Forensics MP Checkpoint 2

Pubali Datta
University of Illinois
CS 461 / ECE 422 - Fall 2019

Educational Objectives

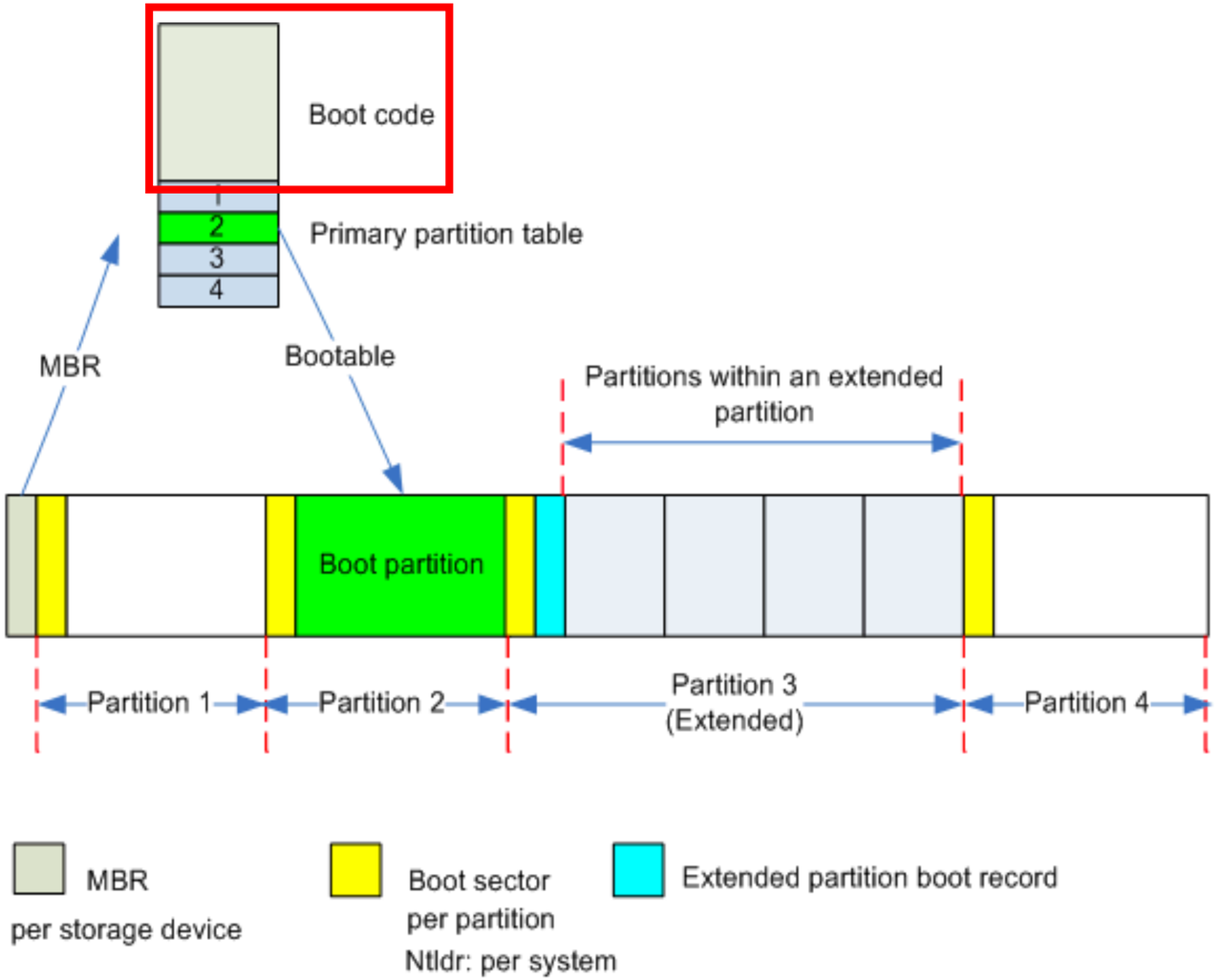
- Understand multi-booting
- Understand SSH connection artifacts
- Understand the difference between file recovery and carving
- How to recover files using file system metadata
- How to carve file contents
- Checkpoint 2 Overview

