Tupelo – Whole Disk Acquisition, Storage and Search

Stuart Maclean

Center for Environmental and Information Systems
Applied Physics Laboratory
University of Washington
stuart@apl.washington.edu

Open Source Digital Forensics Conference, 2016

Outline

1. What, Why, How
2. Acquisition/Analysis
3. Search
4. Availability

Tupelo - What?

- Tupelo is an open-source Java/C codebase for efficient whole disk acquisition, storage and analysis.
- Analysis step leverages existing open-source Sleuthkit disk forensics library to walk filesystems.
- Integrates with emerging standard STIX (Structured Threat Information Expression) to ingest and author shared information about malicious artifacts.
- Makes use of other Java artifacts in the disk forensics arena.

Tupelo - Why?

- If the disk whose content you wish to capture is suspected of containing malicious artifacts, how can software residing on that same disk be relied upon to present accurate disk content?
- To overcome this problem of trust, Tupelo does “dead disk acquisition”, and runs from trusted media, e.g. a bootable CD/USB. Yes, you have to power down and reboot. Alternatives?

Tupelo - How?

- Users acquire whole disk device contents, storing a copy in a *Tupelo store*.
- Data transitions from *unmanaged* (user disk) to *managed* (stored copy).
- Once stored, content is read-only, and analyzed: filesystems, un-allocated areas.
- Analysis results placed in the store alongside the data as *attributes*, key/value pairs with arbitrary values.
- Same disk can be acquired repeatedly, and many disks can be acquired.
- Store then essentially a structured (but not relational) database. The (logical) unit of storage is 'whole disk at a given time'.

Tupelo Terms, Preparation

Tupelo's command-line inspired by git (single driver program, many sub-commands). First, identify the disk to acquire and the store to hold that acquisition:

```
acquirer$ tup device add HD /dev/sda
id   = ATA-WDC-WX71C6287816                     // unique!
size = 320GB

acquirer$ tup store add ES /mounted/external/4TB
space = 4TB
```

Device and store 'adds' associate easy-to-use names with hard-to-use names.

Tupelo Terms, Preparation

Tupelo's command-line inspired by git (single driver program, many sub-commands).
First, identify the disk to acquire and the store to hold that acquisition:

```
acquirer$ tup device add HD /dev/sda
id   = ATA-WDC-WX71C6287816                       // unique!
size = 320GB

acquirer$ tup store add ES /mounted/external/4TB
space = 4TB
```

Device and store 'adds' associate easy-to-use names with hard-to-use names.

Tupelo Terms, Preparation

Tupelo's command-line inspired by `git` (single driver program, many sub-commands). First, identify the disk to acquire and the store to hold that acquisition:

```
acquirer$ tup device add HD /dev/sda
id   = ATA-WDC-WX71C6287816                     // unique!
size = 320GB

acquirer$ tup store add ES /mounted/external/4TB
space = 4TB
```

Device and store 'adds' associate easy-to-use names with hard-to-use names.

Tupelo Disk Capture

True dead-filesystem capture requires a bootable Linux CD with Tupelo added. Capture (push) destination must be 'off-disk'. Both local external drive, remote locations work:

```
acquirer@bootCD$ tup device add HD /dev/sda
id   = ATA-WDC-WX71C6287816

acquirer@bootCD$ tup store add LAS /mounted/external/4TB
space = 4TB

acquirer@bootCD$ tup store add WS https://webAccessedTupeloStore/
space = 2.2TB

acquirer@bootCD$ tup push HD LAS ; tup push HD WS
```

Our prototype boot CD is Caine plus Tupelo.

Tupelo Disk Capture

True dead-filesystem capture requires a bootable Linux CD with Tupelo added. Capture (push) destination must be 'off-disk'. Both local external drive, remote locations work:

```
acquirer@bootCD$ tup device add HD /dev/sda
id   = ATA-WDC-WX71C6287816

acquirer@bootCD$ tup store add LAS /mounted/external/4TB
space = 4TB

acquirer@bootCD$ tup store add WS https://webAccessedTupeloStore/
space = 2.2TB

acquirer@bootCD$ tup push HD LAS ; tup push HD WS
```

Our prototype boot CD is Caine plus Tupelo.

