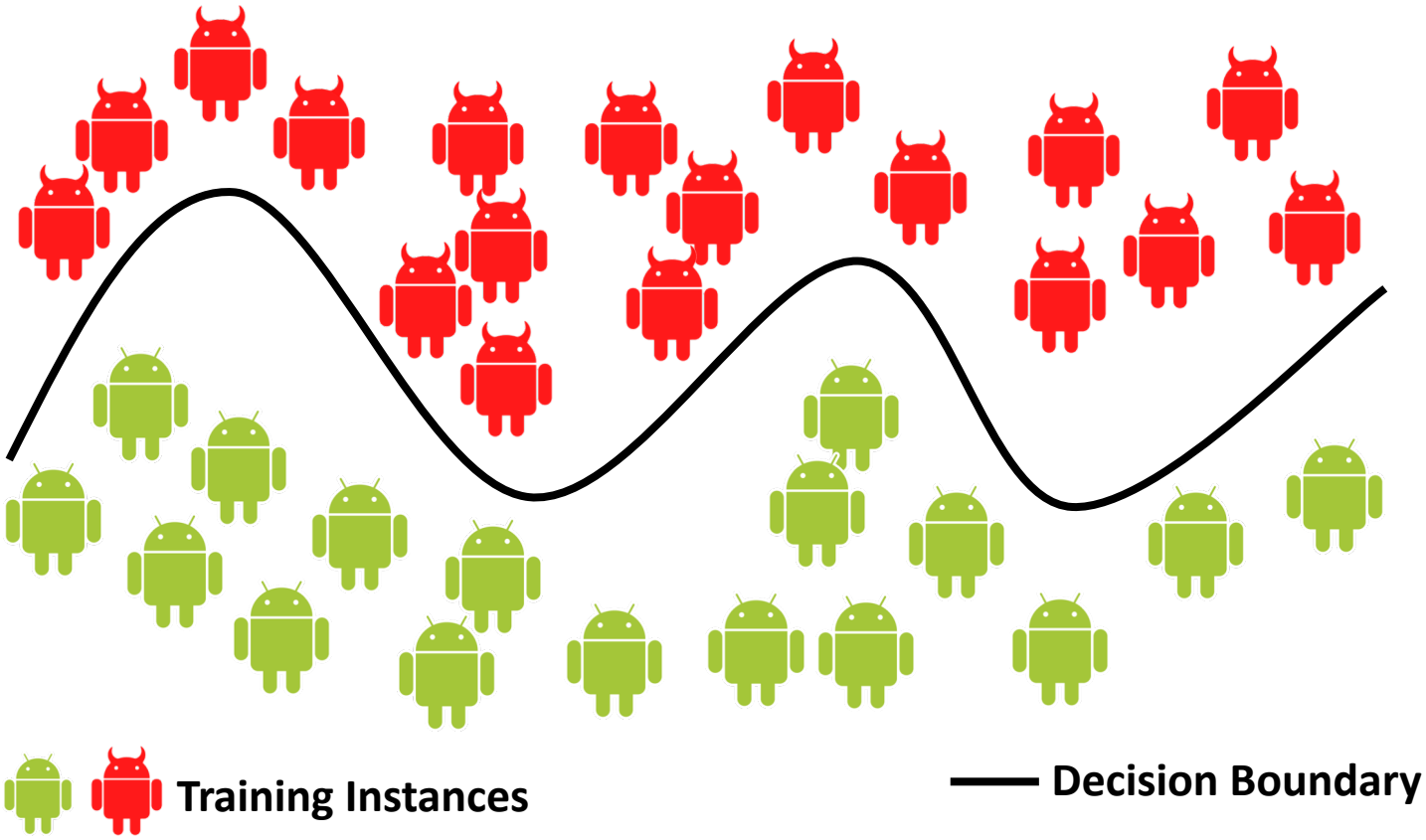
When Does Machine Learning FAIL? Generalized Transferability for Evasion and Poisoning Attacks

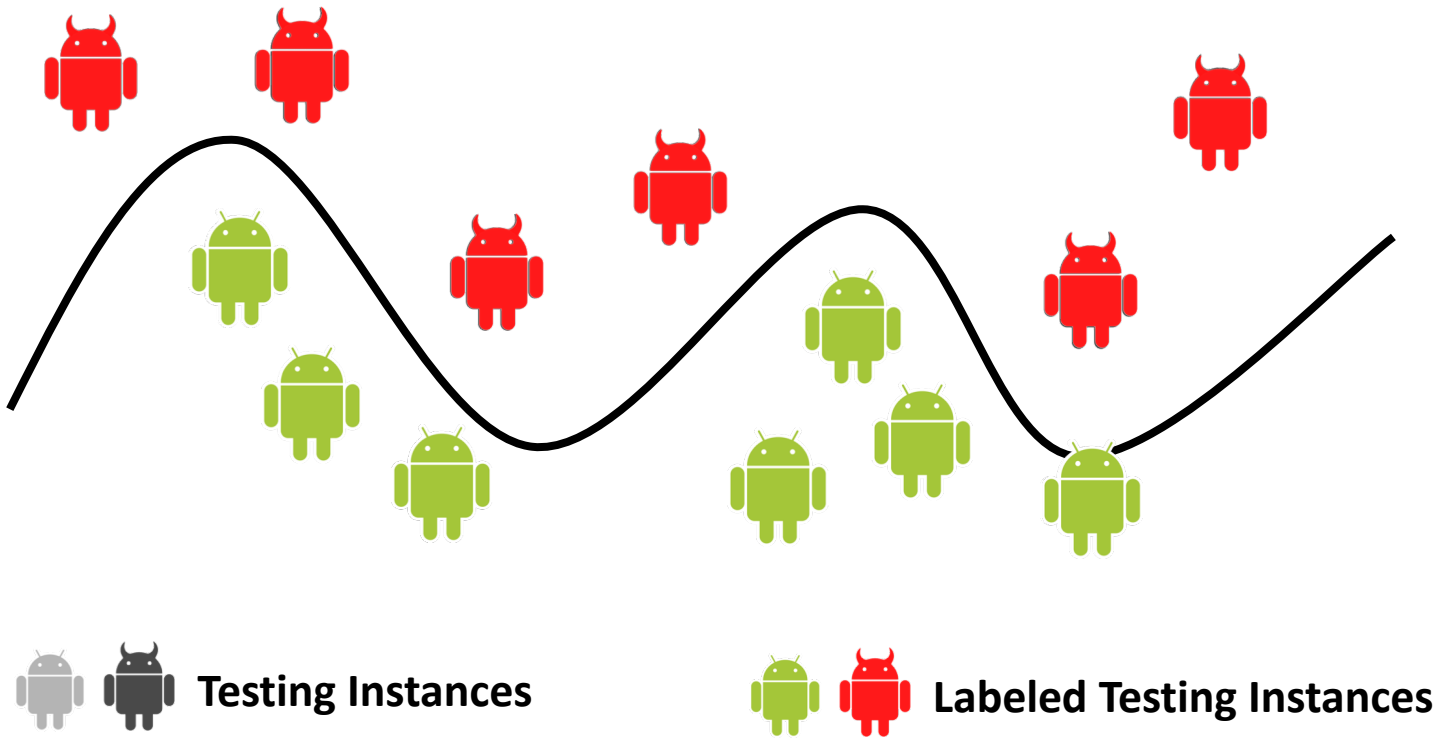
Octavian Suciu Radu Mărginean Yiğitcan Kaya
Hal Daumé III Tudor Dumitraș

