The Art of Defiling

Defeating Forensic Analysis on Unix File Systems

the grugq

Overview

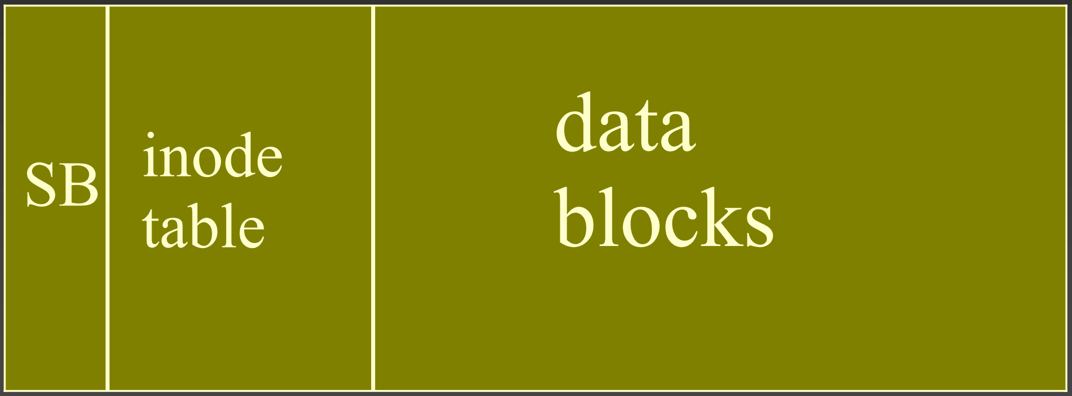
- Introduction
- Unix File Systems
- Forensics
- Anti-Forensics
- Demonstration
- Q & A

Introduction

- Who I am
  - grugq
- What I do
  - Write intrusion prevention software
  - Break forensic tools
- Why anti-forensics?
  - Security is an arms race
  - Trend of increased forensics
  - Trend of increased anti-forensics

Unix File Systems

- Overview of a unix file system
- Super-Blocks
- Data Blocks
- Inodes
- Directory Files



File System Overview

- Two main parts to any file system
- Files
  - Meta data
    - Time stamps, ownership, permissions, etc.
  - Data
    - Disk blocks organised as byte streams
- Meta data files
  - Organise data files for human reference

File System

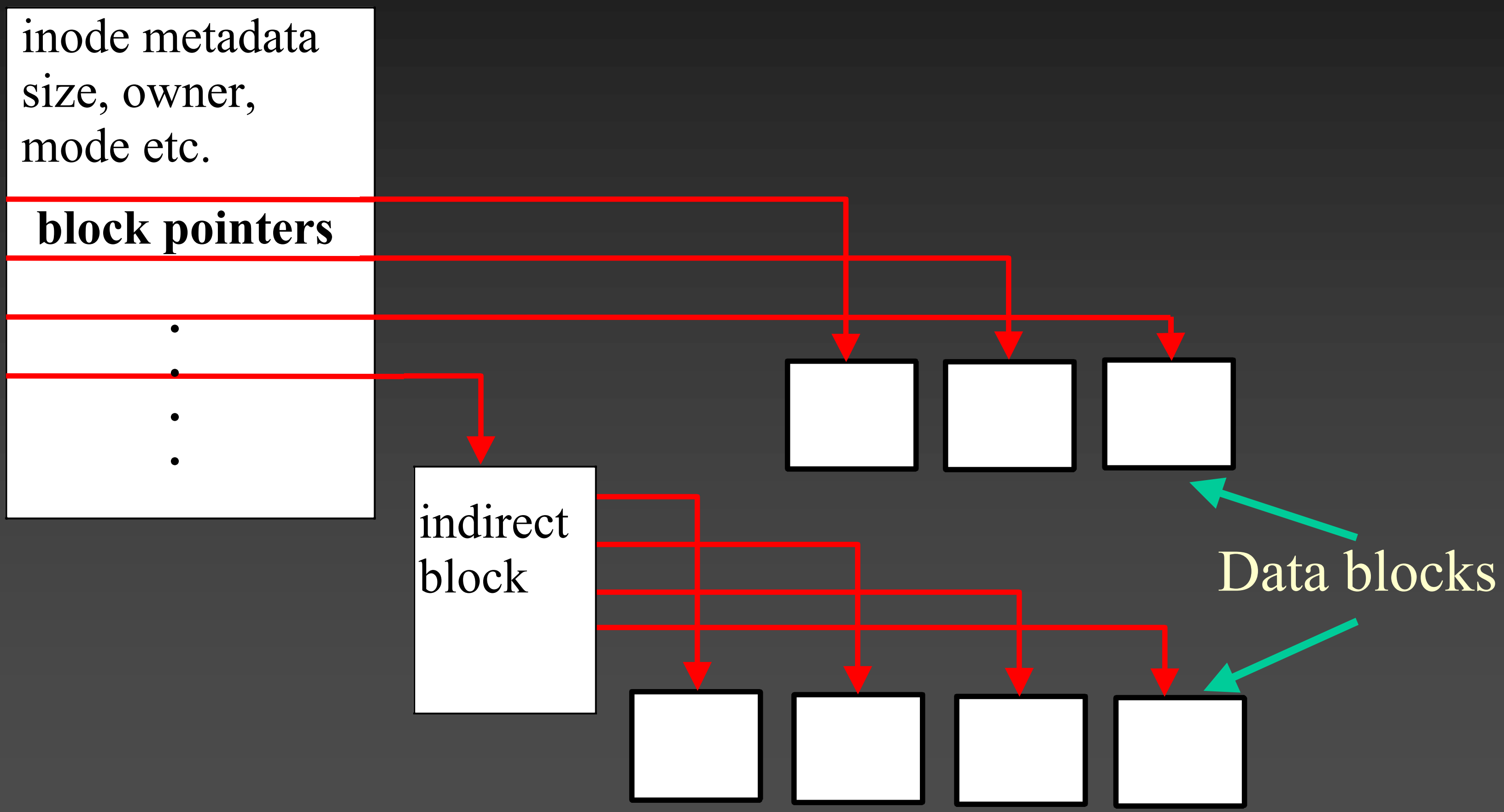
- Superblock
  - Describes the file system
  - Known Location
- Data Block
  - Data blocks store…. data!
  - Block is the lowest atomic component
  - Multiple disk sectors per block

