BASIS
TECH
WEEK

4TH ANNUAL
OSDF

BASIS TECHNOLOGY

2013 Open Source Digital Forensics Conference

DATA TRIAGE: The art of making molehills out of mountains

Tobin Craig, MRSC, CISSP, CCE, CFCE

Lab Chief, Computer Crimes Unit

Department of Transportation, Office of Inspector General

OSDF

WHO AM I

Lab Chief, Computer Crimes Unit, DOT OIG

Prior lives:

Lab Chief, CCD, NASA OIG

Forensic Lab Director, VAOIG

Document Analyst, FSD, US Secret Service

Trace Chemist, Northern Ireland

I AM NOT A PROGRAMMER!

OSDF

INTRODUCTION

Data triage?

Identifying material of potential investigative interest, and presenting it to a case agent for his or her scrutiny.

OSDF

SCOPE: A TYPICAL WARRANT

Warrant environment

- Corporate fraud/white collar crime
  - Financial
  - Paper trails/invoices
  - Emails
- Windows environments
- Industry standard/most commonly used productivity applications

Warrant result

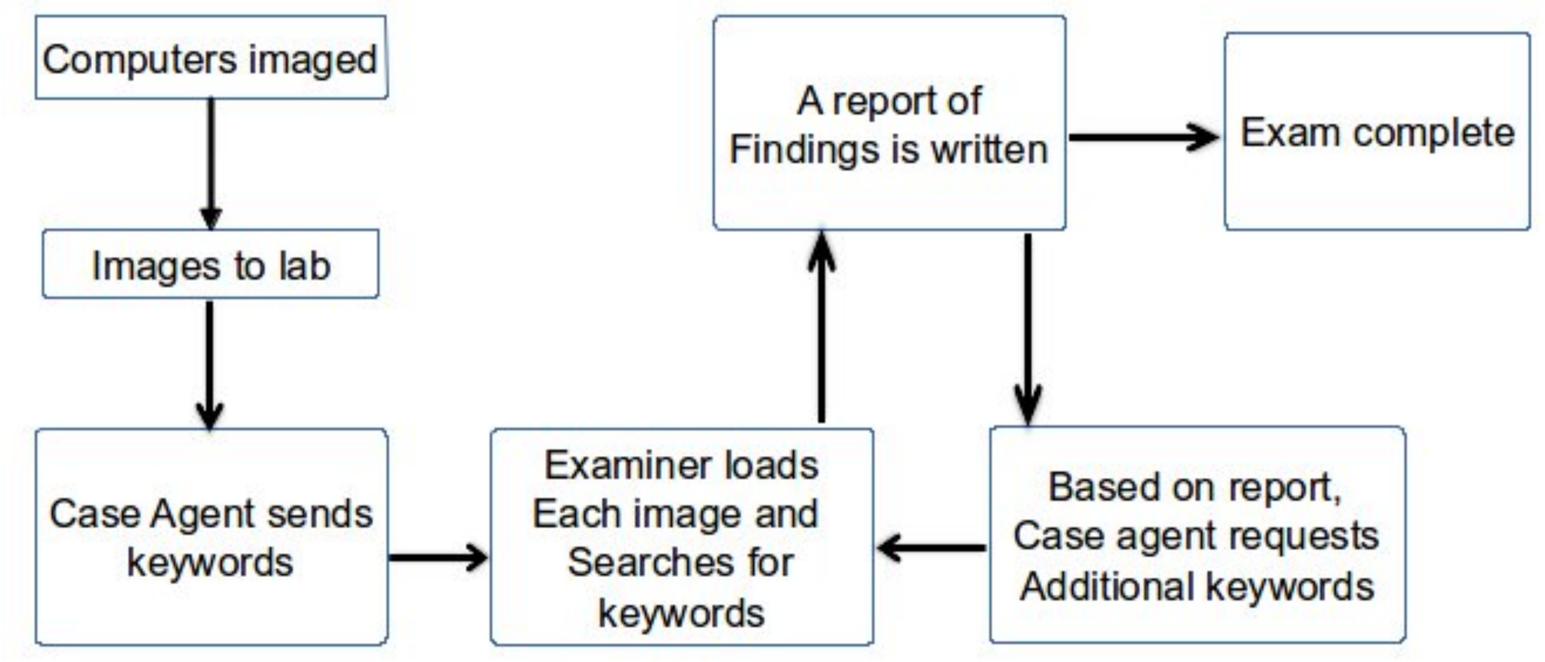
- 6 TB (on a good day!) and up of acquired data, spread over numerous drives and servers
- Process developed to address MOST encountered scenarios.

CURRENT FORENSIC APPLICATIONS

- Dongle dependent;
- $$$$$;
- System intensive;
- Unwieldy if considering more than a few images at a time;
- Depends on examiner to identify and provide relevant results to case agent (not always possible as investigative focus shifts).

OSDF

A TYPICAL SCENARIO


Computers imaged
Images to lab
Case Agent sends keywords
Examiner loads Each image and Searches for keywords
A report of Findings is written
Exam complete
Based on report, Case agent requests Additional keywords

- Typical acquisition in range of 6 TB and upwards;
- Often more than 10 computers;
- Dynamically evolving keyword list;
- Severe time restraints.

PROPOSED SOLUTION

- No size limitation
  - 30+ images
- Efficient
  - 5TB in ~24hrs, 90% reduction
- Cost-effective

  Uses existing resources

  No software licenses

  Entirely open source

MEDIA DATA EXTRACTION PROCESS

USER FILES

- Documents
- Spreadsheets
- Presentations
- Email
- Web pages
- Databases
- Financial records

THE OBJECTIVE

- identify the “low hanging fruit”;
- provide information in a manner familiar to non-technical investigators;
- quickly identify files of interest;
- create comprehensive case notes;
- generate targeted forensic reports.

OSDF

PARAMETERS

- use open source solutions;
- Portable;
- generate meaningful output for both examiner and case agent;
- simple to deploy

THE MDE SCRIPT

- Run in a trusted environment – Deft 8
- Bash script
- Uses linux tools and commands
- Incorporated sleuthkit tools
- Generates standard reports
- Output data retains original metadata information and directory structure

OSDF

LIMITATIONS

- This script is limited NTFS or FAT (ie Microsoft) filesystems
- It will **NOT** perform keyword searches
- Relies (unashamedly) on file extensions.
  - Most of our work involves files which use conventional file extensions. If there is a chance that a user is deploying unique or unusual file extensions (or no extensions!), then use another tool.