Multi-booting

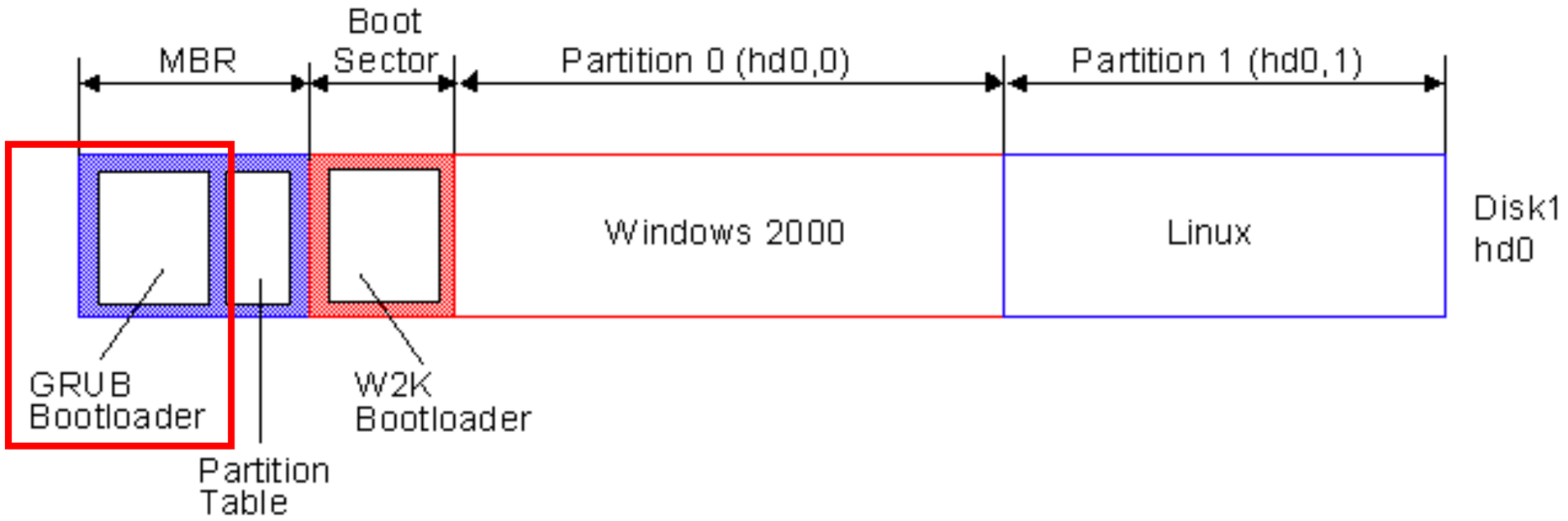
- “The act of installing multiple operating systems on a computer, and being able to choose which one to boot” (https://en.wikipedia.org/wiki/Multi-booting)
- How does the computer know which one to boot?

Recap: MBR and Boot Sectors

- Master Boot Record (MBR) in the first sector of disk
- Boot sector in the first sector of partition



