International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169
Volume: 5 Issue: 10 112 – 116

Android Encrypted Network Traffic to Identify User Actions

K. Ravikumar
Dept: Computer Science
Tamil University,
Thanjavur-613010,India

R. Thanga
Dept: Computer Science
Tamil University
Thanjavur-613010, India
Email Id:thangarajaraman93@gmail.com

Abstract: Network forensics is a sub-branch of digital forensics relating to the monitoring and analysis of computer network traffic for the purposes of information gathering, legal evidence. Unlike other areas of digital forensics, network investigations deal with volatile and dynamic information. Network traffic is transmitted and then lost, so network forensics is often a pro-active investigation. Network forensics generally has two uses. The first, relating to security, involves monitoring a network for anomalous traffic and identifying intrusions. The second form relates to law enforcement. In this case analysis of captured network traffic can include tasks such as reassembling transferred files, searching for keywords and parsing human communication such as emails or chat sessions. Nowadays use of mobile apps to communicate with friends. Not only communication purpose it gets information about sensitive topics such as diseases, sexual or religious preferences, etc. Numerous worries have been raised about the capabilities of these portable devices to occupy the privacy of users actually becoming "tracking devices". Above problem they influence in our work to find solution using machine learning techniques. It is used to protect the content of a packet. Our framework analyzes the network communications and leverages information available in TCP/IP packets like IP addresses and ports, together with other information like the size, the direction, and the timing. Our system, for each app they first pre-process a dataset of network packets labeled with the user actions that originated them, they cluster them in flow typologies that represent recurrent network flows, and finally it analyze them in order to create a training set that will be used to feed a classifier. The trained classifier will then be able to classify new traffic traced. Our approach results shows it accuracy and precision more than 95% for most of the considered actions.

Keywords: *Transmission, Traffic Classifier, Dataset, Communication, Network traffic*

*****

I INTRODUCTION

Sensitive data that users handle with their mobile devices. Uses several mobile apps to communicate with friends. Not only communication purpose it gets information about profound topics such as diseases, sexual or religious preferences, etc. several concerns have been raised about the abilities of these portable devices to occupy the privacy of users actually becoming tracking devices[3]. In this framework, an important aspect is related to the probability of continuously spying and locating an individual[2]. Above problem they Solutions in our work to identify and isolate malicious code running on smartphones as well as to protect against attacks coming from the network might significantly reduce current threats to user privacy[6]. People become more familiar with mobile technologies and their related privacy threats.

A network consists of two or more computers that are linked in order to share resources exchange files, allow electronic communications[8]. The computers on a network may be linked through cables, telephone lines, radio waves, satellites, or infrared light beams. A Local Area Network is a network that is confined to a relatively small area. It is generally limited to a geographic area such as a writing lab, school, or building. Computers connected to a network are broadly categorized as servers or workstations[4]. Servers are generally not used by humans directly, but rather run continuously to provide services to the other computers on the network. Services provided can include printing and faxing, software hosting, file storage and sharing, messaging, data storage and retrieval, complete access control for the network's resources, and many others.

Workstations are called such because they typically do have a human user which interacts with the network through them. Workstations were traditionally considered a desktop, consisting of a computer, keyboard, display, and mouse, or a laptop, with integrated keyboard, display, and touchpad[10]. With the advent of the tablet computer, and the touch screen devices such as iPad and iPhone, our definition of workstation is quickly evolving to include those devices, because of their ability to interact with the network and utilize network services. Servers tend to be more powerful than workstations, although configurations are guided by needs[1]. For example, a group of servers might be located in a secure area, away from humans, and only accessed through the network. In such cases, it would be common for the servers to operate without a dedicated display or keyboard. However, the size and speed of the server's processor(s), hard drive, and main memory might add dramatically to the cost of the system. On the other

hand, a workstation might not need as much storage or working memory, but might require an expensive display to accommodate the needs of its user[9]. Every computer on a network should be appropriately configured for its use.

On a single LAN, computers and servers may be connected by cables or wirelessly. Wireless access to a wired network is made possible by wireless access points. These WAP devices provide a bridge between computers and networks. A typical WAP might have the theoretical capacity to connect hundreds or even thousands of wireless users to a network, although practical capacity might be far less. Nearly always servers will be connected by cables to the network, because the cable connections remain the fastest. Workstations which are stationary (desktops) are also usually connected by a cable to the network, although the cost of wireless adapters has dropped to the point that, when installing workstations in an existing facility with inadequate wiring, it can be easier and less expensive to use wireless for a desktop[10].

