GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

Drebin: Efficient and Explainable Detection of Android Malware in Your Pocket

Daniel Arp, Michael Spreitzenbarth, Malte Hübner,
Hugo Gascon, Konrad Rieck

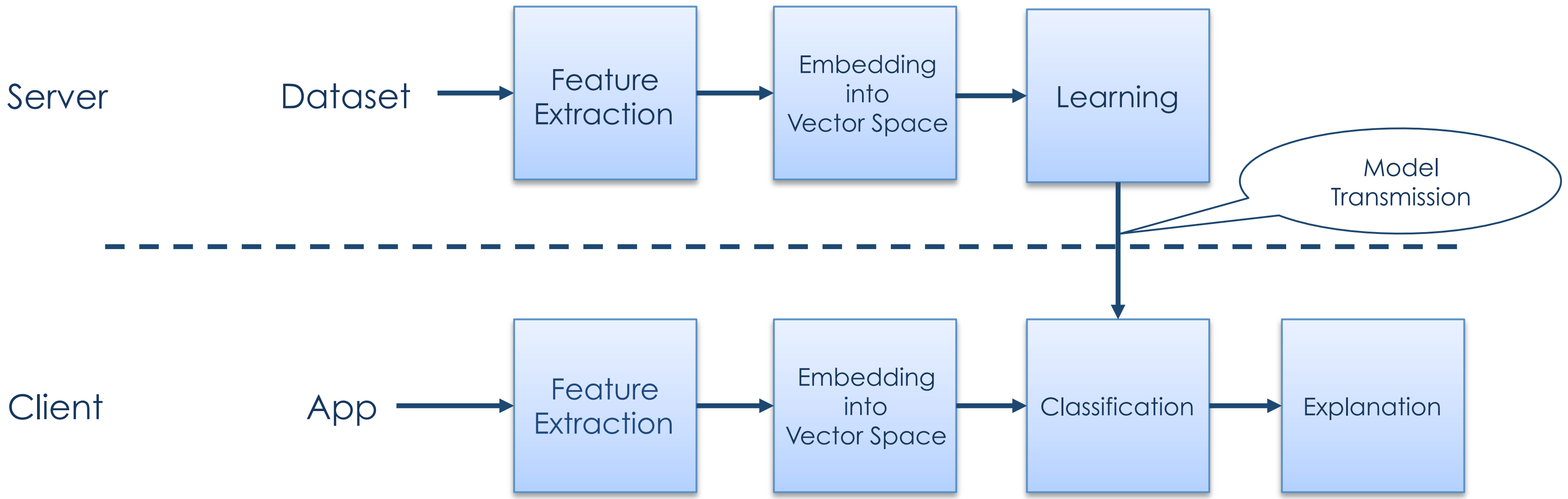
Android-Malware

- Android-Malware
  - Rapid growth in the past few years
  - Mostly distributed through alternative markets
- Mobile Antivirus-Scanners
  - Signature-based detection
  - Unable to identify unknown malware samples

Drebin

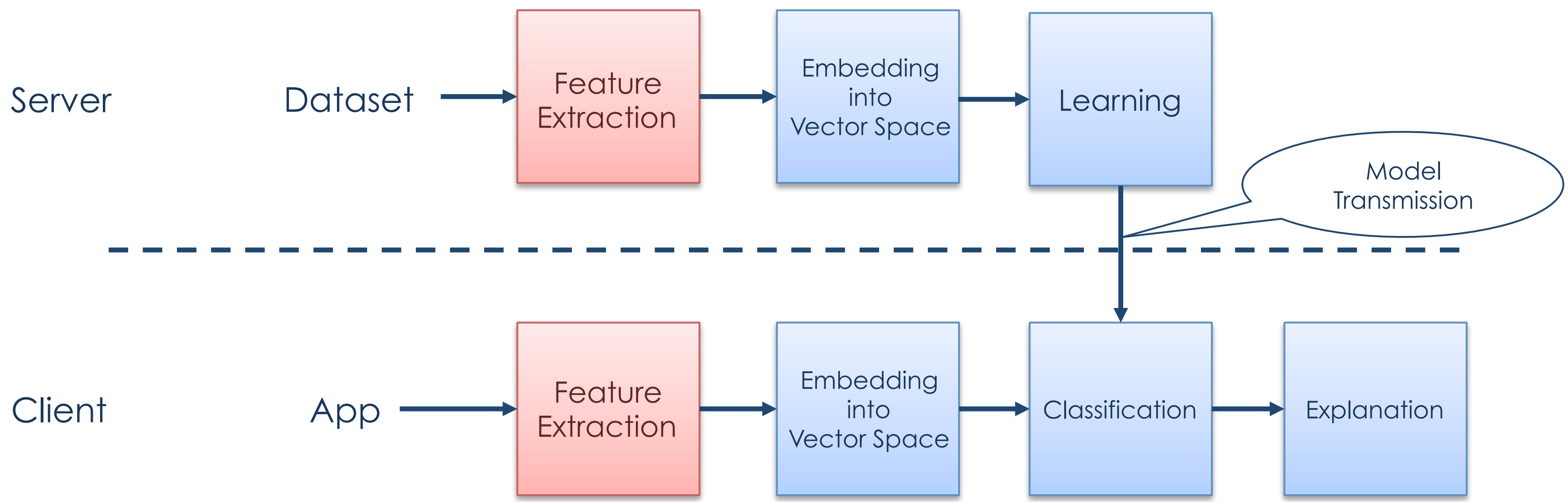
- Detection of unknown malware samples
  - Analysis of known malware
  - Adaptive detection using machine learning techniques
- Detection directly on the smartphone
  - Apps can be installed from many different sources
- Technical Challenges
  - Limited resources of mobile devices

