A Platform for the Analysis and Visualization of Network Flow Data of Android Environments

Abdelkader Lahmadi
University of Lorraine, LORIA
Vandoeuvre-lès-Nancy, F-54506, France
Email: abdelkader.lahmadi@loria.fr

Frederic Beck and Eric Finickel and Olivier Festor
INRIA Nancy Grand-Est
Villers-lès-Nancy, F-54600, France
Email: {frederic.beck, eric.finickel, olivier.festor}@inria.fr

***Abstract*—In this demo, we present a monitoring platform dedicated to the collection, storage, analysis and visualization of logs and network flow data of mobile applications. The platform relies on a set of on-device probes to monitor network and system activities of these applications. The data are collected from these probes and parsed through generic and flexible collectors relying on Flume agents that we have adapted and extended. We are storing the collected data using a column oriented Hbase storage engine which is the Hadoop database. Finally, after being parsed, the data are made available within the Elasticsearch engine to search and visualize them using the Kibana tool.**

I. Introduction

Android-based devices, including smartphones and tablets, are emerging and widely adopted by users because of their interesting capabilities. They evolved from simple telephony devices into sophisticated, yet compact, PC connected to a wide spectrum of networks, including Internet via Wi-Fi, GPRS/EDGE or 3G network access. The network traffic generated by these devices is increasing, since they enable users to access and browse web sites, send emails, play on-line games and exchange information. It is projected that mobile traffic will grow 10 times faster than fixed Internet traffic. The analysis of the network behavior of applications running on a smartphone device requires the collection of information about data leaving the device, and where it is sent. Observing network traffic activities is a popular approach to monitor and assess the security of deployed networked devices, and it is able to provide information about spectrum of traffic mix, attacks, applications, etc. However, the characteristics of the traffic generated by smartphones and their network-active applications remain wholly unexplored, where only few works provide first insights about this traffic, mainly from packet capture datasets. One likely reason why individual smartphone traffic has gone unstudied with fine-grain view, is that mobile dedicated probes and monitoring platforms are missing, unlike wide-area Internet where several probes and platforms have been developed and deployed to monitor and record traffic through access links.

In this paper, we present a monitoring and measurement platform that relies mainly on Big Data oriented components to collect, analyze and visualise network flow data generated by Android running applications. The network flow data of each running application is measured on device using our flow based monitoring probe, named Flowoid, which is dedicated to Android environments. Flow data are recorded and exported by Flowoid to be collected using a scalable architecture. Then, they are stored in column oriented database, indexed and finally visualized and analyzed. This platform allowed us to explore and identify communications patterns of several mobile applications.

The rest of the paper is organized as follows. In section II, we describe the architecture and internals of our monitoring platform. Then, we provide insights about obtained results using the described platform, mainly traffic patterns of Android network-active applications that we have studied, and that we will be able to show in the demo. Finally, conclusion and future work are described in III.

II. Platform architecture and details

Our platform relies on different components to measure, collect, store and visualize logs and network flow data of Android applications: a set of probes and agents running on Android devices, a set of collectors, a storage engine, the index engine and a visualization tool. Figure 1 depicts the overall architecture of the platform. In the following subsections, we will detail each component.

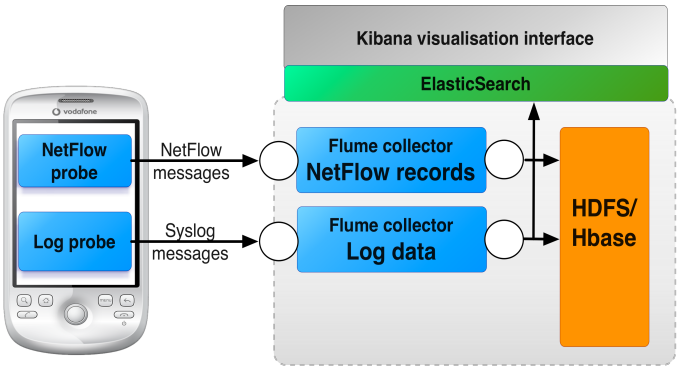

Fig. 1. Architecture and details of the collection and analysis platform.

