DroidScribe

Classifying Android Malware Based on Runtime Behavior

Santanu Kumar Dash, **Guillermo Suarez-Tangil**, Salahuddin Khan, Kimberly Tam, Mansour Ahmadi, Johannes Kinder, and Lorenzo Cavallaro

Royal Holloway, University of London
University of Cagliari

May 26, 2016
Mobile Security Technologies (MoST)

Research supported by the UK EPSRC grants EP/K033344/1 and EP/L022710/1

Background

Automated Analysis

- Obtain rich **static** view of an app
- Obtain rich **dynamic** view of an app

Type of Problems

- Malware Detection
  - Crucial for final users
- **Family Identification**
  - Crucial for analysis of threats and mitigation planning

Smart Phones		Desktop	
Static	Dynamic	Static	Dynamic
1			

- In the mobile realm
  - 1 Dendroid : CFG

API: Application Programming Interface, API-F: Information Flow between APIs, INT: Intents, CG: Call Graph, PER: Requested Permissions, CFG: Control Flow Graph, PKG: Package information of API, SYS: System Calls

Smart Phones		Desktop	
Static	**Dynamic**	**Static**	**Dynamic**
2			

- In the mobile realm
  - 1 Dendroid : CFG
  - 2 DroidLegacy : API

API: Application Programming Interface, API-F: Information Flow between APIs, INT: Intents, CG: Call Graph, PER: Requested Permissions, CFG: Control Flow Graph, PKG: Package information of API, SYS: System Calls

Smart Phones		Desktop	
Static	Dynamic	Static	Dynamic
3			

- In the mobile realm
    - 1 Dendroid : CFG
    - 2 DroidLegacy : API
    - 3 DroidMiner : CG, API

API: Application Programming Interface, API-F: Information Flow between APIs, INT: Intents, CG: Call Graph, PER: Requested Permissions, CFG: Control Flow Graph, PKG: Package information of API, SYS: System Calls

Smart Phones		Desktop	
Static	**Dynamic**	**Static**	**Dynamic**
4			

- In the mobile realm
  - 1 Dendroid : CFG
  - 2 DroidLegacy : API
  - 3 DroidMiner : CG, API
  - 4 DroidSIFT : API-F

API: Application Programming Interface, API-F: Information Flow between APIs, INT: Intents, CG: Call Graph, PER: Requested Permissions, CFG: Control Flow Graph, PKG: Package information of API, SYS: System Calls

Smart Phones		Desktop	
Static	**Dynamic**	**Static**	**Dynamic**
5			

- In the mobile realm
  - 1 Dendroid : CFG
  - 2 DroidLegacy : API
  - 3 DroidMiner : CG, API
  - 4 DroidSIFT : API-F
  - 5 RevealDroid : PER, API, API-F, INT, PKG

API: Application Programming Interface, API-F: Information Flow between APIs, INT: Intents, CG: Call Graph, PER: Requested Permissions, CFG: Control Flow Graph, PKG: Package information of API, SYS: System Calls

State of the Art
On Family Identification

Smart Phones		Desktop	
Static	**Dynamic**	**Static**	**Dynamic**
			

- In the mobile realm
  - 1 Dendroid : CFG
  - 2 DroidLegacy : API
  - 3 DroidMiner : CG, API
  - 4 DroidSIFT : API-F
  - 5 RevealDroid : PER, API, API-F, INT, PKG

`API`: Application Programming Interface, `API-F`: Information Flow between APIs, `INT`: Intents, `CG`: Call Graph, `PER`: Requested Permissions, `CFG`: Control Flow Graph, `PKG`: Package information of API, `SYS`: System Calls

Smart Phones		Desktop	
Static	Dynamic	Static	Dynamic

- In the mobile realm
  - 1 Dendroid : CFG
  - 2 DroidLegacy : API
  - 3 DroidMiner : CG, API
  - 4 DroidSIFT : API-F
  - 5 RevealDroid : PER, API, API-F, INT, PKG

