

# Research Presentation Exam

## **MADAM: Effective and Efficient Behavior-based Android Malware Detection and Prevention**

Andrea Saracino, Daniele Sgandurra, Gianluca Dini, and Fabio Martinelli

Qian Han

Department of Computer Science,  
Dartmouth College

May 14, 2019



DARTMOUTH

# Presentation Outline

- 1 Motivation
- 2 Problem
- 3 Contributions
- 4 MADAM Approach to Malware Detection
  - MADAM Malware Behavioral Classes
  - Multi-Level Behavior-Based Feature Analysis
- 5 MADAM Architecture
  - App Risk Assessment
- 6 Global Monitor
- 7 Per-App Monitor
- 8 User Interface and Prevention
- 9 MADAM Detection Procedure
- 10 Results
  - Malware Detection Results
  - Usability Analysis
- 11 Conclusion
- 12 Related Works
  - Static Analysis
  - Dynamic Analysis

Some Materials are copied from MADAM<sup>1</sup>

<sup>1</sup>Saracino et al., "Madam: Effective and efficient behavior-based android malware detection and prevention"



# Motivation behind MADAM

- ① Smartphones and tablets have become extremely popular (almost 7 billions worldwide at the end of 2014)
- ② Android has currently the largest market share, which is greater than 80%
- ③ Android is the main target of attacks against mobile devices (98.5%)
- ④ Malware constitutes a serious threat to user privacy, money, device and file integrity
- ⑤ *Google Play* has hosted apps which have been found to be malicious

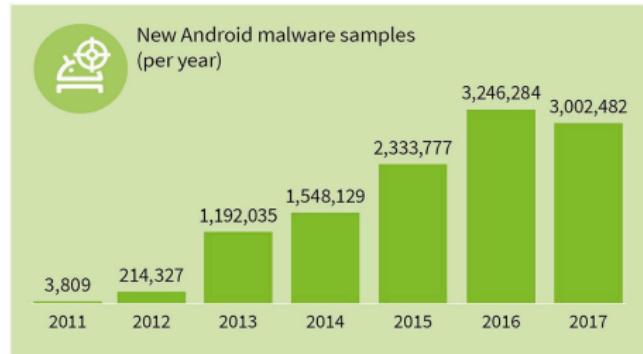
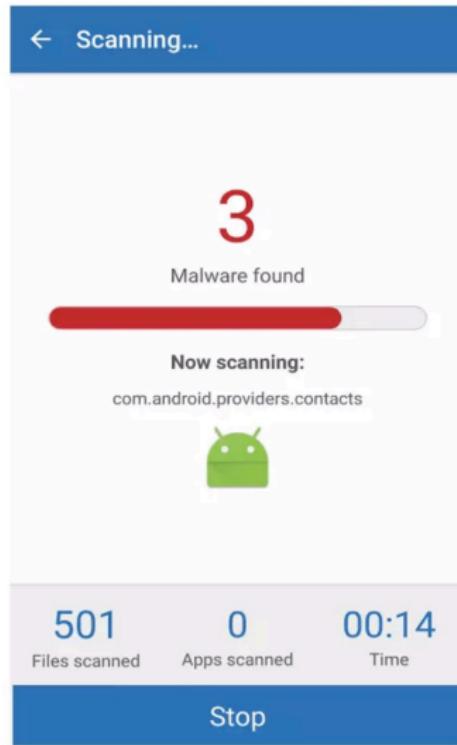


Figure Source<sup>2</sup>

<sup>2</sup> <https://www.gdatasoftware.com/blog/2017/04/29712-8-400-new-android-malware-samples-every-day>

# Problem addressed in the paper

How to detect Android malware  
efficiently and accurately?