Wide Area Networks connect networks in larger geographic areas, such as Florida, the United States, or the world. Dedicated transoceanic cabling or satellite uplinks may be used to connect this type of global network. Using a WAN, schools in Florida can communicate with places like Tokyo in a matter of seconds, without paying enormous phone bills. Two users a half-world apart with workstations equipped with microphones and a webcams might teleconference in real time. A WAN is complicated[8]. It uses multiplexers, bridges, and routers to connect local and metropolitan networks to global communications networks like the Internet. To users, however, a WAN will not appear to be much different than a LAN.

Our framework analyzes the network communications and leverages information available in TCP/IP packets (like IP addresses and ports), together with other information like the size, direction (incoming/outgoing), and timing. By using an approach based on machine learning, each app that isof interest is analyzed independently. To set up our system,for each app they first pre-process a dataset of network packets labeled with the user actions that originated them, they cluster them in flow typologies that represent recurrent network flows, and finally they analyze them in order to create at raining set that will be used to feed a classifier. The trained classifier will be then able to classify new traffic traces that have never been seen before[7]. They fully implemented our system, and they run a thorough set of experiments to evaluateour solution considering three very popular apps: Facebook,Gmail, and Twitter. The results shows that it can achieve accuracy and precision higher than 95%, for most of the considered actions.

Due to network congestion, traffic load balancing, or other unpredictable network behavior, IP packets can be lost, duplicated, or delivered out of order[1]. TCP detects these problems, hence requesting retransmission of lost data, and reordering out-of-order data. It comes out that several TCP packets that do not carry data, may hinder the analysis process. In the data exchange phase, for example, the receiver sends a packet with the ACK flag set to notify the correct reception of a chunk of data. These ACK packets are transmitted in asynchronous mode so they are affected by many factors related to round trip time of the connection link. The order of the received packets may hinder the evaluation of the similarity between two network flows.

The same principles that organizations use to monitor network traffic go into their networks must be applied to the network traffic originating from mobile devices. This means that the techniques and tools, which would normally be used to collect and analyze network activity, can also be used to detect anomalous network traffic or network intrusions related to smartphones[5]. This thesis will therefore outline an architecture model, which can be used to analyze the network communications originating from Android devices and to detect any unusual traffic. As part of the exercise, a set of several tests involving real malware will be executed to gauge the effectiveness of said architecture. In addition to that, the aim of the exercise is to improve the detection mechanisms of the engine by creating new signatures to detect specific threats. Lastly, they will define incident-handling steps, which can be used to combat 0-day malware and known malware for which no signatures exist.

To identify mobile apps in network traffic has significant implications in many domains, including traffic management, malware detection, and maintaining user privacy[8]. App identification methods in the literature typically use deep packet inspection (DPI) and analyze HTTP headers to extract app fingerprints. However, these methods cannot be used if HTTP traffic is encrypted. They investigate whether Android apps can be identified from their launchtime network traffic using only TCP/IP headers. They first capture network traffic of 86,109 app launches by repeatedly running 1,595 apps on 4 distinct Android devices. They then use supervised learning methods used previously in the web page identification literature, to identify the apps that generated the traffic. They find that: (i) popular Android apps can be identified with 88% accuracy, by using the packet sizes of the first 64 packets they generate, when the learning methods are trained and tested on the data collected from same device; (ii) when the data from an unseen device (but similar operating system/vendor) is used for testing, the apps can be identified with 67% accuracy; (iii) the app identification accuracy does not drop

significantly even if the training data are stale by several days, and (iv) the accuracy does drop quite significantly if the operating system/vendor is very different.

They model the app identification problem as a multi-class supervised machine learning problem. A supervised machine learning algorithm is first trained with samples each in the form of a pair (x, y) where x and y are the feature vector and the class of a sample, respectively. The algorithm is then expected to predict the class of an unseen sample given its feature vector. In our problem, app launches are the samples, and the app package names are used as the classes of the samples. The feature vector belonging to an app launch is extracted from the TCP/IP headers of the packets[6]. Many apps use networking immediately upon their launch for several reasons. They retrieve new ads from ad-networks or send information to analytics services. They also perform app specific communication—for example, an email app may check for new emails when it is launched. Results established previously in the field of non-mobile traffic classification suggest that, packet sizes observed within such launch time traffic may yield good feature sets for app identification. Specifically, shows that the first few packets of TCP flows can be used to classify Internet applications into categories such as Web, FTP, and Games. They select three methods from the web page identification literature that have been shown to achieve high accuracies using packet sizes. These methods mainly differ in the way feature vectors are extracted from packet sizes and the classification algorithms.