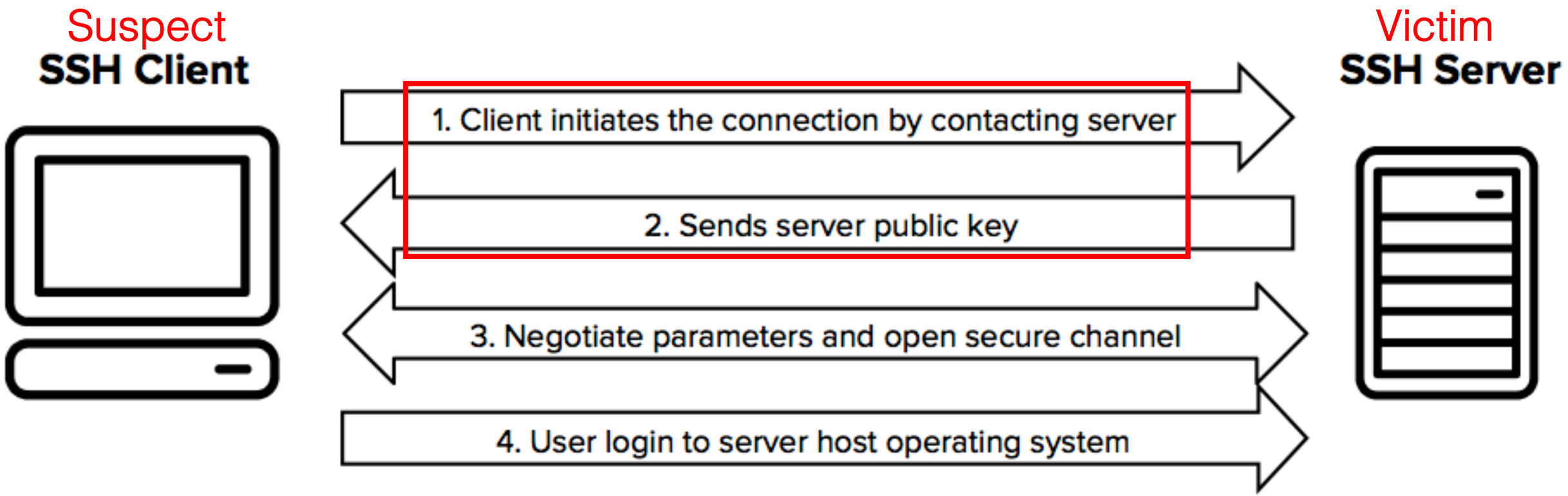
Multi-booting Example



- Boot code: GRUB bootloader (https://help.ubuntu.com/community/Grub2)
- How to boot into non-default partitions?
  - Most bootloaders allow users to manipulate their behaviors when needed.

SSH (secure shell) Connection

- Victim's computer was remotely accessed vis SSH.