Virtual Disk Capture

Tupelo also reads virtual machine data. A powered-off VM satisfies requirements for dead-filesystem capture:

```
acquirer$ tup device add XP /path/to/VirtualBox/WindowsXP-VM
id   = VMDK-2fe54bfe
size = 10GB

acquirer$ tup push XP WS
```

You can of course also capture a 'live system'. Trust?

Whole Disk Acquisition Is Space Efficient

A disk push results in a store entry tagged by what and when. Here we capture a laptop drive, dual-boot Windows/Linux, 320GB:

Acquirer@BootCD

```
acquirer$ tup push HD ES
Timestamp : 2016102301
Unmanaged : 320GB
Managed   : 197GB
Elapsed   : 8536s
```

StoreFilesystem@ExternalDrive

```
admin$ tree /path/to/TupeloStore
ATA-WDC-WX71C6287816
 2016102301
   ATA-WDC-WX71C6287816-2016102301.tmd
```

Define a *grain* as a sequence of sectors, typically 128 sectors (64K). We then push the disk grain-by-grain. Can mark all-zero grains as special and compress all other grains. Result: a 123GB space saving in this case.

Whole Disk Acquisition Is Space Efficient

A disk push results in a store entry tagged by what and when. Here we capture a laptop drive, dual-boot Windows/Linux, 320GB:

Acquirer@BootCD

```
acquirer$ tup push HD ES
Timestamp : 2016102301
Unmanaged : 320GB
Managed   : 197GB
Elapsed   : 8536s
```

StoreFilesystem@ExternalDrive

```
admin$ tree /path/to/TupeloStore
ATA-WDC-WX71C6287816
 2016102301
   ATA-WDC-WX71C6287816-2016102301.tmd
```

Define a *grain* as a sequence of sectors, typically 128 sectors (64K). We then push the disk grain-by-grain. Can mark all-zero grains as special and compress all other grains. Result: a 123GB space saving in this case.

Whole Disk Acquisition Is Space Efficient

A disk push results in a store entry tagged by what and when. Here we capture a laptop drive, dual-boot Windows/Linux, 320GB:

Acquirer@BootCD

```
acquirer$ tup push HD ES
Timestamp : 2016102301
Unmanaged : 320GB
Managed   : 197GB
Elapsed   : 8536s
```

StoreFilesystem@ExternalDrive

```
admin$ tree /path/to/TupeloStore
ATA-WDC-WX71C6287816
 2016102301
   ATA-WDC-WX71C6287816-2016102301.tmd
```

Define a *grain* as a sequence of sectors, typically 128 sectors (64K). We then push the disk grain-by-grain. Can mark all-zero grains as special and compress all other grains. Result: a 123GB space saving in this case.

Whole Disk Acquisition Is Space Efficient

A disk push results in a store entry tagged by what and when. Here we capture a laptop drive, dual-boot Windows/Linux, 320GB:

Acquirer@BootCD

```
acquirer$ tup push HD ES
Timestamp : 2016102301
Unmanaged : 320GB
Managed   : 197GB
Elapsed   : 8536s
```

StoreFilesystem@ExternalDrive

```
admin$ tree /path/to/TupeloStore
ATA-WDC-WX71C6287816
 2016102301
   ATA-WDC-WX71C6287816-2016102301.tmd
```

Define a *grain* as a sequence of sectors, typically 128 sectors (64K). We then push the disk grain-by-grain. Can mark all-zero grains as special and compress all other grains. Result: a 123GB space saving in this case.

Operations On Store Content: Digest

After acquisition, put on Tupelo admin hat and process the new store addition. First, we digest the new content. Produces an MD5 hash of each grain, so can represent 64KB in 16 bytes. Our 320GB disk digests to 16MB.

Administrator@Store

```
admin$ tup digest S 1
Digest : 16MB
```

StoreFilesystem@ExternalDrive

```
admin$ tree /path/to/TupeloStore
ATA-WDC-WX71C6287816
 2016102301
   ATA-WDC-WX71C6287816-2016102301.tmd
   ATA-WDC-WX71C6287816-2016102301.md5
```

Why digest? To make future captures of this disk further space-optimized.

Operations On Store Content: Digest

After acquisition, put on Tupelo admin hat and process the new store addition. First, we digest the new content. Produces an MD5 hash of each grain, so can represent 64KB in 16 bytes. Our 320GB disk digests to 16MB.

