Assessing the cyber-trustworthiness of human-as-a-sensor reports from mobile devices

Syed Sadiqur Rahman*, Ryan Heartfield, William Oliff, George Loukas, Avgoustinos Filippoupolitis
Department of Computing and Information Systems
University of Greenwich, UK
*Email: S.S.Rahman@greenwich.ac.uk

***Abstract*—The Human-as-a-Sensor (HaaS) paradigm, where it is human users rather than automated sensor systems that detect and report events or incidents has gained considerable traction over the last decades, especially as Internet-connected smartphones have helped develop an information sharing culture in society. In the law enforcement and civil protection space, HaaS is typically used to harvest information that enhances situational awareness regarding physical hazards, crimes and evolving emergencies. The trustworthiness of this information is typically studied in relation to the trustworthiness of the human sensors. However, malicious modification, prevention or delay of reports can also be the result of cyber or cyber-physical security breaches affecting the mobile devices and network infrastructure used to deliver HaaS reports. Examples of these can be denial of service attacks, where the timely delivery of reports is important, and location spoofing attacks, where the accuracy of the location of an incident is important. The aim of this paper is to introduce this cyber-trustworthiness aspect in HaaS and propose a mechanism for scoring reports in terms of their cyber-trustworthiness based on features of the mobile device that are monitored in real-time. Our initial results show that this is a promising line of work that can enhance the reliability of HaaS.**

Keywords*—*Human-as-a-Sensor, information trustworthiness, cyber security, provenance of information, quality of information, mobile security, location spoofing, smartphone security

I. Introduction

With almost 5 billion mobile phone users [1], 2 billion of which using increasingly sensor-rich smartphones [2], the amount of data generated from mobile devices is greater than ever before. Every time people notice something unusual or noteworthy, they share it with others in social media or using specialised apps. This sharing culture has created an opportunity for harvesting or generating knowledge from the members of the public to facilitate crisis and emergency response. Reliable and trustworthy information received in this manner can help improve emergency responders' and Law Enforcement Agencies' (LEA) situational awareness and ability to detect and respond to evolving incidents. We refer to this as the Human-as-a-Sensor (HaaS) paradigm for situational awareness. A key such initiative is the TRILLION[1] project [3], which leverages the public knowledge and technological infrastructure, including the use of smartphones, mobile apps, wearables [4] and social media to improve community policing. TRILLION's use of citizen reporting of crimes is the case study that we consider in this work.

[1]**TR**usted, c**I**tizen - **L**EA co**L**laborat**I**on over s**O**cial **N**etworks, http: trillion-project.eng.it

TRILLIONs goal to reduce crime by enhancing community policing through improved LEA-citizen communication and cooperation is expected to make the platform an attractive target for cyber criminals. Beyond confidentiality and privacy, attacks against TRILLION may aim to affect availability preventing LEAs and citizens from accessing it when needed, and integrity, manipulating information, such as the location as recorded or as reported by citizens' mobile devices. An incident report delivered from a malware-infected mobile device should ideally be handled as an untrusted report regardless of the trustworthiness of the user, but there is no practical way for TRILLION to determine this in real-time and remotely. These technical security challenges are amplified by the time criticality of many of the incidents expected to be handled in TRILLION. Beyond strong authentication and encryption, a novel aim in TRILLION is to include the capability of scoring individual crime reports based on the cyber trustworthiness of the platform where they originated from. The focus here is on crime reports (and generally HaaS reports) originating specifically from mobile devices. Our experiments have been conducted on Android mobile devices.

II. Related Work

From a system standpoint, humans are autonomous, multi-sensory systems with the ability to produce inferential output data based on multiple experiential and environmental input data. Within human society, therefore, this ability means that humans (as physical sensors) are often best placed to provide information in various contexts where technical systems alone are not adequate. The concept of Human-as-a-Sensor (HaaS) has been the subject of numerous research activities and services which have aimed to improve situational awareness across a range of contexts such as emergency response in the wake of natural disasters, crisis management, event/incident detection and monitoring or management [5]–[9]. Other examples include the use of HaaS data for neighbourhood watch schemes [10], predicting road traffic information [11] and even defending against security threats in social media [12].

However, whilst HaaS is a concept that promises to help improve a number of real world problems, one of the biggest challenges associated with it is the ability to determine the credibility of the data that is received, as the information is more often than not generated by unknown sources which are untrusted. Therefore, accurately assessing the level of trustworthiness of the information that comes from such sources becomes an important research challenge. Early work in this space has sought to assess the trustworthiness of HaaS data

using the provenance of information [13], [14] and quality of information [15], by analysing people's behaviour and activities on social networks [16], evaluating the performance of human as sensors of social media threats [12], reputation of the data source [17] and so on. More recently, a study by Heartfield et al. [18] evaluated the feasibility of predicting HaaS trustworthiness and efficacy for semantic attack reports. The researchers employed the concept of human-as-a-sensor to detect semantic social engineering attacks [19], where it was shown that the reliability (and therefore credibility) of users' attack reports can be practically predicted by measuring characteristics about their activity profiles. In [14] and [13], authors have proposed to construct one or more world views using the information harvested from human sensors and utilise some of the provenance related factors such as freshness and timeliness of information, proximity and reputation of the reporting users to calculate a trustworthiness score for each of the world views. However, little attention has been paid to measuring the integrity of platforms in which HaaS reporting is made and how this affects a report's credibility, although researchers have merely touched upon the infrastructure integrity in [13]. Considering the integrity of platforms and its possible effect on a report's credibility becomes paramount when crime or incident reports are sent to emergency services which would lead to a physical response.

