Data Security in Mobile Computing Devices

A Guest Lecture for CS 4491/7990 Deep Learning for Cybersecurity (Kennesaw State University)

Bo Chen, PhD
Department of Computer Science
Michigan Technological University

https://cs.mtu.edu/~bchen
https://snp.cs.mtu.edu
bchen@mtu.edu
About Me

• Assistant Professor in Department of Computer Science

Bo Chen
Assistant Professor, Computer Science

906-487-3149
bchen@mtu.edu
Rekhi 301

Areas of Expertise
• Mobile Device Security
• Cloud Computing Security
• Named Data Networking Security
• Big Data Security
• Blockchain

Links of Interest
Faculty Website

https://cs.mtu.edu/~bchen
The Secure and Privacy (SnP) lab at Michigan Technological University was established in early 2018. The mission of SnP lab is to promote research and education of cybersecurity. For research, we aim to tackle cutting-edge security and privacy problems, protecting safety and assets of people from malicious attacks. For education, we are enthusiastic about broadcasting cybersecurity knowledge among graduate and undergraduate students. We are also dedicated to promoting cybersecurity training among underrepresented groups and future cybersecurity professionals through various outreach efforts.

PhD Students
Niusen Chen
Wen Xie
Weijing You (UCAS)

Master Students
Shashank Reddy Danda
Jonah Schulte
Deepti Tandasala

Undergraduate Students
Dominika Bobik

https://snp.cs.mtu.edu
About Security and Privacy (SnP) Lab

- Projects are currently funded by national science foundation, national security agency, etc.
  - Protecting sensitive data in mobile devices, IoT devices
  - Protecting critical data/infrastructures outsourced to public clouds
  - Blockchain and information centric networking
  - Leveraging mobile devices for COVID-19 mitigation (recently)
Mobile Devices are Turning to Mainstream Computing Devices

Number of smartphone users worldwide from 2014 to 2020 (in billions)

Number of tablet users worldwide from 2013 to 2021 (in billions)

11/02/2020
CS 4491/7990 Kennesaw State University
Mobile Devices are Turning to Mainstream Computing Devices (cont.)

Number of connected wearable devices worldwide from 2016 to 2021 (in millions)
Mobile Devices are Turning to Mainstream Computing Devices (cont.)

11/02/2020

CS 4491/7990 Kennesaw State University
Mobile Computing Devices Are Increasingly Used for Critical Applications

- Mobile devices are increasingly used to handle sensitive data
  - Online banking
  - Ecommerce
  - Cryptocurrency/stock trading
  - Naked photos
  - A human rights worker collects evidence of atrocities in a region of oppression
  - Etc.

- Security issues in mobile computing devices
  - Confidentiality (C)
  - Integrity (I)
  - Access control (A)
  - Authentication
  - Etc.
Background on Storage Systems of Mobile Computing Devices
NAND Flash Memory

- Flash memory
  - NAND flash (broadly used for mass-storage devices)
  - NOR flash (used for storing program code that rarely needs to be updated, e.g., a computer’s BOIS)

- NAND flash organization
  - Block
  - Page

Page (e.g., 4KB)
Block (e.g., 128KB)

11/02/2020
CS 4491/7990 Kennesaw State University
Flash Memory Programming

All ‘1’ initially:

Write 0x0b:
Rule:
1) 1 can be programmed to 0
2) 0 cannot be programmed to 1 except performing an erasure

Write 0xf?

Need to erase to all ‘1’ first

11/02/2020
CS 4491/7990 Kennesaw State University
Special Characteristics of Flash Memory

• Update unfriendly
  – Over-writing a page requires first erasing the entire block
  – Write is performed in pages (e.g., 4KB), but erase is performed in blocks (e.g., 128KB)
  – Over-write may cause significant write amplification
  – Usually prefer out-of-place update instead of in-place update
Special Characteristics of Flash Memory (cont.)