https://www.ssh.com/ssh/protocol/

SSH Connection Artifacts

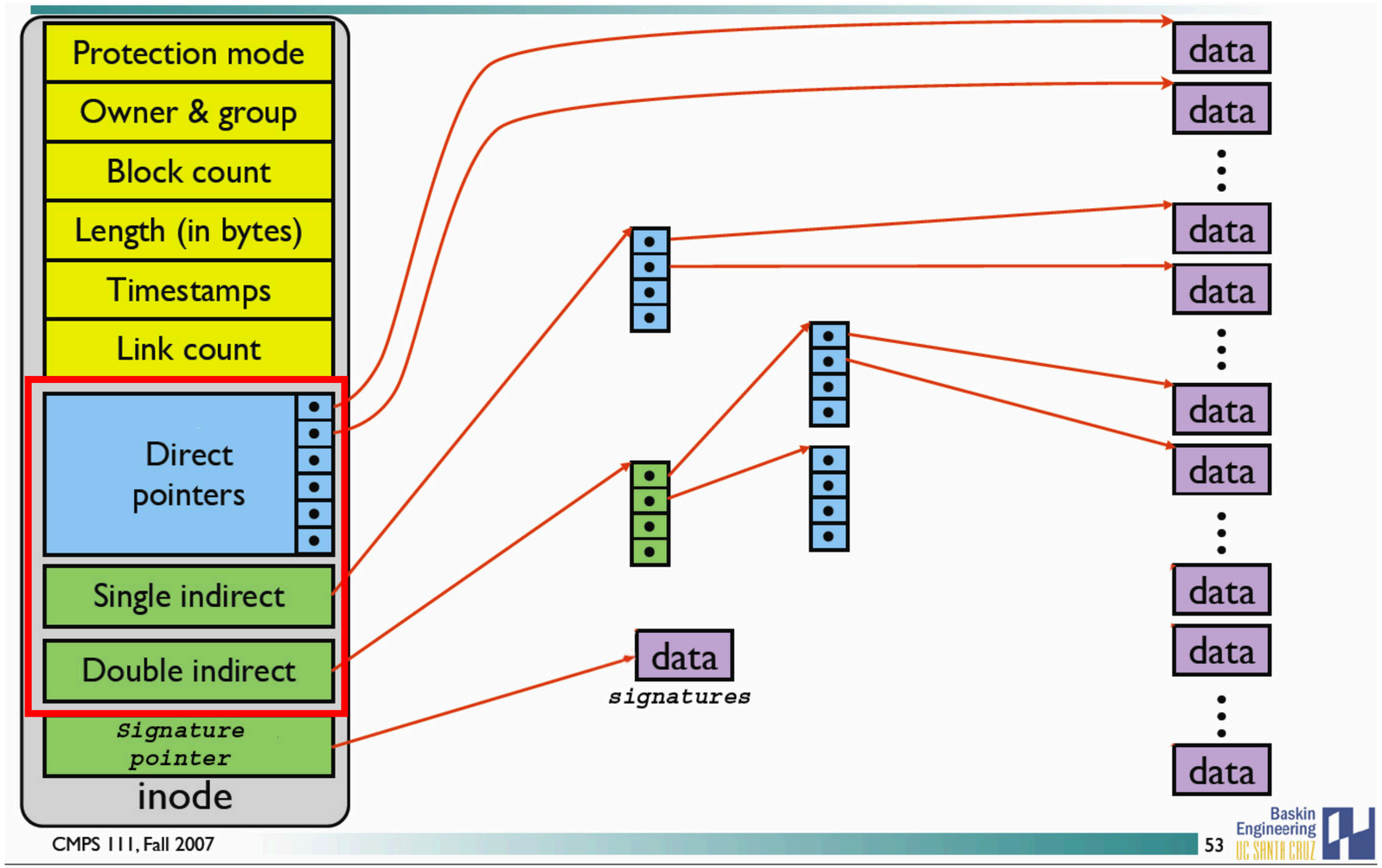
- How do we know if the connections were made from the suspect's computer?
- What IP addresses did the connections come from?
  - What was the suspect's IP address when these connections occurred?
- Which host(s) did the suspect connect to? Which host(s) does the suspect "know" about?
  - Does any of the known host keys match the victim's host key?
  - For more details on host keys, https://www.ssh.com/ssh/host-key

File Recovery (undelete)

- Use file system metadata to re-locate and retrieve the file content.
- Both the file system metadata and content must be intact.
- Download files for a small tutorial:

