Beijing Forest Studio
北京理工大学信息系统及安全对抗实验中心

基于关联行为分析的android恶意软件检测方法

张寒青　硕士

2018年11月11日

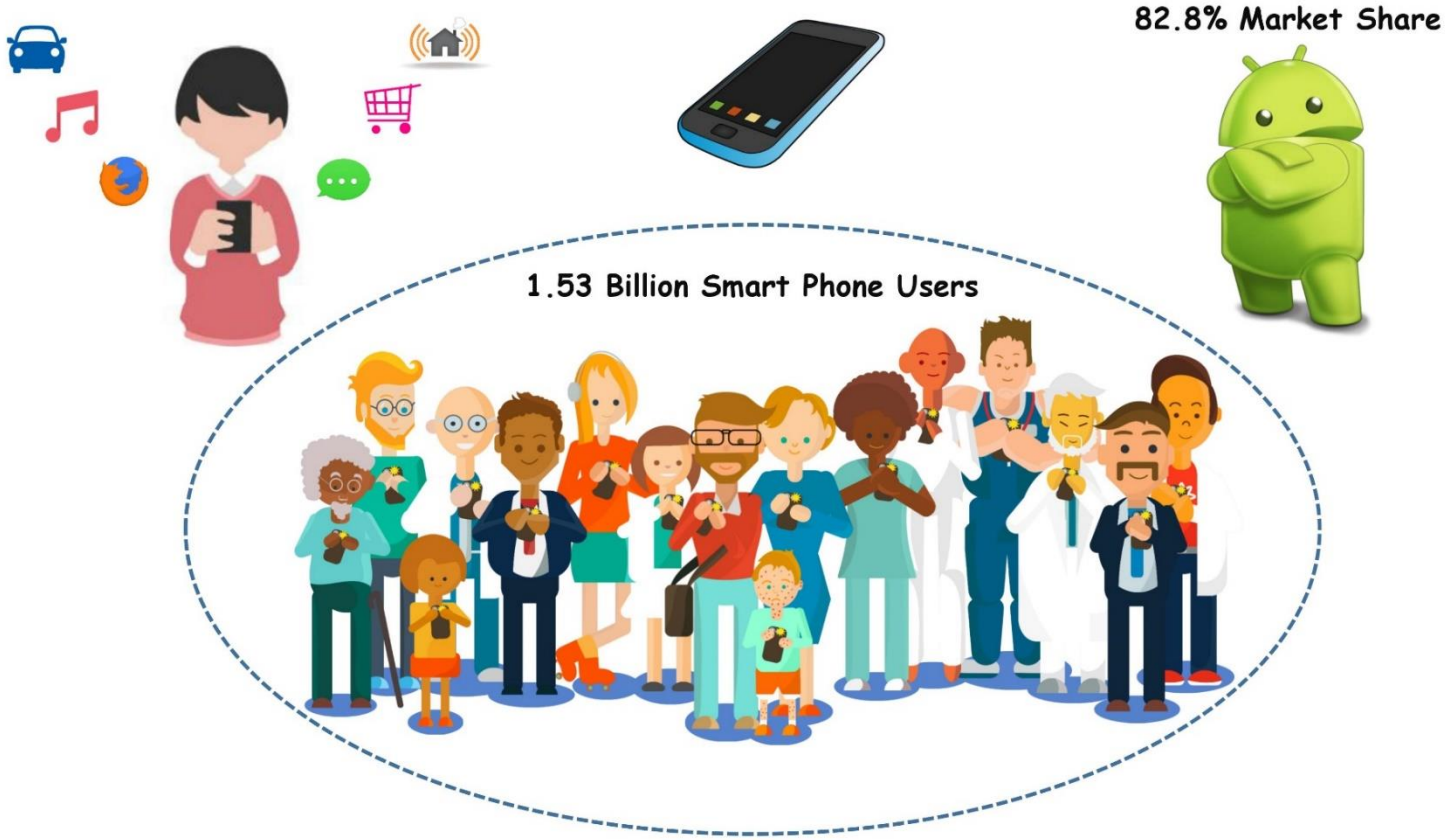
内容提要

- 背景简介
- 基本概念
- 框架原理
- 实验分析
- 优劣分析
- 参考文献

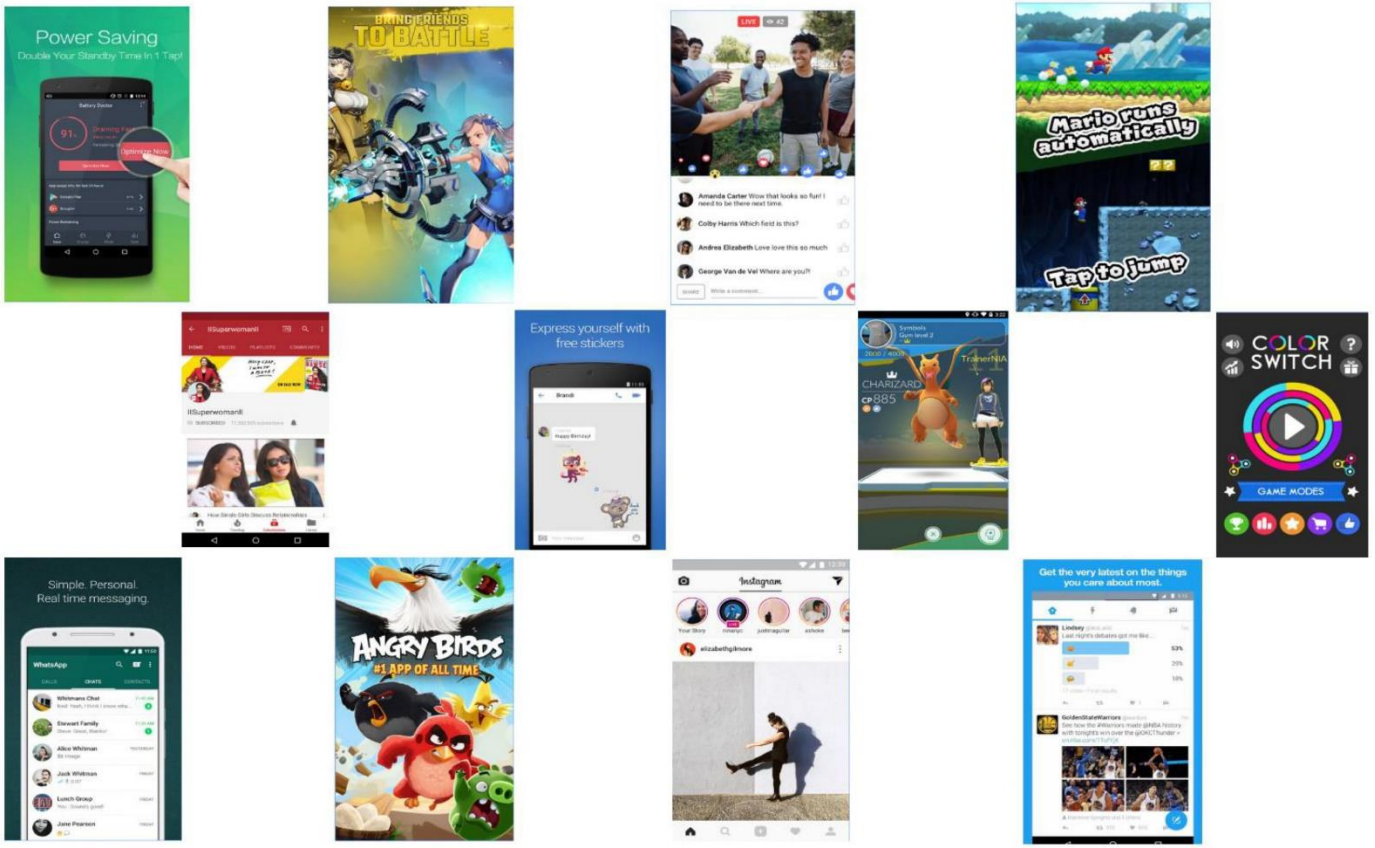
背景简介

- 预期收获
  - 1.了解数据挖掘技术在恶意软件分析上的应用
  - 2.了解数据挖掘解决问题的基本过程

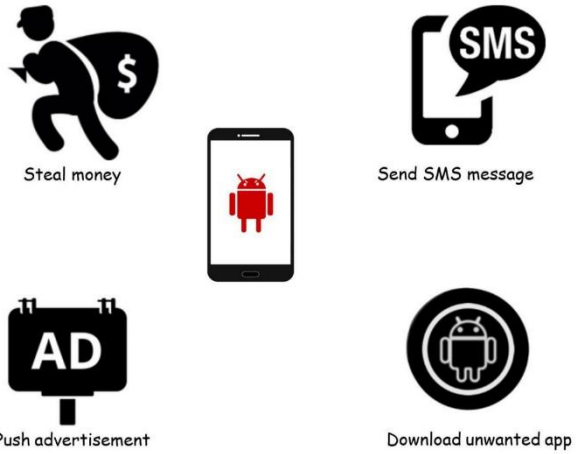
背景简介

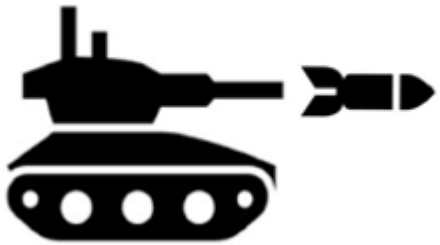


背景简介


Power Saving
Double Your Standby Time In 1 Tap!
BRING FRIENDS TO BATTLE
Mario runs automatically
Tap to jump
Express yourself with free stickers
COLOR SWITCH
GAME MODES
Simple. Personal. Real time messaging.
ANGRY BIRDS
#1 APP OF ALL TIME
Instagram
Get the very latest on the things you care about most.

