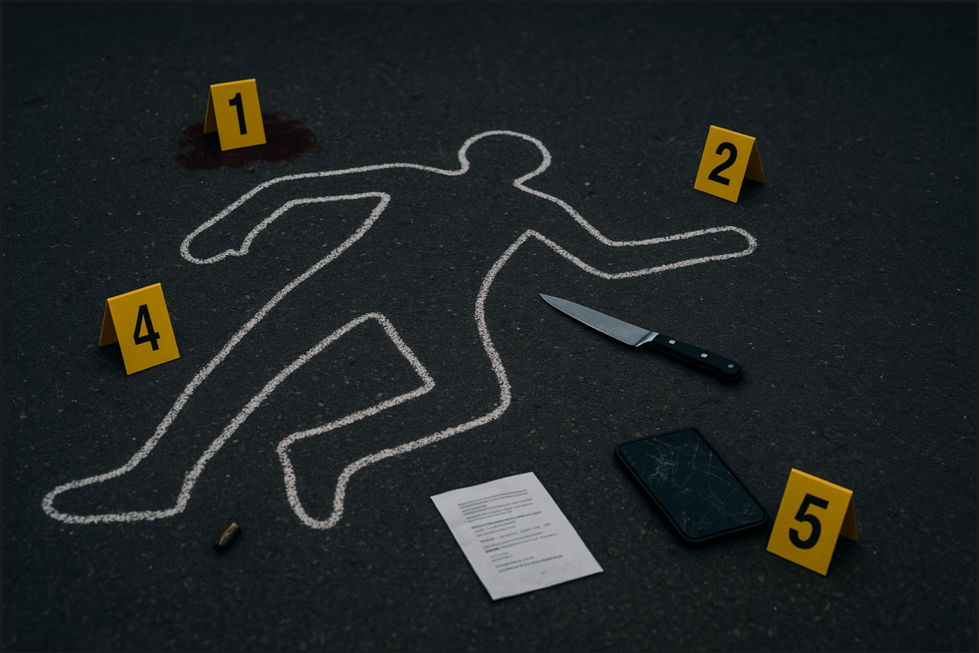
From Sync to Seizure:

A Binary Instrumentation-based Evaluation of the iCloud Backup Process

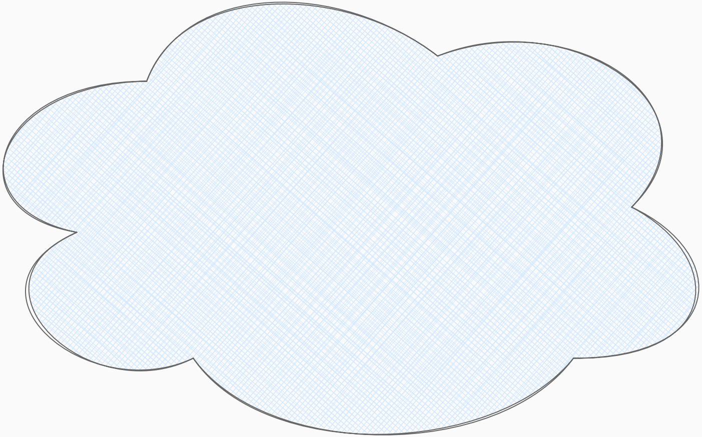
DFRWS APAC ’25

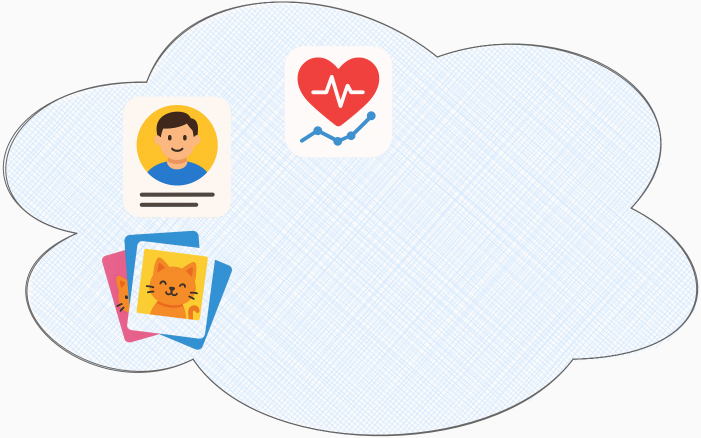
Julian Geus Jan Gruber Jonas Wozar Felix Freiling