University of Maryland, College Park

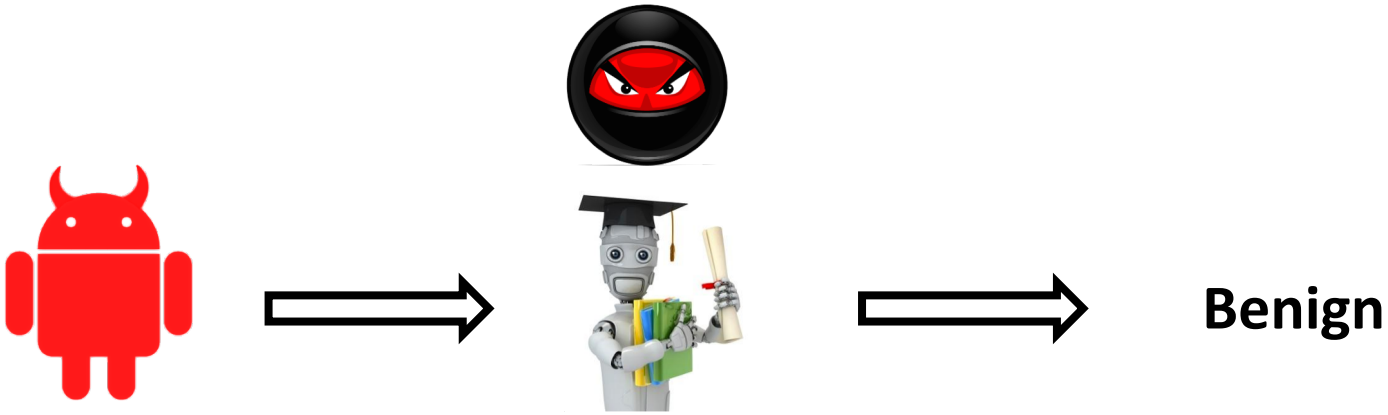
ML - Training a Classifier



ML - Testing the Classifier



Threat Models In Adversarial Machine Larning



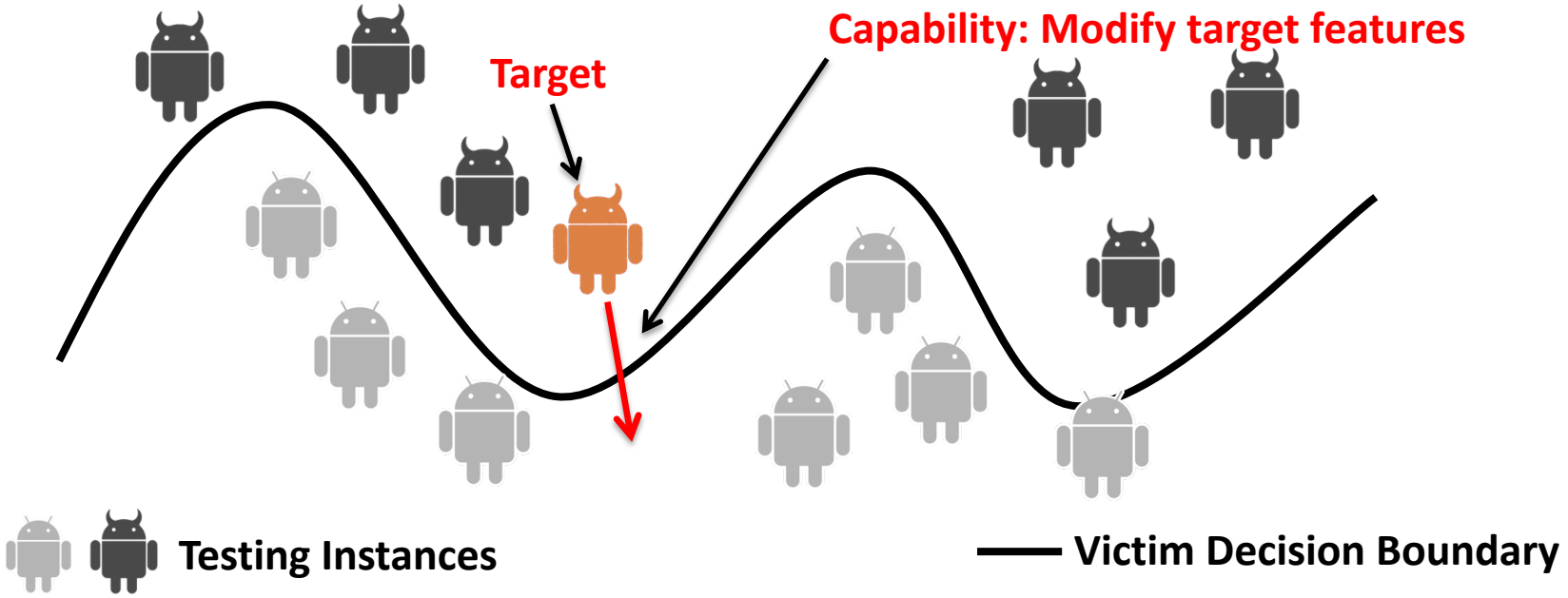
- Lots of proposed attack and defense strategies
  - Various ***assumptions*** about adversaries
- We evaluate the practical impact of assumptions on attack effectiveness
  - This helps design better defense mechanisms

Practical Attack Example

- Let's consider a running example:
- Drebin Android malware detector[1]
  - Support Vector Machine (SVM) classifier
  - Trained using a public dataset

[1] Arp et al. "DREBIN: Effective and Explainable Detection of Android Malware in Your Pocket"; 2014

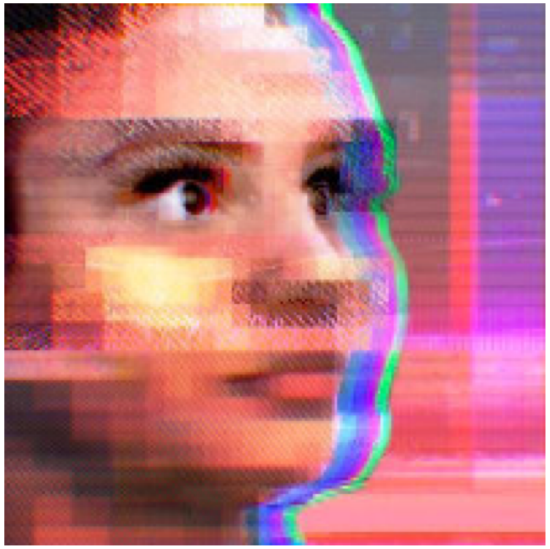
Targeted Evasion Attacks



Evasion: modify target features to cross the decision boundary

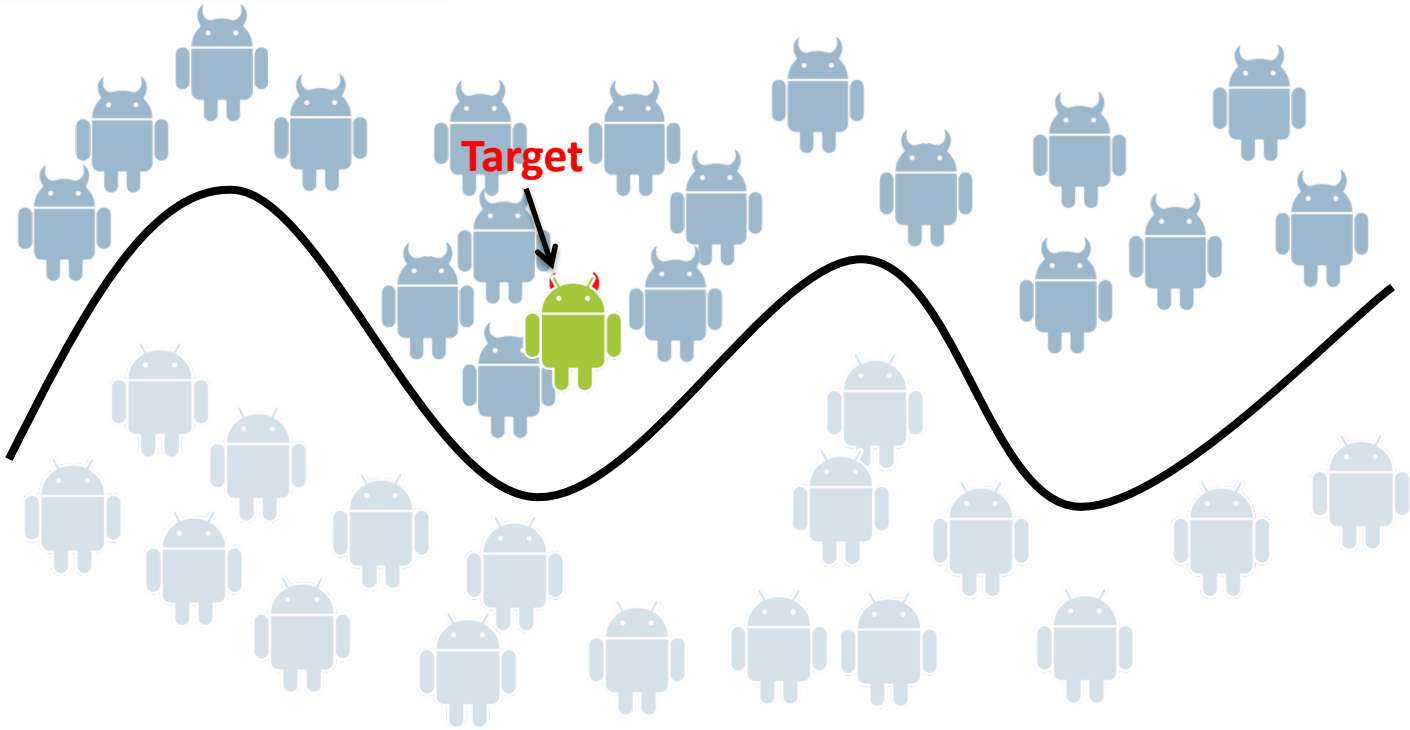
Poisoning Attack Example

- Microsoft's Tay chatbot poisoned through tweets*



* https://www.theverge.com/2016/3/24/11297050/tay-microsoft-chatbot-racist

Poisoning the Drebin Classifier



Victim Training Instances **Victim Decision Boundary**

Attempt 1: add target with flipped label to the training set

Assumption: adversarial control over the labeling function[2][3]

[2] Jagielski et al. "Manipulating Machine Learning: Poisoning Attacks and Countermeasures for Regression Learning"; 2018

[3] Koh et al. "Understanding Black-box Predictions via Influence Functions"; 2017