Drebin



Server
Dataset
Feature Extraction
Embedding into Vector Space
Learning
Model Transmission
Client
App
Feature Extraction
Embedding into Vector Space
Classification
Explanation

Drebin


Server
Dataset
Feature Extraction
Embedding into Vector Space
Learning
Model Transmission
Client
App
Feature Extraction
Embedding into Vector Space
Classification
Explanation

Static Analysis

- Lightweight Analysis of Android Applications
  - Extraction of features (strings) from 8 different categories

APK File

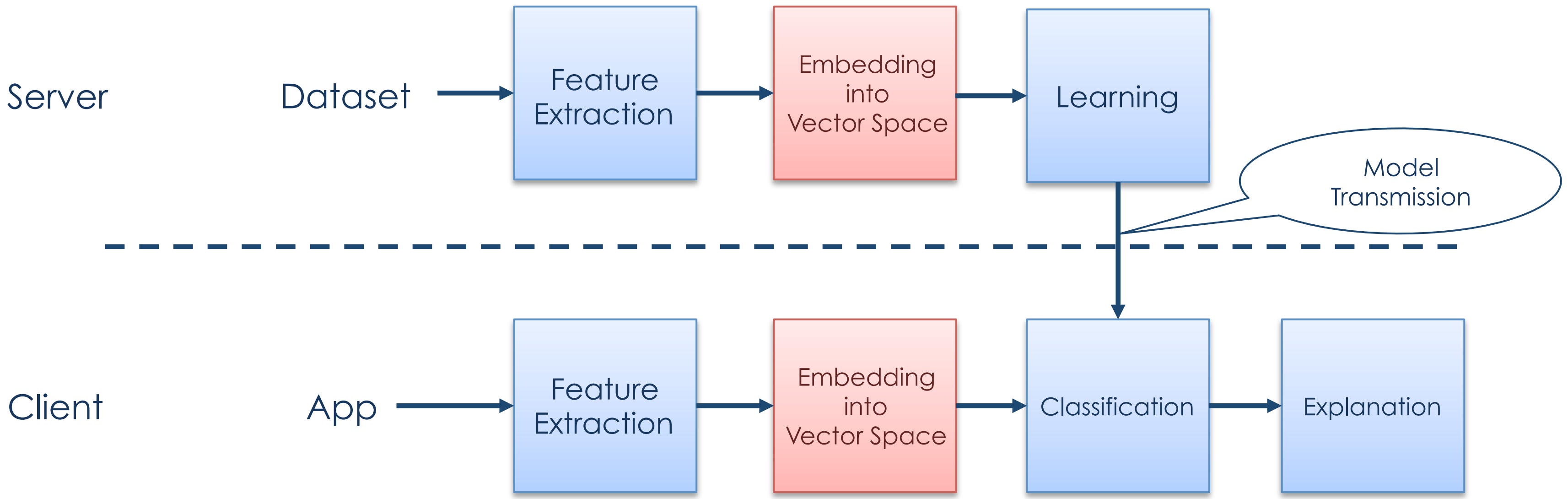
App Components
Filtered Intents
Hardware Components
Requested Permissions

