RAM Forensic Framework

[1]Vineet Mishra [2]Samrat Sutar[3] Pallavi Nigam
[1,2,3]Student (Computer Engineering)
Fr. C. R. Institute of Technology, Vashi.
NaviMumbai,India

[4]Shweta Tripathi
[4]Assistant Professor
Fr. C. R. Institute of Technology, Vashi
NaviMumbai,India

***Abstract*—Computer Forensics, being an integral part of the investigations pertaining to all crimes, has made its existence felt in law enforcement and the business community.The confidential data from the victim's machine may be compromised by some malicious program running in the RAM or through physical access like pen drive, hard disks, etc. Analyzing different processes running in RAM would be useful evidence in the world of computer forensics. In this paper, we have studiedthe importance of RAM Forensics. A framework has been proposed which will be useful in analyzing the various processes running on the RAM and hence identifying the malicious activities.By analyzing the logs, we would also identify the insertion of external media without the consent of the owner.**

Keywords- Computer Forensics, RAM, framework, hard disks

I. INTRODUCTION

Digital evidence collection is being driven by the rapidly changing threats in computing environment. The use of removable media such as a USB stick for installing the applications and are then virtualizing it in RAM without a trace on the hard disk is suspicious. Inability of operating system to detect the Root kits hiding withinprocesses is also a problem. Malware resides completely in the RAM with no trace of existence on the hard disk. Encrypted files or partitions are the areas of the hard drive which are used to hide evidence. Many popular web browsers allow their user to cover their tracks like log files of user activity are created but deleted when the browser is closed.

Significant number of locations and layers are used by computers to store a great amount of information. The two most commonly thought data repositories are hard disk storage and RAM.Apart from these, the useful and important data can also hide outside the system if it is connected to the network.There is also a fact that all the data is volatile. With time, the usefulness of the informationmay reduce, thereby decreasing the ability to recall or validate the data. Also, it is extremely difficult to verify by just lookingwhether the stored information has been changed. Thus, some type of data is generally more persistent, or long-lasting, than others. However, backup tapes can be relied upon to remain unchanged longer than things in RAM as it is less volatile. This hierarchy is called the Order Of Volatility or OOV. The order of volatility shows which data will be lost first. The position of RAM in OOV is shown below.

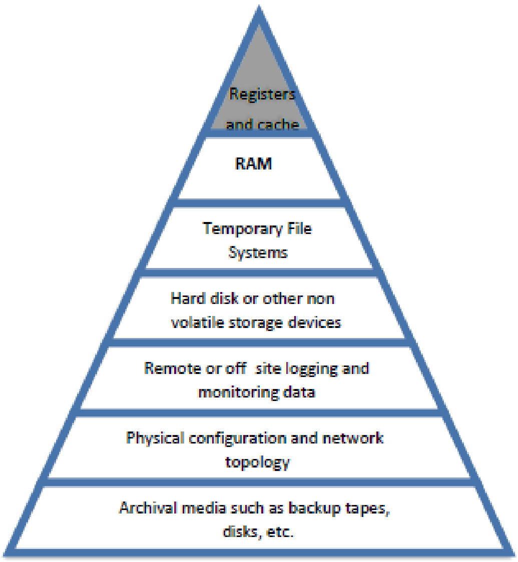

Registers and cache
RAM
Temporary File Systems
Hard disk or other non volatile storage devices
Remote or off site logging and monitoring data
Physical configuration and network topology
Archival media such as backup tapes, disks, etc.

Fig. 1. RAM in OOV

Random Access Memory (RAM) is a form of computer data storage. It is volatile, meaning it can easily be flushed and is not used for long term storage. Hardware devices or software based applications can be used to retrieve the data stored in the memory. Any information or data being used by any program passes through the RAM at that timemaking RAM very important for conducting Forensic analysis.

Hence, according to a computer forensic analyst there is a blob of evidence in RAM. A systematic and planned approach is needed to retrieve the information otherwise there are many chances of losing the data or tampering the evidence available.

