Kharon : Découvrir, comprendre et reconnaître des malware Android par suivi de flux d'information

Rendez-vous de la Recherche et de l'Enseignement de la Sécurité des Systèmes d'Information, Troyes, 2015

Adrien Abraham & Radoniaina Andriatsimandefitra
Thomas Genet & Laurent Guillo
Jean François Lalande & David Pichardie
Valérie Viet Triem Tong

Sécurité des tablettes et smarphones Android

Mobile Malware

Une application populaire, c'est

- 50 000 000 à 100 000 000 téléchargements
- facile à soumettre sur Google Play
- noyée parmi tant d'autres : 800 000 apps début 2013 (Google Play)
- et donc éventuellement malveillante : + 1200 % d'apps malveillantes en 2012

Sécurité des tablettes et smarphones Android

Une application Android c'est aussi

- du bydecode Dalvik
- des ressources : images, fichiers de config de l'appli . . .
- ?

Que font les malware Android ?

Services payants à l'insu de l'utilisateur

SMS surtaxés

Vol de données

rançon, publicité ciblée

Controle à distance du téléphone (RAT)

pour la création de botnet

Que sont les malware Android ?

Souvent dissimulé dans une autre application

pour augmenter la surface d'attaque

du bytecode dalvik

obfusqué / chargé dynamiquement

du code natif

obfusqué / chargé dynamiquement / auto-modifiant

On constate qu'il est difficile de

- reconnaitre statiquement un code malveillant inconnu
- comprendre ce que fait un code malveillant même reconnu

Néanmoins

Si deux applications sont infectées par le même malware alors il y a des similarités dans leur comportements

Le projet labex CominsLabs *Kharon* (2015-2018)

Construire une plateforme d'analyse en ligne

- reconnaissant les apps infectées
- permettant de découvrir de nouveaux malware

Calculer des signatures *comportementales*

- qui caractérise avec précision ce que FAIT un malware

Proposer une bibliothèque de malware bien documentée

- A disposition des chercheurs

Ressources humaines

- 4 enseignants-chercheurs et un ingénieur (CentraleSulelec, ENS Rennes, INRIA, INSA Centre-Val de Loire, Univ. Rennes 1)
- un ingénieur expert (1 an) et un doctorant

Challenges

✓ Capturer le comportement de l'application dans son environnement

∽ Identifier statiquement du code malveillant

⚠ Déclencher l'exécution du code malveillant

⚠ Executer tout le code malveillant

⚠ Faire cela sur des millions d'applications

⚠ Découvrir de nouveaux comportements malveillants

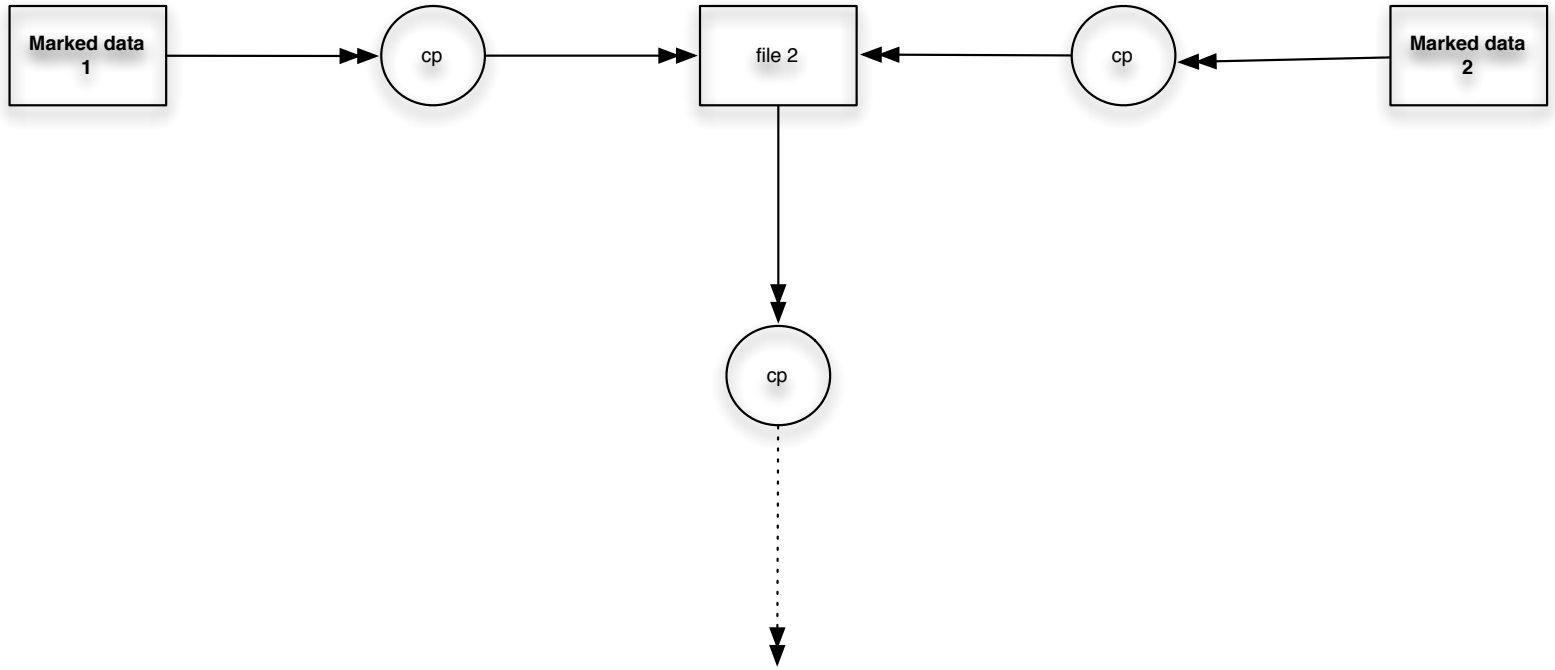
Capturer le comportement d'une application

