MITIK-MGMT: Data Collector Management Tool

Fernando Molano Ortiz, Fernando Dias de Mello Silva, Aline Carneiro Viana, Nadjib Achir

Mobility and contact traces from
non-intrusive passive measurements

ANR PRC call

MITIK-MGMT
Data Collector Management Tool

Version v1.0

Fernando Molano Ortiz[1], Fernando Dias de Mello Silva[2], Aline Carneiro Viana[1], Nadjib Achir[1]

[1] *INRIA, France,* [2] *Universidade Federal do Rio de Janeiro, Brazil.*

Contents

Copyright

Acknowledgment

This work has been partially funded by the ANR MITIK project, French National Research Agency (ANR), PRC AAPG2019.

1 Introduction

The objective of the MITIK project is to carry out non-intrusive passive measurements to analyze the mobility of users through contacts during their travels. The objective is to use probe-request packets coming from mobile devices using WiFi type wireless communications. `MITIK-MGMT` is a management tool developed as part of the MITIK project and which aims to automate the configuration process and management of experiments using WiFi collectors offered in MITIK. The supported functions are:

- Provide a tool that allows the configuration of multiple collectors simultaneously.
- Centralized management of several collectors (synchronization, raw data capture, data transfer and data processing).
- Configuration of parameters and execution of MITIK project modules.

1.1 MITIK Project architecture

For that, the goal is to use probe request packets from off-the-shelf mobile devices with wireless communication (specifically WiFi). To ensure GPDR compliance, a data anonymization/sanitization process is carried out so that probe requests are received by the sniffers. The system, as a whole, is composed of an architecture that comprises three phases, as shown in Figure 1.

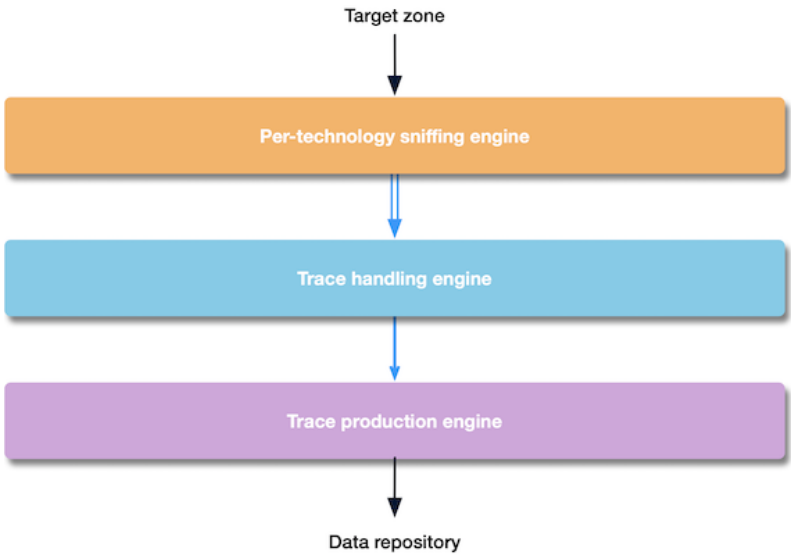

Fig. 1: Functional architecture of Mitik project.

The **Sniffing engine** comprises the raw data capture through wireless antennas. Data can be Stored in pcap files or in a trace format defined for all packets captured. Besides that, the data anonymization implements security algorithms on the raw data captured to ensure privacy of the sniffed users. For that, cryptographic algorithms are used to hash and truncate the MAC address of the mobile devices from users. Some developments have been forwarded.

On the other hand, the **Trace handling engine** defines all the steps necessary for the data analysis, since cleaning, formatting, handling of MAC correspondence, time synchronization between traces from the sniffers, separation and concatenation of traces per antenna. Some developments have been forwarded.

By last, the **Trace production engine** comprises the interpretation of data captured and processed in the previous engines. The goal of this module is to take individual traces to compute and analyze the mobility, and explore different patterns and behaviors of the human mobility.