A. On device probes

We have developed mainly two on-device probes to collect data about running applications. The first probe is Flowoid, which is a NetFlow exporter for Android devices. It is composed of a backend IP driver retrieving the packets using the pcap library, a NetFlow version 9 exporter creating and exporting the flows, and a GUI, the Flowoid app itself. Our exporter associates each observed network flow with geolocation data,

consisting of the GPS coordinates of the device, among others. This information is exported together with the traditional fields defined in the NetFlow/IFPIX protocols: IP version, source and destination addresses, the number of exchanged bytes, the type of protocol, the number of exchanged packets, the source and destination ports and the duration of the flow. In addition, each record contains several additional fields associated to the context to the flow: the identifier of the localization method, a timestamp, the integer part of the latitude, the decimal part of the latitude, the integer part of the longitude and the decimal part of the longitude, the name of the mobile application that initiated the connection, the state of the device screen (On, Off, locked, unlocked), an identifier of the device, the type of the device connectivity (Wifi, EDGE, CDMA, HSPA, LTE, ...) and the state of the application (background, foreground) when the flow has started. The second probe is related to the log data associated to each running application. The data are collected on device using the *dumpsys* and *logcat* tools provided by the Android platform. Then, all data are exported periodically to collectors using the syslog protocol, where it is parsed and stored. We also associated to each log data additional information related the geolocation of the device.

B. Data collection and storage

The data sent by the probes is collected using Flume based collectors. Apache Flume[1] is a distributed, reliable, and available system for efficiently collecting, aggregating and moving large amounts of log data from many different sources to a centralized data store. Each unit of data is considered as an event that flows through a Flume agent. The event travels from a source through a channel to a Sink. An event carries a payload (byte array) that is accompanied by an optional set of headers. Flume has the capability to modify or drop events in-flight. This is done with the help of interceptors. We have developed and integrated in Flume agents a set of sources, sinks and intercepters to handle the collected data. We have mainly two sources dedicated respectively to NetFlow and Syslog messages. We have also developed two interceptors that parse the different messages (NetFlow and Syslog) and transform them into JSON documents. Each JSON document is finally sent to a sink to be stored in a Hbase table. The listing 1 denotes an example of a NetFlow record exported by Flowoid and parsed using a Flume interceptor.

C. Data search and visualisation

The stored data is then indexed using the Elasticsearch[2] engine which allows us to easily search inside log and flow data. We have also deployed Kibana which is a web-based visualization and dashboard builder tool that interacts with Elasticsearch to make search requests, aggregate data and visualize them using several visualisation tools: piecharts, tables, bar charts and tile maps. Figure 2 depicts an example of three layers piechart obtained using Kibana. The piechart shows the most contacted servers (third outer layer), destination ports (second inner layer) used by the Android version of the Facebook application (first inner layer). We mainly observe that the Facebook application relies on the HTTPS service (port 443) to contact its servers. However the recently

[1]http://flume.apache.org/
[2]http://www.elasticsearch.org/

Snippet 1 An example of a NetFlow record exported by Flowoid and parsed by the Flume collector.

```
"IPVersion": "4",
"LongituteDecimal": "1767651",
"LongitudeInteger": "6",
"SrcAddr": "10.35.67.24",
"DeviceID": "362507473",
"DstAddr": "173.252.102.16",
"LatitudeDecimal": "7097251",
"SrcPort": "33294",
"LatitudeInteger": "48",
"Packets": "18",
"AppName": "com.facebook.katana",
"EndTime": "108799063",
"StartTime": "108457518",
"Bytes": "2540",
"DstPort": "443",
"Protocol": "6",
"TCPFlags": "16"
```

available lite version of Facebook is using the 8000 port. All the components of the application are using the TCP protocol.

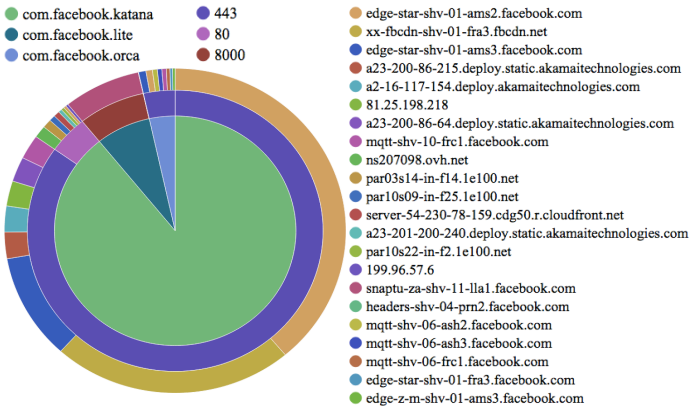

Fig. 2. A piechart of the communication patterns of the Android version of Facebook application.

III. Conclusion

In this paper, we presented the architecture and the different components of a monitoring platform dedicated to Android environments. Relying on this platform, we are mainly collecting and analyzing logs and network flow data related to Android applications lifecycle activities. The used technologies and the available tools within this platform will offer new perspectives to investigate new approaches and solutions for the security monitoring of managed Android environments. We are currently developing and extending our platform with more analysis and visualization modules to enhance its interactivity.

Acknowledgments

This work was partly funded by Flamingo, a Network of Excellence project (ICT-318488) supported by the European Commission under its Seventh Framework Program.