Manifest

Protected API Calls
Used Permissions
Suspicious API Calls
Network Addresses

Dexcode

Drebin



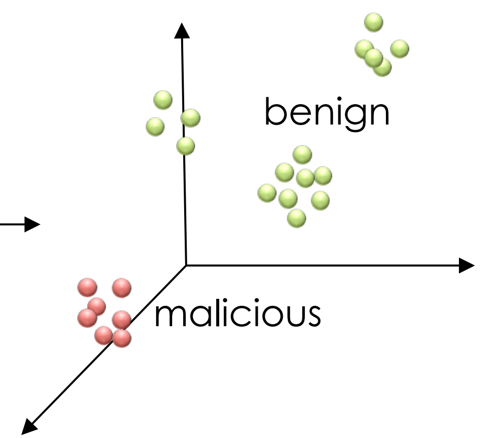
Server
Dataset
Feature Extraction
Embedding into Vector Space
Learning
Model Transmission
Client
App
Feature Extraction
Embedding into Vector Space
Classification
Explanation

Embedding in Vector Space

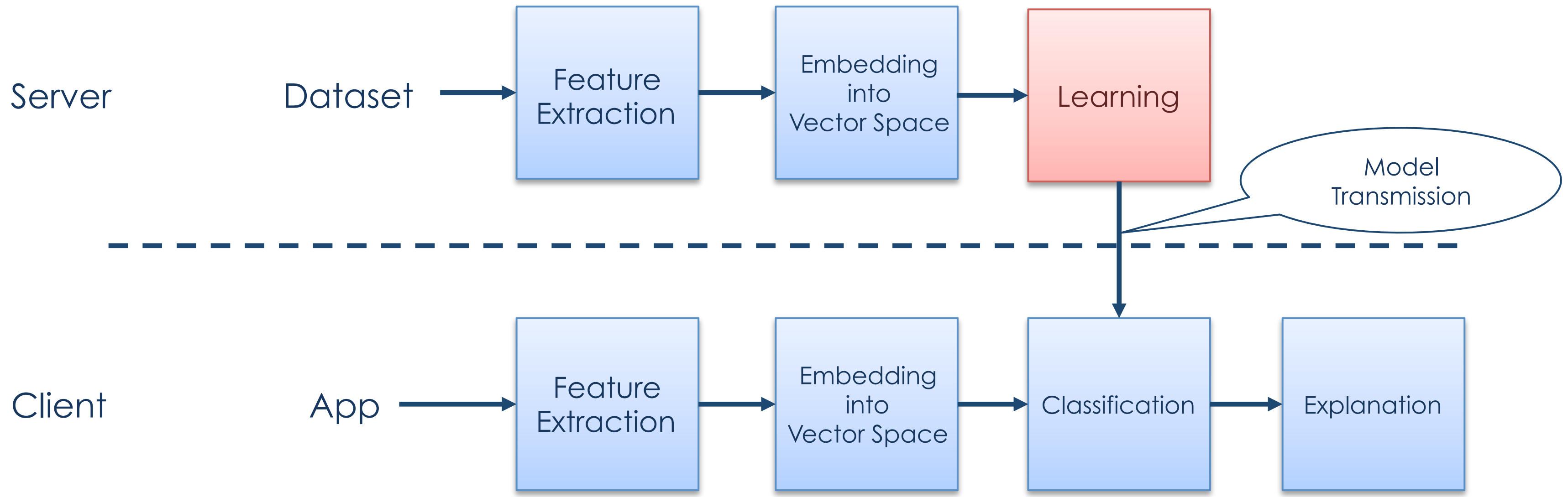
- Embedding of Apps into a vector space

- Vector representation of an App
  - Extracted features are set to 1
  - Small distance between Apps with similar characteristics

$$\xrightarrow{\varphi(x)} \begin{array}{cl} \left[\begin{array}{c} \cdots \\ 0 \\ 0 \\ \cdots \\ 1 \\ 0 \\ \cdots \end{array}\right] & \begin{array}{l} \cdots \\ \textit{hardware :: android.hardware.wifi} \\ \textit{hardware :: android.hardware.telephony} \\ \cdots \\ \textit{permission :: SEND\_SMS} \\ \textit{permission :: DELETE\_PACKAGES} \\ \cdots \end{array} \end{array} \longrightarrow$$



Drebin


Server
Dataset
Feature Extraction
Embedding into Vector Space
Learning
Model Transmission
Client
App
Feature Extraction
Embedding into Vector Space
Classification
Explanation