Blare un moniteur de flux d'information

- observe comment une donnée marquée se propage dans le système
- Utilise des techniques de *tainting*
- `https://www.blare-ids.org/`

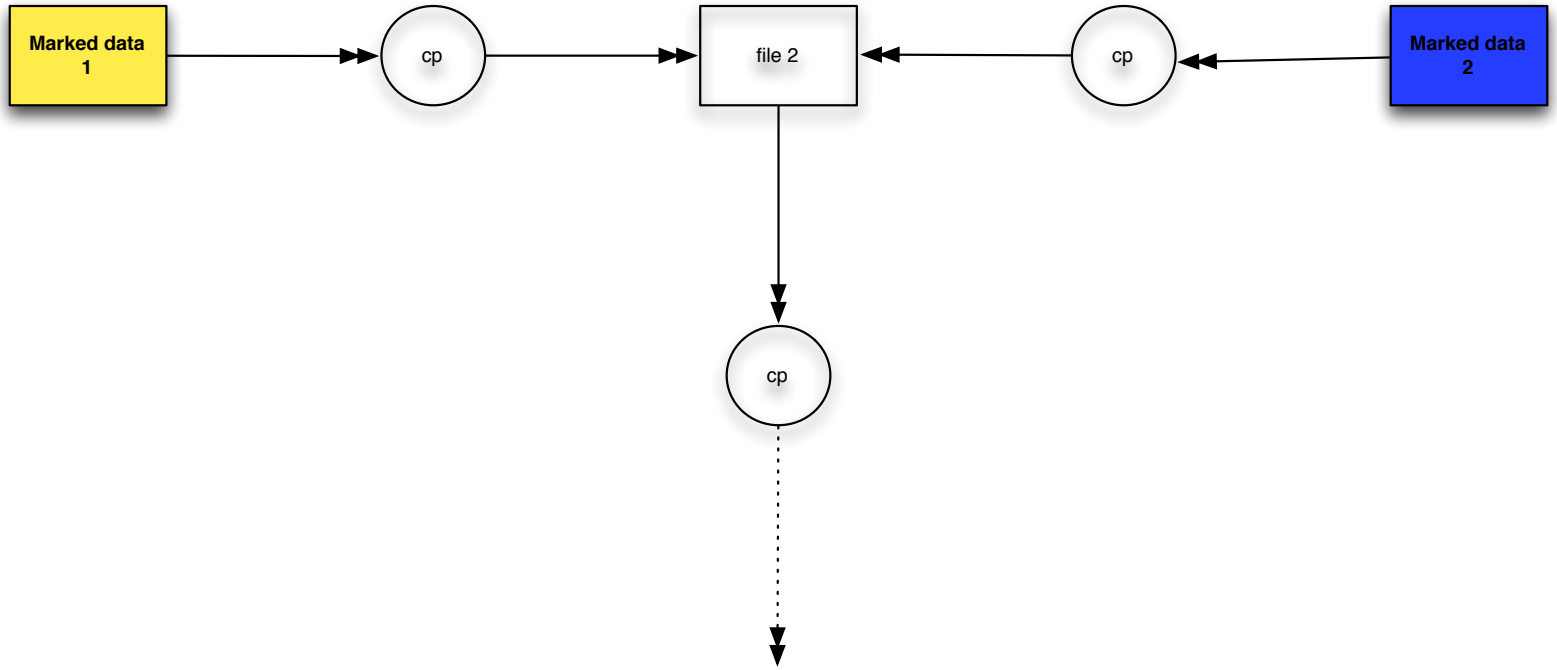
Blare : Comment ça marche ?

