



Michigan  
Technological  
University



CS5740/4740 Spring 2024: Special Topic on Data Security (2)

# Enabling Secure Data Recovery

Niusen Chen

Department of Computer Science  
Michigan Technological University  
[niusenc@mtu.edu](mailto:niusenc@mtu.edu)

# Data Security

- Data security: protecting digital data, such as those in a database, from destructive forces and from the unwanted actions of unauthorized users, such as a cyberattack or a data breach (wikipedia)



- Laws and regulations for data security
  - Family Educational Rights and Privacy Act (FERPA) (US)
  - Health Insurance Portability and Accountability Act (HIPAA) (US)
  - General Data Protection Regulation (GDPR)(EU): organizations may face significant penalties of up to €20 million or 4% of their annual revenue if they do not comply with the regulation
  - Data Protection Act (DPA) (UK)
  - Personal Information Protection and Electronic Documents Act (PIPEDA) (CA)

# Data Security (cont.)

- Confidentiality/Privacy – has been discussed in our special topic on data security (1)
- Recoverability – **will be discussed in this talk**
- Secure deletion - will be discussed in the next talk

Denning, Dorothy E., and Peter J. Denning. "Data security." ACM Computing Surveys (CSUR) 11, no. 3 (1979): 227-249.

# Ransomware

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



# Main Types of Ransomware

- Locker ransomware
- Crypto-ransomware



# How to Combat Locker Ransomware?

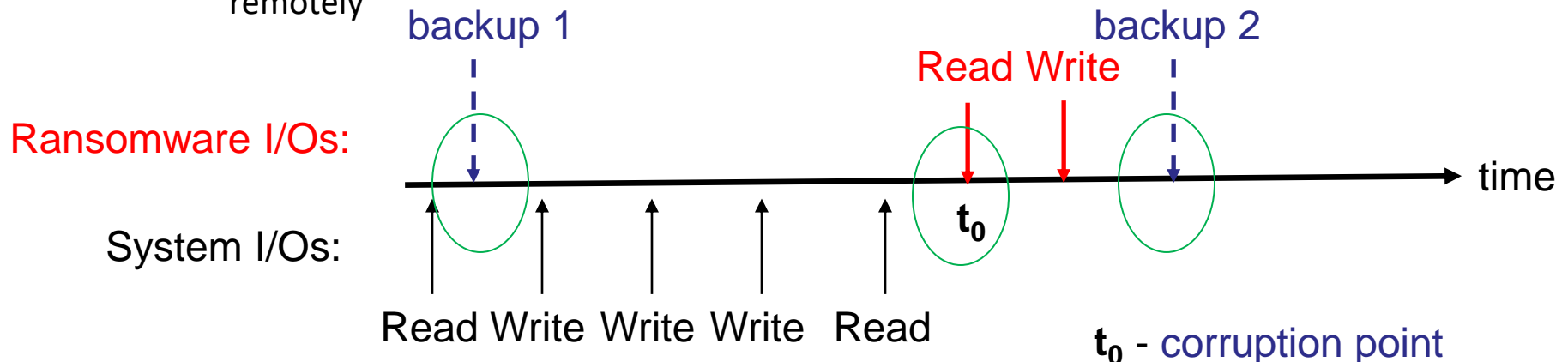
- Observation: only the system is locked by the ransomware, but the data are stored intact
- Unplug the storage medium (e.g., SSD drives, microSD cards), plug the storage medium to a new computing device, and copy out the data
- Plug the storage device back to the device which has been locked, and re-install/initialize 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

# Remote Backups Cannot Ensure Recoverability of Data at The Corruption Point

- Data stored in a computing device may be **periodically** backed up to a remote server (e.g., a cloud server)
  - E.g., iCloud periodically backs up an iPhone
- **The remote backups** cannot ensure recoverability of 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