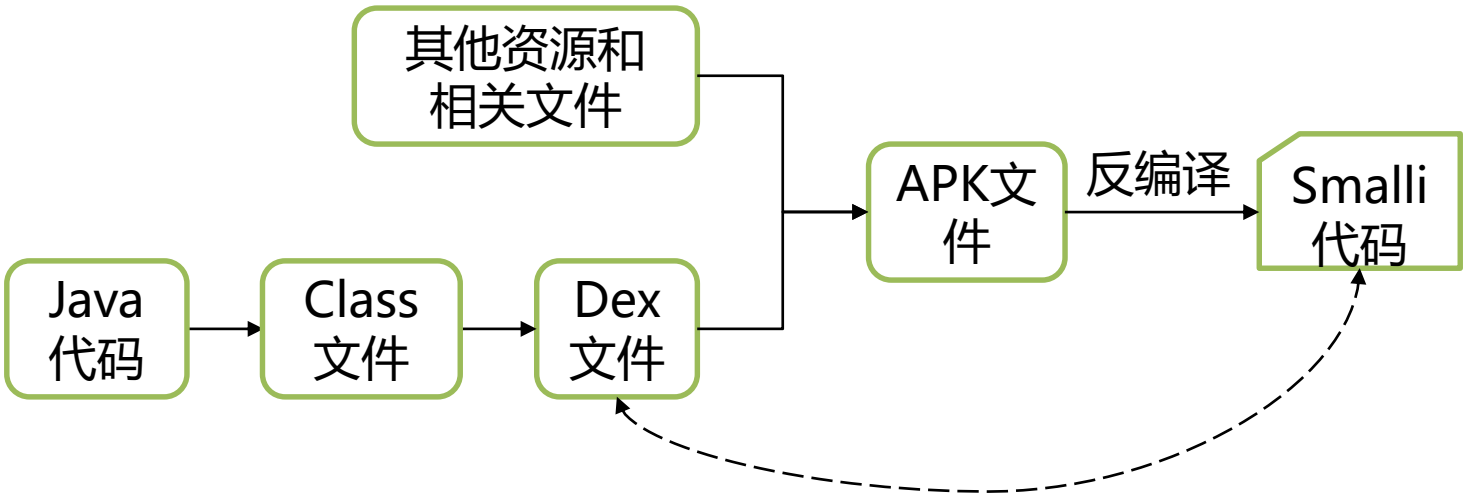
背景简介


$
Steal money
SMS
Send SMS message
AD
Push advertisement
Download unwanted app



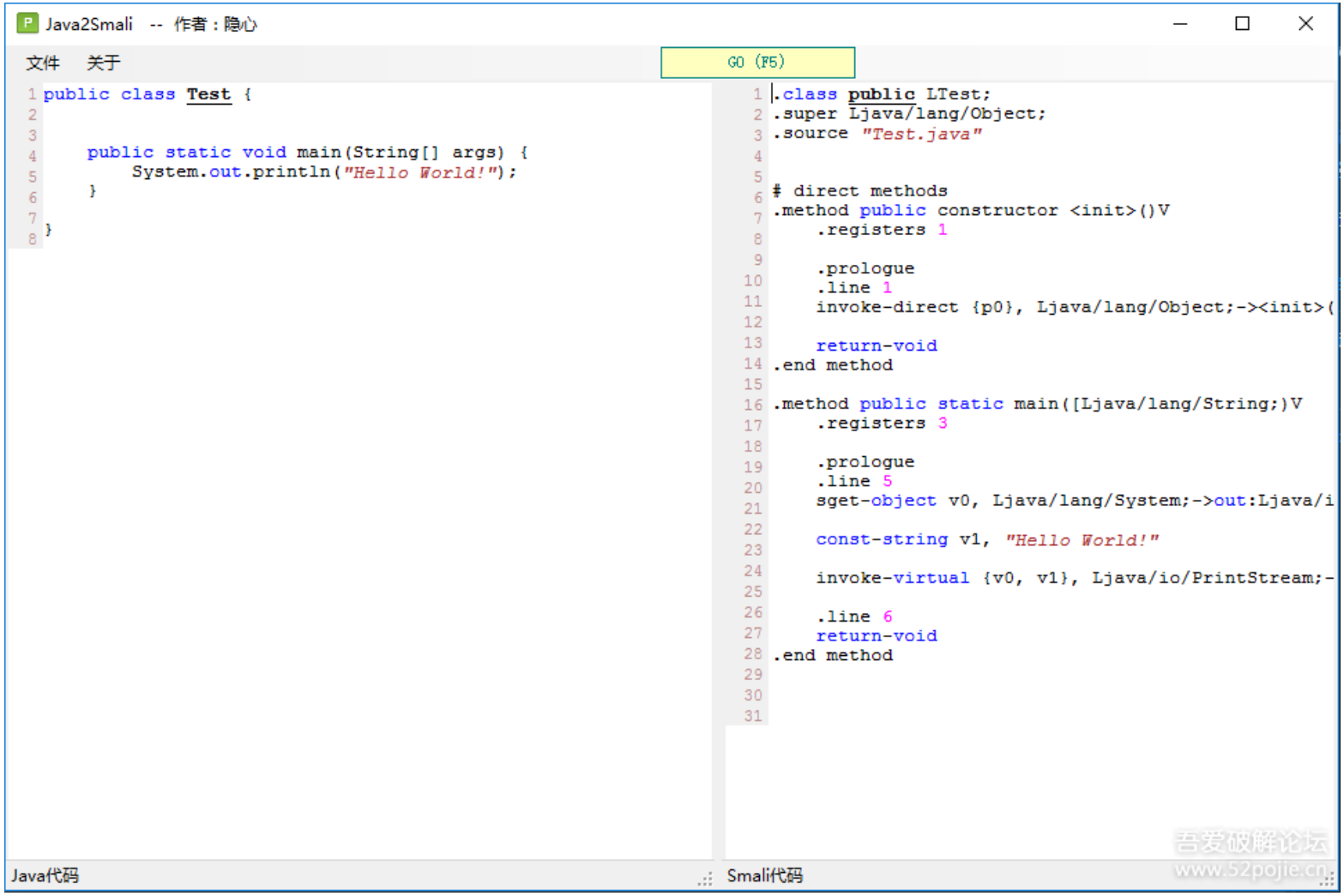
基本概念

- Smali语言
  - Dalvik虚拟机的寄存器语言
  - Dex文件反编译后得到Smalli代码



基本概念

- Smalli代码实例



基本概念

- 关联分析
  - 一种在大规模数据集中寻找有趣关系的非监督学习算法
  - 事务

    事务库中的每一条记录被称为一笔事务
  - 项集

