FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

TECHNISCHE HOCHSCHULE NÜRNBERG
GEORG SIMON OHM

Selective deletion of non-relevant Data

Christian Zoubek, Konstantin Sack

23rd March 2017

Outline

- Introduction
- Selective deletion
- Evaluation
- Conclusion

Motivation

- In law enforcement investigations search and seizure of digital devices is a standard procedure
  - Bitwise copies (imaging)
  - even if reason of investigation is of non-electronic nature

- Problems arising
  - How to handle mass of data (only slightly in scope of this paper)
    ➔ selective imaging
  - More specialized defense counsels
    ➔ selective imaging or ***selective deletion***

Legal considerations

- Privacy laws limit access and usage of information
- ‚Elfes'-decision made by the German Federal Constitutional Court (1957)
  - „One's data is part of a human beings's inviolable dignity and freedom"
  - Law enforcement is forced to spare data blocks irrelevant to cases
  - If not done while imaging
    - Deletion as soon as possible
    - Documentation of obtainment and deletion mandatory
- Sparing blocks while imaging hardly applicable in practise
  - Selective deletion one possible solution
  - Yet, not actively pursued
    - Deletion of data modify images
    - Applicability in court may be endangered

Example: Blogserver hosting hundreds of blogs

- Some blogs involved in illegal activities, most are not
  - Search warrant for serverhost
  - Seizure and imaging of whole server
- A lot of case-irrelevant data, especially data of innocent bystanders
- Question arises: What to do with such data?
  - E.g. in Germany: Delete afterwards
  - How to delete afterwards securely and in a forensically sound way?

Selective deletion of files

- Common forensics software do not allow modification directly in images/ disks
- Deletion by instructions implemented in OS not sufficient
  - Only index entry is modified
  - Content and meta data unaffected
- Deletion by zeroing content (wiping) also not sufficient
  - Meta data still yield enough information about users

Example extended

- One suspect also used the server to store private data, which is not shared amongst all users
- For instance, pictures made in holidays, saved in directories with unique names
  - In Germany: If not case-relevant data and belongs to protected data in regard of privacy laws, deletion of such data is also mandatory
  - Two problems arise
    - How to classify which data is case-relevant and which not? (not in scope of this paper)
    - How to delete affected data without causing damage to residual data and file systems

Forensically sound selective deletion

- With respect to Law
  - Private non-relevant data to be deleted
  - Integrity of residual data

- Technical point of view
  - Deletion of content straightforward: zero or random data

- ***Our requirement / demand for forensically sound selective deletion***
  - Meta data on file system level which still yields enough information about a user‘s activity and/or private life should be *deleted*

Outline

- Introduction
- Selective deletion
- Evaluation
- Conclusion

Our selective deletion tool

- Intention to investigate whether a secure selective deletion is technically achievable
  - Realized as a plugin for the Digital Forensics Framework (unfortunately ArxSys seems to be closed)
  - Plugin is bounded to usage of Microsoft‘s NTFS
- Some more functionality included
  - Detection of duplicated files which are not necessarily flagged/found by an investigator
  - Basic partition table parser (only finds NTFS partitions)
  - Detection of carved files, which are not managed by a file system
  - Hard link detection, if more than one file is linked to the same content
  - Calculation of hash trees
  - and more

Deletion-module

- Modification of corresponding MFT entry/entries
  - Toggle ‚in use'-flag
  - Overwrite all attributes with zeros, care for Fixup-values
- B-tree update
  - Leaf-level
    - Search filename and wipe affected bytes
    - Indent data right of it
  - Node-level
    - Find suitable replacing file
      - Smallest element in right child node
      - or, greatest element in left child node
    - Replace filename you want to delete (careful of filename lengths)
    - Delete replacing filename in leaf

Hashcalculator-module (Integrity)

- Before deletion
  - Calculate hash tree of original image
  - Find, classify and mark every sector affected by a file in a bitmap
  - Prepare modification of sectors in RAM
- Deletion
  - Write path, affected sectors, type of alterations in a separated file
  - Write prepared modifications on the image/disk
- After deletion
  - Calculate hash tree
  - Differences in hash trees should yield the same modified sectors as can be found in the logfiles

Outline

- Introduction
- Selective deletion
- Evaluation
- Conclusion

Experimental setup

I. Scenario (seven various test cases)

- some directories and/or files were to be deleted
- Different allocations of data
  - Resident vs. Non-resident data
- Reformatted devices, some data transparent to the new format
- Test case with a bootable Windows

II. Scenario (functionality comparison against an existing implementation of professional software)

- USB-device with many different directories, many cloned files across directories
- Deletion of one whole folder
  - Search for duplicates across partition
  - Comparison of results

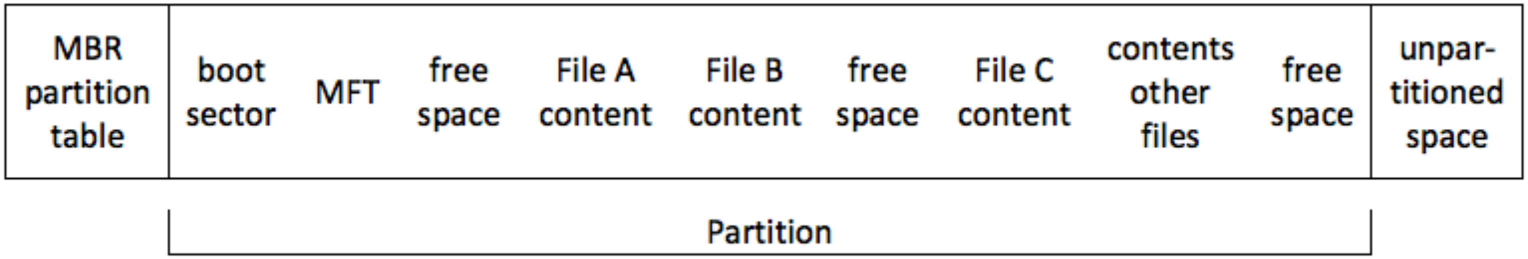
Evaluation (first scenario)

