 8 consecutive days, with each trace being two hours in duration. 6 fitness trackers (servers), 1 IPhone, 1 Nexus (clients).
- 7.5 million packets, 3.5GB.



About 95% of inter-arrival time is <8 seconds, meaning that devices advertise continuously

Cumulative distribution function of the mean inter arrival time of between two consecutive advertisement packets in the same channel for each device

- Most of the advertisement packets come from the servers (fit trackers), while clients (iOS and Android mobiles) do not advertise continuously and change their address often.
- Why do fitness trackers advertise so frequently?

Das, Aveek K., et al. "Uncovering privacy leakage in BLE network traffic of wearable fitness trackers." Proceedings of the 17th international workshop on mobile computing systems and applications. 2016

Network traffic sniffing - fitness tracking case study (2)

- … Because the client device frequently disconnects the fitness trackers in order to reduce its own energy consumption.
- The fitness trackers are only connected to the smartphone when the corresponding smartphone application on the smartphone is running
  - In this phase, the tracker actively communicates and synchronizes the activity data (e.g. steps, calories etc.).
  - When the app is not running, the connection is terminated, leaving the fitness tracker in advertising state.
- This behaviour was observed in all six fitness trackers in the study

Frequent advertising makes the devices vulnerable to tracking.

An attacker can sniff the BLE traffic and track the users as they move around in public places

How to avoid this?

Network traffic sniffing - fitness tracking case study (3)

- … We saw that BLE devices can change their addresses to prevent these situations.
- Reality: none of the trackers did that.
  - Most used static addresses and a few resolvable private addresses. Both kinds did not even change after complete battery discharge.
  - This makes it easy to be used to activity detection.
  - The tracking vulnerability stays.

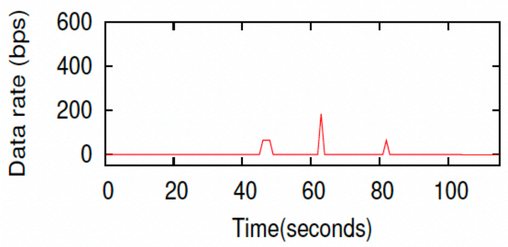
An attacker can sniff the BLE traffic and track the users as they move around in public places

Network traffic sniffing - fitness tracking case study (4)

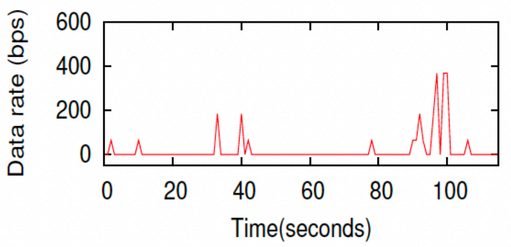
- **Activity detection.**

- The tracking vulnerability can be exploited to perform additional inference in a person's activity.

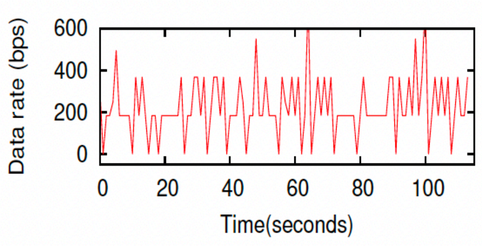
- Observation: the amount of BLE data traffic between the Fitbit and the smartphone is proportional to the (motion) intensity of user's activity.



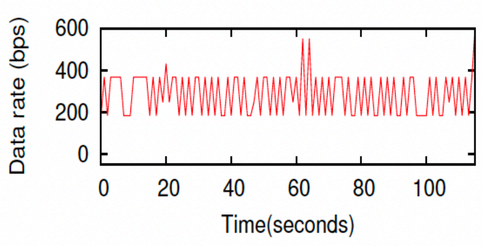
User at rest



User sitting/ working at desk



User walking



User running

Network traffic sniffing - fitness tracking case study (5)

- **People identification.**
- Fit trackers use accelerometers to get statistics like number of steps walked, total calories burnt, total distance covered, flights of stairs climbed.
- **BLE data exhibits different patterns when different users are walking**

Accelerometer	Mean and Max Acceleration Zero-Crossings, Absolute Area Sum of Absolute Acceleration
BLE Data	Start Pkts., Empty Pkts., Payload Size Payload Datarate, Time b/w Start Pkts. Empty Pkts. b/w Start Pkts