In this work, we explore the idea of identifying whether a mobile device's "cyber trustworthiness" state can be evaluated accurately and in real-time, so as to serve as a factor of the overall reliability of a HaaS report.

III. Modelling the cyber trustworthiness of mobile HaaS reports

Since HaaS reports are expected to come mostly from sensor-rich smartphones (and 9 in every 10 smartphones are reported to use the Android operating system [20]), our focus in this work is to assess the device cyber security posture of a typical modern Android device that would be used in a HaaS system.

We have focused on measuring an initial set of commonly available device features available to any Android device (or more generally any modern smartphone), to test whether standard device states are sufficiently informative for detecting whether a device is trustworthy or not for sending HaaS reports. By utilising common device features, we enable the measurement of device cyber trustworthiness in a generalised fashion; as experiment results would then be applicable across a wide range of mobile devices. This is a practical approach in early experiments to identify whether such shared features are sufficient for accurately predicting trustworthiness across different threats, or whether for future experiments more complex, device specific features are needed.

This work's primary contributions are:

- Identification of mobile device characteristics that are indicators of cyber trustworthiness of HaaS reports.
- An initial mechanism for scoring cyber trustworthiness based on the features identified
- An experimental evaluation of the mechanism in the presence of two cyber attacks (denial of service and location spoofing)

A. Cyber Attacks against mobiles in the context of HaaS crime reporting

We investigate the ability to detect three viable attacks that would affect crime reporting from a mobile device. Specifically, we focus on the ability to identify network-based DoS threats against the availability of mobile reports, location spoofing and GPS signal spoofing attacks which would attempt to deceive and influence law enforcement.

1) Network Denial of Service: Depending on the platform type used (e.g., social media, instant messaging, custom application), network-based denial of service attacks against mobile devices can result in crime report messages being queued or dropped completely when network connectivity is lost or saturated. For the former, delayed messages might have a direct effect on the integrity of reports received at a much later date from when the related incident actually occurred. In this case, not only are law enforcement agents potentially delayed in responding to on-going crimes, but they may disregard delayed reports if they contradict older messages or de-prioritise them over newer reports which indicate immediate threats. For reports that are dropped, crimes would simply go unreported. Delaying or blocking HaaS reports via DoS is a highly attractive prospect to organised criminals aiming to avoid detection (for example when planning to commit a robbery). We have conducted a network-based denial of service attack against an Android device 802.11 Wi-Fi interface by flooding its IP address with ICMP packets to reduce the devices ability to send reports. The DoS attack was carried out both when the device was in use and not in use by a user. Within the experiment environment the attack was launched in an isolated private Wi-Fi enabled network, with the android device and attacker machines connected to the same access point. The attacker machines were then configured to flood ICMP packets using the tool *hping* over the network to Android's IP address in order to overload the Wi-Fi interface.

2) Location Spoofing: Spoofing attacks deceive targets by obfuscating or imitating data to elicit a response from a target. In the context of crime reporting in physical space, by spoofing a device's location, would-be attackers are able to send false reports that would appear to originate from any specific physical location of their choice. Through location spoofing, attackers may also be able to make a genuine report originating from a nearby location appear form a different location which is far and away from the actual incident location. This information poisoning also allows sending reports as a decoy to physically divert law enforcement from real crimes happening elsewhere [30]. We have conducted random spoofing of an Android device's location by fabricating its Global Positioning System (GPS) information in order to misrepresent its physical location, both when the device was in use and not in use by a user. In the experiment, a mock location provider was built into the experiment app to send fabricated location data to the location manager of the Android device. Any application using the location service of the device would then receive the fabricated location data from the mock provider. Although special permissions had to be granted to allow mock locations to be established, it is possible for a jail-broken (i.e., rooted) device to set fabricated location data without the need of any special permissions and there are examples of malware that perform the rooting of android devices surreptitiously [31].

