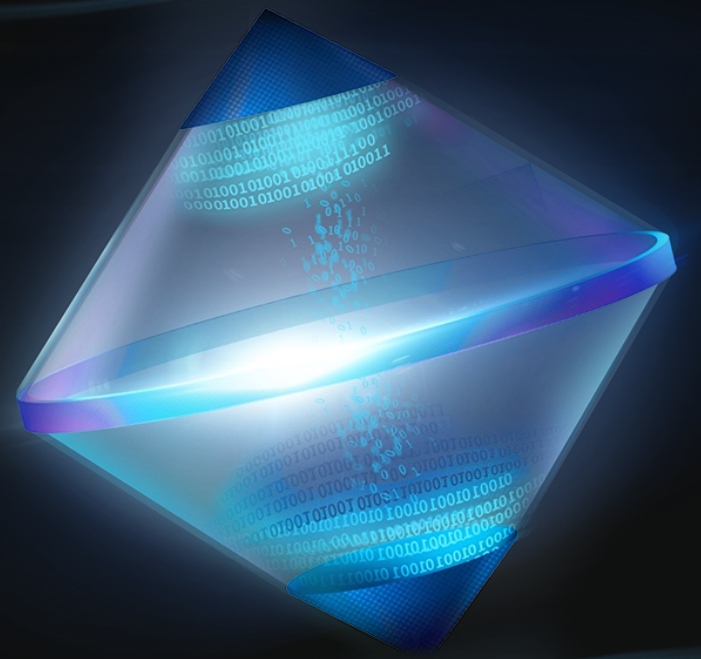
KCon

归源·智变

Returning to the source · Intelligent transformation

低功耗蓝牙黑客：从数字逃逸到现实



演讲人：肖临风 杨大林

简介

小米智能终端安全实验室

小米旗下的安全团队，实验室致力于研究行业安全技术和实践、研发自动化平台、并对智能终端产品进行安全评估和防护，提升小米产品安全性。团队成员曾多次参加 Geekpwn 天府杯 补天杯等赛事并获大奖。

