AFF4: The new standard in forensic imaging

and why you should care

Dr. Bradley Schatz

Director, Schatz Forensic

v1.1 – OSDFCon 2016

About me

- Bradley Schatz
  - PhD, Digital Forensics (2007) ; BSc, Computer Science
- Schatz Forensic / Evimetry (2009-)
  - Practitioner, R&D, tool vendor
- Research affiliations
  - Journal of Digital Investigation (Editorial Board)
  - DFRWS Conference, Chair, Technical Program Committee (2016)
- Practical contributions
  - Volatility Memory Forensics Framework (Vista & Windows 7 support) (2010)
  - Autopsy (index.dat support)
- Queensland University of Technology
  - Adjunct associate professor, doctoral supervision

Agenda

- Why should I care about forensic formats?
- What's wrong with current forensic formats?
- What is AFF4?
- AFF4 Status Update

Why should I care about forensic formats?

Limitations in forensic formats have profound effects on practice

- The image as a linear bitstream
  - Triage: reliance on logical imaging and acceptance of loss of potentially relevant data
- The usage of heavyweight compression
  - Significant delays due to CPU consumption
- The usage of no compression
  - Significant delays due to hashing and copying sparse data
- Inextensible metadata storage
  - Tool interoperability an ongoing problem

AFF4 supports storing multiple streams per container

Pmem aff4acquire = physical memory + mapped files

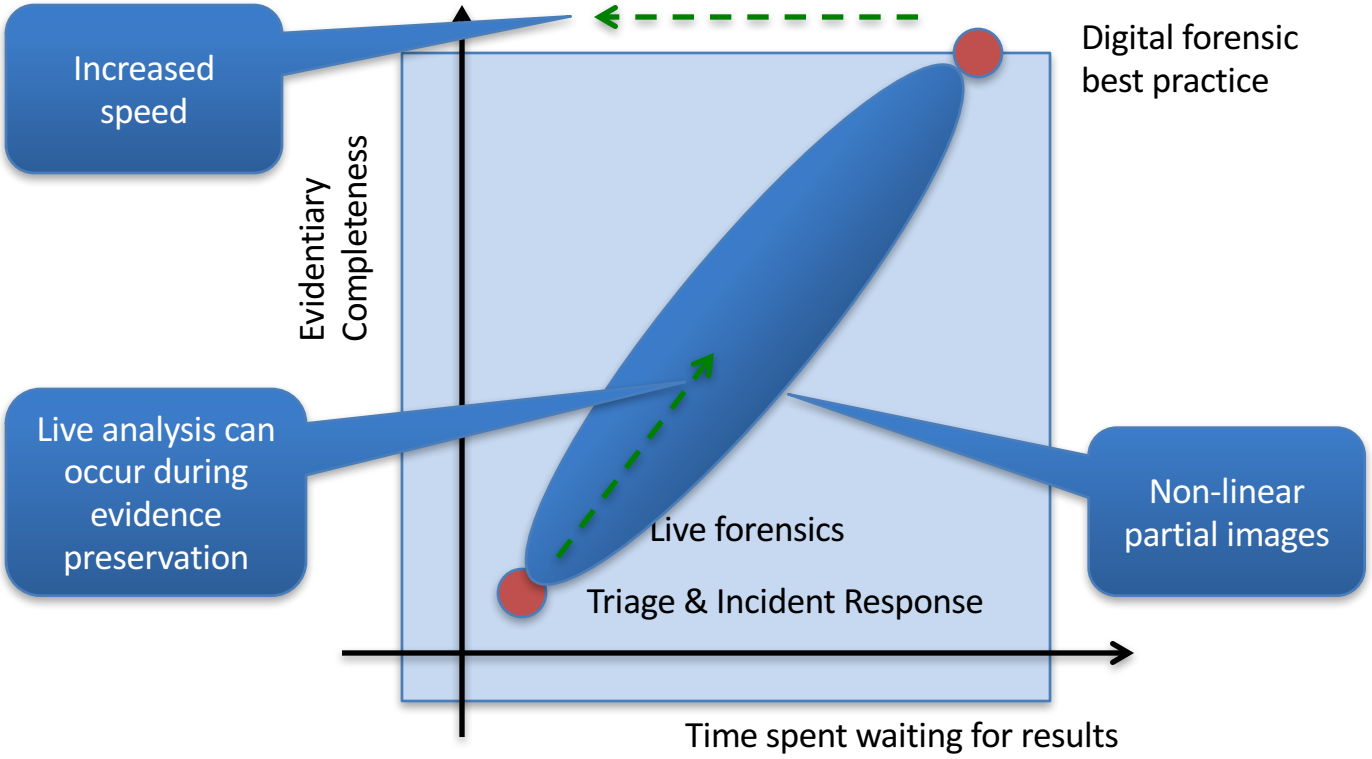
```
0x0000738a0000      0x21000 File Mapping          \Windows\System32\msvcr100.dll (P)
0x0000738c1000       0x3000 Valid                 PhysAS @ 0x176ed000
0x0000738c4000              apping                \Windows\System32\msvcr100.dll @ 0x23400 (P)
0x0000738c5000                                    PhysAS @ 0x34d2e000
0x0000738c6000              apping                \Windows\System32\msvcr100.dll @ 0x25400 (P)
0x0000738cc000                                    PhysAS @ 0x1ac37000
0x0000738cf000              apping                \Windows\System32\msvcr100.dll @ 0x2e400 (P)
0x0000738d0000              000 Valid             PhysAS @ 0xe9e6000
0x0000738d1000              000 File Mapping      \Windows\System32\msvcr100.dll @ 0x30400 (P)
0x0000738dc000              000 Valid             PhysAS @ 0x3d72000
0x0000738de000      0x15000 File Mapping          \Windows\System32\msvcr100.dll @ 0x3d400 (P)
0x0000738f3000       0x1000 Valid                 PhysAS @ 0x33eff000
0x0000738f4000      0x18000 File Mapping          \Windows\System32\msvcr100.dll @ 0x53400 (P)
0x00007390c000       0x2000 Valid                 PhysAS @ 0x25bf9000
0x00007390e000       0x2000 File Mapping          \Windows\System32\msvcr100.dll @ 0x0  (P)
0x000073910000       0x1000 Valid                 PhysAS @ 0x241000
0x000073911000       0x3000 File Mapping          \Windows\System32\msvcr100.dll @ 0x70400 (P)
0x000073914000       0x1000 Pagefile              PF 0 @ 0x323f9000
0x000073915000      0x1c000 File Mapping          \Windows\System32\msvcr100.dll @ 0x74400 (P)
0x000073931000       0x4000 Valid                 PhysAS @ 0x365f000
0x000073935000      0x1d000 File Mapping          \Windows\System32\msvcr100.dll @ 0x93c00 (P)
0x000073952000       0x2000 Valid                 PhysAS @ 0x20a37000
0x000073954000       0x3000 File Mapping          \Windows\System32\msvcr100.dll @ 0xb2800 (P)
0x000073957000       0x3000 Valid                 PhysAS @ 0x2b7d5000
0x00007395a000      0x18000 File Mapping          \Windows\System32\msvcr100.dll @ 0xb5a00 (P)
0x000077880000      0x21000 Valid                 PhysAS @ 0x1536000 (P)
0x0000778a1000       0x2000 Valid                 PhysAS @ 0x222ed000
0x0000778a3000       0x1000 Valid                 PhysAS @ 0x3fb6d000 (P)
0x0000778a4000       0x4000 Transition            PhysAS @ 0xde6e000 (P)
```