DARTMOUTH

# Contributions

The contributions of MADAM:

- ① a behavior-based and multi-level malware detection system for Android devices
- ② a detection accuracy of 96.9% on a testbed of 2,784 malicious apps, divided in 125 families, spanning from 2010 to 2015
- ③ a False Positive Rate of  $2.8 \times 10^{-5}$  in normal usage conditions
- ④ a behavior-based taxonomy of existing Android malware into seven classes



DARTMOUTH

# MADAM Malware Behavioral Classes

- ① **Botnet:** malware that open a backdoor on the device
- ② **Rootkit:** malware that get super user (root) privileges on the device
- ③ **SMS Trojan:** malware that send SMS messages stealthily and without the user consent
- ④ **Spyware:** malware that take private data from the mobile device
- ⑤ **Installer:** malware that install new apps without authorizations
- ⑥ **Ransomware:** malware that prevent the user from interacting with the device
- ⑦ **Trojan:** any malware whose behavior is not considered by the previous classes



DARTMOUTH

# Multi-Level Behavior-Based Feature Analysis

- ① **system calls (level I, kernel-level)**: describing the device behavior at the lowest level
- ② **critical API (level II, application-level)**: performing operations which might be critical on the security side
- ③ **user activity (level III, user-level)**: understanding when the user is interacting with the device
- ④ **SMS features (level III, user-level)**: detecting misbehaviors related to Spyware and Botnet
- ⑤ **app metadata (level IV, package-level)**: permissions declared by apps, rating, marketplace and download number



DARTMOUTH

# Multi-Level Behavior-Based Feature Analysis

Level	Group	Feature	Description	Targeted Misbehavior
Kernel	Sys Calls	open, read, ...	System calls concerning file and inter-component communication	Sudden and unmotivated activity increase
Application	SMS	Number of SMS (SMS Num)	Amount and recipient of outgoing SMS	Unsolicited outgoing messages
Application	SMS	Suspicious SMS (SMS Susp)	Amount of SMS sent to recipients not in contacts	Spyware or registration to premium services
Application	Critical API	Administrator App	Verify if an app attempts to get admin privileges	Apps which attempt to take control of the device
Application	Critical API	New App Installation	Verify if an app attempts to install a new one	Unauthorized app installations
Application	Critical API	Process List	Verify if an app generates high number of processes	Buffer overflow (Rootkit) attacks
Application	Critical API	Critical SysCalls	Amount of critical system calls generated by an app	Apps that access files and resources in background (Spyware, Botnet and Trojan)
Application	Critical API	SMS Default App	Check default SMS manager	Unsolicited outgoing SMS
Application	Critical API	Foreground App	Check which app is interacting with user	Unsolicited SMS and preventing user from interacting with the device (Ransomware)
User	User Activity	User Presence	If the user is interacting with the device	Unsolicited activities of Spyware, Botnet, Installer and Rootkit
User	User Activity	On Call	Verify if a phone call is ongoing	Unsolicited activities of Spyware, Botnet, Installer and Rootkit
User	User Activity	Screen On	Verify if the device screen is on	Unsolicited activities of Spyware, Botnet, Installer and Rootkit
Package	App Metadata	Permissions requested (manifest.xml)	Riskiness of app	Suspicious requests of dangerous permissions
Package	App Metadata	Market Info (User scores, ...)	Popularity of app	Trojan