OSDF

MDE: PART ONE

- What is your Agency?
- What is the Case Number?
- What is the Case Name?
- Who is the Examiner?
- Where do you want to write output to?
    - The script will check the path you provide, and if it doesn't exist, it will create the output folder you specify.

MDE: PART ONE

- What is the scope of the extraction?
  - Files NEWER than a specified date
  - Files OLDER than a specified date
  - Files BETWEEN two specified dates
  - No date restraints
    - Setting up variables for use later
- What do you want to examine?
  - An acquired image or a collection of acquired images
  - An existing accessible file system

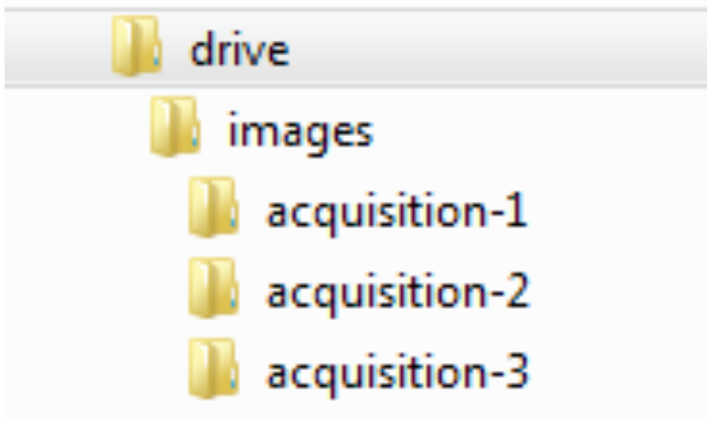
OSDF

MDE: PART ONE

What is the path to the PARENT folder for the data?

- Each folder in the “images” location is uniquely named.
  - use the acquired drive Serial Number to uniquely name the folders. NO SPACES!

In this case, path required will be /drive/images



MDE: PART TWO

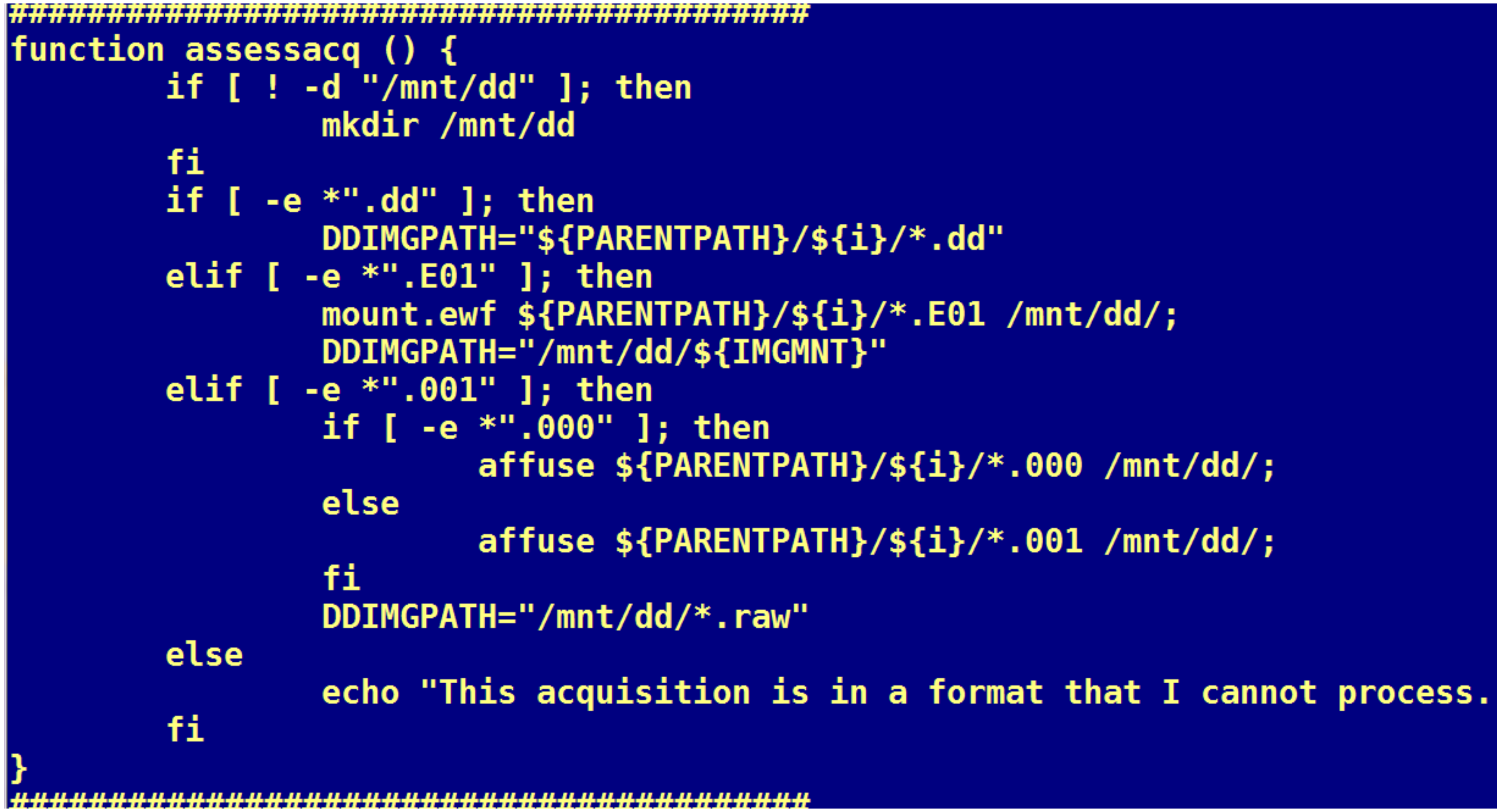
- `PARENTPATH`: variable capturing the location of the collection of acquisition folders
- Script reads the `$PARENTPATH`, and creates a list of folders found there in `/tmp` as a txt list.

```
ls -lad ${PARENTPATH}/* |grep ^"d"|awk -F "/" '{ print $NF }' \
 > /tmp/CCU-${CASENUMBER}-folders.txt
   for i in $(cat /tmp/CCU-${CASENUMBER}-folders.txt); do
   cd ${PARENTPATH}/${i};
#new feature: check for spaces in image filenames and replace \
 with underscore:
       find -name "* *" -type f | rename 's/ /_/g'
```

- identifies the unique name based on the acquisition image naming format - `$IMGMNT`