Web page identification experiments in the literature are conducted in laboratory conditions by logging start and end times of web page loads[2]. Similarly, they log the exact start and stop times of apps during our data collection and consider only the network packets generated during this time interval for app identification. However, given a real world traffic trace, it is not easy to detect the start of web page loads or app launches in it. This problem is still being considered as unsolved and left as a future work in the web page identification literature as well as in this thesis. In addition, an app may have already been launched before joining a network. In this case, it is not possible to capture the launch time network traffic of the app[5]. They show that the packet size based traffic analysis methods developed for webpage identification can also be used for app identification. Similarly, the countermeasures in the literature such as the ones that hide the packet sizes by padding can be used to preserve privacy of users. Nine such methods are analyzed.

II EXISTING SYSTEM

Existing system a framework that allows users to monitor how third-party smartphone applications handle their private data in real time. Many smartphone applications are closed-source, therefore, static source code analysis is infeasible. Even if source code is available, runtime events and configuration often dictate information use. Real time monitoring accounts for these environment specific dependencies. The resource limitations of smartphones precludes the use of heavyweight information tracking systems. Third-party applications are entrusted with several types of privacy sensitive information. The monitoring system must distinguish multiple information types, which requires additional computation and storage. Context-based privacy sensitive information is active and can be trying to recognize even when sent in the clear. Applications can share information. Limiting the monitoring system to a single application does not account for flows via files and IPC between applications, including core system applications designed to disseminate privacy sensitive information. Sensitive data is first recognized at a taint source, where a taint marking indicating the material type is assigned. Dynamic taint analysis tracks how labeled data impacts other data in a way that might leak the original sensitive information. This tracking is often performed at the instruction level. Finally, the impacted data is identified before it leaves the system at a taint sink.

A) Demerits

- Lower performance.
- It does not identify isolate malicious code running.

III PROPOSED SYSTEM

In this thesis, network traffic analysis and machine learning can be used to infer private information about the user. They propose a framework to infer which particular actions the user executes on some app installed on her mobile-phone. In particular, they assume that the traffic is encrypted and the adversary eavesdrops without modifying them the messages exchanged between the user's device and the web services. Our framework analyzes the network communications and leverages information available in TCP/IP packets like IP addresses and ports, together with other information like the size, the direction (incoming/outgoing),and the timing. By using an approach based on machine learning, each app that is of interest is analyzed independently. To set up our system, for each app they first pre-process a dataset of network packets labeled with the user actions that originated themin flow typologies that represent recurrent network flows, andfinally they analyze them in order to create a training set that will be used to feed a classifier. Our traffic analysis approach. In

 ISSN: 2321-8169
112 – 116

particular, they examine in depth the concept of network flow and the metric to evaluate the similarity between them. They also report details of the machine learning techniques they leverage in our method result it can achieve accuracy and precision higher than 95% compare to previous work.

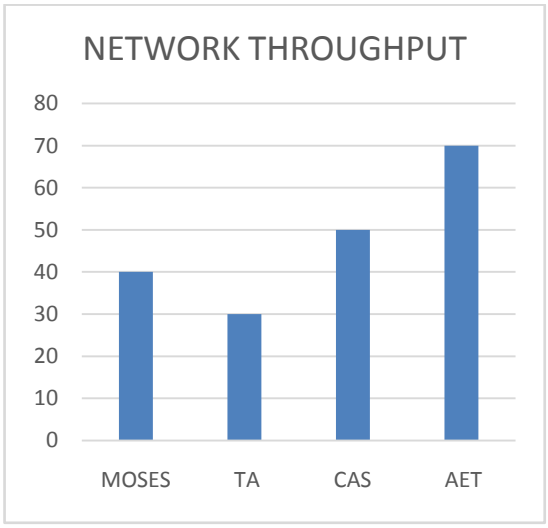
A) Merits

- Higher performance compare to previous work.
- To identify isolate malicious code running on smartphones.
- It can be private information about the user.