Therefore, there is a need to integrate all the current developments into one unique Mitik code. A goal in Mitik project is to develop an automated management system in order to reduce manipulation by the human manager.

2 Implementation

Currently, the sniffer manager is configurated in a PC (Macbook Pro). We use own made Ansible-based tool to configure the sniffers with all the required parameters to manage the configure system. Ansible enables configuration process remotely or locally. Ansible also has a configuration language that allows to describe how to define the state of items. As a result, Ansible allows for more efficient management, automating time-consuming tasks typically performed by users, and minimizing error-prone.

An inventory to define the hosts and groups of hosts participants that will be operated by the playbooks. Once the inventory has been defined, a set of playbooks have been programmed to perform tasks on the sniffers. The execution of the playbooks depends on the modules available in the Ansible platform. Finally, the system global interpreter is based on Python.

The sniffers' deployment require two roles to be developed. The first environment has been created to perform the installation and configuration tasks required on the hardware and the O.S.; the second environment executes the tasks necessary to start the sniffer according to the required parameters, in addition to synchronizing the data with the sniffer manager (Mitik laptop) and the Mitik server. Figure 2 shows the scenario to be automated by the Ansible management tool.

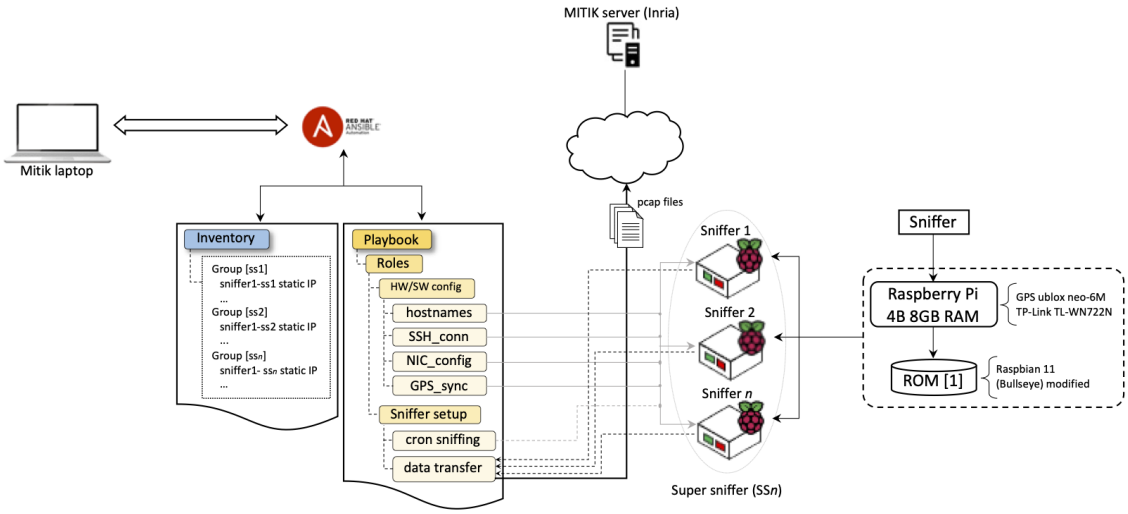

Fig. 2: `MITIK-MGMT` methodology for sniffer management.

2.1 Hardware and software requirements

A modified version of Raspian has been deployed, as part of the experiments evaluating the performance of low-level libraries to capture network traffic in the sniffer. However, it is necessary to carry out additional configurations in the sniffer to add new functionalities on it.

2.2 Sniffer manager

The sniffer manager comprises several functions:

- Centralized management of multiple sniffers (time synchronization, raw data capture, data transfer, data processing, among others).
- One-step, one-time configuration of passwordless ssh access.
- One-step configuration of multiple sniffers on all leverls: networking, software, gps...
- Configuration of parameters and execution of code source of the MITIK-SENS project.
- Integration of functional blocks [4] developed for data capture / data anonymization/sanitization (Phase 1 - Per-technology sniffing engine), handling of collected data (Phase 2 - Trace handling engine), and analysis of human mobility (Phase 3 - Trace production engine), based on data collected from contacts of mobile devices detected by sniffers.

