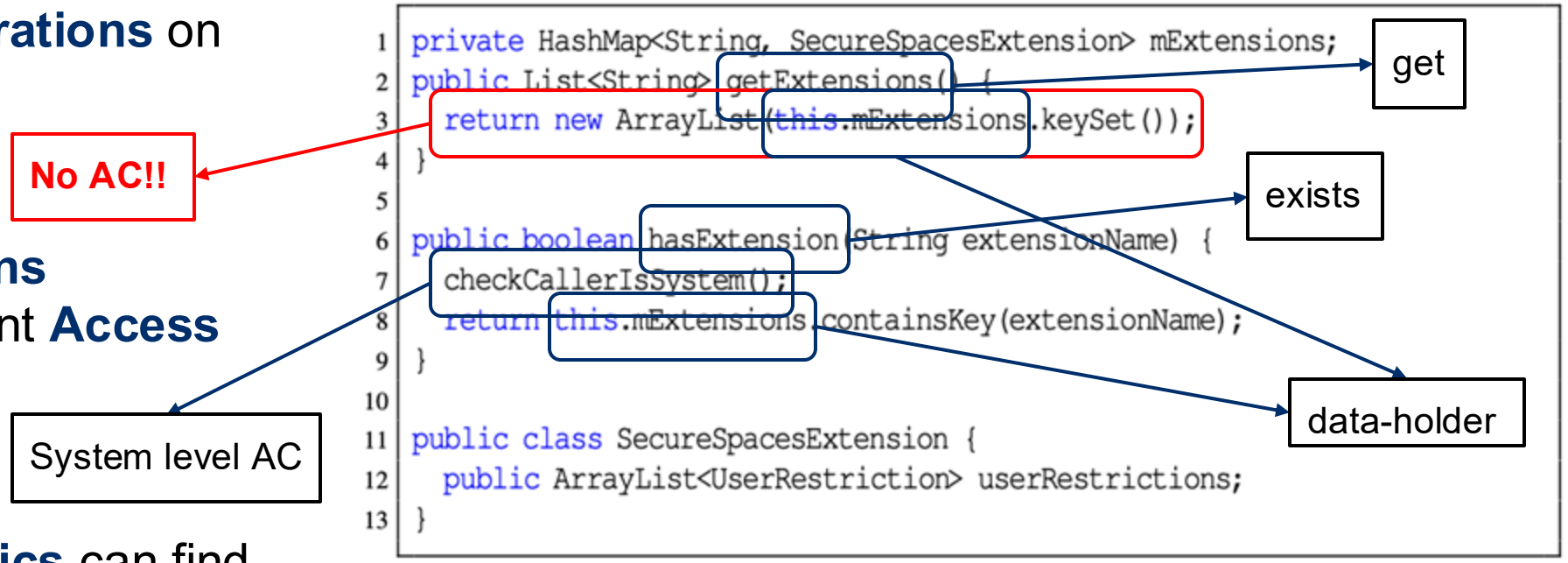
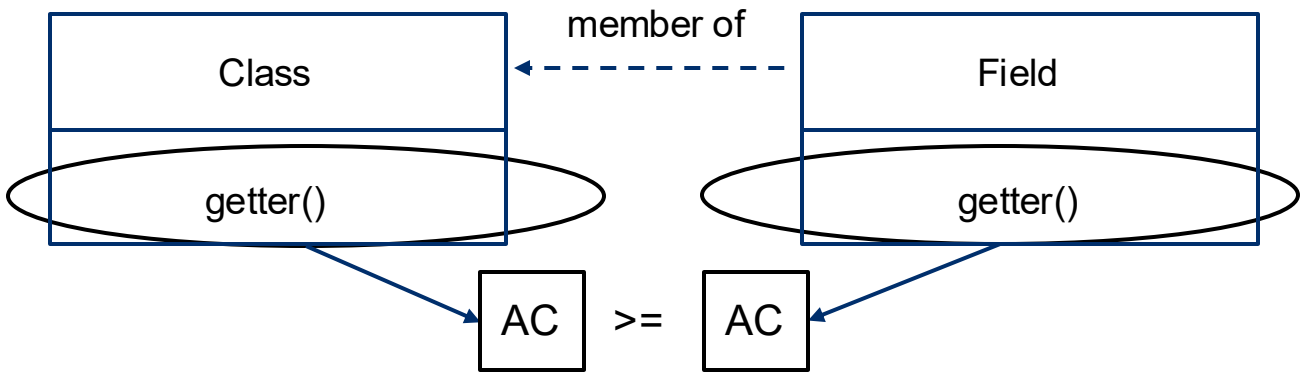
Ariadne: Navigating through the Labyrinth of Data-Driven Customization Inconsistencies in Android