Feature	Description
CPU usage	CPU utilisation indicates device load and level of activity on the device and is expected to be low (if not completely idle) when the device is not actively used. Arbitrary or sustained increases in utilisation (e.g., when no human user is present) could indicate the presence of malware operating on the device, or processing load from a denial of service threat.
RAM usage	Device memory usage increases as the number of running processes (e.g., apps) increases and (in most cases) memory usage decreases when the device is not used or an application is closed. RAM utilisation of a mobile device may indicate anomalous activity, especially if there is no active user or open user applications running in memory. Other researchers have monitored deviation from normality in terms of resource utilisation (e.g. processor and memory usage, I/O operations, storage access, etc.) by different apps or processes to detect cyber threats [21], [22]. In [21], a ransomware prevention technique has been proposed for the Android platform, which continuously monitors the CPU usage, memory usage, number of input/output operations and storage accesses, etc., using statistical methods to differentiate between normal and abnormal uses based on these features.
Network traffic	Volume and ratio of network traffic are consistently used as key indicators of anomalous behaviour in intrusion detection systems and denial of service defence [23]. Since smartphones rely heavily on network connectivity, measuring a smartphone's traffic profile becomes crucial to protect against integrity and availability threats. For example, an abnormal increase in the amount of incoming, unsolicited network traffic might indicate the manifestation of network flooding attack. Here, we also include network traffic received (R_x) and sent (T_x) as well as the ratio of network traffic received over the total network traffic, which is $\frac{R_x}{R_x+T_x}$, as an indication of network traffic proportionality. In related work researchers have employed ratio of transmitted and received data for detecting the presence of malware on smartphones [24]–[26].
GPS SNR	This is a binary value denoting whether the Global Positioning System (GPS) Signal-to-Noise Ratio (SNR) has changed since the last measurement. Due to the distance they cover (from satellites to Earth), mobility and environmental conditions, legitimate GPS signals received by smartphones vary considerably in time in terms of their signal-to-noise ratio. This is a simple binary metric capturing this aspect. Spoofed and jamming GPS signals are of terrestrial origin and as such may exhibit different SNR behaviour.
Location accuracy	Android devices can collect the geographical location of a device from their GPS, WiFi, Bluetooth or mobile network. Android location services is often used to provide location estimates and uses a confidence factor referred to as 'Accuracy'. For example, if device location is reported with an accuracy of 20metres, then there is a 68% probability that the true location of the device is inside the area covered, within a radius of 20m central from the reported latitude and longitude [27]. We collect device location and compare its corresponding accuracy to see if there is any change from normality. Previous research has used location to identify anomalies on mobile devices [28], [29]. Here, we define a change of location accuracy as a binary value denoting whether it has changed since last measurement. The logic here is the same as with the equivalent metric for GPS SNR.
Power consumption	Power consumption of smartphones follows a pattern and this pattern changes according to the users location as the usage level of the mobile phones greatly varies based on their location. Research [28], [29] suggests that it is possible to detect malware infection of smartphones by analysing their power consumption patterns that are, in turn, location dependent. There are certain places for each mobile user where the power consumption is low and these are the places wherein abnormality caused by a malware is more likely to be detected [29]. In [29] researchers have used the cellular network and wifi connections for determining a users location.

TABLE I: Mobile device cyber trustworthy features

3) GPS Spoofing: Here, the location is spoofed not programmatically, but physically by generating fake GPS signals using a Software Defined Radio (SDR) transmitter (e.g. HackRF One). An SDR is a radio communication system where some of its components are implemented using software on a personal computer that are typically implemented in dedicated hardware components. HackRF One is an open source hardware peripheral which can be connected to a computer using the USB port. It is a half-duplex transceiver which can transmit or receive radio signals with the frequency ranging from 1 MHz to 6 GHz [32].

B. Experimental environment

For our experiments, we have produced the three different attack scenarios described above. Figure 1 shows the attack environment of the DoS attack on the wireless network, where we have used a wireless router that provided a wifi network from which a mobile user was able to send and receive information using messaging apps (e.g. WhatsApp) and carry out other usual activities such as installing or updating an app. The mobile device was running Android Version: 6.0.1 (Marshmallow) and Java Version: 8 (1.8 JRE). The mobile phone is configured to allow the location services of the phone to accept mock locations provided by a location spoofing software. The location services of an Android device is responsible for updating the geographical location of the device and allow other applications to obtain that geographical location. Generally, the location services manager gets the location of the device from its GPS, WiFi, Bluetooth or mobile network. However, when the location services manager gets a mock location (not its real location) from another software (a location spoofing software), all other services on the phone think that the phone is really in that mock location, which is not true. Figure 2 shows how to change the device settings on

Fig. 1: Attack environment for network denial of service consists of an Android phone, a laptop and a wireless router

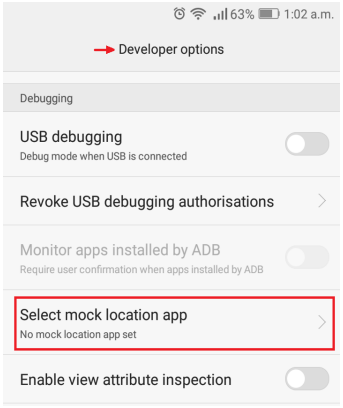


Fig. 2: The Android device setting which enables a location spoofing app to provide a mock (untrue) location of the device

an Android device so that it allows an application to provide a mock location of the device which is not its real location. This represents the case wherein the device has been infected with malware, which modifies programmatically the location reported by the device.

Figure 3 shows the attack environment for hardware assisted location spoofing using Software Defined Radio (SDR) transmitter. In this scenario, the SDR transmitter transmits a radio signal containing a specific geographic coordinate which is not the real coordinates of that location and thus, influences the location services of the Android device. As a result, the geographical location calculated by the GPS module of the device becomes inaccurate.

1) Device Monitoring Modules: In Figure 5, we provide a schematic view of the Android monitoring system, with a breakdown of the individual modules that are used to collect

Fig. 3: Attack environment of hardware-assisted location spoofing consists of an Android phone, a laptop and a software defined radio (SDR) transmitter (HackRF)

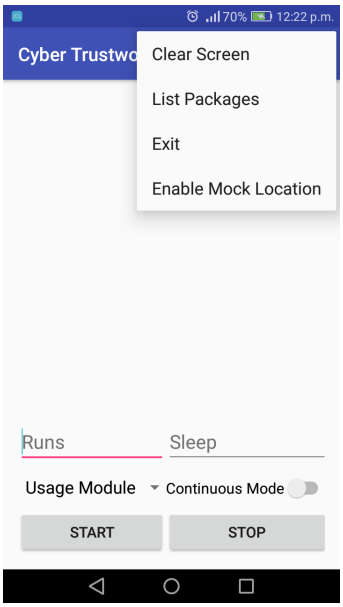


Fig. 4: Sreenshot of the Monitoring App installed on a User Device

device features for calculating a device's cyber-trustworthiness score.