KCon

归源 · 智变

Returning to the source · Intelligent transformation

目录 / CONTENTS

CHAPTER 1

侠 · 盗 · 猎 · 车

侠盗猎自行车

时间倒流

某些厂商在BLE应用层中实现的私有协议存在漏洞。攻击者可以通过刷新时间，使旧命令有效，轻松打开智能U型锁。

侠盗猎自行车

BLE分析基础

数据嗅探

使用nrf52840 dongle嗅探BLE通信，结合Wireshark进行报文的分析

GATT

GATT Server下包括多个不同的Service服务，同一个服务下可以包含多个Characteristic

通信操作

Read读、Write写

Notify通知、Indicate指示

侠盗猎自行车

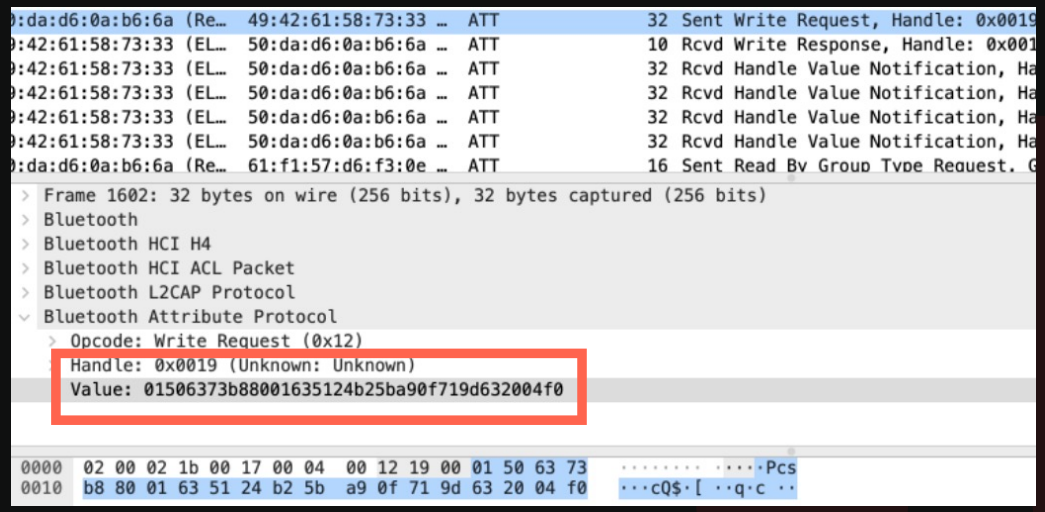
抓包与逆向分析

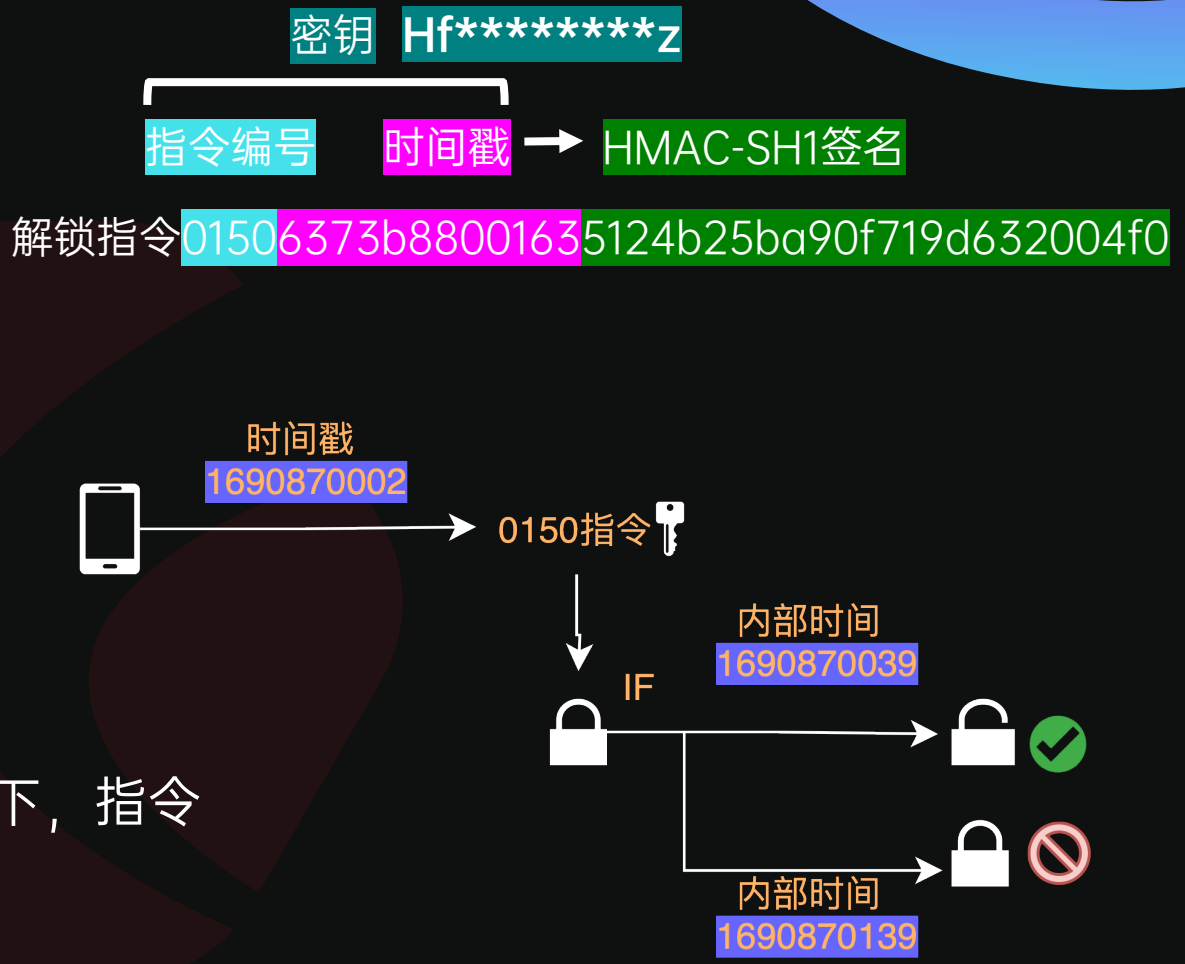
01 使用固定密钥 但锁身没有信息，无法直接获取

02 抓一次包即可爆破密钥　8位随机字符爆破时间感人

03 简单重放攻击 根据时间刷新指令，滚动码？

侠盗猎自行车





当手机时间戳和U型锁时间戳同步的情况下，指令能够正常执行

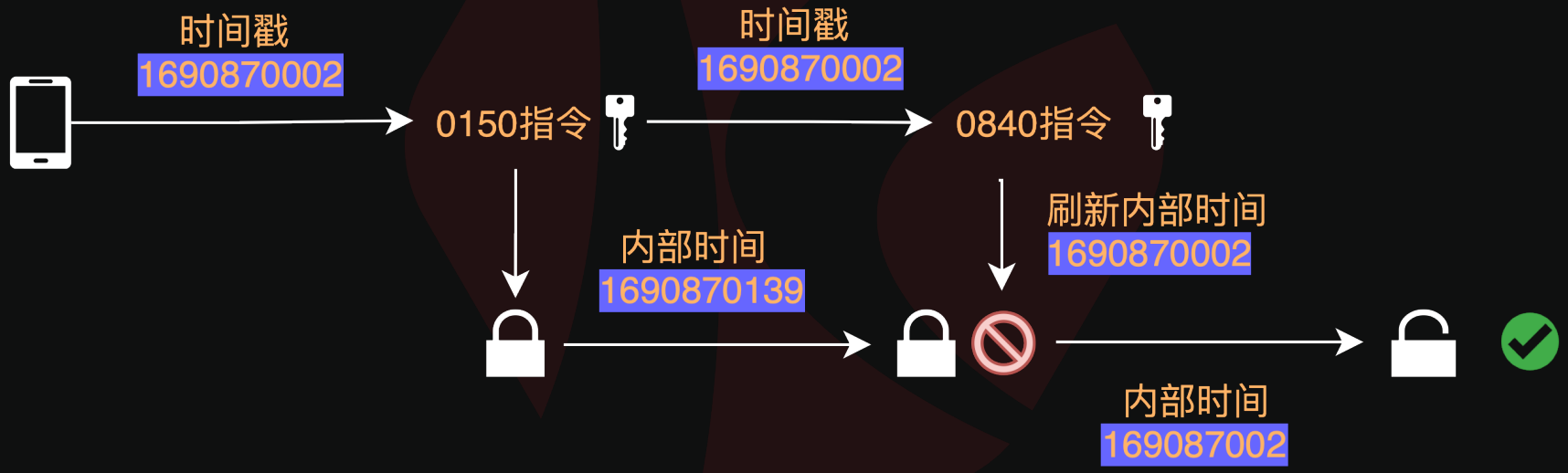
侠盗猎自行车

时间戳回滚

预防时间戳不同步导致无法解锁的问题

指令编号 时间戳 HMAC-SH1签名

解锁指令01506373b88001635124b25ba90f719d632004f0



侠盗猎自行车

重放0840指令

指令编号 时间戳 HMAC-SH1签名

解锁指令01506373b88001635124b25ba90f719d632004f0

1. 劫持0150指令，让APP认为时间戳未同步
2. APP发送0840指令刷新时间戳，捕获0840指令
3. 重放0150指令和0840指令开锁

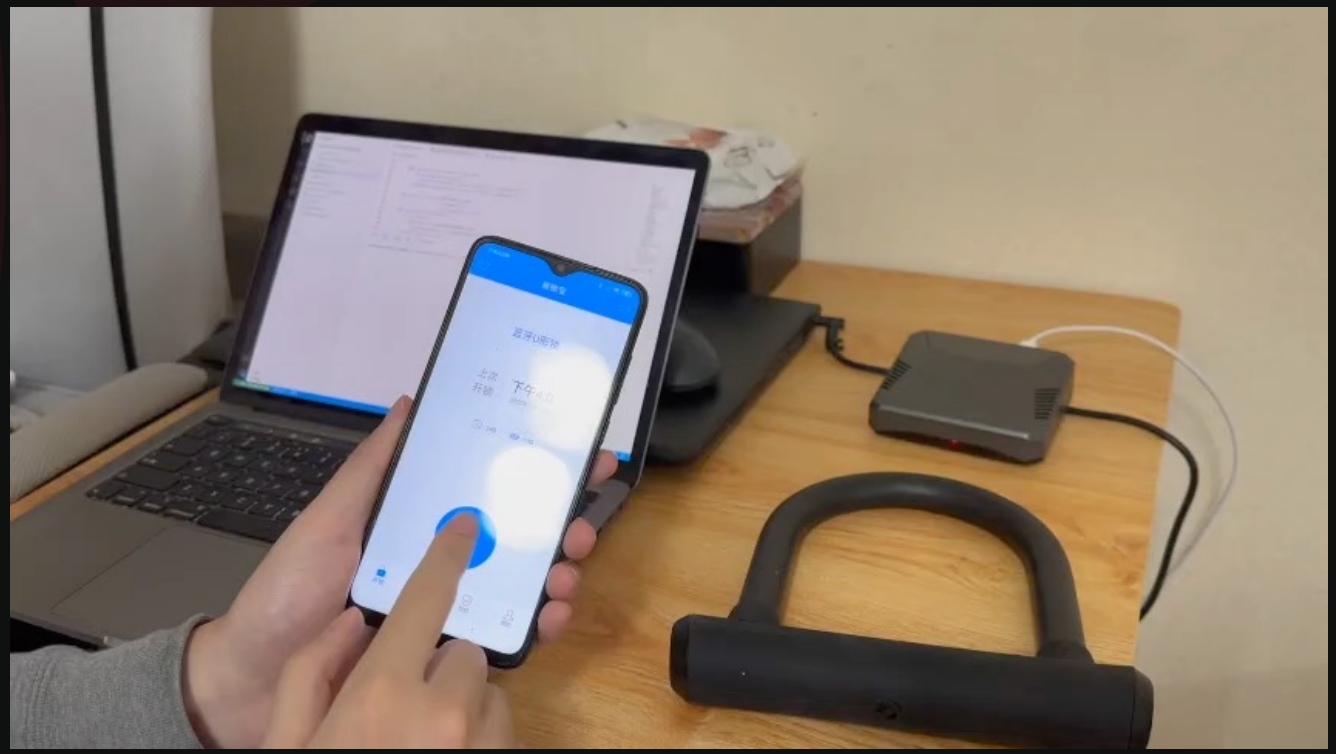
侠盗猎自行车

重放0840指令

1. 劫持0150指令，让APP认为时间戳未同步
2. APP发送0840指令刷新时间戳，捕获0840指令
3. 重放0150指令和0840指令开锁

指令编号 时间戳 HMAC-SH1签名

解锁指令01506373b88001635124b25ba90f719d632004f0



CHAPTER 2

隔 · 空 · 敛 · 财

隔空敛财

场景

扫码收银

从便利店购物，店员使用扫码枪扫描付款码进行收款。

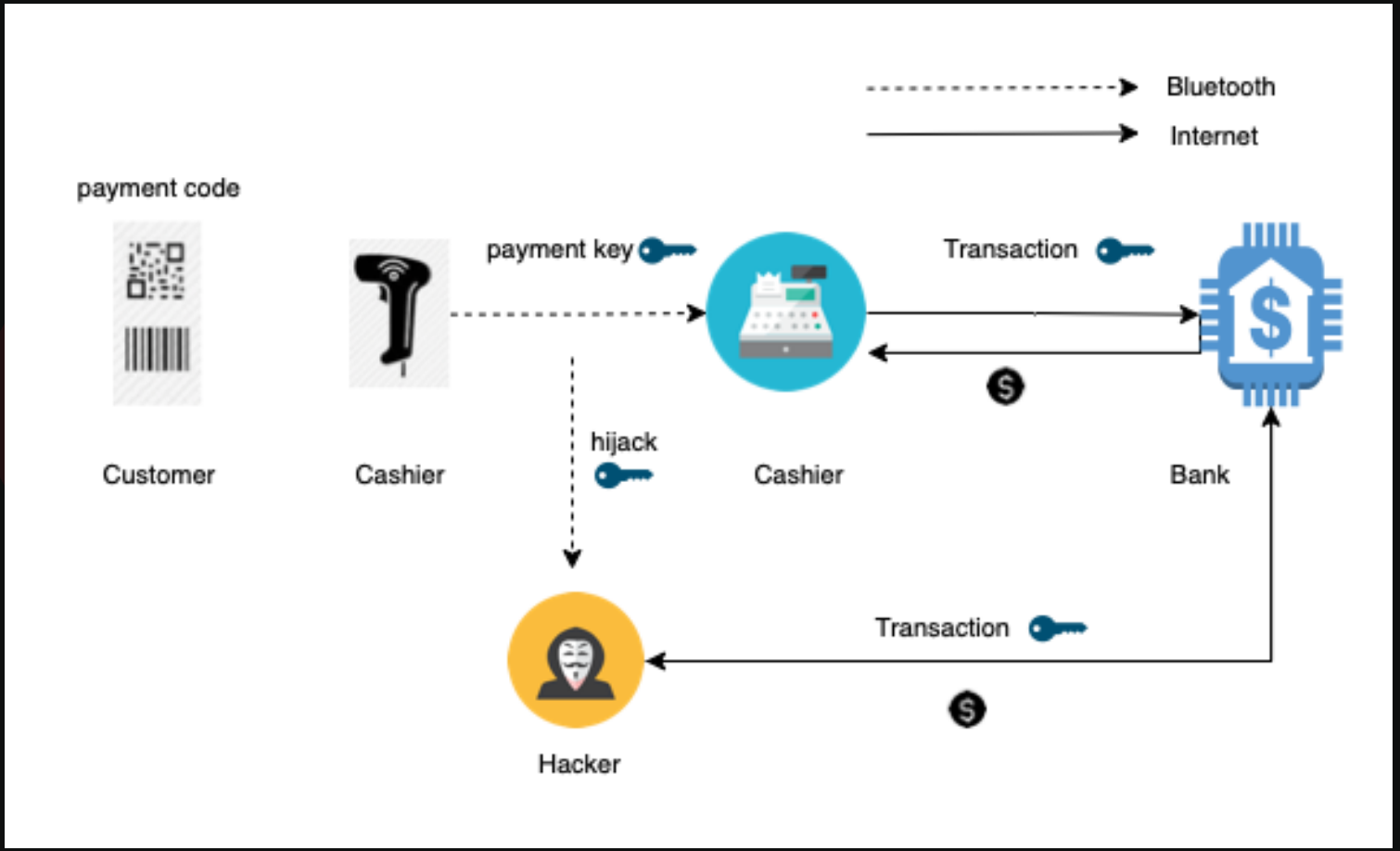
如何"远程" 0-click攻击？想到了某些商家会使用蓝牙扫码枪

隔空敛财

攻击面分析

窃取用户支付凭据?

直接攻击BLE协议



常见的无线电通信

无跳频或者跳频逻辑简单

WIFI和射频遥控通常工作在固定频段

NRF24L01跳频序列可预测[1]

缺少链路层的设计

没有建立"连接"和"交互"

信道不安全，可以抢占信道发送指令，谁吼道声音大听谁

射频遥控

[1] 315晚会报道的无人机是怎么被劫持的？

低功耗蓝牙通信

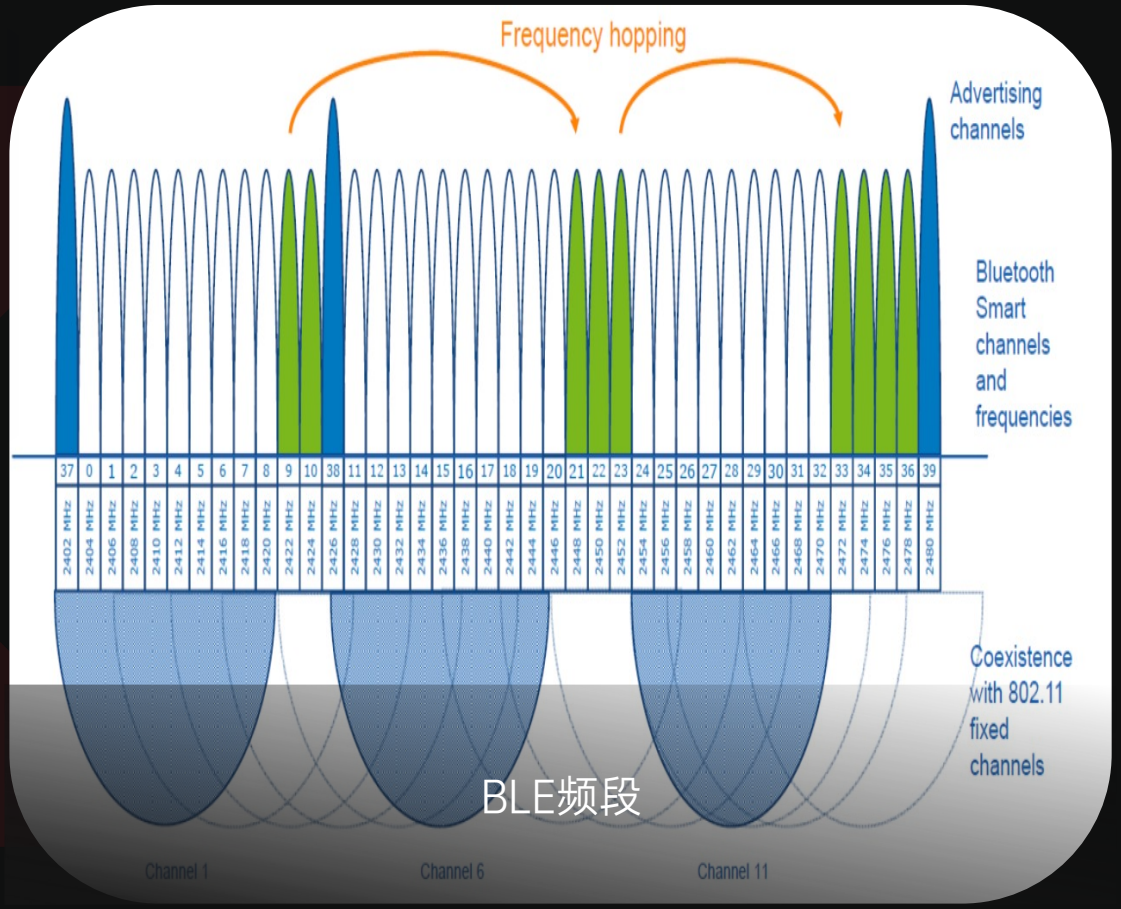
BLE Controller实现了“连接”