1. On attache une marque aux données à surveiller
2. Les marques sont propagées avec ces données



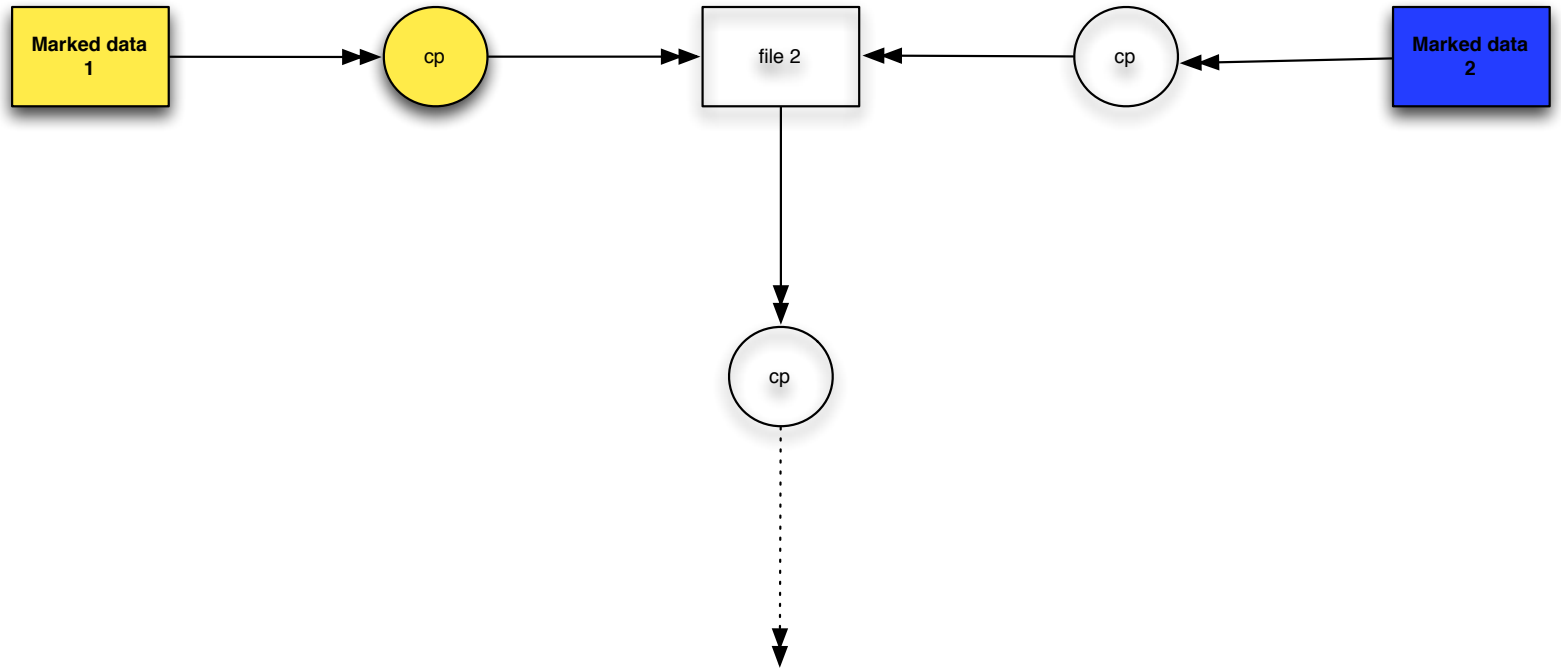
Blare : Comment ça marche ?