Administrator@Store

```
admin$ tup digest S 1
Digest : 16MB
```

StoreFilesystem@ExternalDrive

```
admin$ tree /path/to/TupeloStore
ATA-WDC-WX71C6287816
 2016102301
    ATA-WDC-WX71C6287816-2016102301.tmd
    ATA-WDC-WX71C6287816-2016102301.md5
```

Why digest? To make future captures of this disk further space-optimized.

Operations On Store Content: Analysis

Exposing store contents as a mount point leverages any software that can read device files, e.g. Sleuthkit. All done in-place, no need to 'inflate' anything (which costs disk!)

```
$ mkdir mnt; tup mount ES mnt

$ mmls mnt/ATA-WDC-WX71C6287816/2016102301

$ fls -o 2048 mnt/ATA-WDC-WX71C6287816/2016102301

$ fiwalk mnt/ATA-WDC-WX71C6287816/2016102301

$ autopsy mnt/ATA-WDC-WX71C6287816/2016102301 ?

$ cat mnt/ATA-WDC-WX71C6287816/2016102301 > /dev/sda !!
```

Operations On Store Content: Tupelo Additions

```
$ tup info ES
1 ATA-WDC-WX71C6287816, 2016102301 (320GB)

$ tup hashvs   ES 1 ; tup hashfs ES 1
$ tup bodyfile ES 1 ; tup winrej ES 1

1 ATA-WDC-WX71C6287816, 2016102301 (320GB)
 1 hashvs
 2 hashfs-2048-716800             // NTFS
 3 hashfs-718848-195306576        // NTFS
 4 hashfs-196026368-381857792     // EXT4
 5 bodyfile-2048-716800
 6 bodyfile-718848-195306576
 7 bodyfile-196026368-381857792
```

Operations On Store Content: Tupelo Additions

```
$ tup info ES
1 ATA-WDC-WX71C6287816, 2016102301 (320GB)

$ tup hashvs   ES 1 ; tup hashfs ES 1
$ tup bodyfile ES 1 ; tup winrej ES 1

1 ATA-WDC-WX71C6287816, 2016102301 (320GB)
 1 hashvs
 2 hashfs-2048-716800             // NTFS
 3 hashfs-718848-195306576        // NTFS
 4 hashfs-196026368-381857792     // EXT4
 5 bodyfile-2048-716800
 6 bodyfile-718848-195306576
 7 bodyfile-196026368-381857792
```

Operations On Store Content: Tupelo Additions

```
$ tup info ES
1 ATA-WDC-WX71C6287816, 2016102301 (320GB)

$ tup hashvs   ES 1 ; tup hashfs ES 1
$ tup bodyfile ES 1 ; tup winrej ES 1

1 ATA-WDC-WX71C6287816, 2016102301 (320GB)
 1 hashvs
 2 hashfs-2048-716800             // NTFS
 3 hashfs-718848-195306576        // NTFS
 4 hashfs-196026368-381857792     // EXT4
 5 bodyfile-2048-716800
 6 bodyfile-718848-195306576
 7 bodyfile-196026368-381857792
```

Operations On Store Content: Tupelo Additions

```
$ tup info ES
1 ATA-WDC-WX71C6287816, 2016102301 (320GB)

$ tup hashvs   ES 1 ; tup hashfs ES 1
$ tup bodyfile ES 1 ; tup winrej ES 1

1 ATA-WDC-WX71C6287816, 2016102301 (320GB)
 1 hashvs
 2 hashfs-2048-716800             // NTFS
 3 hashfs-718848-195306576        // NTFS
 4 hashfs-196026368-381857792     // EXT4
 5 bodyfile-2048-716800
 6 bodyfile-718848-195306576
 7 bodyfile-196026368-381857792
```

Analysis Result: Unallocated Areas

```
$ tup hashvs -p ES 1
START      LEN                              MD5
0            1 dd834b02d5e7dbad368cc81194e75958d02e085c
0         2048 b350e07662708ed52fe0ef637e2c87370180c4b7
196025424  944 56f5073640009f85d7576781476df2ea5cb90dd9
577884160 2048 8c993a0b65f6ad7df717101db21ca09cd4d2ccee
625139712 2736 24f9417794527c5b73e92aa6d23e88ee8f6924c0
```

Why? To track unallocated area changes over time.

