Third eye: Inferring the State of Your Smartphone Through Wi-Fi

Abhishek Kumar Mishra, Mathieu Cunche

Third eye: Inferring the State of Your Smartphone Through Wi-Fi

Abhishek Kumar Mishra
INSA-Lyon, Inria, CITI Lab.
Lyon, France
abhishek.mishra@inria.fr

Mathieu Cunche
University of Lyon, INSA-Lyon, Inria, CITI Lab.
Lyon, France
mathieu.cunche@insa-lyon.fr

***Abstract*—Wi-Fi is one of the most notable and prevalent wireless technologies today. Smartphones and other Wi-Fi-enabled devices find nearby networks using management frames known as probe-requests. In this paper, we infer the state of smartphones by passively monitoring their transmitted probe-requests. We leverage the differential behaviour of probe-request bursts and their content, based on their device states such as active/static screen and Wi-Fi/power-saving mode ON/OFF. We use a Random Forest based approach that can successfully predict smartphone states just leveraging individual bursts. Based on an evaluation using a real-world dataset of more than 200 smartphones (having a variety of operating systems), with ground truth data available, we show that our model reliably predicts states with accuracy $\geq 98\%$.**

***Index Terms*—Wi-Fi, probe-requests, privacy, smartphones, state, activity recognition**

I. Introduction

Beside the ubiquitous availabilty of network connectivity, the wide adoption of the Wi-Fi technology has brought a number of unforseen application where Wi-Fi signals are leveraged to obtain information on individuals, such as user trajectory and pedestrian flow estimation [1], [2], human activity recognition [3], and thus sometime threatening users' privacy [4], [5].

Contemporary Wi-Fi devices rely on an active scan method to discover nearby networks. To find nearby Access Points (APs), a device transmit frames called probe-requests during active scans, potentially exposing sensitive data [4]–[7] in clear. Intercepting probe-requests is relatively simple and can be achieved over off-the-shelf sniffers. Moreover, probe-requests falls into public network traffic sniffing regulations which motivated us to explore the same in this paper. Vendors started to implement countermeasures to thwart privacy threats : for instance as address randomization [7] to mitigate MAC-based device tracking.

In this paper, we show that the *state* of user devices, can be inferred with a high accuracy when considering a large pool of available smartphones in the market. Exploiting passively collected probe-request frames, we are able to successfully infer (states): i) if a target device is being actively used or not, ii) whether the power saving mode is present, and, iii) whether the Wi-Fi connectivity is enabled[1].

We notice a wide range of the device's state-specific features such as temporal characteristics of a probe-request burst, sojourn time of advertised randomized MAC addresses, and, content of a probe-request. The changes in these features with respect to its state, could be observed irrespective of the device manufacturer. The contributions of the paper are the following:

- We identify various information/features from passively captured probe-requests which show differential behaviour with devices' states (cf. Section V).
- Using the derived features, we introduce a novel machine learning approach to predict a device's state from its probe-request advertisements (cf. Section VI).
- We demonstrate that state inference is possible with $\geq 98\%$ accuracy across a large set of devices, including all major smartphone operating systems (cf. Section VII).
- Finally, we suggest various countermeasures that can be adopted by device manufacturers (cf. Section VIII).

II. Background

In the following, we explore the Wi-Fi active scanning procedure, focusing specifically on probe-request messages. We investigate its temporal behaviour, content, and, random MAC addresses.

A. Wi-Fi Active Scanning

Wi-Fi-enabled devices employ active scanning to discover nearby wireless networks and their access points (APs) [8]. During active scans, Wi-Fi-enabled devices search for available networks by transmitting management frames referred to as probe-request frames.

When an AP receives a probe-request frame matching its Service Set Identifier (SSID) or advertising a wildcard SSID, it replies with a probe-response frame. Upon receiving probe-response frames from nearby access points, the client can assess its options and choose a network to connect to based on factors like signal strength, security settings, and user preferences.

To save energy, devices broadcast probe-request frames periodically. Figure 1 illustrates the active scanning process from

[1]Even if the user disabled Wi-Fi connectivity, the Wi-Fi interface may still be active for geolocation purposes (https://www.cnet.com/tech/mobile/stop-android-4-3-from-always-scanning-for-wi-fi-networks/).