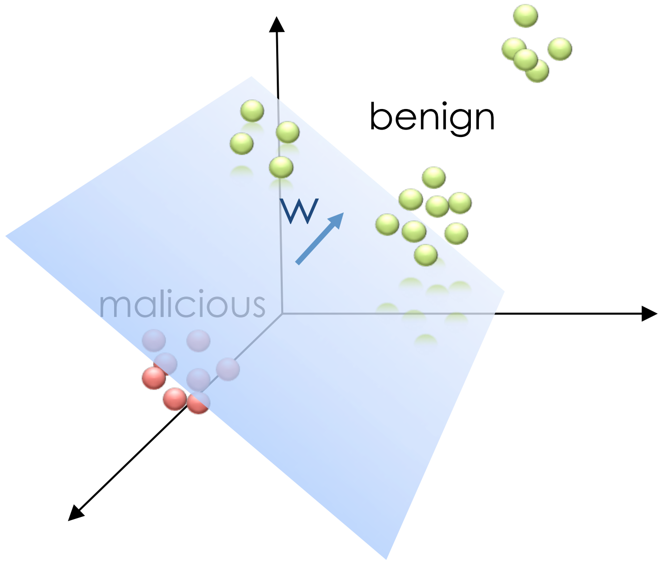
Dataset

- Dataset
  - Training and testing is done on large dataset
  - Collected by Mobile Sandbox project [5]
  - Consists of 123.453 benign and 5.560 malware samples

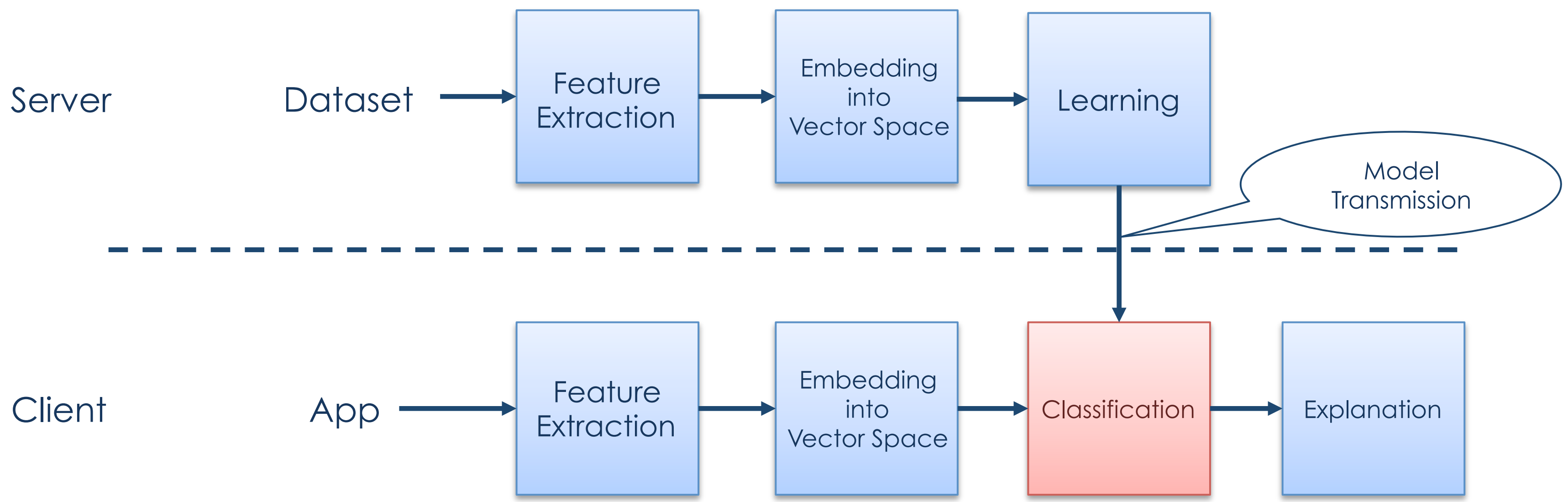
- Malware Samples available at
  - http://user.cs.uni-goettingen.de/~darp/drebin/

Learning

- Linear 2-Class Support Vector Machine
  - Hyperplane, which separates both classes with maximum margin
  - Can be described by model vector w