链路层实现了安全的信道通信

实现了“连接事件”保证BLE双方能够“交互”

跳频的复杂度高

在没有捕获到连接包的情况下很难跟上跳频。例如使用nrf52 dongle进行抓包，只有抓到握手包才能正常跟踪跳频。很难凭空抓取已经建立连接的链路！

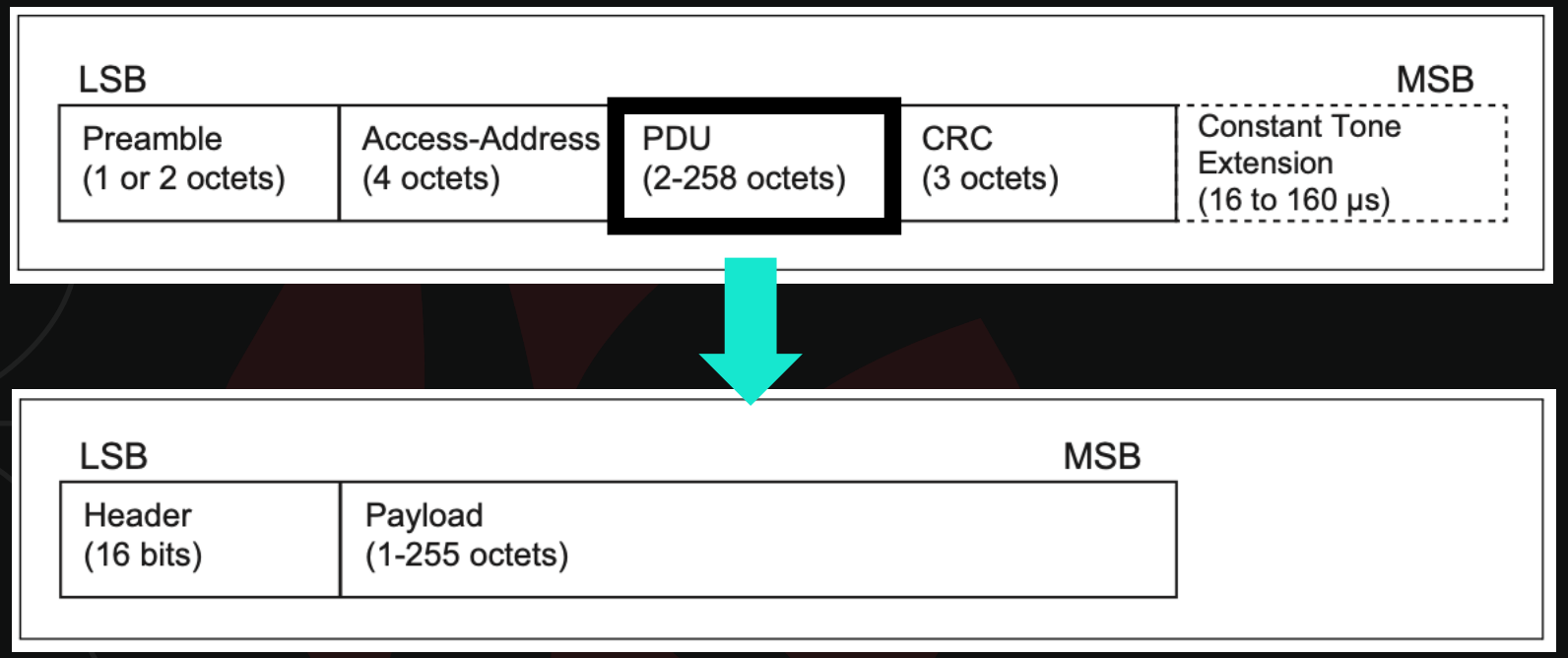


BLE频段

低功耗蓝牙通信

链路层报文

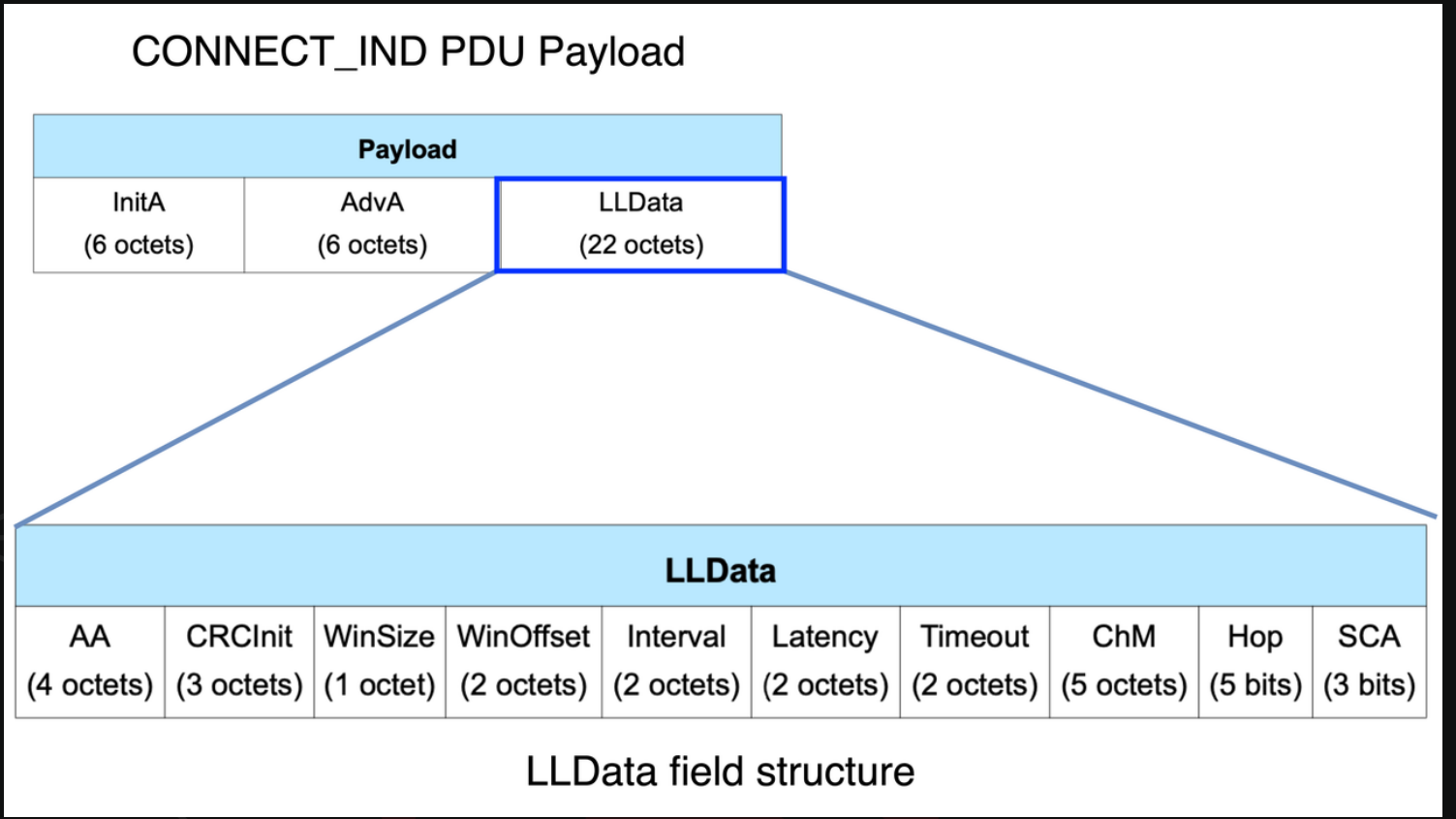
Link Layer packet format for the LE Uncoded PHYs



Header定义：
ChSel:信道选择算法，即跳频算法#1或者#2
TxAdd:指示 InitA 字段中发起者的设备地址是公共的（TxAdd = 0）还是随机的（TxAdd = 1）
RxAdd:指示 AdvA 字段中广告者的设备地址是公共的（RxAdd = 0）还是随机的（RxAdd = 1）。

低功耗蓝牙通信

CONNECT_IND



InitA：连接发起者的地址
AdvA：广播发送者的地址(被连接设备)

低功耗蓝牙通信

LLData									
AA (4 octets)	CRCInit (3 octets)	WinSize (1 octet)	WinOffset (2 octets)	Interval (2 octets)	Latency (2 octets)	Timeout (2 octets)	ChM (5 octets)	Hop (5 bits)	SCA (3 bits)

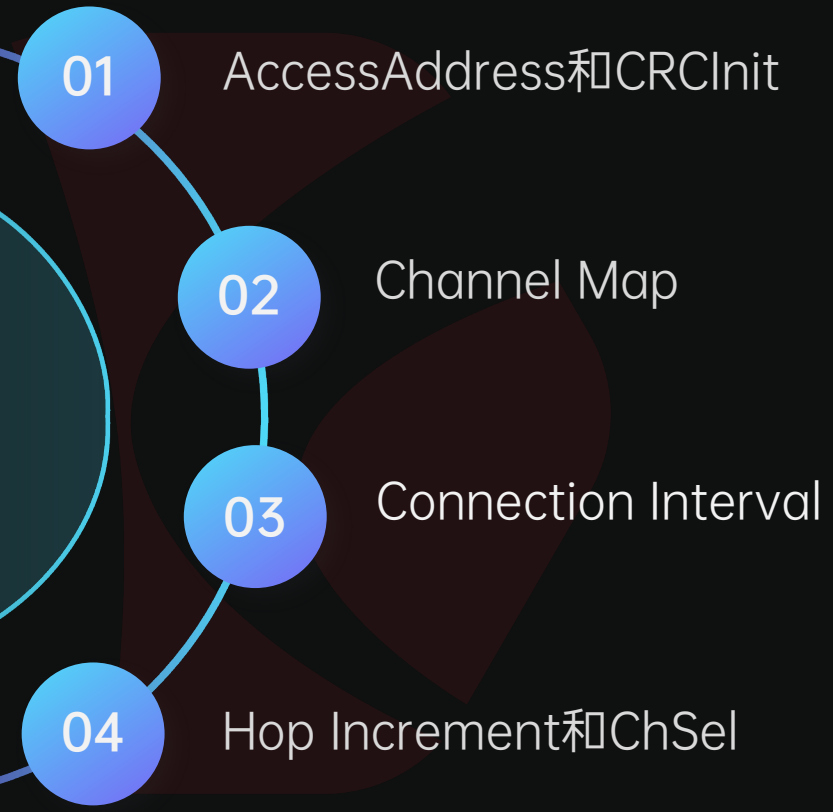
CONNECT_IND

Access Address:接入地址,建立链路层通信的标识
CRCInit:CRC 计算的初始化值
WinSize:主机发送第一包数据的时间窗口 transmitWindowSize = WinSize * 1.25 ms
WinOffset:主机发送第一包数据的偏移时间 transmitWindowOffset = WinOffset * 1.25 ms
Interval:连接间隔 connInterval = Interval * 1.25 ms
Latency:传输延迟。Slaver设备没有数据要发时，跳过一定数目的ConnectionEvent的值
Timeout:超时容忍度 connSupervisionTimeout = Timeout * 10 ms
Channel map:包含指示已使用和未使用数据通道的通道映射。 位值为 0 表示通道未使用。 位值为 1 表示通道已使用
Hop:跳频参数，数据信道选择算法中使用的 hop Increment，具有 5 到 16 范围内的随机值。
SCA:与时钟精度相关