OSDF

MDE: PART TWO: mounting



```
#################################################
function assessacq () {
        if [ ! -d "/mnt/dd" ]; then
                mkdir /mnt/dd
        fi
        if [ -e *".dd" ]; then
                DDIMGPATH="${PARENTPATH}/${i}/*.dd"
        elif [ -e *".E01" ]; then
                mount.ewf ${PARENTPATH}/${i}/*.E01 /mnt/dd/;
                DDIMGPATH="/mnt/dd/${IMGMNT}"
        elif [ -e *".001" ]; then
                if [ -e *".000" ]; then
                        affuse ${PARENTPATH}/${i}/*.000 /mnt/dd/;
                else
                        affuse ${PARENTPATH}/${i}/*.001 /mnt/dd/;
                fi
                DDIMGPATH="/mnt/dd/*.raw"
        else
                echo "This acquisition is in a format that I cannot process.
        fi
}
#################################################
```

OSDF

MDE: PART TWO: mounting

Now we need to find the partitions where users will typically interact

Use the sleuthkit command, `mmls`

```
% mmls -t dos disk.dd
DOS Partition Table
Units are in 512-byte sectors

     Slot    Start        End          Length       Description
00:  Meta    0000000000   0000000000   0000000001   Primary Table (#0)
01:  -----   0000000000   0000000062   0000000063   Unallocated
02:  00:00   0000000063   0002056319   0002056257   Win95 FAT32 (0x0B)
03:  00:01   0002056320   0008209214   0006152895   OpenBSD (0xA6)
04:  00:02   0008209215   0019999727   0011790513   FreeBSD (0xA5)
```

These are the columns I'm interested in.

OSDF

MDE: PART TWO: mounting

```
for p in $(mmls ${DDIMGPATH} | grep -i \
  ‘NTFS\|Win\|FAT\|DOS‘ |  grep -iv \
'table\|extended‘ | awk '{ print $3 ":"
  $6 }'); do
```

- This runs the `mmls` command on the mounted dd image, selects only those rows from the output with NTFS, Win, FAT or DOS, then deselects any rows with “table” or “extended”, and finally identifies the **third** and the **sixth** column values for `$p:`

```
 PARTOFFSET=`echo "${p}" |awk -F : '{ print $1 }'`
PARTTYPE=`echo "${p}" |awk -F : '{ print $2 }'`
```

OSDF

MDE: PART TWO: examnotes

- The **`examnotes`** function:
- documents the assessed image;
- recording the collected variables;
- generates a detailed narrative
  - case information,
  - Examiner,
  - name parameters and
  - partitioning for the image assessed.
- Output is written to

```
${OUTPATH}/CCU_DOCUMENTS/CCU-
EXAM-NOTES-${IMGMNT}-offset-${PARTOFFSET}.doc
```

MDE: PART TWO: `flsexam`

- The **`flsexam`** function uses the sleuthkit command `fls` to generate four spreadsheets:
  - Allocated files;
  - Allocated folders;
  - Deleted files;
  - Deleted folders.
- The function reads values stored in arrays to run the `fls` command with different options,
  – the output is captured in the form of a csv spreadsheet, with the following column headers:

OSDF

MDE: PART TWO: `flsexam`

- `INODE\MFT_ENTRY`
- `FILENAME`
- `MODIFIED_TIME`
- `ACCESSED_TIME`
- `CHANGE_TIME`
- `CREATED_TIME`
- `FILE_SIZE`
- `UID`
- `GID`

- Note that `fls` records details based upon MFT entry, and so path details are not captured.

OSDF

MDE: PART TWO: `flsexam`

How it works:

```
declare -a FLSOPT=(Dlupr Dldpr Flupr Fldpr)
declare -a FLSOUTPUTCSV=(allocated-folders \ deleted-
  folders allocated-files deleted-files)
```

(these are the values that will be used for switches)

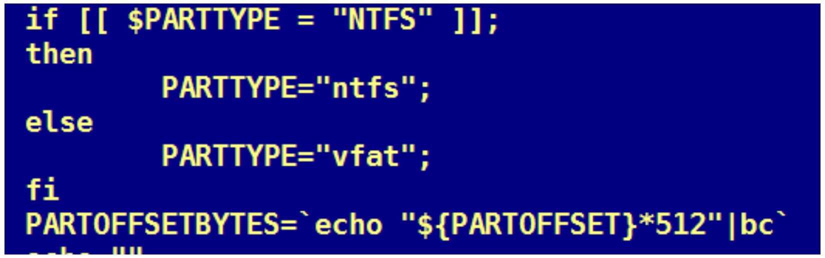
```
fls -${FLSOPT[$i]} -o ${PARTOFFSET} ${DDIMGPATH} \
  >>{OUTPATH}/CCU_DOCUMENTS/CCU-FLS-output-\
${IMGMNT}-offset-${PARTOFFSET}${FLSOUTPUTCSV[$i]}.csv
```

Run the `fls` command, using the switches determined by the value `$i` from the array `FLSOPT`, at offset value defined by `$PARTOFFSET` of `$DDIMGPATH`, pipe the output to a csv file named according to the `$i` value of the array `FLSOUTPUTCSV`.

OSDF

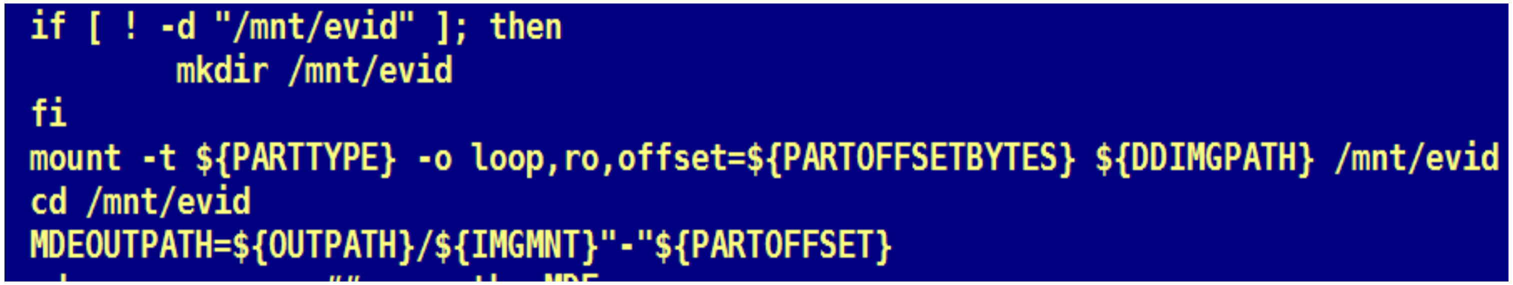
MDE: PART THREE: `mdeprocess`

- Everything to this point has been done using a dd file (or a virtual dd file, by virtue of affuse or mount_ewf).
- `$PARTTYPE` and `$PARTOFFSETBYTES` are previously assigned in the flsexam function:



```
if [[ $PARTTYPE = "NTFS" ]];
then
        PARTTYPE="ntfs";
else
        PARTTYPE="vfat";
fi
PARTOFFSETBYTES=`echo "${PARTOFFSET}*512"|bc`
```