https://uofi.box.com/v/cs-461-cp2-files

Recovery using metadata

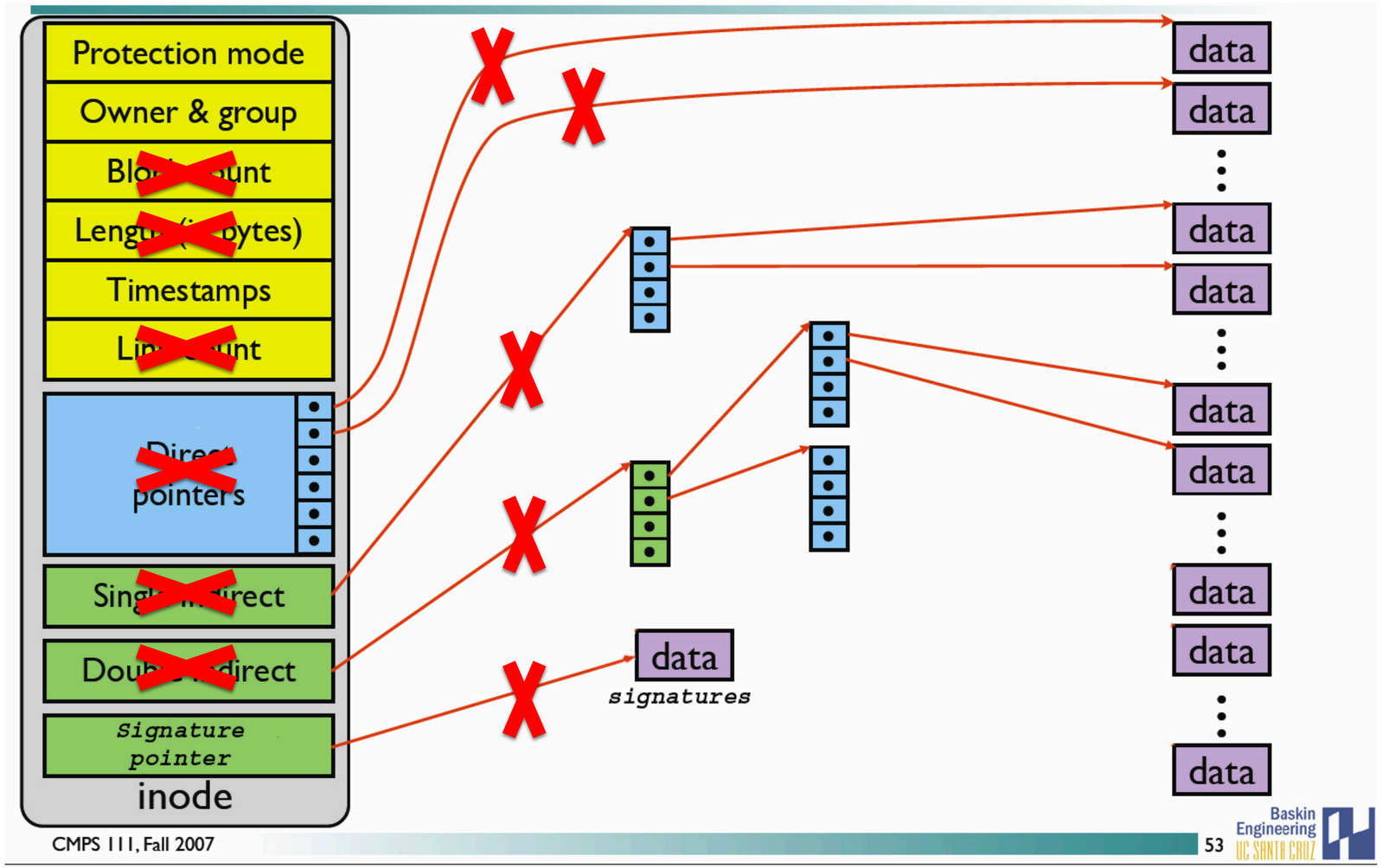


File Recovery on EXT2

[demo]

1. Download `Forensics-cp2_ext2-example.raw`
   - A small ext2 partition image that contains a deleted file.
2. Use `fls` to list the deleted file in the file system.
   ```
   $ fls -rdp Forensics-cp2_ext2-example.raw
   ```
   - The link between the filename and the inode metadata structure is lost.
3. Use `istat` to view the metadata of the orphan file (inode 12).
   ```
   $ istat Forensics-cp2_ext2-example.raw 12
   ```
   - File size, direct/indirect block pointers are all **intact**.
4. Use `icat` to recover the content.
   ```
   $ icat Forensics-cp2_ext2-example.raw 12 > secret-logo_ext2.jpg
   ```
5. Compare the recovered file to the original file `Forensics-cp2_secret-logo.jpg` from the _public repo.