The monitoring system has been designed with modularity in mind, where the aim is to provide a platform for straightforward integration of new (or removal of old) device sensors, interfaces and data without requiring significant redevelopment. Each monitoring module runs in the background as a separate instance with its own task thread. As a result, the main program routine remains free to perform other tasks within the application. The monitoring modules can run in two modes: continuous or limited. This gives more flexibility to how data can be collected. In continuous mode, the module will collect samples forever unless interrupted by another program flow. Whereas, a module in limited mode will only collect the specified number of samples. All the modules behave in the same manner, regardless of the data samples they collect. Since each of these modules is a subclass derived from a module superclass, the superclass can handle the core behaviour and provide general implementation for all modules, while the individual monitoring modules in the subclasses provide module specific implementation (e.g. the data to be collected by the module). This also grants the advantage of being able to add new monitoring modules efficiently as new modules can inherit core implementation and behaviour from the superclass. This thereby effectively sets a standard for all monitoring modules and ensures coherence between the different modules.

When a module is started with its specified run mode, the module will first collect any invariable data (referred to as constant samples in the flowchart shown in Figure 5) that remains constant during a specific session or the life time of the device e.g. Wi-Fi security protocol used, size of the RAM, etc. and then, collect variable (changing) samples periodically according to the set time interval. For example, while monitoring the Wi-Fi connectivity, we may first record

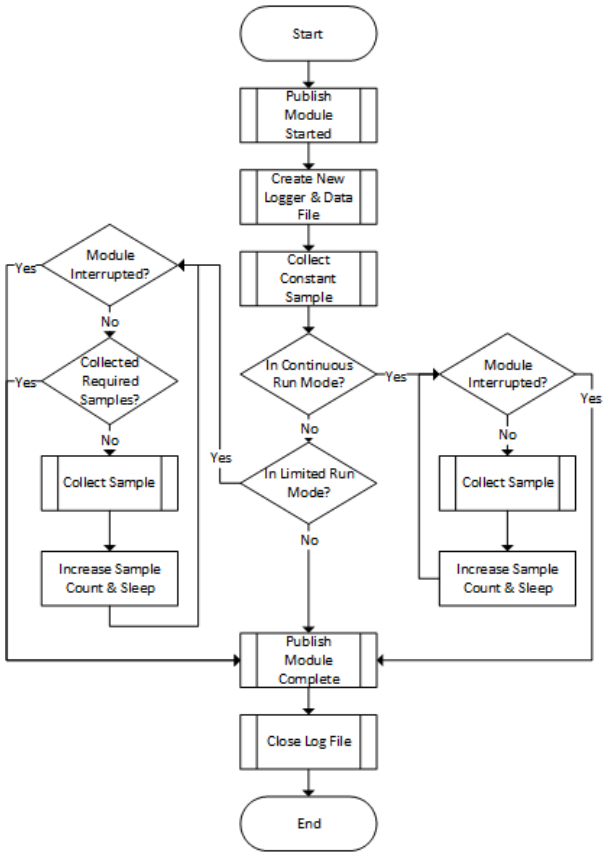


Fig. 5: Monitoring Module Schematic View

the security protocols being used (as invariable data) and then record the total number of Tx (transmitted data) and Rx (received data) bytes (as variable samples) every five seconds. Additionally, each monitoring module instance has its own logger object, which the module uses to log their gathered sample data to various data files within the application.

Lastly, a module listener interface is provided, which allows a subscriber (listener) to receive event based triggers from the modules. Therefore, allowing other program flows to be aware of the current status of the modules and perform tasks based on the module state without having to continually poll the modules. For example, after a module in a limited mode run has collected all required samples and publish completed, another program flow can then begin processing the data gathered by the module. This listener interface effectively allows the application to dynamically react to the current status of each individual monitoring module.

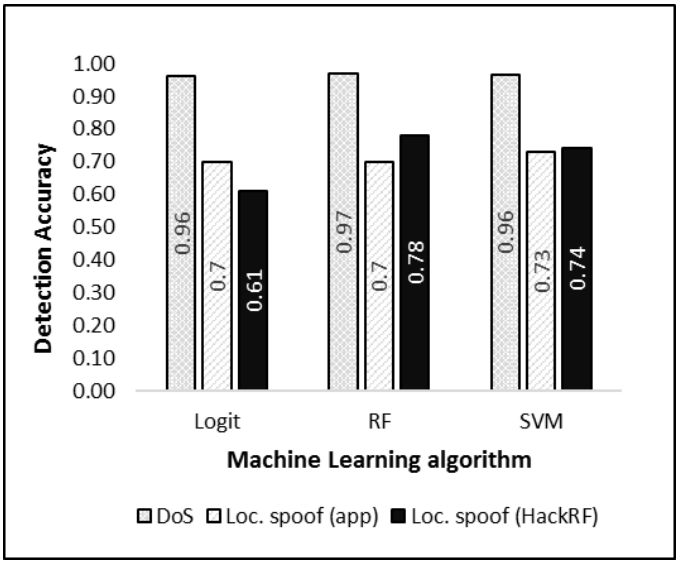
IV. Statistical Analysis and Experimental Results