BLE Hijack

劫持BLE通信

要劫持一个已经建立连接的BLE链路，首先要跟上BLE的跳频通信，也就是要推测出BLE的连接参数



BLE Hijack

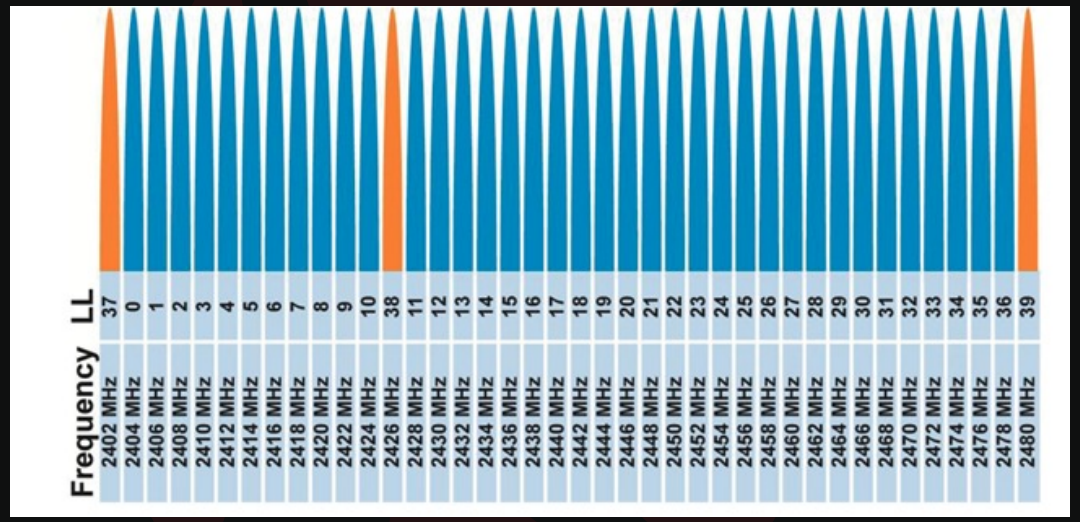
获取Access Address 和 CRCInit：直接在空中包中获取或者推算

01 接入地址可以抓空中包获取

抓用空包去计算CRCinit（crc24算法） 02

BLE Hijack

获取Channel Map：枚举BLE链路所使用的所有信道



01

监听同一个频段，每个信道停留4秒，总耗时 4*37 = 148s

BLE Hijack

获取Connection Interval：信道切换时间，这个间隔也被称为Hop Interval

01 跳频算法#1：$F_{n+1} = (F_n + hop) \bmod 37$

KEY:模运算的周期性

找一个合适的channel，监听两次报文的间隔时间处以37 02

BLE Hijack

获取Hop Increment：跳频参数，数据信道选择算法中使用

01

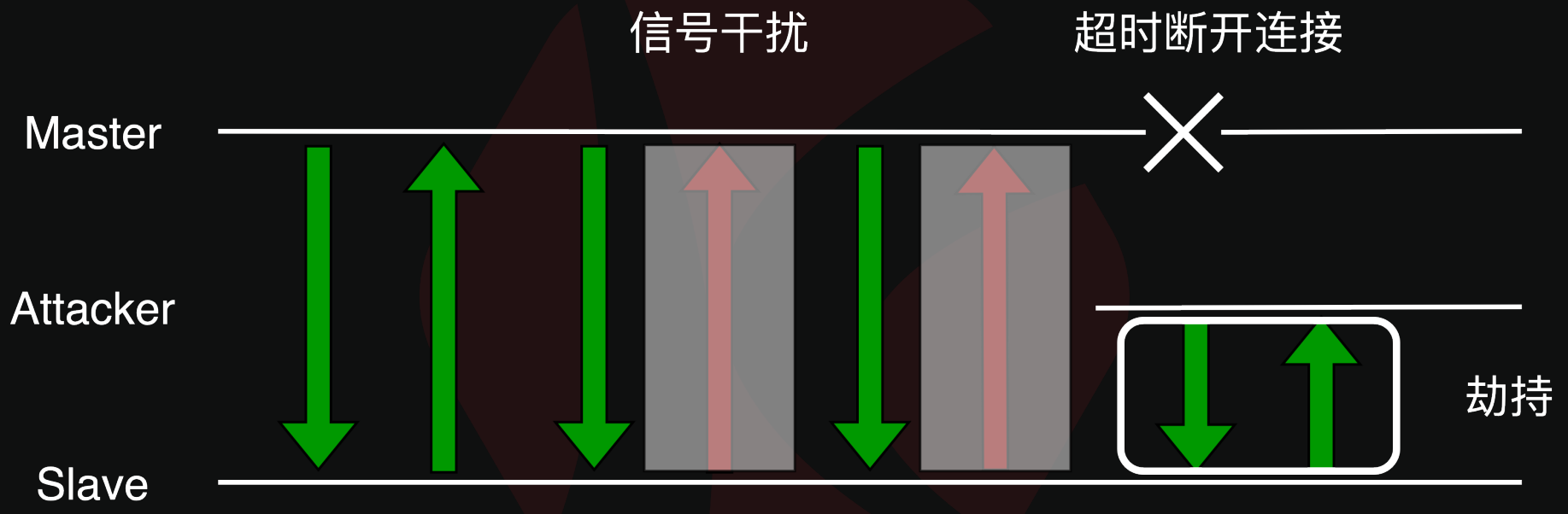
跳频算法#1：$F_{n+1} = (F_n + hop) \bmod 37$

范围是5-16的随机值，测试所有的可能值

02

BLE Hijack

劫持的关键：利用Supervision Timeout机制

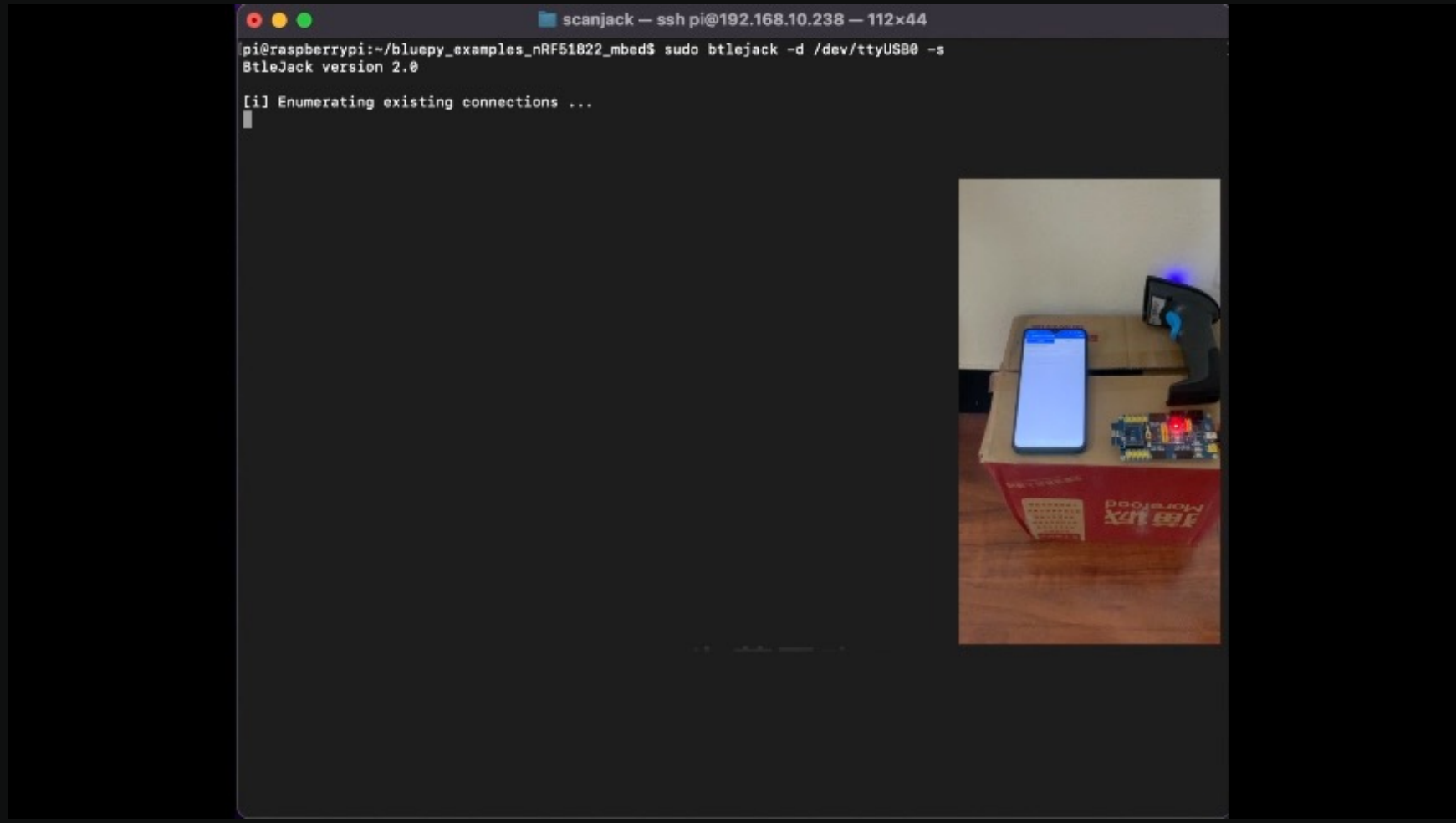


超时之后Master就会主动发送LL_TERMINATE_IND断开蓝牙连接

11418	360.839109	Master_0x9d51…	Slave_0x9d5195…	LE LL	26	Empty PDU
11419	360.839617	Slave_0x9d519…	Master_0x9d519…	LE LL	26	Empty PDU
11420	360.869353	Master_0x9d51…	Slave_0x9d5195…	LE LL	28	Control Opcode: LL TERMINATE IND

