Protecting Mobile Devices from Physical Memory Attacks with Targeted Encryption

Le Guan, Chen Cao, Sencun Zhu, Jingqiang Lin, Peng Liu, Yubin Xia, and Bo Luo
Why do Physical-space Threats Concern for SmartPhones?
Why do Physical-space Threats Concern for SmartPhones?

• Smartphones are easy to be lost or stolen
• Powered-on smartphones run hundreds of background apps
• Once stolen/lost, attackers physically possess the smartphones and sensitive data retain on the phone
  • Password, bank account, health data, etc.

https://patriotpower.ogsd.net/2650/news/the-lost-phone-retriever/
Why do Physical-space Threats Concern for SmartPhones?

• Smartphones are easy to be lost or stolen
• Powered-on smartphones run hundreds of background apps
• Once stolen/lost, attackers physically possess the smartphones and sensitive data retain on the phone
  • Password, bank account, health data, etc.

https://www.theexplode.com/stolen-phone-by-imei-number/
Why do Physical-space Threats Concern for SmartPhones?

• Smartphones are easy to be lost or stolen
• Powered-on smartphones run hundreds of background apps
• Once stolen/lost, attackers physically possess the smartphones and sensitive data retain on the phone
  • Password, bank account, health data, etc.
Why do Physical-space Threats Concern for SmartPhones?

• Smartphones are easy to be lost or stolen
• Powered-on smartphones run hundreds of background apps
• Once stolen/lost, attackers physically possess the smartphones and sensitive data retain on the phone
  • Password, bank account, health data, etc.
Why do Physical-space Threats Concern for SmartPhones?

• Smartphones are easy to be lost or stolen

• Powered-on smartphones run hundreds of background apps

• Once stolen/lost, attackers physically possess the smartphones and sensitive data retain on the phone
  • Password, bank account, health data, etc.
DRAM is a Low-hanging Fruit for Attackers
DRAM is a Low-hanging Fruit for Attackers
DRAM is a Low-hanging Fruit for Attackers
DRAM is a Low-hanging Fruit for Attackers
DRAM is a Low-hanging Fruit for Attackers

• When the smartphone is locked, how can an attacker extract sensitive data?

• Modern smartphones enforce full disk encryption

• Off-chip DRAM is problematic!
DRAM is a Low-hanging Fruit for Attackers

• When the smartphone is locked, how can an attacker extract sensitive data?
• Modern smartphones enforce full disk encryption
• Off-chip DRAM is problematic!
Attacks to DRAM
Attacks to DRAM
Attacks to DRAM

• DDR bus monitoring
• Cold boot attack
Attacks to DRAM

• DDR bus monitoring
• Cold boot attack

https://www.futureplus.com
Attacks to DRAM

- DDR bus monitoring
- Cold boot attack

https://www1.informatik.uni-erlangen.de/frost
MemVault – Memory Vault

- Avoid using DRAM to store cleartext sensitive data

<table>
<thead>
<tr>
<th></th>
<th>Immunity to Physical Attacks</th>
<th>Capacity</th>
<th>Controllability</th>
<th>Intrusiveness</th>
</tr>
</thead>
<tbody>
<tr>
<td>OCRAM/iRAM</td>
<td>✓</td>
<td>~ 128 - 256 KB</td>
<td>Memory Mapped</td>
<td>Not in used after booting</td>
</tr>
<tr>
<td>Cache</td>
<td>✓</td>
<td>~ 1 MB</td>
<td>Transparent</td>
<td>Always in used</td>
</tr>
</tbody>
</table>

- Processor Core Logic
- Instruction Cache
- Data Cache
- External RAM Controller
- On-Chip RAM/iRAM
- Off-Chip DRAM
- Off-Chip ROM
Why OCRAM/iRAM is immune to Physical Attacks?

• DDR bus monitoring
  • No external pins

• Cold boot attack
  • Attacker cannot remove OCRAM/iRAM and install it to another machine
  • SoC bootup code is mandatory for SoC to reboot
  • The code clears OCRAM/iRAM automatically
Questions to Answer

• iRAM has limited size
  • Encrypt data on DRAM
  • Leave “hot” data in cleartext in iRAM

• Performance overhead
  • Only encrypt sensitive data

• How to determine sensitive data?
  • Let developers tell us

• Developers cannot tell if intermediate results are sensitive
  • Taint analysis based on TaintDroid
  • Developers only determine the taint source
MemVault – Overview

Stack Frames

DRAM

Tainted Object

T1

S

T2

Taint Source

Untitled Object

Dummy Stack Frame

DRAM

Encrypted Object

T1

S

T2

T2

iRAM Vault

T2

S
MemVault – Stack Protection

• Local variables on the interpreter stack
• If a variable is tainted, the stack frame is moved to iRAM
• No tainted value is ever written to the original stack frame
• New stack frame in iRAM has a pointer to track the origin stack frame for stack maintenance
MemVault – Object Protection