If a mount point, /mnt/evid doesn't exist, the script will create one.



```
if [ ! -d "/mnt/evid" ]; then
        mkdir /mnt/evid
fi
mount -t ${PARTTYPE} -o loop,ro,offset=${PARTOFFSETBYTES} ${DDIMGPATH} /mnt/evid
cd /mnt/evid
MDEOUTPATH=${OUTPATH}/${IMGMNT}"-"${PARTOFFSET}
```

OSDF

MDE: PART THREE: `mdeprocess`

- Once the partition is mounted to /mnt/evid, the script moves to that directory, and if requested, conducts a registry analysis.   If registry analysis is not requested, the **`mdeprocess`** function begins.

```
FINDCNT=`${SELECTEXTRACT} -iname "$INAMEVAR" | tee \
  >(xargs -d '\n' md5deep -rsczl >> /tmp/srcmd5.csv)
  \ |tee >cpio -dump ${MDEOUTPATH}|wc -l`
```

`SELECTEXTRACT:` Variations of the find command, incorporating the date scope determined at start.

`INAMEVAR:` file extensions drawn from arrays.

MDE: PART FOUR: `mdevalidate`

- generates a sorted list of **unique** md5 values from the srcmd5.csv file written prior to cpio streaming;
- generates a second sorted list of **unique** md5 values of everything streamed through `cpio` to the destination location;
- Lastly, it compares the two lists with a `diff` command piped though `wc;`
- If the value of wc is **ZERO**, then the two lists match, and by inference, the destination files are duplicates of the source files.

OSDF

MDE: PART FIVE: `deletedrecover`

`fls` (to create a list of deleted files)

`icat` (to carve them out)

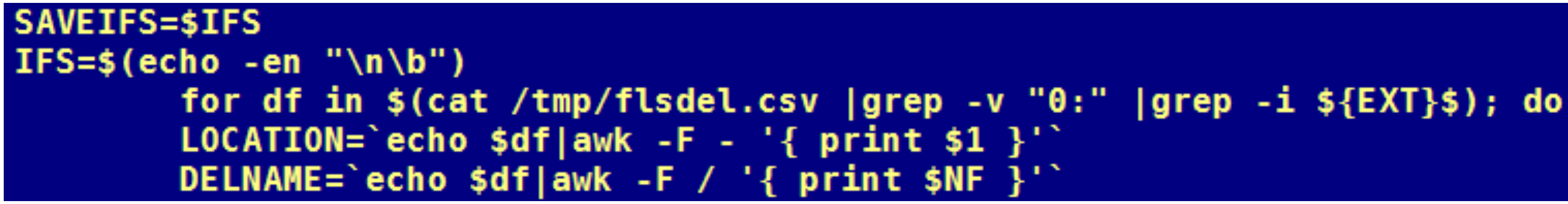
- PROBLEM:
- icat names carved file by inode/MFT number
- The file name is much more meaningful

```
fls -Frd -o ${PARTOFFSET} ${DDIMGPATH} |grep -v ^"d"|
  cut -c 7- > /tmp/flsdel.csv
```

The temporarily created file csvdel.csv contains both the inode number (needed by `icat`) and the file name (needed by people).

(Latest version of sleuthkit includes `fcat`, which recovers files by filename instead of inode (MFT) entry. This will be written into the MDE at a later date.)

MDE: PART FIVE: `deletedrecover`



```
SAVEIFS=$IFS
IFS=$(echo -en "\n\b")
       for df in $(cat /tmp/flsdel.csv |grep -v "0:" |grep -i ${EXT}$); do
       LOCATION=`echo $df|awk -F - '{ print $1 }'`
       DELNAME=`echo $df|awk -F / '{ print $NF }'`
```

Since unix uses blank spaces as separators, this posed a problem when dealing with Windows named files (which use spaces). To get around this, I temporarily set IFS to only use newline \n or break \b as separators.

- Define `$LOCATION` by the first column of the flsdel.csv file and `$DELNAME` by the last column, using two different awk commands and different file separators.

```
icat -f ${PARTTYPE} -o ${PARTOFFSET} ${DDIMGPATH} \
 ${LOCATION} > ./${FILETYPELIST[$m]}/${LOCATION}- \
 ${DELNAME}
```

- Run the icat command on the dd file at `$DDIMGPATH` with partition type `$PARTTYPE`, at offset `$PARTOFFSET` to locate inode/MFT entry `$LOCATION`, and write the output to a folder named `$FILETYPELIST`, file named by <inode/MFT number>-<filename>

MDE: PART SIX: Reporting

- Once deleted file recovery is conducted, the script generates a Media Data Extraction Report.
- This report is a **single** page list of how many files were recovered of each file type, and uniquely documents each processed partition of an image.
- Also records the examiner, and the date and time of processing.
- Saved with all other generated case notes
- Serves as a preliminary examination report.
- Fully automates the mounting, extraction, documentation and reporting process.

OSDF

ADDITIONAL FEATURES

Long paths:

Path lengths for each recovered file is assessed.

If it exceeds 255 characters, then the entire path is appended to a text file; the file is moved to a much shorter path "long path" folder.

File metadata:

Every recovered file is evaluated with the linux command **`exiftool`**. All identified metadata is captured to a Metadata report specific to the partition.

OSDF

ADDITIONAL FEATURES

Virus scanning:

- All recovered files are scanned using `clamav`.

