Wearable Device Bluetooth Sniffing

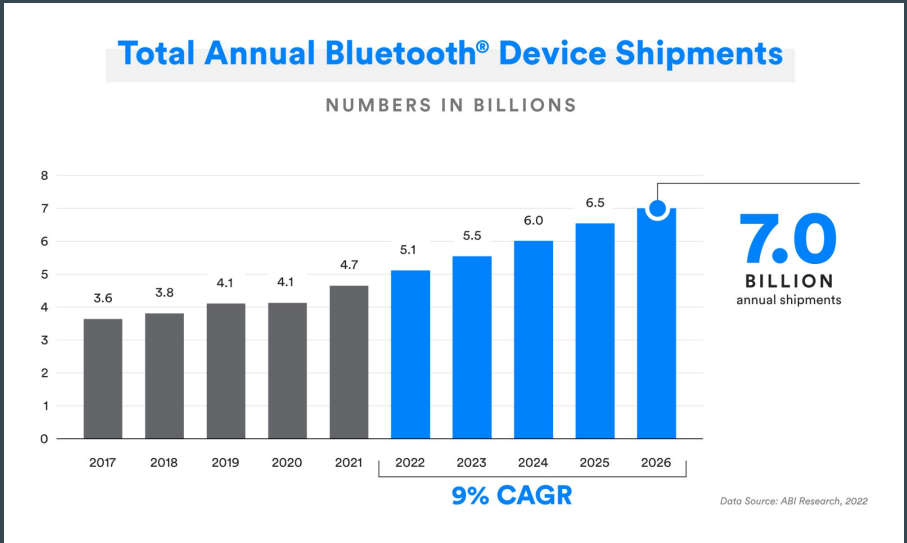
Hamza Ahmed, Sid Burre, Preethi Chidambaram, Matthew Whelan

Outline

Can traffic between wearable devices be analyzed to classify and determine wearable activity?

- Background
- Related Work
- Methods
- Results
- Discussion

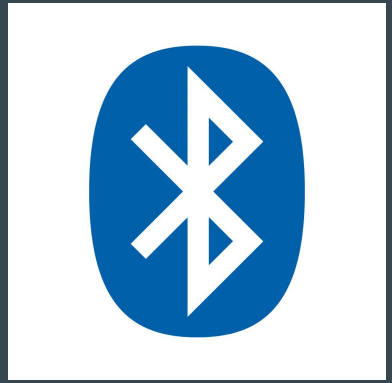
Background

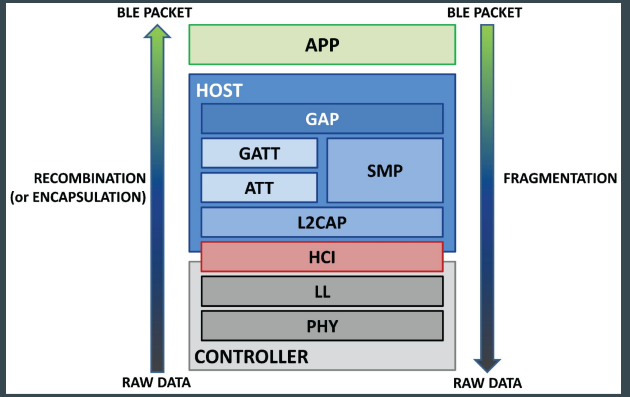


- Increasing use of bluetooth devices to track activities
  - Smart watches, headphones, in-car systems, smart home devices, health and wellness monitoring
  - Estimated 5.5 billion new devices this year
- Bluetooth Devices can contain sensitive information
  - Credit card info, authentication codes, phone call traffic
- Eavesdropping is a concern

Background: Bluetooth protocol

- Bluetooth Classic (2.4 GHZ)
  - Data-intensive or latency-sensitive situations
- Bluetooth Low Energy (BLE)
  - Optimized for low energy consumption
    - Short connections with high data transfer rates (1+ Mbps)
- Process
  - Master device pairs with one or more slaves
  - Slave communicates in master's piconet
    - Use same frequency hopping pattern