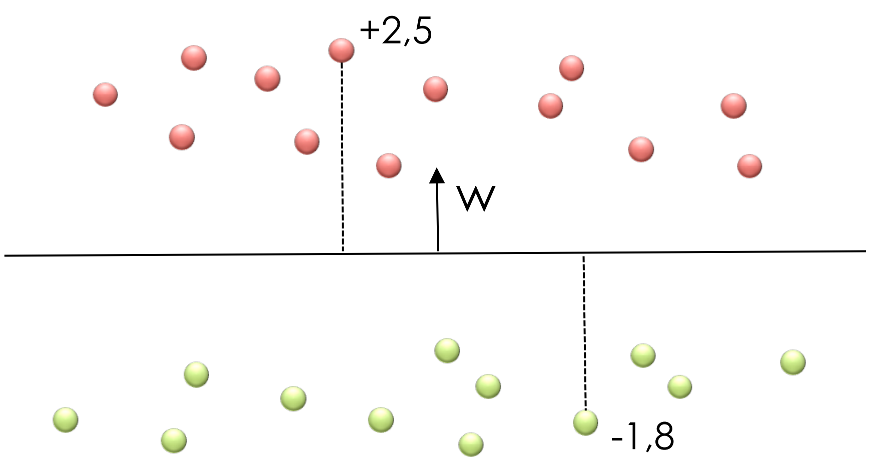
Drebin


Server
Dataset
Feature Extraction
Embedding into Vector Space
Learning
Model Transmission
Client
App
Feature Extraction
Embedding into Vector Space
Classification
Explanation

Classification

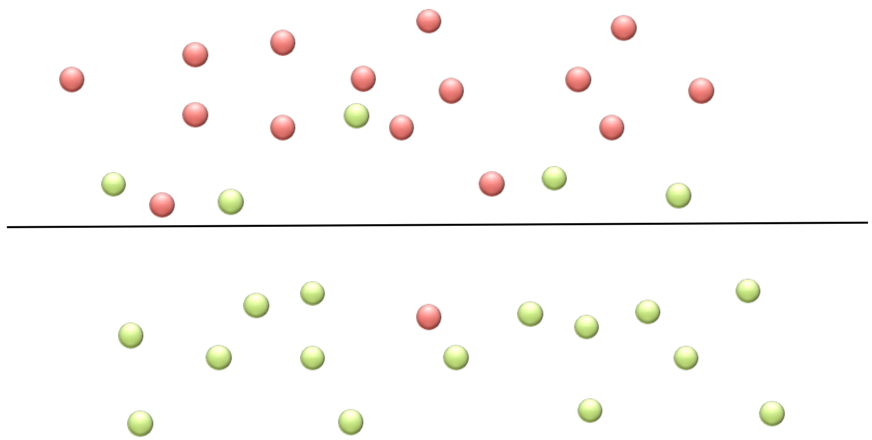
- Classification Score
  - Inner product of model and app vector
  - Sign indicates class of particular sample

$$f(x) = \langle \varphi(x), \vec{w} \rangle$$



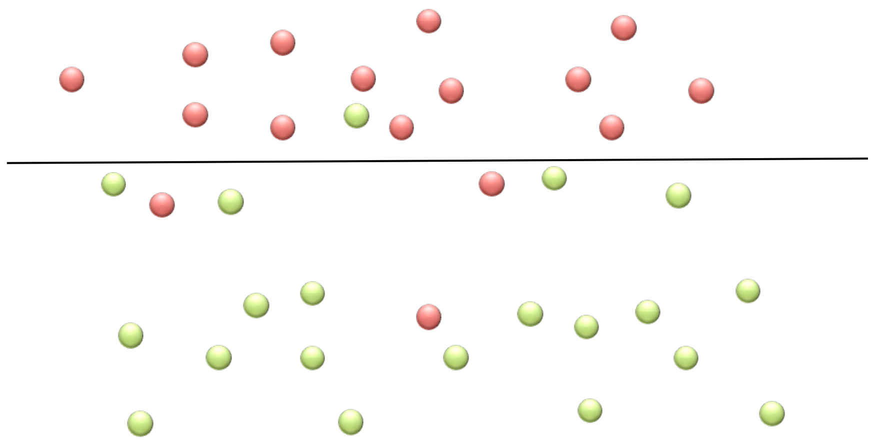
Classification

- Detector Calibration
  - FP-Rate should be less than 1%
  - Choice of threshold unequal to zero