Practical Label Assignment

Labeling Oracle

virustotal

SHA256: 0a0a78000e418ea28fa02e8c162c43396db6141ef8fe876db4027fef04bed663

File name: 0a0a78000e418ea28fa02e8c162c43396db6141ef8fe876db4027fef04bed663.apk

Detection ratio: 38 / 62

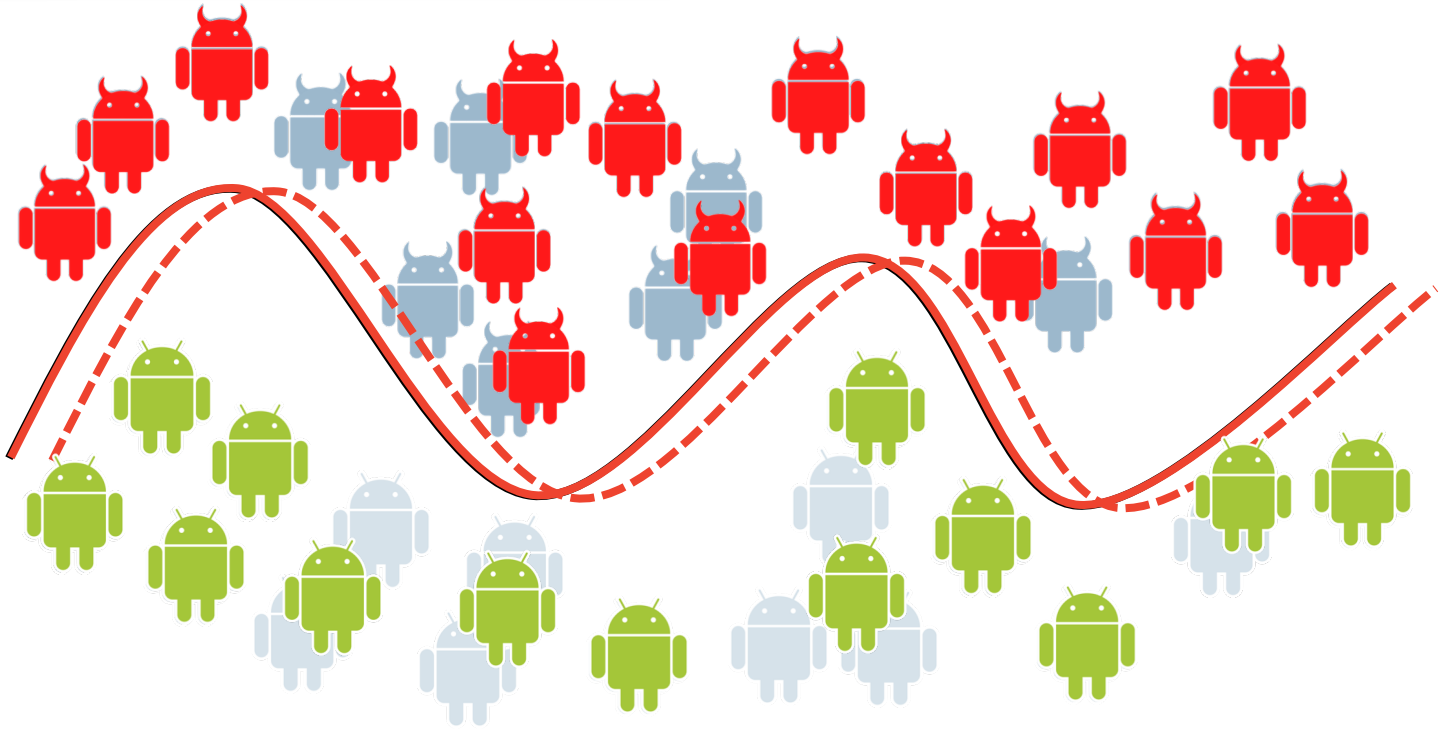
Analysis date: 2018-07-19 15:22:34 UTC (3 weeks, 5 days ago)

Analysis | File detail | Additional information | Comments 0 | Votes

Antivirus	Result
Ad-Aware	Android.Adware.GingerMaster.BD
AegisLab	SUSPICIOUS
AhnLab-V3	Android-Trojan/GinMaster.2510

Our **StingRay attack** achieves targeted poisoning without this assumption

Attacker Limitations Through Existing Models



Victim Training Instances

Victim Decision Boundary

Surrogate Attacker Training Instances[4]

Black-Box Attacker Decision Boundary[5]

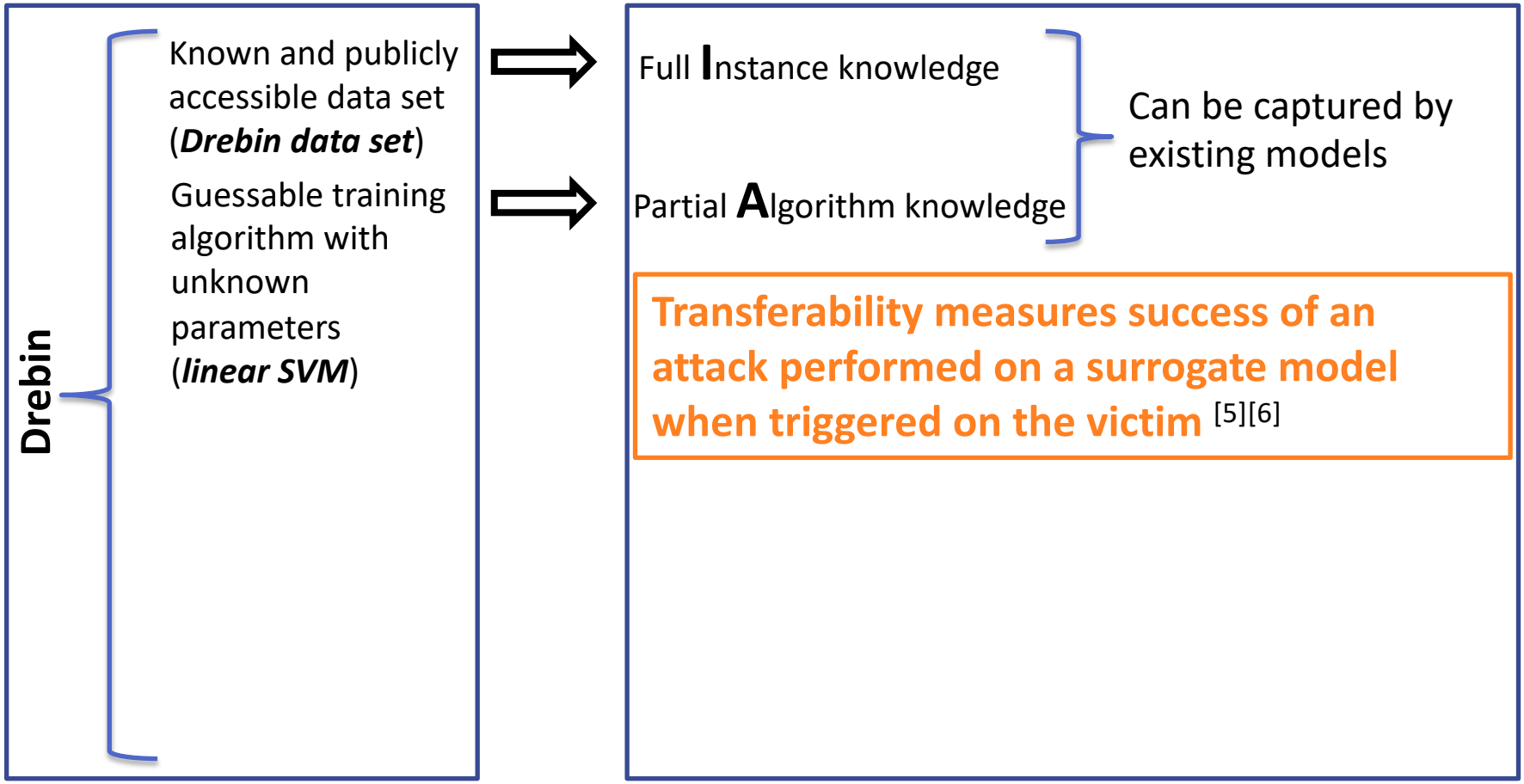
White-Box Attacker Decision Boundary[6]

[4] Šrndic et al. "Practical Evasion of a Learning-Based Classifier: A Case Study”; 2014

[5] Papernot et al. "Practical Black-Box Attacks against Machine Learning”; 2016

[6] Papernot et al. "The limitations of deep learning in adversarial settings.”; 2015