II. LITERATURE SURVEY

A. Digital forensics

The complete definition of computer forensics is as follows: The use of scientifically derived and proven methods towards the preservation, collection, validation, identification, analysis, interpretation, documentation and presentation of digital evidence derived from digital sources for the purpose of facilitating or furthering the reconstruction of events
found to be criminal[3].

B. RAM forensics

B.1 RAM

Live forensics provides the collection of digital evidence in an order of collection that is actually based on the life expectancy of the evidence in question. The volatile data, that is, the RAM residing in the computer contains the most important evidence to be collected in digital evidence collection step. For an analyst, RAM is just a vast collection of data that needs to be exploited in order to extract important information for analysis.

B.2 Need for RAM analysis

RAM analysis is an important part of computer forensics as it helps the investigators in finding out what all happened on the machine right before the crime was committed. As all the running processes in the system pass through the RAM, it is important to retrieve it before the computer is turned off. Any discrepancy in the processes present in the RAM or any kind of unwanted external communication, if detected is a step closer to finding the culprit.

It is very important for the forensic team to understand the importance of the data present in the RAM and extract it at the right moment before it is lost.

B.3Information obtained from RAM analysis

RAM contains a wealth of volatile data that can be extracted and which can aid the field of forensics to a great extent. The following volatile data is obtained from RAM analysis [12]:

a) Past and current network connections:

This gives the remote IP address and port number used in network connections. This information is useful for detecting a computer intrusion, identifying the IP address the malware is communicating with, the source of criminal activity, or where the information is being transferred.

b) List of running processes at the time of RAM capture:

A list of active processes running when the RAM was acquired providesan idea of how the system was being used. The Task Manager only provides an apparent knowledge of what is running on a system. What is not revealed is a process running such as a rootkit, that is, a hidden Trojan used to exfiltrate data or allows remote access, or the key logger that is siphoning all important user data.

c) User names and passwords:

Users input their user name and password to access an account many times for authentication of e-mail, social networking accounts, or their home's wireless access point. All of this data passes through the RAM.

d) Loaded Dynamically Linked Libraries (DLL):

A list of all the DLLs associated with a running process helps in identifying a malicious DLL that might have injected itself into a process.

e) Contents of an open window:

This includes any keystrokes into Webmail, an e-mail client, values into a form field, and an IM chat client and chat sessions, including participants.

f) Open registry keys for a process:

It is crucial to be able to identify registry keys associated with a malicious process. By being able to associate open registry keys to a certain process, an analyst could tie functionality to that process, such as networking capabilities, encryption, or being able to associate the secure identifier (SID) to the user account who started the process. It is also important to identify the method used by the malware to sustain reboot. This information can be identified from the relationships between a process and its registry keys. One notable thing is that the registry values will be those that are ―open‖ at the time of the RAM acquisition. However, the registry key that was responsible for the malware surviving a reboot could still be listed in RAM and could be found by dumping the address space for that process.

g) Open files for a process:

Being able to list open files associated with a process would reveal any open files that are currently being used by the identified malicious process. This is helpful in identifying a resident file that is logging keystrokes, or user names and passwords. This is also important in identifying a configuration file used by a malicious process, even if it is encrypted on disk. This file could then be found in memory and its contents read.

h) Unpacked/decrypted versions of a program:

One of the most valuable contributions that memory forensics can provide to an analyst is the ability to carve out an identified malicious process out of memory. If a malicious file or binary is encrypted on a hard drive the analyst would have a very hard time decrypting the file in order to obtain its contents. However, every file that is read or is executed will have to unpack or decrypt itself to run. By following the process below, the malicious file could be identified, carved out of memory, and analyzed through static analysis or by scanning with an anti-virus tool.

i) Memory resident malware:

Memory resident malware are becoming more prevalent. There is malware in the wild that will only reside in a system's memory, leaving no footprints on the system's hard drive. Any data collected could also just be stored in memory before being exfiltrated to a remote system.