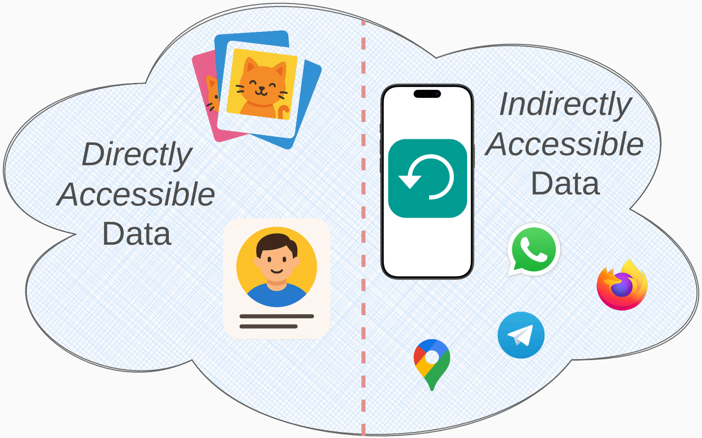
IT Security Infrastructures Lab
Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU)


1
2
4
5


1
2
4
5



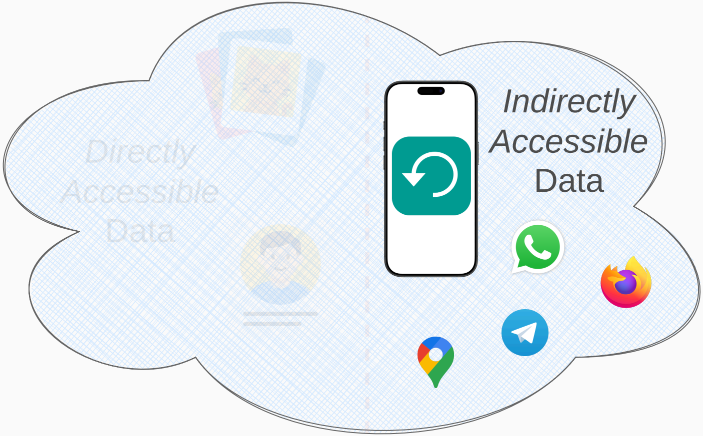



Directly
Accessible
Data
Indirectly
Accessible
Data

Directly Accessible Data on iCloud

- iCloud Web-Interface
- iCloud Client
- Unofficial Web API libraries (e.g., pyicloud[1])

[1] https://github.com/picklepete/pyicloud


Directly Accessible Data
Indirectly Accessible Data

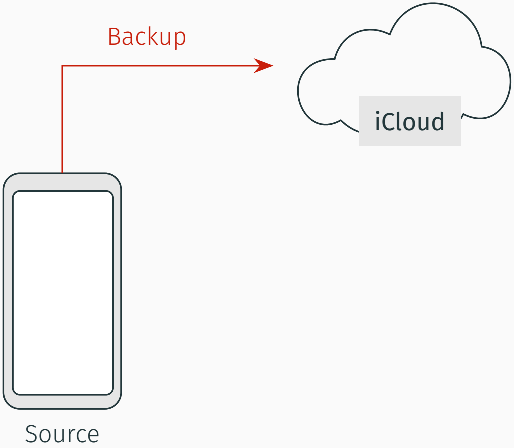
Indirectly Accessible Data on iCloud

Questions:

→ How can we acquire indirectly accessible cloud data (especially backups)?

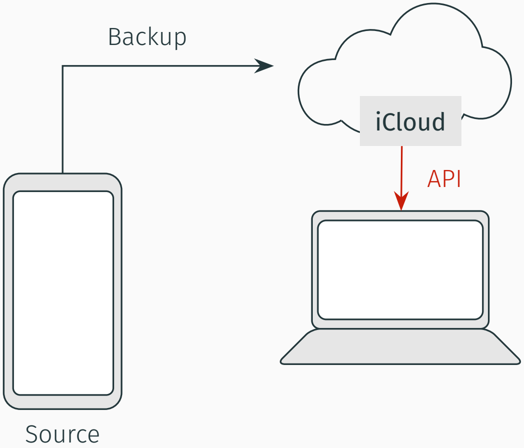
→ Are those methods suitable for forensics?

