Small Block Forensics

Atharva Kale | Individual Project

What is Small Block Forensics?

Goal:
To determine *the existence of any content* from the small dataset of known content in the large target drive

Large Target Drive
(say 200 TiB)

Small Dataset of Known Content
(say 32 GiB)

Based on Garfinkel's paper:
https://simson.net/clips/academic/2012.IEEE.SectorHashing.pdf

Brute-force Way

1) Byte-by-byte scan of blocks of the target disk and compare against the known dataset
2) Faster: hash small blocks of target drive and check hash hits in the dataset

Problem: Takes too long!

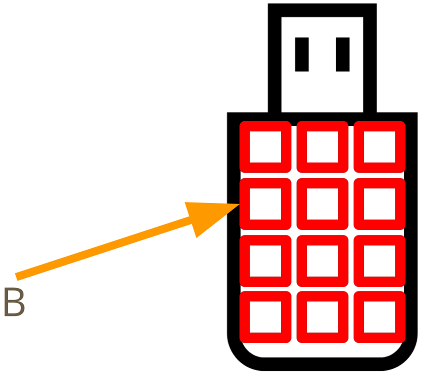
Large Target Drive
(say 200 TiB)

Small Dataset of Known Content
(say 32 GiB)

Small Block Forensics Technique

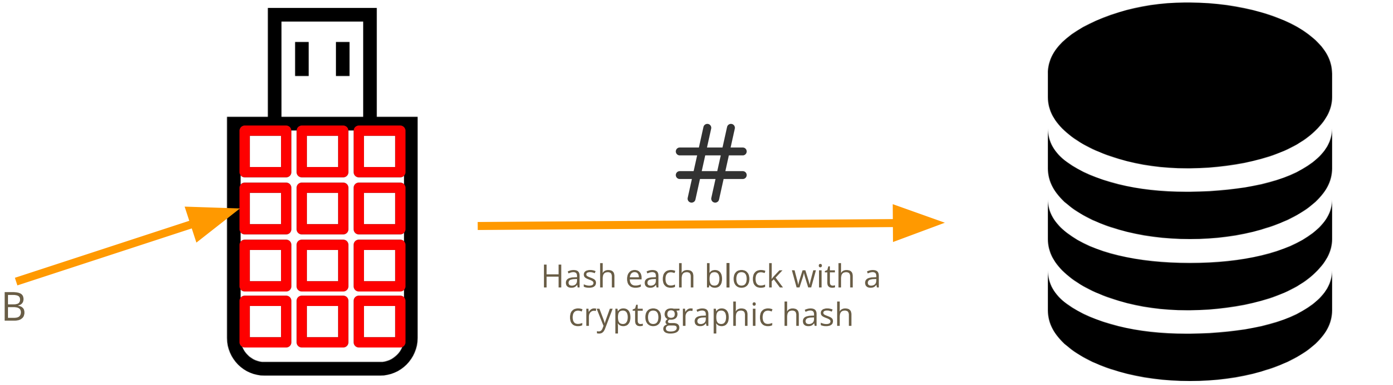
Subdivide known dataset in several blocks

Say block size = 4 KiB



Small Dataset of Known Content
(say 32 GiB)

Small Block Forensics Technique



Subdivide known dataset in several blocks

Say block size = 4 KiB

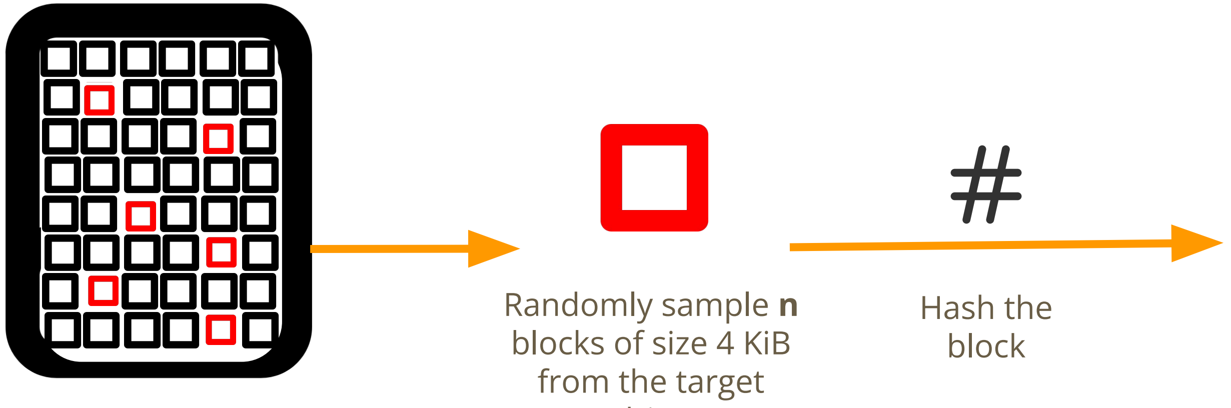
#

Hash each block with a cryptographic hash

Store the hashes in a database

Small Dataset of Known Content
(say 32 GiB)

Small Block Forensics Technique



Randomly sample **n** blocks of size 4 KiB from the target drive

Hash the block

Large Target Drive
(say 200 TiB)

Check for existence of hash in the database

Analysis

With this technique, sampling just ~24,000 blocks (~100 GiB) of the target drive, the probability that one of our checks results in a hit is 99%!