1. On attache une marque aux données à surveiller
2. Les marques sont propagées avec ces données



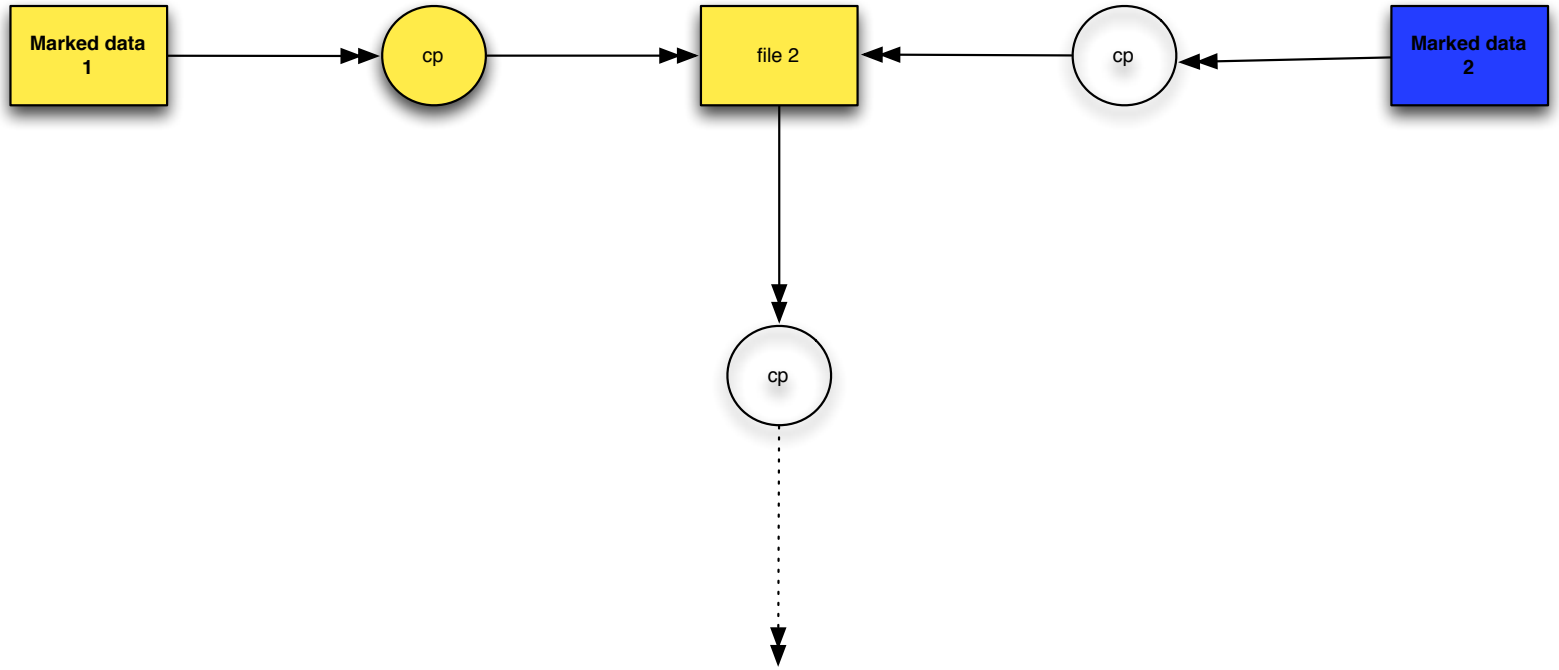
Blare : Comment ça marche ?

