GroDDViewer:
Dynamic dual view of Android malware

Jean-François Lalande
Mathieu Simon Valérie Viet Triem Tong

GraMSec 2020

CIDRE team

June 22th 2020

Introduction

Android malware analysis

Android malware analysis

- static analysis: (byte)code parsing + Control Flow Graph analysis
- dynamic analysis: execution (smartphone, cuckoo sandbox)

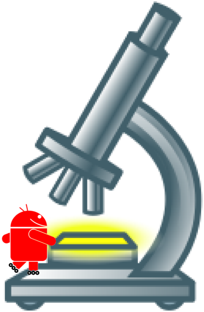
Android malware analysis

Android malware analysis

- static analysis: (byte)code parsing + Control Flow Graph analysis
- dynamic analysis: execution (smartphone, cuckoo sandbox)

Reverse engineering:

- go deep into the bytecode
- **observe** what happens when executed



By Con-struct + replicant community [CC BY-SA 3.0]

Tools for helping the reverser

Dynamic analysis tools for Android apps:

- focus on the quality of outputs
- **do not** focus on **visualizing**

We believe that a good vizualisation tool should:

1. represents what happens at **OS level**
2. represents what is inside the **bytecode**
3. help the investigator to understand a malware

Malware analysis

Examples

Remote Admin Tools:

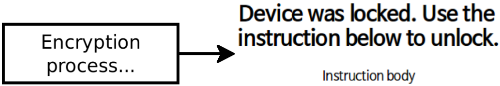
- **Badnews**: Obeys to a remote server + delays attack
- **DroidKungFu1** (well known): Delays attack
- **Mazar**: RAT + Spyware

Blocker / Eraser:

- **WipeLocker**: Wipes of the SD card

Ransomware

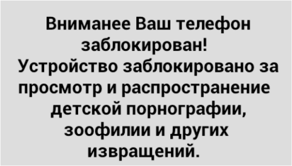
SimpleLocker: Encrypts user’s files and asks for paying



Encryption process...

Device was locked. Use the instruction below to unlock.

Instruction body

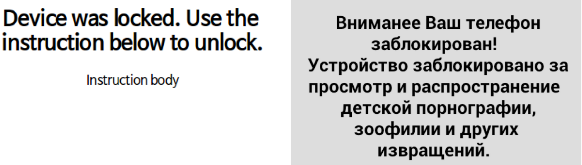


Вниманее Ваш телефон заблокирован!
Устройство заблокировано за просмотр и распространение детской порнографии, зоофилии и других извращений.

Ransomware

SimpleLocker: Encrypts user’s files and asks for paying

Encryption process...


Device was locked. Use the instruction below to unlock.

Instruction body

Вниманее Ваш телефон
заблокирован!
Устройство заблокировано за
просмотр и распространение
детской порнографии,
зоофилии и других
извращений.

⇒ We would like to **see**:

- the encrypted files
- the part of the bytecode involved

Visualization needs

- Observe what happens in the system (files, sockets)
- Identify the involved parts of the code
- Observe malware over time

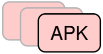
Visualization needs

- Observe what happens in the system (files, sockets)
- Identify the involved parts of the code
- Observe malware over time

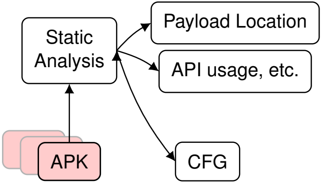
⇒ We created GroDDViewer for answering these problems !

- Grodd: the intelligent monkey of Marvel's comics
- D: Dynamic (replay an experiment)
- D: Dual view (OS + Code)