Analysis Result: File Hashes

```
$ tup hashfs -p ES 1.4

164ebd6889588da166a52ca0d57b9004 bin/bash
0a35aa198d80c3b7ebcdd0cefca38063 bin/bunzip2
ad9d7ce76bac4a59ece0a01f717ce2d5 bin/busybox
...
```

Why? To leverage efficient file identification given content hash, a common Indicator-Of-Compromise (STIX?).

Analysis Result: Sleuthkit Bodyfiles

A bodyfile captures aspects of a file: owner, permissions, timestamps, content hash:

```
$ tup bodyfile -p ES 1.3

127aa81343a7c6f665c22cb1293b0a90|/Windows/splwow64.exe|69122|
r/rrwxrwxrwx|0|0|67072|1402952402|1427400287|1427400287|1402952402

78414f7183e7af72de7f691ed7a37b33|/Windows/TSSysprep.log|85334|
r/rrwxrwxrwx|0|0|5949|1401489442|1428772602|1428772602|1401489442

163a95975e1d8819e653aa3e961371ca|/Windows/twain_32.dll|25115|
r/rrwxrwxrwx|0|0|51200|1290309910|1427400032|1427400032|1290309910
```

Why? To leverage efficient lookup of file changes over time.

Normal Computer Use

Reboot to normal operations. Over time, disk content changes...

```
// Read the news, installs cookies
$ firefox news.bbc.co.uk

// Install new software, intentional
$ apt-get install octaveMatlabClone

// Install new software, un-intentional
$ attachmentInstallsMalwareAndSilencesAntiVirus
```

Next, capture whole disk again, via second Tupelo push. Can then compare disk state before, after this activity.

Repeated Acquisitions Increase Store Performance

So, disk content changed. Boot Tupelo CD, push disk, every Friday perhaps:

Acquirer@BootCD

```
acquirer$ tup push HD ES
Timestamp : 2016102501
Unmanaged : 320GB
Managed   : 1.4GB
Elapsed   : 4410s
```

StoreFilesystem@ExternalDrive

```
ATA-WDC-WX71C6287816
 2016102301
   ATA-WDC-WX71C6287816-2016102301.tmd
   ATA-WDC-WX71C6287816-2016102301.md5
 2016102501
   ATA-WDC-WX71C6287816-2016102501.tmd
```

Note the new stored size of only 1.4GB! By retrieving the pre-computed digest and comparing grains in original and new captures, we can mark many grains in new capture 'same as parent'. Vastly improves the net space efficiency of captured disks.

Repeated Acquisitions Increase Store Performance

So, disk content changed. Boot Tupelo CD, push disk, every Friday perhaps:

Acquirer@BootCD

```
acquirer$ tup push HD ES
Timestamp : 2016102501
Unmanaged : 320GB
Managed   : 1.4GB
Elapsed   : 4410s
```

StoreFilesystem@ExternalDrive

```
ATA-WDC-WX71C6287816
 2016102301
   ATA-WDC-WX71C6287816-2016102301.tmd
   ATA-WDC-WX71C6287816-2016102301.md5
 2016102501
   ATA-WDC-WX71C6287816-2016102501.tmd
```

Note the new stored size of only 1.4GB! By retrieving the pre-computed digest and comparing grains in original and new captures, we can mark many grains in new capture 'same as parent'. Vastly improves the net space efficiency of captured disks.

Operators Applied To Second Acquisition

```
admin$ tup info ES
1 ATA-WDC-WX71C6287816, 2016102301 (320GB)
 1 hashvs
 2 hashfs-2048-716800
 3 hashfs-718848-195306576
 4 bodyfile-2048-716800
 5 bodyfile-718848-195306576
2 ATA-WDC-WX71C6287816, 2016102501 (320GB)
 1 hashvs
 2 hashfs-2048-716800
 3 hashfs-718848-195306576
 4 bodyfile-2048-716800          // Diff with 1.4?
 5 bodyfile-718848-195306576     // Diff with 1.5?
```

Store Search: Indicator Of Compromise

```
$ cat iocs.stix.xml
<cybox:Object><FileObj:Hashes><cyboxCommon:Hash>
<cyboxCommon:Simple_Hash_Value>
 e83cf86a39caf748d2199dc8d3b92e60
</cyboxCommon:Simple_Hash_Value>
</cyboxCommon:Hash></FileObj:Hashes></cybox:Object>

admin$ tup search ES iocs.stix.xml
Hit: e83cf86a39caf748d2199dc8d3b92e60
(ATA-WDC-WX71C6287816, 2016102501) WINDOWS/system32/mstc.exe
```