Parjanya Vyas, Haseeb Ur Rehman Faheem, Yousra Aafer, N. Asokan

Motivation – Operation / Java Object Semantics

- **APIs** perform **operations** on **data-holders**
- Different **operations** necessitate different **Access Control**
- Operation **semantics** can find **access control issues**
- **Java Object Semantics** can also help!





Challenges

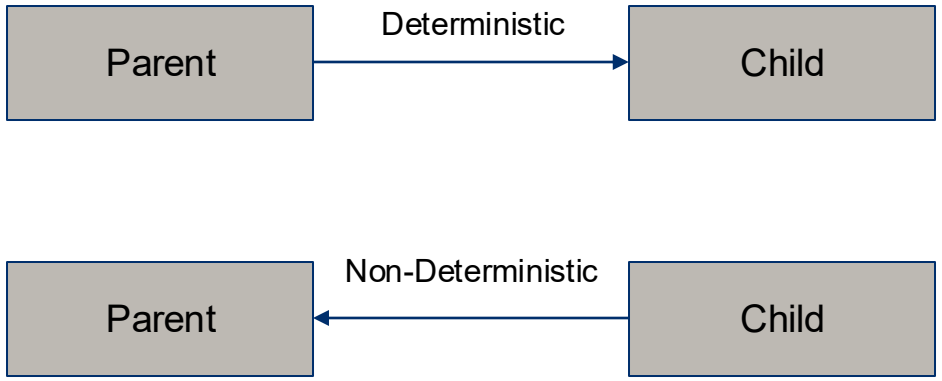
- **TC1: Precise Dataflow Analysis**
  - Field and Object Sensitivity
- **TC2: Defining Analysis Scope**
  - Needs to be selective
- **TC3: Diverse Data Types**
  - Arbitrary structures and Collections

```
1  private ZteAlarmController mZteAlarmController;
2  private ZteLockscreenCleanController mLockscreenCleanController;
3  public void setAlarmAlignType(String name, int type) {
4    enforceCallingOrSelfPermission("android.permission.DEVICE_POWER");
5   mZteAlarmController.setAlarmAlignTypeInternal(name,type);
6  }
7  public void setLockscreenCleanType(String name, int type) {
8    mLockscreenCleanController.setLockscreenCleanInternal(name,type);
9  }
```

```
1  class BaseOptimizerController {
2    private HashMap<String, Integer> mSettingMap;
3    public void setPkgValue(String name, int type) {
4      this.mSettingMap.put(name, Integer.valueOf(type));
5  }
6  class ZteLockscreenCleanController extends BaseOptimizerController {
7    void setLockscreenCleanInternal(String pkgName, int type) {
8      setPkgValue(pkgName, type);
9  }
10 class ZteAlarmController extends BaseOptimizerController {
11   void setAlarmAlignTypeInternal(String pkgName, int type) {
12     setPkgValue(pkgName, type);
13 }
```