Figure: MADAM Levels of Analysis and Features



# MADAM Architecture Overview

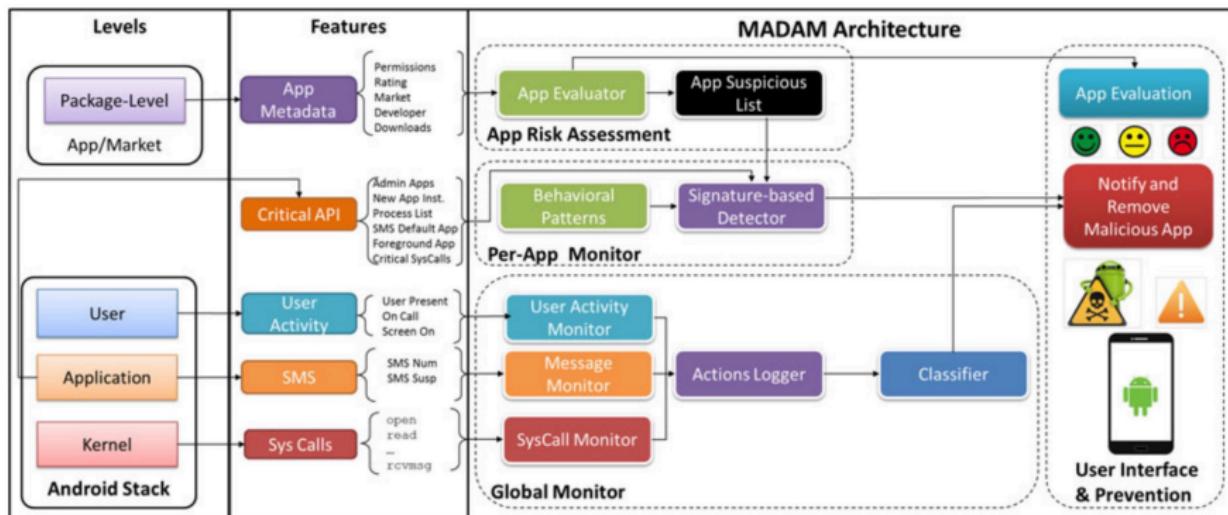


Figure: Architecture of MADAM



# App Risk Assessment

App Risk Assessment: includes the App Evaluator that analyzes metadata of an app package

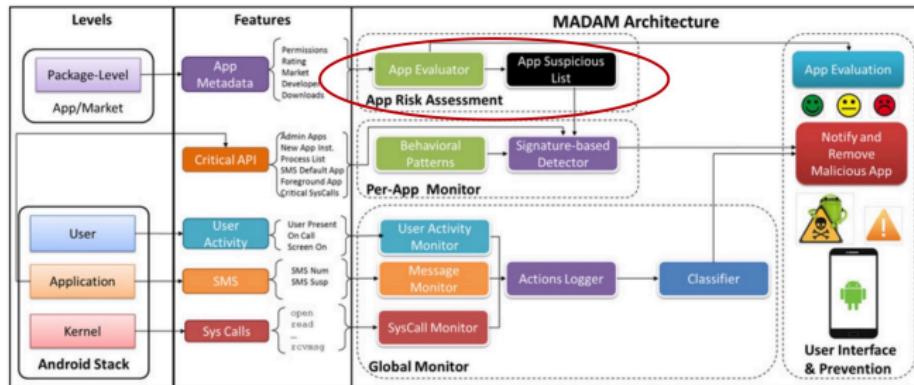


Figure: App Risk Assessment Module



# Global Monitor

Global Monitor: core of the MADAM framework, monitors the device and OS features at kernel, user and application level