The prediction of whether a device is trustworthy or not can be viewed as a typical binary classification problem. As our aim is to evaluate whether certain cyber features' temporal states are associated with device trustworthiness, we employ different machine learning techniques to identify an optimum modelling approach. For this, we have selected to use Logistic Regression (LR), Random Forest (RF) and Support Vector Machines (SVM). LR is an obvious choice for modelling linear relationships between a categorical response variable and one or more categorical or continuous predictor variables because it operates as a special case linear model where the respond variable is binomial. RF and SVM, by comparison are more suited for revealing non-linear relationships between predictor variables and a binary response variable. Here, we have used SVM with a radial basis function which is more efficient for identifying non-linear relationships in a dataset [33]. RF is a naturally non-linear machine learning algorithm which models an ensemble of decision trees with each containing re-sampled versions of the original dataset. The approach is effective because it implicitly reduces the variance that is expected in a decision single tree and improving the general accuracy of the model by averaging the standard error of all decision trees. In RF each decision tree employs Boolean logic in a series of decision rules by evaluating a random subset of n features at each decision split to determine which class (e.g., trustworthy/untrustworthy) a feature's value belongs to; which means by default RF does not presume linearity in the modelling process. In classification tasks, the majority vote in each decision tree split is used to determine the class of a feature value.

By training a model on the average output of grouped time-sensitive features we facilitate the capture of temporal states in device modules based on different snapshots in time; which may be missed by modelling stateless, individual sample observations. During the model training phase of the experiments it was identified that the RAM, Tx and Rx feature for location spoofing attacks were causing significant over-fitting of each of the machine learning models, which resulted in over-optimistic prediction results. This occurred because the RAM feature was not sufficiently obfuscated by general user application noise, and therefore clearly revealing when the software location spoofing was occurring. In the case of the network features Tx and Rx, they were required to be turned off when using the hard-assisted location spoofing and therefore were omitted from the modelling. In future experiments modelling with the RAM feature requires the injection of residual application use to add noise to the feature.

The bar graph in Figure 6 and Table II show the accuracy and overall performance of each classifier for a mobile device under DoS and location spoofing attacks, respectively. Unsurprisingly, for the DoS attack all three classifiers were essentially equal in their detection accuracy, which is somewhat expected given the relative simplicity of the attacks behaviour (being a brute force ICMP flood). For all models, detection accuracy was much less accurate in both (types of) location spoofing attacks, but still sufficiently more accurate than the null model for each attack. For instance, for the software and hardware-assisted attacks the null model (e.g., model without features) is able to achieve an accuracy of 59% and 57%, respectively, but the SVM and RF models achieved classification accuracies of 73% and 78% for these respective attacks; proving the viability of the selected features for the detection of cyber trustworthiness in our experiment attack cases. Overall LR was the worst performing of the three classifiers with an accuracy of 70% (the same as RF but less precise with more false positives) for the software spoof and 61% for the hardware-assisted spoof. The result indicates the lesser practicality of using simpler statistical approaches such as LR

Fig. 6: Attack detection accuracy for Logistic Regression, Random Forest and Support Vector Machine classifiers for DoS and Location spoofing attacks



against a range of attacks where threats do not necessarily indicate an obvious linear change in specific device modules. Whilst SVM and RF exhibited similar performances for all attacks, overall RF was the best performing classifier for each attack with the majority of modelling results showing higher precision, lower false negatives (i.e., untrusted device classified as trusted) and false positives (i.e., trusted device classified as untrusted) than both LR and SVM. By exception for software location spoofing the SVM model was 3% more accurate and detected more true positives (e.g., attack states). Nevertheless, for the attacks evaluated and the results presented in our early experiments RF seems to be an accurate classifier for detecting multiple dissimilar threats against a mobile device. We have used the *Caret* package within *R-Studio* to model each of the classifiers.

Model	Attack	ACC	Prec	TPR	TNR	FPR	FNR
LR	DoS	0.96	0.98	0.97	0.94	0.06	0.03
	Loc. spoof (Mock app)	0.70	0.62	**0.72**	0.69	0.38	**0.28**
	Loc. spoof (HackRF)	0.61	0.54	0.62	0.59	0.41	0.38
SVM	DoS	0.96	0.97	0.98	0.92	0.08	0.02
	Loc. spoof (Mock app)	**0.73**	0.67	0.68	**0.77**	0.23	0.32
	Loc. spoof (HackRF)	0.74	0.72	0.66	0.8	0.20	0.34
RF	DoS	**0.97**	**0.98**	**0.99**	**0.94**	**0.06**	**0.02**
	Loc. spoof (Mock app)	0.70	**0.70**	0.47	**0.86**	**0.14**	0.53
	Loc. spoof (HackRF)	**0.78**	**0.80**	**0.66**	**0.87**	**0.13**	**0.34**

TABLE II: Attack modelling performance for Logistic Regression, Random Forest and Support Vector Machine classifiers, all models were statistically significant at the 0.001 level

To better understand how each of the features evaluated influence the classification for the three models tested, we use variable importance, which provides a common measure of how much each model feature influences the accuracy of prediction of the dependent variable. Whilst in general linear models the t statistic is used to calculate variable performance, for Random Forest out-of-bag error is used and for SVM the area under the ROC curve is computed for each feature in the model to calculate variable performance. So, whilst each model's variable important calculation for each of the cyber features are not directly comparable, they provide an indication of which features are the most indicative of a threat against the mobile device; which helps to identify whether there are commonalities between certain features that are useful for measuring cyber trustworthiness.