1. On attache une marque aux données à surveiller
2. Les marques sont propagées avec ces données



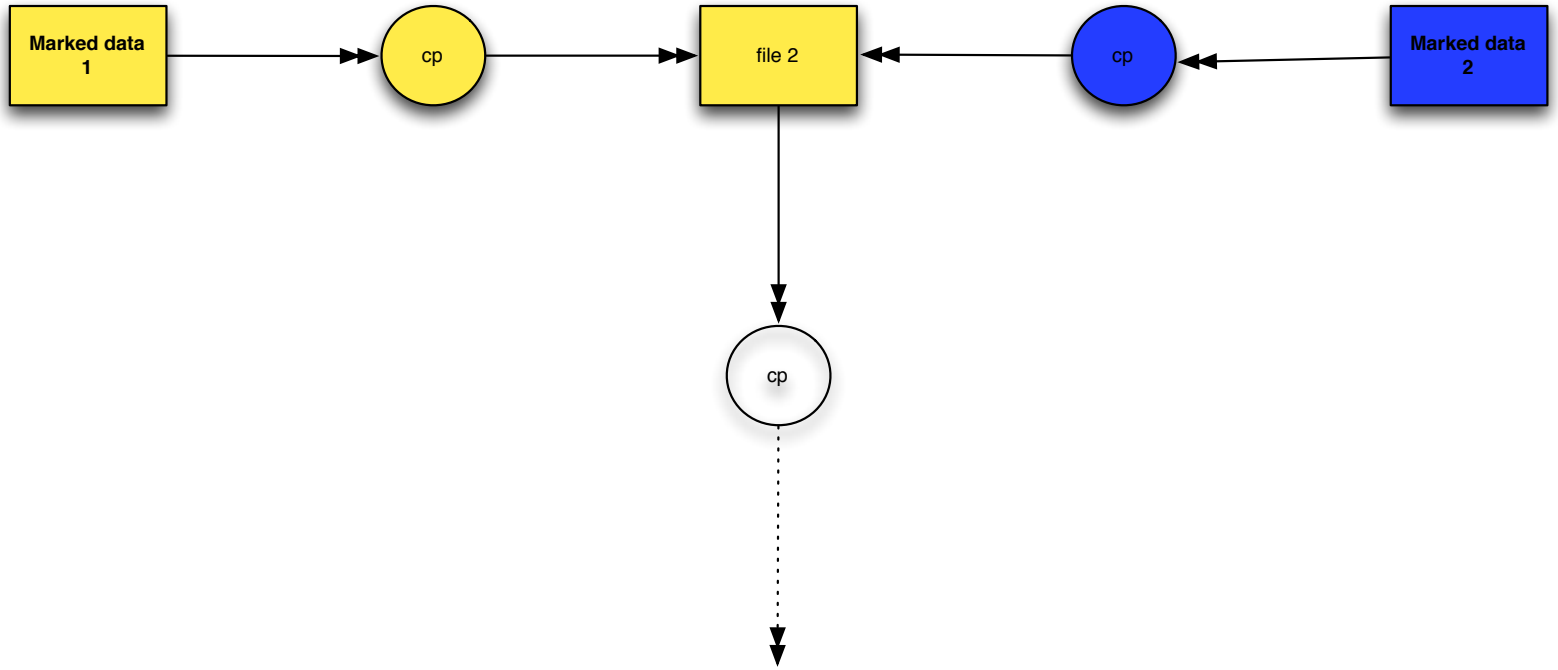
Blare : Comment ça marche ?

1. On attache une marque aux données à surveiller
2. Les marques sont propagées avec ces données



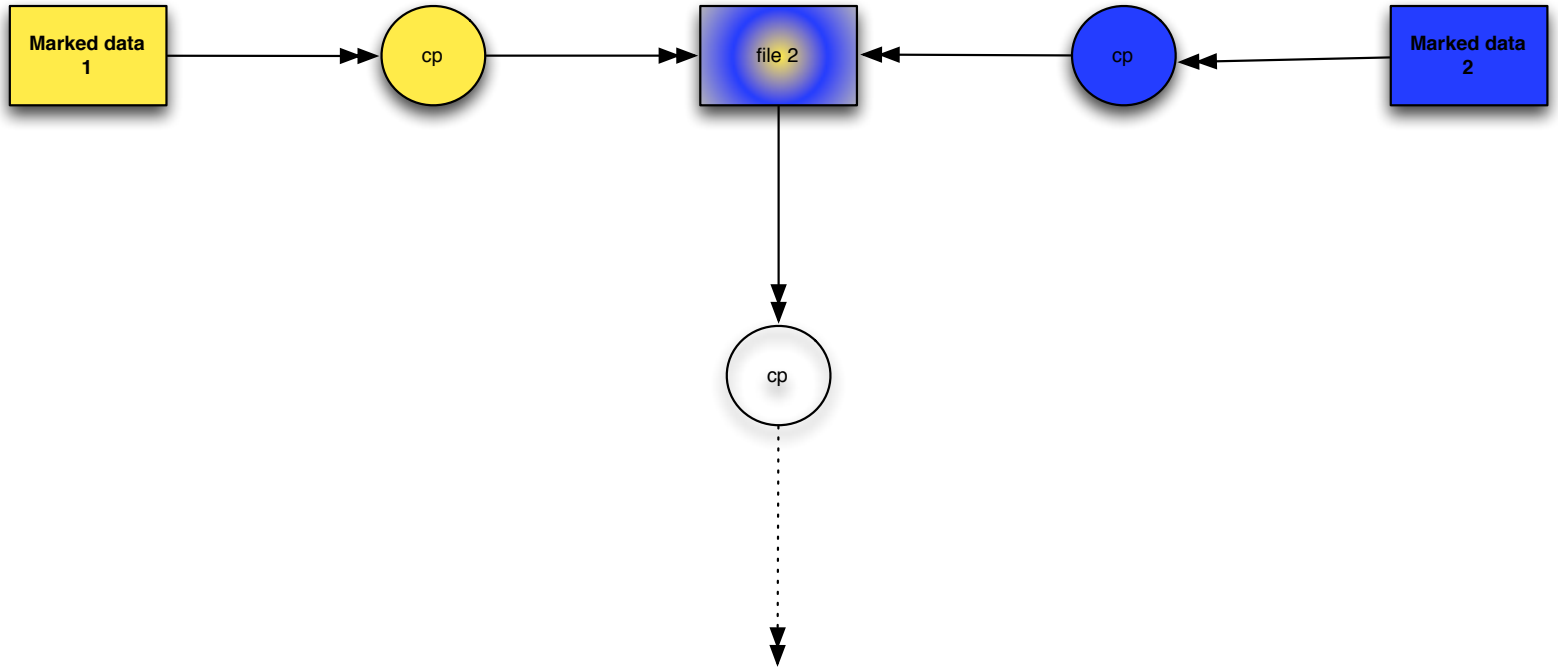
Blare : Comment ça marche ?