pages

Virtual address

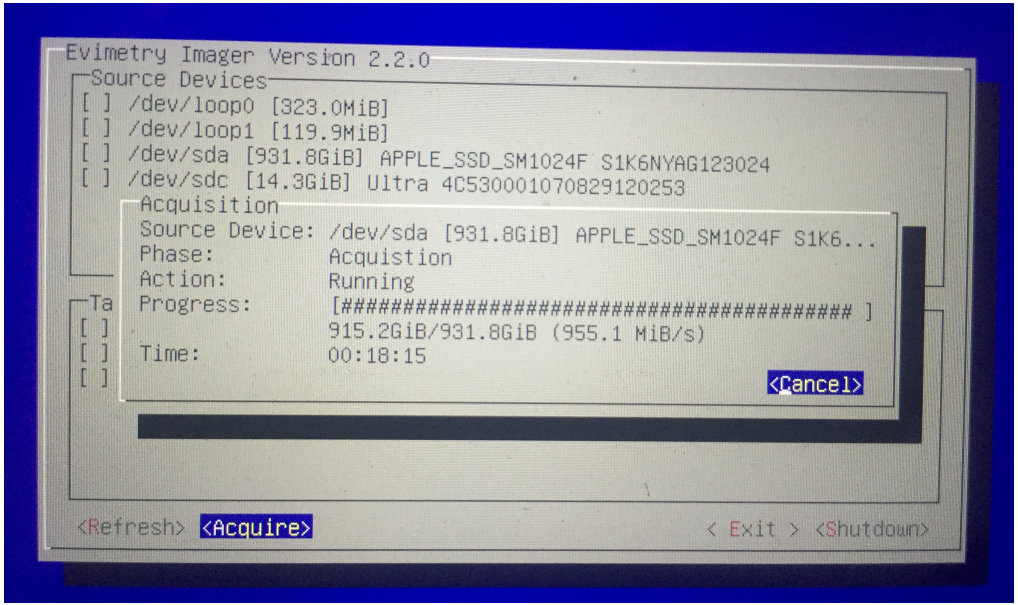
Acquired mapped files & page files

The AFF4 forensic format enables faster and new approaches to acquisition



AFF4 shifts acquisition throughput to being CPU limited

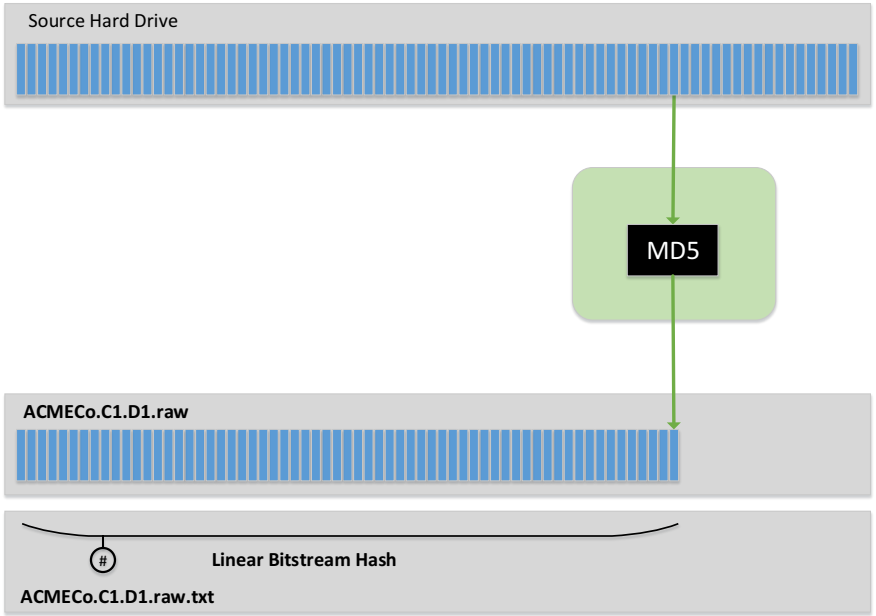
1TB acquired in 20 minutes (~50 GB/min)



1TB NVMe (Core i7-4578U, 2 Cores) Macbook Pro A1502 (Evimetry 2.2.0a)

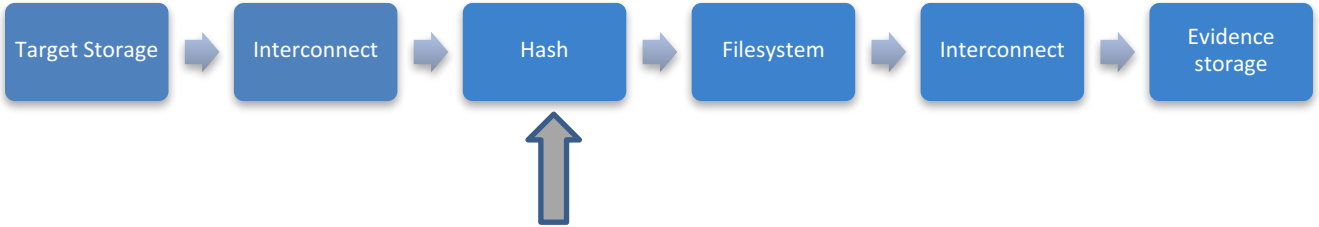
What's wrong with current forensic formats?