    包含0个或者多个项的集合称为项集

	Items
1	面包，牛奶
2	面包，尿布，啤酒，鸡蛋
3	牛奶，尿布，啤酒，蛋糕
4	面包，牛奶，尿布，啤酒

关联规则分析相关算法介绍-李筱雅-2018-10.01.ppt

基本概念

- 关联分析
  - 关联规则

    两个不相交项集之间的蕴含表达式

    {牛奶，面包}->{尿布}、{啤酒}->{牛奶}
  - 置信度

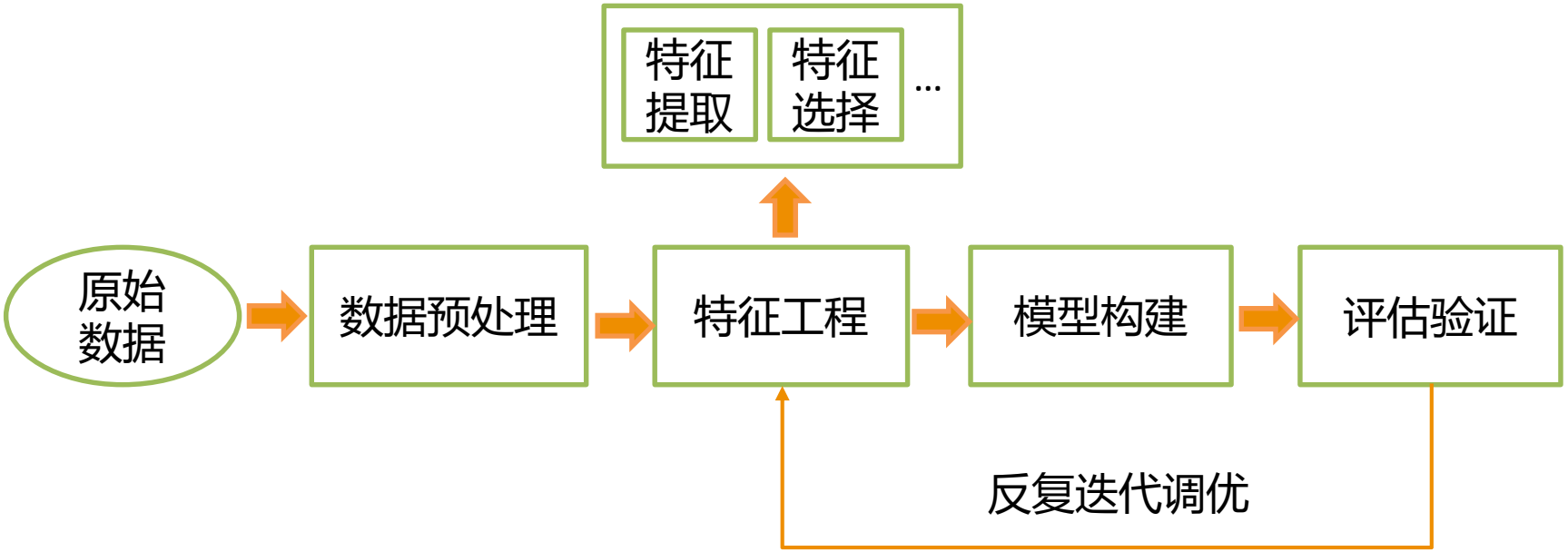
    规则A出现时，规则B一定出现的信服水平

$$N(X) = |t_i | X \in t_i, t_i \in T| \quad C(A \rightarrow B) = \frac{N(A\&B)}{N(A)}$$