What if metadata is lost



File Recovery on EXT4

[demo]

1. Download `Forensics-cp2_ext4-example.raw`
   - A small ext4 partition image that contains the same JPG file from previous demo.
2. Create a directory to mount the image.
   ```
   $ mkdir ext4-example
   ```
3. Mount the image, check the content, then delete the JPG file.
   ```
   $ sudo mount Forensics-cp2_ext4-example.raw ext4-example
   $ ls -l ext4-example/
   $ rm ext4-example/secret-logo.jpg
   ```
4. Use `fls` to list the deleted file in the file system.
   ```
   $ fls -rdp Forensics-cp2_ext4-example.raw
   ```
5. Use `istat` to view the metadata of the orphan file (inode 12).
   ```
   $ istat Forensics-cp2_ext4-example.raw 12
   ```
   - Note: File size, direct/indirect block pointers are all **cleared**.
6. Try recovering the content using `icat`.
   ```
   $ icat Forensics-cp2_ext4-example.raw 12 > secret-logo_ext4.jpg
   ```
   - Note: Nothing (0 bytes) is recovered, as expected.

File Carving

- Recover file content without file system metadata
- Pulls the raw bytes from the media
- Basic carving technique: header-footer carving

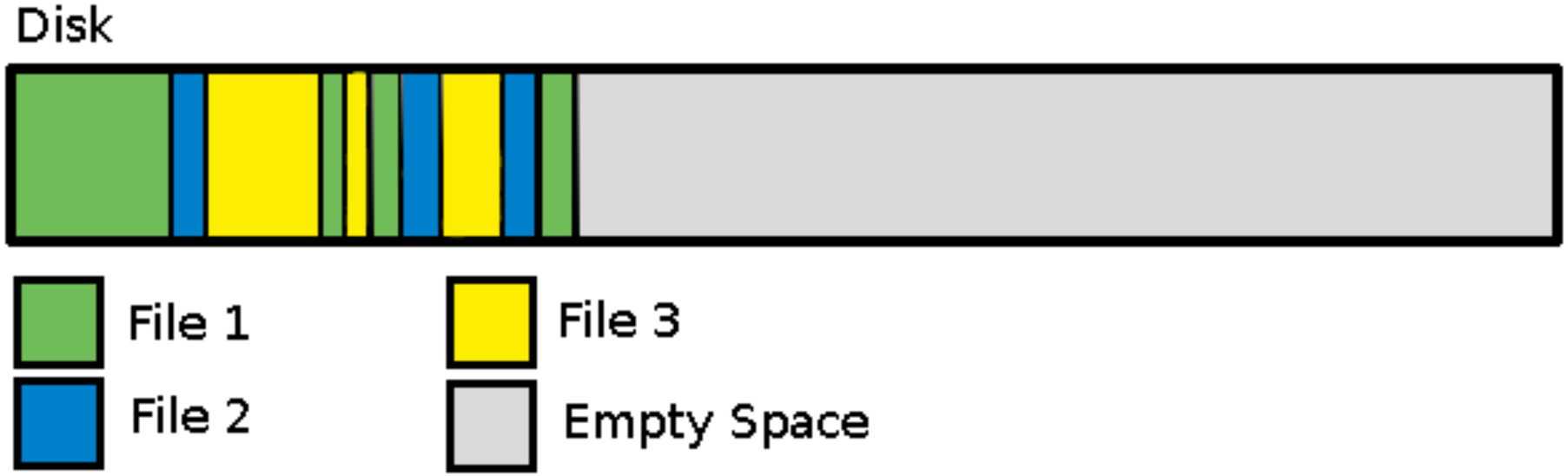
Header-Footer Carving

- Many file types have standard headers and footers stored inside
  - https://www.garykessler.net/library/file_sigs.html
  - Example: A JPEG starts with “Start of image” header `0xFFD8` and ends with “End of image” footer `0xFFD9`
- Carve out everything from header to footer

```
000e0400: ffd8 ffe0 0010 4a46 4946 0001 0100 0048  ......JFIF.....H
000e0410: 0048 0000 ffe1 0074 4578 6966 0000 4d4d  .H.....tExif..MM
                              ...
000f67c0: 0280 0a00 2800 a002 800a 0028 00a0 0280  ....(......(....
000f67d0: 0a00 2800 a002 800a 00ff d900 0000 0000  ..(.............
```