- A trampoline for each object
- If pointer to trampoline is NULL, the object is never tainted
- If the trampoline pointer is non-NULL, the object might be tainted and the object is encrypted
  - If `iramObj` is null, the encrypted object is decrypted to iRAM
  - If `iramObj` is non-null, the cleartext object is directly addressable
- Next and previous for LRU
  - iRAM has limited space
- Cleartext references for GC
Key Management

• Key is randomly generated per app
• AES in CTR mode
• Virtual address of the object is used as IV
• Key and key schedules are also kept in iRAM
• Key is discarded when the app terminates
Implementation

• On top of TaintDroid (port to Android 4.4.3)
• Encryption/decryption is implemented as a redirection layer of the interpreter

<table>
<thead>
<tr>
<th>Instruction Format</th>
<th>Instruction Semantics</th>
<th>Instrumentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>move-op-R vA</td>
<td>vA ← R</td>
<td>S_DS &amp; S_IS</td>
</tr>
<tr>
<td>iget-op vA vB fC</td>
<td>vA ← vB(fC)</td>
<td>R &amp; S_DS &amp; S_IS</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

S_IS: Switch to iRAM stack, if working on DRAM stack and the resulting stack is tainted
S_DS: Switch to DRAM stack, if working on iRAM stack and the resulting stack is untainted
R: Redirect object access if necessary
Evaluation

- WordPress
  - Password
- BankDroid
  - Account Number
  - Password
- KeePass
  - MasterKey
  - Password
- K-9 email client
  - Password
  - Email

```java
private synchronized void loadAccount(Preferences preferences) {
    Storage storage = preferences.getStorage();
    mStoreUri = Base64.decode(storage.getString(mUuid + ".storeUri", null));
    + MemVault.addTaintArray(mStoreUri);
    ...
}
```

Code snippet of K-9 email client
Evaluation - Performance

<table>
<thead>
<tr>
<th></th>
<th>WordPress</th>
<th>BankDroid</th>
<th>KeePass</th>
<th>K-9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Android</td>
<td>985</td>
<td>239</td>
<td>79</td>
<td>269</td>
</tr>
<tr>
<td>TaintDroid</td>
<td>1001</td>
<td>247</td>
<td>82</td>
<td>277</td>
</tr>
<tr>
<td>MemVault</td>
<td>1008</td>
<td>248</td>
<td>83</td>
<td>277</td>
</tr>
</tbody>
</table>

App Start Time (in ms)

Additional Power Consumption

- TaintDroid: + 18.8%
- MemVault: + 37.2%
## Comparison with Existing Memory Encryption Solutions

<table>
<thead>
<tr>
<th>Solution</th>
<th>Architecture</th>
<th>Software Environment</th>
<th>Granularity</th>
<th>Code Modification</th>
<th>Memory Limitation</th>
<th>Overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cryptkeeper</td>
<td>x86</td>
<td>Linux</td>
<td>4 KB</td>
<td>None</td>
<td>✓</td>
<td>1.09x~9.00x</td>
</tr>
<tr>
<td>RamCrypt</td>
<td>x86</td>
<td>Linux</td>
<td>4 KB</td>
<td>None</td>
<td>✓</td>
<td>1.25x~2.70x</td>
</tr>
<tr>
<td>Bear</td>
<td>ARM</td>
<td>Micro-Kernel</td>
<td>16B ~ 128 KB</td>
<td>Significant</td>
<td>✓</td>
<td>1.50x~3.40x</td>
</tr>
<tr>
<td>Esorics’17</td>
<td>x86</td>
<td>Linux</td>
<td>16 B</td>
<td>None/Significant</td>
<td>✓</td>
<td>1.17x~10.00x+</td>
</tr>
<tr>
<td>Case</td>
<td>ARM</td>
<td>Slef-contained</td>
<td>Whole app</td>
<td>Significant</td>
<td>32 KB</td>
<td>1.03x</td>
</tr>
<tr>
<td>Sentry</td>
<td>ARM</td>
<td>Android</td>
<td>4 KB</td>
<td>None</td>
<td>✓</td>
<td>1.48x~2.74x</td>
</tr>
<tr>
<td>MemVault</td>
<td>ARM</td>
<td>Android</td>
<td>Object</td>
<td>Trivial</td>
<td>✓</td>
<td>1.37x</td>
</tr>
</tbody>
</table>
Conclusion

• MemVault is able to minimize the exposure of sensitive data in DRAM
• MemVault only needs minor modifications to the source code
• MemVault selectively encrypts sensitive data to improve performance

• Limitations
  • MemVault only protects data within Dalvik virtual machine
    • E.g., the buffer of the touchscreen driver cannot be protected
  • TaintDroid has false negative

• Future direction
  • Chip level full memory encryption (like Intel SGX or AMD SME)
Thanks!

leguan@cs.uga.edu