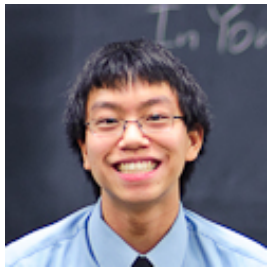
Fork me on Github

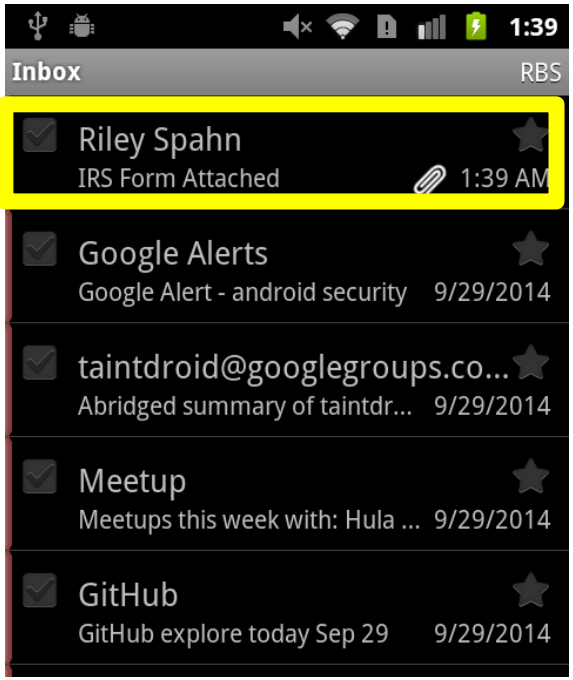
Pebbles: New Data Abstractions for Modern OSes

Riley Spahn, Jonathan Bell, Michael Z Lee, Sravan Bhamidipati, Roxana Geambasu, and Gail Kaiser

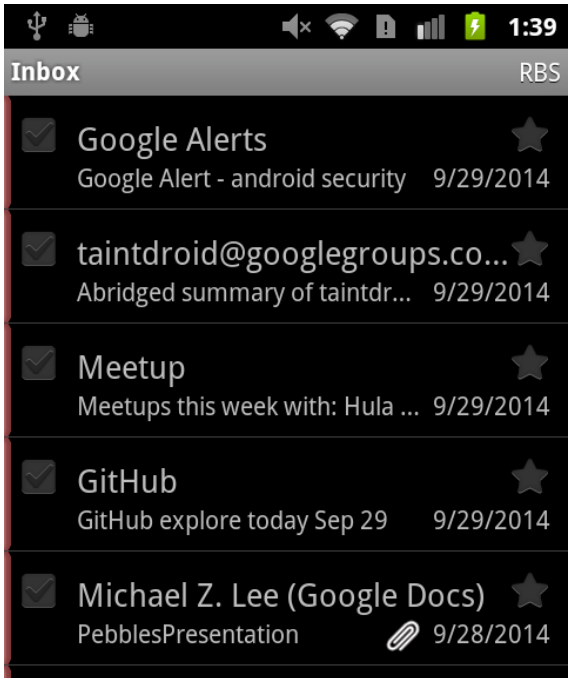




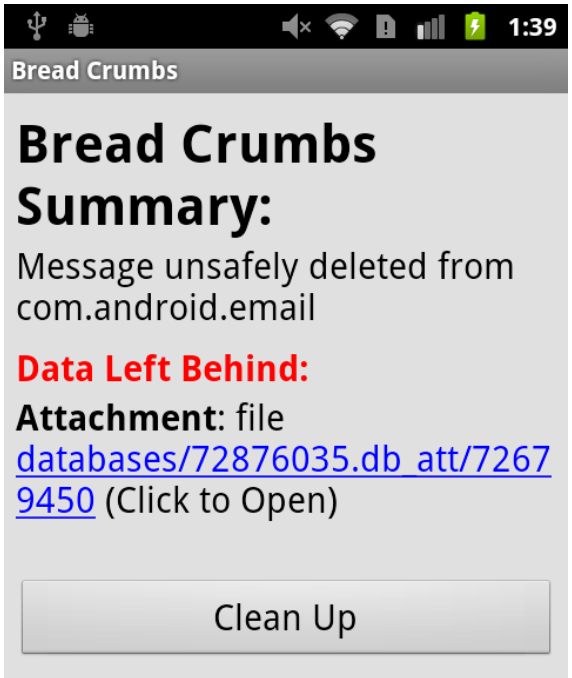

Ex. 1: Is My Email All Gone?



Screenshot of my inbox (contains sensitive email)



Screenshot after deleting email (email doesn't show)



Tool built on Pebbles reveals that attachment is still on the file system

Ex 2: What Was Accessed?

File Access Log

```
Sep 27 09:32:23 access /data/data/com.
documents/databases/documents.db
Sep 27 09:32:58 access /data/data/com.
documents/cache/af712e9431
Sep 27 09:33:15 access /data/data/com.
documents/databases/documents.db
Sep 27 09:33:25 access
/data/data/com.todo/databases/todo.db
Sep 27 09:33:31 access
/data/data/com.todo/databases/todo.db
Sep 27 09:33:50 access
/data/data/com.todo/databases/todo.db
```

Object Access Log

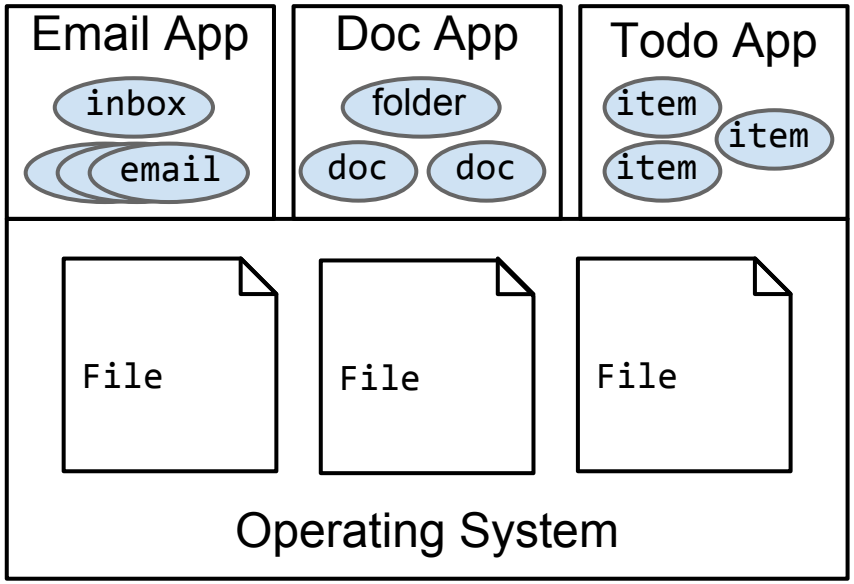
```
Sep 27 09:32:23 access App “com.
documents” accessed Document with
Title ”Pebbles Paper”
Sep 27 09:32:58 access App “com.
documents” accessed file
/data/data/com.
documents/cache/af712e9431 of
Document with Title ”IRS Form
2014”
Sep 27 09:33:15 access App “com.
documents” accessed Document with
Title “Cookie recipe”
Sep 27 09:33:25 access App “com.
todo” added Item with Description
“Write paper”
Sep 27 09:33:25 access App “com.
todo” added Item with Description
“Bake cookies”
Sep 27 09:33:31 access App “com.
todo” updated Item with
Description “Finish Pebbles talk”
```

What is that?!

Problem: Misaligned Abstractions

- OSes lack appropriate abstractions for managing data
  - OSes see low-level, opaque files and directories
  - Users think of higher-level objects, like emails, docs
- Therefore, we must rely on apps to manage our data
  - To delete, appropriately secure, replicate our objects
- That's dangerous, given bugs, insecurities in apps

Our question: Can we teach OSes about these higher-level objects?



Pebbles

- OS-level support for persistent data management at a new level of abstraction: logical data objects (LDOs)

- Examples:
  - Email app: emails (to/from/body/attachment), inboxes, account
  - Document app: documents, folders
  - Personal finance app: bank accounts, transactions

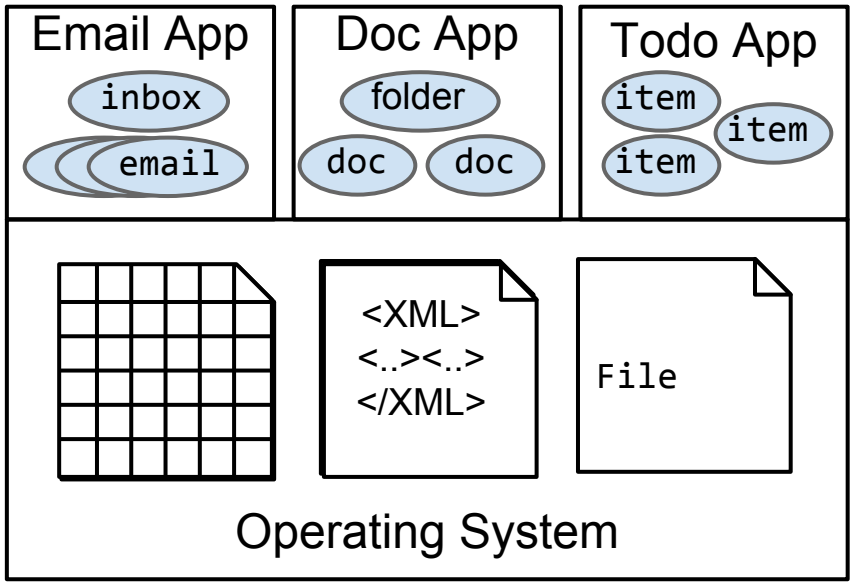
- New tools can leverage LDOs to increase users' visibility and control over data management
  - E.g.: BreadCrumbs, ObjectLog, ...

Design Goals

- LDOs should be fine-grained
  - Individual emails, documents
- Accurate and precise object recognition
  - Avoid leakage (high recall)
  - Avoid overinclusion (high precision)
- No new APIs, no app modifications
  - Assume developers follow common practices

Seem Impossible? It's Not

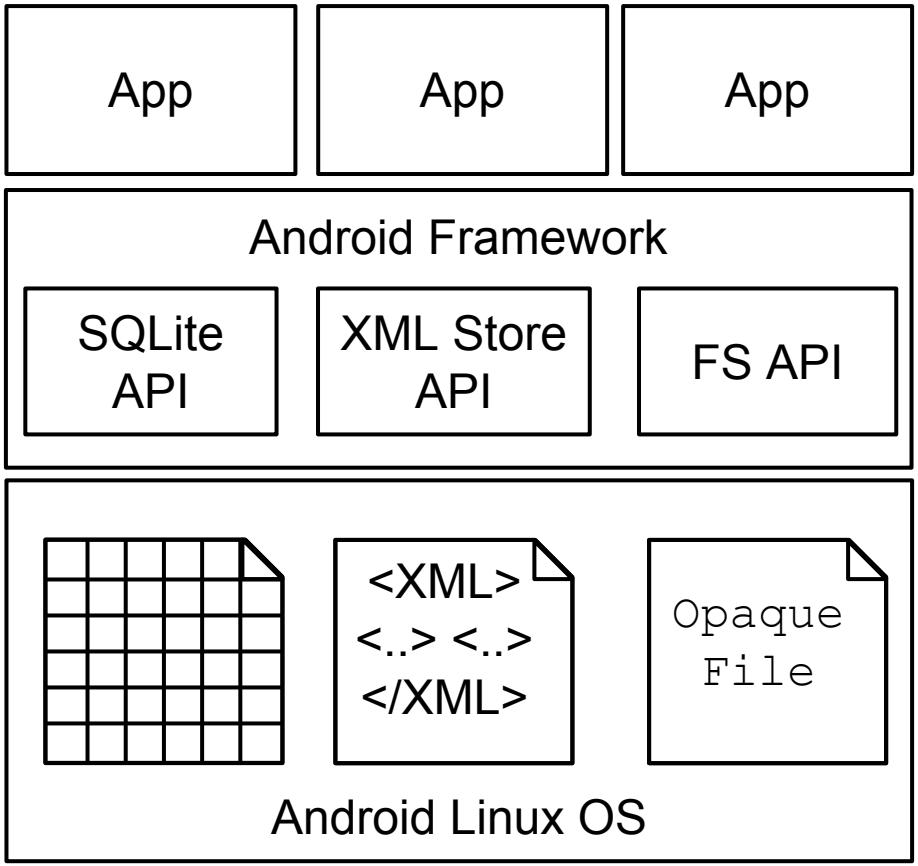
- Applications can use arbitrary file formats or mappings between objects and files
- Insight: modern OSes come with high-level storage abstractions (e.g., DBs,ORMs), which add uniformity and structure to data on disk



Outline

- Motivation
- Android Storage Study
- Pebbles Architecture
- Pebbles-based Tools
- Evaluation
- Conclusions

Android's Storage Abstractions



Measurement Study

- We performed studies on two datasets
  - 470K application static study
  - 98 application in depth dynamic study

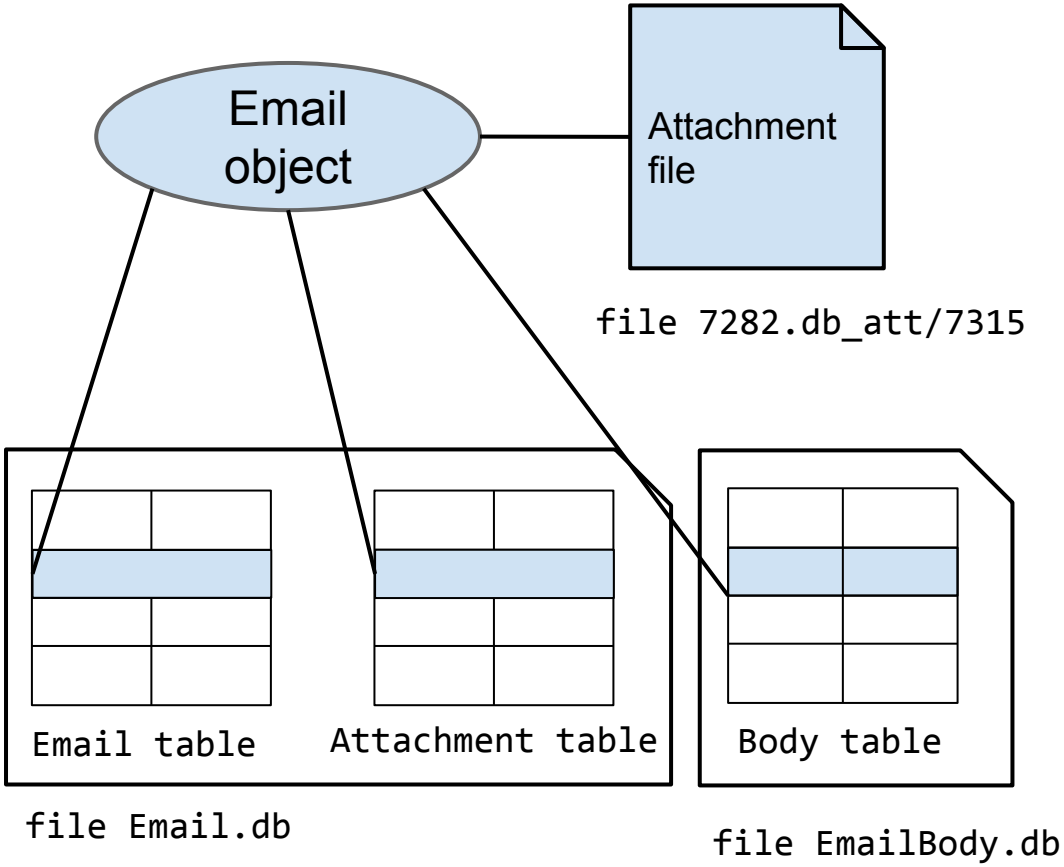
- Questions:
  - What storage abstractions do applications use?
  - How do individual applications organize their data?
  - Do applications use multiple storage abstractions?