Forensic Imaging v1.0: RAW



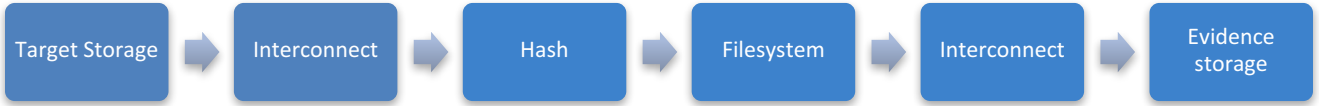
- Good
  - Universal tool support
  - 1:1 mapping
  - Easy to implement
- Bad
  - No standardised metadata storage
  - Copying sparse regions (zero-filled) is a waste of time
  - **Linear bitstream hash is a bottleneck at high speeds**

The linear bitstream hash is a bottleneck at high speeds



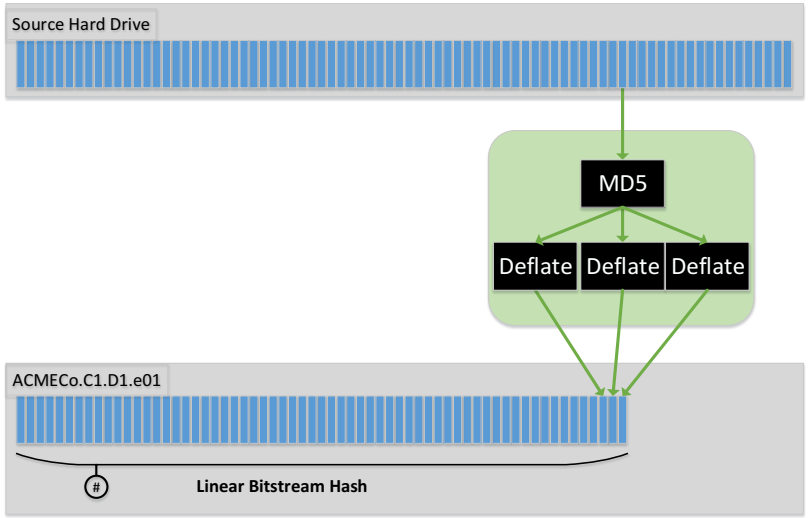
Algorithm	Average Throughput MB/s
SHA1	619.23
MD5	745.65
Blake2b	601.87

Linear bitstream hashing is a bottleneck with current generation storage



Target Storage	Sustained Read
1TB Seagate 3.5” 7200rpm SATA	100 MB/s
Current generation 3.5” 7200rpm SATA	200 MB/s
Intel 730 SSD	550 MB/s
Macbook Pro 1TB	**~1 GB/s**
RAID 15000rpm SAS	**> 1 GB/s**
Samsung 850 NVMe	**1.5 – 2.5 GB/s**

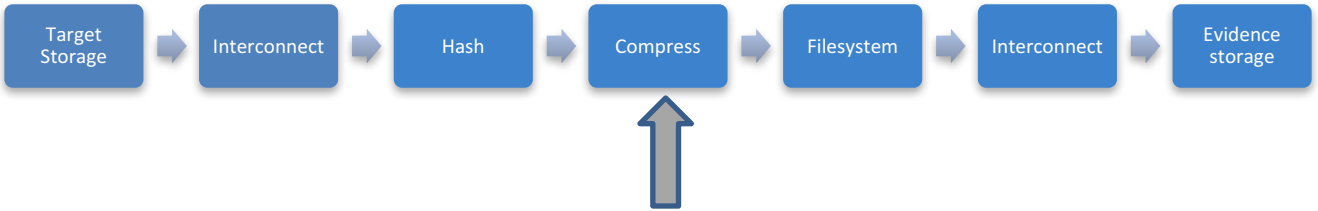
Forensic Imaging v2.1: Threaded EWF



- Good
  - Images are fast to copy (compressed)
  - Near universal tool support
- Bad
  - Inextensible metadata storage
  - Poorly defined*
  - Copying & Compressing sparse regions (zero) is a waste of time**
  - **Deflate compression is a bottleneck**
  - **Linear bitstream hash is a bottleneck at high speeds**

* Despite the excellent work of Metz
** Recent EWF supports sparse regions

The deflate algorithm is a significant bottleneck



Data	Deflate MB/s	Inflate MB/s
High entropy	40.4	439
Low entropy	259	IO bound

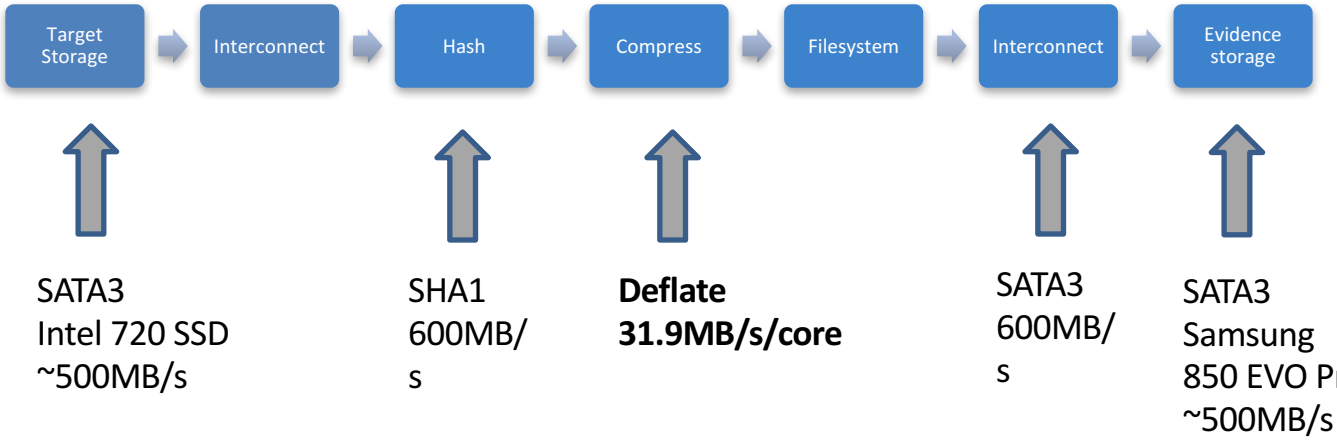
*Single core of quad core i7-4770 3.4Ghz measured with gzip

8-core i7 & uncontended IO?