Name	Number of devices	OS	Version	Vendor
`Pintor` dataset	22	Android Android iOS	4.02 - 11 Oxygen 11 12.05 - 14.6	Samsung, Xiomi, Huawei, Google One Plus Apple
`Furious` dataset	205	Android Windows Phone iOS	4.1 - 10 8.1 10.1 - 13.1	Samsung, Xiomi, Huawei, Google, One Plus, HTC, Motorola, LG, Oppo, Sony, Aquos, ASUS, ZTE, Blackberry, Alcatel Nokia Apple

TABLE I: Investigated Datasets

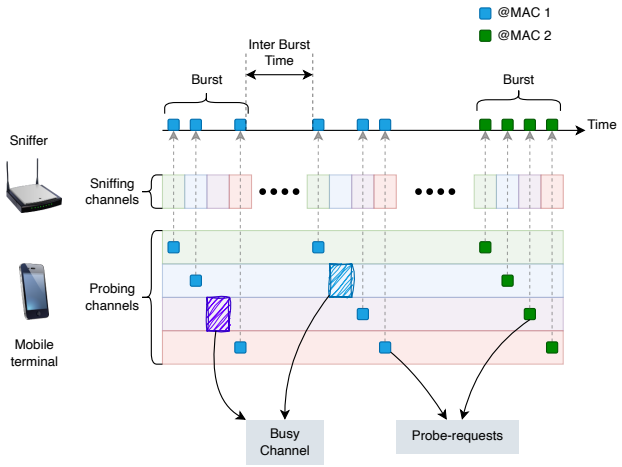


Fig. 1: Wi-Fi active scanning

a Wi-Fi device over time. Mobile devices repetitively send probe-requests on available channels to receive responses from all accessible access points. Each device performs multiple rounds of active scanning across the available channels.

Probe-requests include fields called information element (IE), that allows devices to advertise their capabilities and connection preferences. The content within IE fields can be potentially unique to a particular device [7] or its state.

B. Probe bursts and MAC randomization

Active scanning rounds typically last for a duration on the order of a few seconds, contingent on factors such as the number of known access points and the availability of non-busy channels. As depicted in Figure 1, active scanning consists of multiple *bursts* of probe-requests, captured by the sniffer on channels, with the MAC address of individual probes within a burst remaining consistent.

However, for privacy reasons [7] the MAC is likely to change (randomize) either in subsequent bursts or after a certain number of bursts, a process known as MAC randomization, [8, Sec. 12.2.10]. The longer a device keeps discovering nearby networks, the more probes circulate from the same device, increasing the number of randomized MACs.

The number of bursts/probes a device advertises with a given address varies with manufacturers and devices' state. The inter-burst time (IBT), or the period between successive bursts, is also variable and dependent on the state and the manufacturer.

III. Related works

There have been works inferring user activity from network metadata [6], [9]–[14]. Fingerprint and timing-based attacks are carried out to predict private individual activities like cooking, showering, etc. in homes by eavesdropping wireless sensors [9]. [10] also infer various user activities like walking, exiting the premise, etc. in smart homes from passively sniffed Wi-Fi, ZigBee, and, BLE data. RSSI fingerprints from Wi-Fi sensing are used for detecting such user activities too [13]. Users detected on camera footage are linked with their corresponding smartphone identifier like the MAC address [14]. Wi-Fi-based side-channel information is used to infer mobile passwords [11]. Vulnerabilities have been discovered in the information fields (IE) of the Wi-Fi probe-requests that could reveal private information of the user like language, and socioeconomic status [6].

An active timing attack for Bluetooth using ping flooding is proposed [15], in order to detect device changes in states like locked/idle/active e.t.c. states. They test their solution on 3 smartphones. A small-scale demo with 4 Android smartphones for inferring mobile screen ON/OFF was attempted using Wi-Fi probe patterns [16].

None of the works in the literature have showcased inferring detailed device states, like Wi-Fi ON/OFF, status of mobile batteries, along with the active usage/static mode of the device for a wide range of devices, OS versions, capture settings etc. Moreover, our approach is passive and relies on a capture via off-the-shelf hardware, while concurrent approaches require advanced hardware to get access to lower layer signal information (e.g. CSSI).

IV. Datasets and threat model

In this section, we first have a look at the datasets that we investigate, then we describe the threat model that we consider. We build upon the described threat model to introduce an attack for successfully inferring smartphones' *states*, in the next section.