Acquisition of iCloud Backups



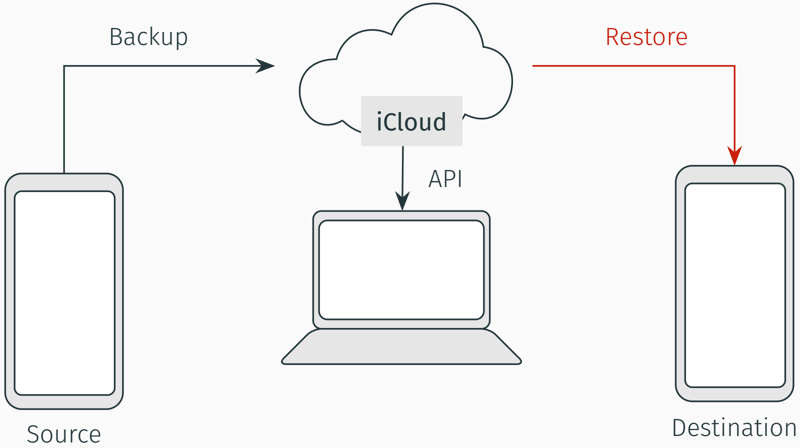
User creates **backup** in the **cloud account**

Acquisition of iCloud Backups



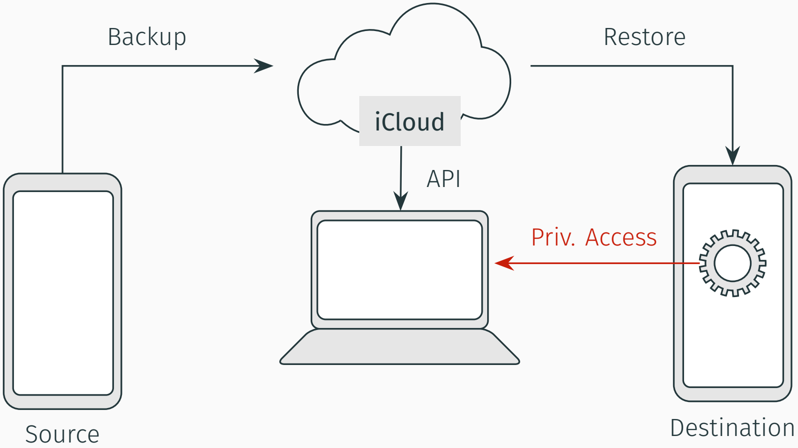
API access requires **reverse engineering** and **API might change**

Acquisition of iCloud Backups



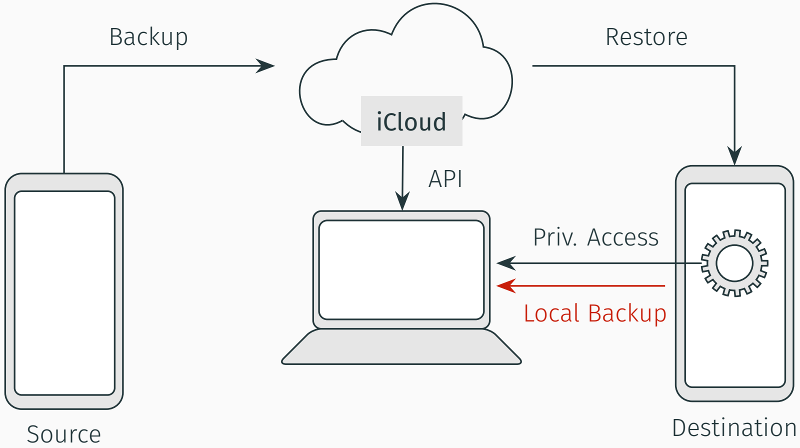
Restore the backup onto an **intermediate device** for **data acquisition**

Acquisition of iCloud Backups



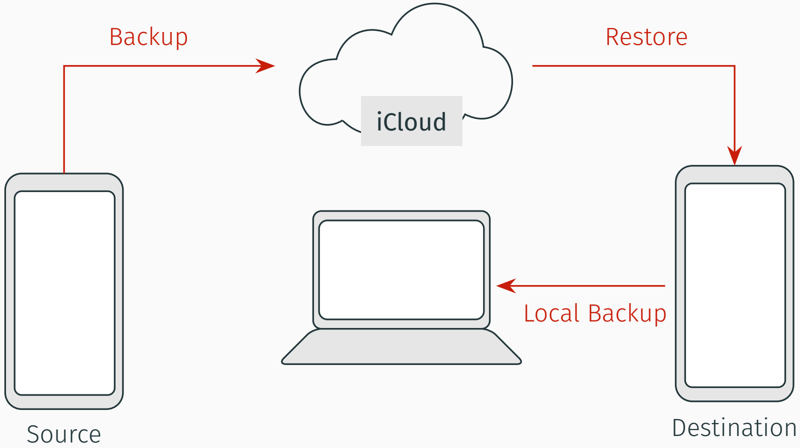
Privileged access enables **full data access** but **identifying the data** is challenging