API: Application Programming Interface, API-F: Information Flow between APIs, INT: Intents, CG: Call Graph, PER: Requested Permissions, CFG: Control Flow Graph, PKG: Package information of API, SYS: System Calls

Smart Phones		Desktop	
Static	Dynamic	Static	Dynamic
			

- In the mobile realm
  - 1 Dendroid : CFG
  - 2 DroidLegacy : API
  - 3 DroidMiner : CG, API
  - 4 DroidSIFT : API-F
  - 5 RevealDroid : PER, API, API-F, INT, PKG

API: Application Programming Interface, API-F: Information Flow between APIs, INT: Intents, CG: Call Graph, PER: Requested Permissions, CFG: Control Flow Graph, PKG: Package information of API, SYS: System Calls

Smart Phones		Desktop	
Static	Dynamic	Static	Dynamic
			

- In the mobile realm
  - (1) Dendroid : CFG
  - (2) DroidLegacy : API
  - (3) DroidMiner : CG, API
  - (4) DroidSIFT : API-F
  - (5) RevealDroid : PER, API, API-F, INT, PKG
- In the desktop realm
  - **SYS** have been successfully used

`API`: Application Programming Interface, `API-F`: Information Flow between APIs, `INT`: Intents, `CG`: Call Graph, `PER`: Requested Permissions, `CFG`: Control Flow Graph, `PKG`: Package information of API, `SYS`: System Calls

State of the Art
On Family Identification

Smart Phones		Desktop	
Static	**Dynamic**	**Static**	**Dynamic**
			

Android System Call Profile

- Android services are invoked through `ioctl`
- `ioctl`s are dispatched to the *Binder* kernel driver, which implements Android's main **IPC** and **ICC**
- Distinguishing Binder calls is essential for the malware classif.

Our Contribution

Smart Phones		Desktop	
Static	Dynamic	Static	Dynamic
☹	☺	☹	☹

Goal To evaluate the use of dynamic analysis for family identification under **challenging conditions**

Challenges

- Similar/sparse behaviors

Our contributions

- **RQ1**: What is the best level abstraction?
- **RQ2**: Can we deal with sparse behaviors?

Dynamic Analysis Component

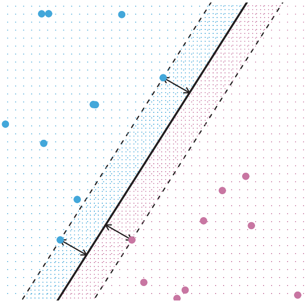
CopperDroid[1]

- Runs apps in a sandbox, records system calls and their arguments, and reconstructs high-level behavior
- Reconstructs contents of all transactions going through the Binder mechanism for inter-process communication

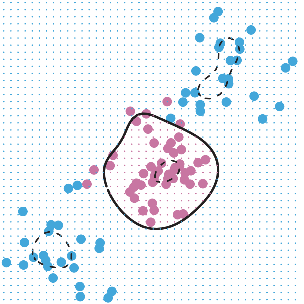
[1] Tam, K., Khan, S.J., Fattori, A. and Cavallaro, L. "CopperDroid: Automatic Reconstruction of Android Malware Behaviors." NDSS. 2015.

Machine Learning Component

- Use existing malware classified into families as training data
- Use Support Vector Machines as the classification algorithm



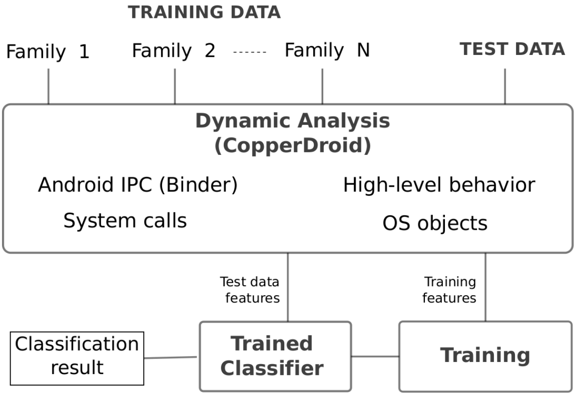
Linear function



Radial-basis function

Source: An Introduction to Statistical Learning–G. James et al.