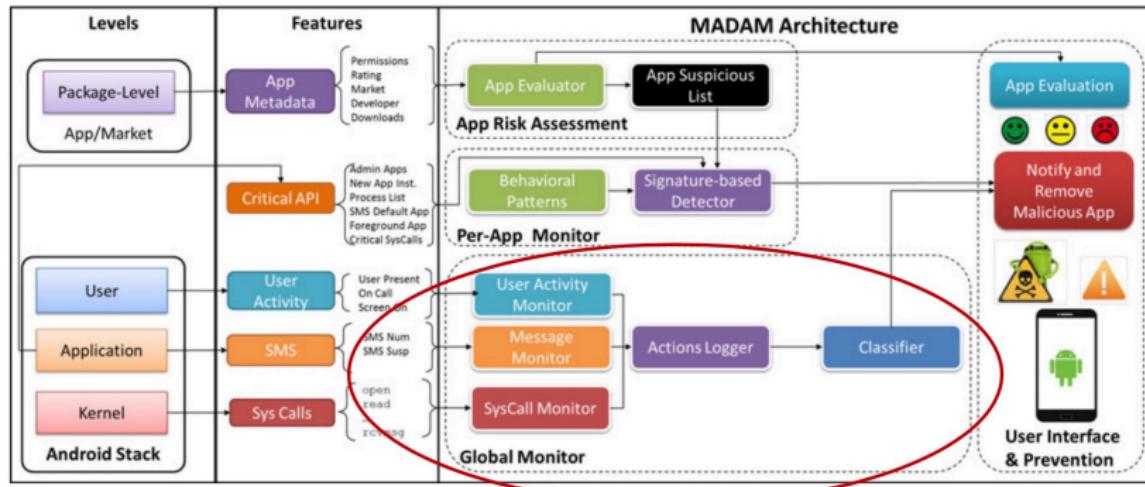


Figure: Global Monitor Module



# System Call Monitor

open	ioctl	brk	read	write	exit	close	sendto	sendmsg	recvfrom	recvmsg	Idleness	SMS Num	SMS Susp
6	19	18	1	4	0	7	16	2	2	0	0	0	0
147	652	192	711	4	282	229	7	15	7	13	1	0	0

Figure: System Calls in MADAM

- System Call Monitor intercepts the system calls (first eleven columns)
- related to file operations and network access
- operations by malware are translated as operations on files at low-level



# User Activity and Message Monitor

open	ioctl	brk	read	write	exit	close	sendto	sendmsg	recvfrom	recvmsg	Idleness	SMS Num	SMS Susp
6	19	18	1	4	0	7	16	2	2	0	0	0	0
147	652	192	711	4	282	229	7	15	7	13	1	0	0

Figure: Device Idleness and SMS Related Features

- User Activity and Message Monitor intercept calls to security relevant APIs related to SMS and user activity
- hijacking `SendTextMessage()` and `SendDataMessage()` to control the outgoing SMS messages
- categorize the smartphone's status of active or idle



# Action Logger and Classifier

open	ioctl	brk	read	write	exit	close	sendto	sendmsg	recvfrom	recvmsg	Idleness	SMS Num	SMS Susp
6	19	18	1	4	0	7	16	2	2	0	0	0	0
147	652	192	711	4	282	229	7	15	7	13	1	0	0

Figure: Comparison of Behavior Vectors: User Idle (Top) vs User Active (Bottom)

- monitored system calls are represented through a feature vector
- two parallel classifiers:
  - short-term classifier: detecting sudden and sharp increase of the system call occurrences (Rootkits)
  - long-term classifier: actions constantly in a long period of time (Spyware)
- anomaly detection system: real genuine behaviors and synthetic malicious behaviors as training data
- K-Nearest Neighbor (euclidean distance) with K = 1 achieving max accuracy



DARTMOUTH

# Per-App Monitor

Per-App Monitor: monitors the actions performed by suspicious apps through known behavioral patterns