Challenges

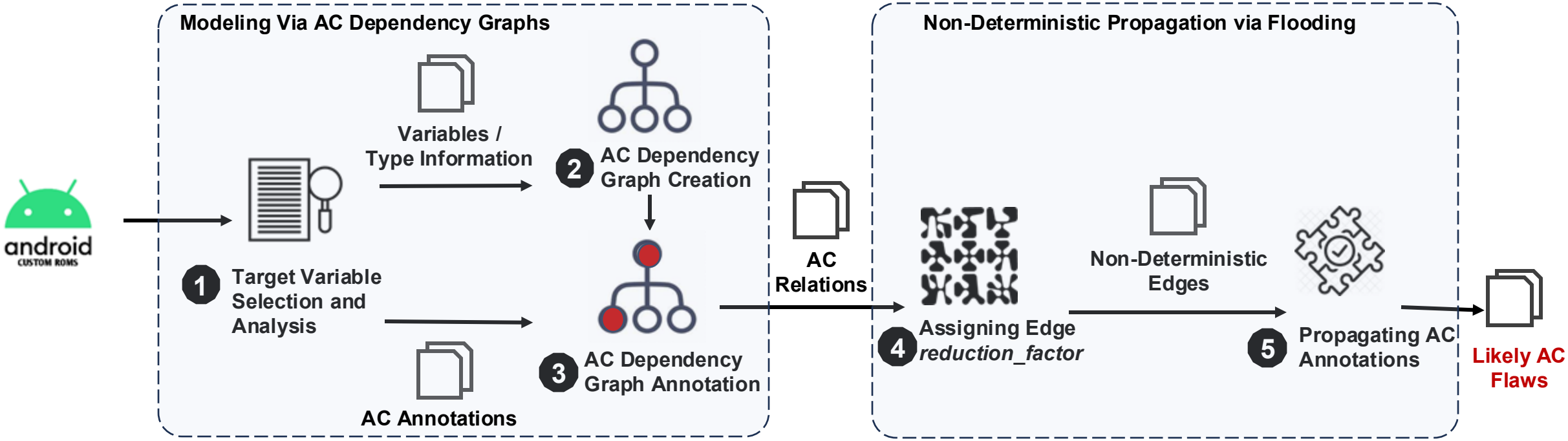
- **TC4: Inherent Ambiguity**
  - AC implications are uncertain



- **TC5: Modeling Operation Semantics**
  - Identifying semantics and granularity

```
public List<String> getDeviceOwnerUserRestrictions() {
  checkCallerIsSystem();
  ArrayList<String> restrictions = new ArrayList<>();
  for (SecureSpacesExtension extention : this.mExtensions.values()) {
    for (UserRestriction restriction : extention.userRestrictions) {
      if (restriction.deviceOwnerOnly) {
        restrictions.add(restriction.name);
        ...
  return restrictions;
}
```

Our Solution – Ariadne



Intuition: Taint Tracking

- Taint tracking
  - Marks sensitive data (“taint sources”) and follows its propagation until it reaches “sinks.”

- Key idea: Systematically trace how dependencies flow through the framework.

- Ariadne
  - Track AC (instead of sensitive data) as they propagate through the system.

Target Variable Selection

- **Excluded Variables**
  - Local
  - Inaccessible to apps
  - Individual primitive
  - Non self-contained

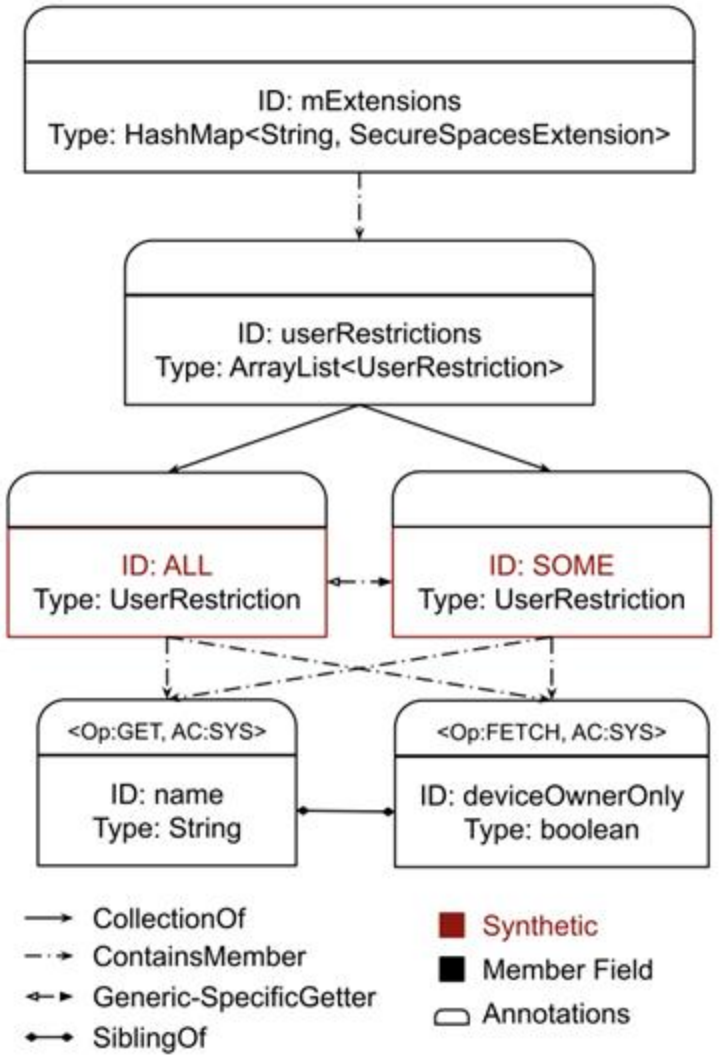
AC Dependency Graph

- **Nodes – Variables**
  - Data holders and their members
  - Synthetic Members for Granularity
- **Edges – AC Implications**
  - Deterministic: Certain relations
  - Non-Deterministic: Uncertain relations

