Studying copy-on-read and copy-on-write techniques on block device level to aid in large environment forensics

E. van den Haak

System and Network Engineering
University of Amsterdam

Master Thesis, July 2014

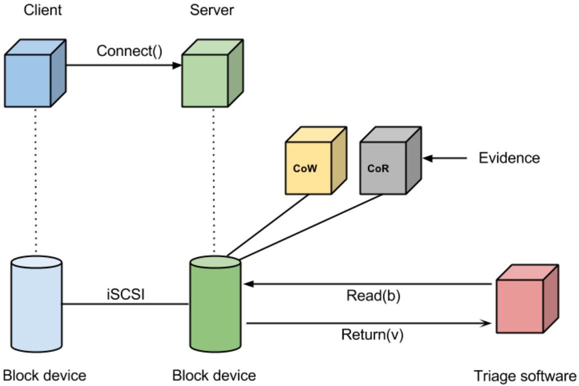
University of Amsterdam

Background

Forensics on cloud solutions and large environments

- Sheer volume of data
- (Remote) Acquisition is very hard
  - Making a copy of all data is impossible
  - Making data available remotely is a long procedure

Concept



University of Amsterdam

Research

Focus on server block device level

- Copy only relevant data to local storage
  - Copy-on-Read
- Enable live forensics without interfering with original block device
  - Copy-on-Write

Important aspects

- Data integrity
- Reproducible
- Storable

UNIVERSITY OF AMSTERDAM

Research

What is a good way to mount block devices read only and store read and changed data in separate sparse files?

- *What methods exist that allow copy-on-write and copy-on-read on block device level?*
- *Can these methods be effectively used to do remote data acquisition while storing read- and changed data locally?*
- *If necessary, how can an existing method be modified in order to meet the requirements of this research?*

UNIVERSITY OF AMSTERDAM

Related Research

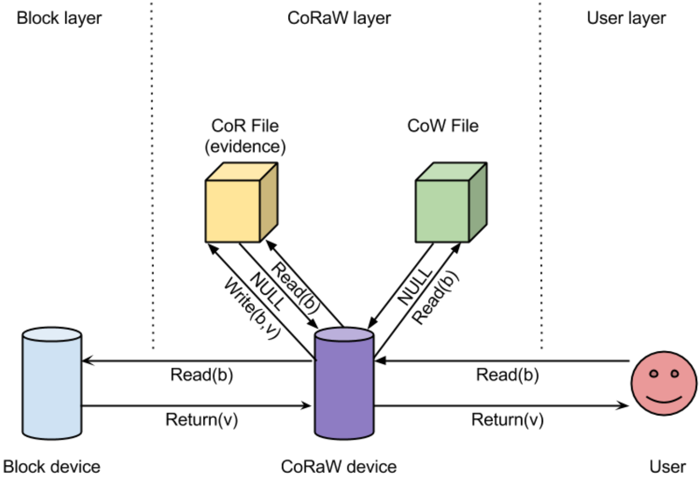
- Forensic mount tool Xmount[1]
- NIST Cloud Computing Forensic Science Challenges[2]

Existing methods

Methods that either support copy-on-read or copy-on-write

- Xmount
- Fusecow
- Bcache

Ideal situation



UNIVERSITY OF AMSTERDAM

Proof of concept[1]