Threaded EWF is CPU bound

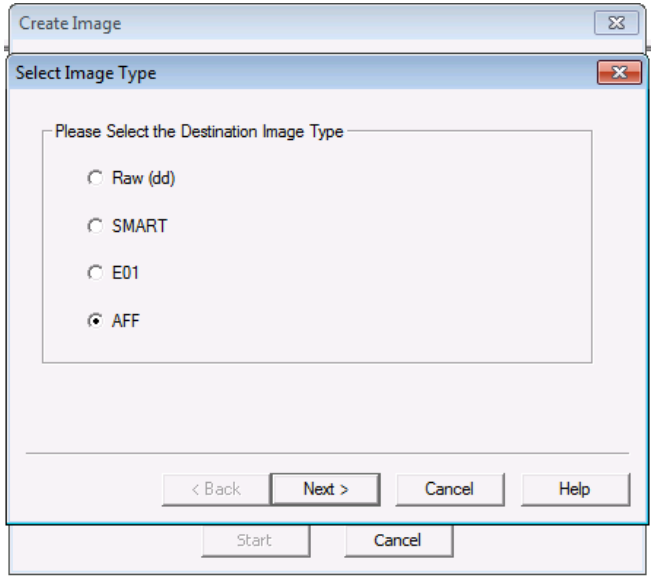
Acquisition	240GB @ **255MB/s**	= 14m 35s
Verification	240GB @ **350MB/s**	= 10m 37s
TOTAL		= 25m

12s



*8 core i7-5820k @ 3.20 GHz

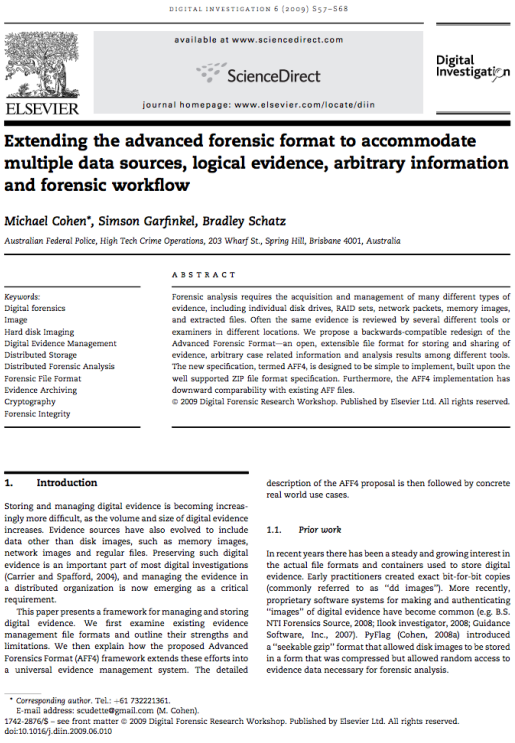
What's wrong with AFF (v1-3)?



- Good
  - Well defined format
  - Open source
  - Extensible Name/Value pair metadata storage
  - Niche commercial tool support
- Bad
  - Copying & Compressing sparse regions (zero) is a waste of time
  - **Deflate compression is a bottleneck**
  - **Large compressed chunk sizes (16M by default) slow w/ NTFS MFT**

What is AFF4?

Forensic Imaging v4.0: AFF4 (2009)



DIGITAL INVESTIGATION 6 (2009) S57–S68

available at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/diin

Digital Investigation

ELSEVIER

Extending the advanced forensic format to accommodate multiple data sources, logical evidence, arbitrary information and forensic workflow

Michael Cohen, Simson Garfinkel, Bradley Schatz*

Australian Federal Police, High Tech Crime Operations, 203 Wharf St., Spring Hill, Brisbane 4001, Australia

Keywords:
Digital forensics
Image
Hard disk Imaging
Digital Evidence Management
Distributed Storage
Distributed Forensic Analysis
Forensic File Format
Evidence Archiving
Cryptography
Forensic Integrity

ABSTRACT

Forensic analysis requires the acquisition and management of many different types of evidence, including individual disk drives, RAID sets, network packets, memory images, and extracted files. Often the same evidence is reviewed by several different tools or examiners in different locations. We propose a backwards-compatible redesign of the Advanced Forensic Format—an open, extensible file format for storing and sharing of evidence, arbitrary case related information and analysis results among different tools. The new specification, termed AFF4, is designed to be simple to implement, built upon the well supported ZIP file format specification. Furthermore, the AFF4 implementation has downward comparability with existing AFF files.

1. Introduction

Storing and managing digital evidence is becoming increasingly more difficult, as the volume and size of digital evidence increases. Evidence sources have also evolved to include data other than disk images, such as memory images, network images and regular files. Preserving such digital evidence is an important part of most digital investigations (Carrier and Spafford, 2004), and managing the evidence in a distributed organization is now emerging as a critical requirement.

This paper presents a framework for managing and storing digital evidence. We first examine existing evidence management file formats and outline their strengths and limitations. We then explain how the proposed Advanced Forensics Format (AFF4) framework extends these efforts into a universal evidence management system. The detailed description of the AFF4 proposal is then followed by concrete real world use cases.

1.1. *Prior work*

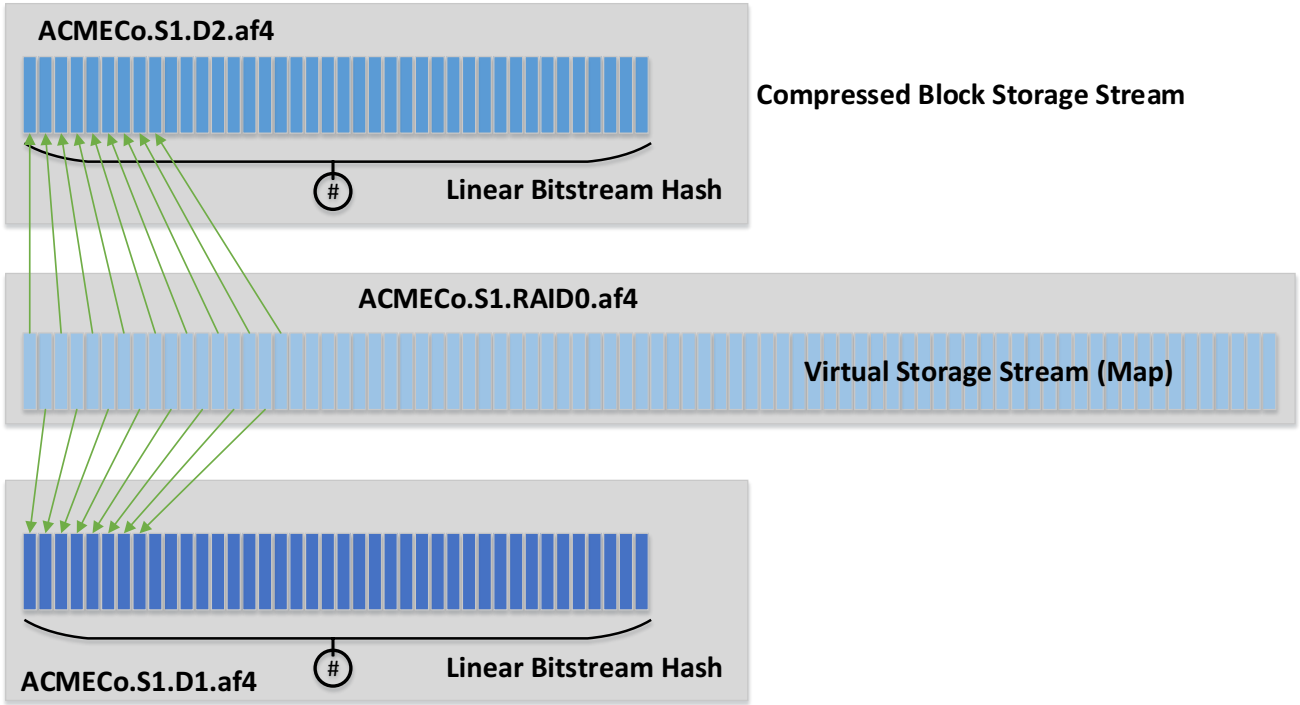
In recent years there has been a steady and growing interest in the actual file formats and containers used to store digital evidence. Early practitioners created exact bit-for-bit copies (commonly referred to as "dd images"). More recently, proprietary software systems for making and authenticating "images" of digital evidence have become common (e.g. B.S. NTI Forensics Source, 2008; Ilook investigator, 2008; Guidance Software, Inc., 2007). PyFlag (Cohen, 2008a) introduced a "seekable gzip" format that allowed disk images to be stored in a form that was compressed but allowed random access to evidence data necessary for forensic analysis.