Acquisition of iCloud Backups



Local backup acquisition is easy to use and should cover a **similar data set**

Acquisition of iCloud Backups



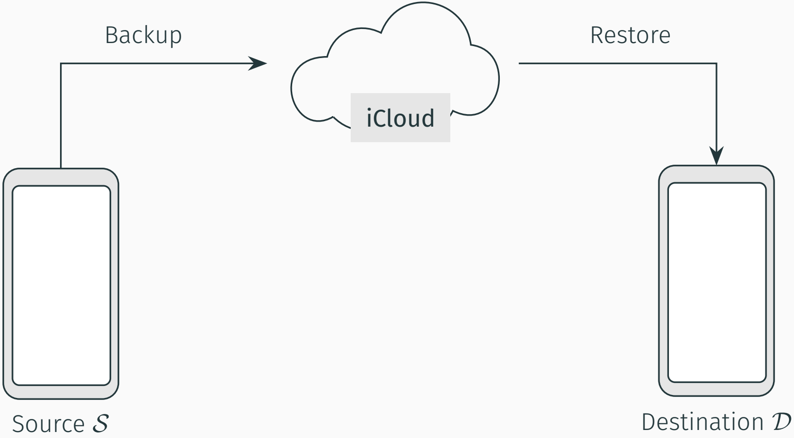
Focus: How well is the **cloud backup restore** acquisition suited for forensics?

Idea:

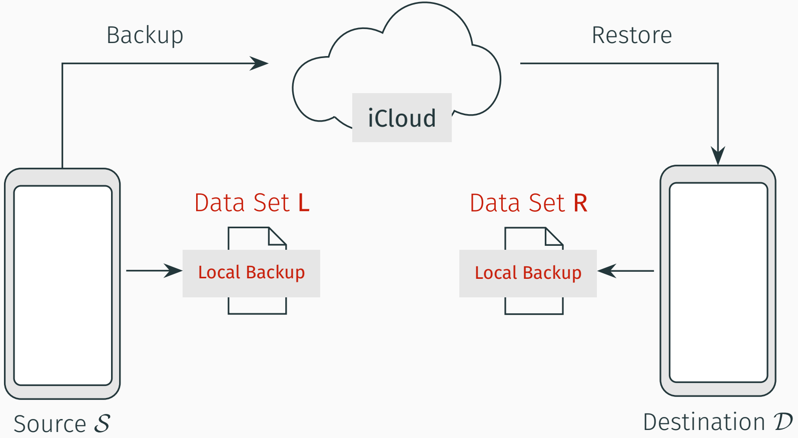
Compare local backups acquired from a source and a destination device!

→ two iPhone 16e with iOS 18.4.1

Evaluation - Approach



Evaluation - Approach



File-based Comparison

1. Identify the amount of counterparts in the data sets:

$$N_{\text{both}} := \text{overlapping files by full path}$$
$$N_{\text{Xonly}} := \text{files only in data set X}$$

2. Compare the files' contents using hash comparison:

$$V_{\text{eq}} := \text{file-pairs with matching hash}$$
$$V_{\text{ch}} := \text{file-pairs with a hash mismatch}$$

Semantic Comparison

1. SQLite database semantic equality by comparing included tables:

$$V_{\sim\text{eq}} := \text{files with semantic equality}$$

2. Determine semantic equality of the entire data set by processing it with *iLeapp*[2] and comparing the TSV reports.

[2] https://github.com/abrignoni/iLEAPP

File-based Comparison

1. Identify the amount of counterparts in the data sets:

$$N_{both} := \text{overlapping files by full path}$$
$$N_{Xonly} := \text{files only in data set X}$$

2. Compare the files' contents using hash comparison:

$$V_{eq} := \text{file-pairs with matching hash}$$
$$V_{ch} := \text{file-pairs with a hash mismatch}$$

Semantic Comparison

1. SQLite database semantic equality by comparing included tables:

$$V_{\sim eq} := \text{files with semantic equality}$$

2. Determine semantic equality of the entire data set by processing it with *iLeapp*[2] and comparing the TSV reports.

[2] `https://github.com/abrignoni/iLEAPP`

Evaluation Results - File-Based

	File Count		Name Comparison			Value Comparison	
	$\lvert\mathscr{S}_L\rvert$	$\lvert\mathscr{D}_R\rvert$	$\lvert N_{both}\rvert$	$\lvert N_{Sonly}\rvert$	$\lvert N_{Donly}\rvert$	$\lvert V_{eq}\rvert$	$\lvert V_{ch}\rvert$
$\sum$	1231	1231	1207	24	25	816 (67.5%)	391 (32.5%)

The number of files is mostly similar between both local backup sets.