IV Experimental Result

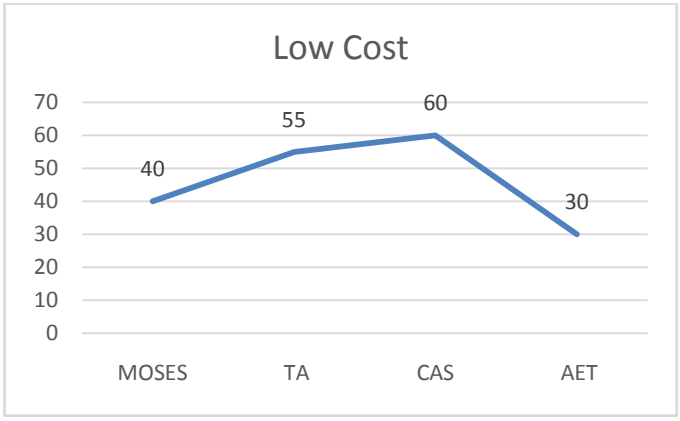
A) Throughput

In these experiments Throughput**.** MOSES gives a result with throughput of 40%. TA gives a result with throughput of 30%. CAS gives a result with throughput of 50%. AET gives a result with throughput of 70%.



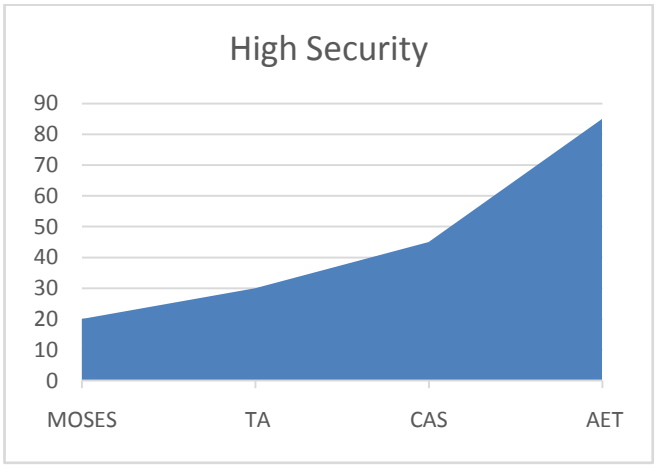
B) Cost

In these experiments Throughput**.** MOSES gives a result with throughput of 40%. TA gives a result with throughput of 55%. CAS gives a result with throughput of 60%. AET gives a result with throughput of 30%.



C) High Security

In these experiments Throughput**.** MOSES gives a result with throughput of 20%. TA gives a result with throughput of 30%. CAS gives a result with throughput of 45%. AET gives a result with throughput of 85%.



V CONCLUSION

In our proposed framework investigated to what level such an outer attacker can identify the specific actions that a user is performing on her mobile apps. They design a system that achieves this goal using advanced machine learning techniques. With this tool an adversary may easily learn habits of the target users. They hope that this work will shed light on the possible attacks that may undermine the user privacy, and that it will stimulate researchers to work on efficient countermeasures that can also be adopted on mobile devices.

VI FUTURE WORK

In future, our thesis work on efficient countermeasures that can also be adopted on mobile devices. Then improve efficiency and scalability.

REFERENCE

[1] Yury Zhauniarovich, Giovanni Russello, "MOSES: Supporting and enforcing security profiles on smartphones", 2014.

[2] Scott E. Coull,Kevin P. Dyer, "Traffic Analysis of Encrypted Messaging Services: Apple iMessage and Beyond", 2014.

[3] Younghwan Go, Denis Foo Kune, "Towards Accurate Accounting of Cellular Data for TCP retransmission", 2013.

[4] Shuaifu Dai, Alok Tongaonka, "NetworkProfiler: Towards Automatic Fingerprinting of Android Apps", 2013.

[5] Balachander Krishnamurthy, "Privacy and Online Social Networks: Can colorless green ideas sleep furiously?" 2013.

[6] G. Suarez-Tangil, J. E. Tapiador, "Evolution, detection and analysis of malware for smart devices", 2013.

[7] Roman Schlegel,Kehuan Zhang, "Soundcomber: A Stealthy and Context-Aware sound Trojan for Smartphones", 2011.

[8] Andriy Panchenko,Lukas Niessen, "Website Fingerprinting in Onion Routing Based anonymization Networks", 2011.

[9] Fabian Schneider,Anja Feldmann, "Understanding Online Social Network Usage from a Network Perspective", 2009.

[10] Dawn Xiaodong, Song David Wagner, "Timing Analysis of Keystrokes and Timing Attacks on SSH", 2008.