File Systems: inodes

- inodes are files
- Store meta data
  - Time Stamps, Reference Counts, Size
- List of data blocks
  - block pointers

```
struct inode {
        int    uid, gid;
        int    size;
        int    blk_cnt;
        int    links;
        int    block_ptrs[ BLOCK_NUM ];
}
```

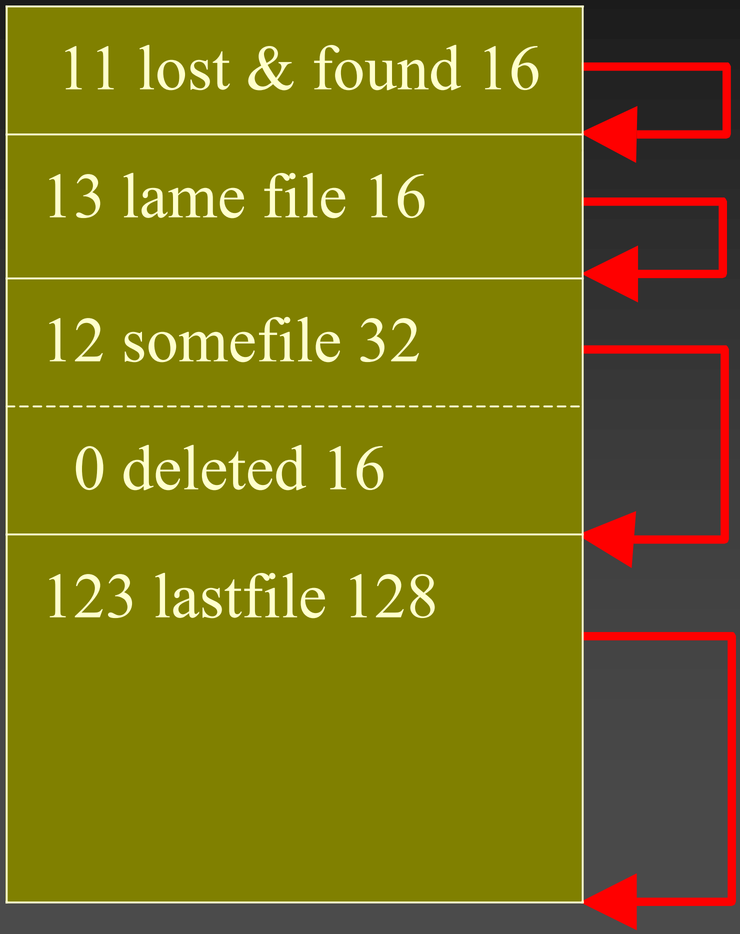
inode structure: graphic



Directory files

- Create the file system directory hierarchy
- Contain structures to map names to inodes

```
struct dirent {
        int         inode;
        short       rec_len;
        short       name_len;
        char        name[];
}
```



Forensics

- Introduction
- Data Recovery
- Data Parsing
- Data Analysis

Introduction

- Forensics defined
- Forensic Food chain..