Adversarial Models in Practice

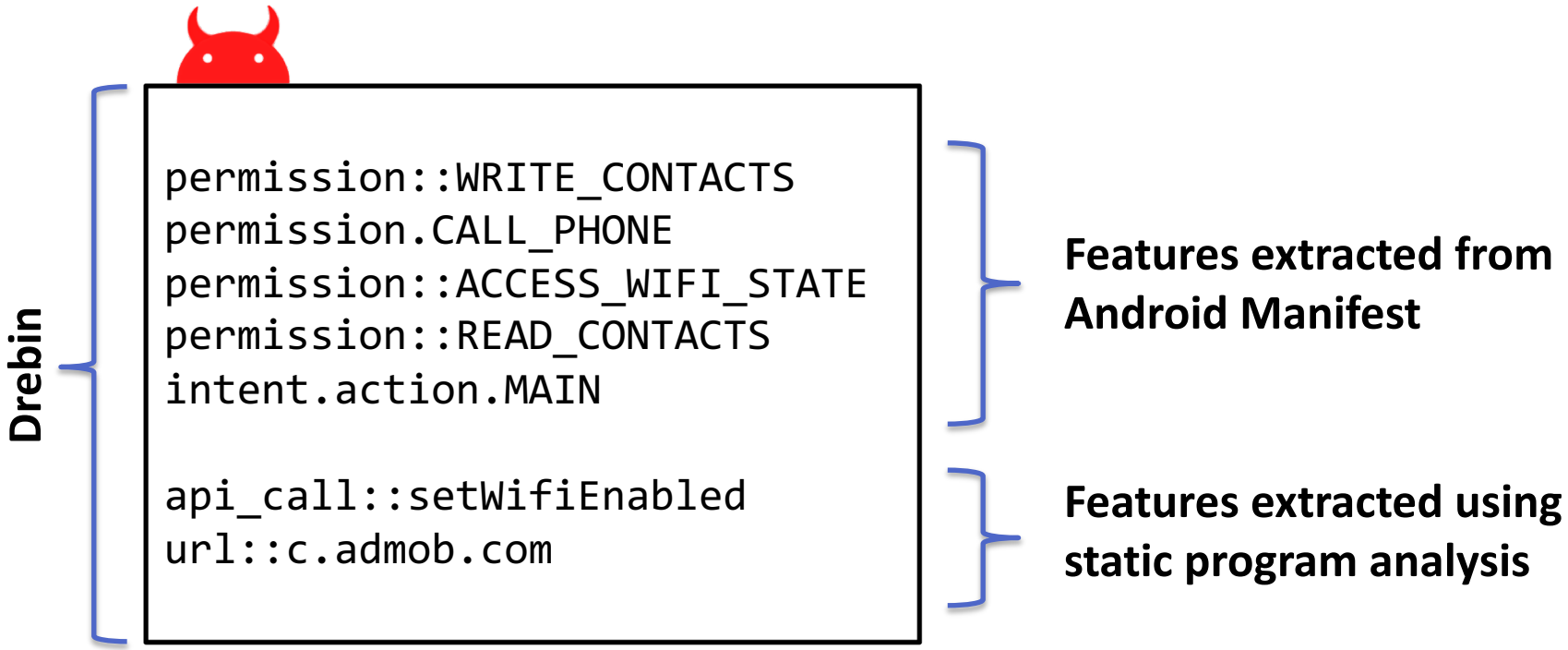


[5] Papernot et al. "Practical Black-Box Attacks against Machine Learning"; 2016

[6] Goodfellow et al. "Explaining and Harnessing Adversarial Examples"; 2014

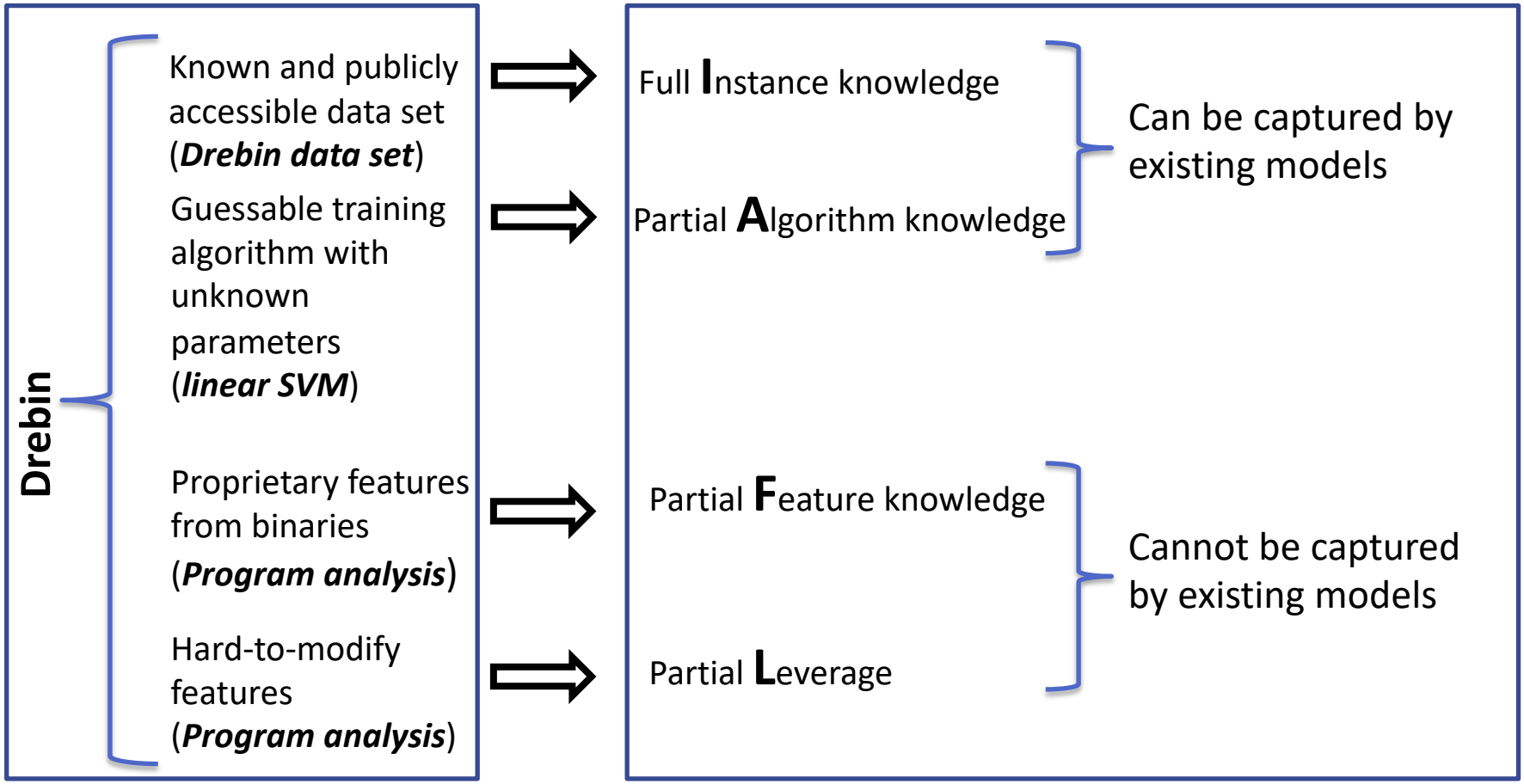
Malware Features in Practice

- Malware detectors use program analysis features
  - Derived from code disassembly



Static program analysis features might be unknown or hard to modify

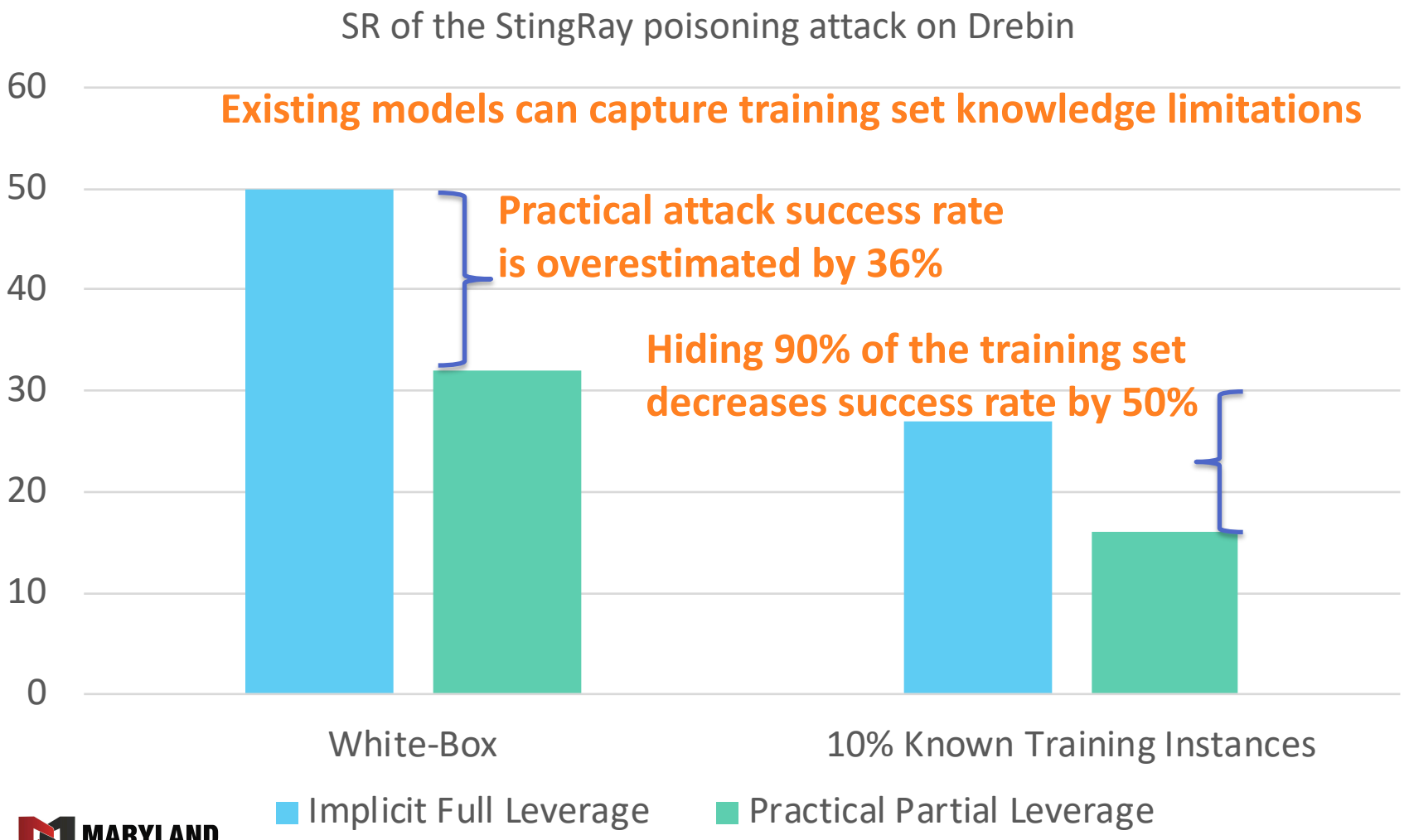
Adversarial Models in Practice (2)