# What about Doing Backups Locally at The Upper Layers?

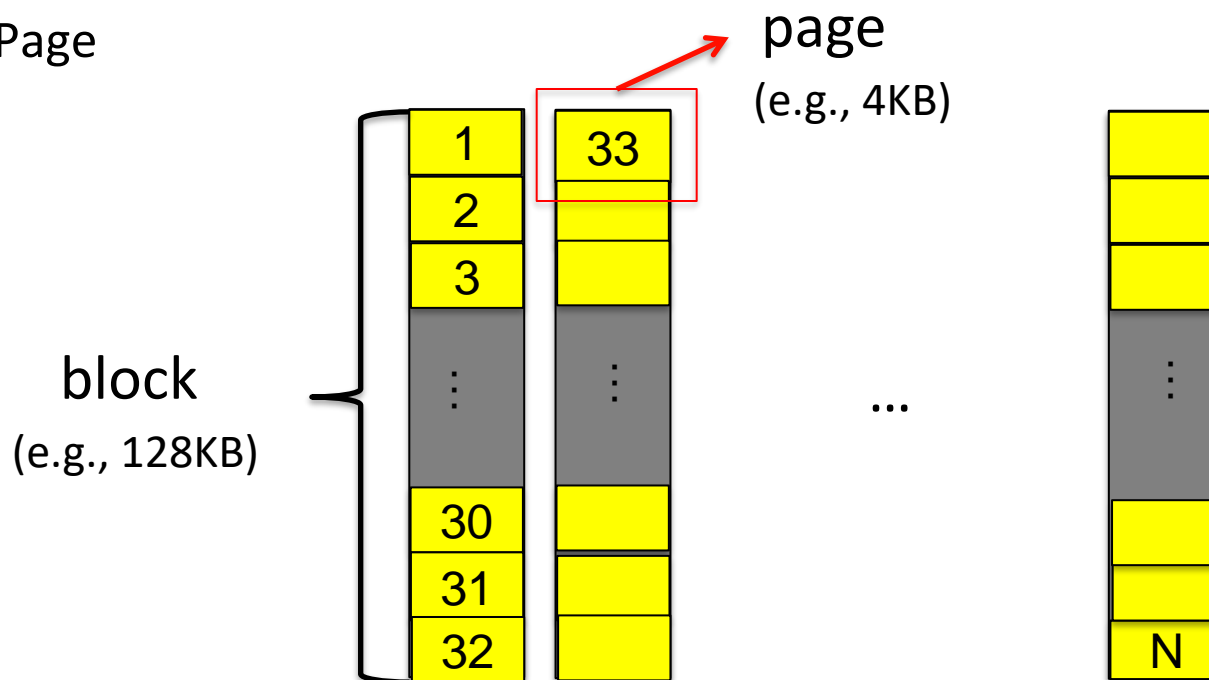
- Data can be backed up locally after each single write
- This could be problematic:
  - Creating backups after each single write incurs **a large overhead**
  - The ransomware may **compromise the entire OS and all the local backups** created at the upper layers may be corrupted and cannot be used for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

# Background on Flash Memory

# NAND Flash Memory

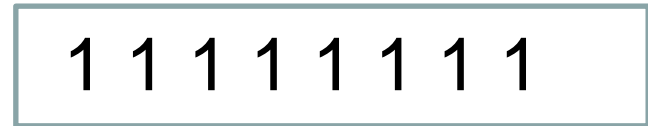


- NAND flash memory as mass storage
  - SSD (used widely in personal computers and servers)
  - Flash memory cards like SD cards, MMC card, UFS cards (used broadly in mobile devices/IoT devices)
- NAND flash organization
  - Block
  - Page



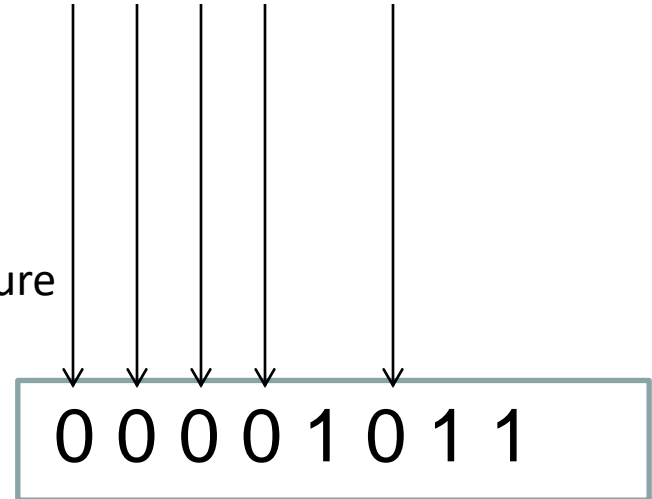
# How to Program Flash Memory?

All '1' initially:

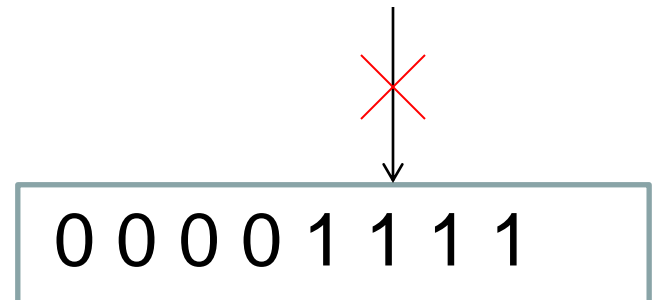


Write **0x0b**:

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



Modify 0x0b to **0x0f**?