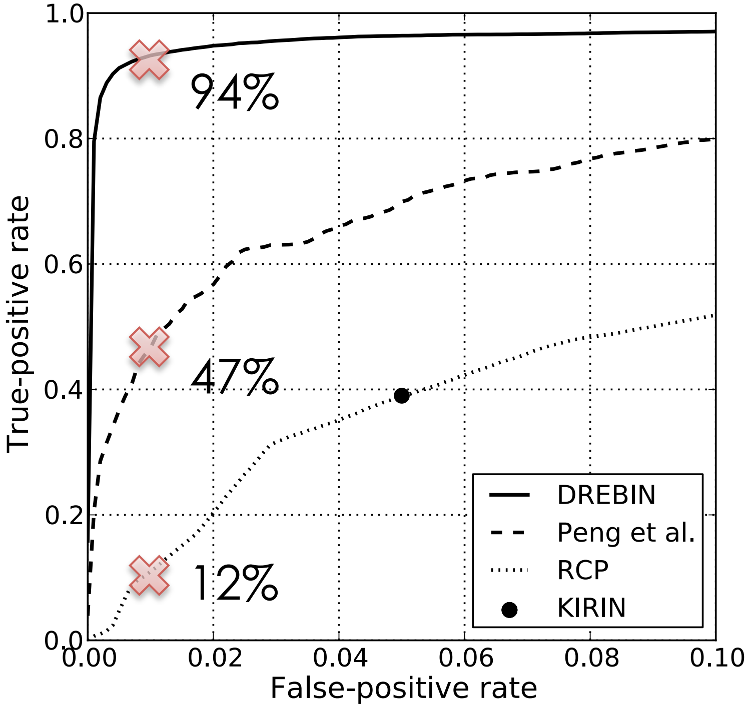


Classification

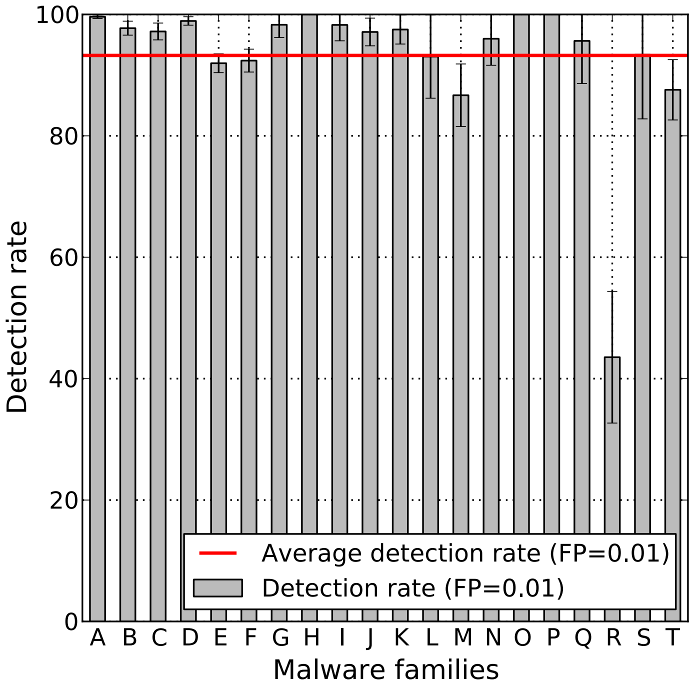
- Detector Calibration
  - FP-Rate should be less than 1%
  - Choice of threshold unequal to zero