Overview of the Classification Framework

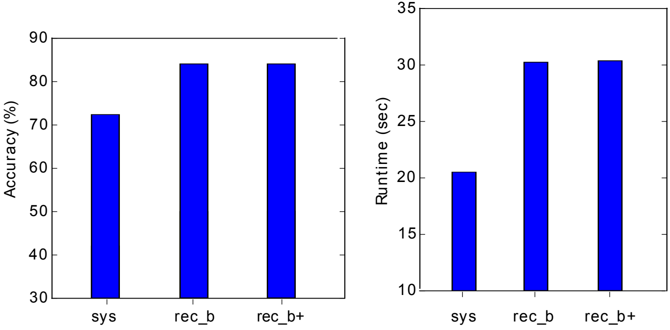


System-calls vs. abstract behaviors

RQ1

What is the best level abstraction?

- Experiments on the Drebin dataset (5,246 malware samples).
- Reconstructing Binder calls adds 141 meaningful features.
- High level behaviors added 3 explanatory features.



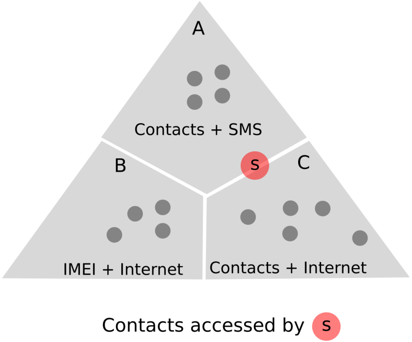
(a) Accuracy (b) Runtime

Set-Based Prediction

- Dynamic analysis is limited by code coverage
- Classifier has only partial information about its behaviors
- Identify when malware cannot be classified into a family
  - Based on a measure of the statistical confidence
- Helpful human analyst by identifying the top matching families

Classification from Observed Features

- When more than one choice of similar likelihood exists, ...
- ... traditional classification algorithms are prone to error



Conformal Predictor (CP)

- Is statistical learning algorithm tailored at classification
- Provides statistical evidences on the results

Credibility

Supports how good a sample fits into a class

Confidence

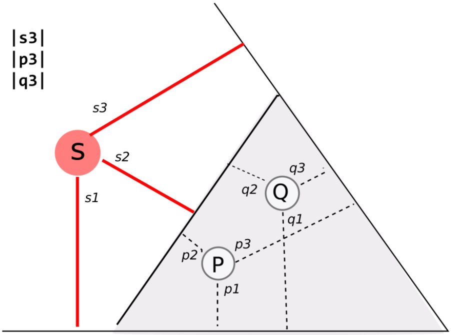
Indicates if there are other good choices

Robust Against Outliers

Aware of values from other members of the same class

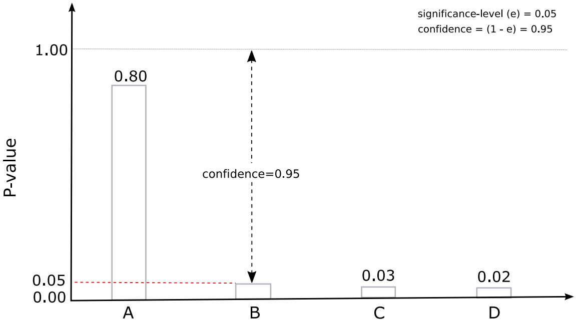
- P-value is the probability of truth for the hypothesis that a sample belongs to a class

```
ncm_S = |s1| + |s2| + |s3|
ncm_P = |p1| + |p2| + |p3|
ncm_Q = |q1| + |q2| + |q3|

ncm_S > ncm_P > ncm_Q

pval_S = 0/3
pval_P = 1/3
pval_Q = 2/3
```