Extracted features from BLE traffic and accelerometer data

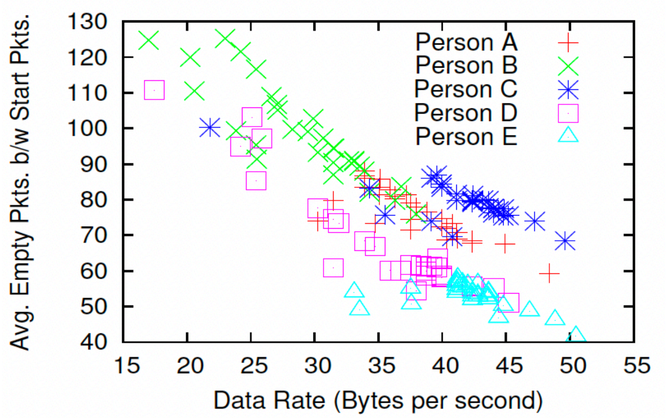
BLE Data Feature	Correlation
Empty Pkts.	0.705
Payload Datarate	0.699
Start Pkts.	0.684
Payload Size	0.676
Time b/w Start Pkts.	0.647
Pkts b/w Start Pkts.	0.634

Positive and kinda strong correlation between accelerometer data and BLE traffic

linear regression models.
Input=accelerometer features
output=BLE network features.

Network traffic sniffing - fitness tracking case study (5)

- **People identification.**
- Fit trackers use accelerometers to get statistics like number of steps walked, total calories burnt, total distance covered, flights of stairs climbed.
- **BLE data exhibits different patterns when different users are walking**

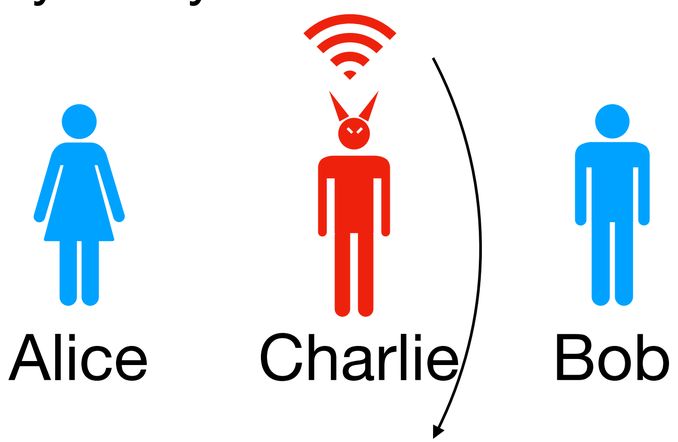


Different people walking exhibit different BLE features

Threat: by using a regression model and observing BLE traffic, an attacker can **infer accelerometer features**, which can be used to identify people's gait, which can be successfully used for people's identification.

KNOB attack

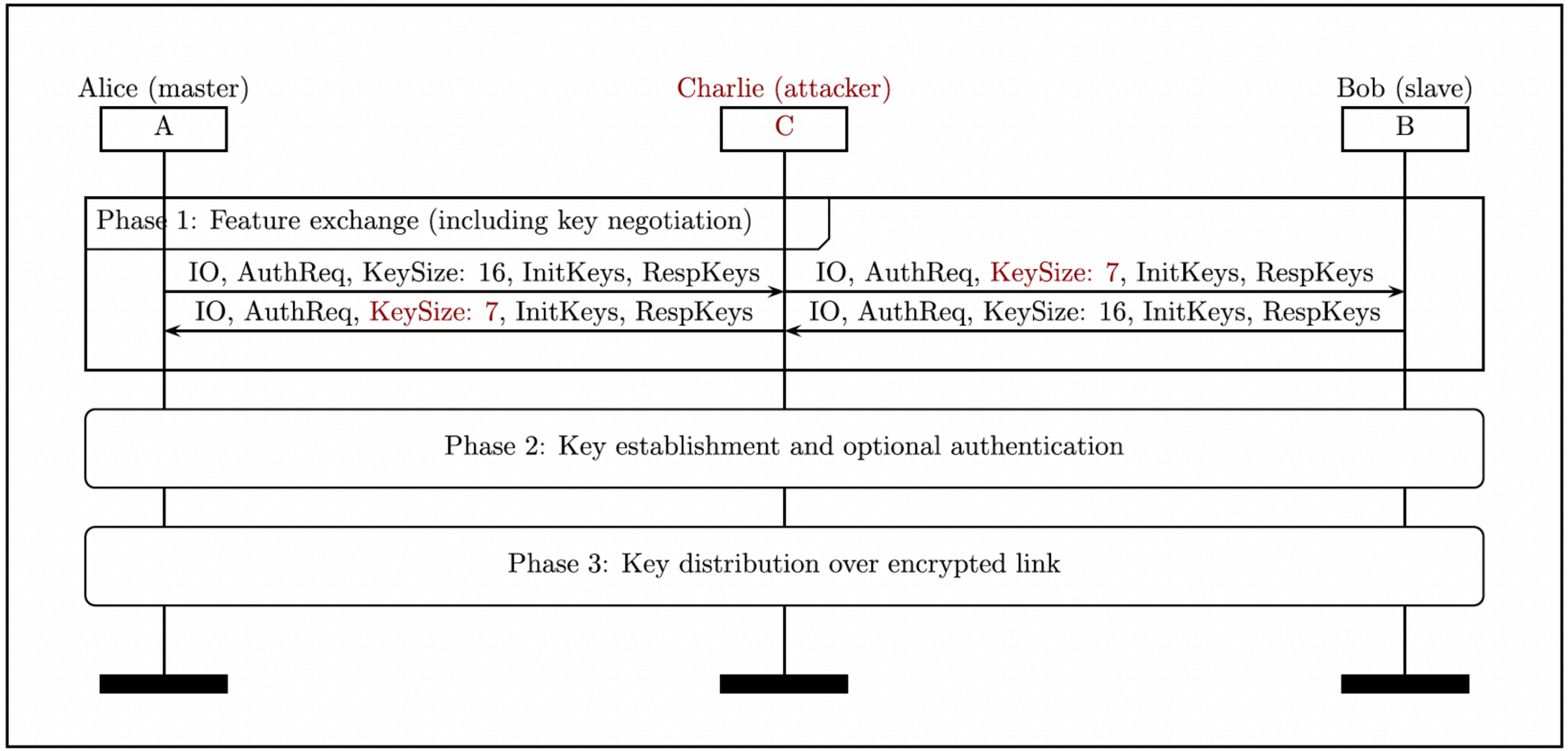
- Exploited Vulnerability: unencrypted feature exchanges.
- An attacker can downgrade the entropy of any BLE LTK and session key to 7 bytes (Key Negotiation of BLE - KNOB) through a Man-in-the-Middle (MitM) attack
- 7 byte keys can be brute-forced.