Detection Performance

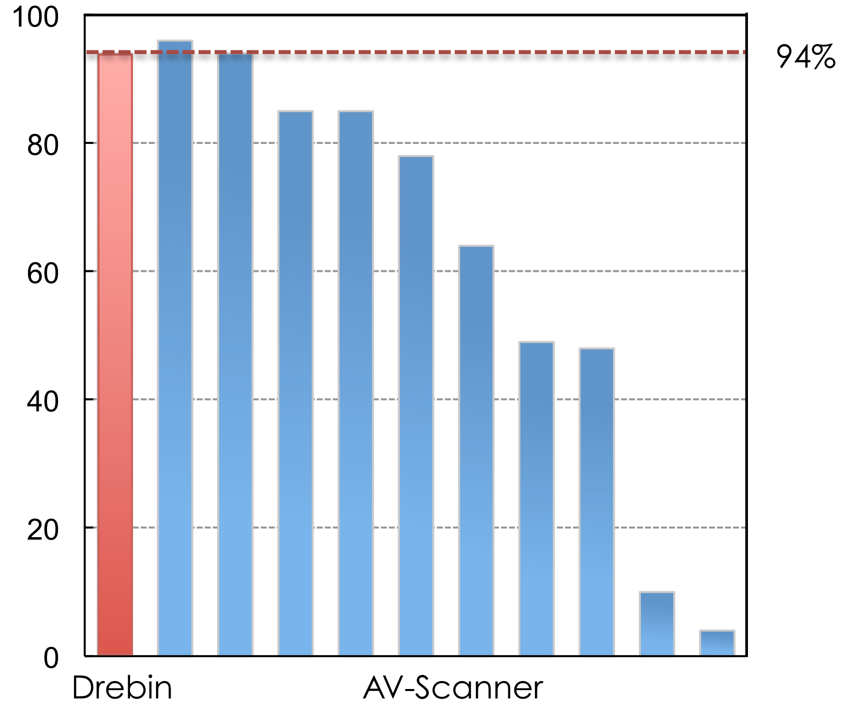


Detection Performance

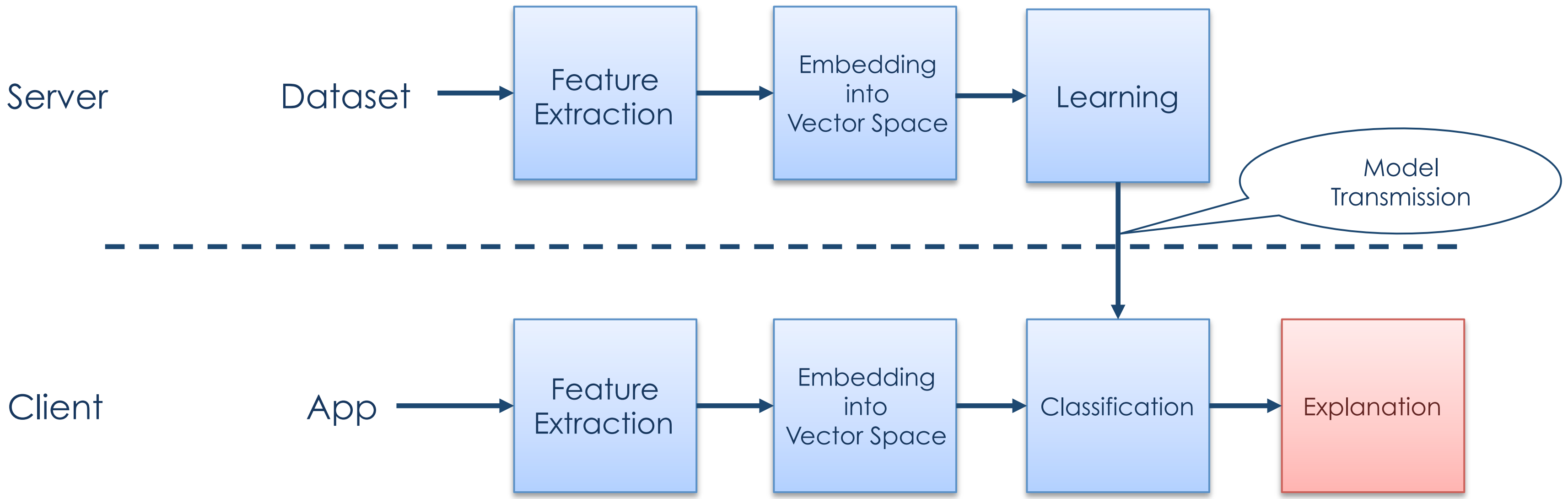


Id	**Family**	**#**	*Id*	**Family**	**#**
A	FakeInstaller	925	*K*	Adrd	91
B	DroidKungFu	667	*L*	DroidDream	81
C	Plankton	625	*M*	LinuxLotoor	70
D	Opfake	613	*N*	GoldDream	69
E	GingerMaster	339	*O*	MobileTx	69
F	BaseBridge	330	*P*	FakeRun	61
G	Iconosys	152	*Q*	SendPay	59
H	Kmin	147	*R*	Gappusin	58
I	FakeDoc	132	*S*	Imlog	43
J	Geinimi	92	*T*	SMSreg	41

Detection Performance



Drebin



Server
Dataset
Feature Extraction
Embedding into Vector Space
Learning
Model Transmission
Client
App
Feature Extraction
Embedding into Vector Space
Classification
Explanation

Explainability

- Interpretation of Results
  - Insights into characteristics of malware
  - Analysis of false positives

- SVM assigns weight to each feature
  - Features with high weight → characteristic for class
  - Only consider features with high weights
  - Interpretation of malware characteristics

Example: DroidKungFu

Feature	Feature Set	Average Weight
SIG_STR	Filtered Intents	2,02
system/bin/su	Suspicious Calls	1,30
BATTERY_CHANGED_ACTION	Filtered Intents	1,26
READ_PHONE_STATE	Requested Permissions	0,54
getSubscriberId()	Suspicious Calls	0,49