	Items
1	面包，牛奶
2	面包，尿布，啤酒，鸡蛋
3	牛奶，尿布，啤酒，蛋糕
4	面包，牛奶，尿布，啤酒

基本概念

- 数据挖掘流程
  - 数据预处理，特征工程，模型构建，模型评估



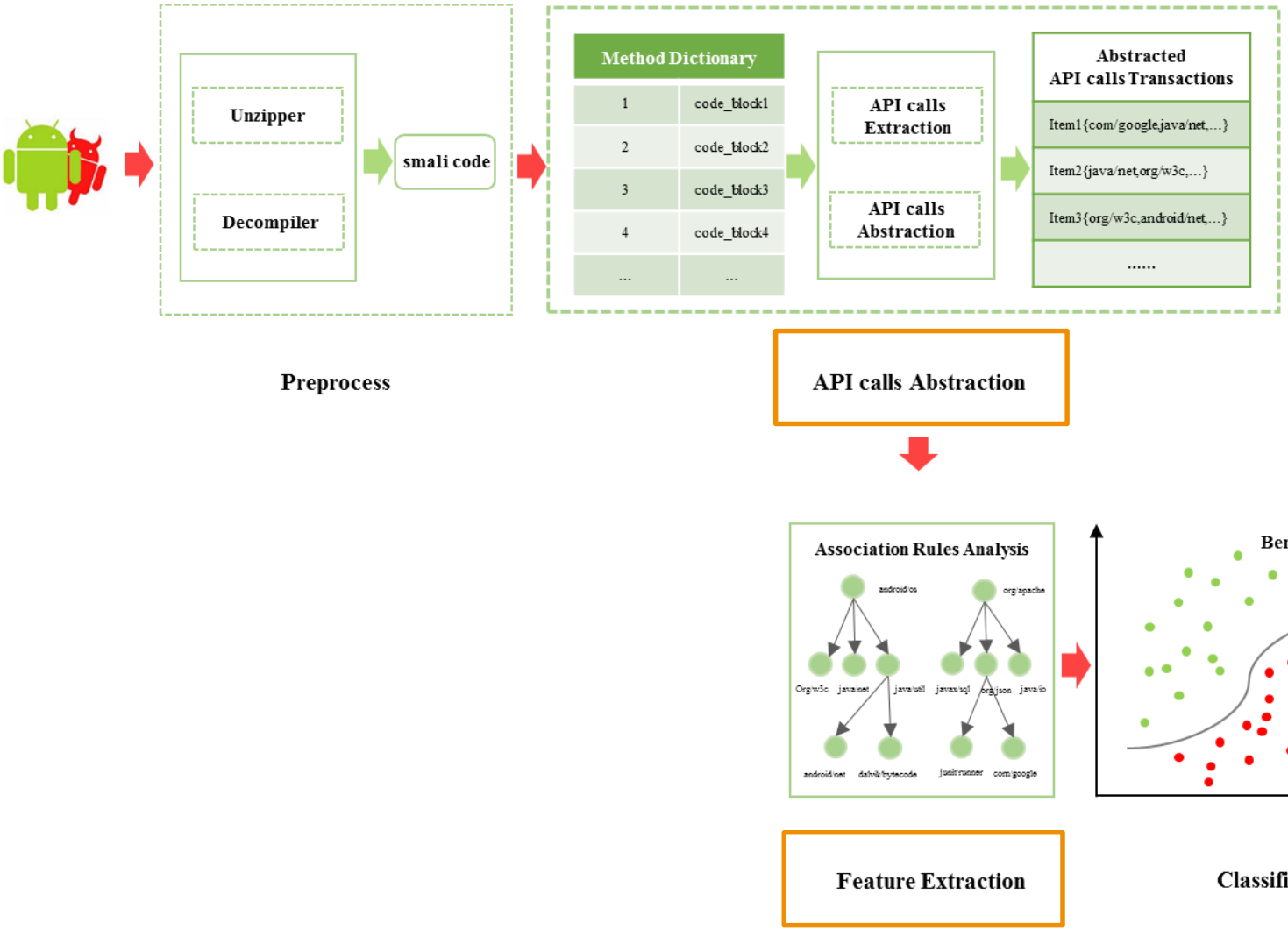
框架原理

T	对Android应用文件进行分析，判断是否为恶意软件
I	APK文件
P	1.预处理 2.特征构建 3.机器学习模型分类
O	APK文件是否为恶意

P	恶意软件在不断的更新进化
C	具备大量有标签APK样本
D	适应Android系统版本升级，构建表征能力强的特征
L	课题工作

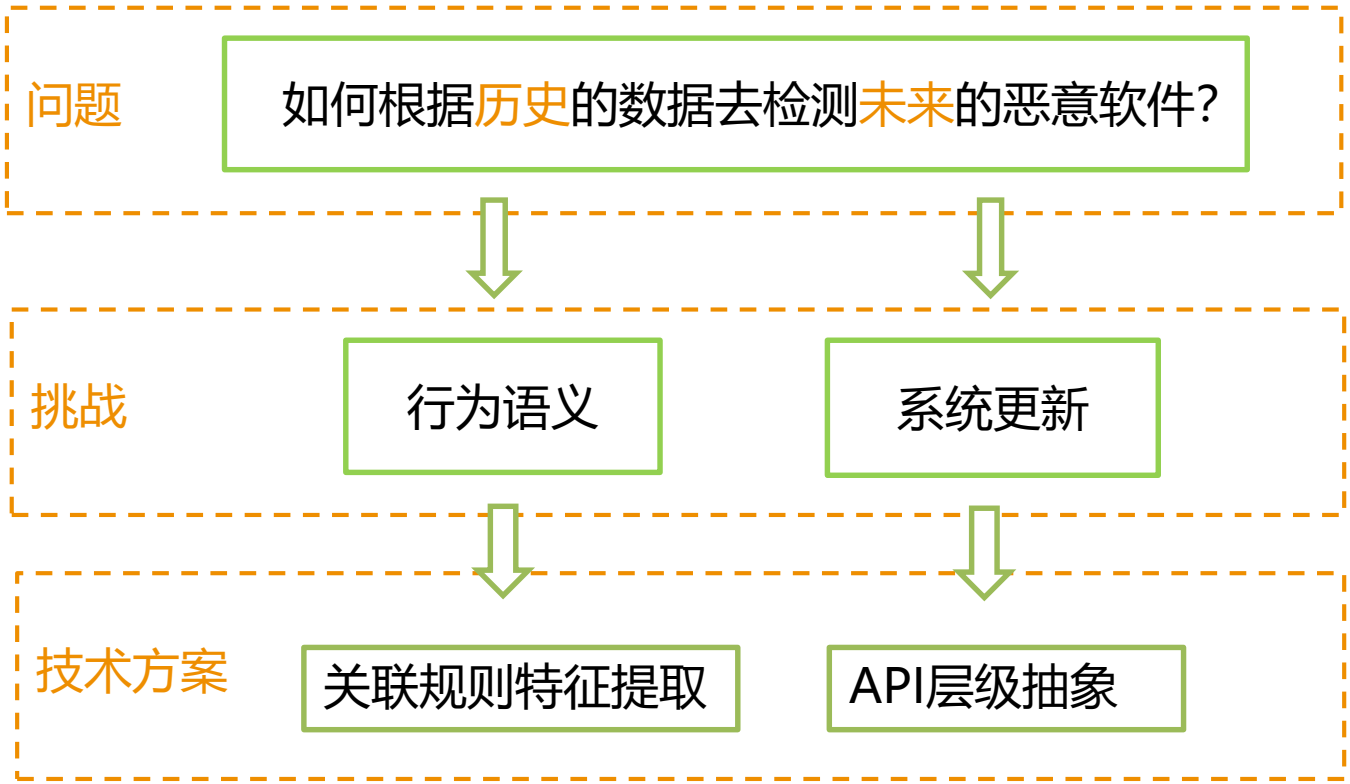
框架原理

- 方法架构


Unzipper
Decompiler
smali code
Method Dictionary
1
code_block1
2
code_block2
3
code_block3
4
code_block4
...
...
API calls Extraction
API calls Abstraction
Abstracted API calls Transactions
Item1{com/google,java/net,...}
Item2{java/net,org/w3c,...}
Item3{org/w3c,android/net,...}
.....
Preprocess
API calls Abstraction
Association Rules Analysis
android/os