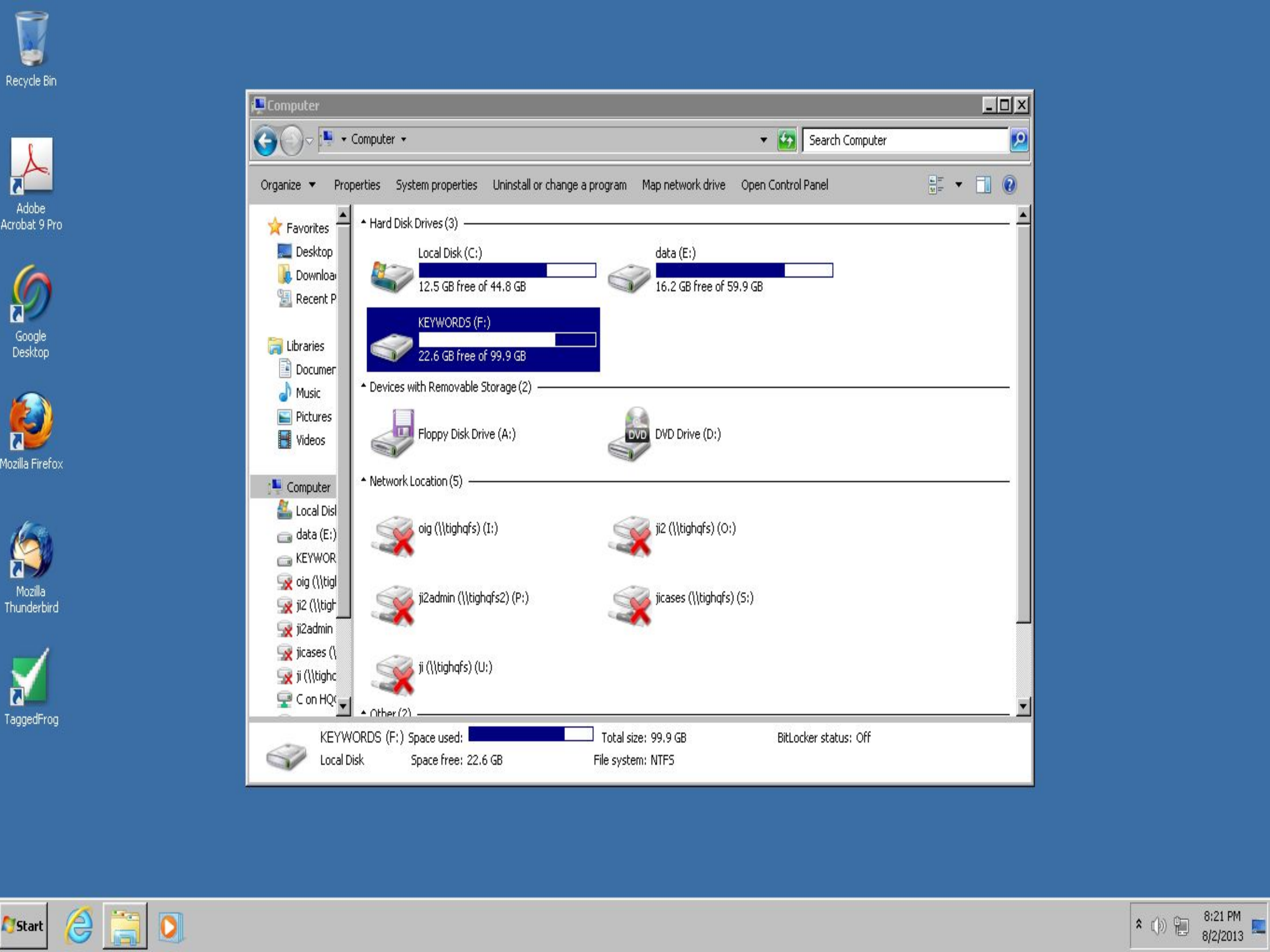
Virus definitions are maintained on a removable device, the location is requested in a virus scan is conducted.

Registry Analysis:

If requested, the partitions are screened for registry files (NTUSER.DAT). If found, Harlan Carvey's `regripper` is invoked, and a registry analysis report is generated for the partition.

OUTPUT

- The resulting output is provided to the Case Agent by means of a virtual Windows 7 environment, and is indexed by the operating system. This means that the Case Agent can browse the output, identify files of potential interest, and alert the examiner to conduct a full forensic examination on just those files.


Computer

Search Computer

Organize Properties System properties Uninstall or change a program Map network drive Open Control Panel

Hard Disk Drives (3)

Local Disk (C:)
12.5 GB free of 44.8 GB

data (E:)
16.2 GB free of 59.9 GB

KEYWORDS (F:)
22.6 GB free of 99.9 GB

Devices with Removable Storage (2)

Floppy Disk Drive (A:)

DVD Drive (D:)

Network Location (5)

oig (\\tighqfs) (I:)

ji2 (\\tighqfs) (O:)

ji2admin (\\tighqfs2) (P:)

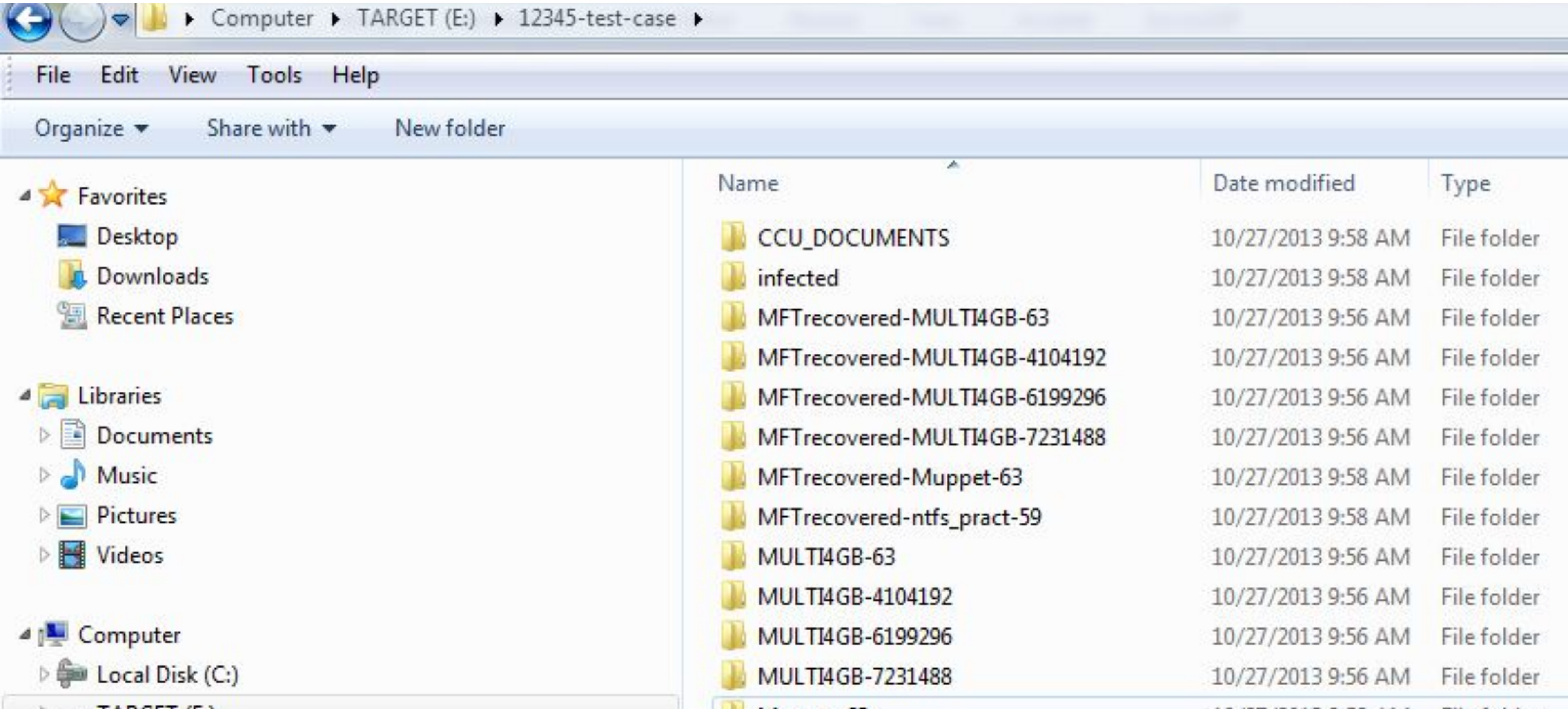
jicases (\\tighqfs) (S:)