* *Corresponding author.* Tel.: +61 732221361.
E-mail address: scudette@gmail.com (M. Cohen).
1742-2876/$ – see front matter © 2009 Digital Forensic Research Workshop. Published by Elsevier Ltd. All rights reserved.
doi:10.1016/j.diin.2009.06.010

- **ZIP64 based container**
- **Storage virtualization**
- **Extensible linked-data metadata storage**
- **Inter-container reference scheme**
- **2-level indexing**

- Open source implementation

AFF4 Storage Virtualisation: the Map



Example uses of the AFF4 Map

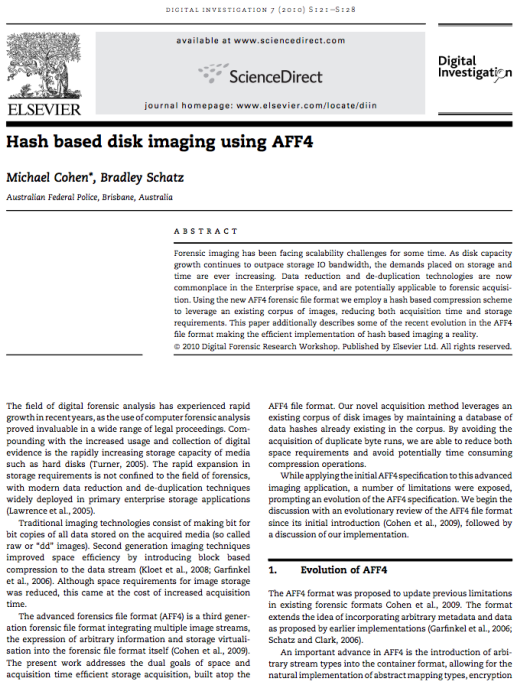
- Reconstructing RAID from images
- Rearranging spare and data ranges in flash images
- Zero storage carving
- Storing discontiguous data ranges
- Storing non-linear images
- Representing sparse regions

Linked data metadata storage

```
<aff4://fd488f0f-95ad-45e4-a948-a36afcb03a08>
    aff4:contains <aff4://08b52fb6-fbae-45f3-967e-03502cefaf92> ;
    aff4:stored <aff4://0658d383-3984-42f5-b1aa-c39f8e0cdbae> ;
    aff4:systemBiosVendor "American Megatrends Inc."^^xsd:string ;
    aff4:systemBiosVersion "F3"^^xsd:string ;
    aff4:systemChassisAssetTag "To Be Filled By O.E.M."^^xsd:string ;
    aff4:systemChassisSerial ""^^xsd:string ;
    aff4:systemChassisType "3"^^xsd:string ;
    aff4:systemChassisVendor "Gigabyte Technology Co., Ltd."^^xsd:string ;
    aff4:systemChassisVersion "To Be Filled By O.E.M."^^xsd:string ;
    aff4:systemEthernetAddress "94:DE:80:7C:EC:6C"^^xsd:string ;
    aff4:systemProductName "Z87X-UD3H"^^xsd:string ;
    aff4:systemProductSerial ""^^xsd:string ;
    aff4:systemProductUUID ""^^xsd:string ;
    aff4:systemProductVersion "To be filled by O.E.M."^^xsd:string ;
    aff4:systemVendor "Gigabyte Technology Co., Ltd."^^xsd:string ;
    aff4:systemboardAssetTag "To be filled by O.E.M."^^xsd:string ;
    aff4:systemboardName "Z87X-UD3H-CF"^^xsd:string ;
    aff4:systemboardSerial ""^^xsd:string ;
    aff4:systemboardVendor "Gigabyte Technology Co., Ltd."^^xsd:string ;
    aff4:systemboardVersion "x.x"^^xsd:string ;
    a aff4:ComputeResource .
```

- Arbitrary information storage
- Refer to data ranges and information
- Inter-container references

Forensic Imaging v4.1: AFF4 (2010)



DIGITAL INVESTIGATION 7 (2010) S121–S128

available at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/diin

Digital Investigation

ELSEVIER

Hash based disk imaging using AFF4

Michael Cohen, Bradley Schatz*

Australian Federal Police, Brisbane, Australia

ABSTRACT

Forensic imaging has been facing scalability challenges for some time. As disk capacity growth continues to outpace storage IO bandwidth, the demands placed on storage and time are ever increasing. Data reduction and de-duplication technologies are now commonplace in the Enterprise space, and are potentially applicable to forensic acquisition. Using the new AFF4 forensic file format we employ a hash based compression scheme to leverage an existing corpus of images, reducing both acquisition time and storage requirements. This paper additionally describes some of the recent evolution in the AFF4 file format making the efficient implementation of hash based imaging a reality.

The field of digital forensic analysis has experienced rapid growth in recent years, as the use of computer forensic analysis proved invaluable in a wide range of legal proceedings. Compounding with the increased usage and collection of digital evidence is the rapidly increasing storage capacity of media such as hard disks (Turner, 2005). The rapid expansion in storage requirements is not confined to the field of forensics, with modern data reduction and de-duplication techniques widely deployed in primary enterprise storage applications (Lawrence et al., 2005).

Traditional imaging technologies consist of making bit for bit copies of all data stored on the acquired media (so called raw or "dd" images). Second generation imaging techniques improved space efficiency by introducing block based compression to the data stream (Kloet et al., 2008; Garfinkel et al., 2006). Although space requirements for image storage was reduced, this came at the cost of increased acquisition time.