Assumption: attackers have full knowledge and leverage on all features

Practical Effectiveness of Poisoning Attacks

Success Rate (SR): Percentage of attacks that are successful on the victim



Contributions

- FAIL adversarial model for highlighting realistic adversarial capabilities
  - Represents knowledge and control along: Features, Algorithms, Instances, Leverage
- StingRay, a generic targeted poisoning attack
  - Implemented on four applications and against three defenses
- Systematic evaluation of how much adversarial success depends on implicit assumptions
  - More accurate threat assessment

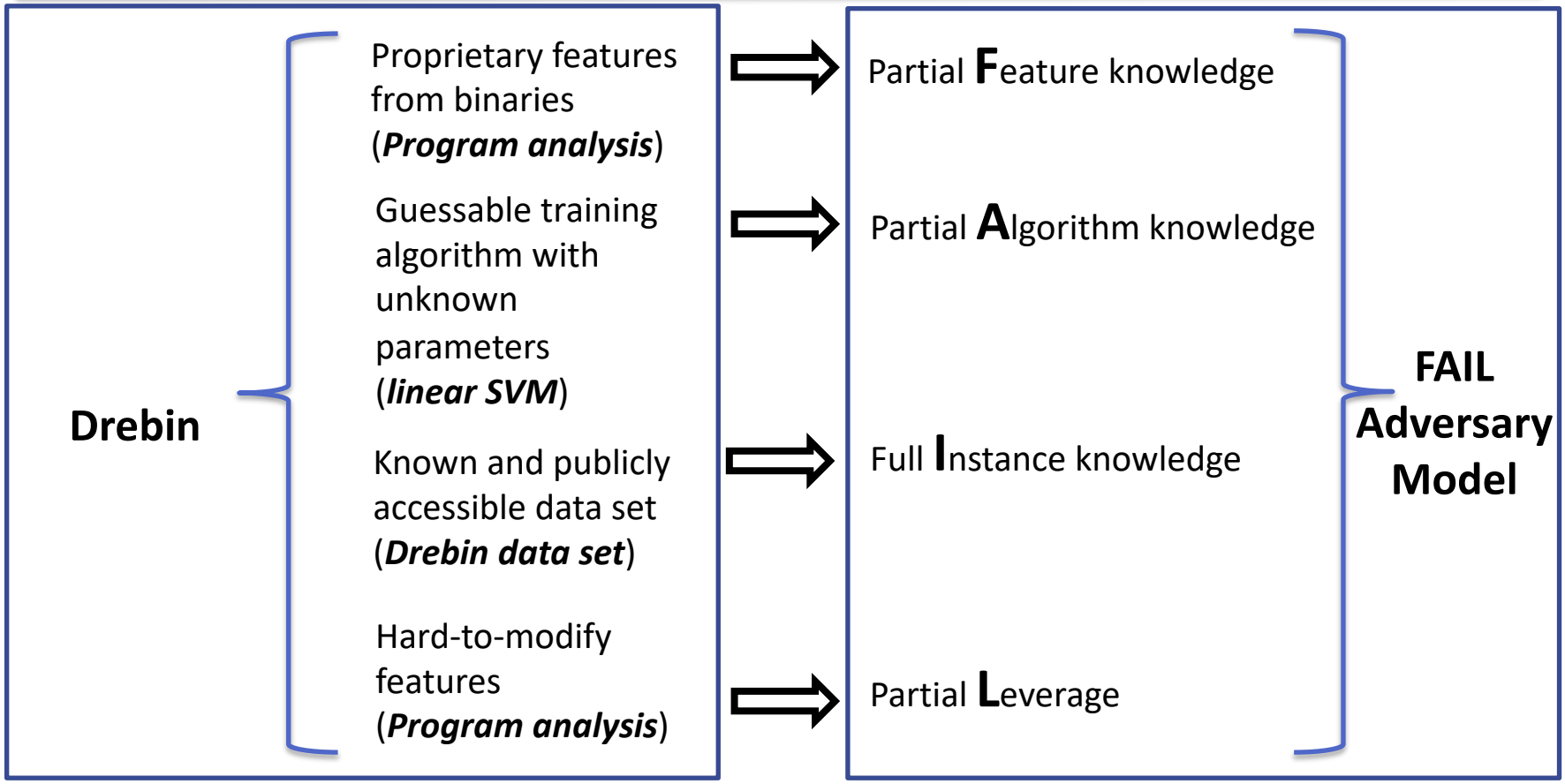
Outline

- **FAIL**
- StingRay
- Evaluation

The FAIL Model

- Models adversaries with variable levels of knowledge and capabilities across four dimensions:
  - **F**eatures
  - **A**lgorithms
  - **I**nstances
  - **L**everage

FAIL in Action



Generalized transferability measures the attacks success rate under realistic knowledge and capabilities assumptions

FAIL - Features

- Models the degree of knowledge about the adversarial features
  - What features can be kept secret?
  - Are the exact feature values known?
- Examples:
  - Unknown program analysis features
  - Unknown image resolution

FAIL - Algorithms

- Models the degree of knowledge about the classifier
  - Is the algorithm class known?
  - Is the training algorithm known?
  - Are the model parameters secret?
- Examples:
  - Unknown linear training algorithm
  - Unknown neural network architecture

FAIL – Instances

- Measures the overlap between the attack and the victim training sets
  - Is the entire training set public?
  - Are some instances known?
  - Are public instances sufficient to train a robust classifier?
- Examples:
  - Unknown malware training set
  - Public image training set

FAIL - Leverage

- Limits the crafting capabilities of the attacker
  - Which feature can be modified by the attacker?
  - Does the attack on some features have side effects?
- Examples:
  - Hard to modify program analysis features
  - Watermarked images

Outline

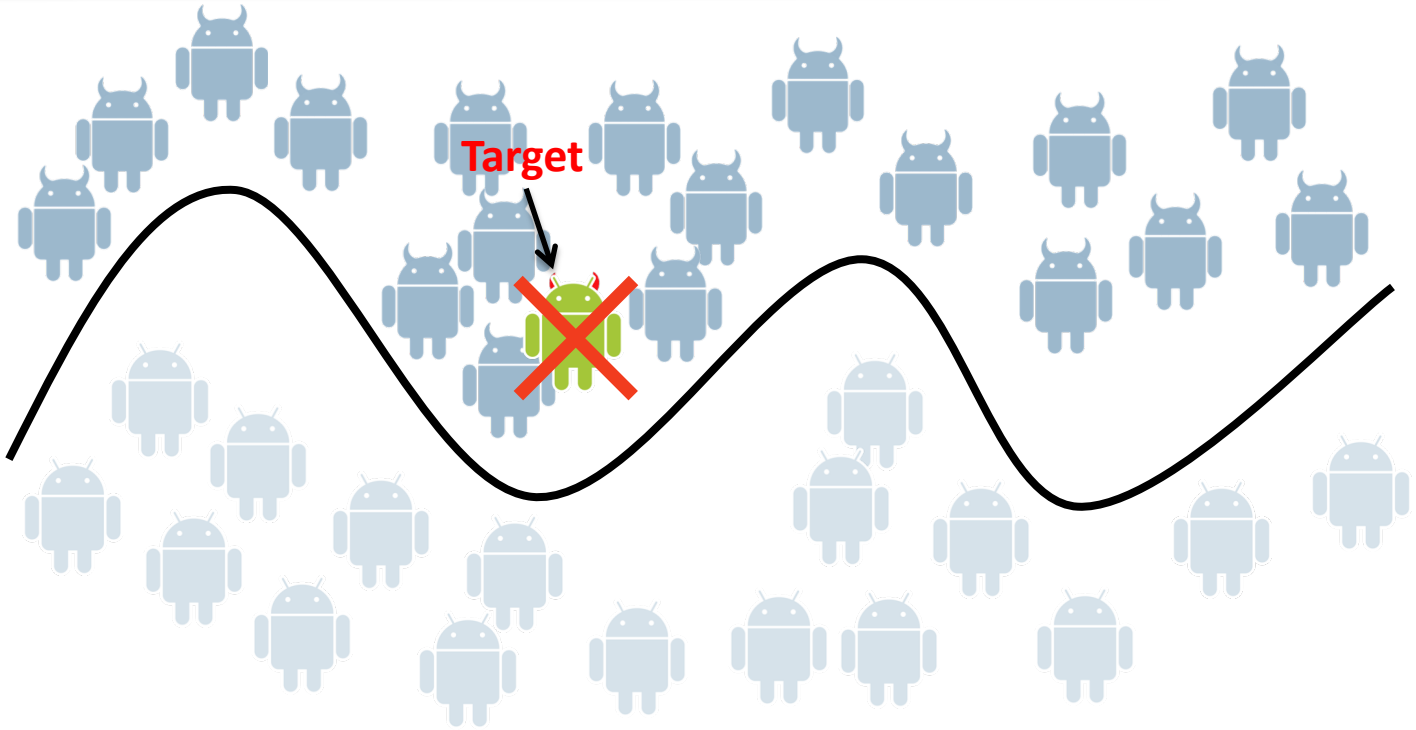
- FAIL
- **StingRay**
- Evaluation

Four Target Applications

- Drebin[1]: Android malware detector based on SVM
- Image classifier: Convolutional Neural Networks
- Twitter exploit predictor[7]: SVM classifier
- Breach predictor[8]: Random Forests on timeseries

[1] Arp et al. "DREBIN: Effective and Explainable Detection of Android Malware in Your Pocket"; 2014
[7] Sabottke et al. "Vulnerability disclosure in the age of social media: exploiting Twitter for predicting real-world exploits"; 2015
[8] Koh et al. "Understanding Black-box Predictions via Influence Functions"; 2017

The Poisoner's Dilemma (1)