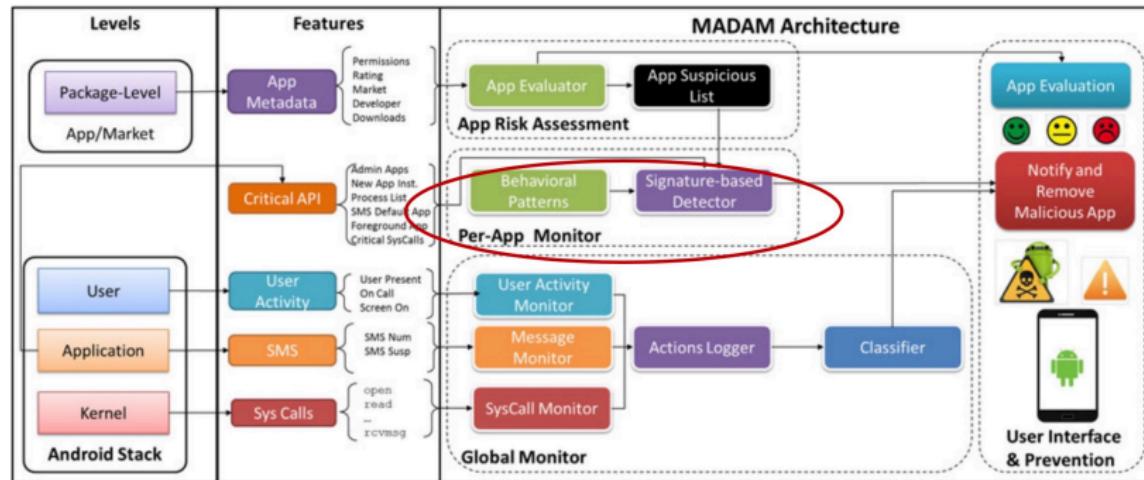


Figure: Per-App Monitor Module



# Malicious Behavioral Patterns

- ① Text messages sent by a non-default message app: *SMS Trojans, Botnet, Spyware*
- ② Text messages sent to numbers not in the user contact list: *SMS Trojans, Botnet, Spyware*
- ③ High number of outgoing message per period of time: *SMS Trojans*
- ④ High number of process per app: *Rootkit*
- ⑤ Excessive foreground time for non interacting and administrator app: *Ransomware*
- ⑥ Unauthorized installation of new apps: *Installers*
- ⑦ Unsolicited kernel level activity of background app: *Botnet, Spyware and some generic Trojans*



DARTMOUTH

# User Interface and Prevention

User Interface and Prevention: stops malicious actions and handles the procedure for removing malware

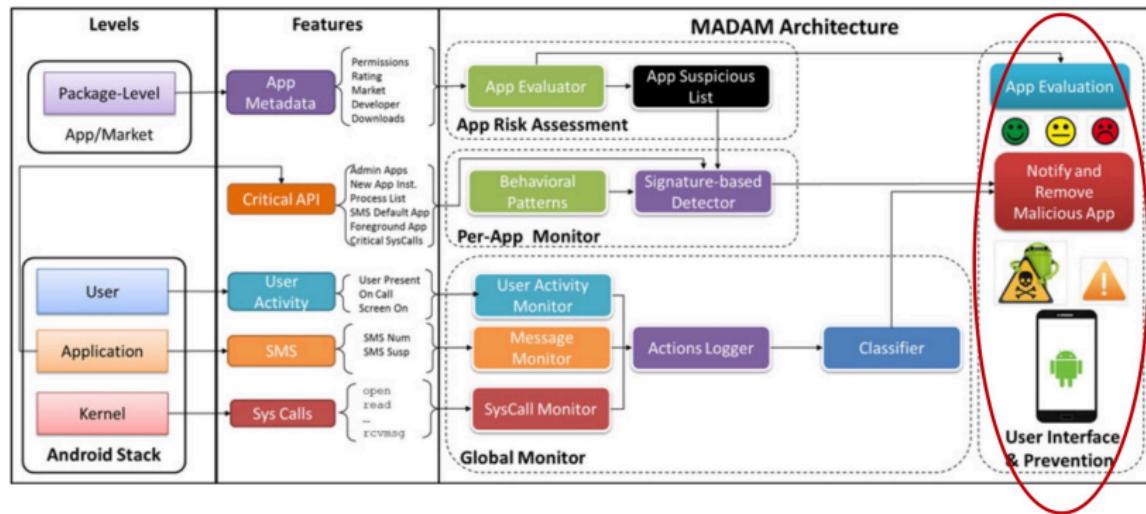


Figure: User Interface and Prevention Module



# MADAM Detection Procedure

- ① App Evaluator is launched in background, to assess new app's risk at deploy-time
- ② Global Monitor is launched in background, to retrieve 14 features and classify the behaviors
- ③ In parallel, the Per-App Monitor is launched, to monitor kernel and API features to detect known behavioral patterns
- ④ Behavioral patterns are checked in background by the Signature-based Detector
- ⑤ Per-App Monitor blocks the misbehavior of apps in the Suspicious List
- ⑥ User Interface & Prevention module kills the malicious app and proposes the user to remove it