The advanced forensics file format (AFF4) is a third generation forensic file format integrating multiple image streams, the expression of arbitrary information and storage virtualisation into the forensic file format itself (Cohen et al., 2009). The present work addresses the dual goals of space and acquisition time efficient storage acquisition, built atop the AFF4 file format. Our novel acquisition method leverages an existing corpus of disk images by maintaining a database of data hashes already existing in the corpus. By avoiding the acquisition of duplicate byte runs, we are able to reduce both space requirements and avoid potentially time consuming compression operations.

While applying the initial AFF4 specification to this advanced imaging application, a number of limitations were exposed, prompting an evolution of the AFF4 specification. We begin the discussion with an evolutionary review of the AFF4 file format since its initial introduction (Cohen et al., 2009), followed by a discussion of our implementation.

1. Evolution of AFF4

The AFF4 format was proposed to update previous limitations in existing forensic formats Cohen et al., 2009. The format extends the idea of incorporating arbitrary metadata and data as proposed by earlier implementations (Garfinkel et al., 2006; Schatz and Clark, 2006).

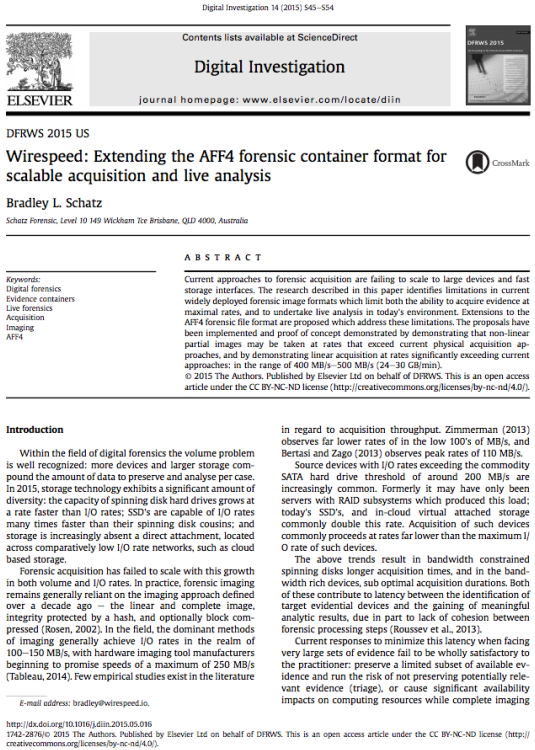
An important advance in AFF4 is the introduction of arbitrary stream types into the container format, allowing for the natural implementation of abstract mapping types, encryption

* Corresponding author.
E-mail address: scudette@gmail.com (M. Cohen).

1742-2876/$ – see front matter © 2010 Digital Forensic Research Workshop. Published by Elsevier Ltd. All rights reserved.
doi:10.1016/j.diin.2010.05.015

- **Non-linear acquisition**
- Hash based imaging (deduplication)

Forensic Imaging v4.2: AFF4 (2015)



Digital Investigation 14 (2015) S45–S54

Contents lists available at ScienceDirect

Digital Investigation

journal homepage: www.elsevier.com/locate/diin

DFRWS 2015 US

Wirespeed: Extending the AFF4 forensic container format for scalable acquisition and live analysis

Bradley L. Schatz

Schatz Forensic, Level 10 149 Wickham Tce Brisbane, QLD 4000, Australia

Keywords:
Digital forensics
Evidence containers
Live forensics
Acquisition
Imaging
AFF4

ABSTRACT

Current approaches to forensic acquisition are failing to scale to large devices and fast storage interfaces. The research described in this paper identifies limitations in current widely deployed forensic image formats which limit both the ability to acquire evidence at maximal rates, and to undertake live analysis in today's environment. Extensions to the AFF4 forensic file format are proposed which address these limitations. The proposals have been implemented and proof of concept demonstrated by demonstrating that non-linear partial images may be taken at rates that exceed current physical acquisition approaches, and by demonstrating linear acquisition at rates significantly exceeding current approaches: in the range of 400 MB/s–500 MB/s (24–30 GB/min).

Introduction

Within the field of digital forensics the volume problem is well recognized: more devices and larger storage compound the amount of data to preserve and analyse per case. In 2015, storage technology exhibits a significant amount of diversity: the capacity of spinning disk hard drives grows at a rate faster than I/O rates; SSD's are capable of I/O rates many times faster than their spinning disk cousins; and storage is increasingly absent a direct attachment, located across comparatively low I/O rate networks, such as cloud based storage.

Forensic acquisition has failed to scale with this growth in both volume and I/O rates. In practice, forensic imaging remains generally reliant on the imaging approach defined over a decade ago – the linear and complete image, integrity protected by a hash, and optionally block compressed (Rosen, 2002). In the field, the dominant methods of imaging generally achieve I/O rates in the realm of 100–150 MB/s, with hardware imaging tool manufacturers beginning to promise speeds of a maximum of 250 MB/s (Tableau, 2014). Few empirical studies exist in the literature in regard to acquisition throughput. Zimmerman (2013) observes far lower rates of in the low 100's of MB/s, and Bertasi and Zago (2013) observes peak rates of 110 MB/s.

Source devices with I/O rates exceeding the commodity SATA hard drive threshold of around 200 MB/s are increasingly common. Formerly it may have only been servers with RAID subsystems which produced this load; today's SSD's, and in-cloud virtual attached storage commonly double this rate. Acquisition of such devices commonly proceeds at rates far lower than the maximum I/O rate of such devices.

The above trends result in bandwidth constrained spinning disks longer acquisition times, and in the bandwidth rich devices, sub optimal acquisition durations. Both of these contribute to latency between the identification of target evidential devices and the gaining of meaningful analytic results, due in part to lack of cohesion between forensic processing steps (Roussev et al., 2013).