ji (\\tighqfs) (U:)

Other (2)

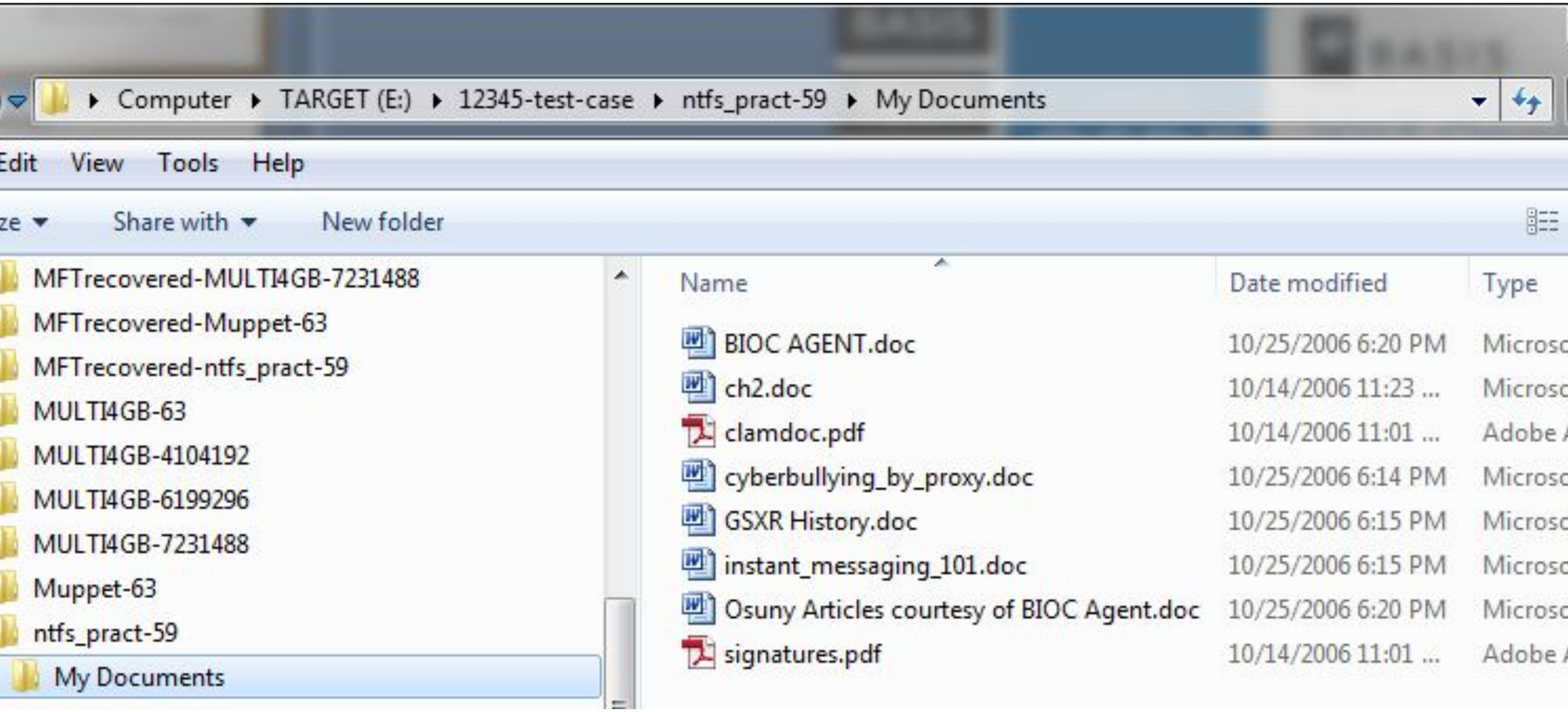
KEYWORDS (F:) Space used: Total size: 99.9 GB BitLocker status: Off
Local Disk Space free: 22.6 GB File system: NTFS

8:21 PM
8/2/2013


Computer ▸ TARGET (E:) ▸ 12345-test-case ▸
File Edit View Tools Help
Organize ▾ Share with ▾ New folder
Favorites
Desktop
Downloads
Recent Places
Libraries
Documents
Music
Pictures
Videos
Computer
Local Disk (C:)

Name	Date modified	Type
CCU_DOCUMENTS	10/27/2013 9:58 AM	File folder
infected	10/27/2013 9:58 AM	File folder
MFTrecovered-MULTI4GB-63	10/27/2013 9:56 AM	File folder
MFTrecovered-MULTI4GB-4104192	10/27/2013 9:56 AM	File folder
MFTrecovered-MULTI4GB-6199296	10/27/2013 9:56 AM	File folder
MFTrecovered-MULTI4GB-7231488	10/27/2013 9:56 AM	File folder
MFTrecovered-Muppet-63	10/27/2013 9:58 AM	File folder
MFTrecovered-ntfs_pract-59	10/27/2013 9:58 AM	File folder
MULTI4GB-63	10/27/2013 9:56 AM	File folder
MULTI4GB-4104192	10/27/2013 9:56 AM	File folder
MULTI4GB-6199296	10/27/2013 9:56 AM	File folder
MULTI4GB-7231488	10/27/2013 9:56 AM	File folder