• Support a finite number of program-erase (P/E) cycles
  – Each flash block can only be programmed/erased for a limited number of times (e.g., 10K)
  – Data should be placed evenly across flash (wear leveling)
How to Manage NAND Flash

• Flash-specific file systems, which can handle the special characteristics of NDND flash
  – YAFFS/YAFFS2, UBIFS, F2FS, JFFS/JFFS2

• Flash translation layer (FTL) – a flash firmware embedded into the flash storage device, which can handle the special characteristics of NAND flash and emulate the flash storage as a regular block device (most popular)
  – SSD
  – USB
  – SD / miniSD/MicroSD
  – MMC cards
  – UFS
Flash Translation Layer (FTL)

FTL usually provides the following functionality:
✓ Address translation
✓ Garbage collection (GC)
✓ Wear leveling (WL)
✓ Bad block management
Plausibly Deniable Encryption (PDE) Systems for Mobile Devices
Coercive Attack against Confidentiality

- To protect confidentiality of sensitive data, we can simply encrypt them
  - AES/3DES
  - Full disk encryption (FDE): TrueCrypt, BitLocker, FileVault, etc

- Conventional encryption is vulnerable to a coercive attack

An attacker forces the device’s owner to disclose the decryption key
Plausible Deniable Encryption (PDE)

- Plausible Deniable Encryption (PDE) [Canetti et al., CRYPTO ’97]: a crypto primitive designed for mitigating coercive attacks
  - Disclose the decoy key
  - Keep the true key secret
• Issues: the size of ciphertext is increased. Deniability is easily compromised.
Implementing PDE in Systems - Hidden Volume

- Hidden volume [TRUECRYPT ’04] realizes the concept of PDE in systems
  - Only the decoy key will be disclosed
  - The encrypted hidden volume cannot be differentiated from the random noise
The Challenges: Over-writing Issues

- The data written to the public volume may over-write the data in the hidden volume
  - The hidden volume is part of the public volume
The Challenges: Over-writing Issues (cont.)

- File systems really matter for over-write issues
  - FAT allocates blocks sequentially
    - public volume
    - hidden volume
    - no over-write
  - EXT4 does not allocate blocks sequentially
    - public volume
    - hidden volume
    - over-write

It is challenging to allow any file systems to be deployed while mitigating the over-write issues
Our Mobile PDE System [ACSAC ’15]

Key Insights: To realize file system friendly design, a new layer is introduced to decouple the file system and the underlying PDE system

1. Provide virtual volumes to file systems
2. Any block-based file system can be built on a virtual volume
3. Non-sequential allocation on the virtual volume will be converted to sequential allocation on the underlying layer
Evaluation Highlights

- Implemented a prototype of our solution on LG Nexus 4

Throughput (MB/s) from AndroBench
Existing PDE Systems for Mobile Devices

- Most of the existing PDE systems deploy hidden volume on top of the block device
  - Mobiflage [Skillen et al., NDSS ’13]
  - MobiPluto (our work) [Chang et al., ACSAC ’15]
  - MobiCeal (our work) [Chang et al., DSN ’18]

The architecture of mobile storage
Deniability May be Compromised When The Attacker Can Have Access to Flash Memory

Block-based PDE systems

Attacker’s view

deniability compromise

OS/File system

Apps/Android framework

block-based access interface

flash translation layer (FTL)

NAND flash

erase block page

Flash storage (eMMC card/SD card)
Compromise of Existing PDEs Built on top of the block device: Example 1

When the attacker can have access to the flash memory

A flash block partially used by the hidden volume

A page

 Cannot be decrypted

A block

Ciphertext

random

random

random

random

Ciphertext

All “1”

erase

All “1”

11/02/2020

11/02/2020

CS 4491/7990 Kennesaw State University
Compromise of Existing PDEs Built on top of the block device: Example 2

When the attacker can have access to the flash memory

Special flash memory operations like wear leveling and garbage collection in the hidden volume will produce duplicate blocks

block i and block j have duplicate randomness

Refer to our paper published in CCS ’17 for more compromises
Fundamental Reasons for Compromises of The Existing Block-based PDE Systems

- Built on top of block device (outside the black box of flash memory), and cannot manage the internal flash memory
- Unexpected "traces" of hidden sensitive data could be created in the flash memory which is out of the control of the block-based PDE
Key insight 1: move the public/hidden volume design down to the flash translation layer (FTL).

Key insight 2: to mitigate the over-write issue, the hidden volume and the public volume should avoid sharing blocks:

1) The public volume will allocate blocks from the head of the free block pool; active garbage collection will be performed to fill the head of the free block pool.

2) The hidden volume will allocate blocks from the tail of the truly free blocks; active garbage collection will be performed to fill the tail of the truly free blocks.
Enabling Data Recovery in Mobile Devices from Malicious Attacks
Ransomware

- A piece of special malware that infects a computer and restricts access to the computer and/or its files
  - A ransom needs to be paid in order for the restriction to be removed

Figure 6: Indexed growth in total number of observed ransomware strains, December 2015-March 2017

Source: Proofpoint Q1 2017 Quarterly Threat Report
Main Types of Ransomware

- Locker ransomware
- Crypto-ransomware
How to Combat Locker Ransomware?