# Correlating Features and Misbehaviors

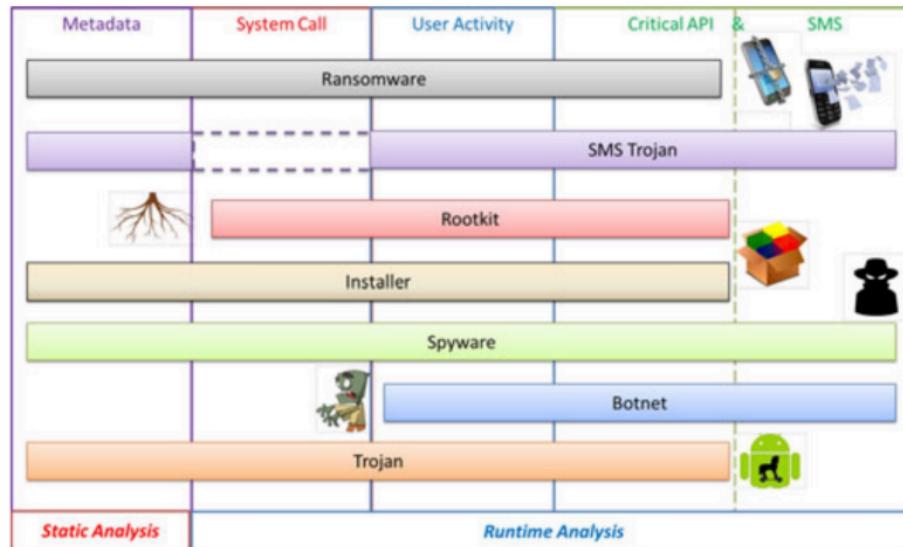


Figure: Relevant features for the detection of the seven malware behavioral classes



DARTMOUTH

# Datasets

About 2,800 applications from 125 different families:

- Genome<sup>3</sup>: 1,242 malicious Android apps collected in 2010 and 2011 in 49 malware families
- Contagio Mobile<sup>4</sup>: 18 malware families since 2012, only few samples (generally one) for each family
- VirusShare<sup>5</sup>: 1,923 malicious apps in 90 malware families, spanning from 2012 to 2015.

---

<sup>3</sup>Zhou and Jiang, "Dissecting android malware: Characterization and evolution"

<sup>4</sup><http://contagiominidump.blogspot.com/>

<sup>5</sup><https://virusshare.com/>



# Malware Detection Results

Malware	Class	Year	Samples	MADAM	VirusTotal					
Lemon	Botnet	2012	6	0	6	Spyoo	Spyware	2012	3	3
MmarketPay	Botnet	2012	1	0	1	Tesbo	Spyware	2012	1	0
Basebridge	Installer	2011	330	330	330	Trackplus	Spyware	2014	6	0
CrWind	Installer	2015	1	1	1	Typstu	Spyware	2011	14	14
FakeFlash	Installer	2012	3	3	3	Vdloader	Spyware	2011	16	16
Gamex	Installer	2012	7	7	7	Walkiwat	Spyware	2011	1	1
Gapev	Installer	2012	7	7	7	Ycchar	Spyware	2012	2	0
Gappusin	Installer	2012	58	58	0	Ksapp	Spyware + Installer	2012	6	6
Anasca	Trojan	2011	1	0	1	DroidDream	Spyware + Rootkit	2011	16	16
Antares	Trojan	2012	2	0	2	Gmuse	Spyware + Rootkit	2014	3	3
Faketime	Trojan	2012	16	16	16	zHash	Spyware + Rootkit	2011	11	11
Fujacks	Trojan	2013	1	0	0	Dabom	SMS Trojan + Installer	2014	2	2
Moghava	Trojan	2012	3	3	3	Updtkiller	SMS Trojan + Installer	2012	1	1
Copycat	Ransomware	2014	10	10	10	Bosm.C	SMS Trojan + Installer	2015	1	1
FoCober	Ransomware	2015	13	13	13	Boxer	SMS Trojan + Installer	2011	27	27
Koler.C	Ransomware	2014	7	7	7	Cawitt	SMS Trojan + Spyware	2012	1	1
AsRoot	Rootkit	2011	8	8	8	Cosha	SMS Trojan + Spyware	2012	10	10
Coogos	Rootkit	2014	8	8	8	Fjcon	SMS Trojan + Spyware	2012	4	4
Droidrooter	Rootkit	2011	3	3	3	MobileTx	SMS Trojan + Spyware	2012	69	69
DroidCoupon	Rootkit	2011	1	1	1	Nandrobox	SMS Trojan + Spyware	2012	13	13
DroidKungFu	Rootkit	2011	402	402	402	Geinimi	SMS Trojan + Botnet	2011	69	69
<b>Total</b>						<b>Total</b>	-	-	<b>2,784</b>	<b>2,700</b>
<b>Accuracy</b>						<b>Accuracy</b>	-	-	-	<b>96.9%</b>