In an ideal world

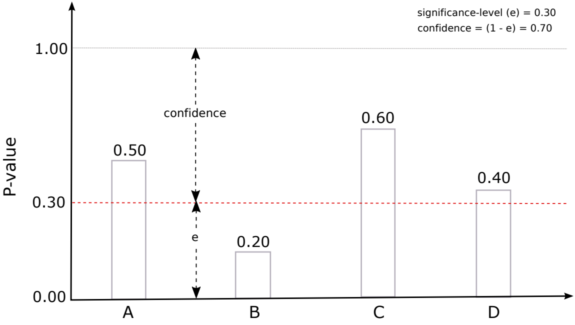
Given a new object s, conformal predictor picks the class with the highest p-value and return a singular prediction.



Prediction = A

Obtaining Prediction Sets

Given a new object s, we can set a significance-level e for p-values and obtain a prediction set Γ^e includes labels whose p-value is greater than e for the sample.



Prediction Set = {A, C, D}

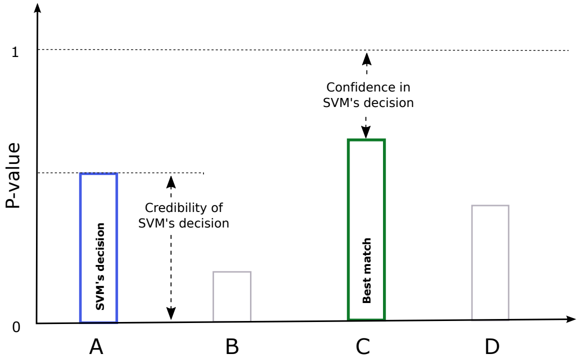
- CP is an expensive algorithm
  - For each sample, we need to derive a p-value for each class
  - Computation complexity of $O(nc)$ where n is number of samples and c is the number of classes

Conformation Evaluation

- Provide statistical evaluation of the quality of a ML algorithm
  - Quality threshold to understand when should be trusting SVM
  - Statistical evidences of the choices of SVM
  - Selectively invoke CP to alleviate runtime performance

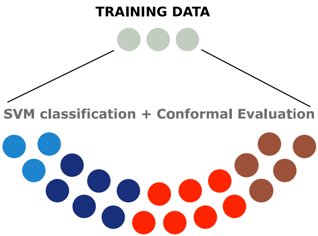
Step 1. Computing Confidence in Training Decisions

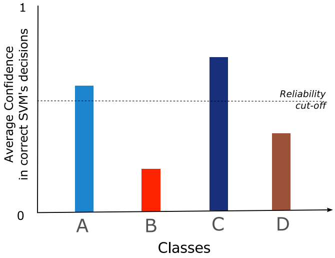
- During training, compute p-values for each sample for each class
- Compute the confidence in the decision for each sample



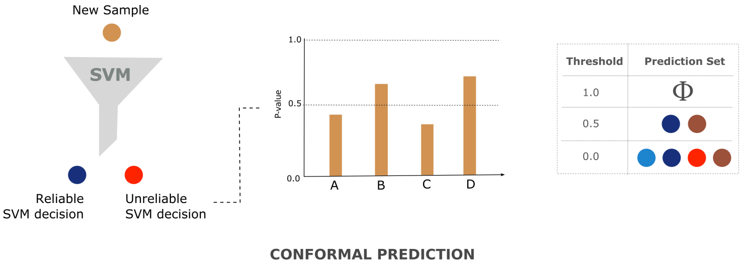
Step 2. Using Class-level Confidence Scores

- For each class, calculate the mean confidence for all decisions mapping to the class
- Use the median of the class-level confidence across all classes as a reliability threshold