In Tables III, IV and V, the variable importance for the DoS, software location spoofing and hardware location spoofing attacks features are shown, respectively. Interestingly, RF as the best performing model also produced variable importance values that contextually relate to the expected primary explanatory features of each attack. For example, it was expected in the DoS attack that received traffic (e.g., Rx) would be a primary feature of a network-based DoS which is backed up by Tx variable importance which represents response from the mobile to the ICMP flood. For each of the location spoofing attacks signal to noise ratio (SNR) was important in identifying the threat, whereas for the software spoofing location accuracy also played a role in helping identify the attack. On the contrary, the CPU feature seemed to be more informative for detecting when a hardware-assisted location spoofing attack was occurring. Similar to the RF model, both LR and SVM variable importance reported contextually relevant values for each respective feature, but assigned different values to RF. In practice, the machine learning algorithms can have a symptomatic effect on various feature choices (such as feature weight bias) due to their different algorithmic treatment of feature values and the associated output.

Feature	LR	RF	SVM
RAM	100%	9.18%	0 %
CPU	77.6%	0 %	19.93%
Tx	3.12%	32.10%	98.78%
Rx	38.29%	100%	100 %
Rx/(Rx+Tx)	35.5%	32.10 %	90.62%

TABLE III: Model variable importance for DoS attack

Feature	LR	RF	SVM
SNR	0 %	100%	0%
Loc. Accuracy	77.44%	10.78%	24%
CPU	100%	0 %	100%

TABLE IV: Model variable importance for software-based location spoofing (mock app)

Feature	LR	RF	SVM
SNR	0.7%	35.53%	100%
Loc. Accuracy	0 %	0 %	46.51 %
CPU	100%	100 %	0%

TABLE V: Model variable importance for hardware-assisted software-defined radio location spoofing (HackRF)

A. Scoring Trustworthiness: A probability measure

Binary classifiers are useful for detecting whether a mobile device is under attack or not, but are somewhat less helpful for generating a relative cyber trustworthiness score. Irrespective of a reports context or severity, the profile of the device would be either "trustworthy" or "not trustworthy". Consequently,

Fig. 7: RF DoS (blue dashed curve), mock app GPS spoofing (black curve) and hardware assisted location spoofing - HackRF (orange curve) classification ROC curve with TPR and FPR threshold

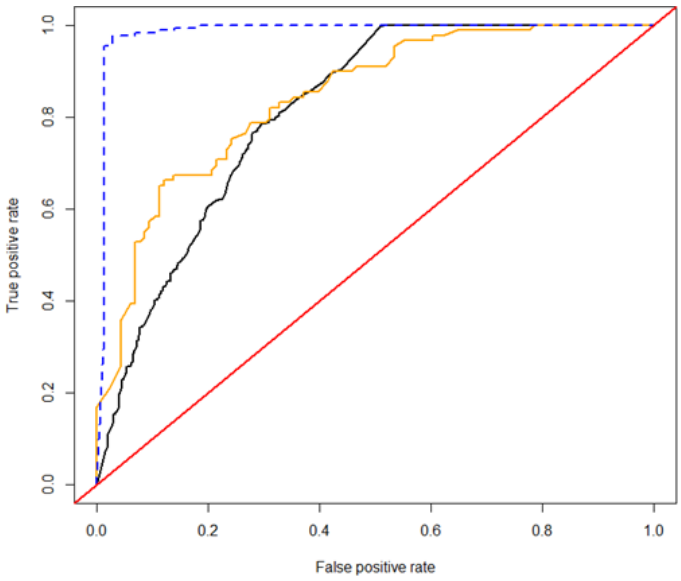
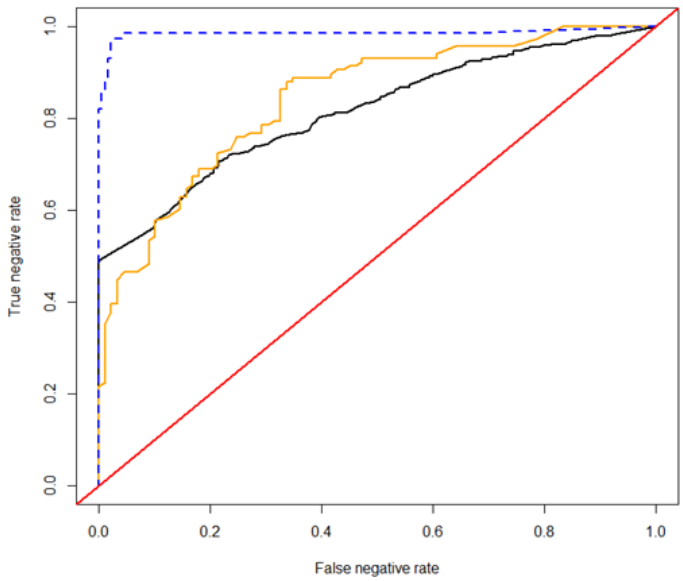


Fig. 8: RF DoS (blue dashed curve), mock app GPS spoofing (black curve) and hardware assisted location spoofing - HackRF (orange curve) classification ROC curve with TNR and FNR threshold



reliance on strict classification of reports prevents recipients (such as law enforcement or emergency services) from prioritising or correlating report cyber trustworthiness scores. Furthermore, optimisation of a two level scoring measure becomes difficult because one must employ a single threshold criterion (e.g., probability = 0.5) to determine whether a report is trustworthy or not; thus limiting the ability for dynamic filtering of inaccurate report scores (e.g., false positives, false negatives etc.).