Store Search: Context Of Compromise

By differencing stored bodyfiles, can see what other files appeared with the IOC hit:

```
admin$ tup bodyfile -d ES 2.3 1.3
e83cf86a39caf748d2199dc8d3b92e60|/WINDOWS/system32/mstc.exe|
10858|r/rrwxrwxrwx|0|0|181248|
1477446120|1477446120|1477446120|1321483658

732cfc10b216a79e8ae5cd186015b476|
/Documents and Settings/apluw/Application Data/FNTCACHE.BIN|
10862|r/rrwxrwxrwx|0|0|32|
1477446128|1477446128|1477446128|1456262350
```

First file is bot-net malware binary. Second file is its keystroke log.

Store Search: Context Of Compromise

By differencing stored bodyfiles, can see what other files appeared with the IOC hit:

```
admin$ tup bodyfile -d ES 2.3 1.3
e83cf86a39caf748d2199dc8d3b92e60|/WINDOWS/system32/mstc.exe|
10858|r/rrwxrwxrwx|0|0|181248|
1477446120|1477446120|1477446120|1321483658

732cfc10b216a79e8ae5cd186015b476|
/Documents and Settings/apluw/Application Data/FNTCACHE.BIN|
10862|r/rrwxrwxrwx|0|0|32|
1477446128|1477446128|1477446128|1456262350
```

First file is bot-net malware binary. Second file is its keystroke log.

Store Search: Context Of Compromise

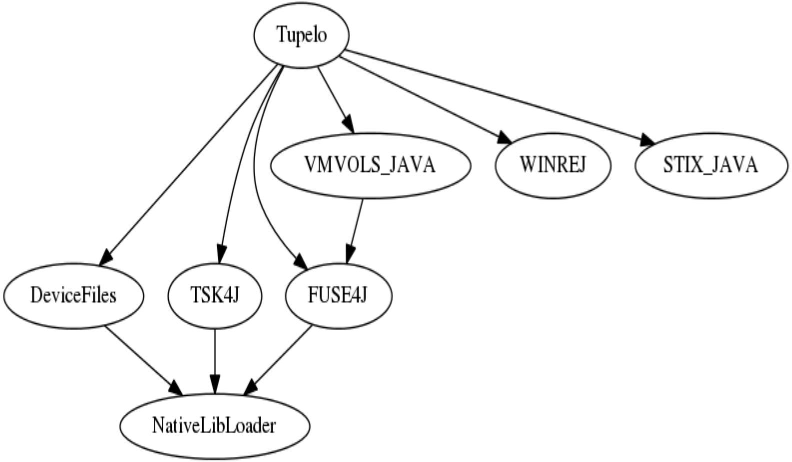
By differencing stored bodyfiles, can see what other files appeared with the IOC hit:

```
admin$ tup bodyfile -d ES 2.3 1.3
e83cf86a39caf748d2199dc8d3b92e60|/WINDOWS/system32/mstc.exe|
10858|r/rrwxrwxrwx|0|0|181248|
1477446120|1477446120|1477446120|1321483658

732cfc10b216a79e8ae5cd186015b476|
/Documents and Settings/apluw/Application Data/FNTCACHE.BIN|
10862|r/rrwxrwxrwx|0|0|32|
1477446128|1477446128|1477446128|1456262350
```

First file is bot-net malware binary. Second file is its keystroke log.

Tupelo Software Composition, Dependencies



Each is a public git repo, with one or more Java/Maven artifacts. Use in other projects!

Talk Is Cheap. Where Is The Code?

Tupelo and several Java libraries on which it depends are open-source:

github.com/UW-APL-EIS/tupelo Main Tupelo logic. Disk acquire, differencing, stores.

github.com/UW-APL-EIS/vmvols-java - Virtual machine disk access to host software.

github.com/UW-APL-EIS/winrej - Windows registry hive parser.

github.com/uw-dims/tsk4j - Java bindings to Sleuthkit.

github.com/uw-dims/stix-java - Java bindings to STIX.

github.com/uw-dims/fuse4j - Java bindings to FUSE.

github.com/uw-dims/device-files - Reads disk serial number, size.

github.com/uw-dims/java-native-loader - Framework for split Java/C codebases.