Personally Identifiable Info.	# Leaking Apps	# Users
IMEI	5	4
Android Device ID	4	6
Phone Number	1	1
Email address	1	1
Location	1	2

Table 2: Number of applications that leaked personally identifiable information in the collected data.

the ability to check whether any of the installed applications are sending her PII out to the Internet. With AntMonitor, checking whether PII is leaked can happen entirely at the client, without the need to either send the PII to any remote server [7], or have a customized OS or rooted device [20].

In our dataset, we tested for leaks of the following PII groups: (i) IMEI, which uniquely identifies a device within a mobile network, and Android Device ID, which is an identification code associated with a device; and (ii) phone number, email address and location, which uniquely associate with users. Table 2 presents the results of our analysis. We observe that 44% and 66% of the users have applications that leak their IMEI and Android Device ID. The majority of leaks are from a small number of applications, and these applications are seemingly harmless, *e.g.*, iWindsurf, a weather application, and Radio FM, an audio streaming application. PII in the second group is rarely leaked, presumably because it is considered sensitive by software developers. However, there are a few exceptions, such as Evite, which leaks email accounts in plain text.

5.3 Application Classification

We use supervised learning methods to build a multi-class model that classifies network flows to specific applications. Building profiles of applications using network-level features can be useful in several ways, *e.g.*, it can assist ISPs who may want to understand and optimize the type of traffic passing through their networks, and it can also facilitate anomaly detection on the device. Our novelty compared to previous approaches [21, 22] is that we use only layer-3 and 4 packet header features, and we have realistic network traffic traces that reflect normal user behavior.

Methodology. We calculated a set of 84 network-level features on both the upstream and downstream flows. These are widely used features from network traffic classification literature and can be partitioned into the following groups: packet length statistics, payload length statistics, inter-arrival time statistics, burstiness, overall flow statistics, and TCP flags. We then performed classification on 70 applications that have at least 30 flows. We used 10-fold cross-validation to avoid overfitting and kept the same proportions of the applications in the testing and training.

Results. We are able to classify a flow to a specific application with F1-score of 70.1% using a Linear SVM. To put this result into context, Meddle [7] reports a 64.1% precision score in classifying flows for the 92 most popular Android applications by using payload features: Host and User-Agent. AppPrint [22] reports that in a large dataset of millions of applications, only 1% of the flows can be classified by using features from HTTP headers. Interestingly, just by using off-the-shelf learning tools, we are already able to classify applications better than state-of-the-art approaches. This is thanks to the fine-grained information available at training: we can associate a packet with the application that generated it.

6. CONCLUSION

In this work, we present AntMonitor – a system for crowdsourcing large-scale, yet fine-grained network measurements from mobile devices. AntMonitor is designed from the bottom up to bring benefits to both the crowdsourcing system, network researchers and the participating users. AntMonitor is designed to scale and is carefully implemented to achieve high network performance with low CPU and battery overhead. Our pilot deployment of AntMonitor shows that it can greatly assist network research activities, including but not limited to, network measurements, detection of malicious behaviors, and traffic classification. Additional materials and a video demo of AntMonitor can be found on our project website [23].

7. REFERENCES

[1] Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update 2014-2019. http://goo.gl/Zu8f2r.

[2] Q. Xu, J. Erman, A. Gerber, Z. Mao, J. Pang, and S. Venkataraman. Identifying Diverse Usage Behaviors of Smartphone Apps. *IMC'11*.

[3] X. Chen, R. Jin, K. Suh, B. Wang, and W. Wei. Network Performance of Smart Mobile Handhelds in a University Campus WiFi Network. *IMC'12*.

[4] H. Falaki, D. Lymberopoulos, R. Mahajan, S. Kandula, and D. Estrin. A First Look at Traffic on Smartphones. *IMC'10*.

[5] N. Vallina-Rodriguez, A. Auçinas, M. Almeida, Y. Grunenberger, K. Papagiannaki, and J. Crowcroft. RILAnalyzer: A Comprehensive 3G Monitor on Your Phone. *IMC'13*.

[6] Netalyzr. https://goo.gl/i7HyLU.

[7] A. Rao, A. M. Kakhki, A. Razaghpanah, A. Tang, S. Wang, J. Sherry, P. Gill, A. Krishnamurthy, A. Legout, A. Mislove, and D. Choffnes. Using the Middle to Meddle with Mobile. Technical report, Northeastern University, Dec. 2013.

[8] Android VpnService. http://goo.gl/kV7ZZL, 2014.

[9] Android Versions. developer.android.com/about/dashboards.

[10] PCAPNG File Format. http://goo.gl/y89d9U.

[11] tPacket. www.taosoftware.co.jp/en/android/packetcapture.

[12] PhoneLab, University at Buffalo. https://www.phone-lab.org/.

[13] J. Sommers and P. Barford. Cell vs. WiFi: On the Performance of Metro Area Mobile Connections. *IMC'12*.

[14] X. Wei, L. Gomez, I. Neamtiu, and M. Faloutsos. ProfileDroid: Multi-layer Profiling of Android Applications. *MobiCom'12*.

[15] Private Internet Access Privacy Policy. http://goo.gl/Yt8jNx.

[16] strongSwan VPN Client. https://goo.gl/okVQYL.

[17] Netty. http://netty.io.

[18] MultiMedia. Smartphones: So Many Apps, So Much Time, 2014.

[19] FRep - Finger Replayer. https://goo.gl/Qtfcva.

[20] W. Enck, P. Gilbert, B. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth. Taintdroid: an information-flow tracking system for realtime privacy monitoring on smartphones. *OSDI'10*.

[21] S. Dai, A. Tongaonkar, X. Wang, A. Nucci, and D. Song. Networkprofiler: Towards Automatic Fingerprinting of Android Apps. *INFOCOM'13*.

[22] S. Miskovic, G. M. Lee, Y. Liao, and M. Baldi. AppPrint: Automatic Fingerprinting of Mobile Applications in Network Traffic. *PAM '15*.

[23] AntMonitor Project Website. http://antmonitor.calit2.uci.edu.