org.apache
Org/w3c
java/net
java/util
javax/sql
org.json
java/io
android/net
dalvik/bytecode
junit/runner
com/google
Benign
Malware
Feature Extraction
Classifier

框架原理

- 问题分析


问题
如何根据历史的数据去检测未来的恶意软件?
挑战
行为语义
系统更新
技术方案
关联规则特征提取
API层级抽象

框架原理

- 行为语义
  - 为什么需要行为语义
  - 已有方法的问题
  - 我们的方法

框架原理

- 行为语义-为什么需要行为语义
  - 行为语义

    指能够反映程序动作意图的抽象信息
  - 恶意行为模式是不变的

通过程序行为语义分析，挖掘通用的恶意行为模式
方可应对恶意软件的不断变化！

框架原理

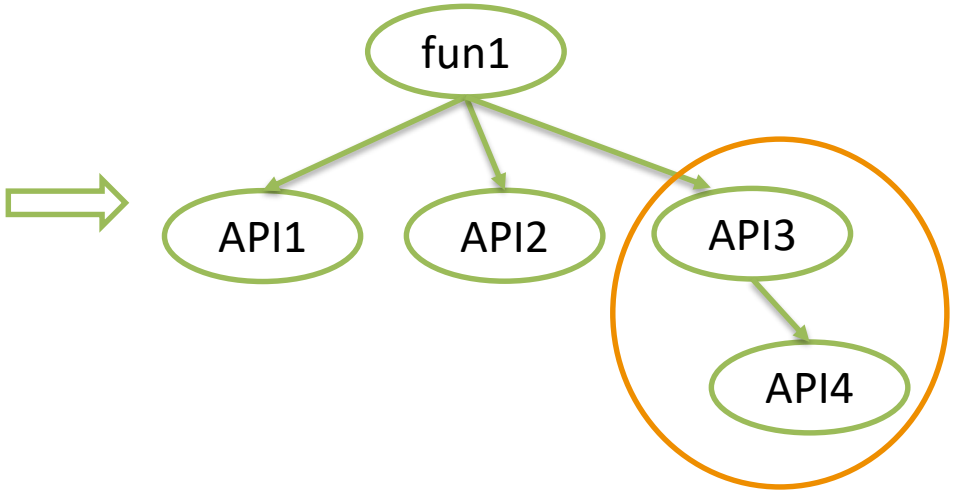
- 行为语义-典型的方法有什么问题
  - API、权限等结构化特征

    无法建立细致的行为语义，造成误判
  - 函数关系调用图 [MaMaDroid]

    空间、时间资源消耗极大，无法实用

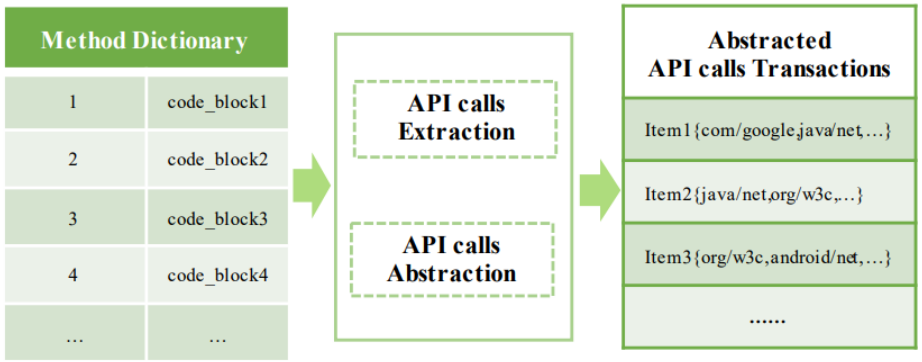
    行为语义信息不全

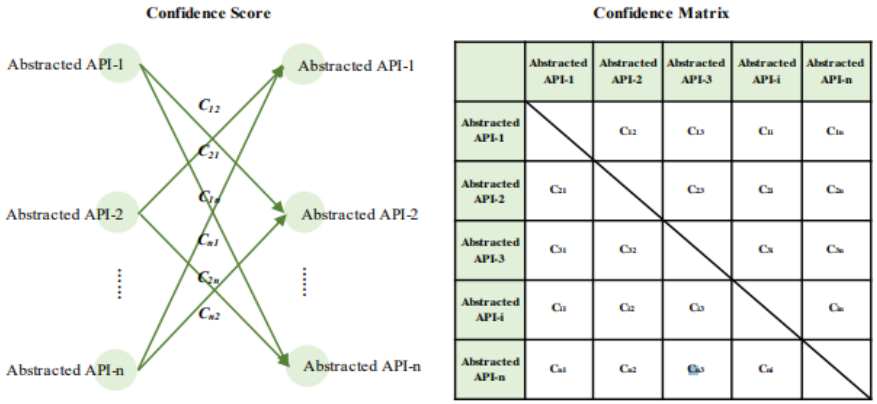
```
fun1(){
a=API1().API4()
b=API2()
c=API3()
}
```