- seven test cases worked without major problems
- Comparison of logfiles, hash trees and resulting image verified correct behaviour
  - Data content erased in a whole
  - B-trees rearranged properly
  - Images/disks were mountable
    - Directories were readable without any warning/error
    - No traces of deleted files
  - Meta data could not be found anymore
- One exception
  - Deletion of a user‘s home in Windows 7
  - Windows could see a broken home, warning popped up
  - Yet, only username was found, anything else was irrecoverable

Evaluation (second scenario)

- Comparison against pro software
  - Both tools could find all duplicates
  - Pro software deletes files by sparing only data content
    - Meta data still usable
    - Even full filenames were found
  - Pro software cleanses image by creating a new image
    - Input image is not modified in any way
    - Marked entries are deleted by only skipping a file‘s content on disk when copying the image
    - Files can still be found and accessed, yet no content is readable
- Our tool operates directly on the image
- Further verification of correctness with FTK Imager

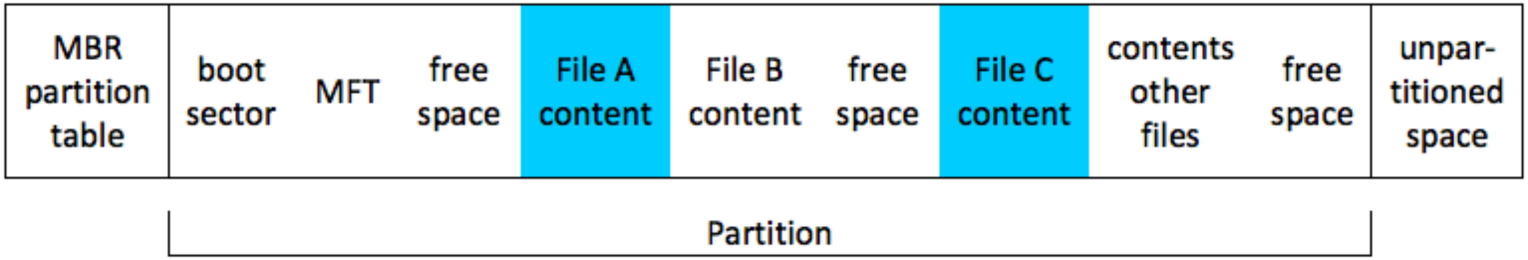
Original Disk



MBR partition table	boot sector	MFT	free space	File A content	File B content	free space	File C content	contents other files	free space	unpar-titioned space

Partition

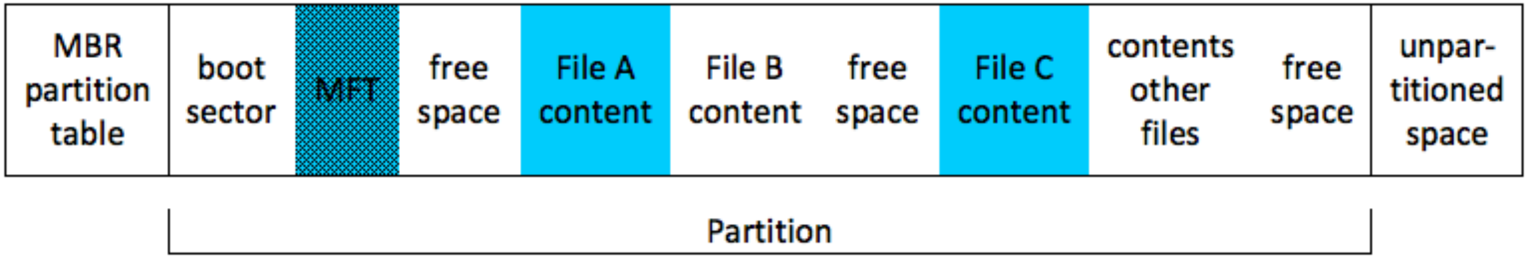
Cleansed Image of the Disk: blue = wiped, excluded areas (***X-Ways Forensics***)



MBR partition table	boot sector	MFT	free space	File A content	File B content	free space	File C content	contents other files	free space	unpar-titioned space

Partition

selective deletion image of the Disk: blue = wiped, excluded areas; checkered = modified MFT



MBR partition table	boot sector	MFT	free space	File A content	File B content	free space	File C content	contents other files	free space	unpar-titioned space

Partition

Outline

- Introduction
- Selective deletion
- Evaluation
- Conclusion

Conclusion

- Practical approach of prototypical selective deletion tool
- Legal requirements are fulfilled in case of non-relevant data
  - Content and meta data is erased/wiped
  - Residual data stays untouched
  - File system data structures are not damaged, hence disks/images are still usable without professional software
  - Calculation and comparison of hash trees for verification of data integrity
  - Continuous logging of every single step

Problems?

  - Logging while deleting could also reveal information about bystanders

Thank you for your attention!

Questions?

(christian.zoubek@th-nuernberg.de)

Tool can be found here:

https://www1.informatik.uni-erlangen.de/content/selective_deletion