1. On attache une marque aux données à surveiller
2. Les marques sont propagées avec ces données



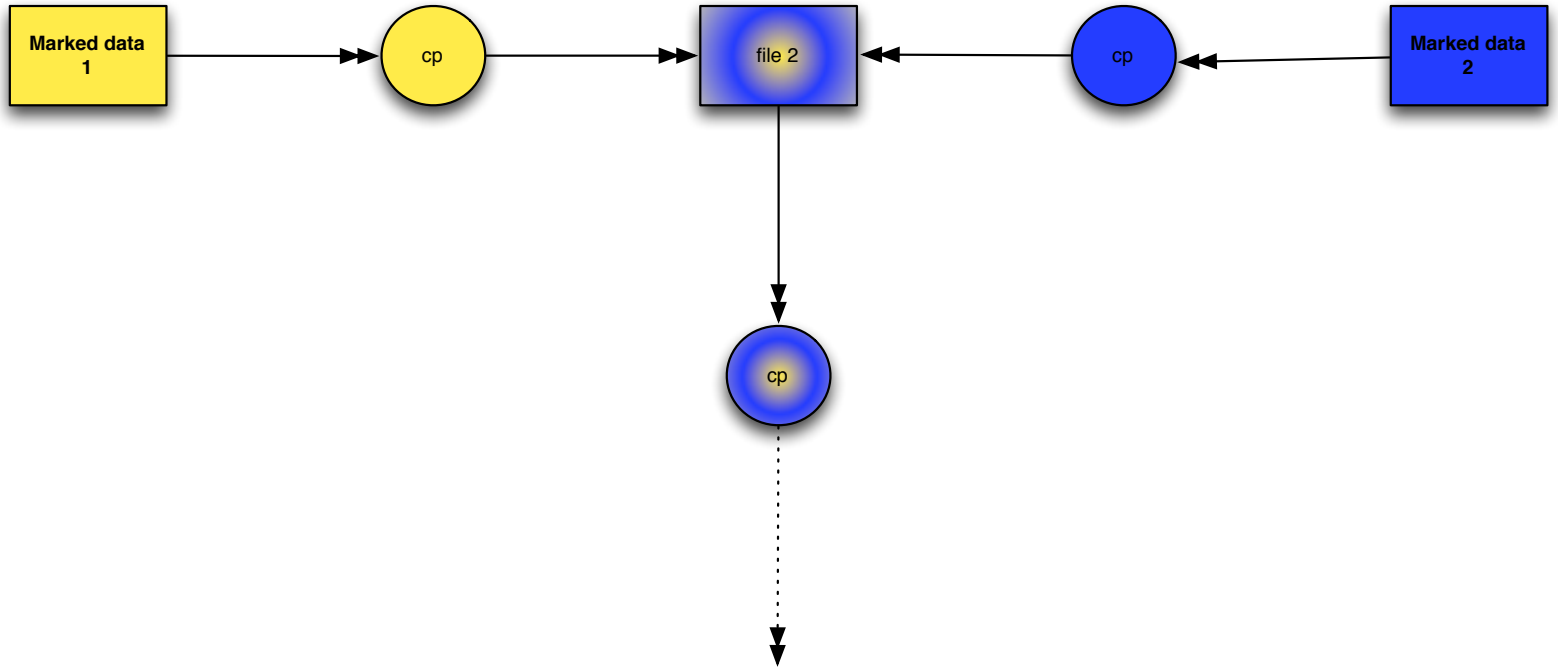
Blare : Comment ça marche ?