Result 1: Android Abstractions Pervade

Storage Library	Top 100 Free Apps	476,375 Free Apps
SQLite	86	205,655
ORMLite	0	6,846
SQLCipher	0	168
DB40	0	116
H2	0	18
Perst	0	14
Sybase	0	12
Neodatis	0	8

Result 2: Objects Span Multiple Abstractions

Abstraction	98 Popular Free Apps
SQLite	83
Raw File System	44
XML Store	23
No Storage	5



Outline

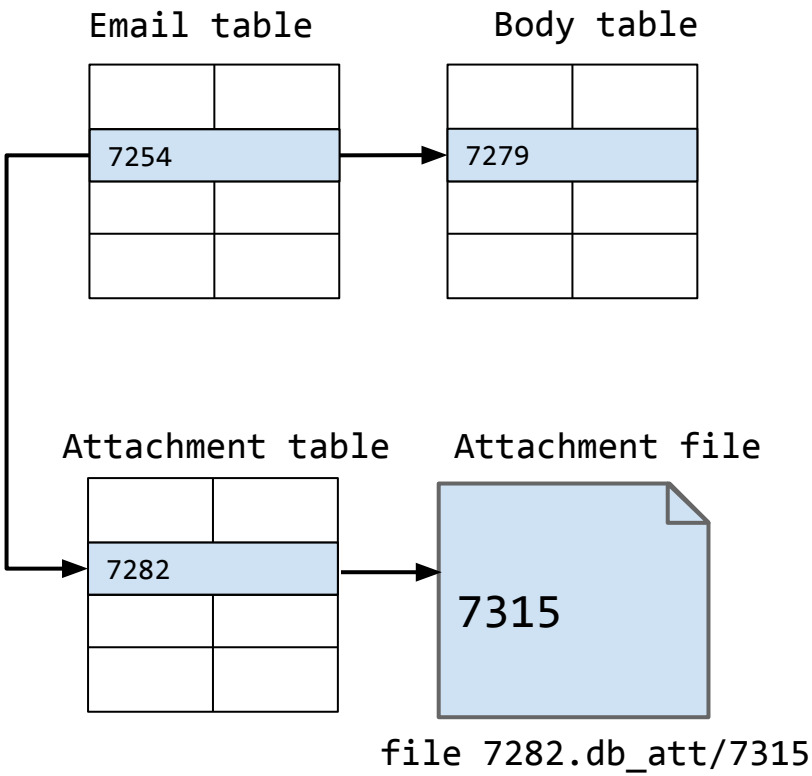
- Motivation
- Android Study
- The Pebbles Architecture
- Pebbles-based Tools
- Evaluation
- Conclusions

Pebbles Overview

- Plugs into popular storage abstractions and extracts inherent structure to reconstruct LDOs
  - SQLite DB, XML store, raw file system APIs
- Builds a device-wide object graph
  - Nodes: each DB row, XML item, raw file
  - Directed edges: relationships between objects (e.g., the email attachment file node “belongs to” the email node)
- An LDO is defined as the transitive closure of a node

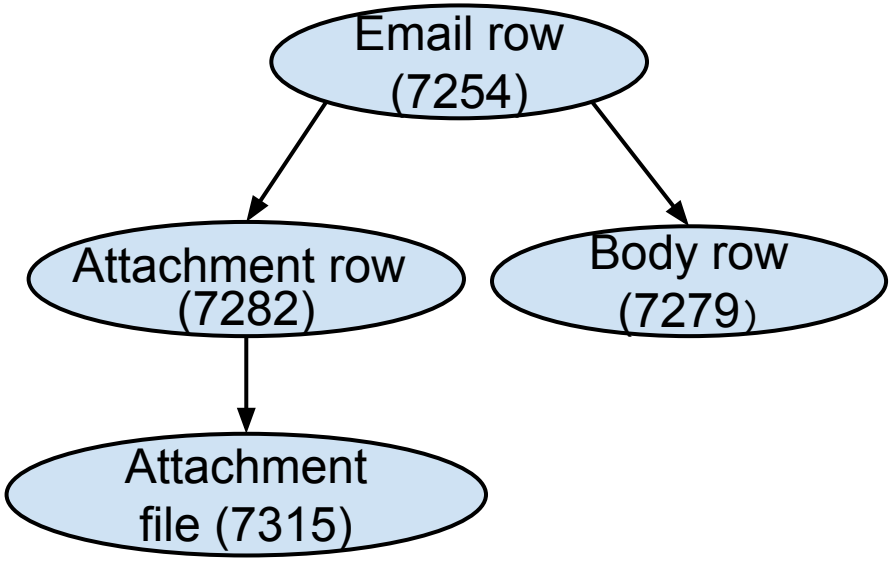
Example: The Email LDO

Data's layout on disk

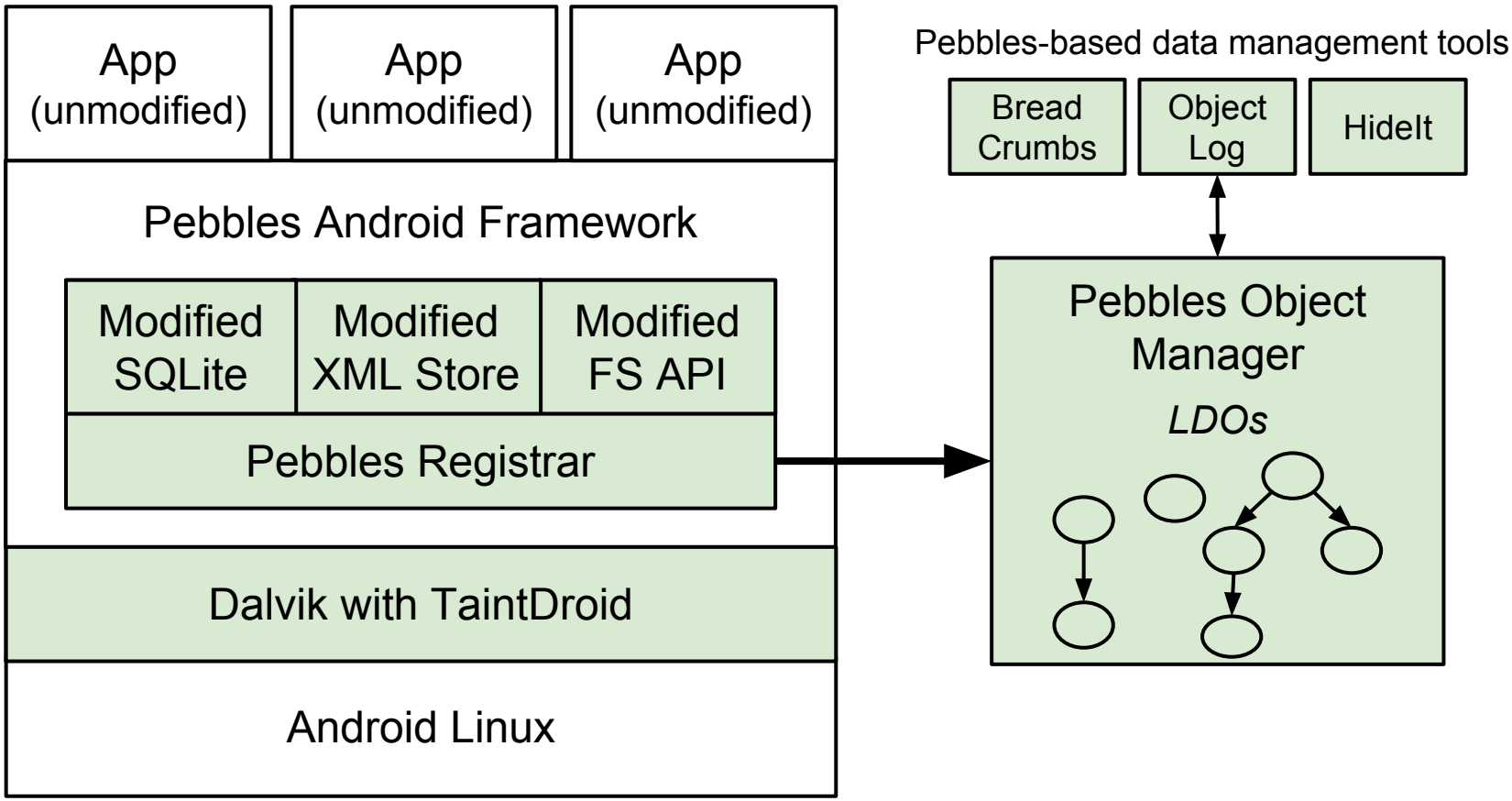


Pebbles's object graph

(only one email's graph shown)

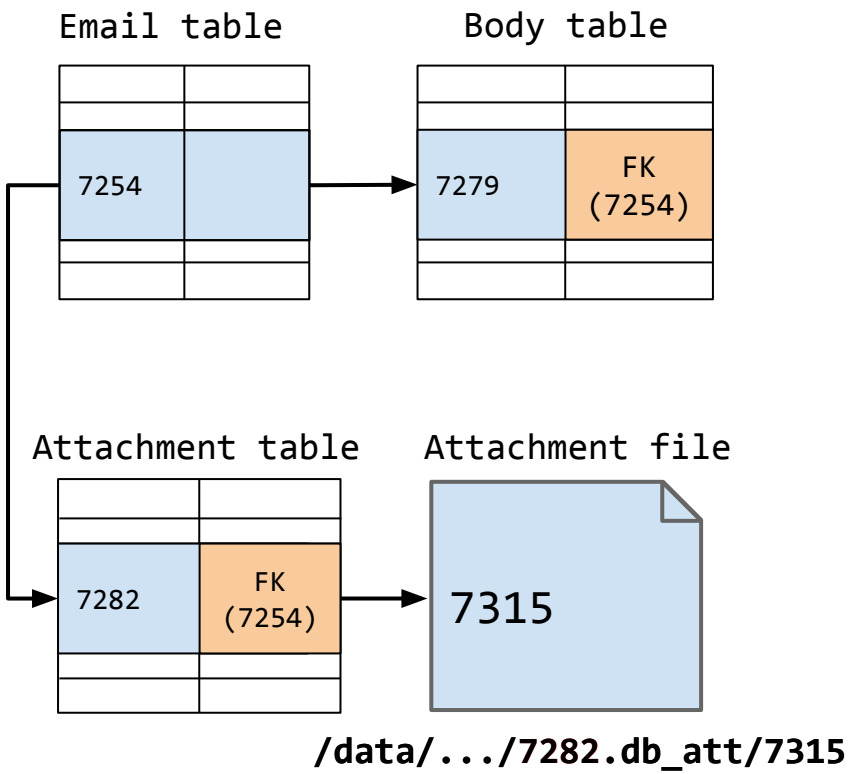


The Pebbles Architecture



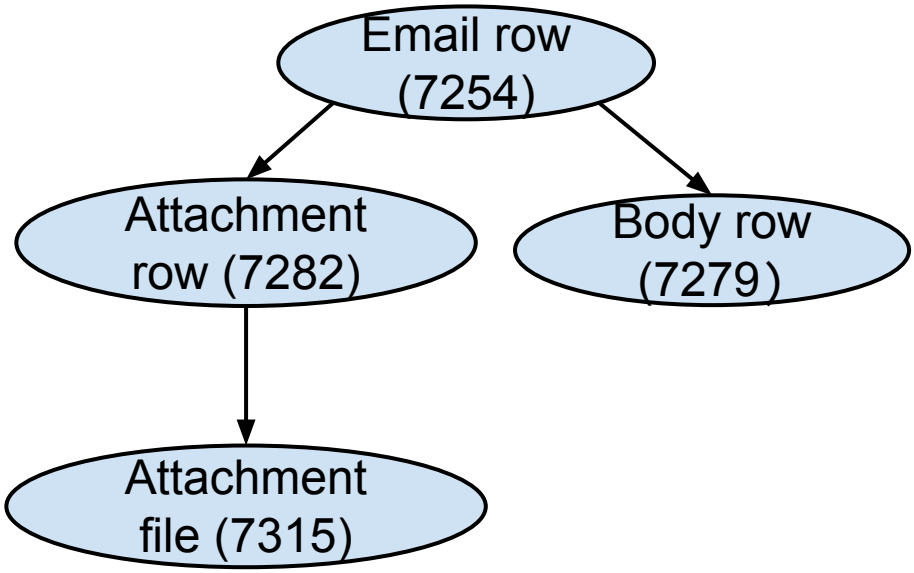
Example: Constructing the Email LDO

Data's layout on disk



Pebbles's object graph

(part of the device-wide object graph)



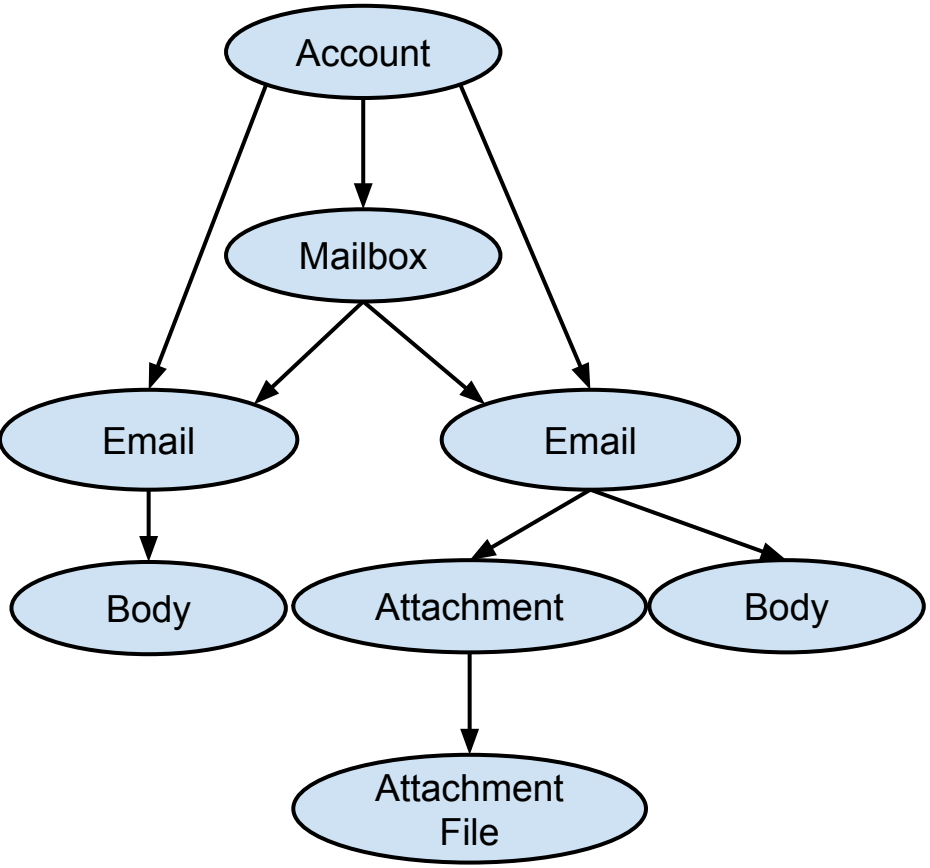
Pebbles Mechanisms

- Pebbles embeds a number of mechanisms for object relationship detection
  - Explicit structure: defined in schemas, XML/FS structure
  - Data flow: data tainted with one node's GUID is written into another
  - Access-based relations: data tainted with a GUID is used to access another object

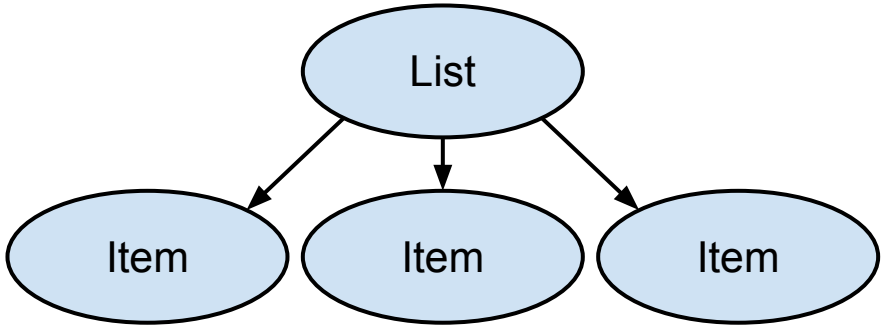
- These mechanisms are rooted in common programming practices, which we validate experimentally at scale