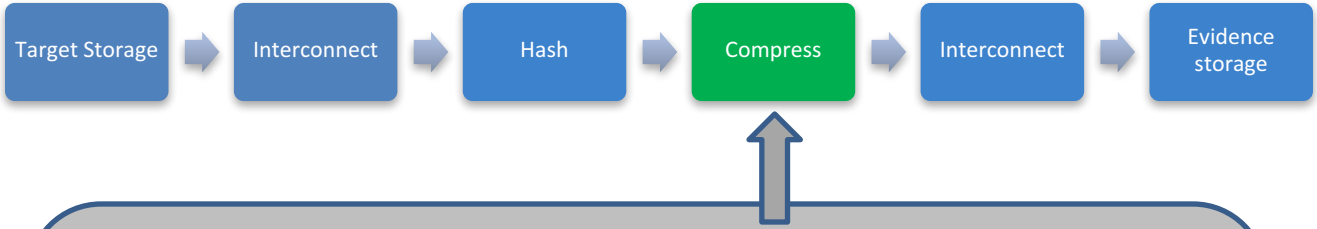
Current responses to minimize this latency when facing very large sets of evidence fail to be wholly satisfactory to the practitioner: preserve a limited subset of available evidence and run the risk of not preserving potentially relevant evidence (triage), or cause significant availability impacts on computing resources while complete imaging

E-mail address: bradley@wirespeed.io.

http://dx.doi.org/10.1016/j.diin.2015.05.016
1742-2876/

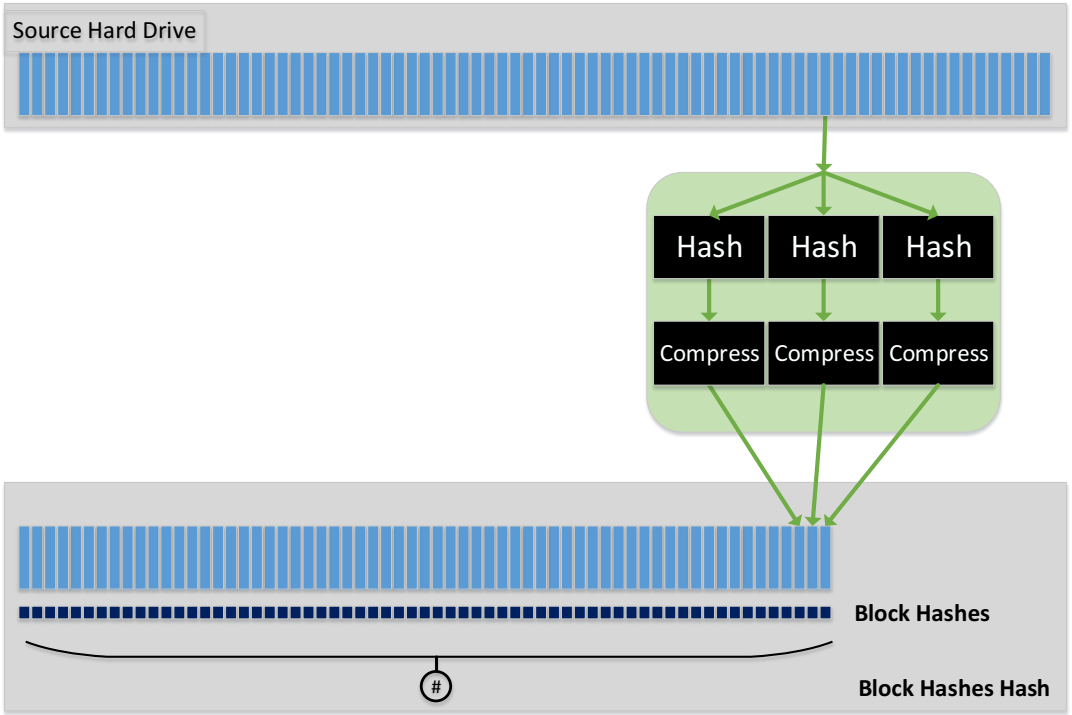
- **Lightweight compression**
- **Block based hashing**
- **Partial acquisition**
  - what we didn't acquire
  - what we couldn't acquire

Lightweight compression



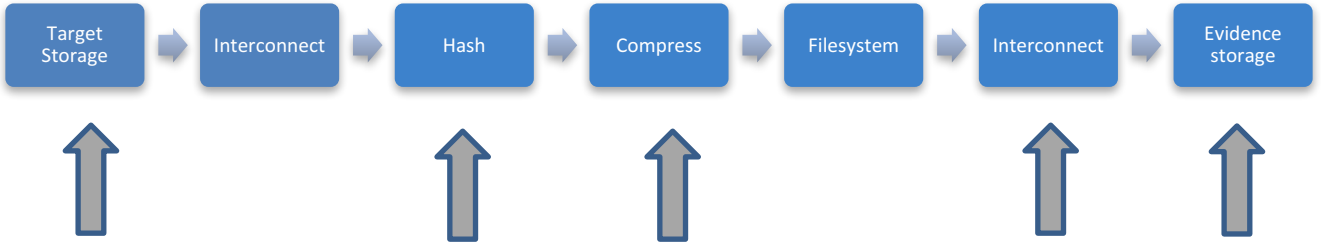
Compression Algorithm	Throughput MB/s/core*
Deflate (ZIP, gzip)	31.9
Snappy (Google BigTable)	**1,400**
LZO (ZFS)	1,540

Block based hashing allows hashing to scale across all cores



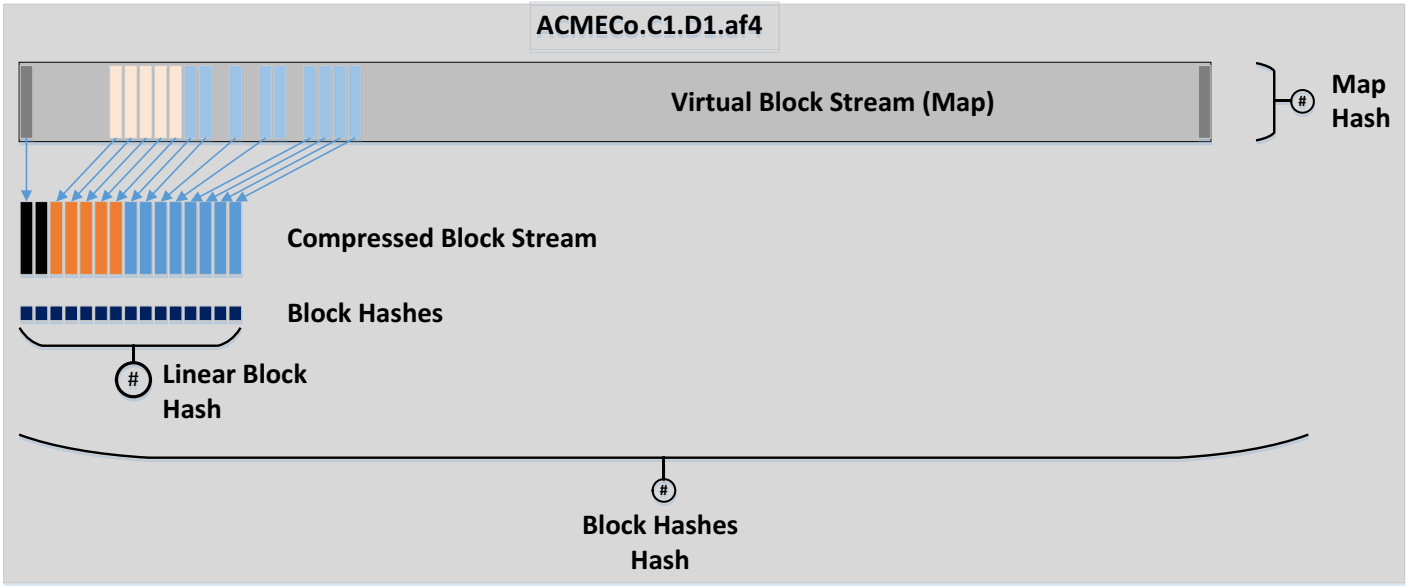
Block hashing shifts the bottleneck from from CPU to I/O