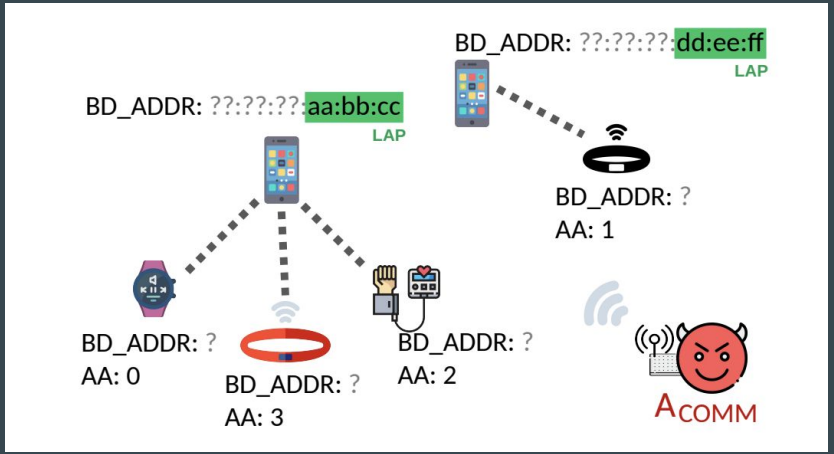
Related Work: Carplay Example

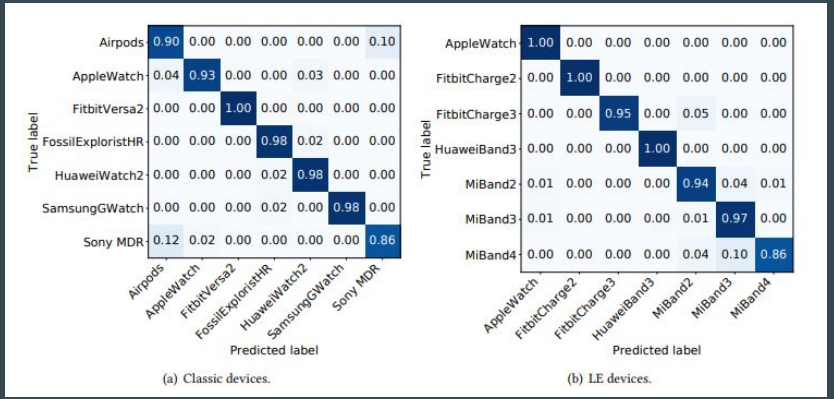
Digital Forensic Case Studies for In-Vehicle Infotainment Systems Using Android Auto and Apple CarPlay

- Analyzed wired and wireless connections between phones and in-vehicle infotainment systems
  - Mobile device uses WiFi or LTE to communicate with cloud server
    - Runs Man in the Middle (MitM) attack
  - Wireless communication uses WiFi and Bluetooth,
    - Collected and analyzed WiFi and BT packets
  - Research used limited configurations
    - Could not analyze encrypted WiFi traffic
- We decided to look deeper into only Bluetooth traffic in different environments

Every Byte Matters Example

- Analyzed traffic generated from various wearable devices to infer sensitive information about the user (such as health status, physical activity, and location)
  - Utilized a commercial wide-band scanner
  - 10,700 Bluetooth captures
- Was able to show great specificity in predicting the devices and actions of users





How do we depart from Every Byte Matters

Approach:

- Single Device
- Small number of “more sensitive” applications
- Focus on **using** application over simply **opening** applications

Applications	20Min, ASB, Alarm, AppInTheAir, AthanPro, AthkarOfPrayer, Battery, BeurerApp, Bild, Bring, Calm, Camera, ChinaDaily, Citymapper, DCLMRadio, DailyTracking, DenverApp, DiabetesM, DuaKhatq-mAlQuran, Endomondo, FITIVApp, FITIVPlus, FindMyPhone, Fit, FitBreathe, FitWorkout, Fitbit, Flashlight, FoursquareCityGuide, Glide, GooglePay, GooglePlayMusic, HealthyRecipes, HeartRate, HuaweiApp, Kaia, KeepNotes, Krone, Lifesum, MapMyFitness, MapMyRun, Maps, Medisafe, Meduza, MiApp, Mobills, Music, MyFitnessPalApp, NYT, NoApp, Outlook, PearApp, Phone, PhotoApp, PillReminder, PlayMusic, PlayStore, Qardio, RamadanTime, Reminders, Running, SalatTime, SamsungHealthApp, Shazam, Sleep, SleepTrack-ing, SmartZmanim, SmokingLog, Spotify, Strava, Telegram, Timer, Translate, Walgreens, WashPost, WearCasts, Weather, Workout.
Actions	AddCalorie, AddCarbs, AddFat, AddFood, AddGlucose, AddInsulin, AddProteins, AddWater, Browse, BrowseMap, CaloriesAdd, Coffees, EmailReceived, HeartRate, Leisure, LiveStream, NightLife, Open, PhoneCallMissed, PhotoTransfer, Play, Restaurants, Running, SearchRecipe, Shopping, Skip, SMSReceived, Sync, Walking, Workout.