A. Datasets

We utilize two extensive datasets, named `Pintor` and `Furious` dataset, released in 2022 and 2021 respectively. They capture probe-requests from the smartphones in a variety

of *states* as depicted in Table II. We detail metrics concerning both datasets in Table I.

Pintor dataset: `Pintor` dataset has 22 popular device models in practice, which were sniffed in 6 devices *states* [17]. It contains 20-minute duration captures of individual devices in a Faraday cage.

There are active-screen states (A, PA, and WA) and inactive-screen states (S, PS, and WS) (cf. Table II). In states PA and PS, the device keeps the power-saving setting active with the screen being active and passive respectively. For WA and WS states, the device has the Wi-Fi interface switched off in various screen modes. Each device configuration is observed in the three non-overlapping channels (1,6, and, 11) of the 2.4 GHz frequency band.

Mode	Screen ON	Power-saving ON	Wi-Fi ON
A	Yes	No	Yes
S	No	No	Yes
PA	Yes	Yes	Yes
PS	No	Yes	Yes
WA	Yes	No	No
WS	No	No	No

TABLE II: Device's *states*

Furious dataset: `Furious` dataset contains probe-requests from 205 devices [18]. Tests are conducted in a Faraday cage with 4 Wi-Fi cards collecting traffic on channels 1, 6, and, 11 in the 2.4 GHz band while on channel 36 in the 5 GHz band.

They conduct experiments in 4 states that include active (A) and static (S) screen states. In state S, the phone screen is locked for 20 minutes or 200 probes, whichever is first. During state A, the screen is active and a robotic arm over the device screen interacts with the device at a rate of 1 interaction per 5 seconds. Both A/S states are recorded with Wi-Fi ON/OFF option separately, yielding in states WA and WS respectively.

B. Threat model

We assume a passive attacker that controls receiving sniffers in the targeted area, potentially over extended periods of time. As off-the-shelf sniffers are cheap and accessible, the attacker can even scale passive-sniffing over a sizeable geographical area like homes, shopping malls, public squares etc., affecting a multitude of users. Inferring states of devices in public places or other targeted areas could lead to larger security concerns like user harassment, targeted advertisements, and, profiling.

We assume that the attacker might have already trained on various smartphone device models that are possibly available to target users. We assume the smartphone is left unmodified, and the attacker cannot physically access it. They just listen to emitted probe-requests on various frequency bands. We assume that the attacker does not possess abilities to link randomized MAC addresses [7] from a particular device.

V. Inferring smartphone states

In this section, we first motivate the exploitation of burst-based metrics for the purpose of passively inferring the smartphone's state. Then, we proceed to define our proposed model and select features that could discriminate various states.

Characterizing probe-bursts: We first combine the probe-requests captured in various *active* and *static* usage modes, from both `Pintor` and `Furious` dataset. Basically, states: A, PA, and, WA in Table II refer to active smartphone usage, while states: S, PS, and WS denote static smartphone usage.

$Metric$	$Feaure$	$Notation$
Burst-based	The duration of the burst	T_b
	Size of the burst	S_b
	Number of bursts using the MAC	N_b
	Inter-burst time	IBT
Content-based	Number of present IE fields	N_{ie}
	Inter-burst seq. number gap	SEQ_{ib}
	Bursts' mean seq. number gap	SEQ_b
MAC-based	Sojourn time of burst's MAC	T_{mac}

TABLE III: Considered features.

For initial investigation, we consider six features (cf. Table III) that characterize the behaviour of smartphone's probe-request burst:

- *The duration of the burst (T_b):* T_b measures the duration for which a single burst was observed at the receiving sniffer.
- *Size of the burst (S_b):* S_b denotes the number of frames sent by the sending device, which were part of the same burst.
- *Number of bursts per MAC (N_b):* MAC address changes in the active scanning state, after a few bursts of probe requests with a particular randomized MAC. N_b measures the number of bursts that were observed with the same MAC as that of the burst in consideration.
- *Inter-burst time (IBT):* The time gap between two successive probe-request bursts from a device is denoted by IBT.
- *Number of present IE fields (N_{ie}):* The probe-request frames do contain IE element fields which contain information about the device's capabilities and preferences. Out of around 256 specific elements that a smartphone could specifically advertise, in practice, many of them are not included. N_{ie} measures the number of non-empty IE fields for a random frame chosen from the burst.
- *Inter-burst seq. number gap (SEQ_{ib}):* SEQ_{ib} is the difference between the sequence number of the last and the first frame for a consecutive burst-pair.