1. On attache une marque aux données à surveiller
2. Les marques sont propagées avec ces données



Blare : Comment ça marche ?

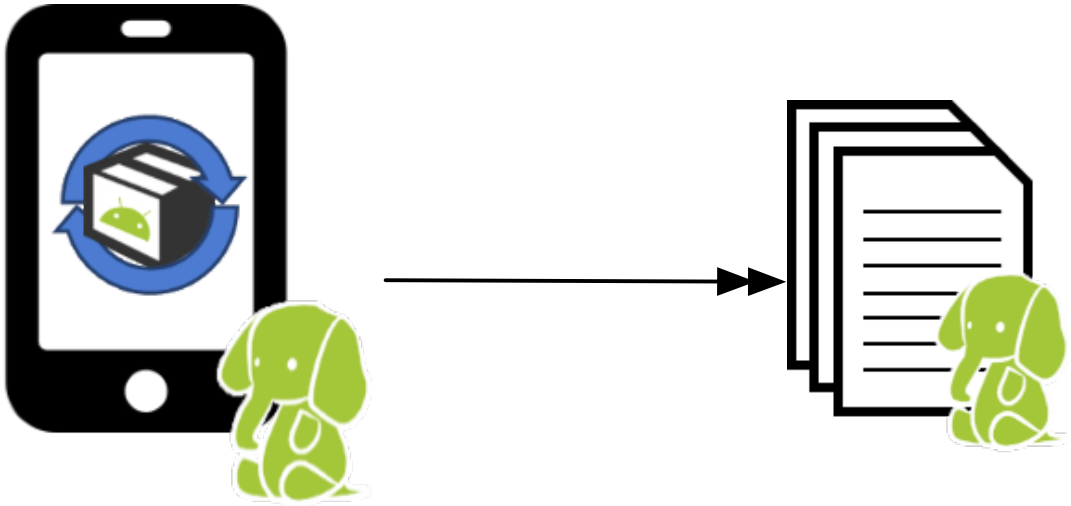
1. On attache une marque aux données à surveiller
2. Les marques sont propagées avec ces données



Capturer le comportement avec Blare

Blare produit des logs . . .

- qui sont exactement l'histoire de l'information dans le système
- . . . mais ces logs sont longs
- . . . mais ces logs sont redondants