OUTPUT REPORTS

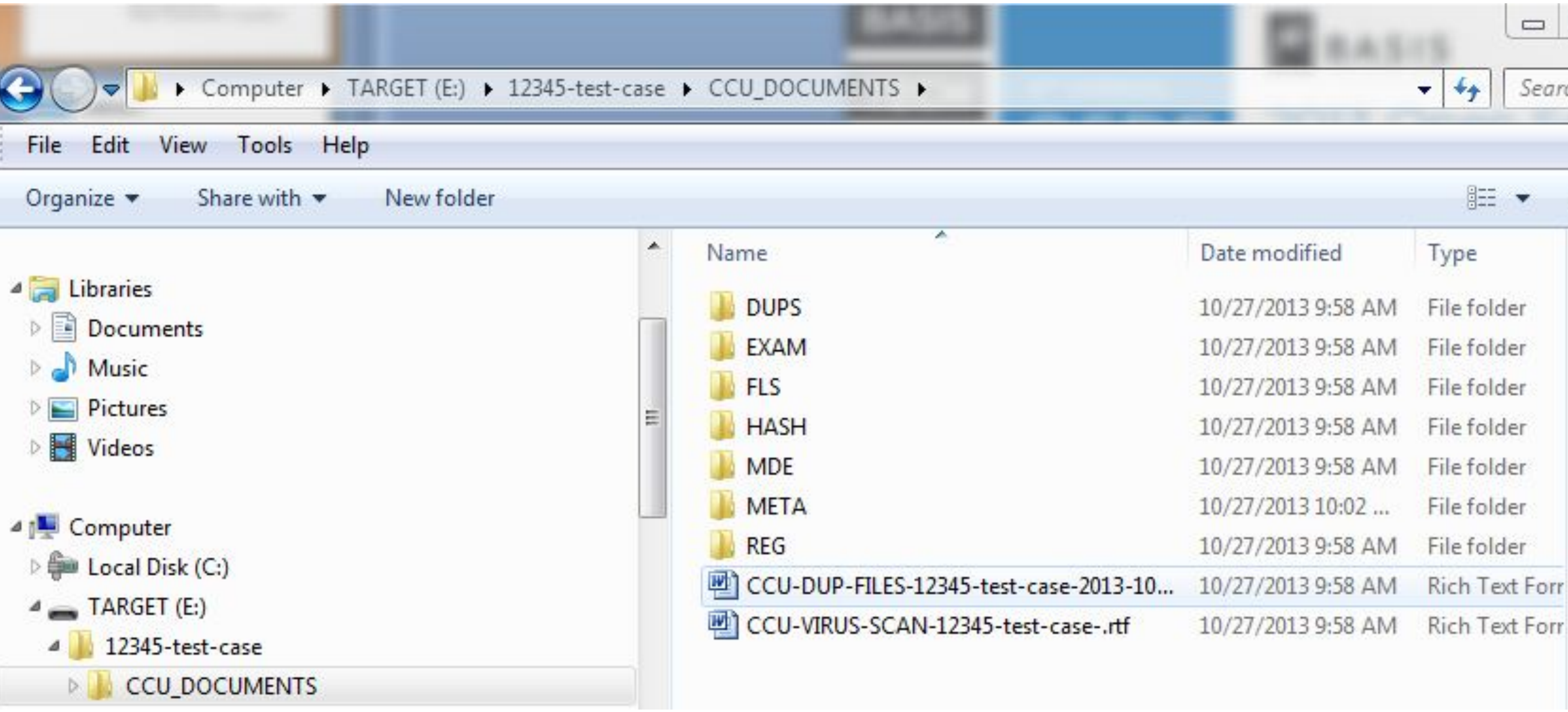


Computer ▸ TARGET (E:) ▸ 12345-test-case ▸ ntfs_pract-59 ▸ My Documents
Edit View Tools Help
ze ▾ Share with ▾ New folder

MFTrecovered-MULTI4GB-7231488
MFTrecovered-Muppet-63
MFTrecovered-ntfs_pract-59
MULTI4GB-63
MULTI4GB-4104192
MULTI4GB-6199296
MULTI4GB-7231488
Muppet-63
ntfs_pract-59
My Documents

Name	Date modified	Type
BIOC AGENT.doc	10/25/2006 6:20 PM	Microsc
ch2.doc	10/14/2006 11:23 ...	Microsc
clamdoc.pdf	10/14/2006 11:01 ...	Adobe
cyberbullying_by_proxy.doc	10/25/2006 6:14 PM	Microsc
GSXR History.doc	10/25/2006 6:15 PM	Microsc
instant_messaging_101.doc	10/25/2006 6:15 PM	Microsc
Osuny Articles courtesy of BIOC Agent.doc	10/25/2006 6:20 PM	Microsc
signatures.pdf	10/14/2006 11:01 ...	Adobe