Relation	Edge Propagation Type
Member to containing class	Deterministic
Class to member fields	Non-deterministic
Specific to generic	Non-deterministic
Generic to specific	Non-deterministic
Siblings	Non-deterministic

Graph Creation

```
1  private HashMap<String, SecureSpacesExtension> mExtensions;
2  public List<String> getUserRestrictions() {
3    ArrayList<String> restrictions = new ArrayList<>();
4    for (SecureSpacesExtension extention : this.mExtensions.values()) {
5      for (UserRestriction restriction : extention.userRestrictions) {
6        restrictions.add(restriction.name);
7          ...
8    return restrictions;
9  }
10 public List<String> getDeviceOwnerUserRestrictions() {
11   checkCallerIsSystem();
12   ArrayList<String> restrictions = new ArrayList<>();
13   for (SecureSpacesExtension extention : this.mExtensions.values()) {
14     for (UserRestriction restriction : extention.userRestrictions) {
15       if (restriction.deviceOwnerOnly) {
16         restrictions.add(restriction.name);
17           ...
18   return restrictions;
19 }
20 public class SecureSpacesExtension {
21   public ArrayList<UserRestriction> userRestrictions;
22 }
23 public class UserRestriction {
24   public String name;
25   public boolean deviceOwnerOnly;
```



AC Propagation

- **Selective Flooding**
  - No back-propagation
- **Relevance Weights (Starts with HIGH)**
  - Quantifying relevance of AC to data holders
- **Reduction Factors**
  - Quantifying uncertainty of non-deterministic edges

Edge Definitions and Reduction Factors

Edge ID	Edge Label	Condition	*reduction_factor* (rf)
1	Class-1-Member	$n_1 \in GN \wedge n_2 \in GN \wedge contains(n_1, n_2) \wedge one\_member(n_1)$	0.95
2	Class-m-Members	$n_1 \in GN \wedge n_2 \in GN \wedge contains(n_1, n_2) \wedge m\_members(n_1)$	0.6
3	Siblings	$n_1 \in GN \wedge n_2 \in GN \wedge sibling(n_1, n_2)$	0.6
4	shareSuperType	$n_1 \in GN \wedge n_2 \in GN \wedge same\_supertype(n_1, n_2) \wedge n_1 \neq n_2$	0.7
5	Similarity	$n_1 \in GN \wedge n_2 \in GN \wedge (contains(n_1, n_2) \vee sibling(n_1, n_2))$	$0.6 \times sim(n_1, n_2)$
6	Specific-Generic	$n_1 \in GN \wedge n_2 \in GN \wedge specific\_generic(n_1, n_2)$	0.95
7	Generic-Specific	$n_1 \in GN \wedge n_2 \in GN \wedge specific\_generic(n_2, n_1)$	0.4

Identifying Inconsistencies

- **For Each Data-holder**
  - Inferred AC ranked by eventual relevance weight
- **Propagate AC to APIs**
  - Based on operations on data-holders
- **Select empirically chosen cut-off to identify final AC recommendations**
- **Inferred AC > Actual AC = Inconsistency!!**

Evaluation and Conclusion

- **Evaluated on 13 ROMs (2 AOSP + 11 custom)**
  - Spanning across 8 vendors
  - Android v8-v14
- **Achieves 97.05% accuracy on AOSP ROMs**
  - 96.6% on Pixel 5 and 97.5% on Pixel 6 Pro
- **Identified 90 unique inconsistencies**
  - 13 Proof-of-concept exploits
- **Complements existing tools by covering a gap!**
  - Identifies 30 *new* inconsistencies compared to prior works

Where to Find Us

Ariadne in Open Source
https://doi.org/10.5281/zenodo.15612788

SSG @ University of Waterloo
https://ssg-research.github.io/platsec/probandroid