A more practical approach would be to utilise the class probability of a report (i.e., the likeliness of the device being under attack / an attacker or not) to evaluate a mobile device's cyber trustworthiness. Using this approach, the probability response forms an automatic scoring measure that informs report recipients of the level of trust (and confidence) that can be applied to the report data. For instance, a report with a device cyber trustworthiness score of 90% would tell report recipients that it is highly unlikely that the device has been compromised or is the device of an attacker, whereby comparison a probability of 10% would almost certainly indicate that the device was compromised in some way or indeed an attacker. The resultant probability value also serves to help expedite the most trustworthy reports, which avoids spending time processing reports that may be fake or those intended to create noise and confusion. Moreover, by utilising class probability as a scoring metric, correlation and clustering analysis can be used to group similar trustworthiness results to compare report information which would build a picture of a particular incident or crime through an aggregate trusted view.

In Figures 7 (for true positive rate) and 8 (for true negative rate), the receiver operating characteristic (ROC) curves demonstrate the optimal probability thresholds for each of the attacks evaluated in our experiments. The ROC curve helps to identify different thresholds for minimising false negatives, false positives or achieving the highest classification accuracy. They can therefore be used as a tool for determining the likeliness of cyber trustworthiness accuracy at different probability thresholds, with indications of what cut-off point a probability threshold minimises mis-classification. This is also useful criterion upon which to benchmark cyber trustworthiness probability results, as probability thresholds between this cut-off point can be prioritised over lower probability thresholds that drop below the cut-off, with confidence of their accuracy.

V. Conclusion

The HaaS paradigm is utilised increasingly and in a variety of applications where human users can act as sensors or early warning detectors of events and incidents, from policing and emergency response to pollution detection. A critical factor for the success of HaaS is to be able to estimate or predict the reliability of reports received from human sensors. This depends on the ability of the human sensors themselves (are they able to detect accurately and are they trustworthy?), as well as the trustworthiness of the devices and infrastructures they use to generate and communicate these reports. Here, the focus was on what we referred to as the cyber-trustworthiness of HaaS reports coming from Android mobile devices; which we have defined as a probability measure of whether the device is trusted or not. As a first attempt in this direction, we utilised three standard statistical machine learning approaches for estimating the likelihood of existence of three different attacks, one relating to the health of the network used to communicate the reports, and two relating to the integrity and accuracy of the locations reported. Our initial results show that even a small number of easily monitored features can help estimate HaaS cyber-trustworthiness and enhance the reliability of the paradigm.

ACKNOWLEDGMENT

This work is funded by the European Commission under project TRILLION, grant number H2020-FCT-2014, REA grant agreement no 653256.

REFERENCES

[1] Statista, "Number of mobile phone users worldwide from 2013 to 2019 (in billions)," https://www.statista.com/statistics/274774/forecast-of-mobile-phone-users-worldwide/, Accessed on 02/03/2017.

[2] The Hub, "Smartphone Ownership, Usage And Penetration By Country)," http://thehub.smsglobal.com/smartphone-ownership-usage-and-penetration, 2015, Accessed on 02/03/2017.

[3] C. Patrikakis and A. Konstantas, "Trillion: Trusted, citizen-lea collaboration over social networks," in *International Conference on e-Democracy*. Springer, 2015, pp. 228–232.

[4] C. Z. Patrikakis, G. Loukas *et al.*, "Wear it and share it: Wearables and security," *Cutter Executive Update*, 2017.

[5] C. Macdonell, "Ushahidi: a crisis mapping system," *ACM SIGCAS Computers and Society*, vol. 45, no. 2, pp. 38–38, 2015.

[6] M. Manso and B. Manso, "The Role of Social Media in Crisis: A Holistic Approach to European Adoption of the Online and Mobile Communications in Crisis Response and Search and Rescue Efforts," in *International Command and Control Research and Technology Symposium, ICCRTS*, ser. WWW '12 Companion, 2012.

[7] M. A. Cameron, R. Power, B. Robinson, and J. Yin, "Emergency situation awareness from twitter for crisis management," in *Proceedings of the 21st international conference companion on World Wide Web*, ser. WWW '12 Companion. New York, NY, USA: ACM, 2012, pp. 695–698. [Online]. Available: http://doi.acm.org/10.1145/2187980.2188183

[8] J. Yin, S. Karimi, B. Robinson, and M. Cameron, "ESA: emergency situation awareness via microbloggers," in *Proceedings of the 21st ACM international conference on Information and knowledge management*, ser. CIKM '12. New York, NY, USA: ACM, 2012, pp. 2701–2703. [Online]. Available: http://doi.acm.org/10.1145/2396761.2398732

[9] T. Sakaki, M. Okazaki, and Y. Matsuo, "Earthquake shakes Twitter users: real-time event detection by social sensors," in *Proceedings of the 19th international conference on World wide web*, ser. WWW '10. New York, NY, USA: ACM, 2010, pp. 851–860. [Online]. Available: http://doi.acm.org/10.1145/1772690.1772777

[10] T. Bennett, K. Holloway, and D. P. Farrington, "Does neighborhood watch reduce crime? a systematic review and meta-analysis," *Journal of Experimental Criminology*, vol. 2, no. 4, pp. 437–458, 2006. [Online]. Available: http://dx.doi.org/10.1007/s11292-006-9018-5

[11] J. He, W. Shen, P. Divakaruni, L. Wynter, and R. Lawrence, "Improving Traffic Prediction with Tweet Semantics," in *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, ser. IJCAI '13. AAAI Press, 2013, pp. 1387–1393. [Online]. Available: http://dl.acm.org/citation.cfm?id=2540128.2540328

[12] R. Heartfield and G. Loukas, "Evaluating the reliability of users as human sensors of social media security threats," in *2016 International Conference On Cyber Situational Awareness, Data Analytics And Assessment (CyberSA)*, June 2016, pp. 1–7.