The immediate objective is to integrate these functional architectures to have a global functional vision of the solution, and add or make the necessary improvements.

2.3 Installation

A sniffer manager is defined in a Mitik laptop (Macbook Pro). All the instructions executed in the sniffers are defined in the sniffer manager. It requires installing ansible on your Sniffer manager (Mitik laptop):

```
$ sudo apt install ansible -y
```

Then clone the `MITIK-MGMT` repository and enter that folder:

```
$ cd ~
$ git clone https://gitlab.inria.fr/mitik/measurement-management/mitik-mgmt
$ cd mitik-mgmt
```

To establish communication with the sniffers, an inventory with specific parameters is defined.Each sniffer is assigned a static IP. Besides that, sniffers are organized in groups (super-sniffers). inventory shows the definition for each sniffer. Four groups (super-sniffers) are defined (ss1 to ss5), and each one contains five sniffers (sniffer#-ss#). In addition to the inventory, the sniffer manager sends all the instructions and configurations contained in the playbooks to be executed in each sniffer.

```
[ss1]
sniffer1-ss1 ansible_host=192.168.0.101
sniffer2-ss1 ansible_host=192.168.0.102
#
[sniffers:children]
ss1
ss2
#
[all:vars]
ansible_connection=ssh
ansible_ssh_user=<user>
ansible_ssh_pass=<passwd>
ansible_python_interpreter=/usr/bin/python3
```

It is also possible to test the connection with all the nodes:

```
# PING remote hosts
$ ansible -i inventory -m ping all
```

2.4 Playbooks

The playbooks deployed for the `MITIK-MGMT` are described below:

```
|_ hostname.yml
|_ hosts.yml
|_ SSH_keygen.yml
|_ NIC_config.yml
|_ GPS_sync.yml
|_ test.yml
|_ scapy-sniffer_GPS.yml
|_ data_transfer.yml
```

3 How to use the tool

Next, The playbooks used for configuration are described. As showed in Figure 2, roles of HW/SW configuration and Sniffer setup are deployed for the `MITIK-MGMT` tool.

3.1 HW/SW configuration

Playbooks related to HW/SW configuration can be executed just one time.

Authentication

SSH key-based is used as authentication method between the sniffer manager and the sniffers. It is indispensable for the secure exchange information and data between the entities (sniffer manager and sniffers), besides of the execution of specific functions that requires SSH authentication. To enable the SSH Key-based authentication setup between the sniffer manager and the sniffers, playbook_SSH_Keygen generates the public key of each sniffer, and also copy their SSH public keys to the sniffer manager.

```
# SSH keygen
$ ansible-playbook -i inventory playbook_SSH_Keygen.yml -u <user>
```

Wireless interfaces

Each Raspberry Pi is equipped with external TP-Link TL-WN722N wireless interfaces which make available the **monitor mode**, required for the analysis of captured network traffic in the sniffer.

To avoid randomness in the network interface names, the Predictable network interface names is disabled, and new udev rules are defined for assigning static interface names for each USB port.

It is also defined a wireless network to connect each sniffer to the remote server through wlan0. These parameters are defined in playbook_NIC_config. It is worth noting that all wireless interfaces used to sniff must be connected in the same USB port for all sniffers.

```
# SSH keygen
$ ansible-playbook -i inventory playbook_NIC_config.yml -u <user>
```

GPS Synchronization

We consider to use time synchronization via GPS with PPS (Pulse Per Second) signal. PPS provides accurate timing signal to assist with precision synchronization clock. It is provided as a TTL signal at either 3.3 V or 5 V. Each Raspberry Pi is equipped with an u-blox NEO 6M-0-001. This module has one timepulse PPS module, in addition to the stand-alone GPS receiver.