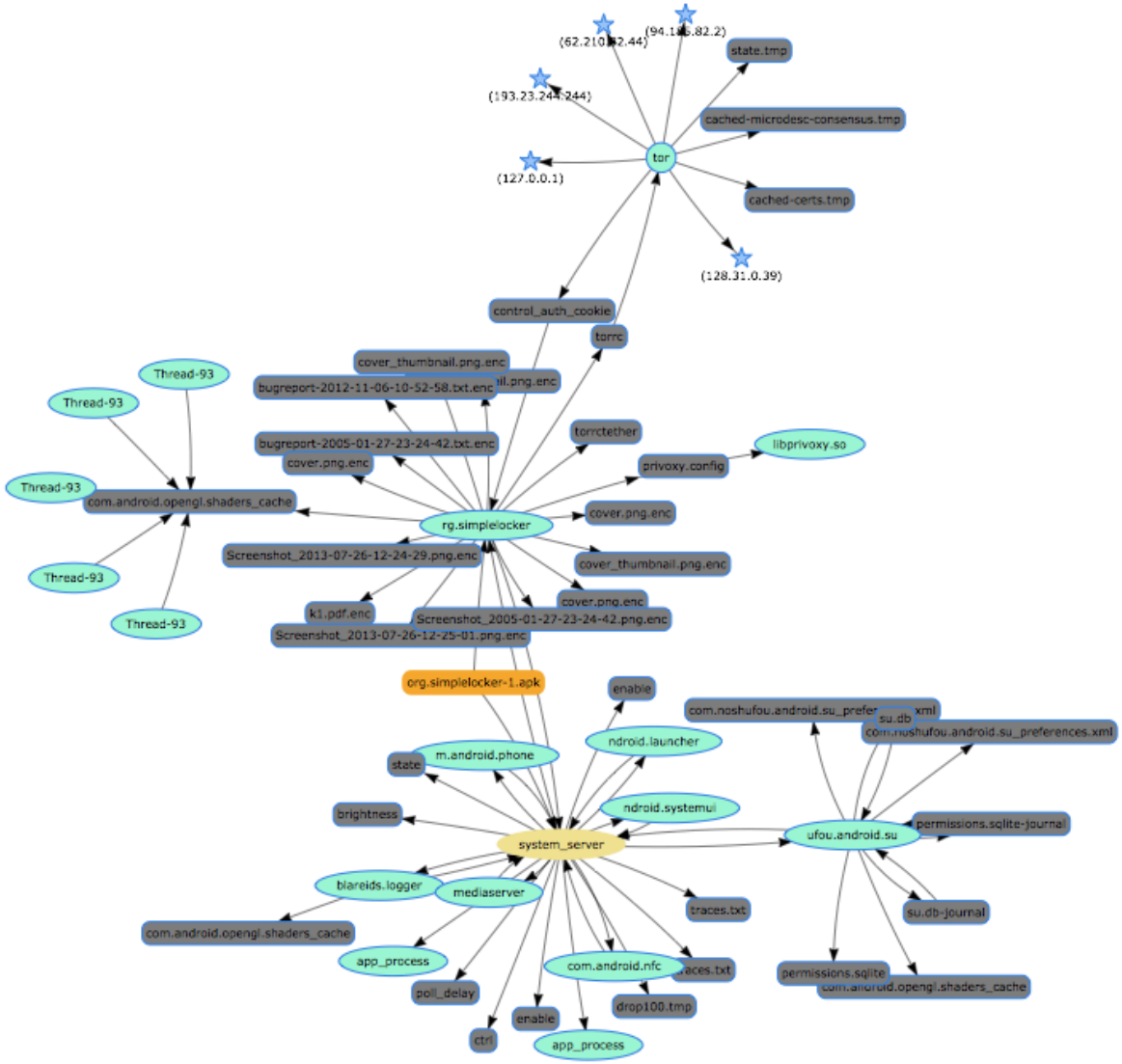
System Flow Graph : visualiser les logs Blare ..

Logs become graphs

- **Nœuds** sont des conteneurs d'information *Fichiers, Processus, Sockets*,
- **Arcs** indiquent des flux d'information

Les arcs et les nœuds sont étiquetés

- **nœuds** type, system id, nom
- **Arcs** données impliquées, timestamps


state.tmp
cached-microdesc-consensus.tmp
tor
cached-certs.tmp
(127.0.0.1)
(128.31.0.39)
control_auth_cookie
torrc
cover_thumbnail.png.enc
bugreport-2012-11-06-10-52-58.txt.enc
Thread-93
Thread-93
bugreport-2005-01-27-23-24-42.txt.enc
cover.png.enc
torrctether
libprivoxy.so
privoxy.config
Thread-93
com.android.opengl.shaders_cache
rg.simplelocker
cover.png.enc
Screenshot_2013-07-26-12-24-29.png.enc
cover_thumbnail.png.enc
Thread-93
Thread-93
k1.pdf.enc
cover.png.enc
Screenshot_2005-01-27-23-24-42.png.enc
Screenshot_2013-07-26-12-25-01.png.enc
org.simplelocker-1.apk
enable
com.noshufou.android.su_prefer xml
su.db
com.noshufou.android.su_preferences.xml
ndroid.launcher
state
m.android.phone
ndroid.systemui
brightness
system_server
ufou.android.su
permissions.sqlite-journal
blareids.logger
mediaserver
traces.txt
su.db-journal
com.android.opengl.shaders_cache
app_process
com.android.nfc
aces.txt
permissions.sqlite
com.android.opengl.shaders_cache
poll_delay
drop100.tmp
enable
ctrl
app_process

Un aperçu

Conclusion

La suite

- identifier statiquement du code suspicieux
- exécuter et surveiller ce code suspicieux
- être complet
- passer à l'échelle
- répertorier de nouveaux malware
- offrir un service en ligne