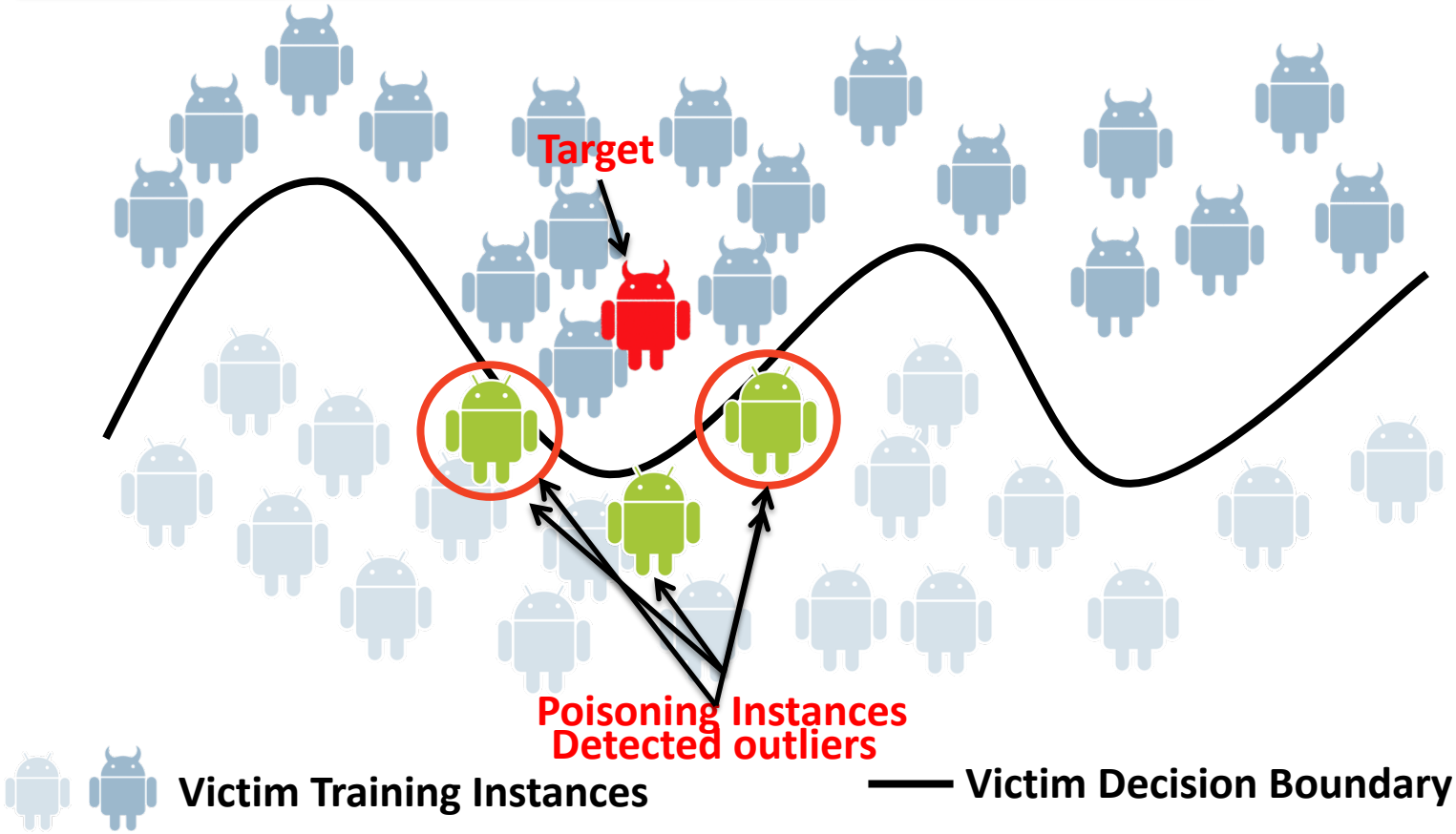


Victim Training Instances

Victim Decision Boundary

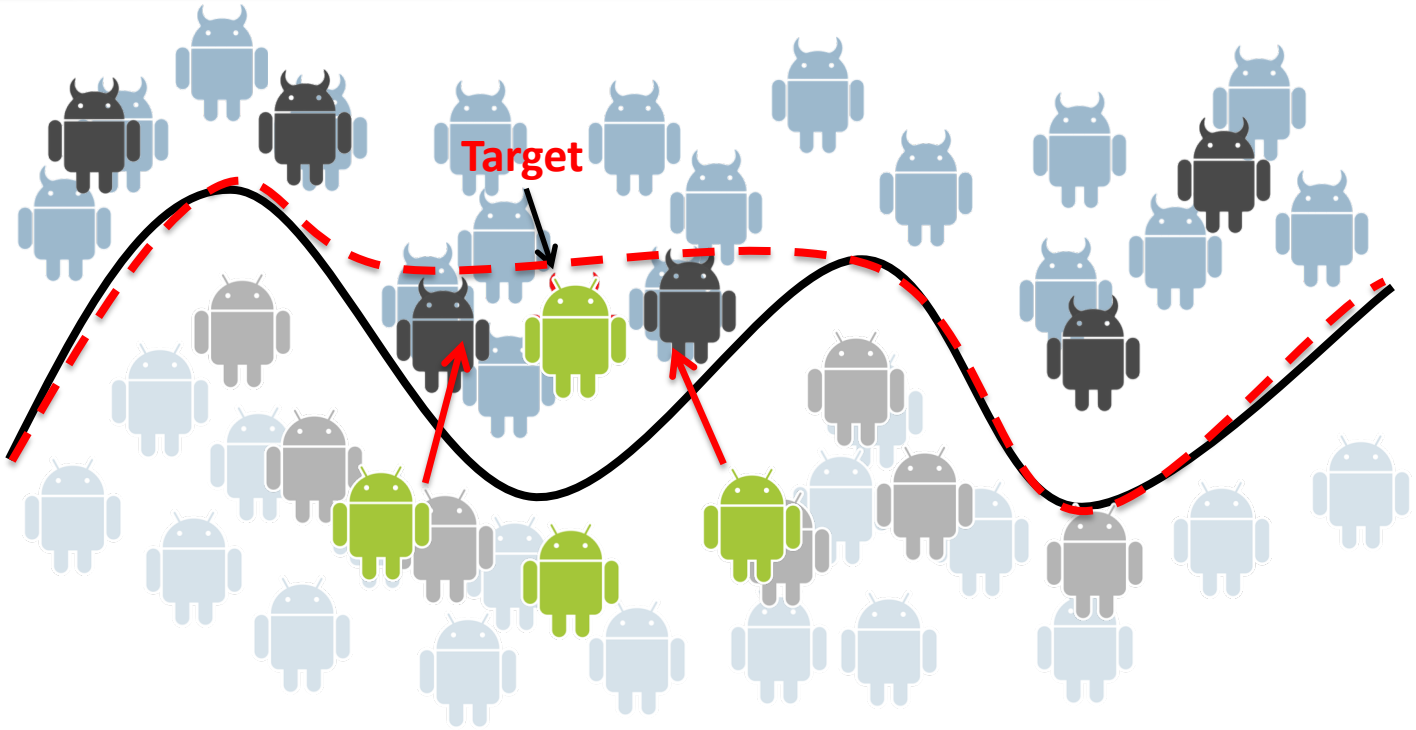
Instance labels cannot be assigned by the attacker

The Poisoner's Dilemma (2)



Poisoning instances could be detected by existing defenses

The Poisoner's Dilemma (3)