Install `gpsd` for GPS decoding of both time and position; `pps-tools` to verify PPS signals from the GPS; and `chrony` to handle PPS signals and time synchronization of the sniffers.

```
$ sudo apt install gpsd gpsd-clients pps-tools chrony
```

Whole the steps for the deployment of the GPS are deployed in playbook_GPS_sync:

```
# SSH keygen
$ ansible-playbook -i inventory playbook_GPS_sync.yml -u <user>
```

3.2 Sniffer setup

Unlike the single execution tasks of role 1 (HW/SW configuration), the tasks of role 2 (Sniffer setup) can be executed multiple times, as long as they correspond to execution variables of the sniffer script. Therefore, a playbook to test the super-sniffers/sniffers state is deployed:

```
# TEST remote hosts interfaces
$ ansible-playbook -i inventory playbook_test.yml -u <user>
    ## EXEC tests individually
    # TEST Wi-Fi interfaces
    $ ansible-playbook -i inventory playbook_interfaces_status.yml -u <user>
    # TEST GPIO ports (Detect GPS module)
    $ ansible-playbook -i inventory playbook_GPIO_status.yml -u <user>
    # TEST O.S. Services status
    $ ansible-playbook -i inventory playbook_service_status.yml -u <user>
    # TEST Time sync
    $ ansible-playbook -i inventory playbook_time.yml -u <user>
```

It is necessary to enter the online parameters to run the sniffer. Host variables are defined in playbook_scapy-sniffer. Also, a timeout to stop sniffer execution has been added. By last, a job scheduling utility has been added to ensure that network time protocol set by Chrony is up to date for all sniffers. To start the sniffers, the following parameters must be defined:

```
# EXEC sniffer via crontab
$ ansible-playbook -i inventory playbook_scapy-sniffer_GPS.yml -u <user>
    # Arguments:
    - Location of the experiment
    - Hour to start the experiment,
    - Minutes to start the experiment,
    - Runtime duration (seconds,
    - Wireless interface (by default, wlan1),
    - Packet capture filter (by default, probe-req, probe-resp, beacon),
    - Channel (by default, system),
    - Hash funtion (by default, MD5),
    - Hash pattern (by default, 15),
    - Folder destination.
```

The capture filename structure produced by sniffers will have the next format:

```
packet_capture_{sniffer_id-super-sniffer_id}-ts-{timestamp}-ch{channel}-gps{lat/lon}.
    pcap
```

On the other hand, each single capture file from the sniffers is sent via SSH connection to the Mitik laptop or Mitik server to be analyzed in the Trace handling engine and the Trace production engine. The playbook_data_transfer contains the instructions to send the data to the Mitik laptop or Mitik server.

```
# TRANSFER pcap files to laptop
$ ansible-playbook -i inventory playbook_data_transfer.yml -u <user>
```

4 Link for the tool

The tool can be found on the following link:
`https://gitlab.inria.fr/mitik/measurement-management/mitik-mgmt`

5 License

This code has been developed within the ANR MITIK project for research purposes. It is released under the license GNU General Public License v3.0 or later. While you are welcome to explore and utilize it for academic or research purposes, we cannot guarantee ongoing support or updates. Use of this code is at your own discretion, and we encourage you to exercise caution and discretion in its adaptation. Terms and conditions to use this software are detailed at the GitLab text of the tool license in `https://gitlab.inria.fr/mitik/measurement-management/mitik-mgmt/-/blob/main/LICENSE?ref_type=heads`.

References

[1] Fernando Molano Ortiz et al. “Collecte de traces WiFi publiques: de la protection de la vie privée à l’analyse de trajectoires”. In: *CoRes 2024 - 9èmes Rencontres Francophones sur la Conception de Protocoles, l’Évaluation de Performance et l’Expérimentation des Réseaux de Communication*. May 2024, pp. 1–4.