Other LDO Examples

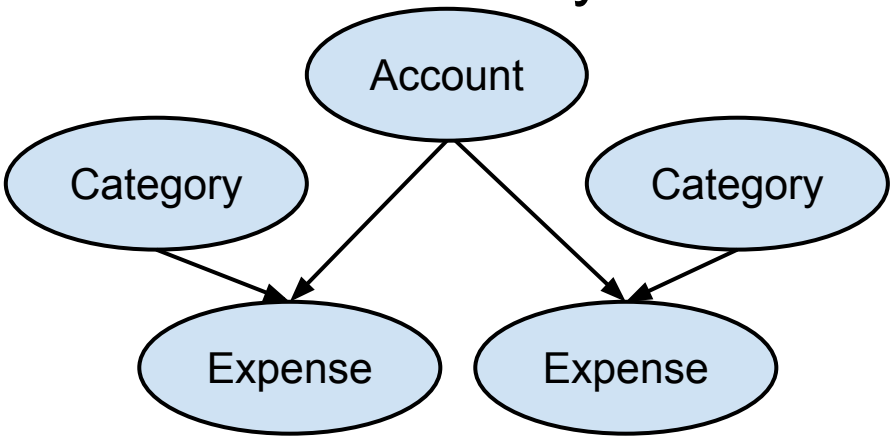
Default Email



Wunderlist



CWMoney



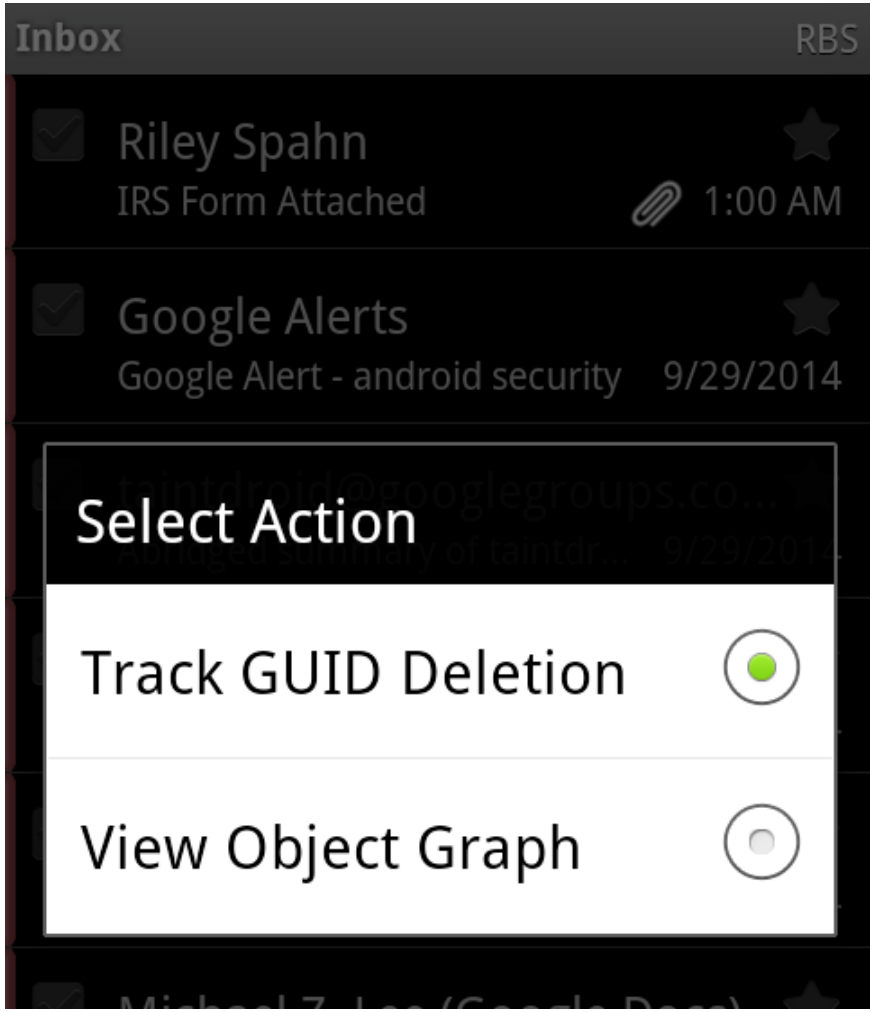
TaintDroid modifications

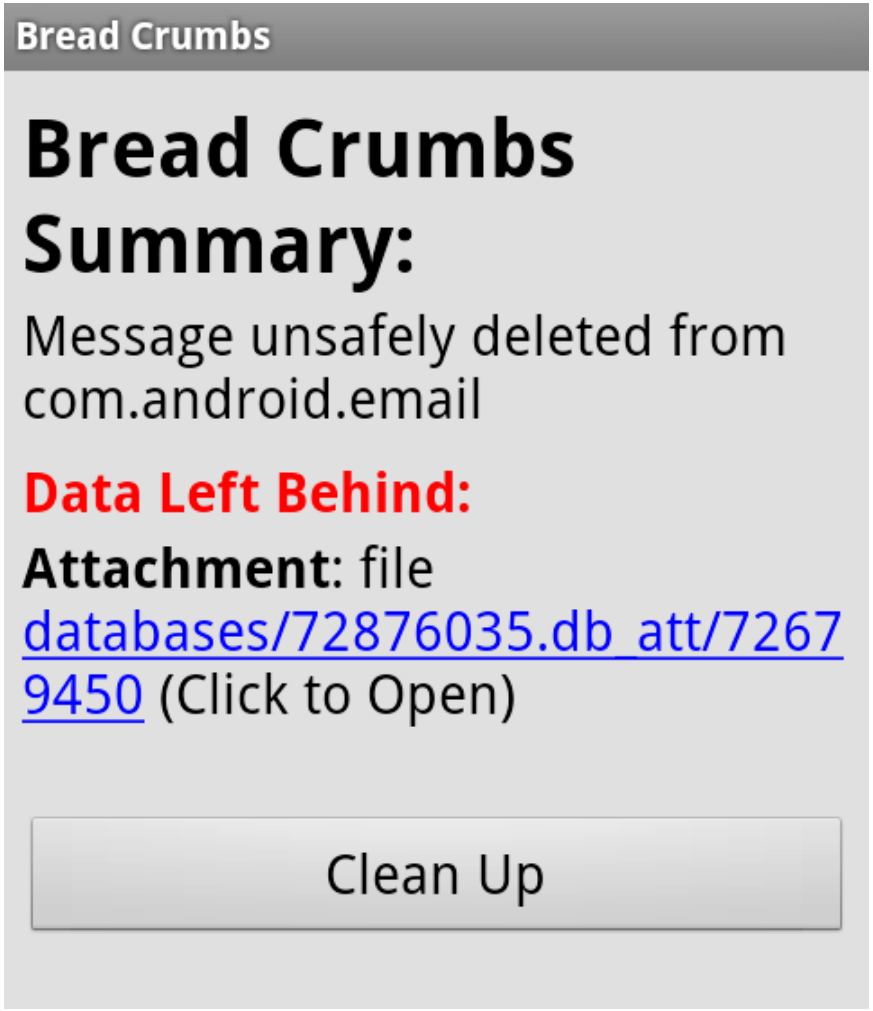
- Using TaintDroid out of the box led to explosions due to its coarse-grained taints
  - Each array gets one taint
- We changed it to do fine-grained tainting
  - Precise and in our experience doesn't explode
- We also added support for multitainting
- We release the patch on GitHub

Outline

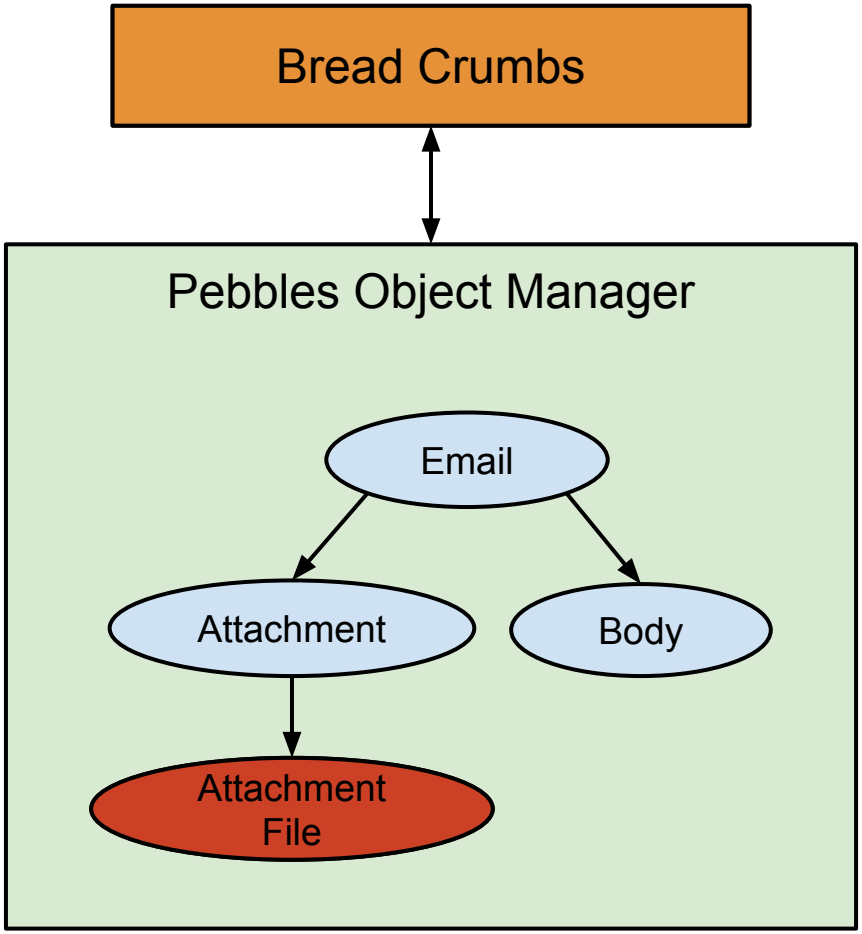
- Motivation
- Android Study
- Pebbles Architecture
- Pebbles-based Tools
- Evaluation
- Conclusions

BreadCrumbs



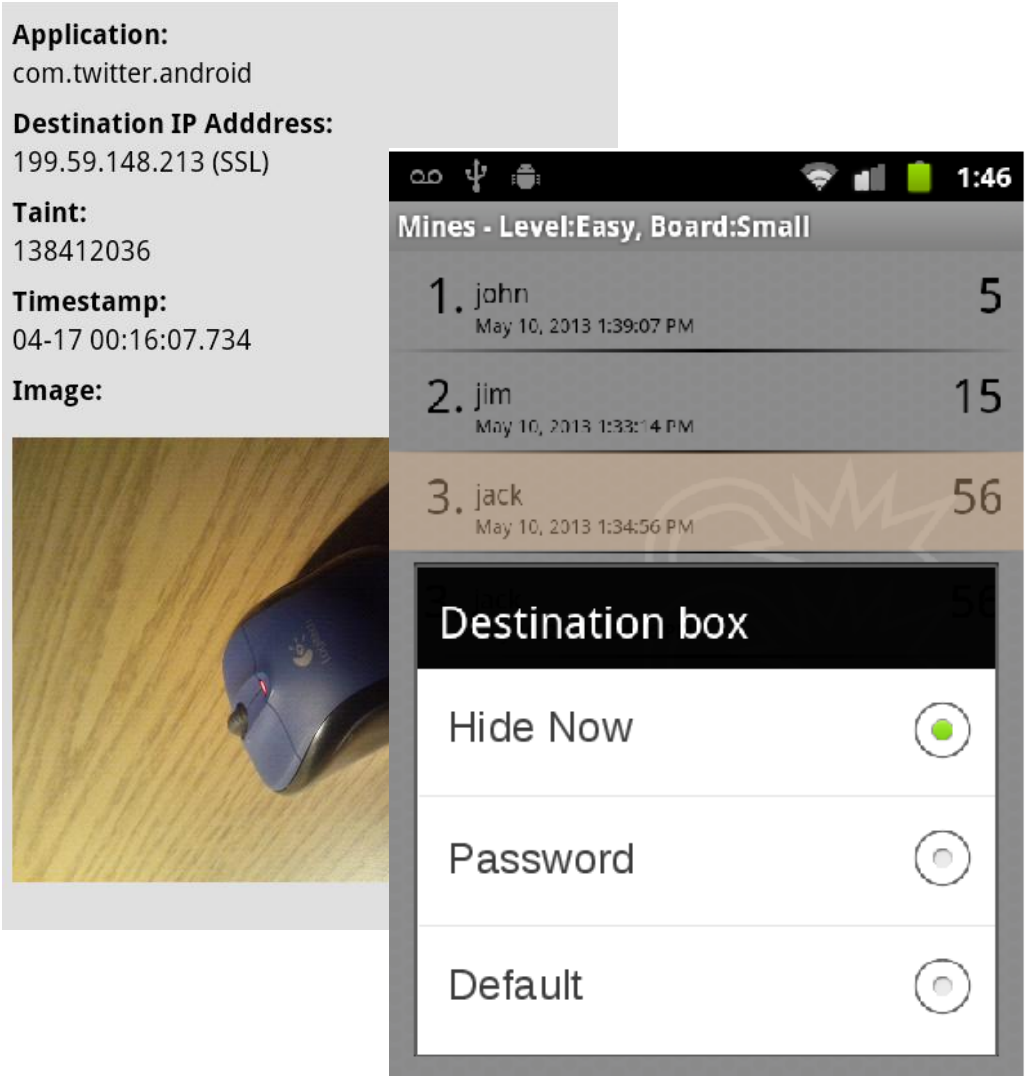


BreadCrumbs



Other Pebbles-Based Tools

- ObjectLog: provides high level auditing
- PebblesDIFC: access control at object level
- HideIt: hiding individual objects



Outline

- Motivation
- Android Study
- Pebbles Architecture
- Pebbles-based Tools
- Evaluation
- Conclusions

Pebbles Evaluation

- Is Pebbles accurate and precise?
- Does Pebbles perform well?
- What is Pebbles' memory overhead?
- What do object structures look like?

Accuracy and Precision

- We evaluated object recognition accuracy within 50 popular applications
  - Focused on applications that store data locally
- We manually identified 68 unique types of objects within the applications

Pebbles is Accurate

App	LDO	Detected	Precise
Browser	History	Y	Y
	Bookmark	Y	Y
	History Item	Y	Y

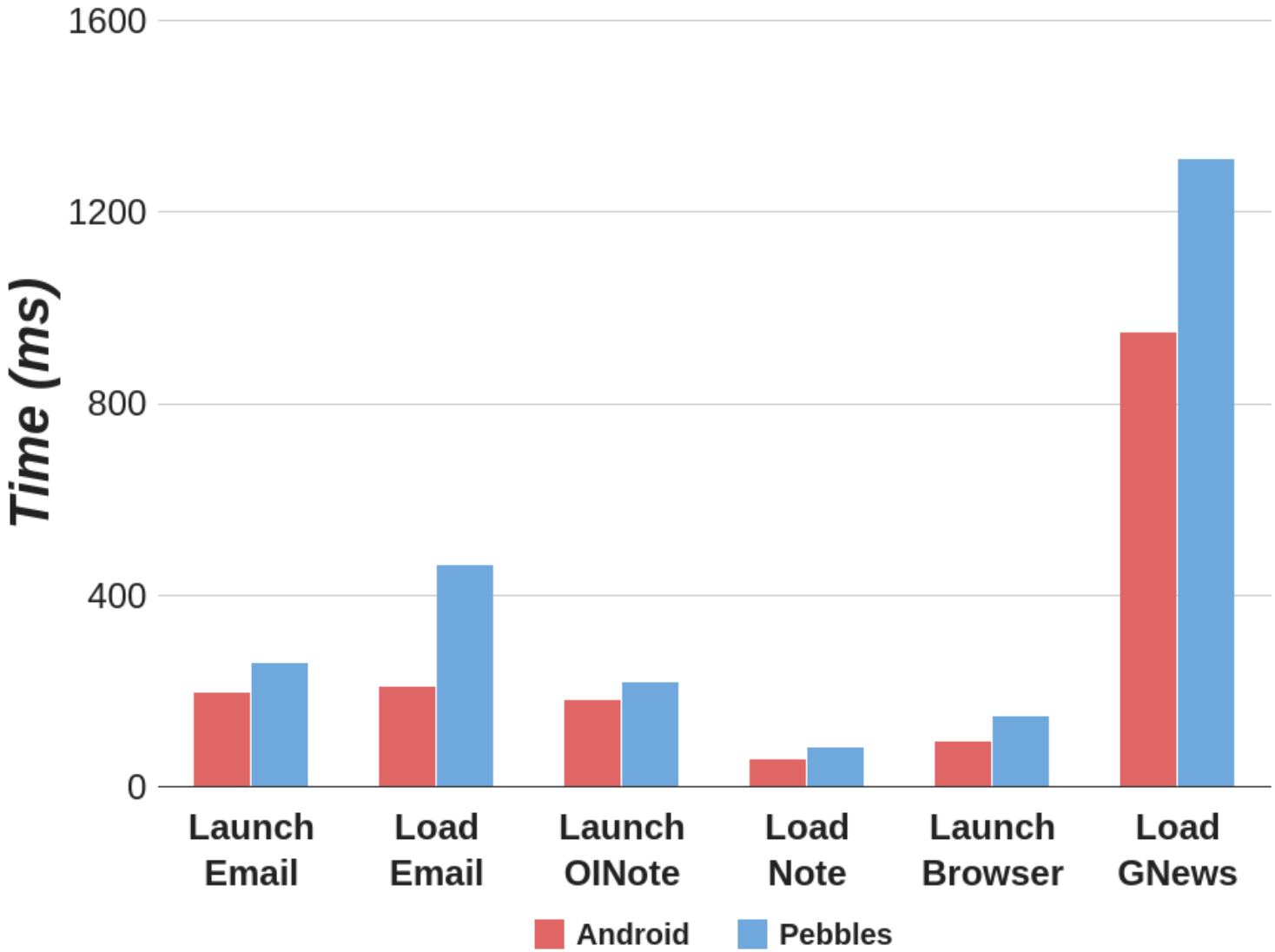
App	LDO	Detected	Precise
LC Bible	Version	Y	**N**
BMI	User	Y	Y
	Result	Y	Y

App	LDO	Detected	Precise
Bloomberg	Stock	**N**	Y
	Price	Y	Y

App	LDO	Detected	Precise
CWMoney	Account	Y	Y
	Category	Y	Y
	Expense	Y	Y

- **60 / 68 uniques objects identified correctly**
- **43/50 applications work for all objects**
- **6 objects show leakage**
- **2 objects show overinclusion**

Performance Is Reasonable



Cool Finding

- BreadCrumbs found improper deletion in 18 of 50 applications
  - Email, Twitter, Evernote, Wunderlist, OnTrack, Expense Manager
- Most don’t delete the entire object
  - Email, LifeChurch Bible, Calendar
- Others just hide deleted objects from the UI
  - Wunderlist, Evernote, PodcastAddict

Outline

- Motivation
- Android Study
- Pebbles Architecture
- Pebbles-based Tools
- Evaluation
- Conclusions

Starting Backup Slides

Conclusions

- Without system support for data management we must rely on apps to do the “right” thing
- Pebbles is a new system-level service that supports data management at a new level of abstraction: logical data objects
  - Relies on structure inherent in high-level storage APIs to accurately recognize objects in unmodified apps
- Supports tools that increase users’ visibility and control over their data

Threat Model

- Protection tools are trusted systems.
- Applications will not obfuscate their data storage structure.
- Applications will not willfully interfere with analysis.
- The scope is restricted to applications that store data locally.
  - Pebbles cannot handle server side data.