The attacker (Charlie) secretly intercepts and relays messages between two parties (Alice and Bob) who believe they are communicating directly with each other

In wireless communication, the attacker sends out disruptive and jamming signals so that Alice and Bob cannot hear each other directly.

Antonioli, Daniele, Nils Ole Tippenhauer, and Kasper Rasmussen. "Key negotiation downgrade attacks on bluetooth and bluetooth low energy." ACM Transactions on Privacy and Security (TOPS) 23.3 (2020): 1-28.



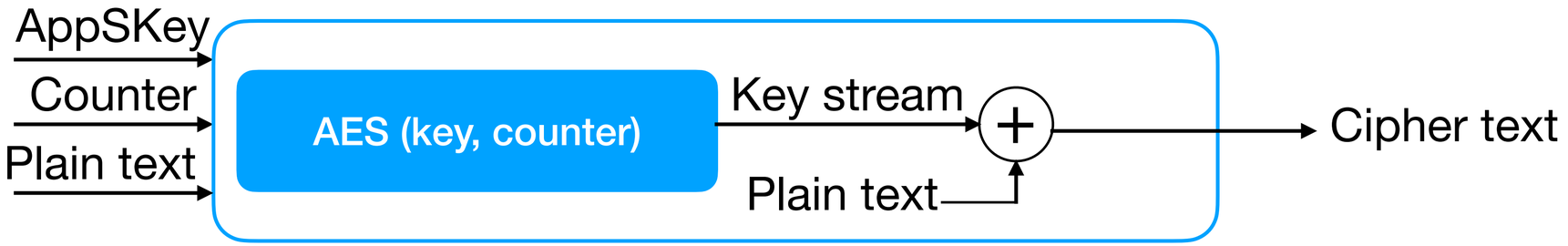
Once Charlie knows SK, he can decrypt all the ciphertext exchange by Alice and Bob and introduce valid ciphertext in the session.

Security measures/vulnerabilities/ attacks of popular IoT Protocols 8.2 LoRaWAN

Eavesdropping LoRa (1)

- Eavesdropping attacks occur when a hacker intercepts, deletes, or modifies data that is transmitted between two devices.

- Recall that LoRa uses AES-CTR as encryption scheme.