As we observe in Figure 2, there is a distinct differential probe-request bursts' behavior observed between active and static usage modes in smartphones. Bursts are sent with distinct temporal and content-wise behaviour for different states for all six features.

For instance, the figure shows that bursts tend to be shorter in terms of the duration and the number of frames in the active mode. Devices probe frequently during active usage and tend to send more information in their IE fields. These findings hold across all present OS's that are there in `Pintor` and `Furious` datasets.

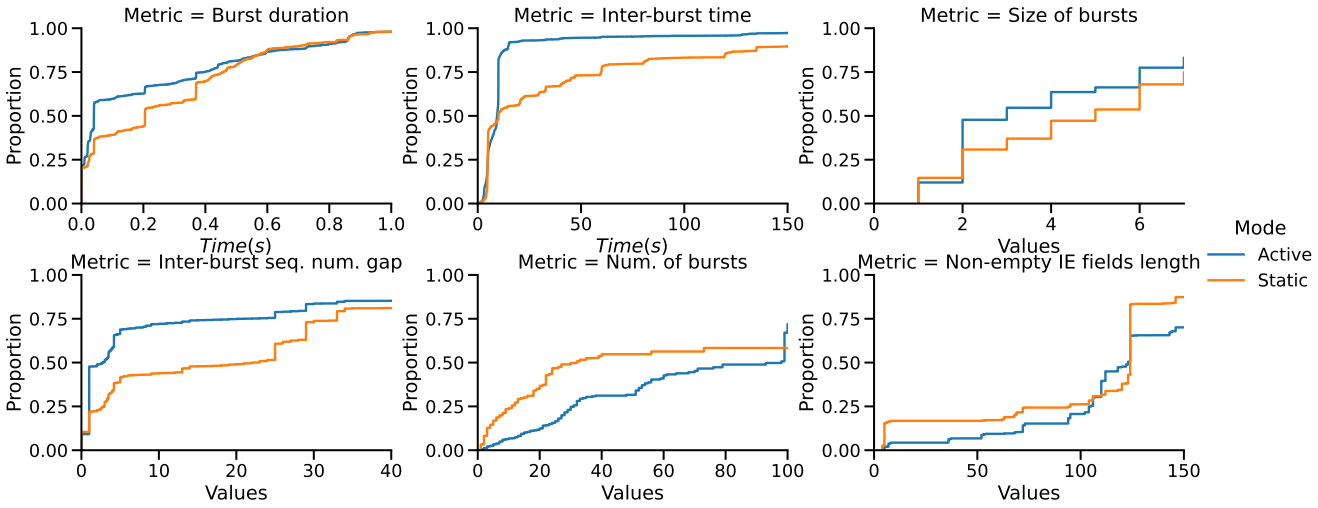


Fig. 2: Differential probe-request bursts' behavior observed between active and static usage modes in smartphones

The differential behaviour can be attributed to the increased number of applications/services that reach out for connections during active device usage, while still trying to conserve energy. Subsequently, it motivated us to learn these unique trends in burst's behaviour for classifying devices' states, just using captured probes.

Model and Feature selection: From our initially considered six features, we complement them with two more features that contribute to identifying *state*-specific behaviour:

- *Bursts' mean seq. number gap (SEQ_b):* SEQ_b measure the average difference in sequence numbers found in successive frames of a single burst.
- *Sojourn time of burst's MAC (T_{mac}):* T_{mac} denotes the time for which a particular MAC address was observed.

We select a Random Forest (RF) based model to learn the patterns and turn the *state*-prediction as a multi-class classification problem. RF is fast, robust to outliers, can identify non-linear patterns, and, does not suffer from overfitting even if more trees are appended [19].

As stated in Section IV-B, we stick to a relatively weaker attacker with no MAC association ability. We observe that a large chunk of MAC addresses (40%) change their MAC addresses every burst, making the adversary unsure about the device sending previous bursts from the burst in the consideration. Considering this inability, we drop two features: Inter-burst time (IBT) and Inter-burst seq. number gap (SEQ_{ib}) from the primary investigation of results in Section VII, leaving us with the feature-set: $\{T_b, S_b, N_b, N_{ie}, SEQ_b, T_{mac}\}$.