BLE Hijack

劫持BLE扫码枪的支付凭证



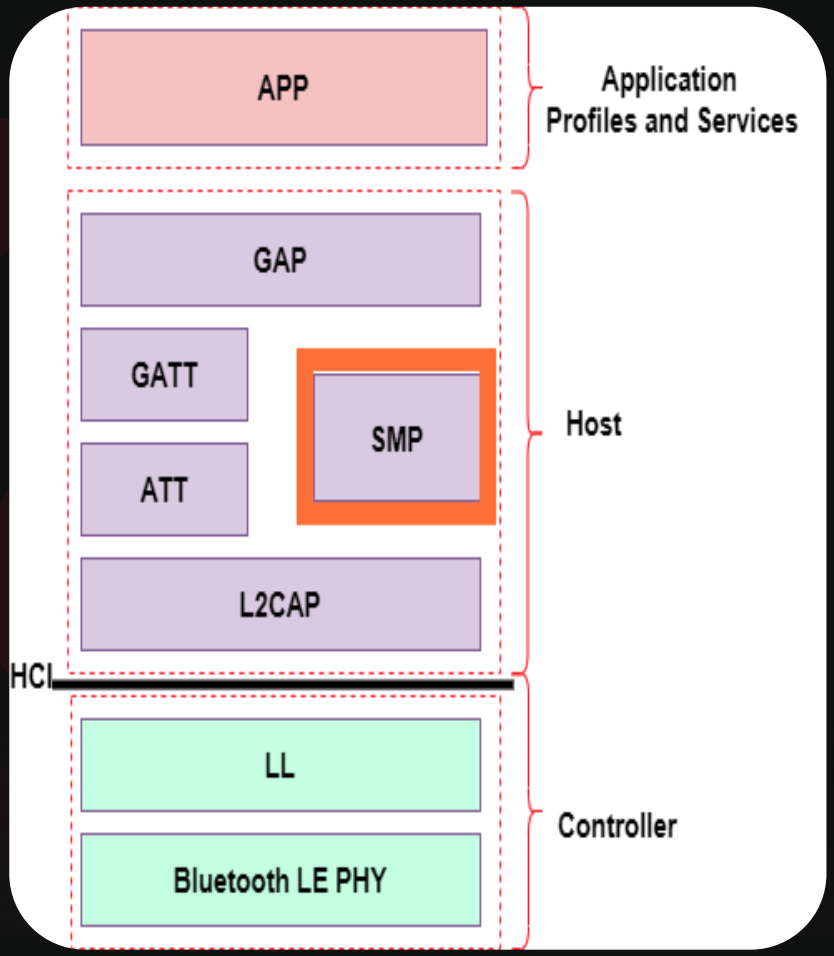
CHAPTER 3

无 · 感 · 入 · 侵

无感入侵

BLE无钥匙中继

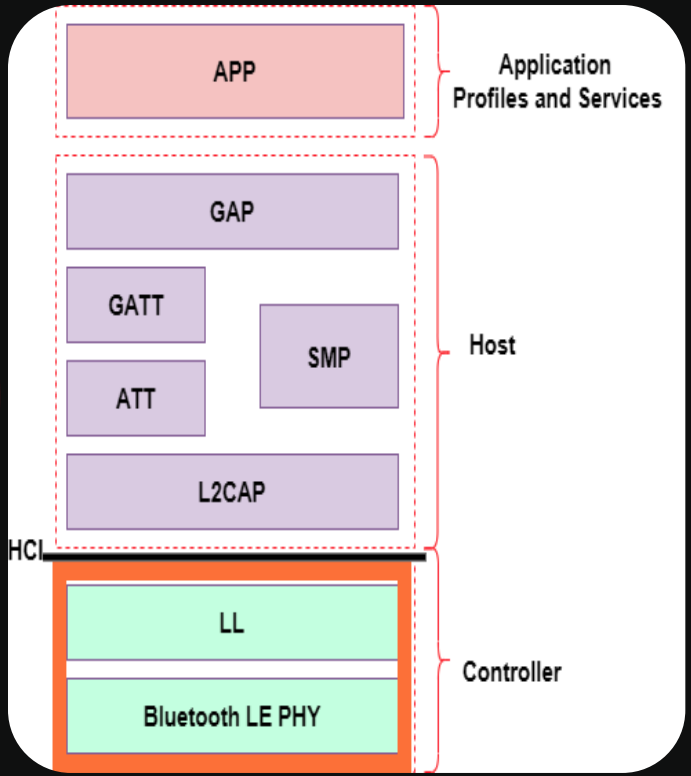
常规的BLE中继工具主要针对主机层中的GAP层进行中继攻击。设备厂商采用BLE自带的链路层加密功能就能有效阻止中继攻击。



深入Controller

那么我们攻击的目光自然就从Host转移到Controller，是否能够尝试对Controller进行中继攻击。

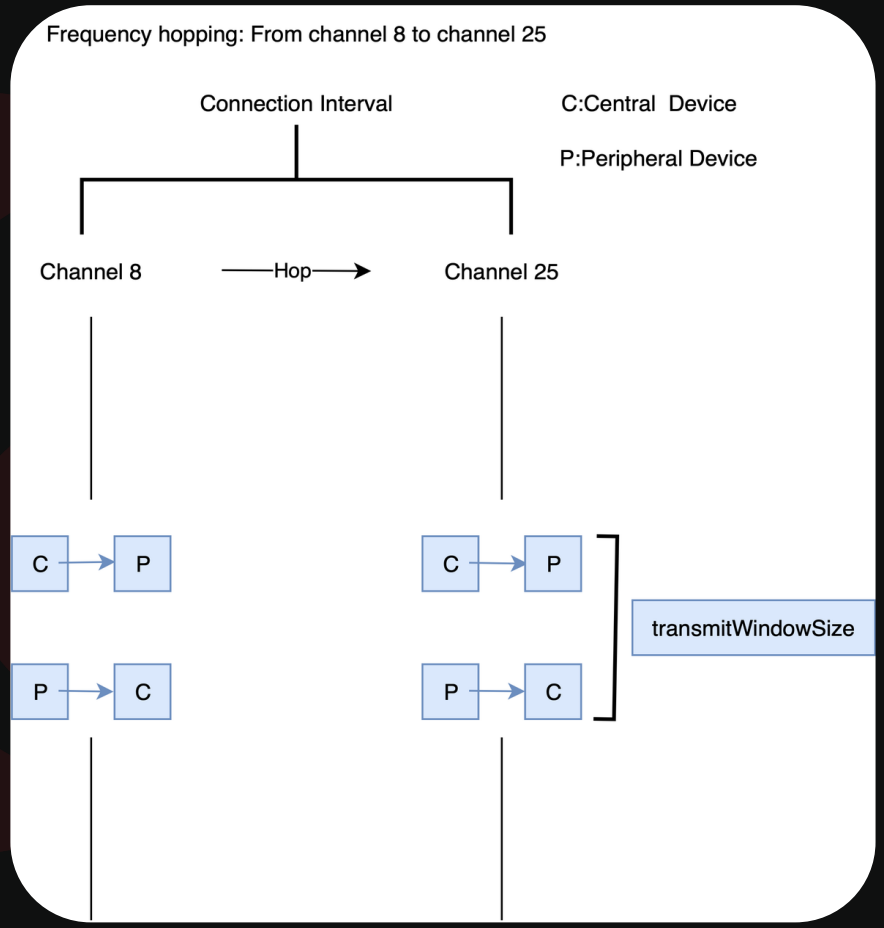
BLE就是一种无线电通信，参考RF中继的方案直接放大信号？



无感入侵

深入Controller

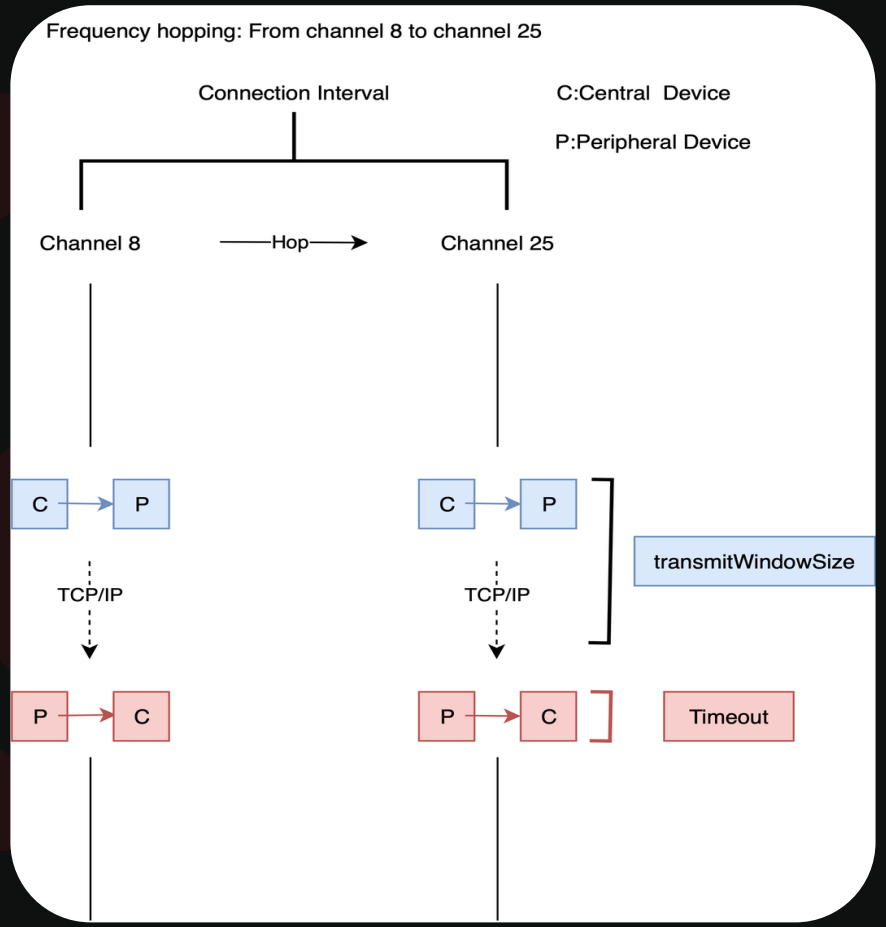
右图是一次BLE设备的连接事件。在该事件中，初始通信发生在Channel 8信道。Central设备向Peripheral设备发送信息，并在tranmitWIndowSize时间范围内等待响应。Peripheral设备接收到请求后，通过同一信道返回一个响应包。这次通信顺利完成后，通信跳转至Channel 25，继续进行相同的操作，并成功完成了下一次通信。



无感入侵

深入Controller

右展示了另一次BLE设备的连接事件，这次我们使用TCP/IP对BLE信号进行中继。初始通信发生在Channel 8信道，Central设备向Peripheral设备发送信息并等待响应。然而，由于TCP/IP延迟的存在，当设备响应返回时，Central设备已经跳转到了Channel 25信道等待下一次通信。由于超时事件超过了限制（connSupervisionTimeout = Timeout * 10 ms），Central设备认为Peripheral设备已断开连接，并发送终止指令，导致通信失败。



无感入侵

谜底就在谜面上

整个数据结构式BLE在物理信道上传输的全部，但其中最重要的Link Layer PDU，承载BLE所有Host层通信数据。其它几个参数都是用于维持链路，所以我们只要能够传递Linker Layer PDU，抛弃Access Address和CRCInit。