B.4 Standard procedure for RAM analysis

The official ACPO Guidelines recommend the following standard procedure for capturing a memory dump during RAM Analysis [14]: Perform a risk assessment of the situation so as find out whether it evidentially required and safe to perform volatile data collection. If so, install volatile data capture device e.g. USB Flash Drive, USB hard drive etc. Run the volatile data collection script. Once complete, stop the device particularly important for USB devices which if removed before proper shutdown can lose information. Remove the device. Verify the data output on a separate forensic investigation machine other than the suspect system. Immediately follow with standard power-off procedure.

B.5 Tools and techniques for analyzing and capturing memory dumps

A range of tools and methods are available to capture memory dumps. From the forensic perspective, there are certain requirements that any such tool must strictly conform to. In no particular order, the list of essential requirements goes like this [14].

1. Kernel-mode operation
2. Smallest footprint possible
3. Portability
4. Read-only access

Kernel-mode operation:It is essential for a forensic memory capturing tool. With many applications proactively protecting their memory sets against dumping, running a memory acquisition tool that operates in user-mode is simple suicide. At best, such tools will read zeroes or random data instead of the actual information. In a worst-case scenario, a proactive anti-debugging protection will take immediate measures to effectively destroy protected information, then lock up and/or reboot the computer, making any further analysis impossible. To avoid this from happening, investigators must use a proper memory acquisition tool running in the system‘s most privileged kernel mode. Notably, current versions (as of April 24, 2014) of two popular forensic memory dumping tools, AccessData FTK Imager and PMDump, run as user-mode applications and are unable to overcome protection imposed by anti-debugging systems operating in a privileged kernel mode.

Smaller footprint: Smaller footprint is left by a memory acquisition tool. Using a tool like that already leaves traces and potentially destroys certain evidence. The less of this technique is used, the better it is.

Portable: Memory dumping tools must be ready to run from an investigator-provided device like USB flash drive or a network location. Tools requiring installation are inadmissible because they will further take up RAM while installation, thus overriding potential evidences. Finally, any sane forensic tool would never write anything onto the disk of the computer being analyzed, will not create or modify Registry values, etc.

III. PROPOSED FRAMEWORK

The proposed framework consists of various tools that will help in finding out the culprit among the given suspects once we have the access to the victim’s machine and the machine of different suspects.

For designing the framework we have considered the following case study.

A victim ’V’ received an anonymous message which claims that a culprit ‘C’ has access to V’s confidential data. C demands money in exchange for not making the data public or returning of the confidential data. We have a list of suspects S_1, S_2, S_3 …….S_n and also the access to suspects’ machine. We are assuming that the confidential data is/was present in one of the suspect’s machine. The framework willbe useful in performing following functions:

1. It will help the investigators in finding how the confidential data was compromised.
2. Itwill check all the suspects’ machine for deleted data and gather evidence regarding accessing of the confidential data.
3. It will check the metadata of the file to support/refute the given testaments of the suspects.
4. It will perform additional functions to identify the culprit.

Hence, it will be useful in gathering important artifacts from the machines to aid the investigators in finding out the culprit ‘C’ (if present) among the suspects S_1, S_2, S_3 …….S_n.

We have identified seven stages which define the overall process. Each stage has its own layout and methodology. The block diagram of the process is explained below.

Stage I: Prerequisites

This stage elaborates the permissions to work on case and also on the victim’s and the suspect’s machine to do the analysis of the data present on it.

Stage II: Collection of Evidence

In this stage, all the necessary data from the victim’s and suspects machine is collected and stored on a separate machine to have a confiscated copy of the original data that was present on the when the crime had occurred.

Stage III: Check for Integrity

The integrity check of the data will indicate whether the data was tampered and also who accessed it last and how and when and it was modified. Integrity check is performed to create a reference point of the data before the analysis is done, this later can be used to check if any data was tampered during the process of investigation.