## Detection Results and Comparisons (Partial Results)<sup>6</sup>

<sup>6</sup>VirusTotal: a web-service for malware static analysis, including 50-60 well known anti-virus software and using the majority voting approach



DARTMOUTH

# Malware Detection Results

- correctly detected 2,700 samples out of 2,784, showing an accuracy of 96.9%
- able to detect 9 malware families which evade VirusTotal checks
- more accurate against low profile malware whose signature is known



DARTMOUTH

# Usability Analysis: False Positives

Test	FPs	FPR	FPs/day
<i>Light</i>	3	$1 \cdot 10^{-5}$	0.5
<i>Medium</i>	8	$2.8 \cdot 10^{-5}$	1.1
<i>Heavy</i>	75	$2.6 \cdot 10^{-4}$	10.7

Figure: False Alarms Experimental Results

- Length of Experiments: 1 week, every day from 10:00 to 21:00
- Light: performing/receiving phone calls and/ or sending/receiving text messages from the default app
- Medium: Light + accessing the Internet, using three instant messaging apps, playing 2D games and taking pictures
- Heavy: Medium + screen always on, new legitimate apps and recording video



DARTMOUTH

# Usability Analysis: Performance Overhead

Test	Vanilla	MADAM	Overhead
Total	2,911	2,868	1,4%
CPU	5,509	5,459	0,9%
Memory	2,660	2,409	9,4%
I/O	3,860	3,705	4%
2D	327	327	0%
3D	2,250	2,250	0 %

Figure: Benchmark Tests

- measured through a standard benchmark tool: Quadrant Standard Edition<sup>7</sup>
- higher value means a better performance
- overhead is caused by the *kernel module* and the *Global Monitor*

<sup>7</sup>Quadrant Standard Edition is a CPU, I/O and 3D graphics benchmark. The Standard Edition requires an Internet connection to compute benchmark results and is supported by ads



# Conclusion

This paper proposes MADAM:

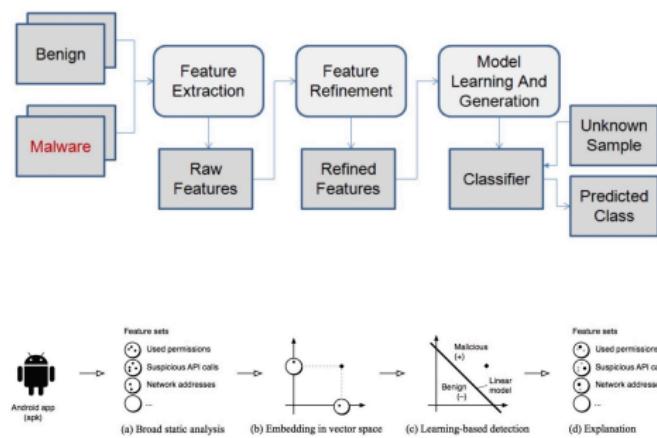
- ① a multi-level host-based Android malware detector
- ② analyzing and correlating features at four different Android levels
- ③ able to detect misbehaviors from malware behavioral classes that consider 125 existing malware families
- ④ first system that detects malware at run-time using a behavior-based and multi-level approach