Data Recovery

- Convert bitstream to file system
  - The Coroner’s Toolkit
    - Recovers deleted files
  - TCT Utils
    - Examine deleted directory entries
- Total file system awareness
  - Read “deleted” data

Data Parsing

- Convert file systems into evidence candidates -- files
- File content requires understanding file formats
  - Email, jpeg, .doc, ELF, etc

Data Analysis

- Keyword searches
- Extract “evidence” from data
  - JPEG files containing illegal images
  - Log files containing access information

Anti-forensics

- *Data is evidence*
- Anti-Forensic Theory
  - Data Destruction
  - Data Hiding
  - Data Contraception

“Attempting to limit the quantity and quality of forensic evidence (since 1999)”

Data Destruction

- Deleted file residue
  - Dirty inodes
  - Directory entries
  - Dirty data blocks
- File System Activity
  - inode time stamps

The Defiler’s Toolkit

- Necrofile
  - Sanitize deleted inodes
- Klismafile
  - Sanitize directory entries

Before and after

Data Hiding

- Requirements
- Theory
- Implementations
- Demos

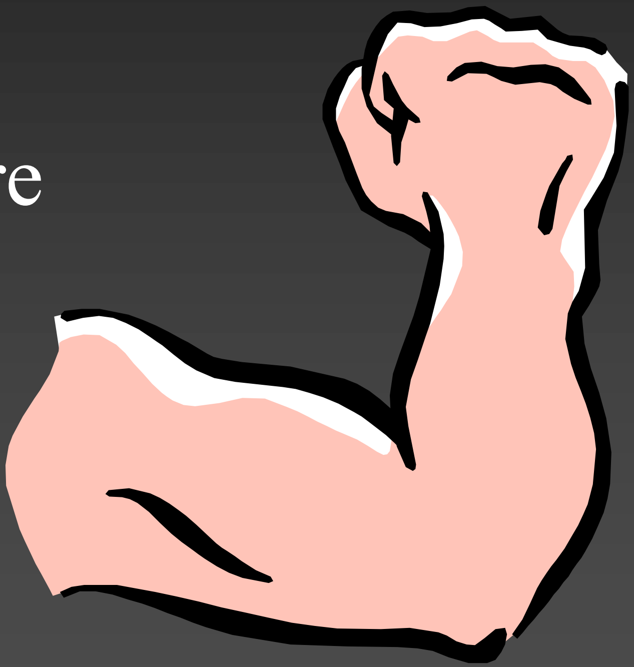
“Aspire to subtlety”

Data Hiding – Requirements

- Covert
- Outside the scope of forensic tools
  - Temporarily – ergo, insecure long term storage
- Reliable
  - Data must not disappear
- Secure
  - Can't be accessed without correct tools
  - Encrypted