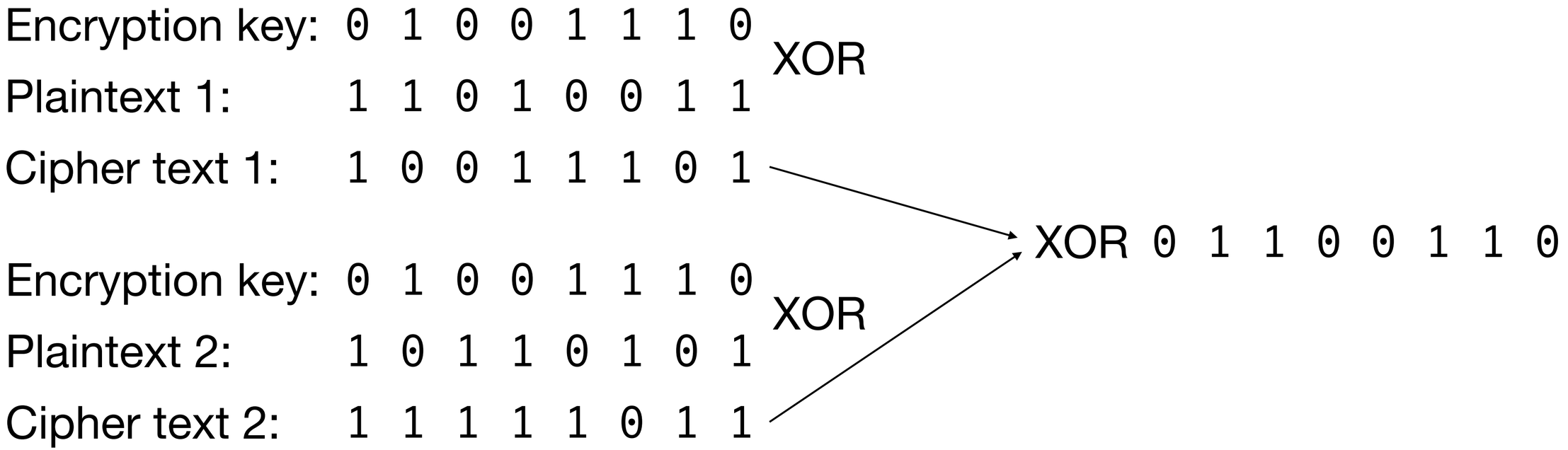


- AppSKey is the key used for end-to-end encryption and the counter is the frame counter in LoRa's MAC frame, not really a nonce.

Yang, Xueying, et al. "Security vulnerabilities in LoRaWAN." 2018 IEEE/ACM Third International Conference on Internet-of-Things Design and Implementation (IoTDI). IEEE, 2018.

Eavesdropping LoRa (2)

- What is the problem with using the same key stream multiple times?



Eavesdropping LoRa (3)

- What is the problem with using the same key stream multiple times?
- The XOR of the two cipher texts is equal to the XOR of the plain texts!

Plaintext 1:	1 1 0 1 0 0 1 1	XOR
Plaintext 2:	1 0 1 1 0 1 0 1	
	0 1 1 0 0 1 1 0	

- Since LoRa uses AppSKey as a key, the counter must be always different in order to obtain a different stream key.
- This is not the case, as LoRa uses the frame counter as a counter, which is reset to 0 after a transmission session. This means that the same key stream is used multiple times.

- Why is it dangerous?

- The XOR of the plain text still does not reveal the original messages. Nevertheless, an attacker may know or guess part of the plain text.

  - For instance, if LoRa is used by a temperature sensor, the attacker might guess that part of the original massage is “`temperature: `“

  - Crib-dragging: guess possible words and XOR those against the XORedd cipher text to possibly reveal parts of the original message.

  - A simple tutorial to do that: https://samwho.dev/blog/toying-with-cryptography-crib-dragging/

Bibliography

- Coppolino, Luigi, et al. "My smart home is under attack." 2015 IEEE 18th International Conference on Computational Science and Engineering. IEEE, 2015.
- Meneghello, Francesca, et al. "IoT: Internet of threats? A survey of practical security vulnerabilities in real IoT devices." IEEE Internet of Things Journal 6.5 (2019): 8182-8201.
- Das, Aveek K., et al. "Uncovering privacy leakage in BLE network traffic of wearable fitness trackers." Proceedings of the 17th international workshop on mobile computing systems and applications. 2016.
- Yang, Xueying, et al. "Security vulnerabilities in LoRaWAN." 2018 IEEE/ACM Third International Conference on Internet-of-Things Design and Implementation (IoTDI). IEEE, 2018.
- Antonioli, Daniele, Nils Ole Tippenhauer, and Kasper Rasmussen. "Key negotiation downgrade attacks on bluetooth and bluetooth low energy." ACM Transactions on Privacy and Security (TOPS) 23.3 (2020): 1-28.
- Vidgren, Niko, et al. "Security threats in ZigBee-enabled systems: Vulnerability evaluation, practical experiments, countermeasures, and lessons learned." 2013 46th Hawaii International Conference on System Sciences. IEEE, 2013.