OUTPUT FILES – NTFS_PRACT-59

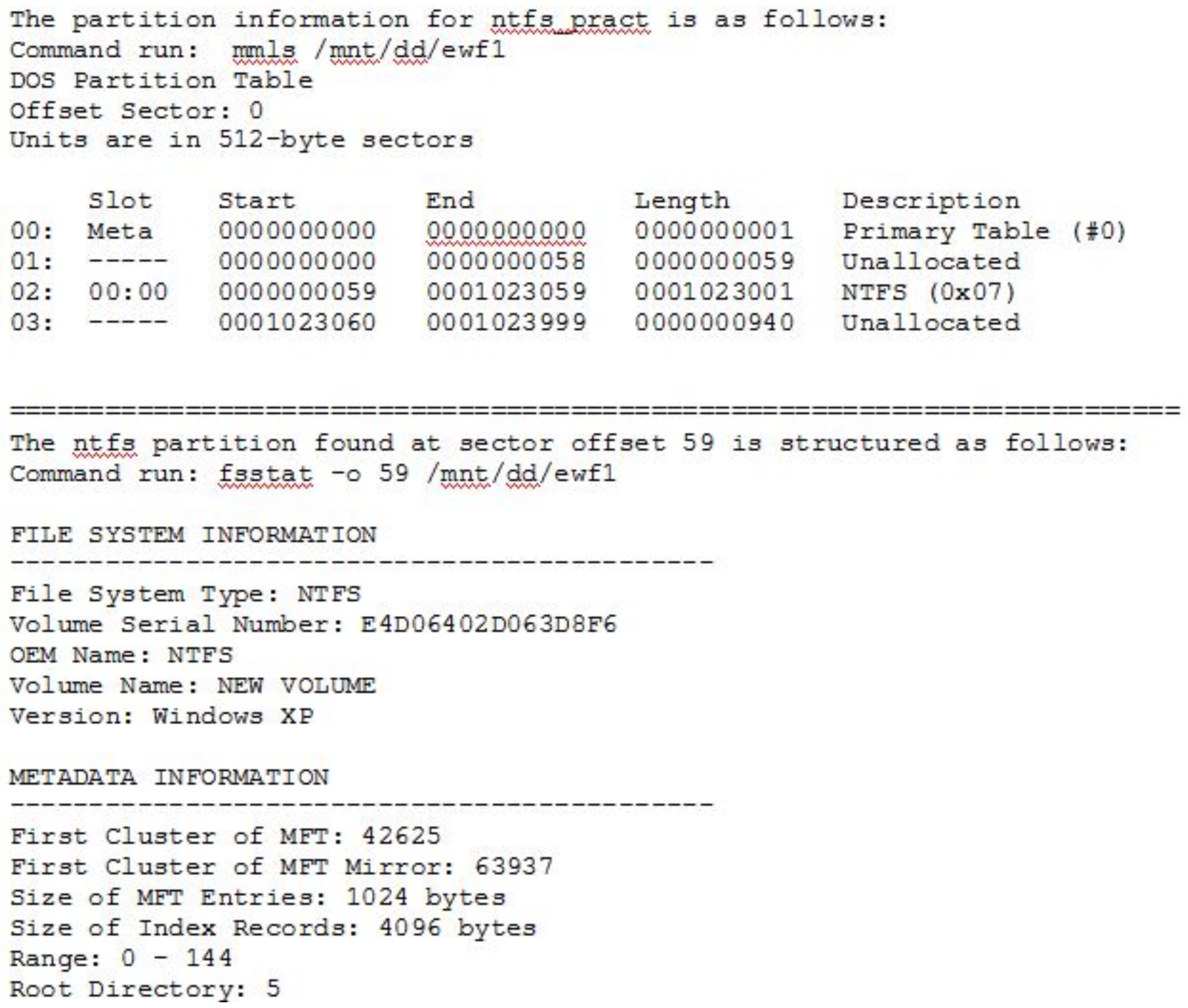


Computer ▸ TARGET (E:) ▸ 12345-test-case ▸ CCU_DOCUMENTS ▸
File Edit View Tools Help
Organize ▾ Share with ▾ New folder
Libraries
Documents
Music
Pictures
Videos
Computer
Local Disk (C:)
TARGET (E:)
12345-test-case
CCU_DOCUMENTS

Name	Date modified	Type
DUPS	10/27/2013 9:58 AM	File folder
EXAM	10/27/2013 9:58 AM	File folder
FLS	10/27/2013 9:58 AM	File folder
HASH	10/27/2013 9:58 AM	File folder
MDE	10/27/2013 9:58 AM	File folder
META	10/27/2013 10:02 ...	File folder
REG	10/27/2013 9:58 AM	File folder
CCU-DUP-FILES-12345-test-case-2013-10...	10/27/2013 9:58 AM	Rich Text Forr
CCU-VIRUS-SCAN-12345-test-case-.rtf	10/27/2013 9:58 AM	Rich Text Forr

CCU DOCUMENTS

```
========================================================================
           MEDIA DATA EXTRACTION REPORT
           OSFC AGENCY
           OSFC Organization

     CASE NUMBER:                  12345-test-case
     CASE TITLE:                   LINUXLEO

     EXAMINER:                     Tobin Craig
     DATA EXTRACTION DATE:         Sun Oct 27 09:54:04 EDT 2013

========================================================================
Data was extracted from the system identified as
ntfs_pract


========================================================================

8 files were recovered, with the following extensions: doc docx pdf rtf
0 files were recovered, with the following extensions: xls xlsx csv
0 files were recovered, with the following extensions: pst ost eml msg
mbx dbx mbox msf nsf
0 files were recovered, with the following extensions: ppt pptx
0 files were recovered, with the following extensions: mdb mdbx accdb

========================================================================

VERIFICATION OF EXTRACTED ALLOCATED FILES
All extracted files are verified replicas of those found in
/media/root/SOURCE/testcase/images//ntfs_pract, offset 59,
verified by md5 hash comparison.

========================================================================


A total of 8 files were recovered from
/media/root/SOURCE/testcase/images//ntfs_pract, offset 59,
and written to /media/root/TARGET/12345-test-case/ntfs_pract-59.

A total of 1 deleted files were also recovered from this partition.

The deleted files matching specified criteria have been successfully
validated

Extraction conducted by Tobin Craig, on Sun Oct 27 09:58:30 EDT 2013.

Media Data Extraction Script - VERSION 8.2 - SEPT 2013
```



```
The partition information for ntfs_pract is as follows:
Command run:  mmls /mnt/dd/ewf1
DOS Partition Table
Offset Sector: 0
Units are in 512-byte sectors

     Slot    Start        End          Length       Description
00:  Meta    0000000000   0000000000   0000000001   Primary Table (#0)
01:  -----   0000000000   0000000058   0000000059   Unallocated
02:  00:00   0000000059   0001023059   0001023001   NTFS (0x07)
03:  -----   0001023060   0001023999   0000000940   Unallocated


==========================================================================
The ntfs partition found at sector offset 59 is structured as follows:
Command run: fsstat -o 59 /mnt/dd/ewf1

FILE SYSTEM INFORMATION
--------------------------------------------
File System Type: NTFS
Volume Serial Number: E4D06402D063D8F6
OEM Name: NTFS
Volume Name: NEW VOLUME
Version: Windows XP

METADATA INFORMATION
--------------------------------------------
First Cluster of MFT: 42625
First Cluster of MFT Mirror: 63937
Size of MFT Entries: 1024 bytes
Size of Index Records: 4096 bytes
Range: 0 - 144
Root Directory: 5
```

http://code.google.com/p/forensic-data-extraction/

Are there better solutions? Absolutely!

I am not a programmer…remember?
Priced competitively,

All help, advice, improvements gratefully welcome and received!

Tobin Craig, MRSC, CISSP, CCE, CFCE

Lab Chief, US DOT-OIG, CCU
Email: tobin.craig@oig.dot.gov