Both Xmount and Fusecow

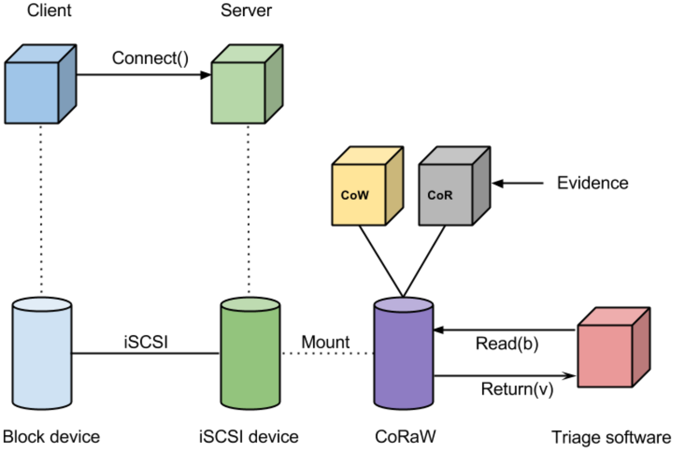
- Open source
- C
- GPL

Scope

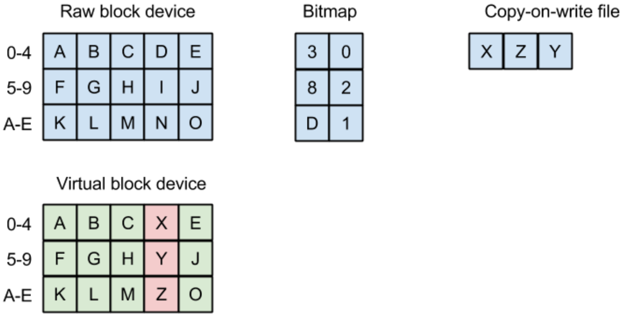
- Copy-on-read file
- Read only feature

[1] Sources on github[3]

Detailed concept



Copy-on-write implementation (existing)



Raw block device

0-4	A	B	C	D	E
5-9	F	G	H	I	J
A-E	K	L	M	N	O

Bitmap

3	0
8	2
D	1

Copy-on-write file

X	Z	Y

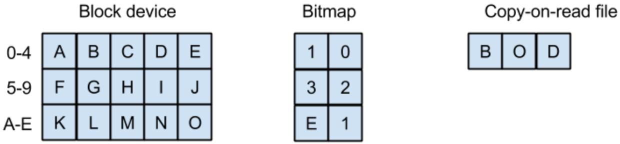
Virtual block device

0-4	A	B	C	X	E
5-9	F	G	H	Y	J
A-E	K	L	M	Z	O

write(3,X); write(D,Z); write(8,Y)

- Fusecow has two separate files
- Xmount puts bitmap into header of CoW file

Copy-on-read implementation



Block device

0-4	A	B	C	D	E
5-9	F	G	H	I	J
A-E	K	L	M	N	O

Bitmap

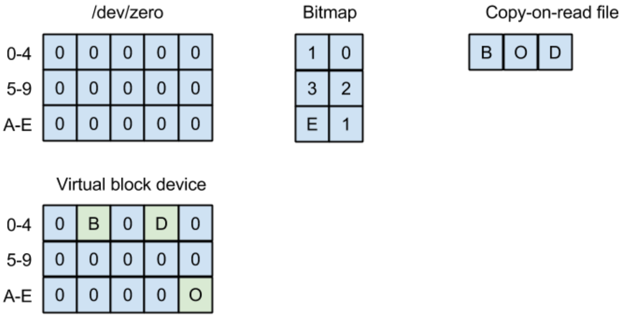
1	0
3	2
E	1

Copy-on-read file

B	O	D

read(1); read(E); read(3)