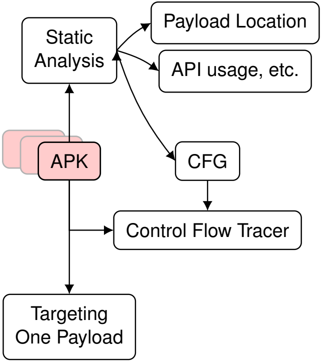
Our analysis framework: GroddDroid



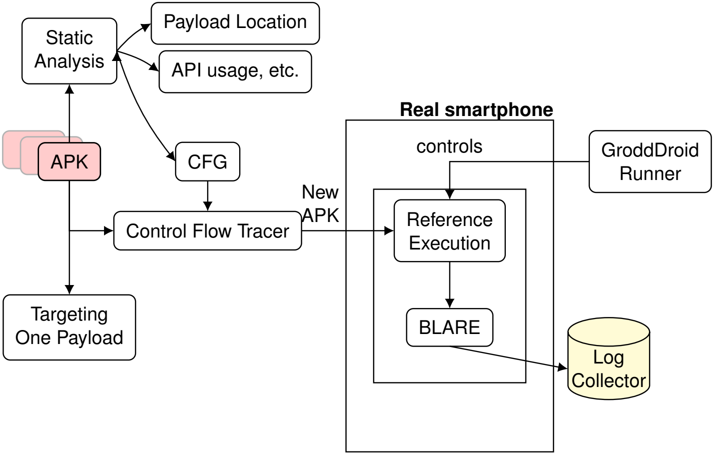
Our analysis framework: GroddDroid



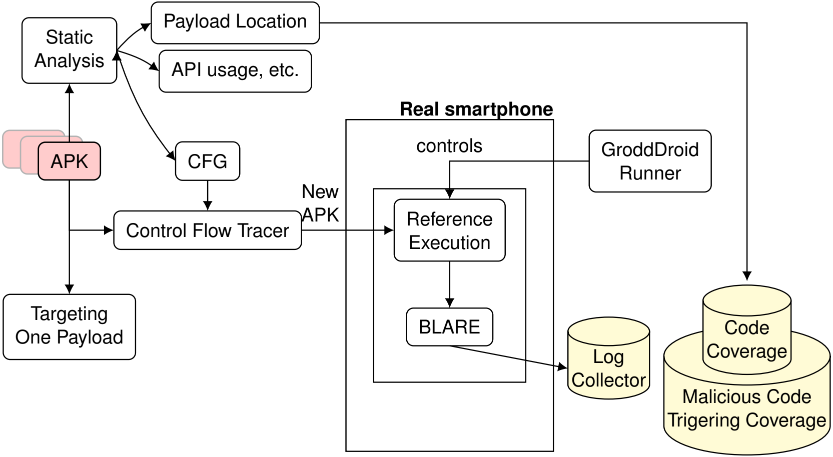
Our analysis framework: GroddDroid



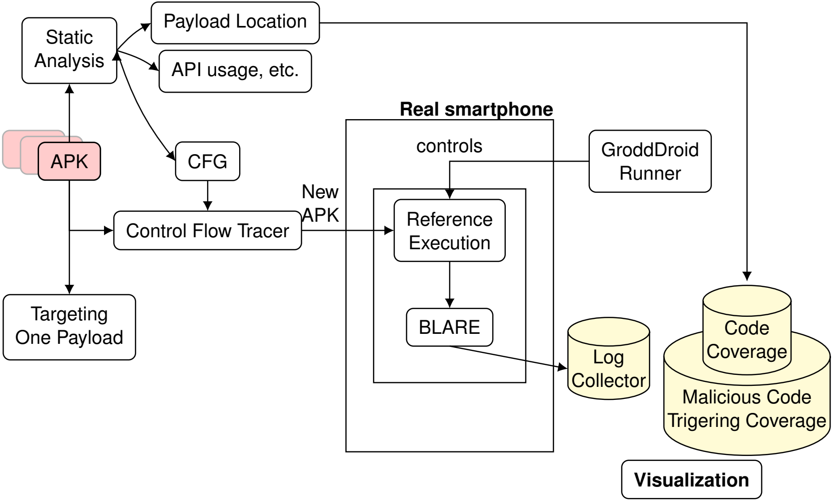
Our analysis framework: GroddDroid



Our analysis framework: GroddDroid

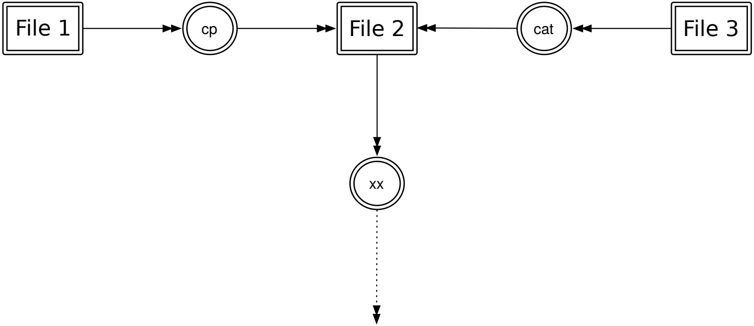


Our analysis framework: GroddDroid



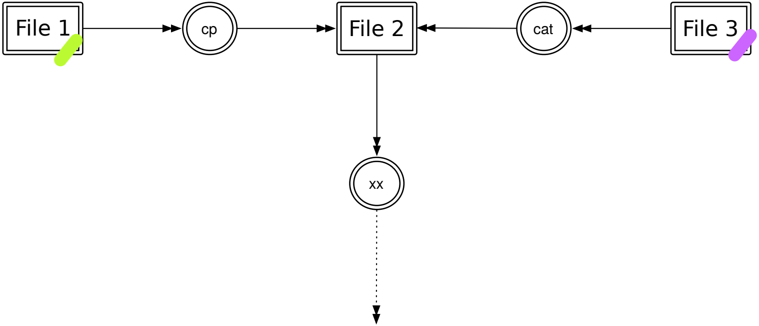
Blare monitoring: principle

1. Marks files with a mark
2. Observes propagation of flows



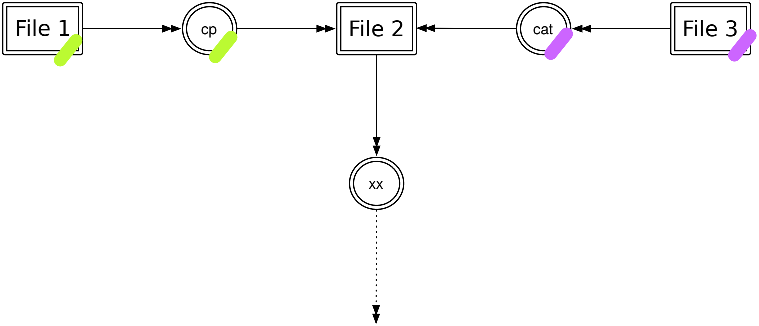
Blare monitoring: principle

1. Marks files with a mark
2. Observes propagation of flows