[13] J. R. Nurse, I. Agrafiotis, M. Goldsmith, S. Creese, and K. Lamberts, "Two sides of the coin: measuring and communicating the trustworthiness of online information," *Journal of Trust Management*, vol. 1, no. 1, p. 5, 2014. [Online]. Available: http://dx.doi.org/10.1186/2196-064X-1-5

[14] S. S. Rahman, S. Creese, and M. Goldsmith, "Accepting Information with a Pinch of Salt: Handling Untrusted Information Sources," in *Security and Trust Management*, ser. Lecture Notes in Computer Science, C. Meadows and C. Fernandez-Gago, Eds. Springer Berlin Heidelberg, 2012, vol. 7170, pp. 223–238. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-29963-6_16

[15] J. R. Nurse, S. S. Rahman, S. Creese, M. Goldsmith, and K. Lamberts, "Information Quality and Trustworthiness: A Topical State-of-the-Art Review," in *The International Conference on Computer Applications and Network Security (ICCANS) 2011*. IEEE, 2011.

[16] S. Adali, R. Escriva, M. K. Goldberg, M. Hayvanovych, M. Magdon-Ismail, B. K. Szymanski, W. A. Wallace, and G. Williams, "Measuring behavioral trust in social networks," in *Intelligence and Security Informatics (ISI), 2010 IEEE International Conference on*, May 2010, pp. 150 –152.

[17] M. Alrubaian, M. Al-Qurishi, M. Hassan, and A. Alamri, "A credibility analysis system for assessing information on twitter," *IEEE Transactions on Dependable and Secure Computing*, vol. PP, no. 99, pp. 1–1, 2016.

[18] R. Heartfield, G. Loukas, and D. Gan, "You are probably not the weakest link: Towards practical prediction of susceptibility to semantic social engineering attacks," *Cutter Executive Update*, pp. 6910–6928, 2016.

[19] R. Heartfield and G. Loukas, "A taxonomy of attacks and a survey of defence mechanisms for semantic social engineering attacks," *ACM Computing Surveys (CSUR)*, vol. 48, no. 3, p. 37, 2016.

[20] Tech Transformers, CNBC, "Google Android hits market share record with nearly 9 in every 10 smartphones using it," http://www.cnbc.com/2016/11/03/google-android-hits-market-share-record-with-nearly-9-in-every-10-smartphones-using-it.html, 2016, Accessed on 04/003/2017.

[21] B. K. Sanggeun Song and S. Lee, "The effective ransomware prevention technique using process monitoring on android platform," *Mobile Information Systems*, vol. 2016, p. 9, 2016.

[22] D. Kim, "Automated control method and apparatus of ddos attack prevention policy using the status of cpu and memory," Mar. 1 2012, uS Patent App. 13/216,486. [Online]. Available: http://www.google.com/patents/US20120054823

[23] G. Loukas and G. Öke, "Protection against denial of service attacks: A survey," *The Computer Journal*, p. bxp078, 2009.

[24] L. Xue and G. Sun, "Design and implementation of a malware detection system based on network behavior," *Security and Communication Networks*, vol. 8, no. 3, pp. 459–470, 2015. [Online]. Available: http://dx.doi.org/10.1002/sec.993

[25] G. Canfora, E. Medvet, F. Mercaldo, and C. A. Visaggio, "Acquiring and analyzing app metrics for effective mobile malware detection," in *Proceedings of the 2016 ACM on International Workshop on Security And Privacy Analytics*, ser. IWSPA '16. New York, NY, USA: ACM, 2016, pp. 50–57. [Online]. Available: http://doi.acm.org/10.1145/2875475.2875481

[26] A. Shabtai, U. Kanonov, Y. Elovici, C. Glezer, and Y. Weiss, ""andromaly": a behavioral malware detection framework for android devices," *Journal of Intelligent Information Systems*, vol. 38, no. 1, pp. 161–190, 2012. [Online]. Available: http://dx.doi.org/10.1007/s10844-010-0148-x

[27] Android Developers), "Location - getAccuracy," https://developer.android.com/reference/android/location/Location.html#getAccuracy%28%29, Accessed on 14/03/2017.

[28] B. Dixon, S. Mishra, and J. Pepin, "Time and location power based malicious code detection techniques for smartphones," in *2014 IEEE 13th International Symposium on Network Computing and Applications*, Aug 2014, pp. 261–268.

[29] B. Dixon, Y. Jiang, A. Jaiantilal, and S. Mishra, "Location based power analysis to detect malicious code in smartphones," in *Proceedings of the 1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices*, ser. SPSM '11. New York, NY, USA: ACM, 2011, pp. 27–32. [Online]. Available: http://doi.acm.org/10.1145/2046614.2046620

[30] G. Loukas, *Cyber-physical attacks: A growing invisible threat*. Butterworth-Heinemann, 2015.

[31] The Register, "Android-rooting Gooligan malware infects 1 million devices," https://www.theregister.co.uk/2016/11/30/gooligan_android_malware/, 2016, Accessed on 04/003/2017.

[32] Michael Ossmann, "HackRF One," https://github.com/mossmann/hackrf/wiki/HackRF-One, Accessed on 20/03/2017.

[33] S. S. Keerthi and C. J. Lin, "Asymptotic behaviors of support vector machines with gaussian kernel," *Neural computation*, vol. 15, no. 7, pp. 1667–1689, 2003.