Copy-on-read implementation remount



/dev/zero

0-4	0	0	0	0	0
5-9	0	0	0	0	0
A-E	0	0	0	0	0

Bitmap

1	0
3	2
E	1

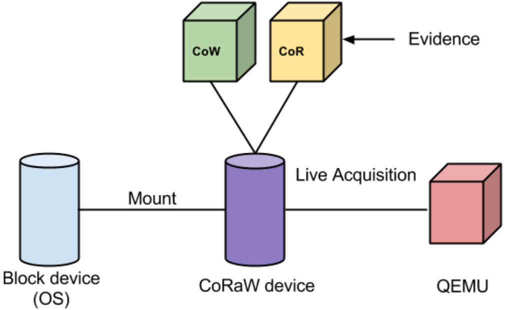
Copy-on-read file

B	O	D

Virtual block device

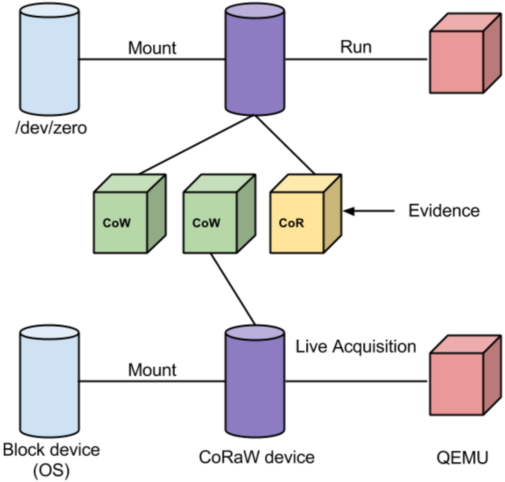
0-4	0	B	0	D	0
5-9	0	0	0	0	0
A-E	0	0	0	0	O

Test setup



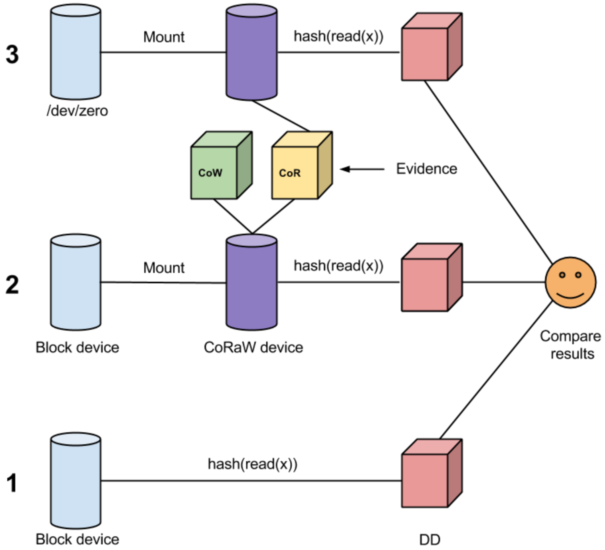
UNIVERSITY OF AMSTERDAM

Test setup



University of Amsterdam

Second test setup



University of Amsterdam

Results

QEMU

- Fusecoraw works flawless

DD

Results

QEMU

- Fusecoraw works flawless
- Xmount has trouble remounting as it performs lots of tests

DD

Results

QEMU

- Fusecoraw works flawless
- Xmount has trouble remounting as it performs lots of tests
  - For now Read only or Copy-on-Read file as Copy-on-Write file

DD

Results

QEMU

- Fusecoraw works flawless
- Xmount has trouble remounting as it performs lots of tests
  - For now Read only or Copy-on-Read file as Copy-on-Write file
  - Requires future work

DD

Results

QEMU

- Fusecoraw works flawless
- Xmount has trouble remounting as it performs lots of tests
  - For now Read only or Copy-on-Read file as Copy-on-Write file
  - Requires future work

DD

- Both techniques work as expected, hashes match.

University of Amsterdam

Conclusion

- Both proof-of-concepts perform a good job
    - Remounting writable works only with Fusecoraw
    - No issue for current concept
- Read data is persistent
- Fusecoraw recommended if writable remounting is desired
- Xmount recommended if not

Future Research

- Fusecoraw
- Xmount
- Integrate in concept

Questions

?

UNIVERSITY OF AMSTERDAM

References

Gillen Daniel.
xmount, 2008.
https://www.pinguin.lu/index.php.

NIST Cloud Computing Forensic Science Working Group.
Nist cloud computing forensic science challenge (draft), 2014.
http://csrc.nist.gov/publications/drafts/nistir-8006/draft_nistir_8006.pdf.

Eric van den Haak.
Evdh's git repository, 2014.
https://github.com/evdh-nl.