- Observation: only the system is locked by the ransomware, but the data are stored intact

- Unplug the storage medium (e.g., SSD drives, microSD cards), plug the storage medium to a new computing device, and copy out the data

- Plug the storage device back to the device which has been locked, and re-install/initialize the system, then copy the data back
Crypto-ransomware Defense

• Crypto-ransomware behaviors:
  – Encrypt the victim data, and delete the original data
    • In systems, the delete operation is implemented by **overwriting** the data with garbage data
  – Or encrypt the victim data, and use the ciphertext to **overwrite** the original data

• Data recovery from crypto-ransomware attacks
  – Option 1: obtain the decryption key
    • Pay the ransom: money loss; cannot guarantee the key can work after paying the ransom
    • Extract the key locally: may work if the ransomware uses symmetric encryption, but no guarantee the key can be extracted
  – Option 2: data recovery from backups
    • More reliable
A Challenging Issue When Restoring Victim Data from Backups

- After a computing device is hacked by ransomware, the victim data will be recovered by backups.

- A challenging issue is how to ensure data stored in the victim device is recoverable to the exact point right before the corruption (i.e., corruption point)?

![Diagram showing Ransomware and System I/Os with t₀ - corruption point]
Why Remote Backups Cannot Recover Data at The Corruption Point?

- Each backup operation usually happens periodically (e.g., daily, hourly) **rather than continuously**
  - No enough battery
  - Internet is not necessarily available any time
  - There is no guarantee that the data **at the corruption point** have been backed up remotely

![Diagram showing backup and ransomware I/Os]

- **Ransomware I/Os:**
  - Backup 1
  - Backup 2

- **System I/Os:**
  - Read Write Write Write Read
  - Recovery to this point

This data (i.e., delta) is corrupted, and unrecoverable

11/02/2020
What about Backing Up Delta by OS/Apps Locally?

- Data can be backed up locally after each single write

- This could be problematic:
  - Creating backups after each single write incurs a large overhead
  - The ransomware may compromise the entire OS and all the local backups created at the upper layers may be corrupted and cannot be used for data recovery
Our Observation

Ransomware I/Os:
- Write A
- Write B

System I/Os:
- Read A and B
- Encrypt A to A', overwrite A with A'
- Encrypt B to B', overwrite B with B'

Flash memory layer
- Old data ‘A’ and ‘B’ encrypted by ransomware are still there (temporarily preserved)

File system layer
- Application layer

Logic Page (OS view)
- Physical Page
- Invalid Physical Page

11/02/2020
Our Solution

• A ransomware detection tool is integrated into the flash translation layer
• Each time if no ransomware is detected, a backup is created and stored in the remote server
• Each time if the ransomware is detected, we will restore the data at the corruption point by:
  – Retrieve the latest version of the data from the remote server
  – Extract the delta from the flash memory (the delta will be preserved in the flash memory due to out-of-place update)
  – Apply delta to the latest data version
A Few Additional Questions

• How can we ensure that the temporarily preserved delta will not be reclaimed by garbage collection?
  – Garbage collection in the flash memory storage may reclaim space occupied by invalid data, leading to deletion of the temporarily preserved old data
  – Our solutions:
    • Completely disable garbage collection
    • a phase garbage collection strategy: The garbage collection runs regularly if no ransomware is detected; the garbage collection will be temporarily disable if any ransomware is detected

Le Guan, Shijie Jia, Bo Chen, Fengwei Zhang, Bo Luo, Jingqiang Lin, Peng Liu, Xinyu Xing, and Luning Xia. Supporting Transparent Snapshot for Bare-metal Malware Analysis on Mobile Devices. 2017 Annual Computer Security Applications Conference (ACSAC ’17), Orlando, Florida, USA, December 2017 (Acceptance rate: 19.7%)

Peiying Wang, Shijie Jia, Bo Chen, Luning Xia and Peng Liu. MimosaFTL: Adding Secure and Practical Ransomware Defense Strategy to Flash Translation Layer. The Ninth ACM Conference on Data and Application Security and Privacy (CODASPY ’19), Dallas, TX, USA, March 2019 (Acceptance rate: 23.5%)
Acknowledgments

• The presented research has been supported by US National Science Foundation under grant number 1928349-CNS and 1938130-CNS

• Our cybersecurity education has been supported by a few Gencyber grants from NSA and NSF (co-funded)
Q & A