Relationship Generation

1. Properties
   a. Apps define explicit relationships through foreign keys in the database, XML hierarchies, or FS.
   b. SQLite is the hub of all persistent data storage and accesses.
2. Object Graph Construction algorithm:
   a. Data Propagation: if data from A is written to B then A←→B.
   b. If possible, refine A←→B to A→B using *Property a*.
   c. Access Propagation: If data from A is used to read B then A←→B.
   d. If possible, refine A←→B to A→B using *Property a*.
   e. Utilize *Property b* eliminating access based on data propagation relationships that do not include any DB nodes.

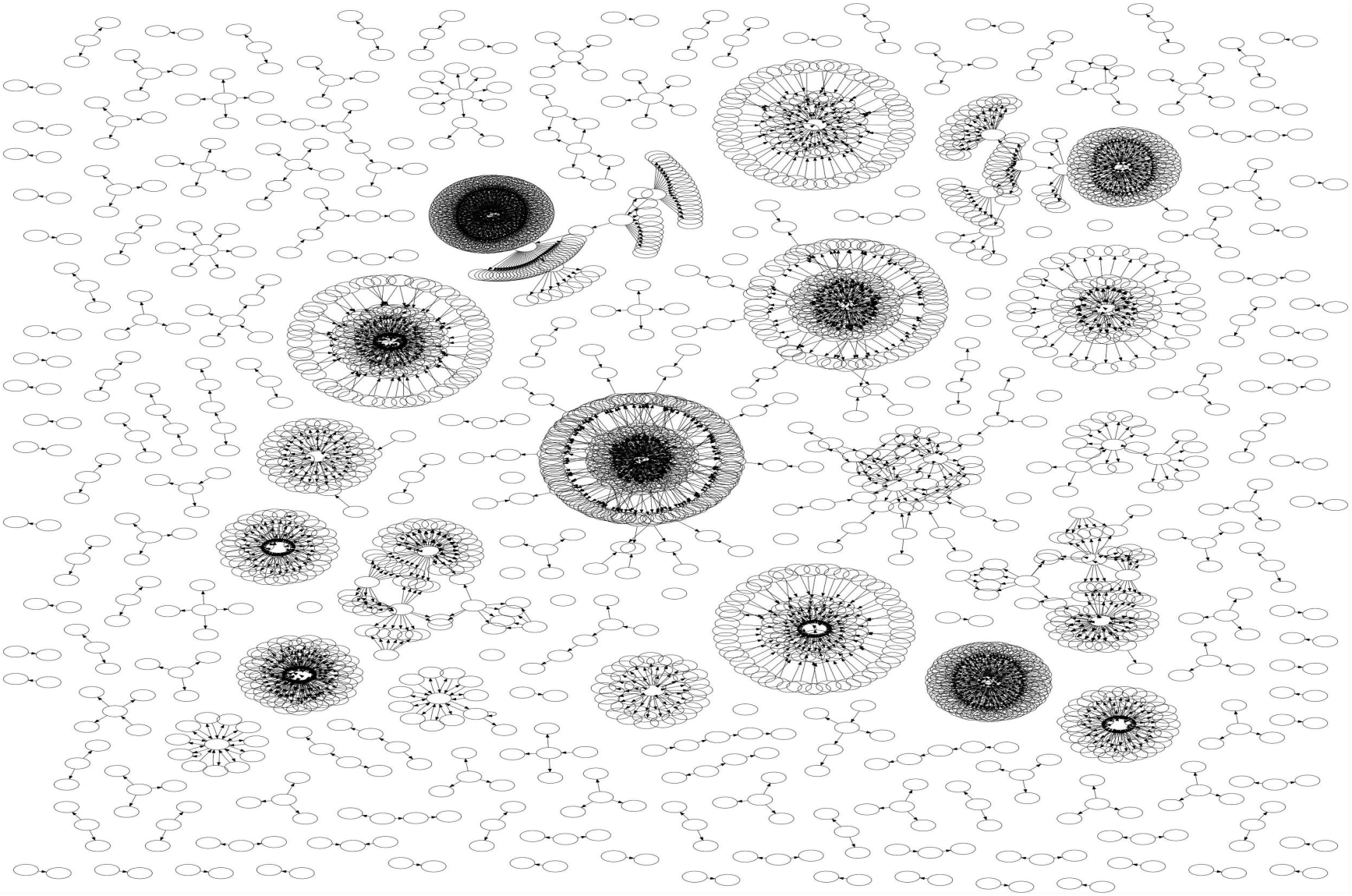
Related Work

- Taint Tracking
  - Pebbles is the first system to use taint tracking to discover higher level data objects without developer intervention
  - Taint tracking has been used for data provenance
  - Pebbles could be used to eliminate static data labels in code
- Fine-grained protection in OSes
  - Multics/Hydra - considered difficult to implement and manage
  - SELinux/SEAndroid - our work is complementary
- Securing and Hiding Data
  - Many applications and file systems secure at the block or file level.
  - VaultHide/KeepSafe must be modified for each application.
- Inferring Structure
  - Laika derives relationships from in memory data
  - Other systems have inferred relationships in relationships and between files but not between high level object

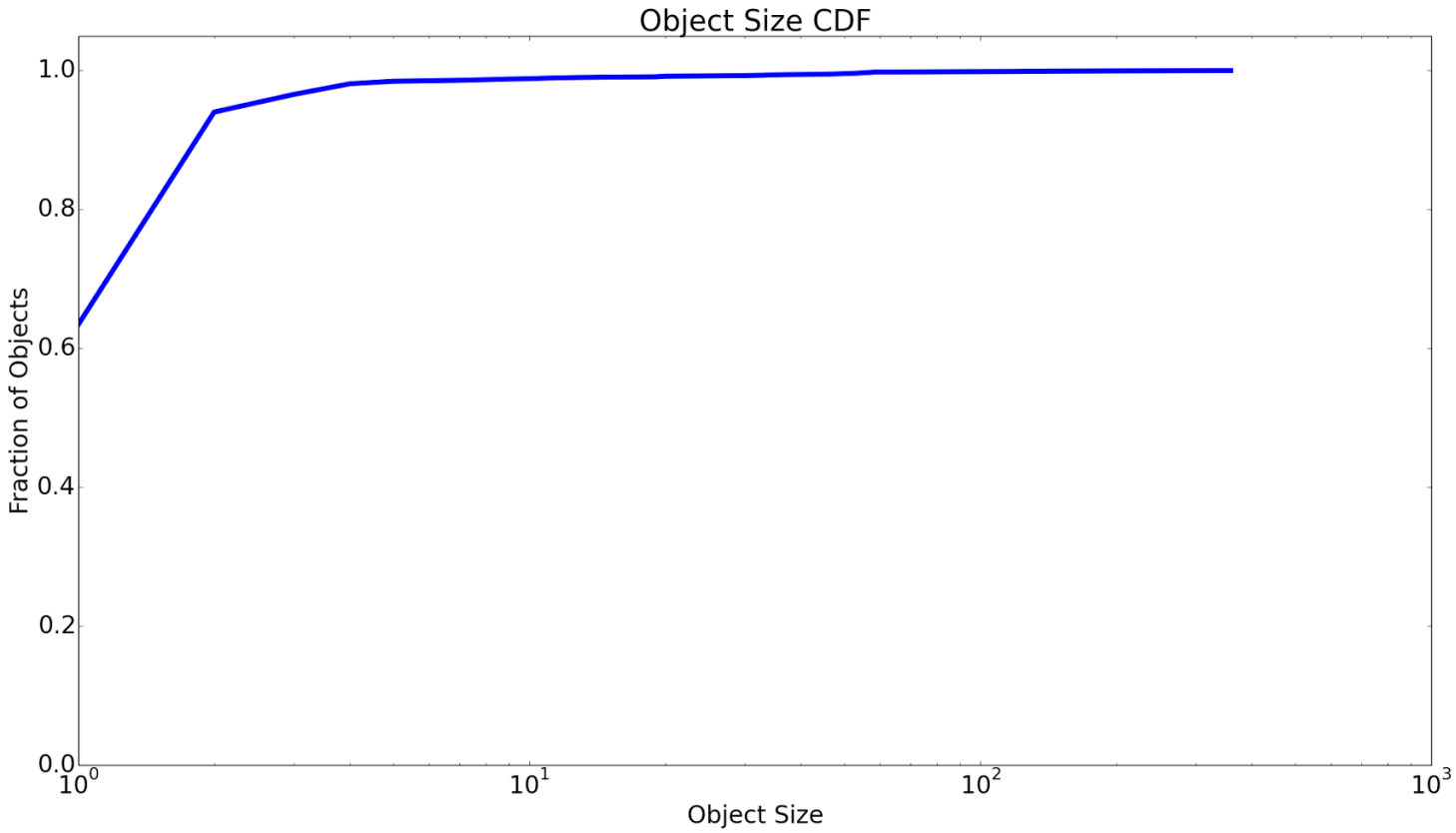
Memory Usage

Application	Android	Pebbles	Overhead
Email	196 MB	230 MB	34 MB
OINote	208 MB	224 MB	16 MB
Browser	238 MB	267 MB	26 MB

Device Wide Object Graph



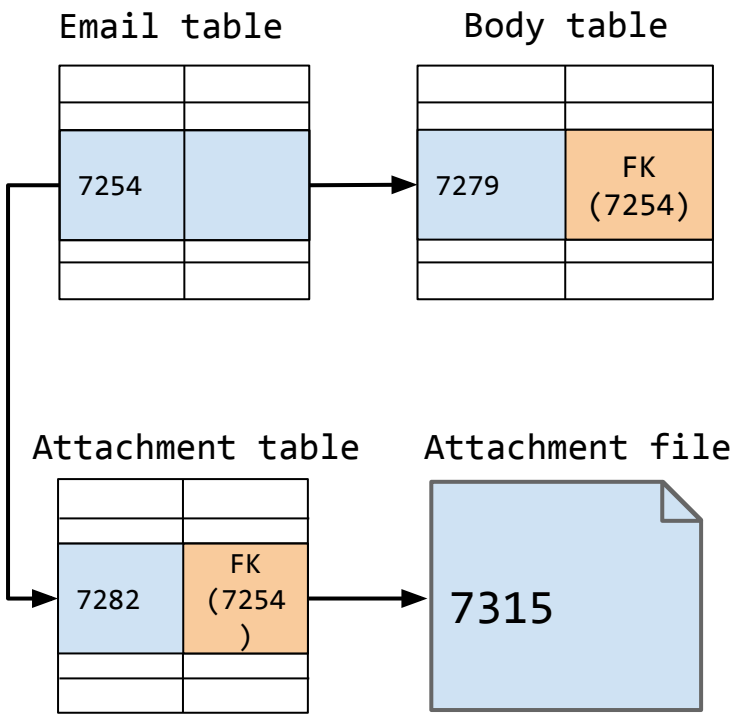
Object Size


Object Size CDF
Fraction of Objects
1.0
0.8
0.6
0.4
0.2
0.0
10^0
10^1
10^2
10^3
Object Size

<Slides past here not in use./>

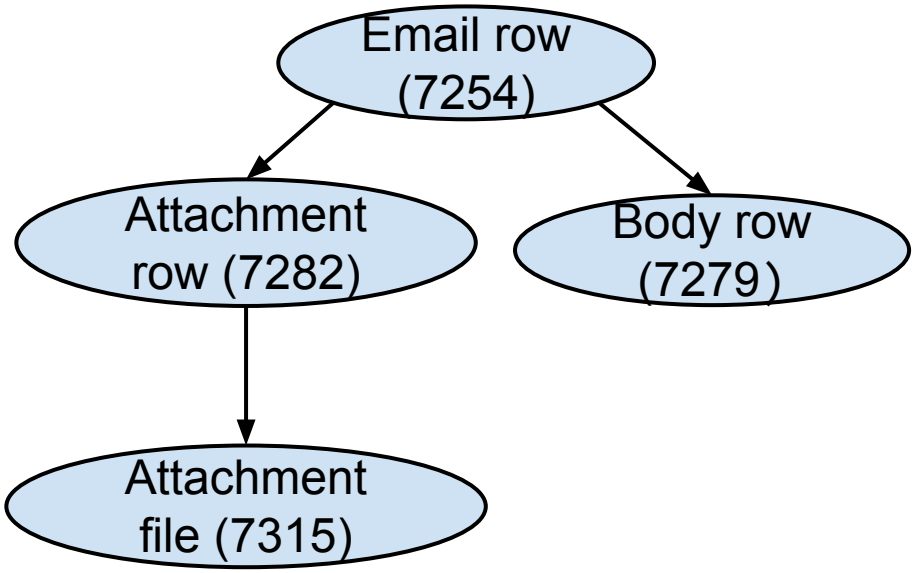
Example: Constructing the Email LDO

Data's layout on disk



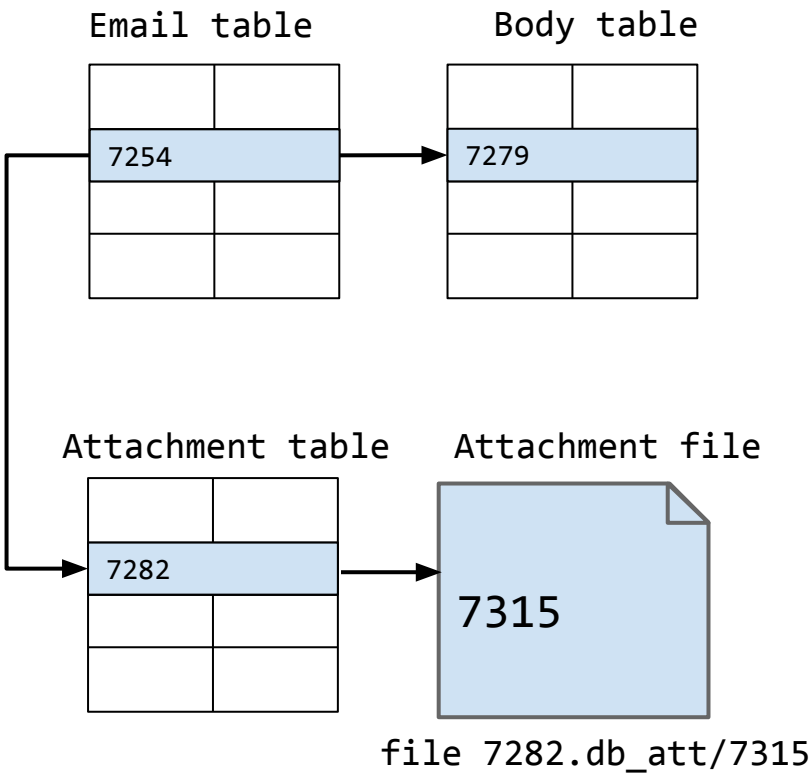
Pebbles's object graph

(part of the device-wide object graph)