框架原理

- 行为语义-我们的方法





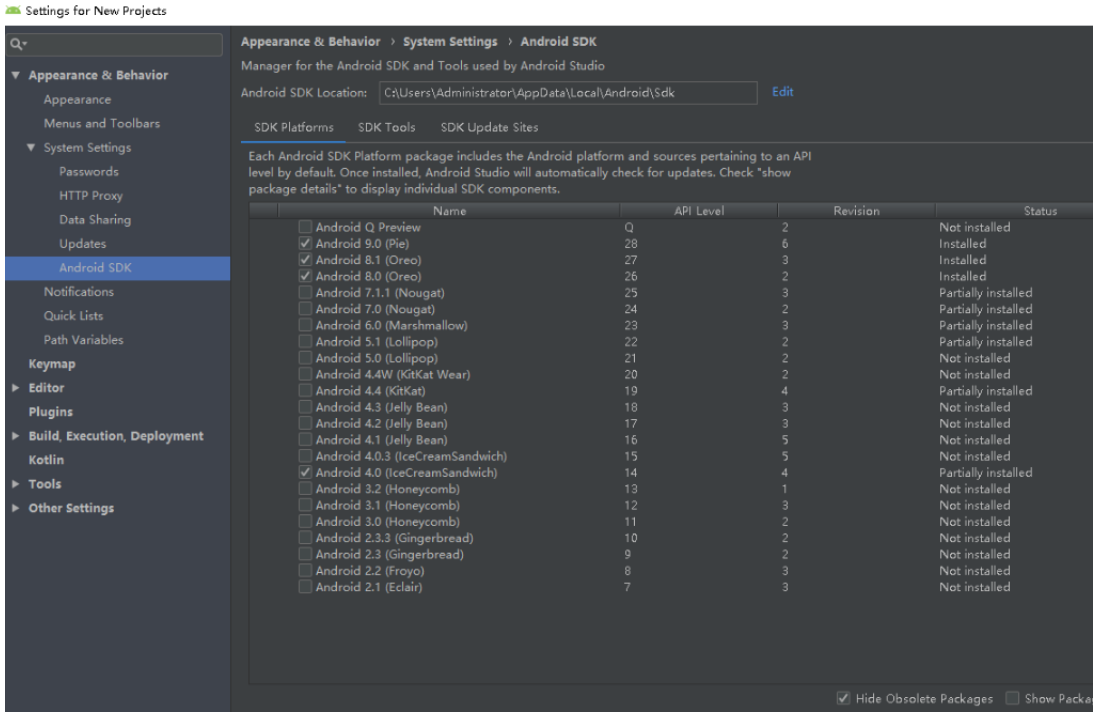
框架原理

- 系统更新
  - 为什么要处理系统更新
  - 如何去处理

框架原理

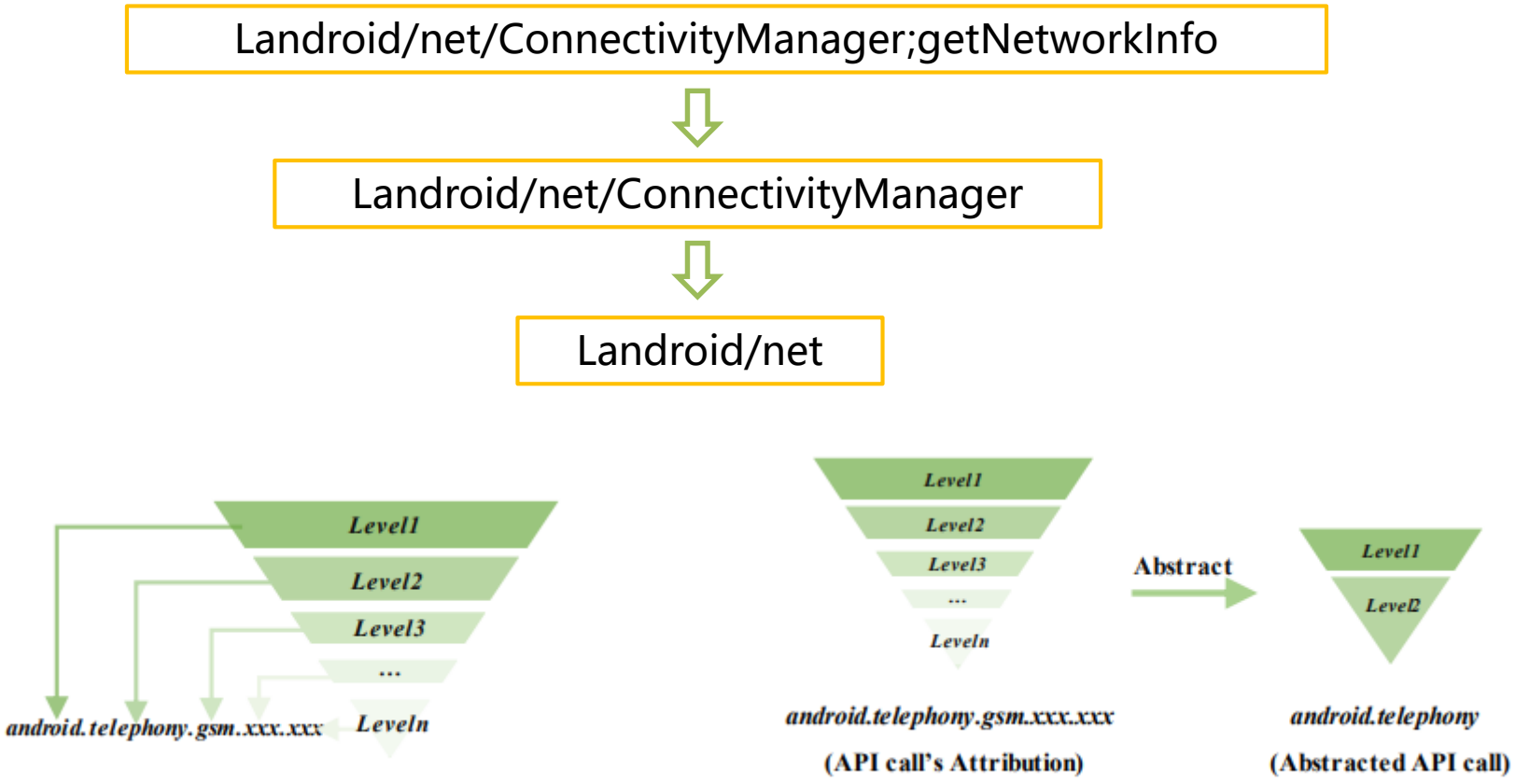
- 系统更新-为什么要处理系统跟新
  - 2008年-API level1
  - 2018年-API Level28