DARTMOUTH

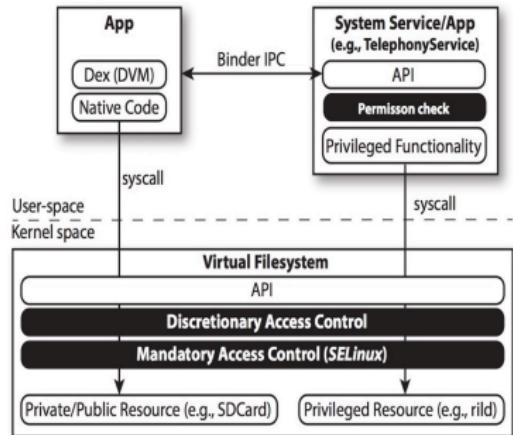
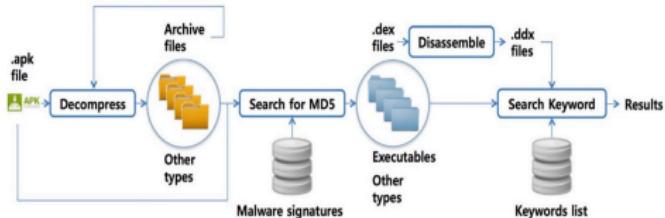
# Related Works: Static Analysis

- Gascon et al., "Structural detection of android malware using embedded call graphs" (2013)
- Aafer, Du, and Yin, "Droidapiminer: Mining api-level features for robust malware detection in android" (2013)  
→
- Gates et al., "Effective risk communication for android apps" (2014)
- Arp et al., "Drebin: Effective and explainable detection of android malware in your pocket." (2014) →
- Suarez-Tangil et al., "Thwarting obfuscated malware via differential fault analysis" (2014)



# Related Works: Dynamic Analysis

- Seo et al., "Detecting mobile malware threats to homeland security through static analysis" (2014) →
- Backes et al., "Android Security Framework: Extensible multi-layered access control on Android" (2014) →
- Sun et al., "Design and implementation of an android host-based intrusion prevention system" (2014)
- Enck et al., "TaintDroid: an information-flow tracking system for realtime privacy monitoring on smartphones" (2014)



Thank You!  
Questions?



# Appendix: Related Works

System	Dynamic/Static	Rooting	Detection Rate	Overhead	Attack
MADAM	Both	Yes	96.9%	1.4%	Several Classes of Attacks
TaintDroid [29]	Dynamic	Custom-ROM	N.A.	14%	Privacy Leak
Patronus [30]	Dynamic	Yes	87%	7.1%	SMS - Spyware
DroidAnalyzer [31]	Static	Offline	N.A.	N.A.	Rootkits
DroidSIFT [32]	Static	Offline (Server)	93%	N.A.	General
AlterDroid [33]	Static	Offline	97%	N.A.	Obfuscated Malware
ASF [34]	Dynamic	Custom ROM	N.A.	2-3 %	Access Control

Figure: Comparison of MADAM Results with Existing Frameworks

**System call table overriding:** We need a toolchain (set of programs to cross-compile the sources) as well as the appropriate version of the kernel sources. Once the sources are downloaded, create the default kernel config with the command make hammerhead defconfig. To enable system call hooking, recommend adding loadable module support, exposing interface, and exporting the global kernel symbols.

**Intent Filters:** used to call an another activity. Android OS uses filters to pinpoint the set of Activities, Services, and Broadcast receivers to help specified set of data scheme



## Appendix: Related Works

**Patronus:** a system for Android that can prevent mobile malware intrusions and detect malware at run-time. Patronus implements API hijacking to the binder at client and server side, to overcome the malware bypassing. The authors report a total overhead of their tool of 7.1 percent, while MADAM is 1.4 percent.

**TaintDroid:** a security framework for Android devices which tracks information flow to avoid malicious stealing of sensitive information. Differently from MADAM, TaintDroid targets a very specific class of attacks. Moreover, TaintDroid requires a custom ROM of the Android system, to implement the information flow mechanisms.

**buffer overflow:** occurs when more data is put into a fixed-length buffer than the buffer can handle. The extra information, which has to go somewhere, can overflow into adjacent memory space, corrupting or overwriting the data held in that space.



DARTMOUTH