Victim Training Instances

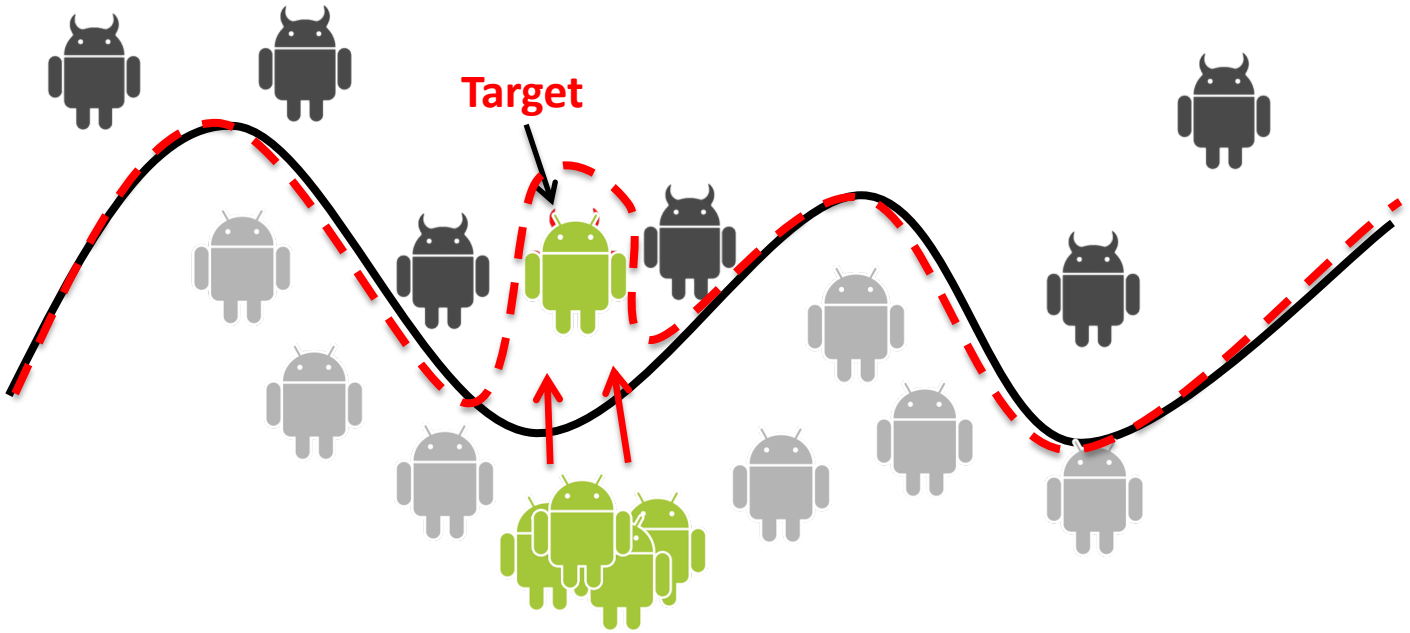
Victim Testing Instances

Victim Decision Boundary

Poisoned Decision Boundary

Poisoning instances could cause collateral, indiscriminate damage

The StingRay Apprach



Victim Testing Instances

Victim Decision Boundary

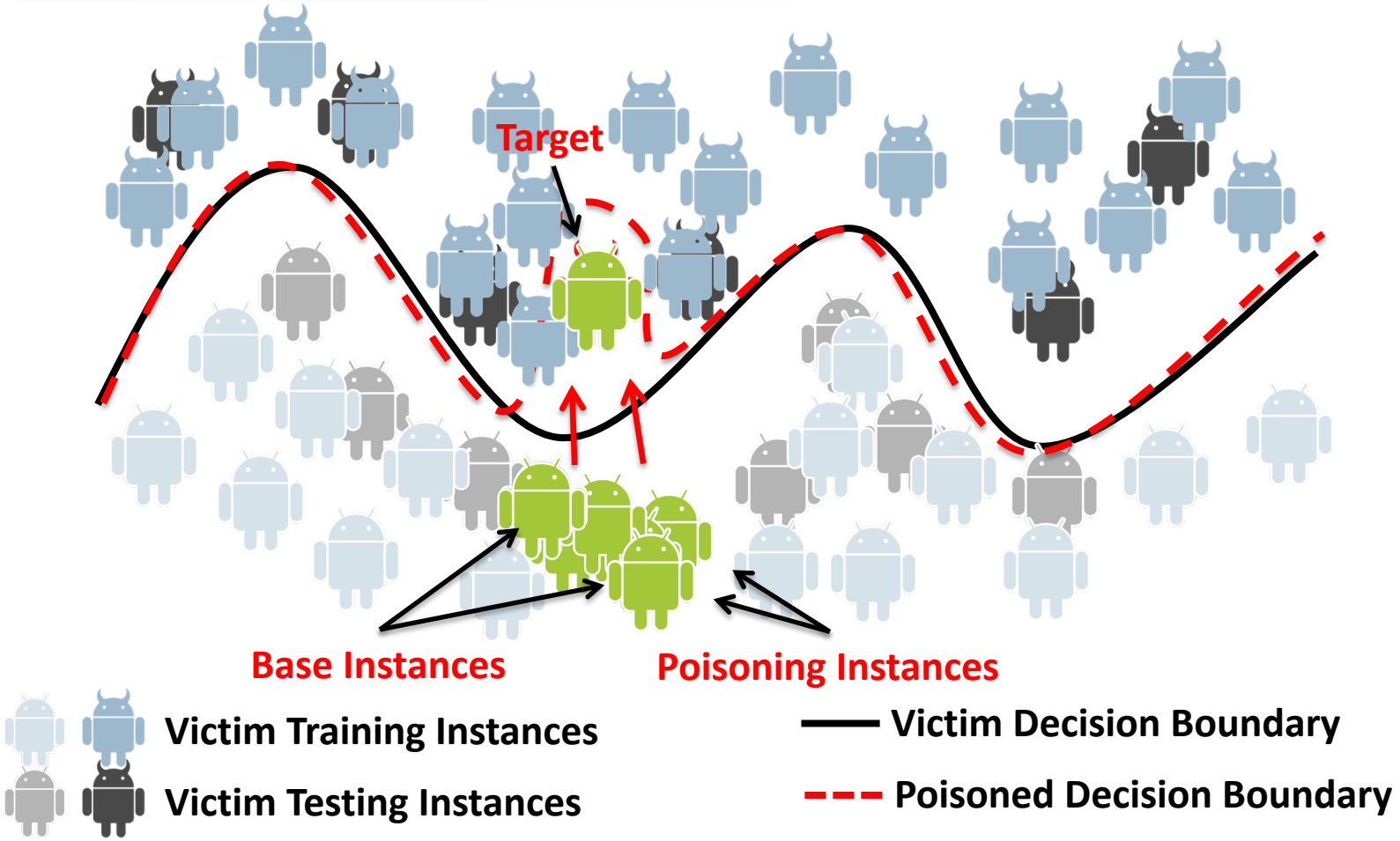
Poisoned Decision Boundary

StingRay achieves both individual and collective inconspicuousness

Attack Requirements

- StingRay design requirements:
  - No assumed control over the labeling function
  - Individually and collectively inconspicuous poisoning
  - Practical FAIL considerations

StingRay High Level Illustration



Crafting Example - Drebin

```
api_call::setWifiEnabled
permission::WRITE_CONTACTS
permission.CALL_PHONE
permission::ACCESS_WIFI_STATE
permission::READ_CONTACTS
intent.action.SEARCH
intent.action.MAIN
```

VirusTotal highlights some features as more **suspicious** than others

malicious label

StingRay – Choosing a Base Instances

```
api_call::setWifiEnabled
permission::WRITE_CONTACTS
permission.CALL_PHONE
permission::ACCESS_WIFI_STATE
permission::READ_CONTACTS
intent.action.SEARCH
intent.action.MAIN
```

```
api_call::setWifiEnabled
permission::ACCESS_WIFI_STATE
activity::MainActivity
permission::READ_CONTACTS
```

Choose base instances with some similarity to target

benign label

StingRay – Individual Inconspicuousness

```
api_call::setWifiEnabled
permission::WRITE_CONTACTS
permission.CALL_PHONE
permission::ACCESS_WIFI_STATE
permission::READ_CONTACTS
intent.action.SEARCH
intent.action.MAIN
```

```
api_call::setWifiEnabled
permission::ACCESS_WIFI_STATE
activity::MainActivity
permission::READ_CONTACTS
```

Reusing existing instances mitigates lack of leverage on some features

StingRay – Collective Inconspicuousness

```
api_call::setWifiEnabled
permission::WRITE_CONTACTS
permission.CALL_PHONE
permission::ACCESS_WIFI_STATE
permission::READ_CONTACTS
intent.action.SEARCH
intent.action.MAIN
```

```
api_call::setWifiEnabled
permission::ACCESS_WIFI_STATE
activity::MainActivity
permission::READ_CONTACTS
```

```
api_call::setWifiEnabled
permission::ACCESS_WIFI_STATE
activity::MainActivity
permission::READ_CONTACTS
```

Poison instances bypass three defenses: RONI, targeted RONI and Micromodels

```
api_call::setWifiEnabled
permission::ACCESS_WIFI_STATE
activity::MainActivity
permission::READ_CONTACTS
```

StingRay – Uncontrolled Labels

```
api_call::setWifiEnabled
permission::WRITE_CONTACTS
permission.CALL_PHONE
permission::ACCESS_WIFI_STATE
permission::READ_CONTACTS
intent.action.SEARCH
intent.action.MAIN
```

```
api_call::setWifiEnabled
permission::ACCESS_WIFI_STATE
activity::MainActivity
permission::READ_CONTACTS
intent.action.MAIN
```

89% of the 19,000 crafted apps are labeled as benign by VirusTotal

```
api_call::setWifiEnabled
permission::ACCESS_WIFI_STATE
activity::MainActivity
permission::READ_CONTACTS
```

```
api_call::setWifiEnabled
permission::ACCESS_WIFI_STATE
activity::MainActivity
permission::READ_CONTACTS
intent.action.SEARCH
```