We end Section VII by relaxing the association constraint and considering a stronger attacker who could successfully link [7] randomized addresses to a particular user. We utilize all eight considered features showcased in Table III in that case.

VI. Methodology

In this section, we detail our machine learning-based approach for inferring the device's state from passively captured Wi-Fi probe-requests. Along with the active usage of the device from its idle state but also try to infer other parameters like the phone battery, Wi-Fi ON/OFF option etc.

A. Data processing and training

We split the probe-requests from each device seen in the datasets into individual bursts by separating the frame sequences that have inter-frame duration longer than 1 second. We only consider bursts that have multiple captured frames. The MAC address of a device remains the same during a burst and acts as an identifier for the sending smartphone. This allows us to calculate the burst and device-based metrics or features needed to train the model. The model takes individual bursts (b_n) for training as well as the input for prediction.

To obtain a robust model against the unseen data, bursts in the dataset are split into two subsets: 75% is used for training, and the remaining 25% is only exploited during the testing phase. We train the model on individual datasets and utilize the trained model to predict smartphones' *states* on respective datasets. This is because both datasets show different numbers of state labels (6 and 4 respectively). For binary active usage prediction, we investigated accuracy obtained by the model trained on `Pintor` dataset while testing on `Furious` dataset (and vice versa). In this case, too, we observe a similar performance as expected due to the generic state-specific behaviour (cf. Figure 2) of probe-requests. We use the `scikit-learn` [20] Python library [21], which provides the implementation of the Random forest model.

B. Performance evaluation

To evaluate the classification efficiency, we use three metrics:

1) **Accuracy (Acc):** The first metric is the accuracy Acc, which is the fraction of correct predictions in the test dataset.
2) **Confusion matrix (C):** We compute C to further delve into the accuracy of our *state* classification. The confusion matrix is such that C_{ij} is equal to the fraction of total observations known to be in state i and predicted to be in state j.

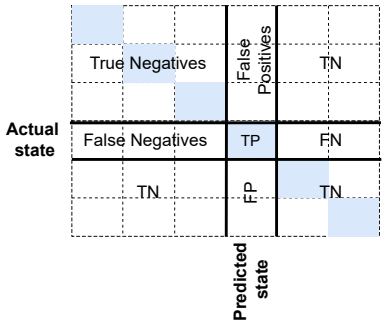


Fig. 3: Inferences from Confusion matrix

Anything outside of the leading diagonal is a misclassification. The fraction of True negatives (TN), False negatives (FN), True positives (TP), and False positives (FP) can be inferred from C as illustrated in Figure 3.

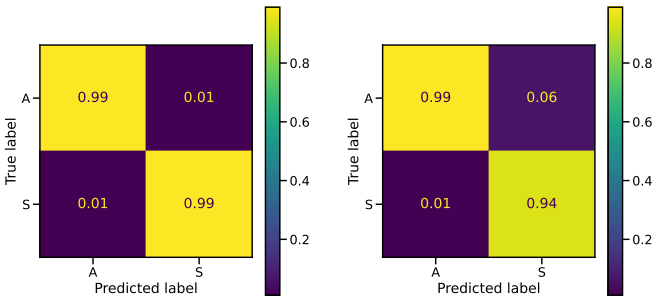
3) **Matthew correlation coefficient (MCC):** Finally, we complement the above metrics with MCC, as it is considered a more effective metric to other classical methods like F1-score or receiver operating characteristic curve (ROC AUC) [22], [23]. Specifically, for our primary investigation of classifying binary (A/S) states, MCC does consider the imbalanced classes and is the standard metric [23].
MCC lies in the interval [-1; 1] and a high value (MCC close to 1) is achieved only if the classifier scored a high value for all the four basic rates of the confusion matrix: sensitivity, specificity, precision, and negative predictive value [23].

VII. Results

In this section, we first test our model's effectiveness in predicting active usage. Next, we infer *sub-states* as detailed in Table II, investigate the importance of each feature for accurate state prediction, and examine advanced threats from attackers enabling MAC association.

Inferring active usage: We report an overall accuracy of 0.994 and 0.981 in inferring the active usage of the smartphone denoted by state A, for `Pintor` and `Furious` dataset respectively.