Evaluation Results - File-Based

	File Count		Name Comparison			Value Comparison	
	$\lvert \mathscr{S}_L \rvert$	$\lvert \mathscr{D}_R \rvert$	$\lvert N_{both} \rvert$	$\lvert N_{Sonly} \rvert$	$\lvert N_{Donly} \rvert$	$\lvert V_{eq} \rvert$	$\lvert V_{ch} \rvert$
$\sum$	1231	1231	1207	24	25	816 (67.5%)	391 (32.5%)

Most files have a counterpart, deviations might be caused by differing states (i.e., the phone's state is not completely restored).

Evaluation Results - File-Based

	File Count		Name Comparison			Value Comparison	
	$\lvert\mathscr{S}_L\rvert$	$\lvert\mathscr{D}_R\rvert$	$\lvert N_{both}\rvert$	$\lvert N_{Sonly}\rvert$	$\lvert N_{Donly}\rvert$	$\lvert V_{eq}\rvert$	$\lvert V_{ch}\rvert$
$\sum$	1231	1231	1207	24	25	816 (67.5%)	391 (32.5%)

Only about two thirds of the files are hash-identical.

Evaluation Results - Content-Based

	SQLite Semantic Comparison		iLEAPP (TSV) Comparison		
	$\lvert V_{ch} \rvert$	$\lvert V_{\sim eq} \rvert$	*full*	*partial*	*missing*
$\overline{\sum}$	125	87	30	31	5

About 70% of mismatching SQLite databases are actually content-equal.

Evaluation Results - Content-Based

	SQLite Semantic Comparison		iLEAPP (TSV) Comparison		
	$\lvert V_{ch} \rvert$	$\lvert V_{\sim eq} \rvert$	*full*	*partial*	*missing*
$\sum$	125	87	30	31	5

Data changes are mainly related to device differences, missing data includes health and browsing information.

Evaluation Results - Remaining Question

Question:

Where do the changes originate from?

$\rightarrow$ cloud processing, the local backup, or restore processes on the device

Idea:

Add further measurement points!

$\rightarrow$ use jailbroken devices to obtain additional possibilities

Idea:

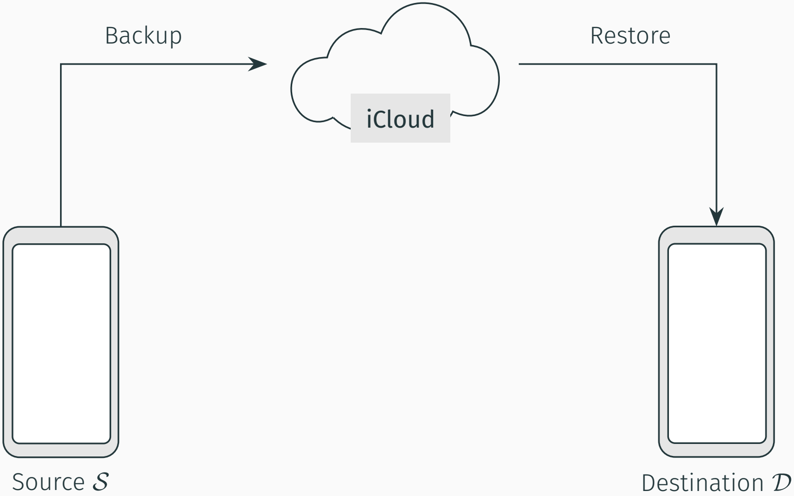
Compare local backups from the source and destination device and add further measurement points in-between!

→ two iPhone 8 with iOS 16.7.10

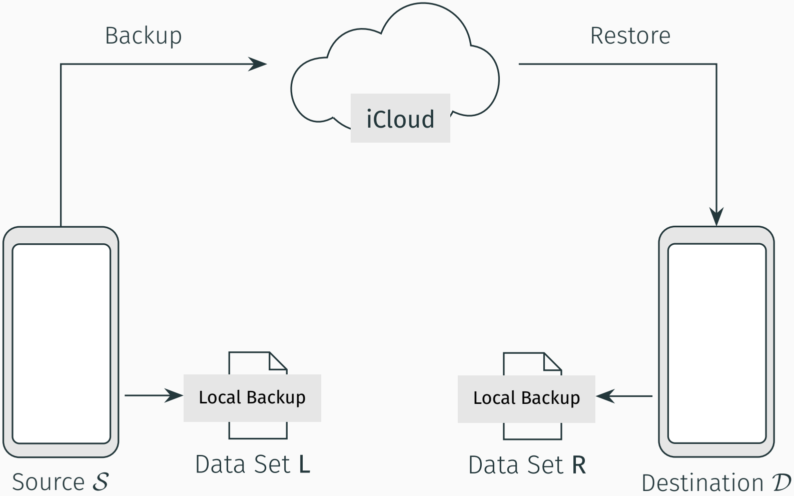
→ jailbreak the devices using *palera1n*[3]