Data Hiding Theory

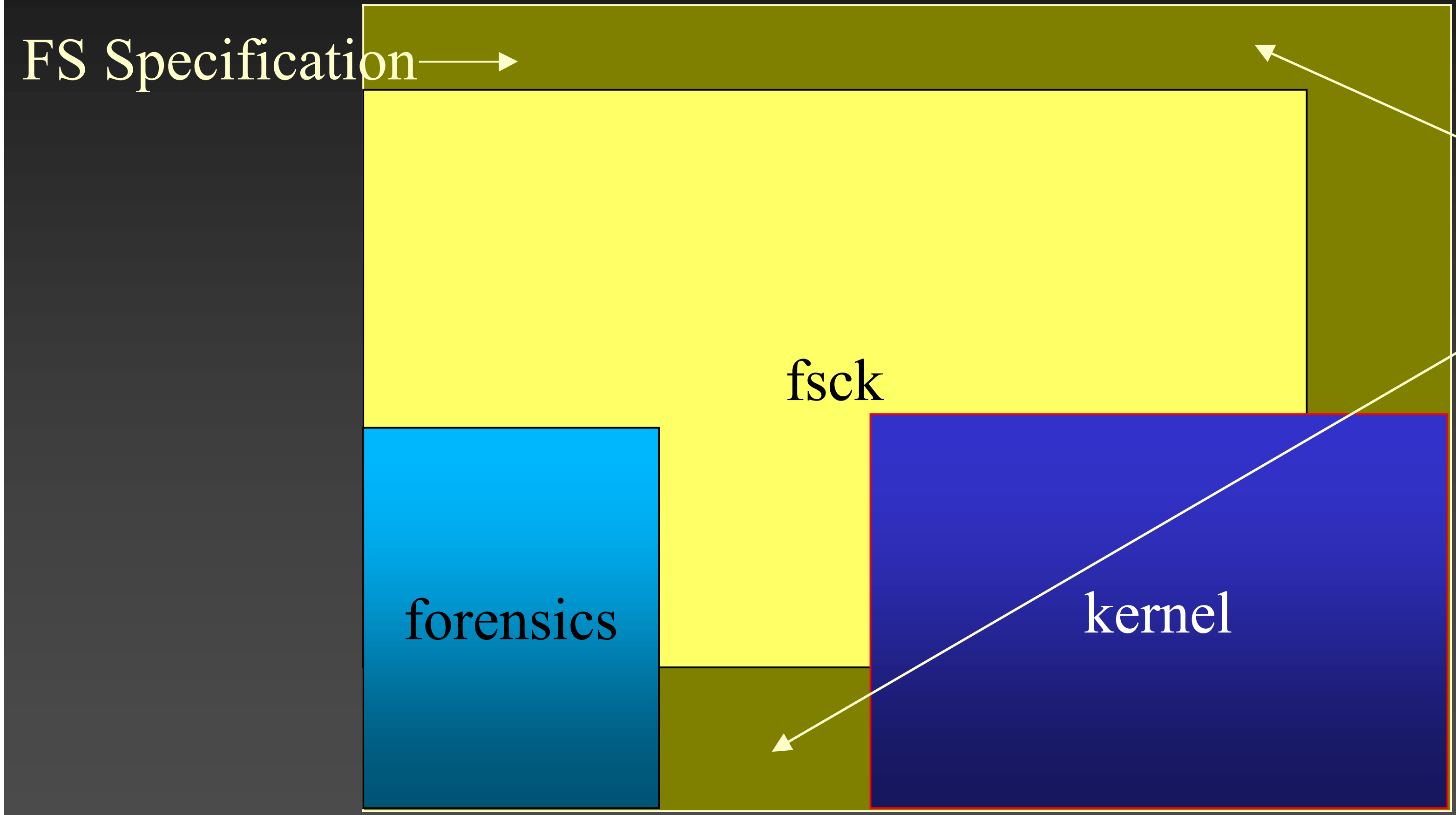
“Ladies and Gentlemen, I'm here to talk about FISTing”



Filesystem Insertion & Subversion Technique

- FISTing is inserting data into places it doesn't belong
- Data storage in meta-data files
  - e.g. Journals, directory files, OLE2 files, etc.
- Modifying meta-data is dangerous!
  - What holes can you FIST?

Holes for FISTing



FISTing implementations

- Rune FS
  - Stores data in the “bad blocks” file
- Waffen FS
  - Stores data in the ext3 journal file
- KY FS
  - Stores data in directory files

Rune FS

- Bad Blocks inode 1, root ('/') inode 2
- Exploits (historically) incorrect ext2 implementation within TCT
- Up to 4GB storage
- TCT pseudo code (old):

```
if (inode < ROOT_INODE || inode > LAST_INO)
    return BAD_INODE;
```

- Just a regular inode file

Waffen FS

- Adds an ext3 journal to an ext2 FS
- Exploits e2fsck (and lame forensic tools)
  - e2fsck supports both ext2 & ext3
    - Has to guess which FS it's looking at
- Usually 32Mb storage (average journal sz)
- e2fsck pseudo code:

```
for (j_ent = journal; ; j_ent += j_ent->size)
        if (IS_VALID(j_ent) == FALSE) /* end of the journal */
                return JOURNAL_OK;
```

- Regular file with a fake journal meta-data

KY FS

- Utilizes null directory entries
- Exploits the kernel, e2fsck & forensic tools
- Storage space limited by disk size

Kill Your File System

KY FS details

- Kernel + fsck pseudo code:

```
for (dp = dir; dp < dir_end; dp += dp->rec_len)
    if (dp->inode == 0) /* is deleted? */
        continue;
```

- Forensic tools pseudo code:

```
if (dp->inode == 0 && dp->namelen > 0)
    /* recover deleted file name */
```

Data Contraception

- Better not to create data than to destroy it
- Prevent data from ever being stored on disk
- Use common Unix utilities to reduce the quality of evidence

"What is the act of not creating?"

Data Contraception: Implem.

- Rexec
  - Remote execution of binaries <u>without</u> creating a file on disk
    - Uses non-exotic utilities to create a remote process image
  - Solves the bootstrapping issue for accessing hidden data stores
    - Reduces effectiveness of honeypots – no binaries to “capture”

Summary

- Summarised Unix File System
- Presented overview of forensics
- Presented a methodology for anti-forensics
- Demonstrated simple mechanisms to defeat digital forensic analysis
- 0wned your file system

Q & A