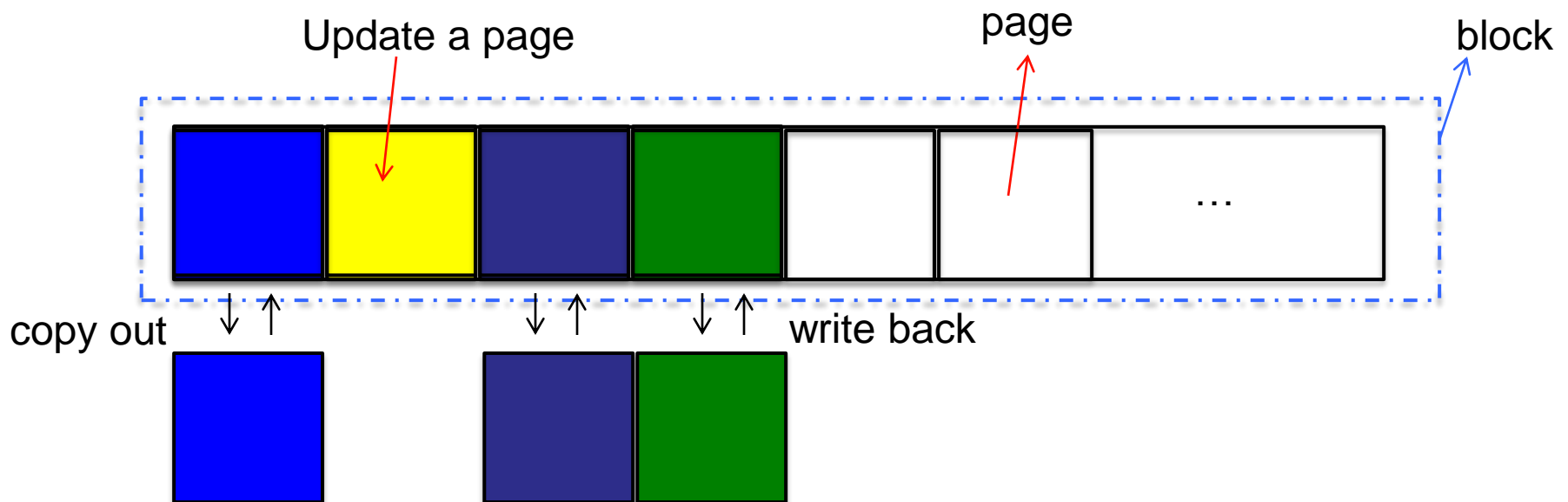


Need to erase to all '1' first

# 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)



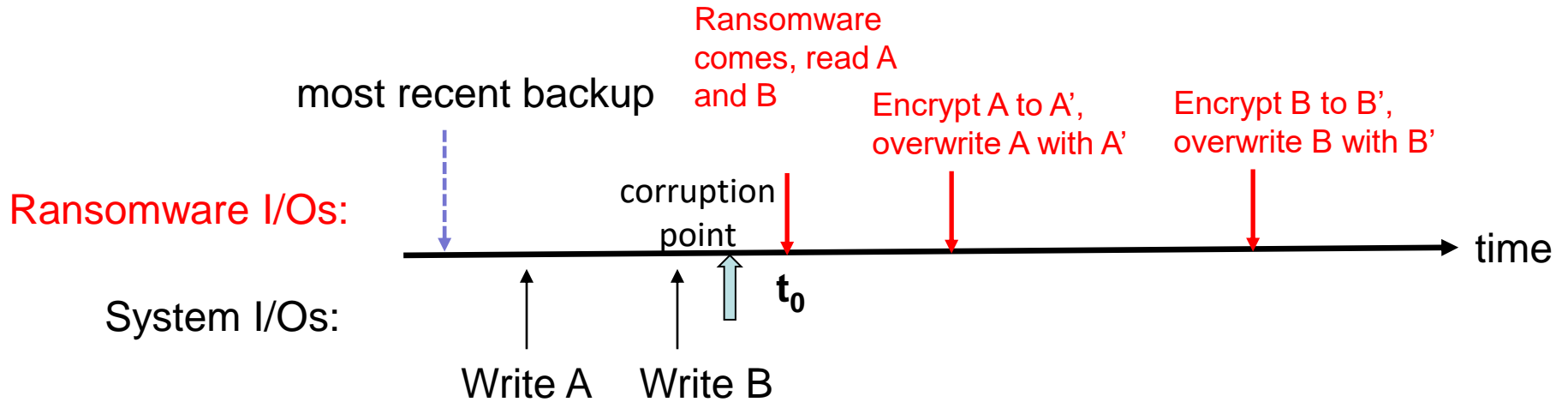
- Over-write may cause significant **write amplification**
- Usually prefer **out-of-place update** instead of in-place update