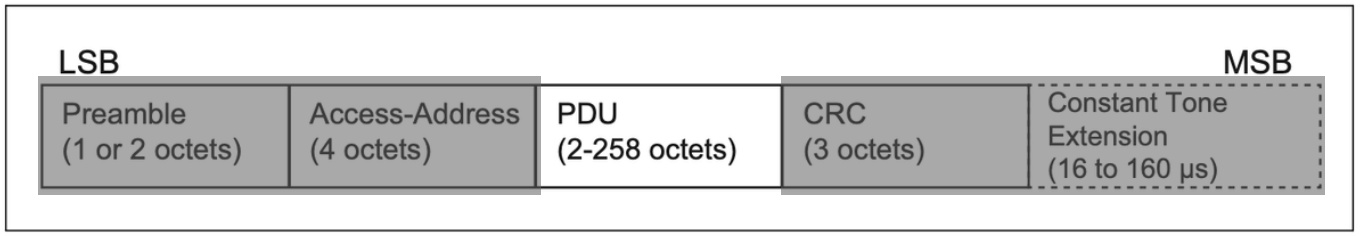
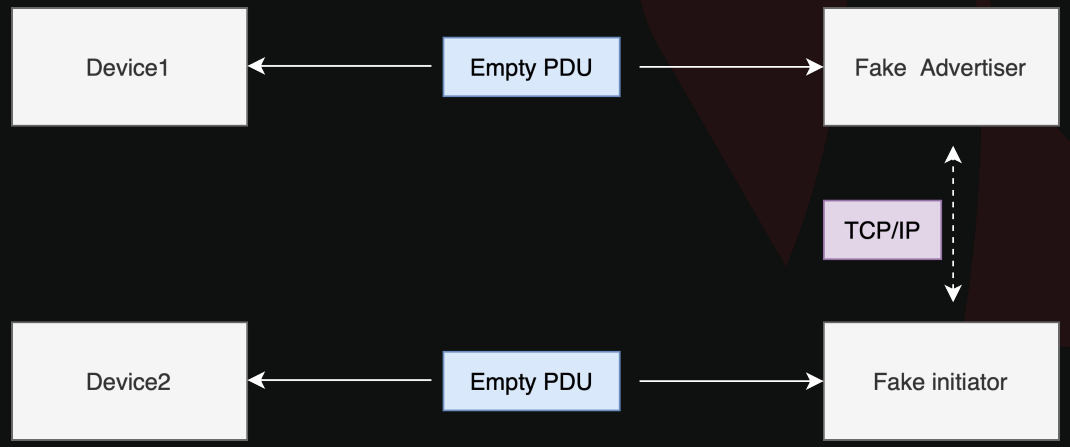


Figure 2.1: Link Layer packet format for the LE Uncoded PHYs

无感入侵

绕开马奇诺防线

1. 抛弃数据报文中的Access Address参数，表现为两边设备分别建立链路连接。两组设备分别协商连接参数。
2. 使用空包填充每一次Connection Interval(C->P和P->C)，有交互数据则填充数据。每一次连接事件(connInterval)不再受到TCP/IP延迟的影响，避免了无响应产生的connSupervisionTimeout。



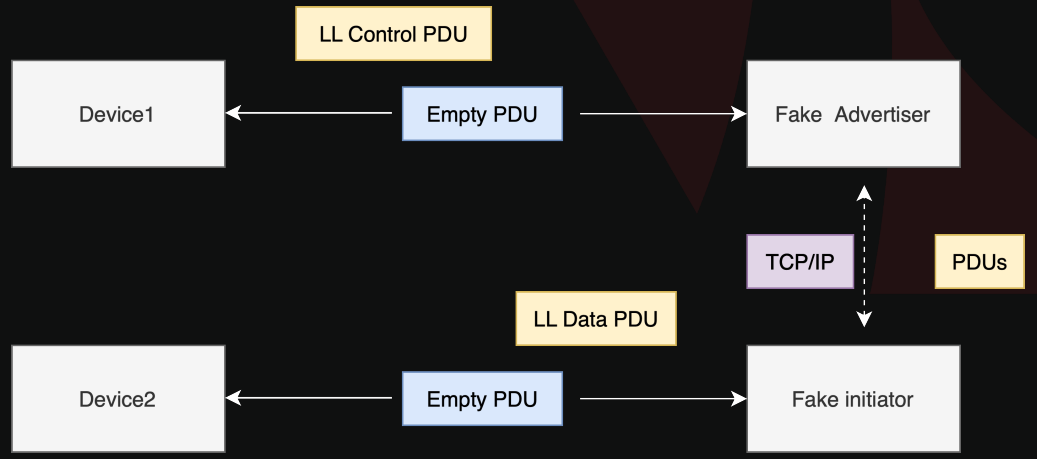
c1:e5:83:9c:5a:f6	e7:ba:be:19:79:72	LE LL	60	CONNECT_IND
Master_0x46670707	Slave_0x46670707	LE LL	35	Control Opcode: LL_FEATURE_REQ
Slave_0x46670707	Master_0x46670707	LE LL	26	Empty PDU
Master_0x46670707	Slave_0x46670707	LE LL	26	Empty PDU
Slave_0x46670707	Master_0x46670707	LE LL	35	Control Opcode: LL_FEATURE_RSP
Master_0x46670707	Slave_0x46670707	LE LL	26	Empty PDU
Slave_0x46670707	Master_0x46670707	LE LL	26	Empty PDU
Master_0x46670707	Slave_0x46670707	LE LL	26	Empty PDU
Slave_0x46670707	Master_0x46670707	LE LL	26	Empty PDU

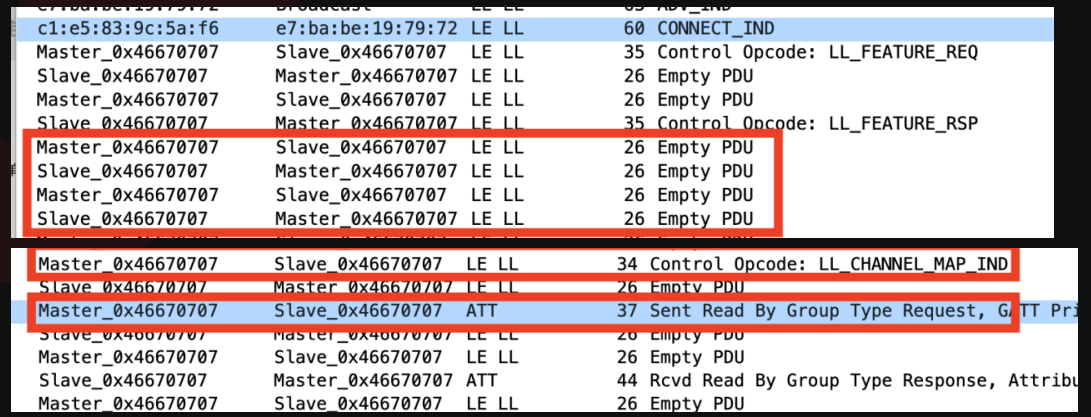
无感入侵

绕开马奇诺防线

3. 建立了空包的连接后，在让Device1和Device2他俩自己聊。两边设备自然而然相互发送Link Layer Control PDU或者Link Layer Data PDU，我们只负责中继这些Link Layer PDUs。

4. 具体实践过程中需要自己实现一个Linker Layer层代码。为了能够建立最基础的Linker Layer链路，需要实现CHM、跳频算法等。可建议参考蓝牙标准协议《Bluetooth Core Specification》Vol 6 Low Energy Controller，推荐开源项目Sniffle。





无感入侵

对抗加密链路

部分供应商在加密链路之后跳转到PHY 2M信道来躲避嗅探或者中继。具体而言，当信道进行加密后，链路层控制包（LL Control packets）将被加密，导致我们无法解析其中的内容。LL Control包含了关于信道状态和跳频信息的重要数据，因此无法准确追踪信道的改变。

为了防止由于跳频而丢失跟踪，我们选择在加密之前主动跳转到PHY 2M信道。通过采用这种策略，在加密链路之前固定选择PHY 2M信道，从而避免跳频过程对跟踪的影响。

无感入侵

视频演示

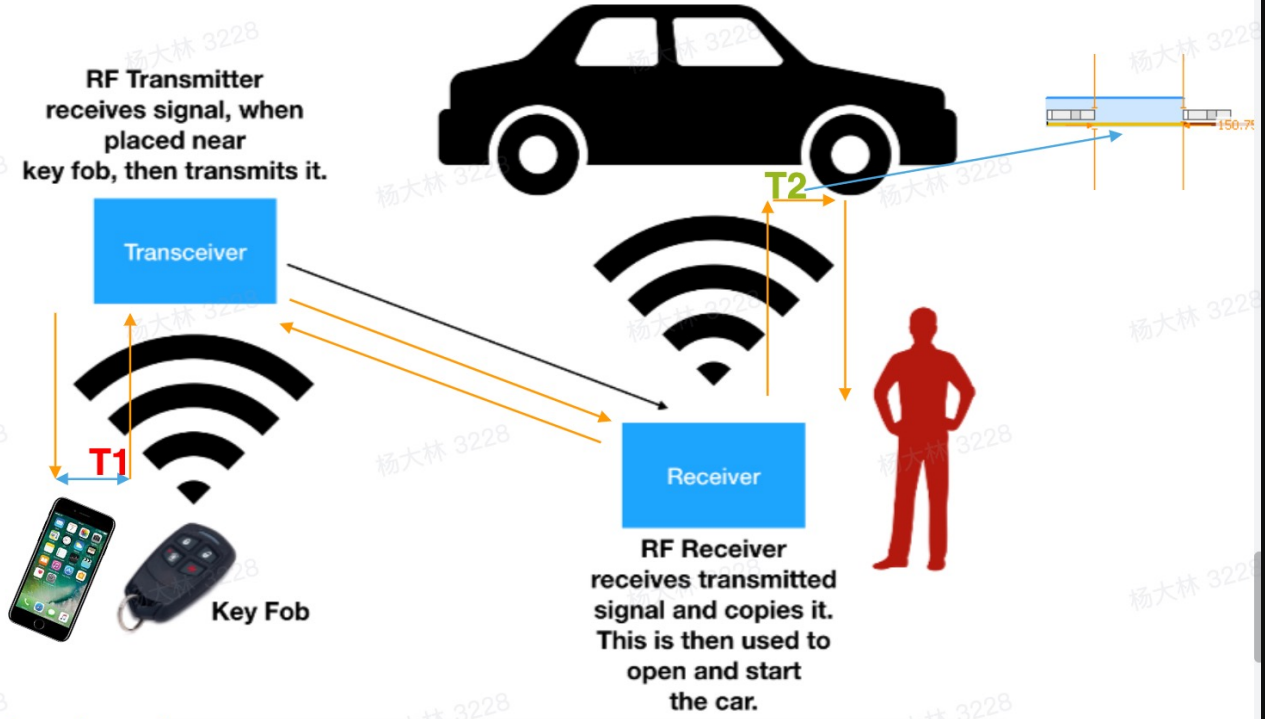
CHAPTER 4

未 · 雨 · 绸 · 缪

下面N厂的防BLE中继方案是否可靠?