We observe the confusion matrix in Figure 4. The fraction of TP is very high for predicting both states A and S when considering `Pintor` dataset. It stands at 0.99 for both states. In `Furious` dataset too, the fraction of TP is 0.99 and 0.94 for A and S states. For `Furious` dataset that has a higher number of devices (205), it is slightly more difficult to predict



(a) `Pintor` dataset (b) `Furious` dataset

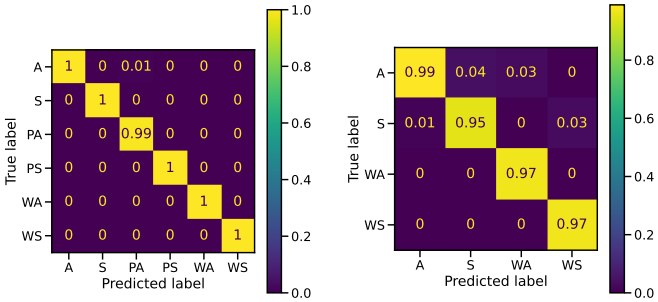
Fig. 4: Predicting active usage

the static screen state. The FP is just 0.01 for `Pintor` and `Furious` datasets when predicting active smartphone usage.

Finally, looking at MCC, we obtain very high values of 0.989 and 0.942 for `Pintor` and `Furious` datasets respectively. This attests to the reliability and the correctness of the model's state prediction.

Inferring sub-states: In `Furious` dataset, we classify into 4 states (A, S, WA, WS). On the other hand, we try to infer 6 states (A, S, PA, PS, WA, WS) in `Pintor` dataset, owing to the availability of more *ground-truth* labels (cf. Table II).

We report an overall accuracy of 0.983 and 0.995 for `Furious` and `Pintor` datasets respectively. Despite challenging `Pintor` dataset to predict deeper sub-states, we achieve high accuracy.



(a) `Pintor` dataset (b) `Furious` dataset

Fig. 5: Predicting sub-states: phone battery state, Wi-Fi option ON/OFF

In Figure 5, we observe the confusion matrix. The fraction of TP is very high for predicting various sub-states in both `Furious` and `Pintor` datasets. It is greater than 0.95 for all sub-states: (A, S, PA, PS, WA, WS). FPs also remain low for both datasets.

Observing the MCC, we obtain high values of 0.948 and 0.994 for `Furious` and `Pintor` datasets respectively. It shows that even when breaking down the active and static smartphone screen usage, the model still predicts substates.

Importance of features: For assessing the importance of the chosen features, we extract the weights leveraged by the Random Forest classifier in Figure 6. We find that number of bursts, MAC sojourn time, and, non-empty IE field length, are very essential with a weight of ≥ 0.19. Burst duration and bursts' mean sequence number gap also play a part, although smaller, in classification.

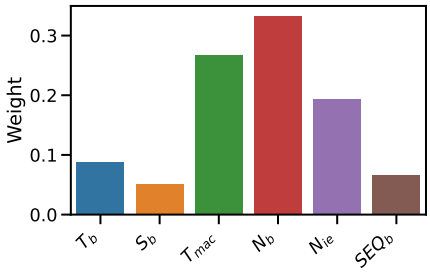
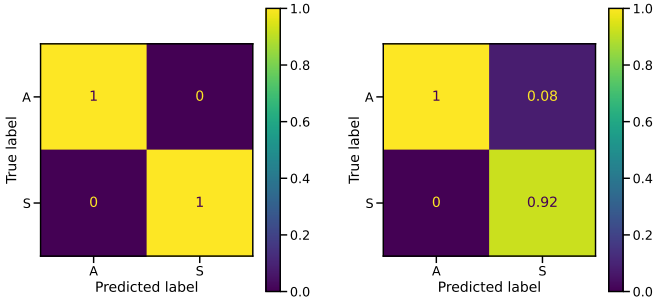


Fig. 6: Feature weights in the Random Forest model

Advanced threats: As we already mentioned in our threat model (cf. Section IV-B), we assume that the adversary has no knowledge of performing MAC association. Now, we relax the constraint and use all the features stated in Table III for training the model. This makes the attacker more potent as now it can track the states of each device instead of just particular MAC addresses.