Methods: Data Collection

Devices:

- Apple Watch Series 4 with Watch OS 9.3.1
- iPhone 14 Pro with IOS 16.0.2

Data Collection

- Collected in 3 minute segments
- Xcode Packet Logger extension used
- 10 sets of data collected per data type

Tests:

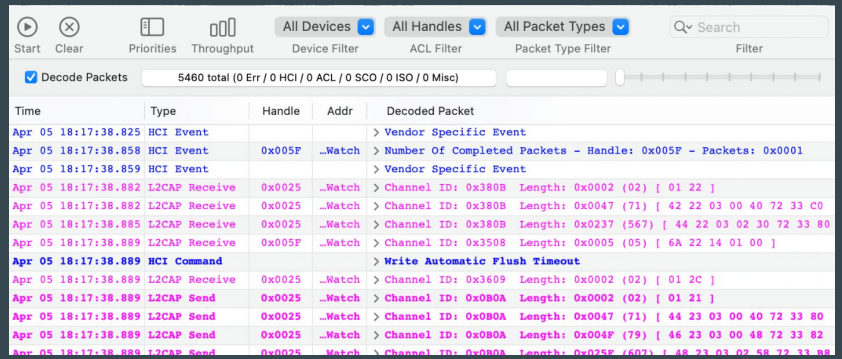
- Phone Call (answered)
- Phone Call (notification)
- Messages
- Duo notification
- ECG Data collection
- Strength Training
- Outdoor walks
- Spotify music playback

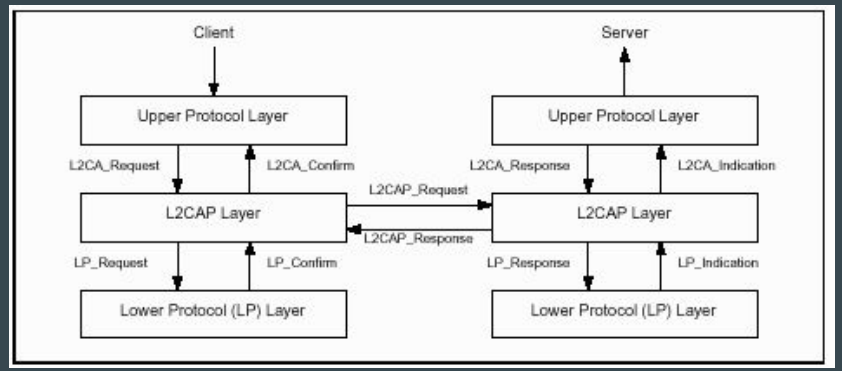
Methods: Data Processing and Packet Analysis

Separated Data to only contain L2CAP packets

L2CAP:

- Logical Link Control Adaptation Protocol
- Responsible for segmentation and reassembly of operation data
- Passes packets to the Host Controller Interface (HCI)





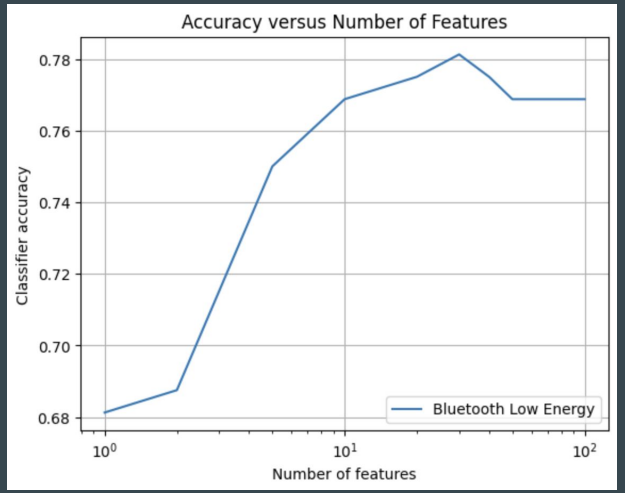
Methods: Machine Learning approach

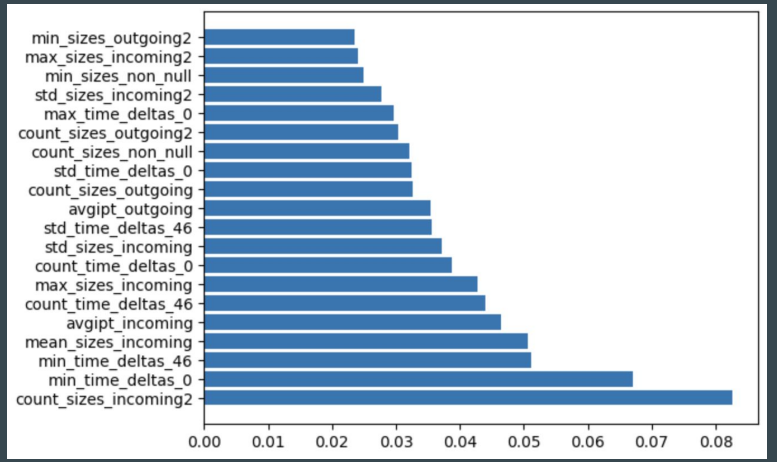
Goal: Determine the type of data being transferred from packet information