unknown label

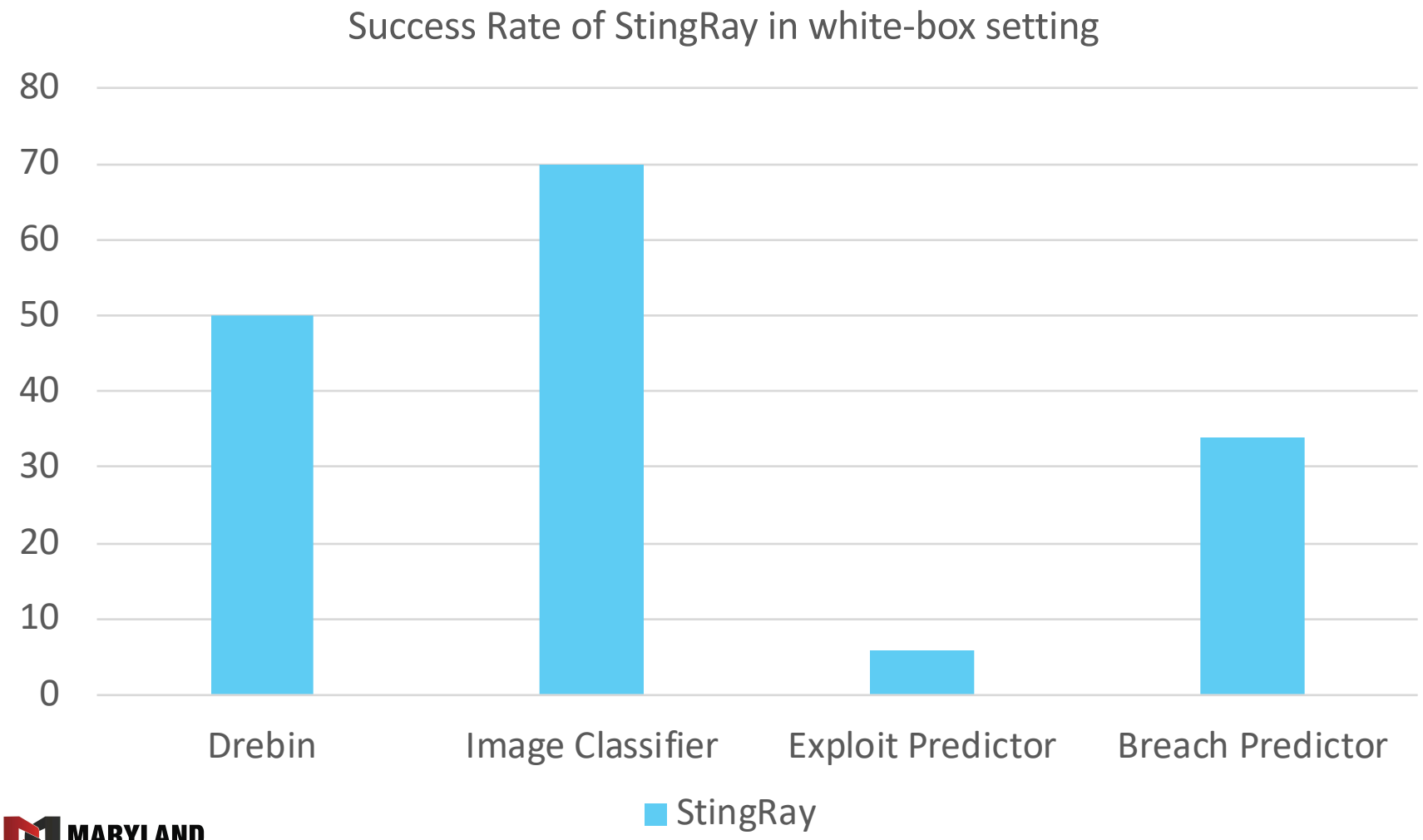
Crafting Example – Neural Networks

- Neural Networks learn features from raw data
- Adapting JSMA[6] for poisoning
  - JSMA pushes instances towards class and not an instance
  - We modify JSMA's objective function to move the poisoning instances towards the target

[6] Papernot et al. "The limitations of deep learning in adversarial settings"; 2016

StingRay - White-Box Performance

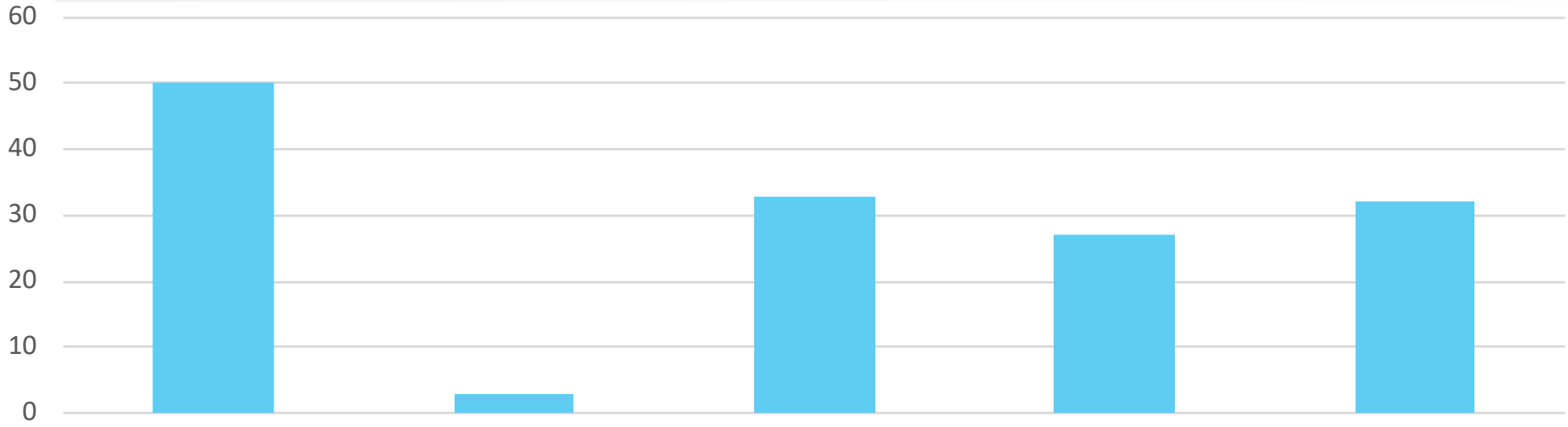
Success Rate (SR): Percentage of attacks that are successful on the victim



Outline

- FAIL
- StingRay
- **Evaluation**

StingRay - Success on Drebin

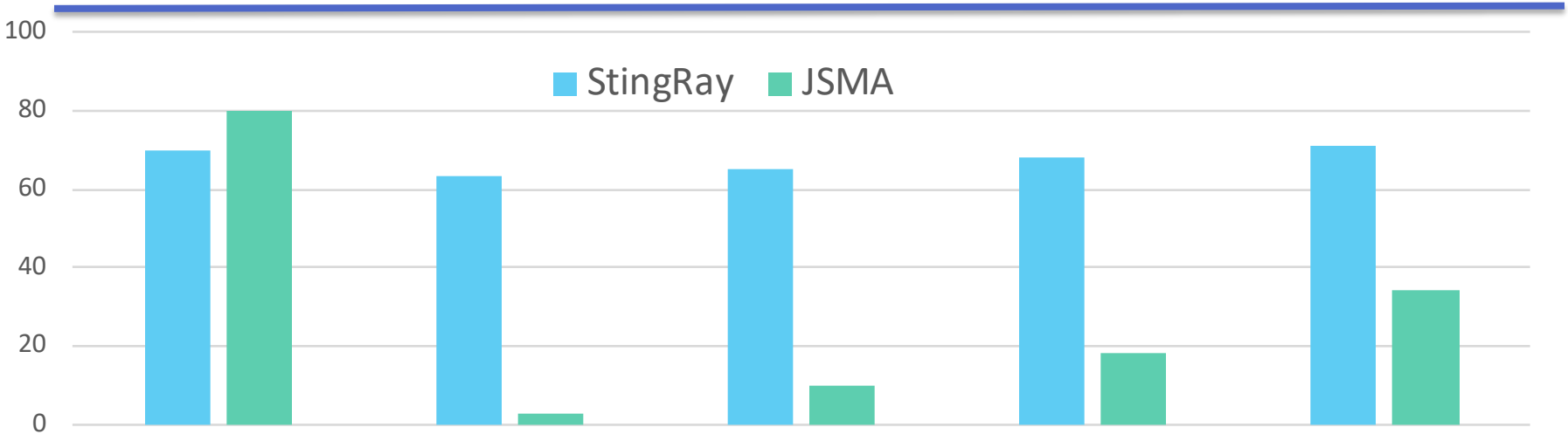


Full	Limited F	Limited A	Limited I	Limited L
Unconstrained	Random 20% features known	SGD algorithm (vs SVM)	10% training set known	60% read-only features

Some **attacks perceived failed** on the surrogate model **are actually successful** on the victim

Feature secrecy appears to be the **most powerful limiting factor**

StingRay and JSMA - Success on the Image Classifier



Full	Limited F	Limited A	Limited I	Limited L
Unconstrained	30-40% center pixels known	CNN missing one layer	70% training set known	40-50% border pixels read-only

StingRay remains **successful on all dimensions**, sometimes even with **increased efficiency** due to a constrained localized strategy

JSMA is more effective in white-box settings, but **performs poorly on all other dimensions**, in contrast to prior observations for Algorithms[9]

[9] Papernot et al. "Transferability in machine learning: from phenomena to black-box attacks using adversarial samples"; 2016

FAIL Captures Adversaries from Prior Work

Prior Work	F	A	I	L
Testing-time attacks				
FGSM Evasion (Goodfellow et al., 2014)	○	○	N/A	○
Model Stealing (Tramer et al., 2016)	●	●	●	○
Genetic Evasion (Xu et al., 2016)	●	●	◔	●
Black-box Evasion (Papernot et al., 2017)	○	●	●	○
C&W Evasion (Carlini et al., 2017)	○	●	N/A	○
Training-time attacks				
SVM Poisoning Attack (Biggio et al., 2012)	○	◔	N/A	○
DNNs Poisoning (Munoz-Gonzalez et al., 2017)	◔	●	●	○
NNs Backdoors (Gu et al., 2017)	◔	●	◔	●
NNs Trojaning (Liu et al., 2017)	◔	●	●	●

● Fully considered ◔ Considered, not evaluated ○ Not considered

Conclusions

- FAIL
  - Models realistic adversarial assumptions
  - Captures existing adversaries
  - Generalizes the notion of transferability
  - Applicable to both evasion and poisoning
- StingRay
  - Targeted poisoning attack
  - Crafts inconspicuous samples
  - No assumed control over the labeling function
  - Implemented against four applications

Thank you!

Octavian Suciu

osuciu.com

osuciu@umiacs.umd.edu

FAIL Framework available at:

ter.ps/fail