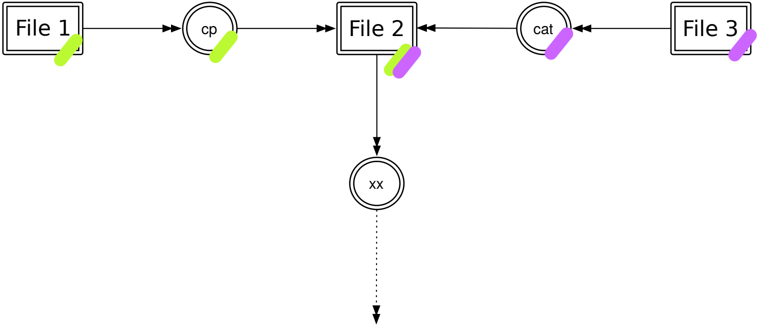
Blare monitoring: principle

1. Marks files with a mark
2. Observes propagation of flows



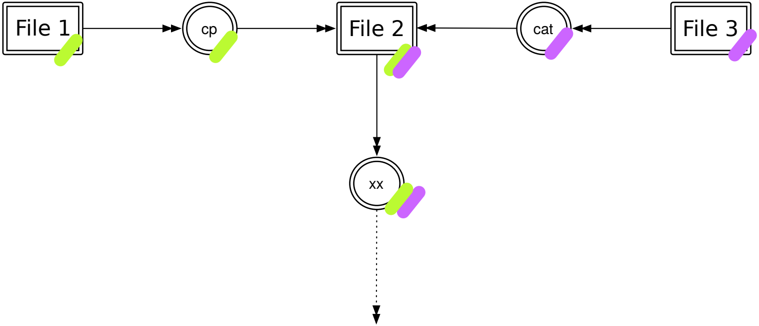
Blare monitoring: principle

1. Marks files with a mark
2. Observes propagation of flows



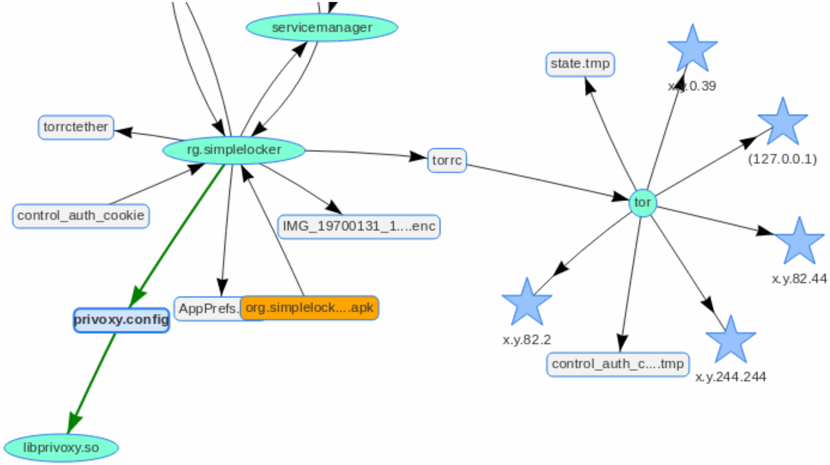
Blare monitoring: principle

1. Marks files with a mark
2. Observes propagation of flows



Visualization

GroddViewer example: simplelocker



GroddViewer demo

Conclusion

Future works

Not solved problems for dynamic observation

- Native code
- Obfuscation
- Remote servers

New vizualisation problems

- Enhance the navigation into the code
- Deal with the visualization of protocols

Questions ?

©Inria / C. Morel