In this case, we report a slightly increased overall accuracy of 0.998 in inferring the active usage for `Pintor` dataset.

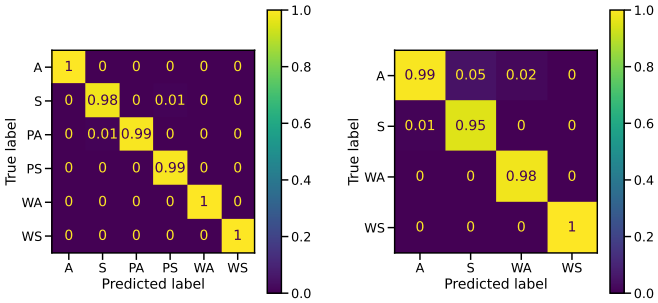


(a) `Pintor` dataset (b) `Furious` dataset

Fig. 7: Active usage inference (with MAC association)

We observe the confusion matrix for binary state classification in Figure 7. The fraction of TP is very high for predicting both states A and S in `Pintor` dataset. The value observed is 1 across states. In `Furious` dataset, the fraction of TP is 1 and 0.92 for A and S states. The FP is 0 for both `Pintor` and `Furious` datasets when predicting the active/static smartphone usage. Observing the MCC, we obtain slightly higher values of 0.997 for `Pintor` datasets, while `Furious` gives the same value as without de-randomization.

When predicting sub-states with this new model with MAC de-randomization, accuracy practically remains the same in



(a) `Pintor` dataset (b) `Furious` dataset

Fig. 8: Sub-states inference (with MAC association)

inferring the active usage of the smartphone, for `Furious` and `Pintor` datasets respectively. The confusion matrix for sub-state classification is shown in Figure 8. TP fraction remains high for predicting various sub-states when considering both `Furious` and `Pintor` datasets. For sub-states WA and WS in `Furious` dataset, we could notice a slight improvement in accuracy, with the fraction of FPs also remaining very low for both datasets.

The findings suggest stronger inferences with an effective MAC de-randomization. We chose to classify states using a single burst. With knowledge of the smartphone's location and context, an adversary could achieve greater accuracy.

VIII. Countermeasures

Currently, devices send probe-requests deferentially across various states, which increases the chances of being fingerprinted (cf. Section V). In the future, device manufacturers need to make the probing behaviour judicious and more uniform across various states of smartphones. The API calls by operating systems for network selection should be made independent of the current state of the device and its selection of user-defined features like Wi-Fi/Power-saving option (unlike in `Android` [24], for instance).

Sequence numbers should be randomized at the start of each burst (present in iOS [25]). Finally, timing and frame content-related parameters of probe-request bursts should remain consistent throughout the device's lifetime regardless of usage.

IX. Conclusion

We demonstrate the successful inference of smartphone states by analyzing Wi-Fi probe-requests. Our approach initially identifies active/static usage and can extend to detecting Wi-Fi/power-saving modes. Critical features are extracted from probe-request bursts, which vary by device state. Our Random Forest model classifies these states with $\geq 98\%$ accuracy using extensive datasets of over 200 smartphones across various operating systems. We also discuss advanced threats from MAC association and suggest potential countermeasures for device manufacturers. Future work will include more states and state-specific features to enhance inference accuracy and effectiveness.

REFERENCES

[1] Z. Koh, Y. Zhou, B. P. L. Lau, C. Yuen, B. Tuncer, and K. H. Chong, "Multiple-perspective clustering of passive Wi-Fi sensing trajectory data," *IEEE Transactions on Big Data*, pp. 1–1, 2020.

[2] B. Huang, G. Mao, Y. Qin, and Y. Wei, "Pedestrian flow estimation through passive WiFi sensing," *IEEE Transactions on Mobile Computing*, vol. 20, no. 4, pp. 1529–1542, 2021.

[3] A. K. Mishra, A. Carneiro Viana, N. Achir, and C. Palamidessi, "Public wireless packets anonymously hurt you," in *2021 IEEE 46th Conference on Local Computer Networks (LCN)*, pp. 649–652, 2021.

[4] B. Greenstein, R. Gummadi, J. Pang, M. Y. Chen, T. Kohno, S. Seshan, and D. Wetherall, "Can Ferris Bueller Still Have His Day Off? Protecting Privacy in the Wireless Era.," in *HotOS*, 2007.