- Feature Extraction - various functions to extract specific stats:
  - deltas - time difference between consecutive packets
  - extract_bins - organizes packet sizes into certain ranges
  - avgIPT - average interpacket time for incoming and outgoing packets
  - stats - obtains min, max, mean, standard deviation of packet sizes and time deltas
- Used different ML models to train and test
  - naive-bayes, svm, knearest neighbors, random forest

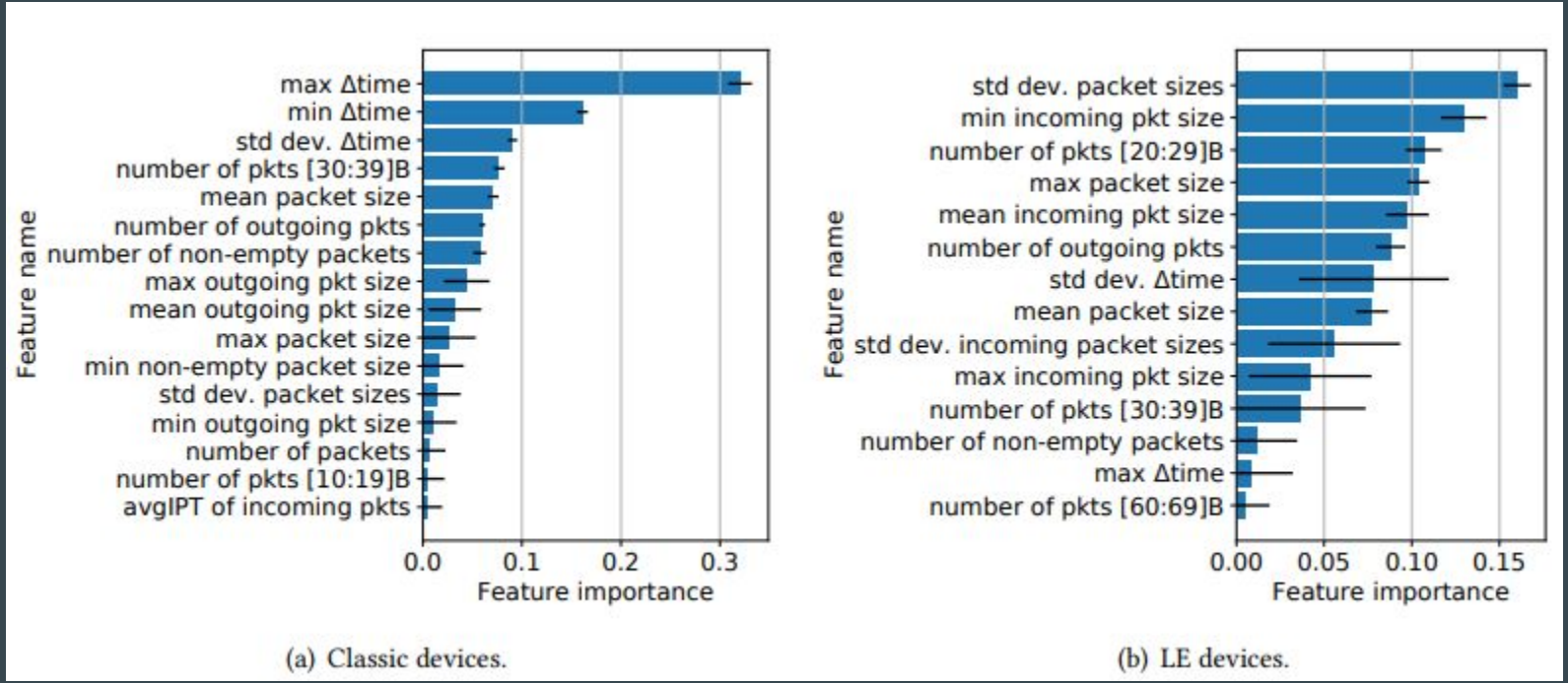
Results

- Random forest achieved 78% accuracy
- Naive-bayes achieved 93.75% accuracy
- Extracted feature importance
  - Number of incoming packets had highest importance





Comparison to Every Byte Matters



(a) Classic devices.

(b) LE devices.

Limitations and Extensions

Limitations:

- Collecting sufficient data difficult to train robust model

Future Work:

- Collecting more descriptive data
  - Improve feature importance and prediction accuracy
- Expand to different devices and smartwatches
- Experiment on different types of notification data (different apps)
- Testing every byte matters model on our data
  - Can their model generalize well to new data?
- Deploying our model to consume a stream of data