Step 3. Invoking the Conformal Predictor

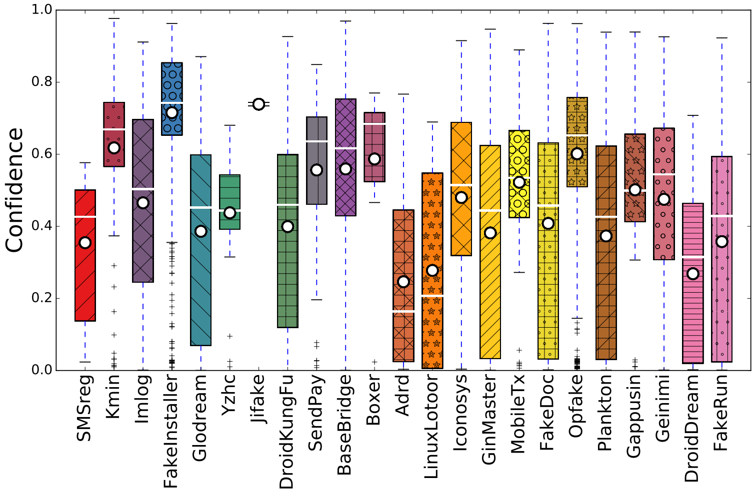


Threshold

The threshold for picking prediction sets is fully tunable

Confidence of correct SVM decisions

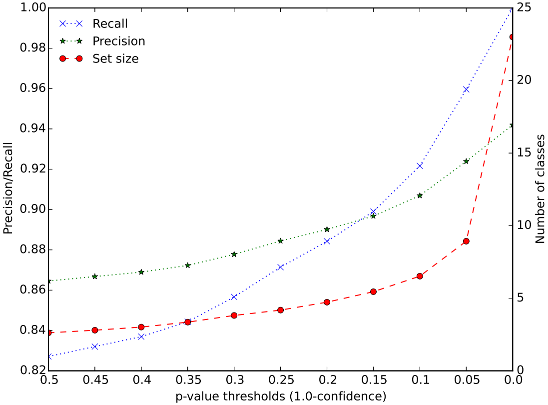
Invoke CP with a set of desired p-value cutoff size



Accuracy vs. Prediction Set Size

RQ2

Can we deal with sparse behaviors?



- Accuracy improves with the prediction set size

Conclusion

- Resolving Binder invocations improves classification accuracy
- Poor coverage leads to misclassification in dynamic analysis
- Predicting sets of top matches ameliorates this problem
- Statistical evaluation can be used to minimize computation
- DroidScribe can be integrated into dynamic analysis frameworks such as CopperDroid

DroidScribe

Classifying Android Malware Based on Runtime Behavior

Santanu Kumar Dash, **Guillermo Suarez-Tangil**, Salahuddin Khan, Kimberly Tam, Mansour Ahmadi, Johannes Kinder, and Lorenzo Cavallaro

Royal Holloway, University of London
University of Cagliari

May 26, 2016
Mobile Security Technologies (MoST)

Research supported by the UK EPSRC grants EP/K033344/1 and EP/L022710/1

Computing P-values

- *Nonconformity Measure* (NCM) is a **geometric measure** of how well a sample is far from a class.
  - For SVM, the NCM $\mathcal{N}_D^z$ of a sample z w.r.t. class D is sum distances from all hyperplanes bounding the class D.

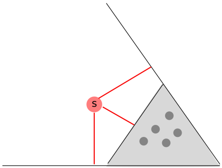
$$\mathcal{N}_D^z = \sum_i d(z, \mathcal{H}_i)$$

- *P-value* is a **statistical measure** of how well a sample fits in a class.
  - P-value $\mathcal{P}_D^z$ represents the proportion of samples in D that more different than z w.r.t. D.

$$\mathcal{P}_D^z = \frac{|\{j = 1, ..., n : \mathcal{N}_D^j \geq \mathcal{N}_D^z\}|}{n}$$

Probability of Membership

- Standard classification algorithms calculate probability of a sample belonging to a class
- For the case of SVM, this is based on Euclidean distance (Platt's scaling)



Using Probabilites

- Platt's scaling is based on logistic regression
- Logistic regression is sensitive to outliers which introduces inaccuracies
- Probabilities to sum up to one which introduces skewing