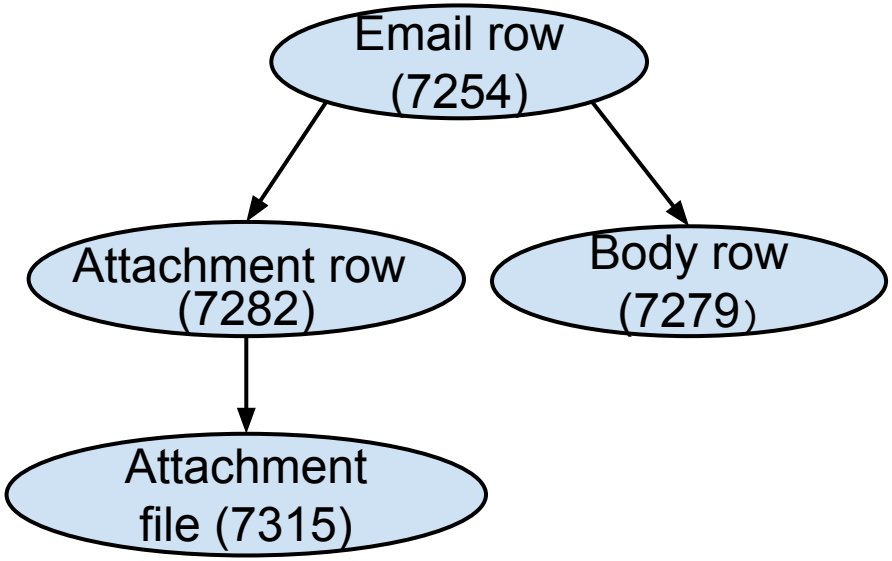
Example: The Email LDO

Data's layout on disk



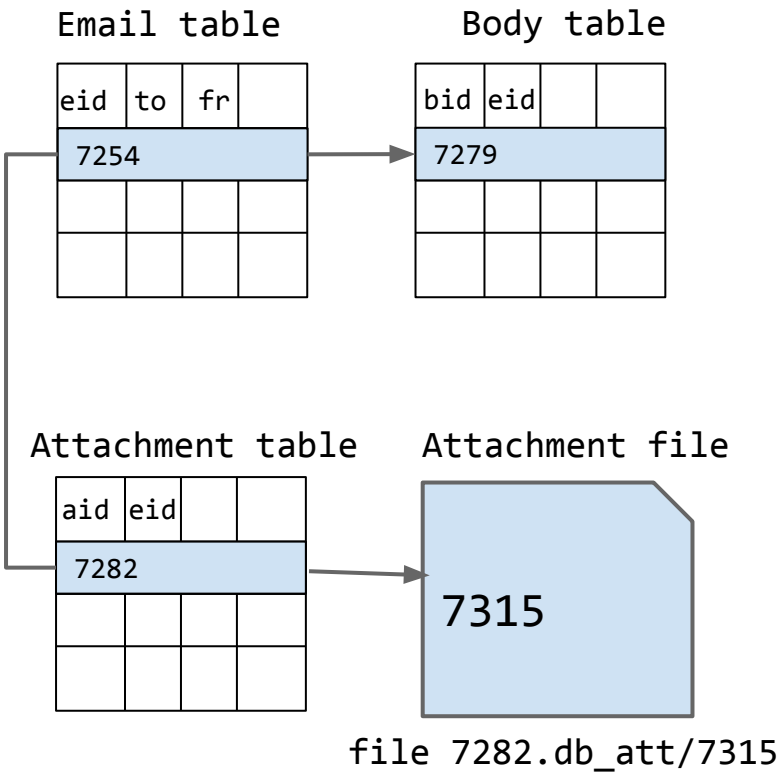
Pebbles's object graph

(only one email's graph shown)

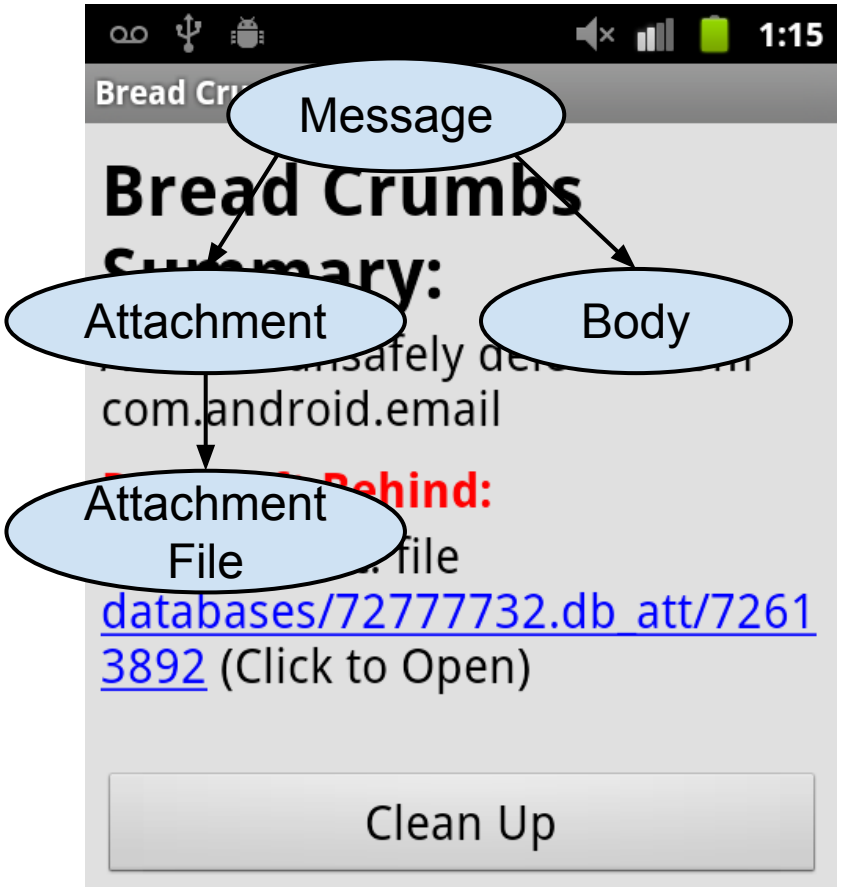


BackupSlides

- Assumptoins
- Related Work
- ThreatModel
- Common Practices

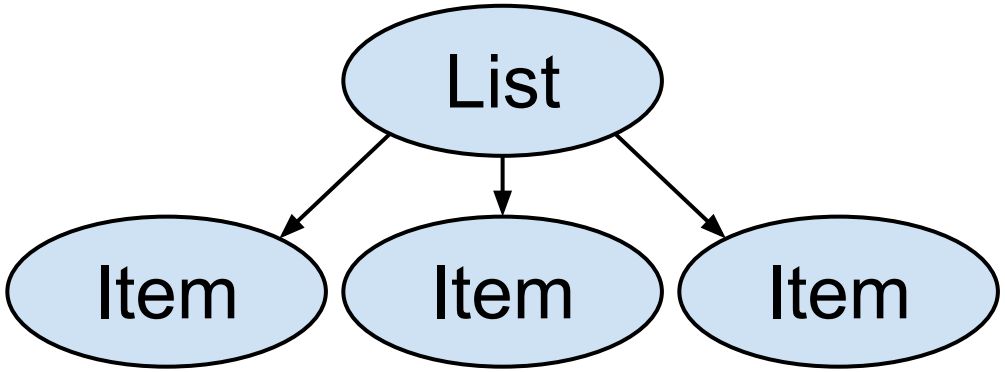

Email table
eid
to
fr
7254
Body table
bid
eid
7279
Attachment table
aid
eid
7282
Attachment file
7315
file 7282.db_att/7315

Ex. 1: Is My Email Deleted?



Files Are More Than Files

```
accessed:
/app/todo/todo.db
written:
/app/todo/todo.db
accessed:
/app/todo/todo.db
```



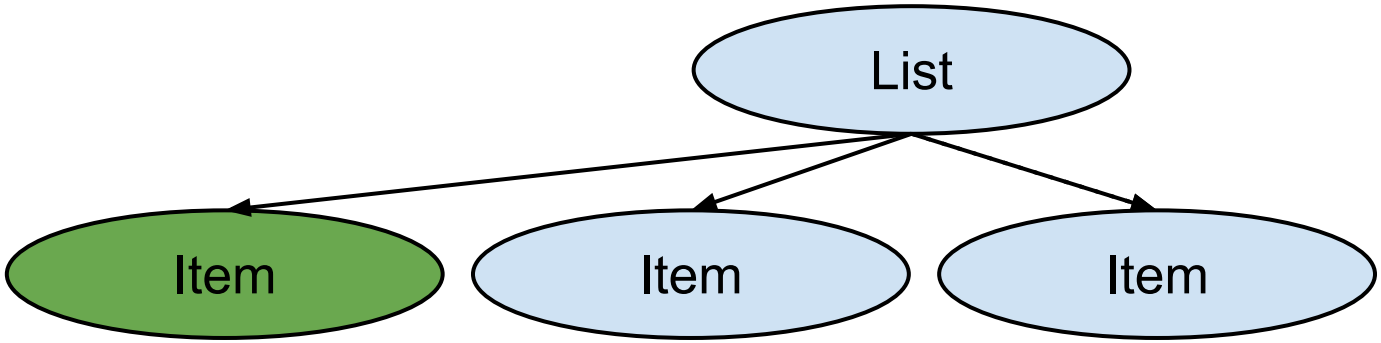
Problem: OSes Don't Manage Data

- OSes have ceded data management to applications
- We don't want to rely on applications
  - xxx

- What we need: OSes should incorporate abstractions to increase users' control over data management

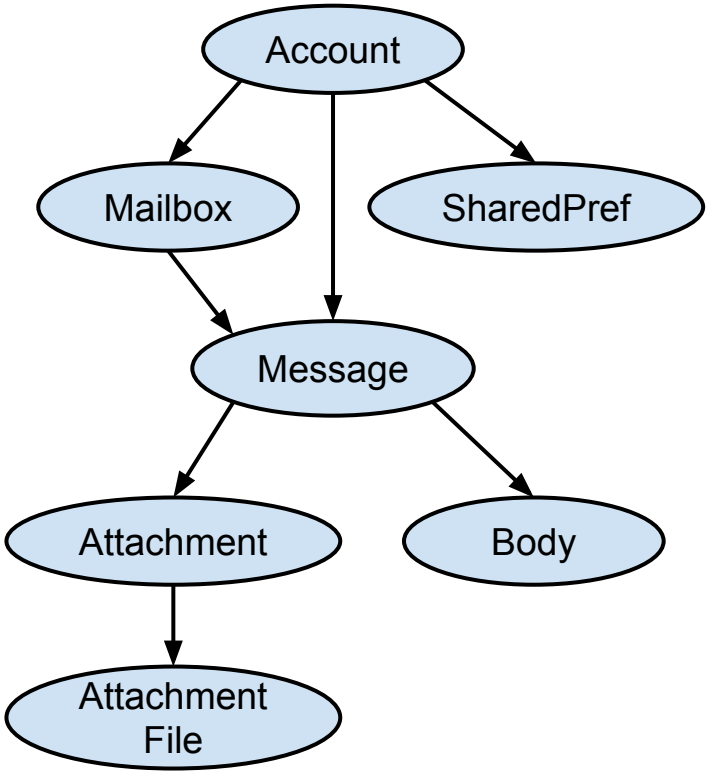
Pebbles's Goals

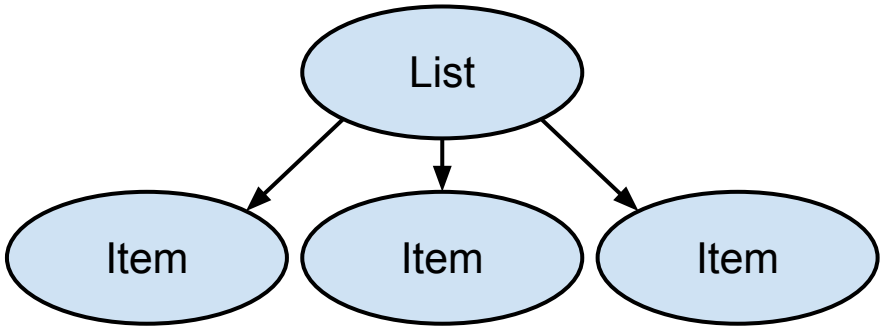
- Accurate and Precise Object Recognition
  - No Leakage
  - No Overinclusion



Pebbles

Pebbles provides data management at the level of a logical data object.

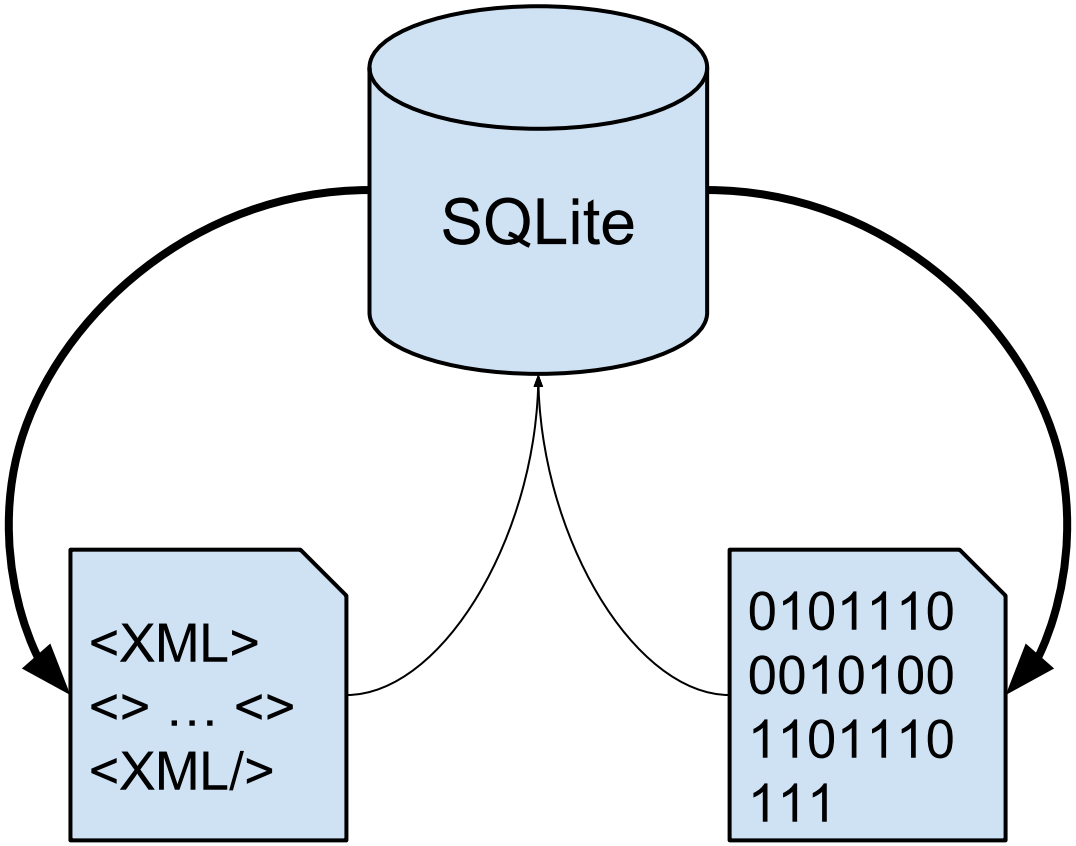




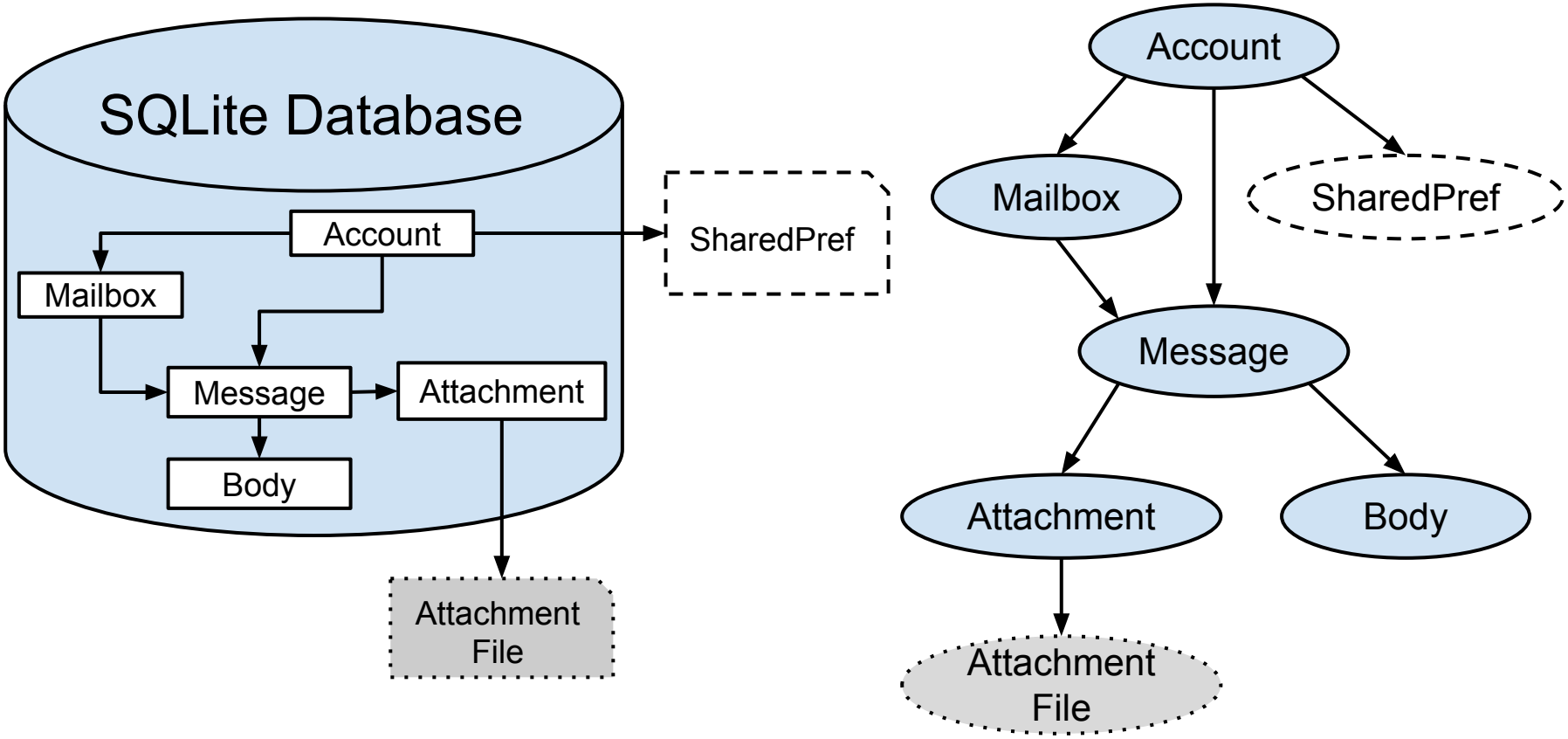
Pebbles’s Goals

- Meaningful Granularity
  - Discovered objects must but recognizable and useful to the end user.
- No New Application APIs
  - Developers only need to follow common best practices.
  - Explored APIs in CleanOS.