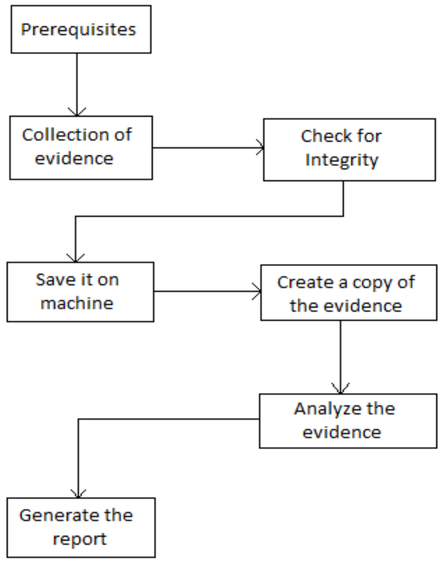


Fig. 2. Block Diagram of the various stages of the case

Stage IV: Save it on machine

After the integrity check is done, it is saved on the investigator's machine or a separate repository for evidence storage as the evidence of the crime scene.

Stage V: Create a copy of the data

A copy of the data that is saved in the previous stage is created on which the various tools for the analysis are applied. The analysis will be performed on the copy and the original data will be left untouched to provide testament in the court [1].

Stage VI: Analyze the evidence

Toolkit shall perform various operations in order to find patterns and discrepancies which can be helpful in the case [2].

Stage VII: Generate the report

Based on the analysis of the evidence collected in the previous stages, a report will be generated that will indicate whether the culprit is present among the suspects or not. The report shall also present what all proofs were found during the analysis phase.

IV. IMPLEMENTATION

The toolkit must be platform independent to provide the flexibility and versatility to the investigator using it. We will be using Python programming language. We have commenced with the formulation of the toolkit with our main focus on integration of Dead Box analysis and Live Box analysis.

A. Live BoxAnalysis

Within the Live Box analysis, we have used a Volatility Framework derivative for gathering important artifacts from memory dumps. Information like Operating system footprinting, processes running, external communication, list of account users logged in during capture, registry information, etc have be successfully extracted using the toolkit. The results are stored in a file so that it can be viewed as often as required. Volatility framework offers a bash based service where the inputs and output are presented on the terminal itself. A separate script was prepared to execute the Volatility based scans on the terminal without physically invoking it and also the results were redirected to a file from the terminal.

B. Dead Box Analysis

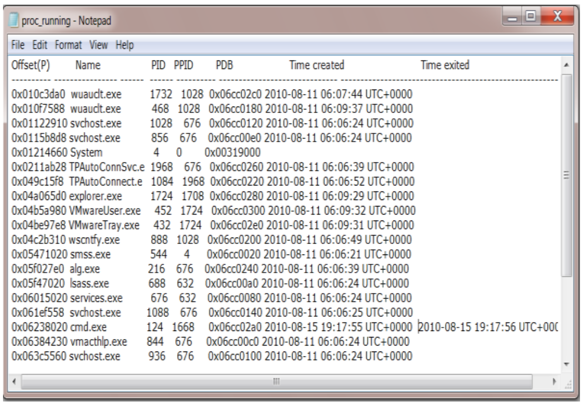
Within Dead Box analysis, we have partially implemented the modules of metadata analysis and file recovery. Metadata analysis currently generates the name of the author, date of creation, date of modification and the version of editor used to create it. A third part library called 'Pypdf' was used for this purpose. For file recovery, we are in a rudimentary stage where we are able to detect deleted files in the recycle bin. This is being achieved by physically locating the 'Recycler' directory in the windows environment and checking for any files present in this directory. We aim to detect and recover deleted files from the entire secondary storage media.

C. Results

To perform the forensics, attack is required. Hence, we have taken an infected RAM image. This system is infected by Zeus malware. It is a Trojan horse computer malware that runs on versions of the Microsoft Windows . Very often it is used to steal banking information by man-in-the-browser keystroke logging and form grabbing. It is also used to install the CryptoLocker ransomware and is spread mainly through drive-by downloads and phishing schemes.