[5] M. Cunche, M.-A. Kaafar, and R. Boreli, "Linking wireless devices using information contained in Wi-Fi probe requests," *Pervasive and Mobile Computing*, vol. 11, pp. 56–69, Apr. 2014.

[6] M. V. Barbera, A. Epasto, A. Mei, V. C. Perta, and J. Stefa, "Signals from the crowd: uncovering social relationships through smartphone probes," in *ACM IMC*, pp. 265–276, 2013.

[7] M. Vanhoef, C. Matte, M. Cunche, L. S. Cardoso, and F. Piessens, "Why mac address randomization is not enough: An analysis of wi-fi network discovery mechanisms," in *ACM on Asia CCS*, ASIA CCS '16, (New York, NY, USA), p. 413–424, Association for Computing Machinery, 2016.

[8] "Ieee standard for information technology–telecommunications and information exchange between systems - local and metropolitan area networks–specific requirements - part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications - redline," *IEEE Std 802.11-2020 (Revision of IEEE Std 802.11-2016) - Redline*, 2021.

[9] V. Srinivasan, J. Stankovic, and K. Whitehouse, "Protecting your daily in-home activity information from a wireless snooping attack," in *UBICOMP*, pp. 202–211, 2008.

[10] A. Acar, H. Fereidooni, T. Abera, A. K. Sikder, M. Miettinen, H. Aksu, M. Conti, A.-R. Sadeghi, and S. Uluagac, "Peek-a-boo: I see your smart home activities, even encrypted!," in *ACM WiSec'20*.

[11] Y. Meng, J. Li, H. Zhu, X. Liang, Y. Liu, and N. Ruan, "Revealing your mobile password via wifi signals: Attacks and countermeasures," *IEEE Transactions on Mobile Computing*, 2019.

[12] A. K. Das, P. H. Pathak, C.-N. Chuah, and P. Mohapatra, "Uncovering privacy leakage in ble network traffic of wearable fitness trackers," in *Proceedings of the 17th international workshop on mobile computing systems and applications*, pp. 99–104, 2016.

[13] Y. Gu, F. Ren, and J. Li, "Paws: Passive human activity recognition based on wifi ambient signals," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 796–805, 2015.

[14] H. Liu, A. Alali, M. Ibrahim, B. B. Cao, N. Meegan, H. Li, M. Gruteser, S. Jain, K. Dana, A. Ashok, *et al.*, "Vi-fi: Associating moving subjects across vision and wireless sensors," in *2022 21st ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, pp. 208–219, IEEE, 2022.

[15] G. Celosia and M. Cunche, "Detecting smartphone state changes through a bluetooth based timing attack," WiSec '18, (New York, NY, USA), p. 154–159, Association for Computing Machinery, 2018.

[16] S. Jamil, S. Khan, A. Basalamah, and A. Lbath, "Classifying smartphone screen on/off state based on wifi probe patterns," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*, pp. 301–304, 2016.

[17] L. Pintor and L. Atzori, "A dataset of labelled device wi-fi probe requests for mac address de-randomization," *Computer Networks*, vol. 205, p. 108783, 2022.

[18] E. Fenske, D. Brown, J. Martin, T. Mayberry, P. Ryan, and E. C. Rye, "Three years later: A study of mac address randomization in mobile devices and when it succeeds.," *Proc. Priv. Enhancing Technol.*, 2021.

[19] A. Chaudhary, S. Kolhe, and R. Kamal, "An improved random forest classifier for multi-class classification," *Information Processing in Agriculture*, vol. 3, no. 4, pp. 215–222, 2016.

[20] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*.

[21] S. learn (version 1.3.2), "Scikit-learn implementaion." https://scikit-learn.org/stable/index.html.

[22] D. Chicco and G. Jurman, "The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation," *BMC genomics*, vol. 21, no. 1, pp. 1–13, 2020.

[23] D. Chicco and G. Jurman, "The matthews correlation coefficient (mcc) should replace the roc auc as the standard metric for assessing binary classification," *BioData Mining*, vol. 16, no. 1, pp. 1–23, 2023.

[24] Android, "Android mac randomization." https://source.android.com/docs/core/connect/Wi-Fi-network-selection.

[25] Apple, "ios mac randomization." https://support.apple.com/en-in/guide/security/secb9cb3140c/web.