[3] https://github.com/palera1n/palera1n

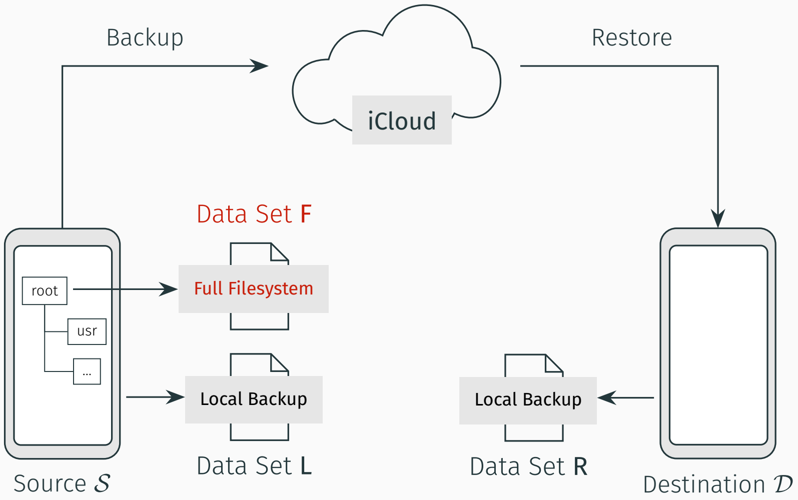
Extended Evaluation - Approach



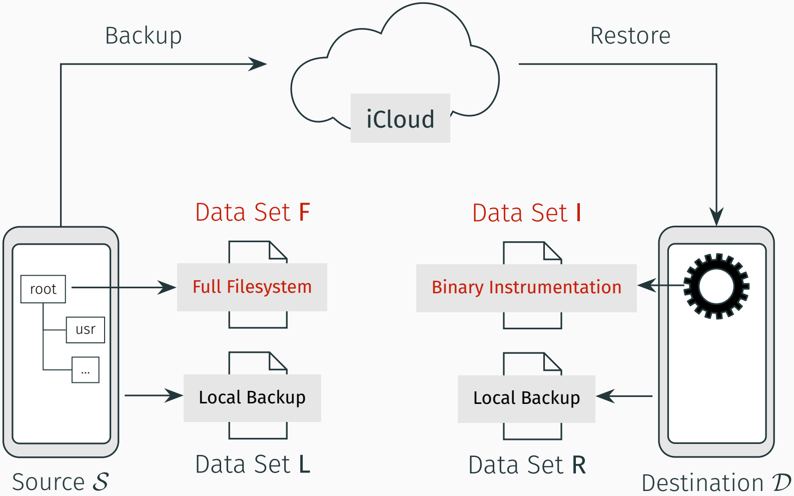
Extended Evaluation - Approach



Extended Evaluation - Approach



Extended Evaluation - Approach



Measurement Point - Binary Instrumentation

1. Reverse Engineering of the iCloud Restore Process
   - identify involved processes with system utilities (e.g., htop, log files, ...)
     → ***cloudd***: communication with iCloud, ***backupd***: restoring backup data
   - implement *strace* for iOS using *Frida*[4] skript
     → **identify system calls** from the respective processes

[4] https://frida.re/

1. Reverse Engineering of the iCloud Restore Process
   - identify involved processes with system utilities (e.g., htop, log files, ...)

     → ***cloudd***: communication with iCloud, ***backupd***: restoring backup data
   - implement *strace* for iOS using *Frida*[4] skript

     → **identify system calls** from the respective processes
2. Process Instrumentation with *Frida*
   - run custom code in *cloudd* and *backupd* context

     → intercept all relevant **`open(...)`** and **`close(...)`** system calls
   - on each **`close(...)`** halt the process and create a copy of the file

[4] `https://frida.re/`

Measurement Point - Binary Instrumentation

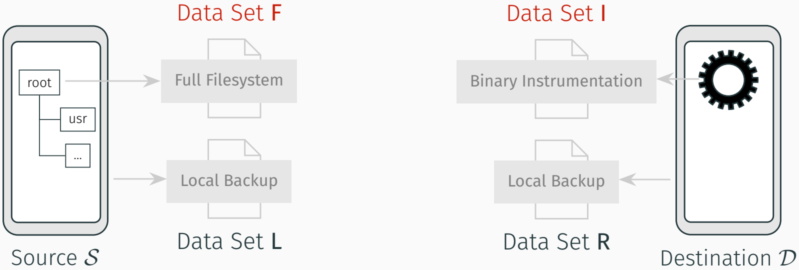
1. Reverse Engineering of the iCloud Restore Process
   - identify involved processes with system utilities (e.g., htop, log files, ...)
     → ***cloudd***: communication with iCloud, ***backupd***: restoring backup data
   - implement *strace* for iOS using *Frida*[4] skript
     → **identify system calls** from the respective processes
2. Process Instrumentation with *Frida*
   - run custom code in *cloudd* and *backupd* context
     → intercept all relevant `open(...)` and `close(...)` system calls
   - on each `close(...)` halt the process and create a copy of the file