# 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**)

# Solution on Restoring Data to The Corruption Point after Ransomware Attacks

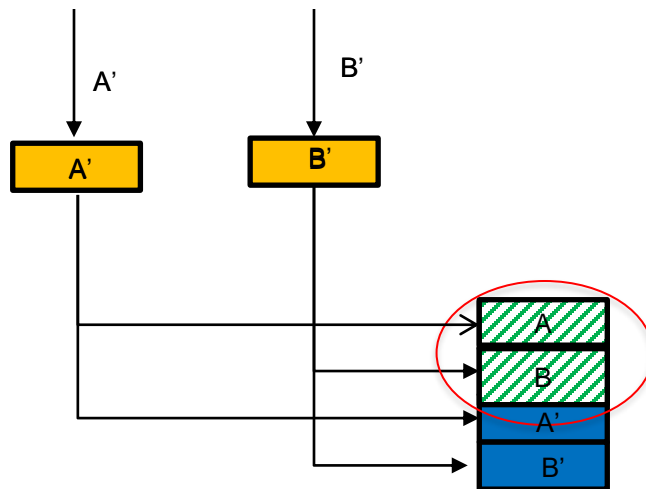
# Towards Restoring The Corruption Point



**Data at the corruption point = most recent backup + (A + B)**

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

Application layer  
File system layer  
Flash memory layer

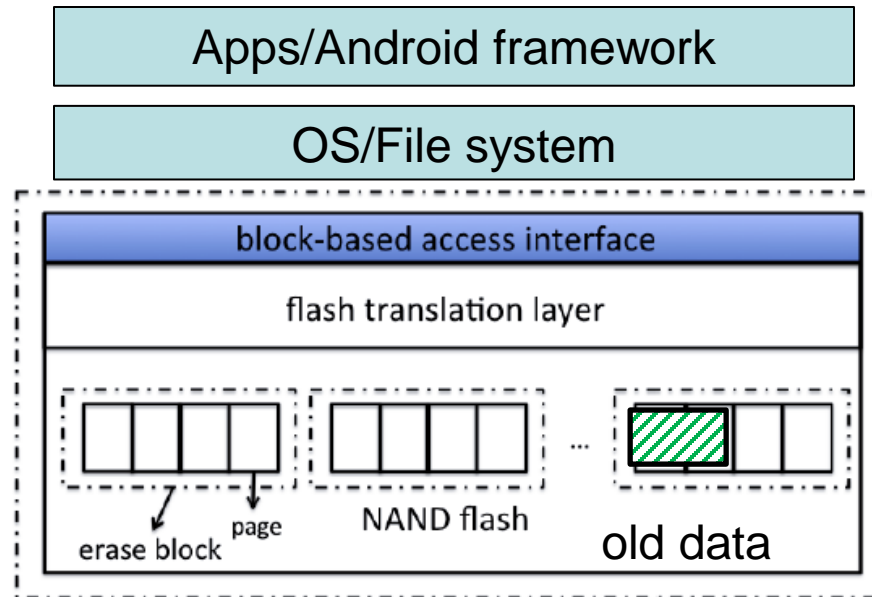


Old data 'A' and 'B' encrypted by ransomware are still there (temporarily preserved)



# Taking Advantage of The Temporarily Preserved Old Data

- The temporarily preserved old data are the exact data encrypted by ransomware at the corruption point
  - They have been invalidated by the FTL and hence are invisible to the OS and apps from upper layers, and will not be “touched” by ransomware which can compromise the entire OS
  - They can be extracted to restore the data at the corruption point



# A Few Additional Questions

- How can we ensure that the temporarily preserved old data 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 solution: temporarily freeze the garbage collection on those flash blocks which hold the data not yet been backed up remotely.

# A Few Additional Questions (cont.)

- Can I recover the data in a fine-grained manner?
  - For example, the victim user can choose which files to be restored rather than restoring the entire storage (slow, or may not be necessary)
- Integrate both file system forensics and the flash memory data extraction

# Acknowledgments

- Our secure data recovery project is currently supported by US National Science Foundation under grant number 2225424-CNS

<https://snp.cs.mtu.edu/research/cloudsec.html>