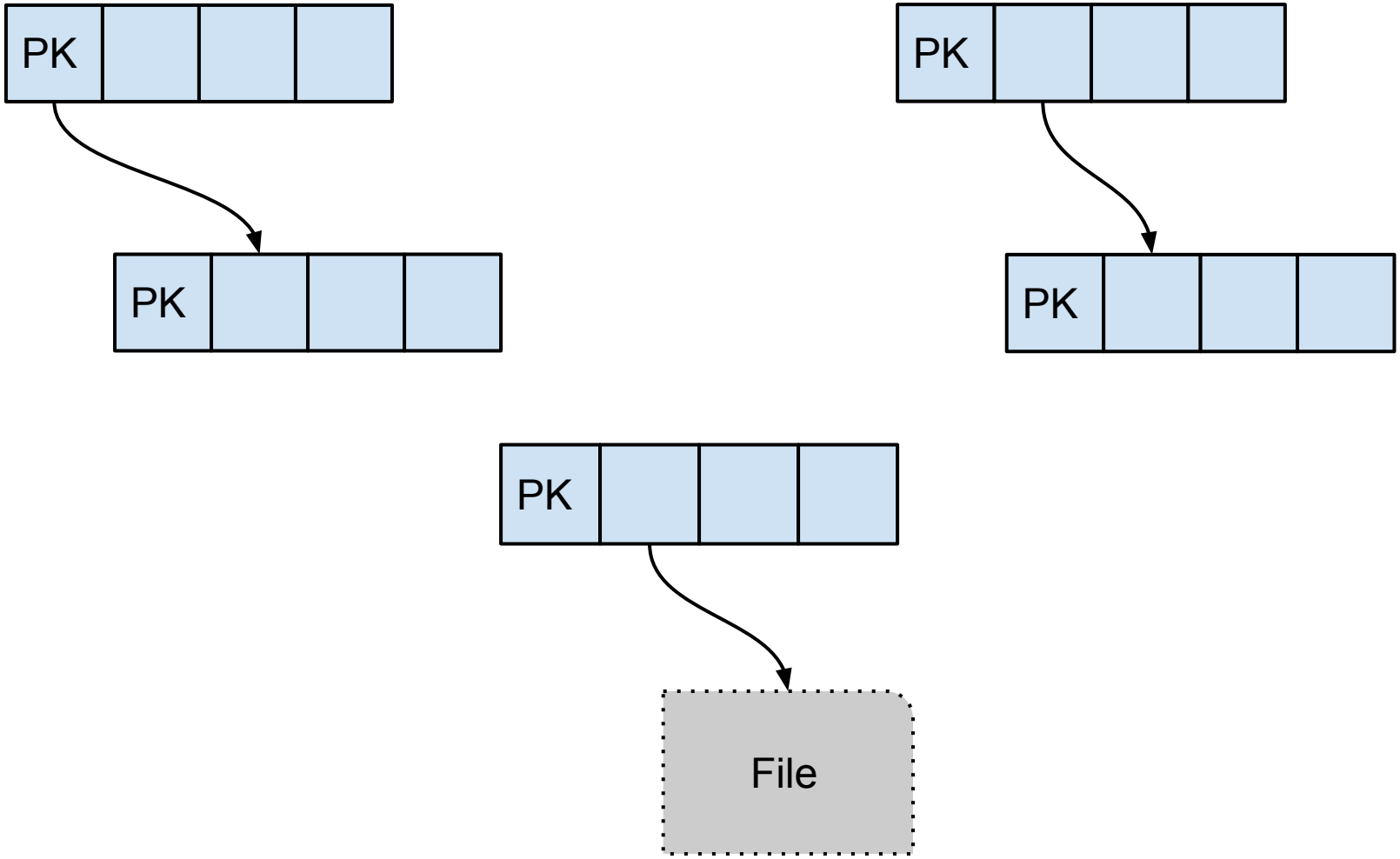
Android Storage Environment



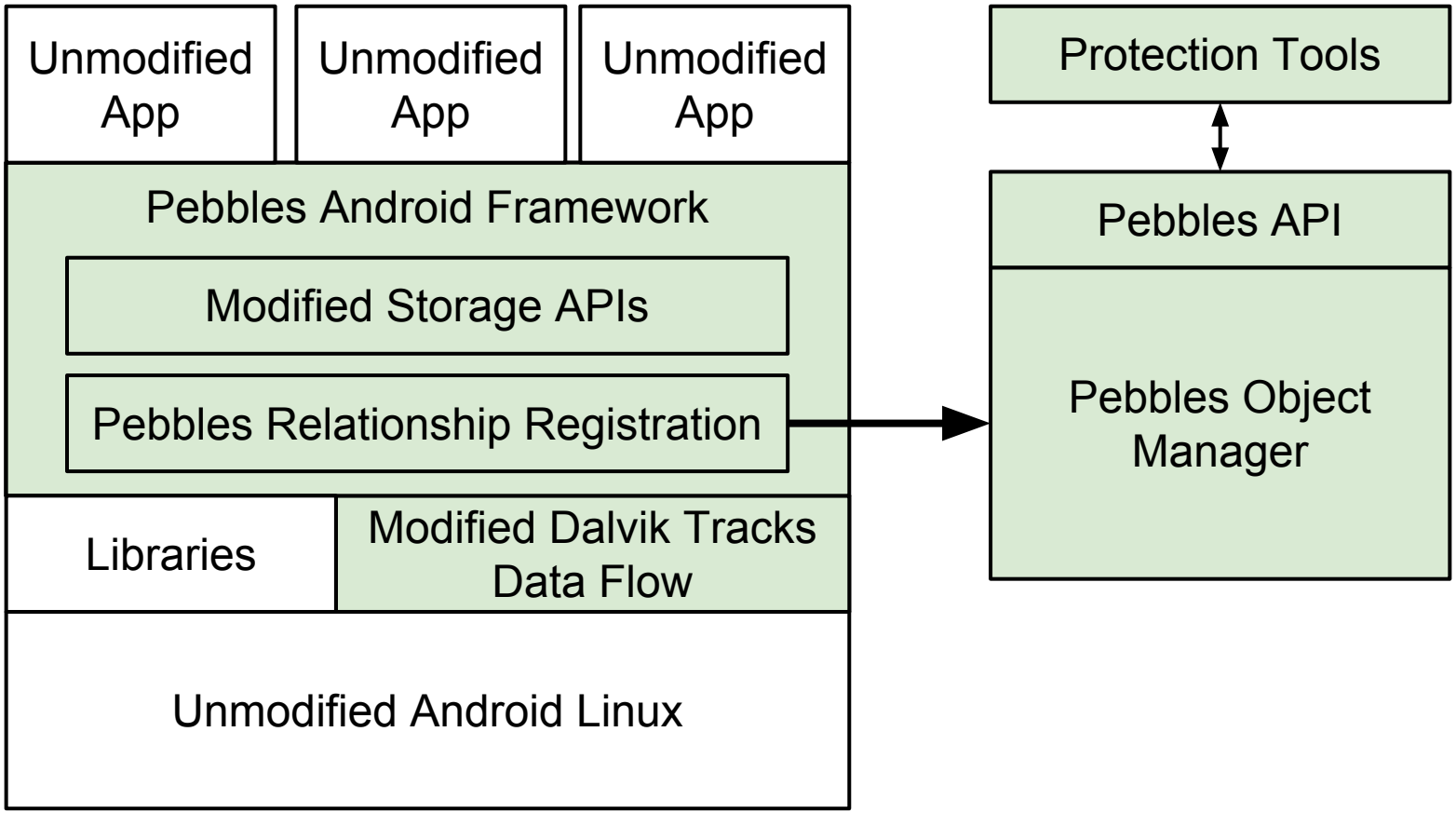
Pebbles Logical Data Objects



Relationship Generation

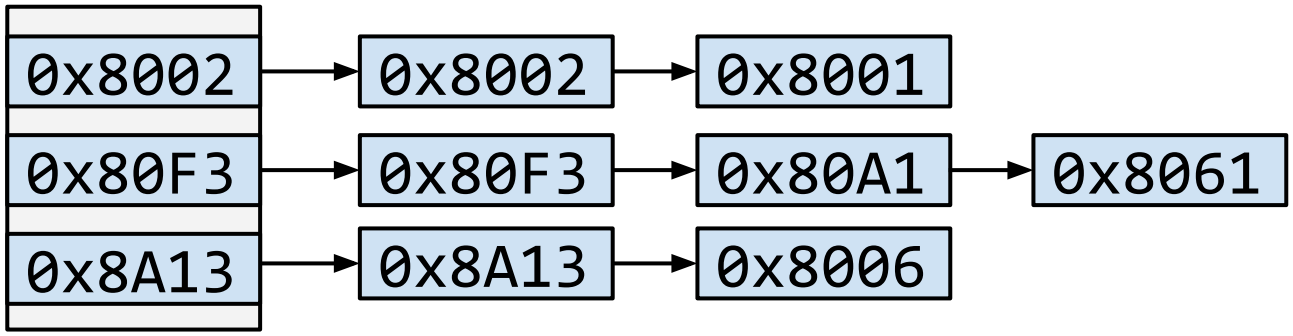


Pebbles Architecture

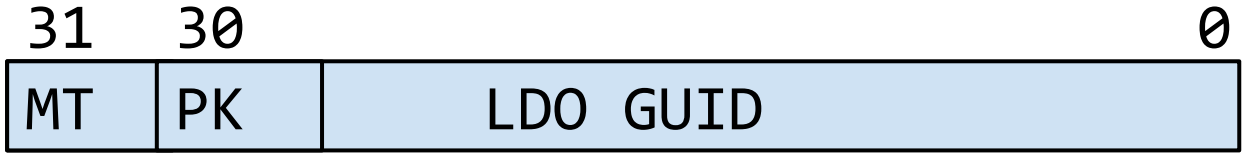


Object Tracking

- Related taints are stored in a lookup table.



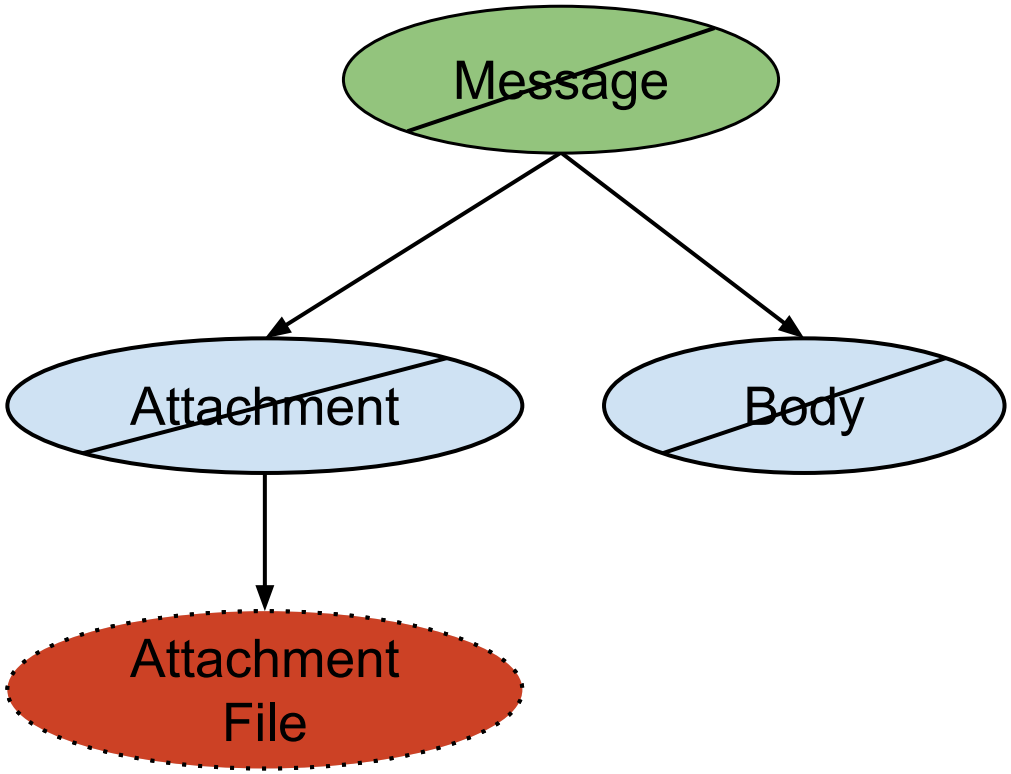
- Pebbles reserves the two upper bits for mark multi-tainted and primary key objects.



Pebbles Application

- *BreadCrumbs* - Provides transparency into application's data deletion practices.
- PebblesNotify - Provides high level information about data exfiltration.
- PebblesDIFC - Object level interapplication access control.
- HideIt - Object level data protection.

BreadCrumbs

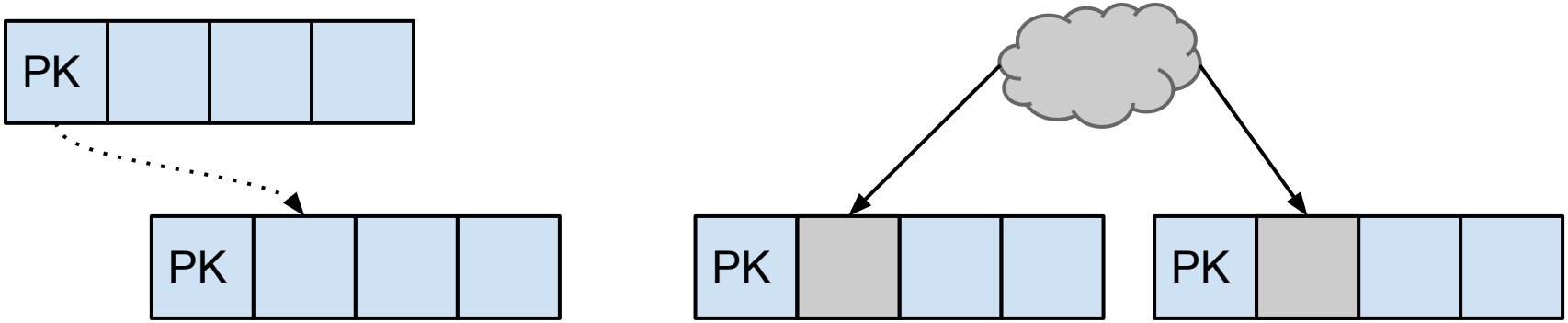


Pebbles Evaluation

- Evaluated pebbles for leakage and overinclusion.
- Correctly identified 60 of 68 object types.
- Correctly identified all objects in 43 of 50 applications.
- 6 objects exhibited leakage
- 2 objects showed overinclusion.

Failures

- Leakage from parallel writes.