⇒ **Acquisition of the entire (unprocessed) data set downloaded from iCloud!**

---

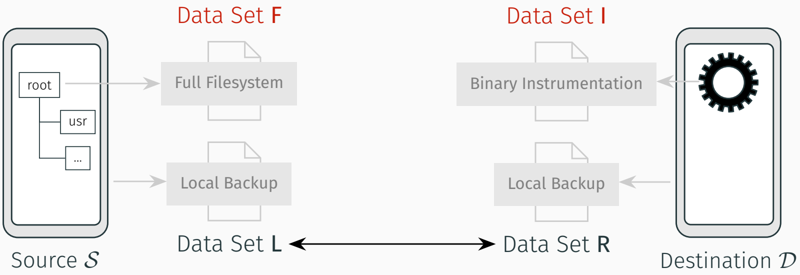
[4] `https://frida.re/`

Extended Evaluation - Comparison



Comparison $L \leftrightarrow R$: Data changes due to the *entire process*.

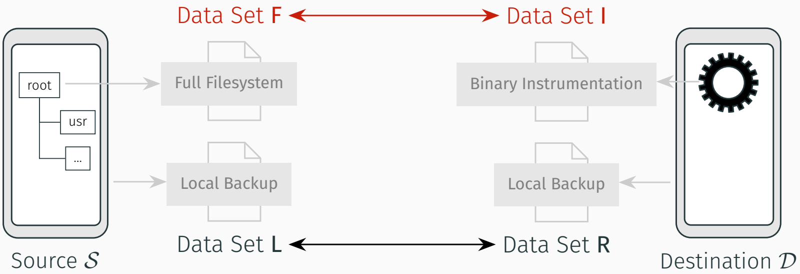
Comparison $F \leftrightarrow I$: Data changes due to the *cloud backup and restore* process.



Comparison $L \leftrightarrow R$: Data changes due to the *entire process*.

Comparison $F \leftrightarrow I$: Data changes due to the *cloud backup and restore* process.

Extended Evaluation - Comparison



Comparison $L \leftrightarrow R$: Data changes due to the *entire process*.

Comparison $F \leftrightarrow I$: Data changes due to the *cloud backup and restore* process.

Results - File-Based

	File Count		Name Comparison			Value Comparison	
	Source	Dest.	$\lvert N_{both}\rvert$	$\lvert N_{Sonly}\rvert$	$\lvert N_{Donly}\rvert$	$\lvert V_{eq}\rvert$	$\lvert V_{ch}\rvert$
	$\lvert\mathscr{S}_L\rvert$	$\lvert\mathscr{D}_R\rvert$	*local backup (L) vs. local backup (R)*				
$\sum$	899	**946**	870	29	72	518 (59.5%)	352 (40.5%)
	$\lvert\mathscr{S}_F\rvert$	$\lvert\mathscr{D}_I\rvert$	*full file system (F) vs. binary instrumentation (I)*				
$\sum$	13256	**1123**	732	12525	391	601 (82.1%)	130 (17.9%)

The number of files in the binary instrumented acquisition exceeds the local backup data set.

Results - File-Based

	File Count		Name Comparison			Value Comparison	
	Source	Dest.	$\lvert N_{both}\rvert$	$\lvert N_{Sonly}\rvert$	$\lvert N_{Donly}\rvert$	$\lvert V_{eq}\rvert$	$\lvert V_{ch}\rvert$
	$\lvert\mathscr{S}_L\rvert$	$\lvert\mathscr{D}_R\rvert$	*local backup (L) vs. local backup (R)*				
$\overline{\sum}$	899	946	870	29	72	518 (59.5%)	352 (40.5%)
	$\lvert\mathscr{S}_F\rvert$	$\lvert\mathscr{D}_I\rvert$	*full file system (F) vs. binary instrumentation (I)*				
$\overline{\sum}$	13256	1123	732	12525	391	601 (82.1%)	130 (17.9%)

There are many files without a counterpart because temporary files are captured during the binary instrumentation.