Acquisition application	Linear Acquisition	Verification
X-Ways Forensics	14:35 255 MB/s (15.3 GB/min)	10:37 350 MB/s (21.0 GB/min)
Evimetry (linear)	**7:23** **500 MB/s (30.3 GB/min)**	**4:12** **888 MB/s (53.33 GB/min)**



*8 core i7-5820k @ 3.20 GHz

Generation of a single hash w/ block based hashing



Implementations

- 4 scientifically peer reviewed papers over 6 years
- 3 prototype implementations (2009 – 2013)
  - 3 languages, 4 revisions of the Map
  - Subtle implementation differences due to ambiguity
  - Heritage visible in Google Response Rig
- 2 current implementations
  - Evimetry (Java) & Rekall/libaff4 (C and Python)
- ***Convergence is required***

Convergence:
AFF4 Standardisation Effort

- AFF4 Working Group
  - Bradley Schatz (Evimetry), Michael Cohen (Google) chairing
  - Joe Sylve (Blackbag)
  - First meeting @ DFRWS 2016
- Intended outputs
  - Corpora of standard images [draft]
  - Specification (AFF4s) [draft]
  - Open source implementations (C, Python, Java) [pre-draft]

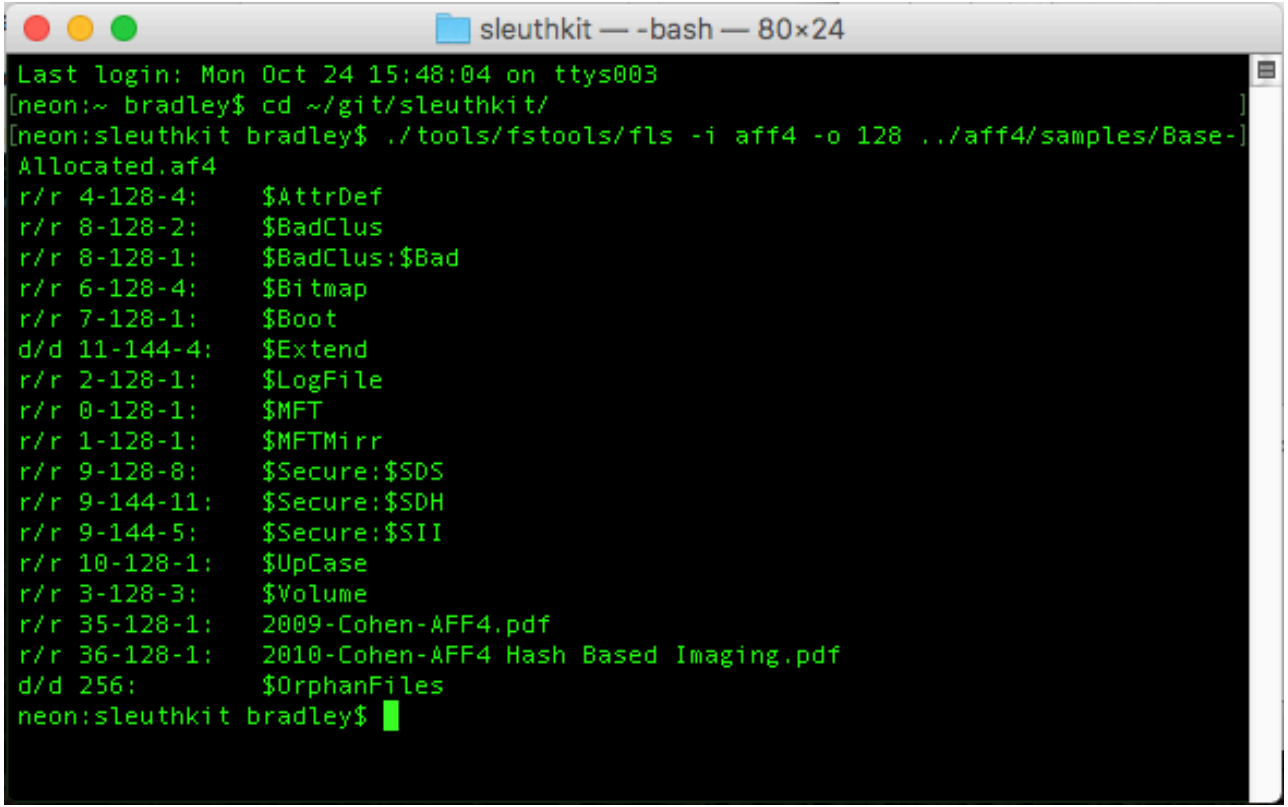
AFF4 Standardisation Effort:
Changes & Clarifications

- Namespace change
  - Was http://afflib.org/ now http://aff4.org/
- Property naming made consistent
- Image Stream Index (compressed block storage)
  - Was [$offset_0$, $offset_1$, $offset_2$, $offset_3$ .. $offset_n$]
  - Now [($offset_0$,$length_0$), ($offset_1$, $length_1$) …]
- Identification of Image in container

Sleuthkit AFF4 support

- Draft standard image (produced by Evimetry)
- Read with libaff4 + AFF4s patches
- Patches to sleuthkit and libaff4 coming very soon

Sleuthkit AFF4 support



Next steps

Volatile memory

- Partial volatile memory acquisition
  - Multi-tenant considerations
  - Virtual machine host kernel memory
- Live volatile memory analysis and acquisition

Front-loaded pre-processing

- File hashing during acquisition
  - Already implemented
  - Spinning disk – optimizing path across disk vs RAM (low seek)
  - SSD – not such an issue
  - No need for expensive processing for known files
- Carving landmark and feature identification
  - No need for expensive disk scans

AFF4 as an interchange format

- Relationship with DFAX etc?

More to come

- Central point of communication
  - http://aff4.org/
- Updates to come
  - DFSci mailing list
  - sleuthkit-users
- Interested in helping?
  - Send us an email
  - bradley@schatzforensic.com.au & scudette@google.com

Contact

Dr Bradley Schatz
https://evimetry.com/
bradley@evimetry.com
@blschatz

'Hard Disk Drive X-Ray'
image by Jeff Kubina