- Data copies cause overinclusion

BreadCrumbs Evaluation

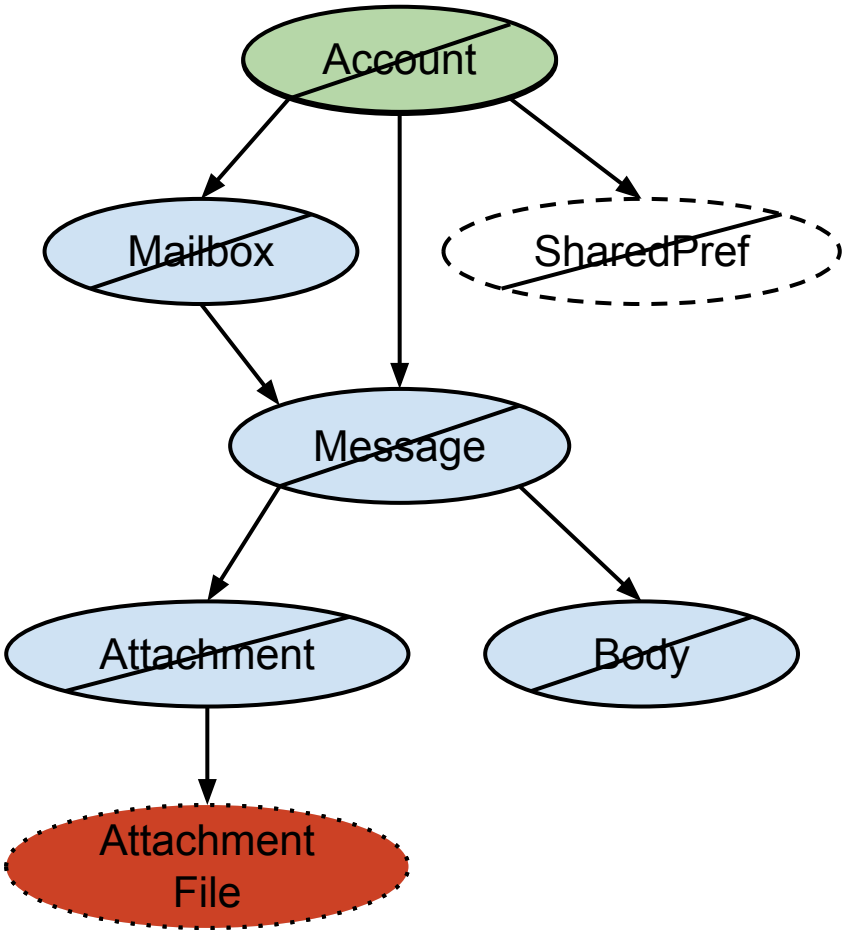
- 18 of 50 applications have unsound deletion practices.
- Several applications wrote data but did not clean it up.
- Others did not delete data at the user's request.
  - These were cloud applications where data is likely not deleted server side.

BreadCrumbs and Snapchat

In The Future

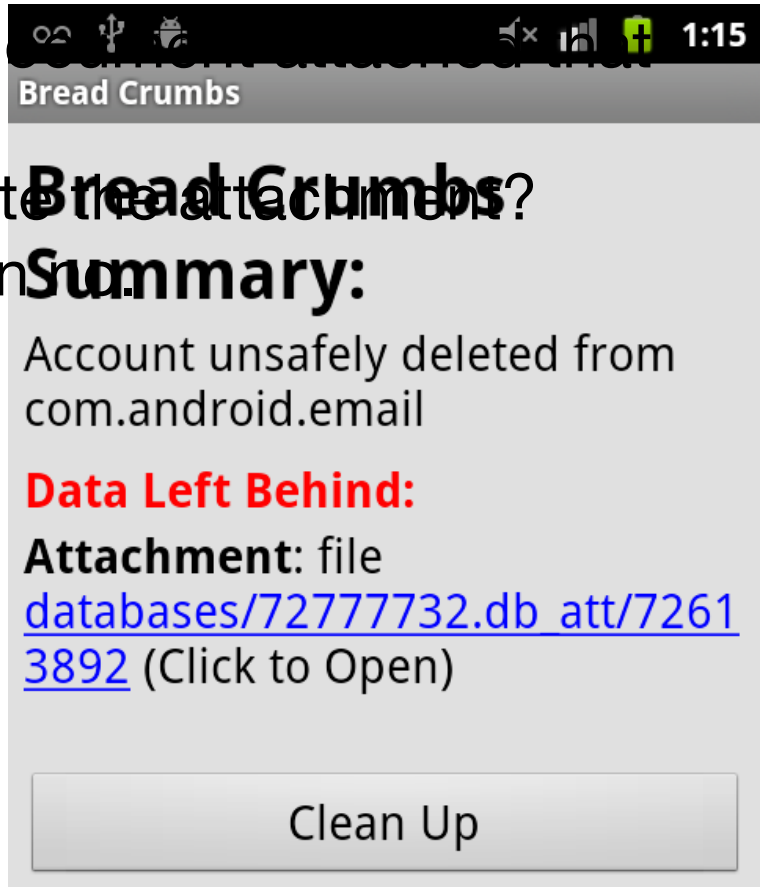
- BreadCrumbs could be used for developer testing.
- The LDO could be applied to web frameworks or other abstracted persistence APIs.

BreadCrumbs



Does Delete Actually Delete?

- I receive an email with an IRS document attached that contains my SSN
- If I delete the email does it delete the attachment?
- If you're using Gingerbread then no.



What is this Object?

```
accessed:
/app/todo/todo.db
written:
/app/todo/todo.db
accessed:
/sdcard/021.jpg
```

We Have Little Control

The Problem: OSs Don’t Manage Data

- Users have no visibility into applications.
  - Don’t know how data is stored.
  - Don’t know if data gets deleted.
- Users have no control over their data.
  - No control over how data is shared.
  - No control over if data is even deleted.

Pebbles

- We developed Pebbles to provide insight into and control over how applications manage our data.
- We do this **without** modifying the application.

Pebbles’s Goals

- Discover application data structures.
- Provide transparency and control.
- Work with most applications.
- Work with unmodified applications.

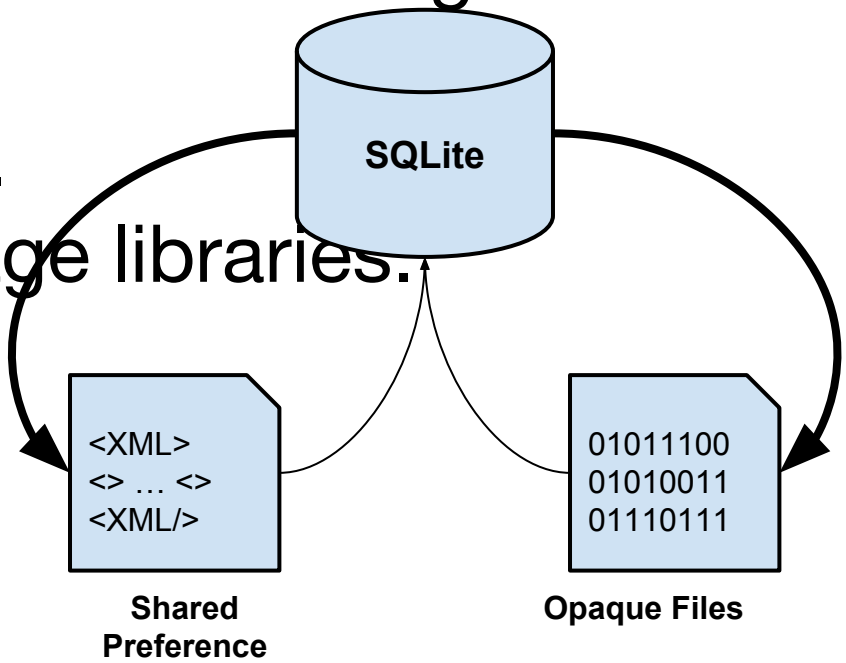
<Logos for Applications/>

Pebbles’s Challenges

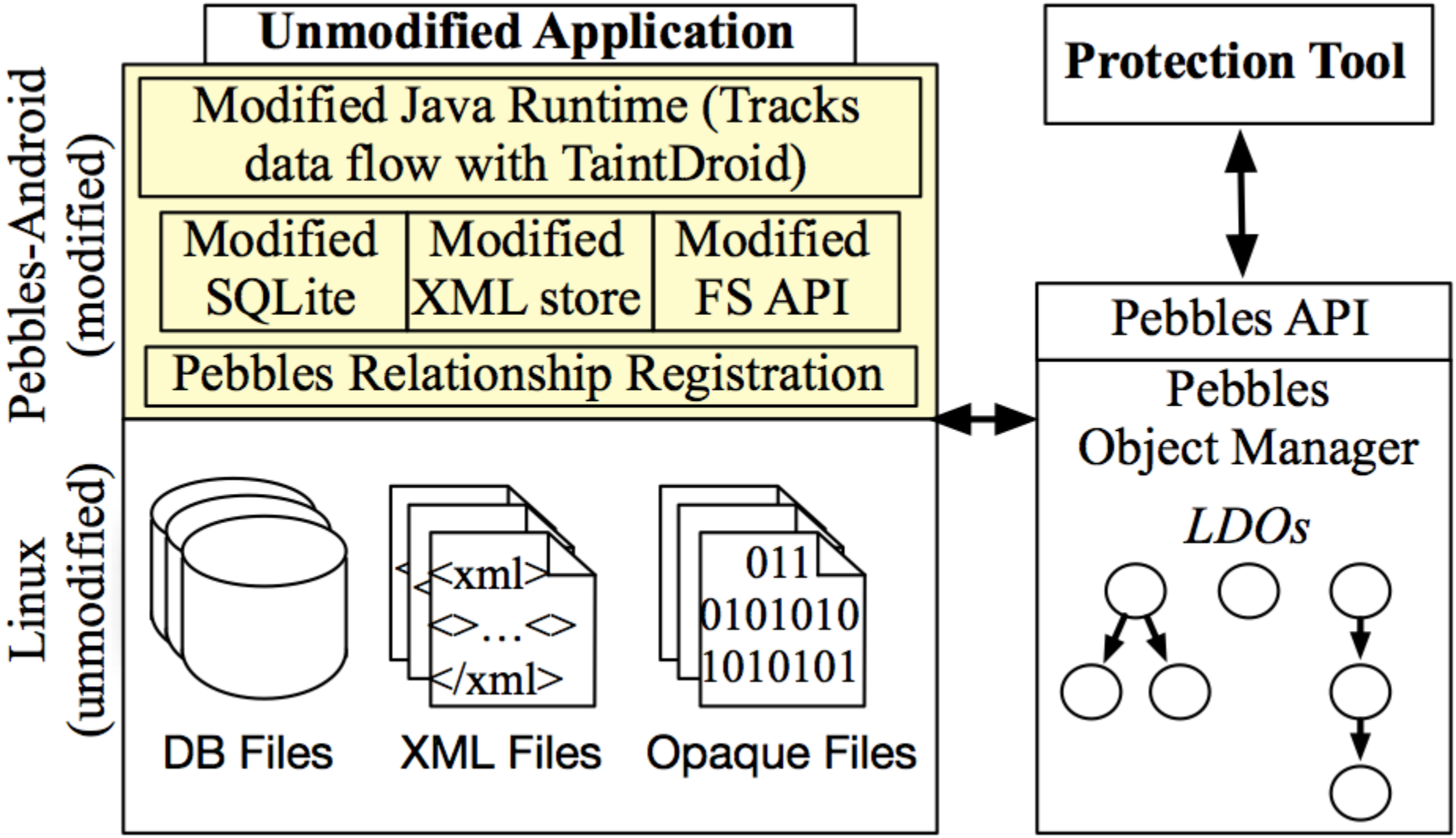
- At what level do we define application objects?
- Can we support all storage practices?
- Do we need a large homogeneous ecosystem?

Measurement Study

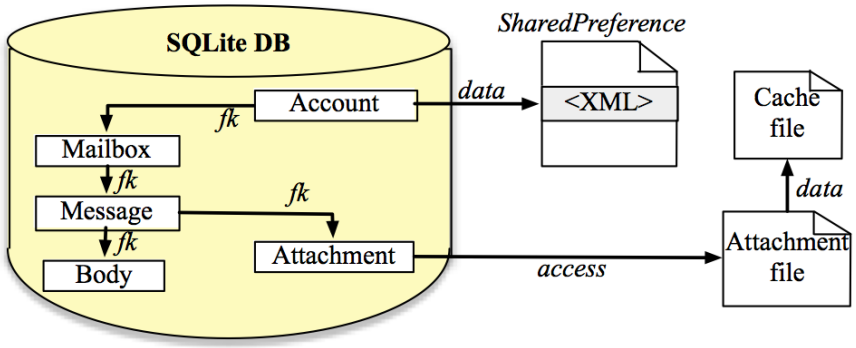
- 93/98 applications stored data using the Android APIs.
- 83 of those use SQLite.
- ~2% use external storage libraries.

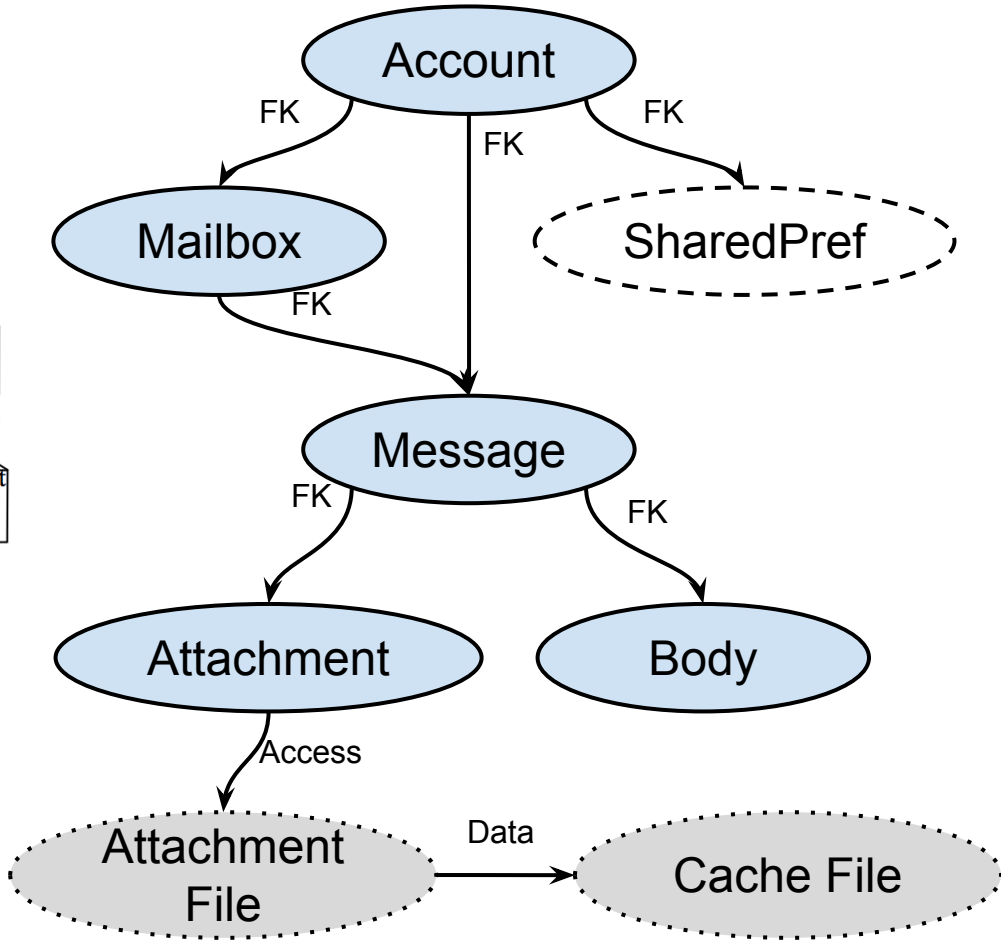


Pebble's Architecture



Pebbles Logical Data Objects



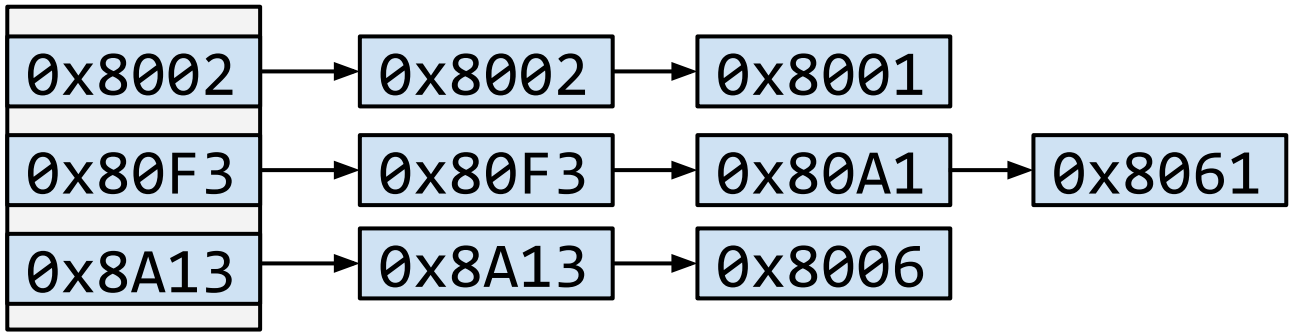


Relationship Generation

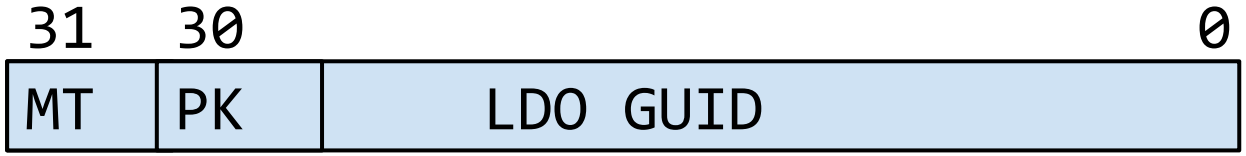
1. Properties
   a. Apps define explicit relationships through foreign keys in the database, XML hierarchies, or FS.
   b. SQLite is the hub of all persistent data storage and accesses.
2. Object Graph Construction algorithm:
   a. Data Propagation: if data from A is written to B then A←→B.
   b. If possible, refine A←→B to A→B using *Property a*.
   c. Access Propagation: If data from A is used to read B then A←→B.
   d. If possible, refine A←→B to A→B using *Property a*.
   e. Utilize *Property b* eliminating access based on data propagation relationships that do not include any DB nodes.