WHY ANTI-RELAY WORKS - 1

- T2 >=150us
- T1 = T2+T_{others}
- $T1\_max=(T_{IFS}+2us)+2*T_{range}$
  $=(150us+2us)+2*2D*4ns$
  $=152us+16*Dns$
  $=152us+16*100ns$
  $=153.6us$(max range, 100M)
- $T_{others}=T1-T2=3.6us$



3.6us might be too big challenge to have info relayed

链路层中继主要难点

整合PDU数据结构如下，让建立连接的Master Slave自己维持连接。只介绍对象，不提供维持恋爱关系方法。

```
send_dict = {
        #"pdu_data":data1[4:],
        "time":msg.ts_epoch,
        "pdu_data":data1,
        "channel": msg.chan,
        "accaddr": aa.upper(),
        #"type":pdu_type,
        "pdu_type":dpkt.pdutype,
        "phy":msg.phy
    }

if send_dict['pdu_type']== "LL DATA CONT" and send_dict['pdu_data'][4:] == "": #跳过空包
    continue
```

链路层加密切换链路层参数、苹果设备、more data分包传输等

BLE安全检测工具

链路层中继

上述链路层中继实现便携、自动化检测

Controller劫持

对未加密链路层数据进行劫持篡改。

嗅探/伪造

嗅探广播，伪造广播以及主动连接请求。

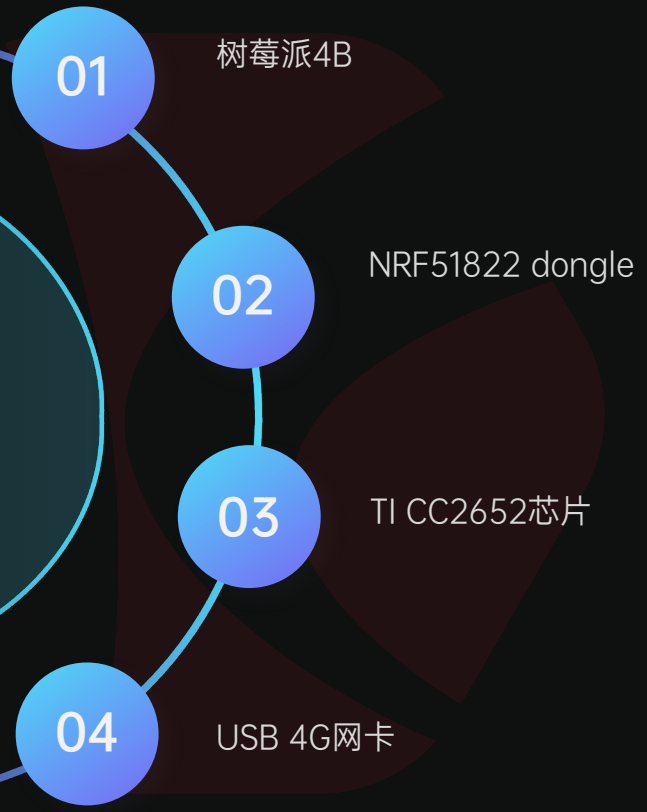
UUID识别

识别非标准（私有）的UUID服务

工具的背景

硬件架构

由于硬件开发周期较长，在工具硬件架构上我们采用模块化堆迭，实现锂电池供电、4G联网、python开发环境、触屏自动化控制。



工具的背景


软件架构
基于蓝牙模块固件+上位机开发模式
01
QT界面
02
python3
03
基于Sniffle开发
04
Hijack

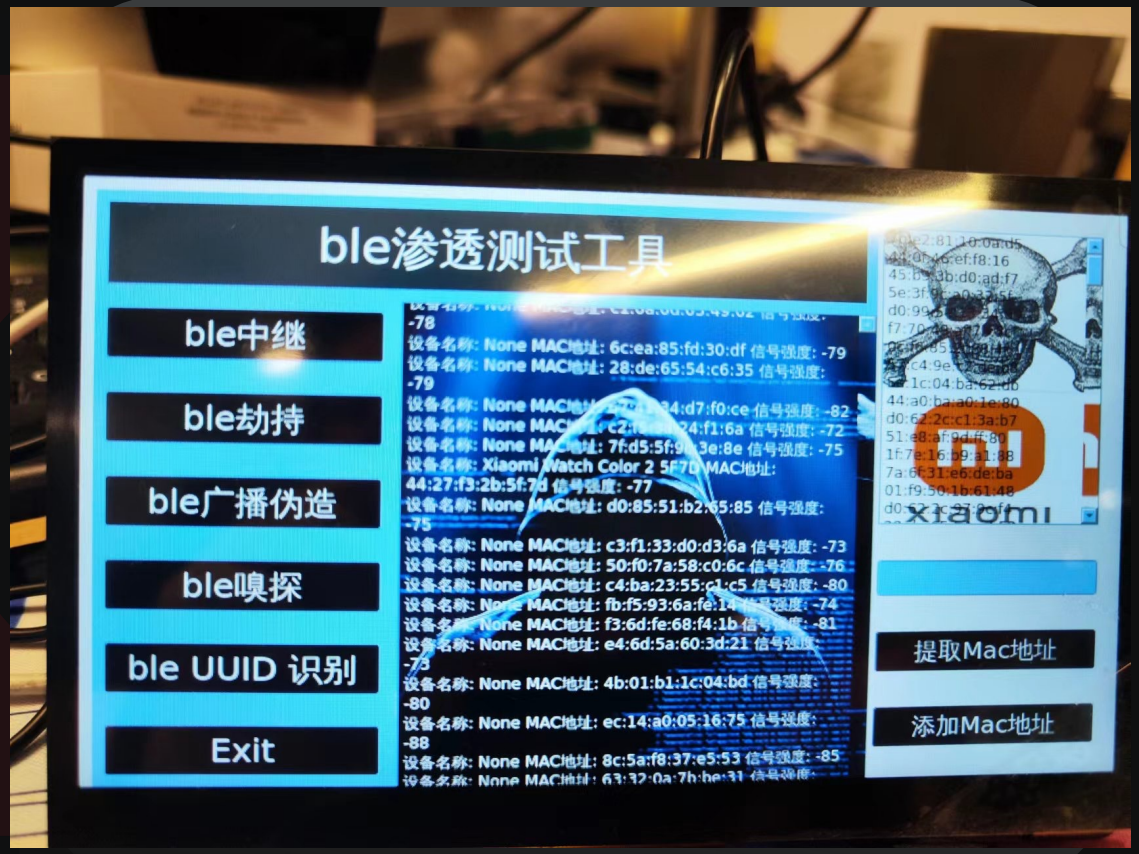
亮点和优势

BLE测试用例集成

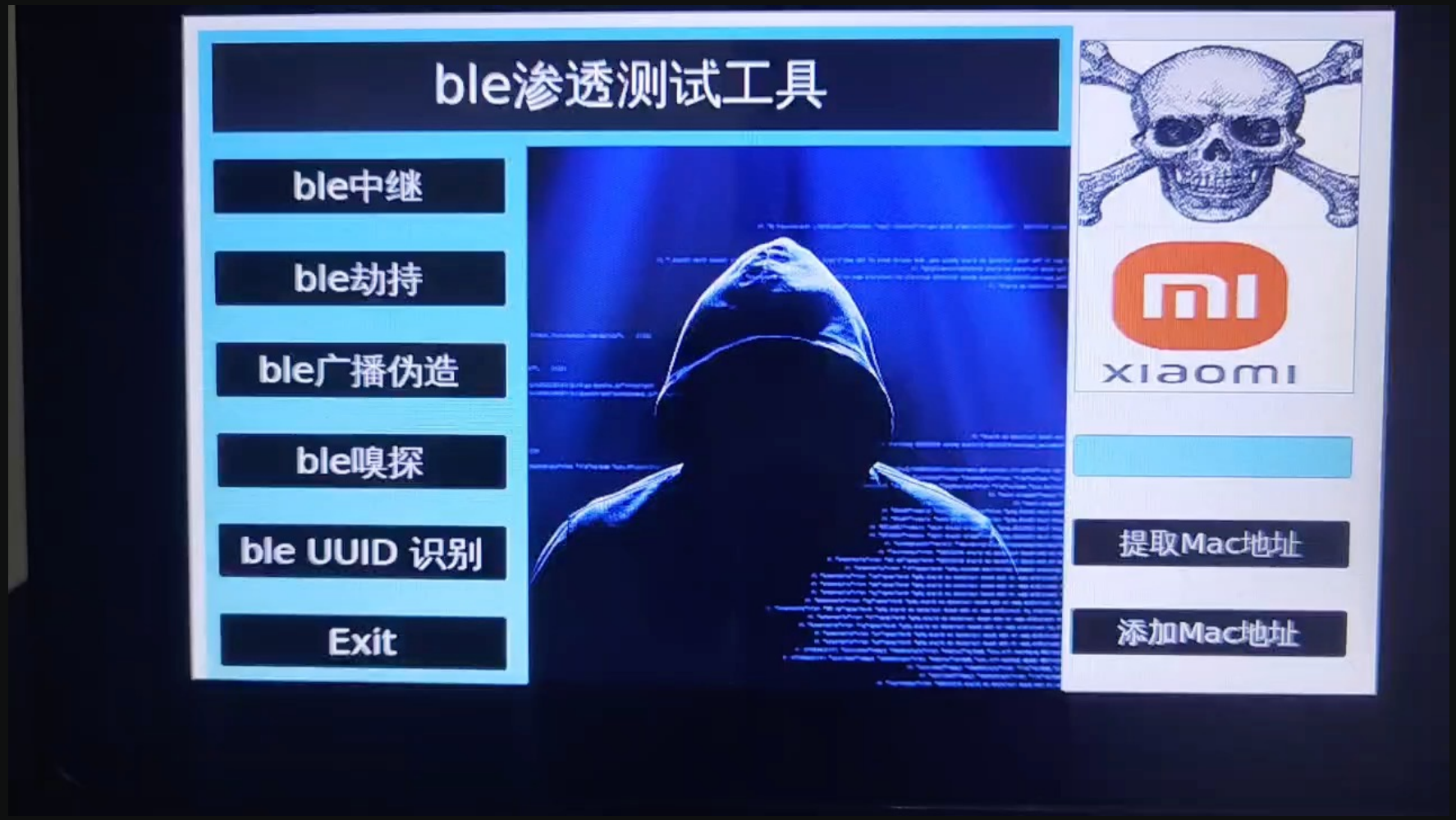
集成了BLE嗅探，BLE中继，BLE劫持，BLE广播伪造，UUID风险识别。后续继续集成可视化抓包，GATT层fuzz，BLE历史CVE测试等。