Results - File-Based

	File Count		Name Comparison			Value Comparison	
	Source	Dest.	$\lvert N_{both} \rvert$	$\lvert N_{Sonly} \rvert$	$\lvert N_{Donly} \rvert$	$\lvert V_{eq} \rvert$	$\lvert V_{ch} \rvert$
	$\lvert \mathscr{S}_L \rvert$	$\lvert \mathscr{D}_R \rvert$	*local backup (L) vs. local backup (R)*				
Σ	899	946	870	29	72	518 (59.5%)	352 (40.5%)
	$\lvert \mathscr{S}_F \rvert$	$\lvert \mathscr{D}_I \rvert$	*full file system (F) vs. binary instrumentation (I)*				
Σ	13256	1123	732	12525	391	601 (82.1%)	130 (17.9%)

Bigger percentage of hash-matches compared to the local backup comparison.
$\rightarrow$ Some data is altered after the download.

Results - Content-Based

	SQLite Semantic Comparison		iLEAPP (TSV) Comparison		
	$\lvert V_{ch} \rvert$	$\lvert V_{\sim eq} \rvert$	*full*	*partial*	*missing*
	local backup (L) vs. local backup (R)				
$\sum$	88	62	28	31	13
	full file system (F) vs. binary instrumentation (I)				
$\sum$	70	65	32	7	77

Again, most mismatching SQLite databases are semantically equal.
$\rightarrow$ Database alterations are caused by the cloud backup process.

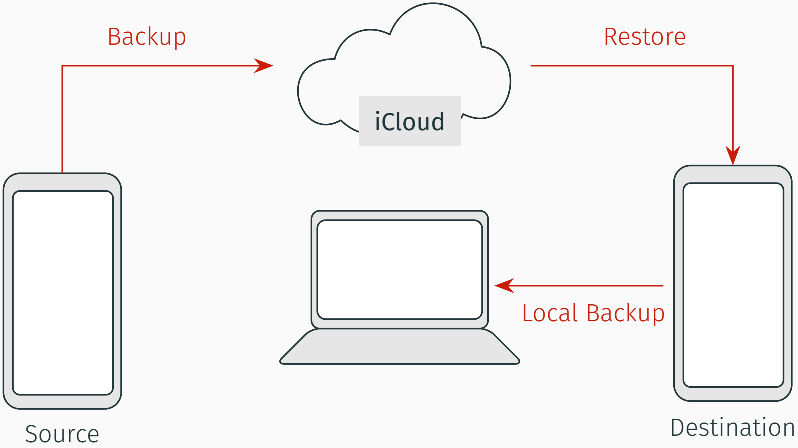
Results - Content-Based

	SQLite Semantic Comparison		iLEAPP (TSV) Comparison		
	$\lvert V_{ch} \rvert$	$\lvert V_{\sim eq} \rvert$	*full*	*partial*	*missing*
	local backup (L) vs. local backup (R)				
$\overline{\sum}$	88	62	28	31	13
	full file system (F) vs. binary instrumentation (I)				
$\overline{\sum}$	70	65	32	7	77

A smaller data set compared to local backup acquisition.

$\rightarrow$ Parsing errors from iLeapp and possibly unknown synchronization processes.

Reminder: Research Question



Focus: How well is the **cloud backup restore** acquisition suited for forensics?

Results

1. iCloud backups include less data compared to (encrypted) local backups.

→ More data can be accessed with synchronization!

2. SQLite databases are altered during the cloud backup process.

→ Their content, however, stays the same.

3. No semantic changes were observed that limit the forensic usability!

→ Nevertheless, files are changed according to their hash values.

Thank you for your attention!

Any questions or comments?

Results

1. iCloud backups include less data compared to (encrypted) local backups.

→ More data can be accessed with synchronization!

2. SQLite databases are altered during the cloud backup process.

→ Their content, however, stays the same.

3. No semantic changes were observed that limit the forensic usability!

→ Nevertheless, files are changed according to their hash values.

Thank you for your attention!

Any questions or comments?

Results

1. iCloud backups include less data compared to (encrypted) local backups.

→ More data can be accessed with synchronization!

2. SQLite databases are altered during the cloud backup process.

→ Their content, however, stays the same.

3. No semantic changes were observed that limit the forensic usability!

→ Nevertheless, files are changed according to their hash values.

Thank you for your attention!

Any questions or comments?

Results

1. iCloud backups include less data compared to (encrypted) local backups.

→ More data can be accessed with synchronization!

2. SQLite databases are altered during the cloud backup process.

→ Their content, however, stays the same.

3. No semantic changes were observed that limit the forensic usability!

→ Nevertheless, files are changed according to their hash values.

Thank you for your attention!

Any questions or comments?