Object Tracking

- Related taints are stored in a lookup table.



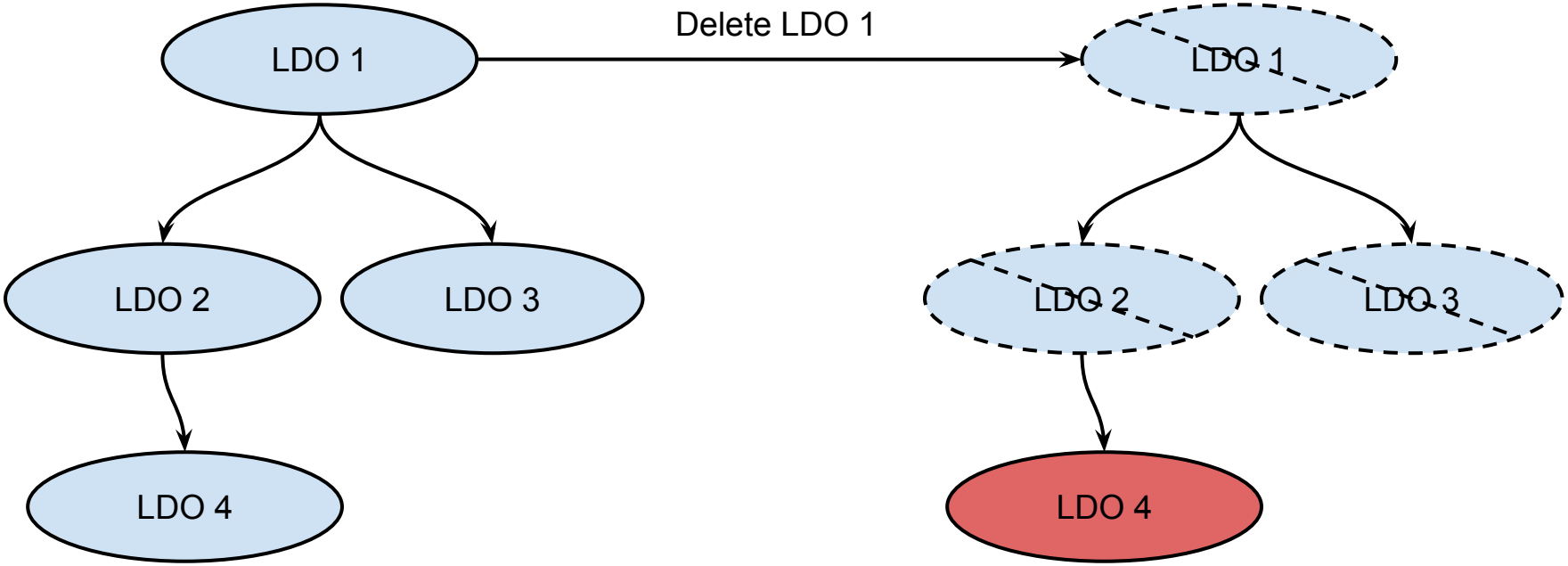
- Pebbles reserves the two upper bits for mark multi-tainted and primary key objects.



Pebbles Application

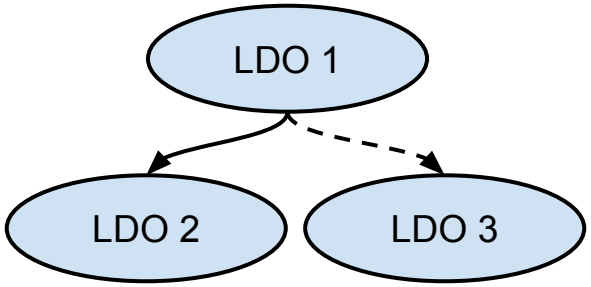
- BreadCrumbs - Provides transparency into application’s data deletion practices.
- PebblesNotify - Provides high level information about data exfiltration.
- PebblesDIFC - Object level interapplication access control.
- HideIt - Object level data protection.

BreadCrumbs

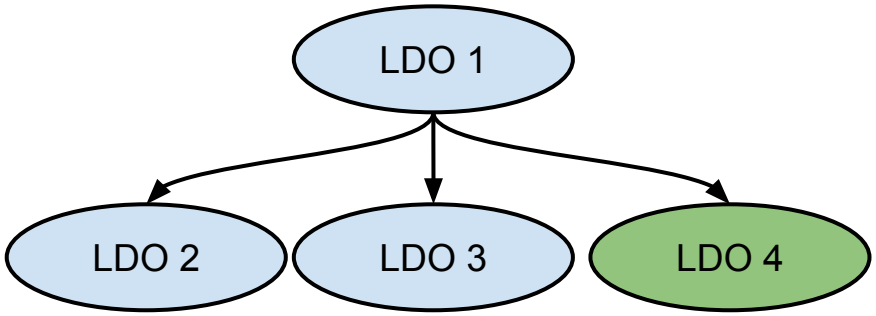


Pebbles Evaluation

- Evaluated Pebbles using 50 popular applications with a total of 68 object types.
- Addressed overinclusion and leakage.



Leakage



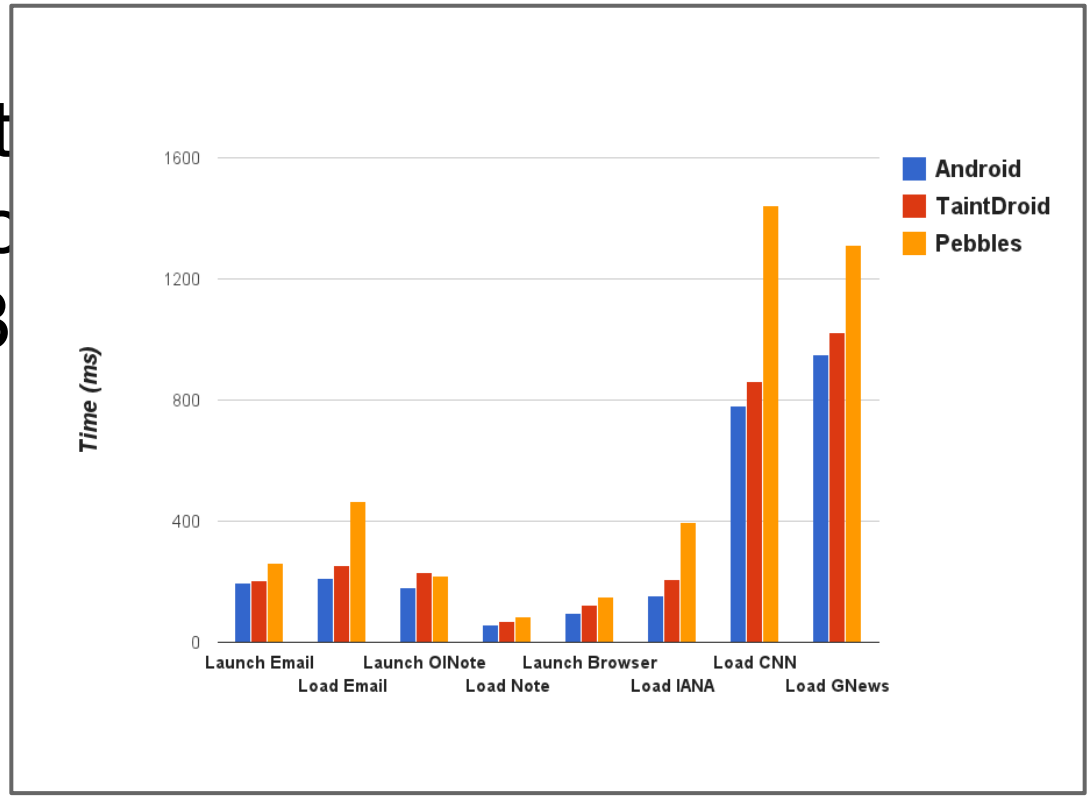
Overinclusion

Pebbles Evaluation

- Pebbles worked correctly identified 60 of 68 objects and worked for all objects in 43 of 50 applications.
- 6 objects exhibited leakage
- 2 objects showed overinclusion.

Pebbles Performance

- 20% t
  macro
- 16MB



Future Work

- Can we this object to detection web applications?
- Can we use high level object recognition to improve software development process?

Acknowledgements

Thanks and Acknowledgements

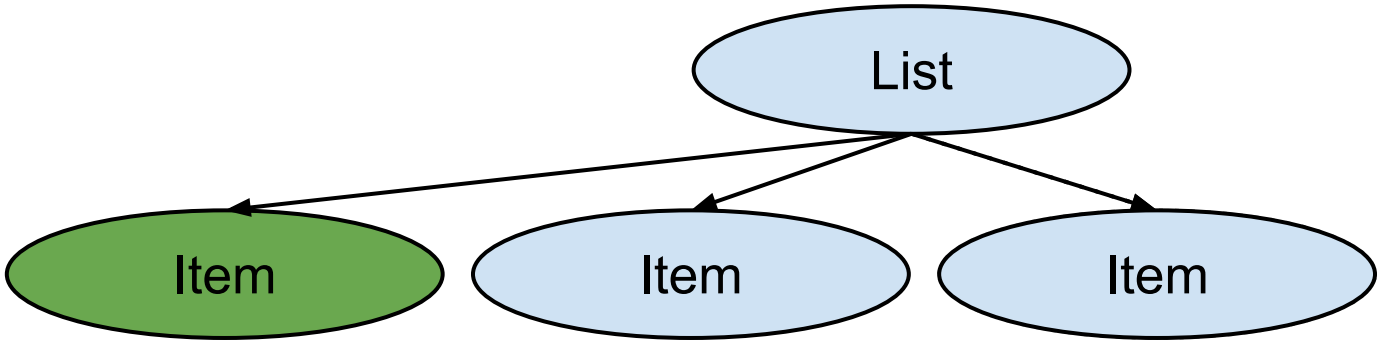
My Sensitive Todo list

- My todo list application has business tasks and my family addresses.
- Can I show the business tasks to a coworker without sharing my family list?

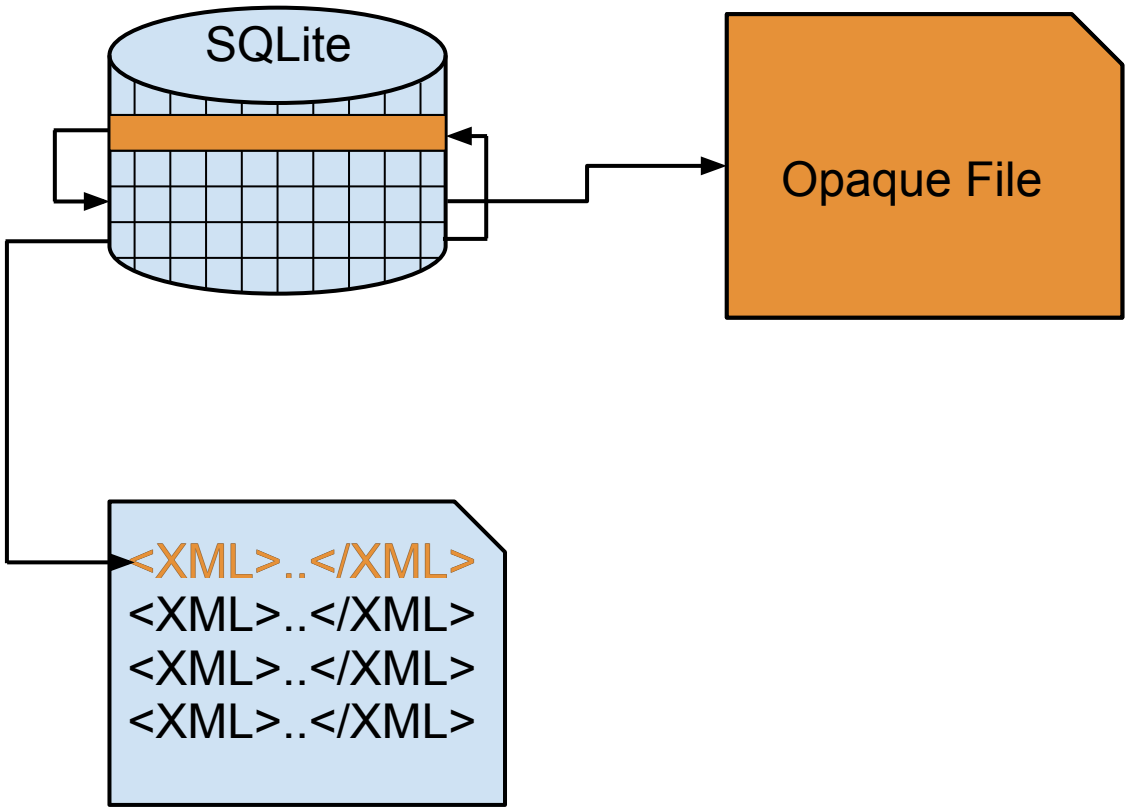
<Wunderlist Image/>

Pebbles's Goals

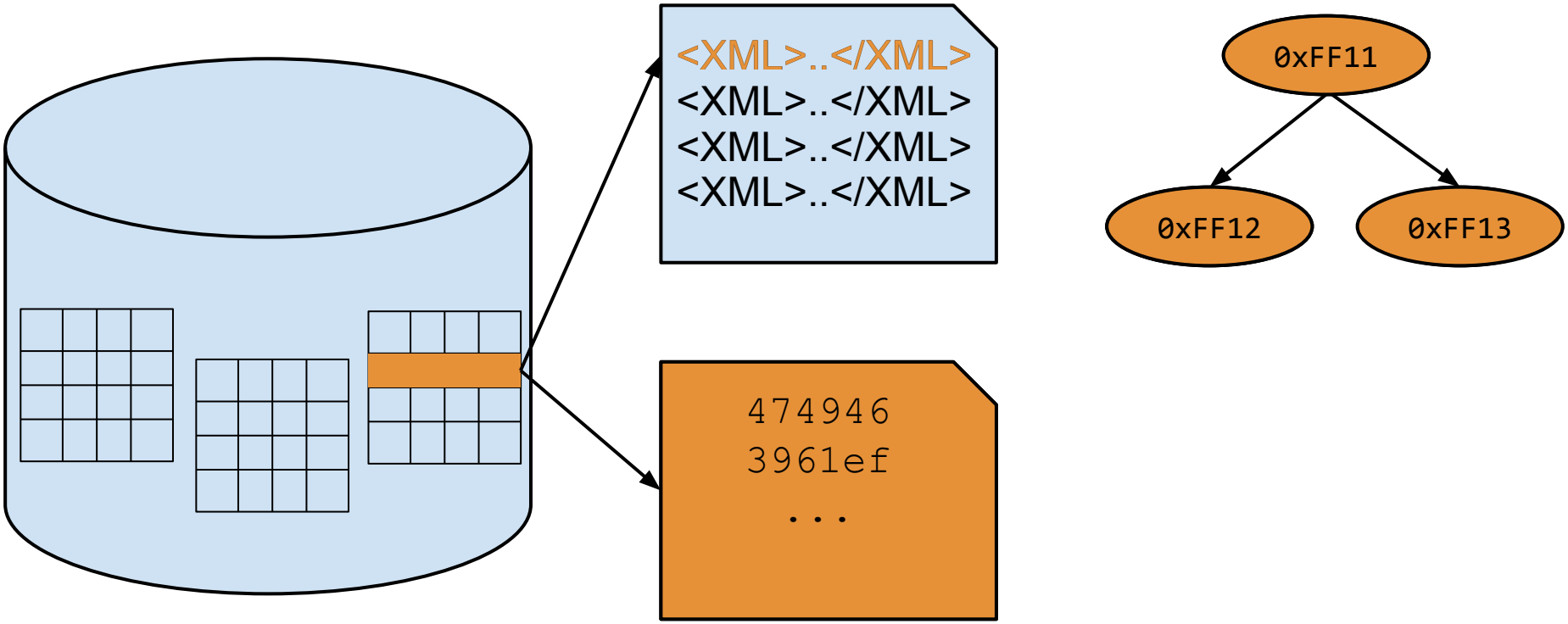
- Accurate and Precise Object Recognition
  - No Leakage
  - No Overinclusion



The Pebbles Object Graph



Logical Data Objects (LDOs)



My Pebbles Object Graph