移动便携

更适用于网络环境恶劣，无电源等户外测试。比如地下车库，户外停车场等。



BLE中继演示


ble渗透测试工具
ble中继
ble劫持
ble广播伪造
ble嗅探
ble UUID 识别
Exit
xiaomi
提取Mac地址
添加Mac地址

工具主要难点

整合硬件平台，硬件开发周期较长，不同模块均须适配系统。

01 多种硬件模块工具融合。

主控软件环境、硬件性能。 02

03 网络环境配置，网络延时优化

应用场景

新能源汽车BLE钥匙，以及车载BLE设备的安全验收测试，以及安全威胁测试用例分析。

BLE

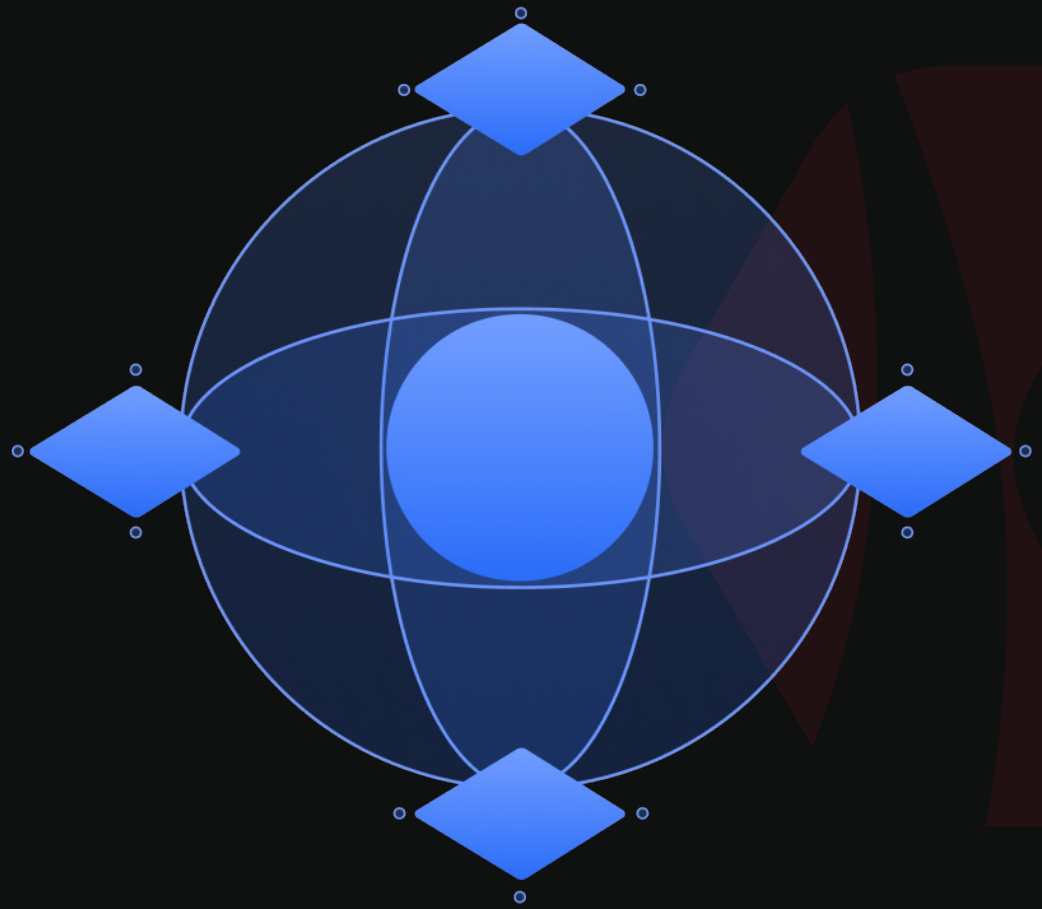
物联网设备

针对安全从业者，无论甲方乙方，在物联网BLE测试用例规划以及渗透测试过程中。

安全审计

车联网设备

未来规划



硬件集成

将硬件集成化，将硬件USB dongle 集成到主板。

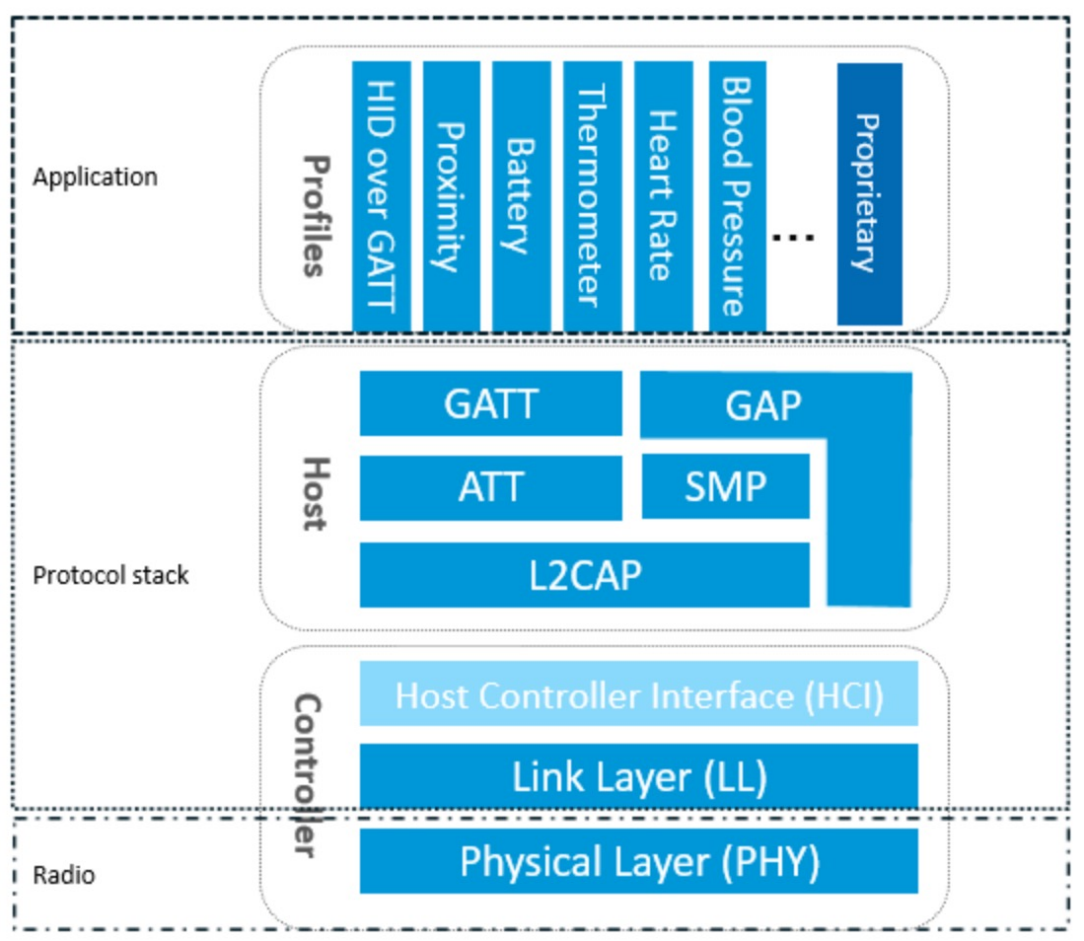
软件扩展

BLE 历史CVE检测，链路层fuzz，GATT层fuzz。

开源优化

Github开源软硬件方案，众人拾柴。

BLE的安全防御



链路层加密

加密链路中调用LL_CHANNEL_MAP_IND等指令更新链路参数（如ChM、WinSize等）中继者针对链路的sniffer无法跟上。

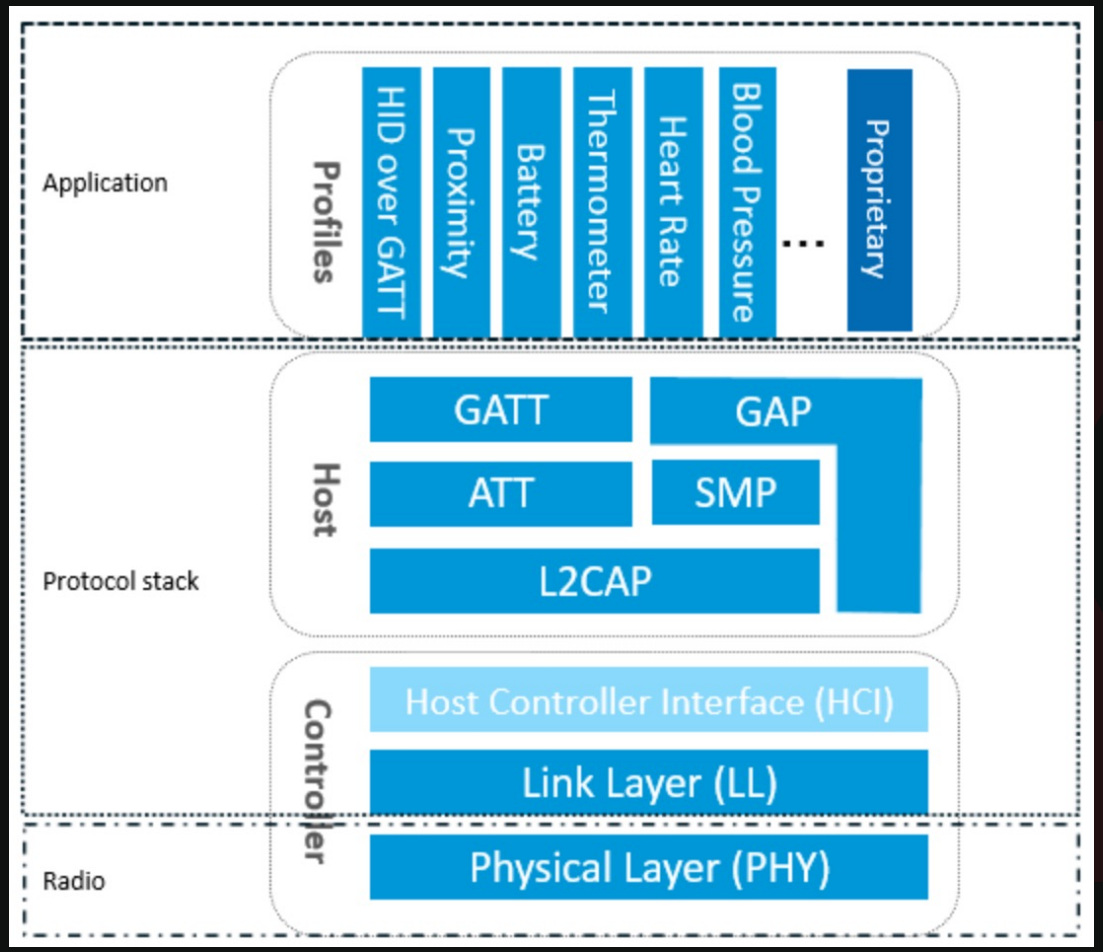
MD配置

More data 类似于TCP分片，告知还有数据包要传，请继续打开窗口。一个connection event包含多个数据包交互。

多连接标记

汽车使用多个蓝牙模块辅助标记连接设备。

BLE的安全防御



SMP

处理BLE设备之间的安全性，包括身份验证、加密和密钥管理等。SMP 的主要目标是确保蓝牙设备之间的通信安全，并提供对抗窃听、篡改和伪造等威胁的保护。

Service加密

重要服务数据进行加密后再进行蓝牙的加密传输。对于UUID的读写权限严格控制。

KCon

归源·智变

Returning to the source · Intelligent transformation

感谢您的观看！

THANK YOU FOR YOUR WATCHING