Header-Footer Carving Problems

- Header or footer might be a common string.
  - Many false positives
- The disk could be fragmented



Source: https://www.maketecheasier.com/defragment-linux/

Header-Footer Carving on EXT4

[demo]

1. Use `xxd` to find the location of the deleted JPG file.
   ```
   $ xxd -g 0 -c 10 Forensics-cp2_ext4-example.raw | less
   ```
   - Look for header (`0xffd8`) and footer (`0xffd9`).
2. Determine the byte offsets.
   - Header starts at `0xe0400` (918528 in decimal).
   - Footer ends at `0xf67d4` (1009624 in decimal).
   - The content length is 91105 (end – start + 1).
3. Use `xxd` to extract the data.
   ```
   $ xxd -p -s 918528 -l 91105 Forensics-cp2_ext4-example.raw | xxd -r -p
   > secret-logo_ext4.jpg
   ```
4. Compare the carved file to the original file `Forensics-cp2_secret-logo.jpg` from the box link.

Header-Footer Carving on EXT4 (with `strings`)

[demo]

1. Use `strings` to find the location of the deleted PDF file.
   ```
   $ strings -t d -n 4 Forensics-cp2_ext4-example.raw | grep '%PDF-'
   $ strings -t d -n 4 Forensics-cp2_ext4-example.raw | grep '%%EOF'
   ```
   - Look for header (`%PDF-`) and part of footer (`%%EOF`).
2. Determine the byte offsets.
   - Header starts at `0xf6800` (1009664 in decimal).
   - Footer ends at `0xfd9c6` (1038784 in decimal).
   - The content length is 29121 (end – start + 1).
3. Use `xxd` to extract the data.
   ```
   $ xxd -p -s 1009664 -l 29121 Forensics-cp2_ext4-example.raw | xxd -r -p > cheatsheet.pdf
   ```
4. Compare the carved file to the original file `Forensics-cp2_cheatsheet.pdf` from the box link.

Test your `grep` pattern

- You may need to escape certain characters.
- `grep` may behave differently on different operating systems.
- If you enclose your pattern with double quotes, shell may process the pattern differently.

[demo]

1. Create a text file with some content, including the string that you wish to match.

```
$ cat test.txt # show example content
{\abc*
{\abc1
```

2. Test your `grep` match pattern on the sample text file.

```
$ grep '{\abc*' test.txt   # no match
$ grep '{\\abc*' test.txt  # matches both lines
$ grep '{\\abc\*' test.txt # correct match
$ grep "{\\abc\*" test.txt # no match
```


Checkpoint 2 Overview

- Explore traces left by suspects.
- Relate findings from Checkpoint 1 to the suspects' traces.
- Identify system's default boot behavior.
- Identify application artifacts and metadata.
- Use password cracking tools to access encrypted contents.
- Identify deleted evidence and recover them.

Checkpoint 2 Overview

- 2.1 – try doing live analysis.
  - Let the system boot normally. What happens if you reboot?
  - How to find the file responsible for the boot behavior?
    - Hint: Do you recall any particular behavior or message on the screen while booting? Which file is responsible for executing what you observed?
- 2.2 & 2.3 – similar to Checkpoint 1 questions.
- 2.4 – application artifacts.
  - Hint: See Checkpoint 1 discussion slides.
- 2.5 – Crack a zip file password using a cracking tool.

Checkpoint 2 Overview

- 2.6 – find evidence to prove that the suspect actually connected to the victim's computer.
  - How did the suspect know to connect via SSH? How did the suspect obtain credentials to successfully log in?
  - Hint: Check command history, authentication log, system log, and/or system configuration files. Check the victim's computer as well.
- 2.7 – find application artifacts.
  - Hint: Don't rely on online map services (e.g. Google Maps). Look at system and application-specific metadata.
- 2.8 & 2.9 – recover files using file recovery/carving.
  - Hint: What do you know about the files you are trying to recover? File signatures (header, footer)? Any string that's likely to be present in the content? What application was used to create/modify/access the files?