11年28次
平台版本更新



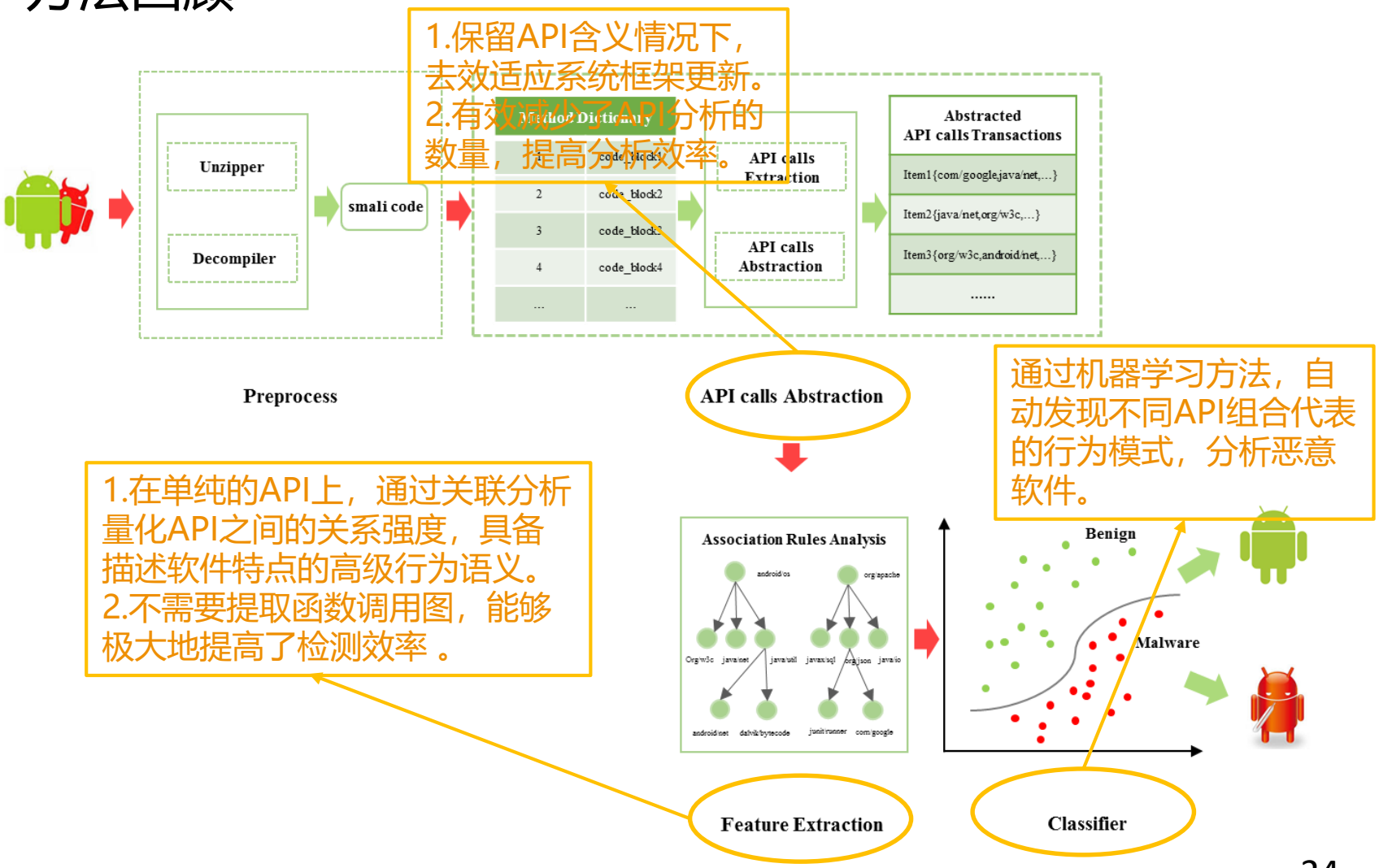
框架原理

- 系统更新-我们的方法



框架原理

- 方法回顾


1.保留API含义情况下，去效适应系统框架更新。
2.有效减少了API分析的数量，提高分析效率。
Unzipper
Decompiler
smali code
Preprocess
Method Dictionary
1
code_block1
2
code_block2
3
code_block3
4
code_block4
...
...
API calls Extraction
API calls Abstraction
Abstracted API calls Transactions
Item1{com/google,java/net,...}
Item2{java/net,org/w3c,...}
Item3{org/w3c,android/net,...}
.....
API calls Abstraction
通过机器学习方法，自动发现不同API组合代表的行为模式，分析恶意软件。
1.在单纯的API上，通过关联分析量化API之间的关系强度，具备描述软件特点的高级行为语义。
2.不需要提取函数调用图，能够极大地提高了检测效率 。
Association Rules Analysis
android/os
org/apache
Org/w3c
java/net
java/util
javax/sql
org/json
java/io
android/net
dalvik/bytecode
junit/runner
com/google
Benign
Malware
Feature Extraction
Classifier

实验分析

- 评价指标

TABLE 1. Performance indices of Android malware detection.

Indices	Description
Precious	$\frac{TP}{TP+FP}$
Recall	$\frac{TP}{TP+FN}$
F-measure	$\frac{2 * Precious * Recall}{Precious + Recall}$
Acc	$\frac{TP+TN}{TP+TN+FP+FN}$

实验分析

- 统检测方法在公开数据集的表现

TABLE 2. Overview of datasets used for our system evaluation experiments.

Dataset	Malware		Benign	
	Number	Description	Number	Description
#Drebin	5,560	From Drebin[5] (2010 to 2012)	5,945	From Androzoo[28] (2010 to 2012)
#AMD	20,843	From AMD[6] (2010 to 2016)	20,519	From Androzoo[28] (2010 to 2016)

TABLE 3. Detection performance results with different algorithms on #Drebin dataset

Algorithm	Precious	Recall	F-measure	Acc
KNN	0.90	**0.96**	0.93	0.92
RF	**0.97**	0.95	**0.96**	**0.96**
SVM	0.94	0.94	0.94	0.94

TABLE 4. Detection performance results with different algorithms on #AMD dataset

Algorithm	Precious	Recall	F-measure	Acc
KNN	0.95	0.97	0.96	0.96
RF	**0.99**	**0.98**	**0.98**	**0.98**
SVM	0.97	0.97	0.97	0.97

单种特征下的效果!