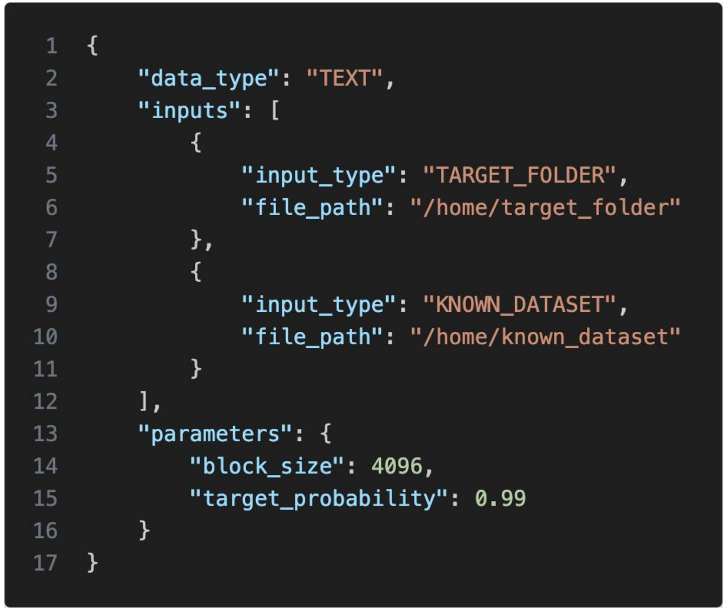
*assuming the target drive contains all 32 GiB of known content

Large Target Drive
(say 200 TiB)

Small Dataset of Known Content
(say 32 GiB)

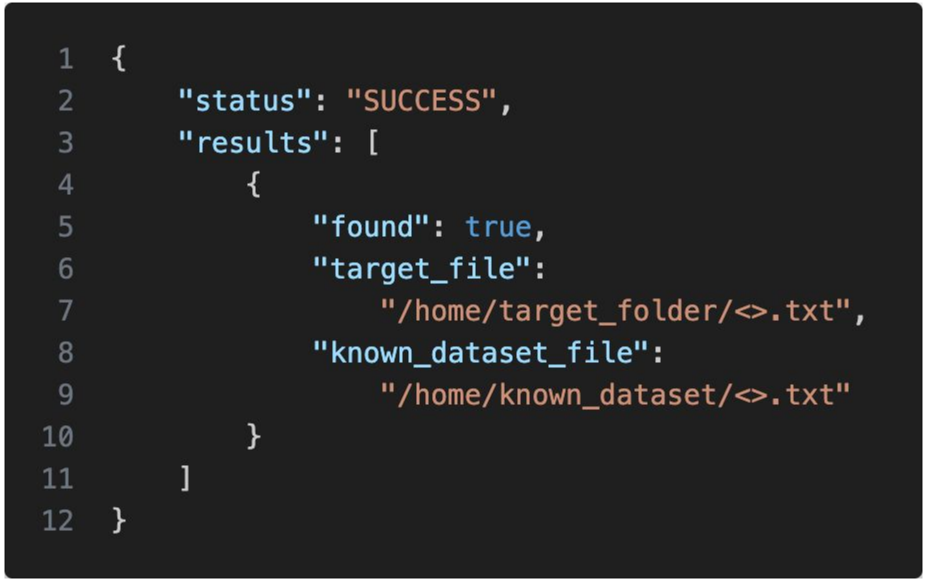
Based on Garfinkel's paper:
https://simson.net/clips/academic/2012.IEEE.SectorHashing.pdf

Individual Project - An Approximation of SBF



```json
{
    "data_type": "TEXT",
    "inputs": [
        {
            "input_type": "TARGET_FOLDER",
            "file_path": "/home/target_folder"
        },
        {
            "input_type": "KNOWN_DATASET",
            "file_path": "/home/known_dataset"
        }
    ],
    "parameters": {
        "block_size": 4096,
        "target_probability": 0.99
    }
}
```

Sample Request*



```json
{
    "status": "SUCCESS",
    "results": [
        {
            "found": true,
            "target_file":
                "/home/target_folder/<>.txt",
            "known_dataset_file":
                "/home/known_dataset/<>.txt"
        }
    ]
}
```

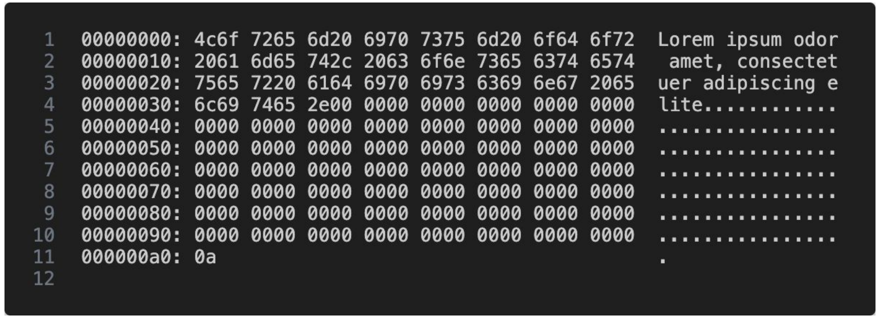
Sample Response

* as of writing of this presentation

Implementation

Considerations:

1) Padding files with `0x00` that are not divisible by `block_size`
2) Use Python's md5 hash function
3) Use sqlite3 to store hashes



```
00000000: 4c6f 7265 6d20 6970 7375 6d20 6f64 6f72  Lorem ipsum odor
00000010: 2061 6d65 742c 2063 6f6e 7365 6374 6574   amet, consectet
00000020: 7565 7220 6164 6970 6973 6369 6e67 2065  uer adipiscing e
00000030: 6c69 7465 2e00 0000 0000 0000 0000 0000  lite............
00000040: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000050: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000060: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000070: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000080: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000090: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000000a0: 0a                                       .

```