Example: DroidKungFu

Feature	Feature Set	Average Weight
SIG_STR	Filtered Intents	2,02
system/bin/su	Suspicious Calls	1,30
BATTERY_CHANGED_ACTION	Filtered Intents	1,26
READ_PHONE_STATE	Requested Permissions	0,54
getSubscriberId()	Suspicious Calls	0,49

Example: DroidKungFu

Feature	Feature Set	Average Weight
SIG_STR	Filtered Intents	2,02
system/bin/su	Suspicious Calls	1,30
BATTERY_CHANGED_ACTION	Filtered Intents	1,26

```
<receiver android:name="com.google.ssearch.Receiver">
    <intent-filter>
        <action android:name="android.intent.action.BATTERY_CHANGED_ACTION" />
        <action android:name="android.intent.action.SIG_STR" />
        <action android:name="android.intent.action.BOOT_COMPLETED" />
    </intent-filter>
</receiver>
```

1. Service is triggered by intent messages

Example: DroidKungFu

Feature	Feature Set	Average Weight
SIG_STR	Filtered Intents	2,02
system/bin/su	Suspicious Calls	1,30
BATTERY_CHANGED_ACTION	Filtered Intents	1,26
READ_PHONE_STATE	Requested Permissions	0,54
getSubscriberId()	Suspicious Calls	0,49

2. Malware tries to gain root access on the device

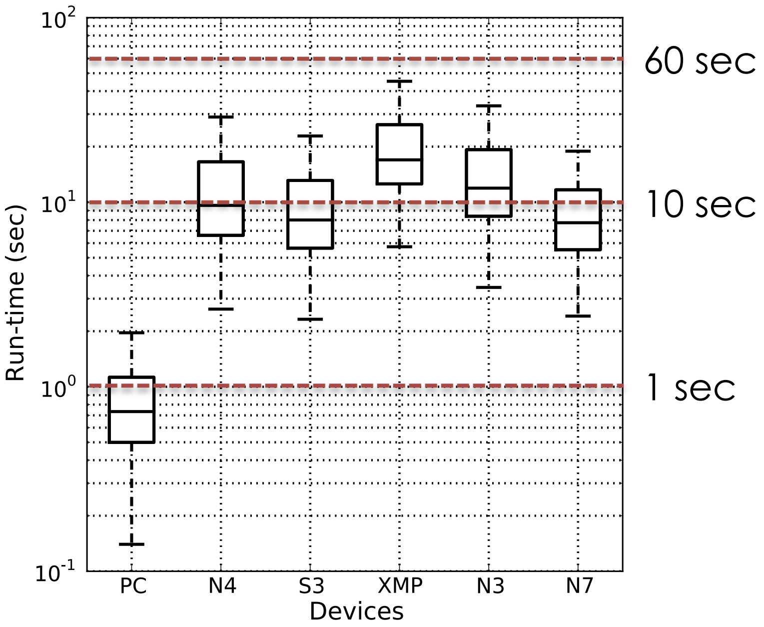
Example: DroidKungFu

Feature	Feature Set	Average Weight
SIG_STR	Filtered Intents	2,02
system/bin/su	Suspicious Calls	1,30
BATTERY_CHANGED_ACTION	Filtered Intents	1,26
READ_PHONE_STATE	Requested Permissions	0,54
getSubscriberId()	Suspicious Calls	0,49

3. Malware steals sensitive data

Run-time Analysis

- Run-time evaluation using prototype implementation
  - Smartphones: Nexus 4, Galaxy S3, Xperia Mini Pro, Nexus i9250
  - Tablets: Nexus 7



Limitations

- Lack of Dynamic Analysis
  - Encryption of payload
  - Loading of malicious code during run-time

- Pollution Attacks
  - Poisoning of dataset by attacker

Conclusion

- Drebin allows reliable detection of Android malware
- Malware can be detected directly on the device
- Explanations are presented to the user

Thanks for your attention!

Questions?

References

- [1] Dissecting Android malware: Characterization and evolution
  - (Zhou and Jiang) (Oakland 2012)
- [2] A study of Android application security
  - (Enck et al.) (USENIX 2011)
- [3] Using probabilistic generative models for ranking risks of Android apps
  - (Peng et al.) (CCS 2012)
- [4] Android permissions: a perspective combining risks and benefits
  - (Sarma et al.) (SACMAT 2012)
- [5] Mobile sandbox: having a deeper look into android applications
  - (Spreitzenbarth et al.) (SAC 2013)

Detection Performance