实验分析

- 对比分析

TABLE 5. Overview of the datasets used for comparative experiments with MaMaDroid.

Category	Name	Date range	#sample	#our work	# [3]
Benign	oldbenign	2014	2,992	2,923	2,942
	newbenign	2017	3,000	2,915	2,907
		Total Benign	**5,992**	**5,838**	**5,948**
Malware	Drebin	2010-2012	5,560	5,439	5,518
	2013	2013	3,000	2,997	2,873
	2014	2014	3,998	2,987	2,510
	2015	2015	2,962	2,886	2,809
	2016	2016	2,960	2,923	2,740
	2017	2017	3,040	2,916	2,830
		Total Malware:	**20,520**	**20,148**	**19,280**

TABLE 6. Detection performance on F-measure of our work vs MaMaDroid [3]. As shown in the table, among the 36 test results, 25 of our work are better than or equal to the MaMaDroid.

	Testing Sets											
	Our work	[3]	Our work	[3]	Our work	[3]	Our work	[3]	Our work	[3]	Our work	[3]
Training Set	Drebin&oldbenign		2013&oldbenign		2014&oldbenign		2015&oldbenign		2016&oldbenign		2017&oldbenign	
Drebin&oldbenign	**0.97**	0.93	0.73	**0.84**	0.60	**0.80**	0.59	**0.82**	0.55	**0.78**	0.44	**0.51**
2013&oldbenign	**0.78**	0.67	**0.93**	0.90	0.79	**0.87**	0.67	**0.73**	0.68	**0.70**	**0.29**	0.20
2014&oldbenign	**0.90**	0.45	**0.90**	0.81	0.91	**0.92**	**0.89**	0.76	**0.86**	0.72	**0.61**	0.22
	Drebin&newbenign		2013&newbenign		2014&newbenign		2015&newbenign		2016&newbenign		2017&newbenign	
2015&newbenign	**0.97**	0.94	**0.97**	**0.97**	**0.96**	0.93	**0.94**	0.93	0.90	**0.91**	**0.73**	0.57
2016&newbenign	**0.93**	0.83	**0.96**	0.92	**0.96**	0.89	**0.95**	0.91	**0.94**	**0.94**	**0.89**	0.84
2017&newbenign	**0.97**	0.61	**0.93**	0.80	**0.88**	0.85	**0.85**	**0.85**	0.90	**0.92**	**0.93**	0.92

实验分析

- 对比分析

TABLE 7. Runtime performance of our work vs MaMaDroid [3].

Group	Min(second)	Max(second)	Mean(second)	Mean ratio
	[3] our work	[3] our work	[3] our work	[3]:our work
2010-2012	1.56 0.01	165.69 6.42	15.7 0.72	21.8
*2010-2012	1.56 0.01	165.69 6.42	15.7 0.72	21.8
2013	1.56 0.01	462.22 5.43	33.69 1.61	21.0
*2013	1.56 0.01	462.22 5.43	33.69 1.61	21.0
2014	4.53 0.07	546.19 12.45	33.15 2.00	16.6
*2014	4.53 0.07	546.19 12.45	33.15 2.00	16.6
2015	0.47 0.05	**7162.13** 12.45	**160.32** 2.68	**59.7**
*2015	0.47 0.05	**1341.46** 12.45	**40.3** 2.68	**15.0**
2016	3.61 0.04	**12770.95** 14.77	**374.01** 3.97	**94.2**
*2016	3.61 0.04	**1459.54** 14.77	**87.21** 3.97	**22.0**
2017	0.94 0.02	**4092.77** 15.37	**103.98** 6.56	**15.7**
*2017	0.94 0.02	**515.75** 15.37	**64.21** 6.56	**9.8**
		Total Mean:	**120.14 2.90**	**41.4**
		***Total Mean:**	**45.70 2.90**	**15.6**

note: the groups that marked with * mean the experiment results after removing files with the analysis time of more than 20 minutes

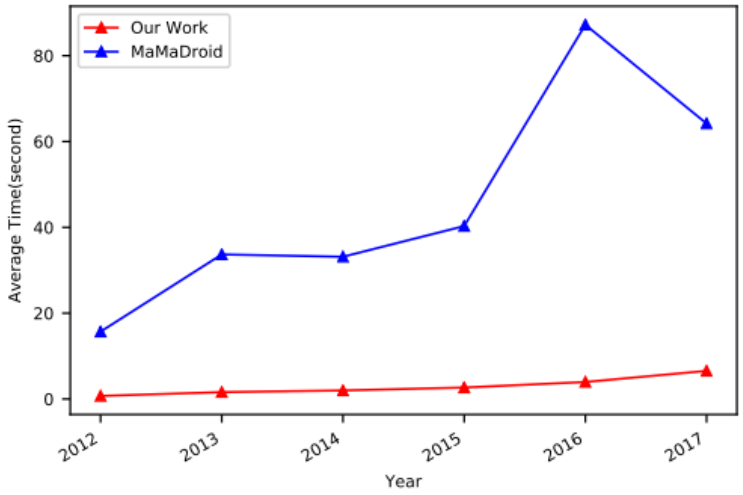


FIGURE 9. Results on the average analysis time in each group of our work vs MaMaDroid

2.9s VS 47.9s

优劣分析

- 优点
  - 表征能力强。让恶意软件变种难以逃脱检测。
  - 效率高可以直接应用于工业场景。
  - 能够有效的应对代码混淆。
- 不足
  - 基于静态分析，无法处理加固或者具有动态加载功能的恶意软件。
  - 容易被有意添加的无关代码干扰。

参考文献

- E. Mariconti, L. Onwuzurike, P. Andriotis, E. De Cristofaro, G. Ross,and G. Stringhini, “Mamadroid: Detecting android malware by buildingmarkov chains of behavioral models,” in Proceedings of the Annual Symposium on Network and Distributed System Security (NDSS), 2017
- http://www.android/doc.com/reference/packages.html
- https://blog.csdn.net/u011936381/article/details/46048893

谢谢！

大成若缺，其用不弊。大盈若冲，其用不穷。大直若屈。大巧若拙。大辩若讷。静胜躁，寒胜热。清静为天下正。