Rewriting the code of framework of the tool Volatility, we have executed it on the above image. The result is obtained in the form of reports. The snapshots of the reports are explained below.

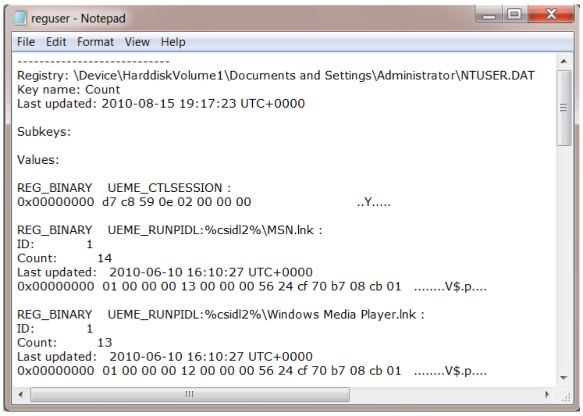
The information about the processes running along with process Id and offset are generated in this report which gives an idea about the processes running during RAM capture.



```
Offset(P)     Name              PID  PPID    PDB           Time created                 Time exited
---------- ---------------- ------ ------ ---------- ---------------------------- ----------------------------
0x010c3da0  wuauclt.exe       1732  1028  0x06cc02c0 2010-08-11 06:07:44 UTC+0000
0x010f7588  wuauclt.exe        468  1028  0x06cc0180 2010-08-11 06:09:37 UTC+0000
0x01122910 svchost.exe        1028   676  0x06cc0120 2010-08-11 06:06:24 UTC+0000
0x0115b8d8 svchost.exe         856   676  0x06cc00e0 2010-08-11 06:06:24 UTC+0000
0x01214660 System                4     0  0x00319000
0x0211ab28 TPAutoConnSvc.e    1968   676  0x06cc0260 2010-08-11 06:06:39 UTC+0000
0x049c15f8  TPAutoConnect.e   1084  1968  0x06cc0220 2010-08-11 06:06:52 UTC+0000
0x04a065d0 explorer.exe       1724  1708  0x06cc0280 2010-08-11 06:09:29 UTC+0000
0x04b5a980 VMwareUser.exe      452  1724  0x06cc0300 2010-08-11 06:09:32 UTC+0000
0x04be97e8 VMwareTray.exe      432  1724  0x06cc02e0 2010-08-11 06:09:31 UTC+0000
0x04c2b310 wscntfy.exe         888  1028  0x06cc0200 2010-08-11 06:06:49 UTC+0000
0x05471020 smss.exe            544     4  0x06cc0020 2010-08-11 06:06:21 UTC+0000
0x05f027e0  alg.exe            216   676  0x06cc0240 2010-08-11 06:06:39 UTC+0000
0x05f47020  lsass.exe          688   632  0x06cc00a0 2010-08-11 06:06:24 UTC+0000
0x06015020 services.exe        676   632  0x06cc0080 2010-08-11 06:06:24 UTC+0000
0x061ef558  svchost.exe       1088   676  0x06cc0140 2010-08-11 06:06:25 UTC+0000
0x06238020 cmd.exe             124  1668  0x06cc02a0 2010-08-15 19:17:55 UTC+0000 2010-08-15 19:17:56 UTC+00(
0x06384230 vmacthlp.exe        844   676  0x06cc00c0 2010-08-11 06:06:24 UTC+0000
0x063c5560 svchost.exe         936   676  0x06cc0100 2010-08-11 06:06:24 UTC+0000
```

Fig. 3. Processes running during the capture

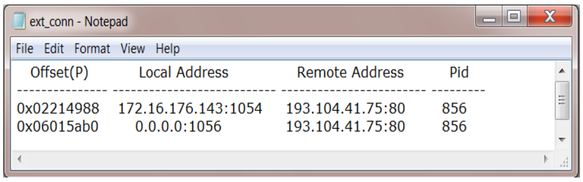
All the processes communicating with the registry are listed down in this report along with the location of the file and register binary. This binary can be used to check with the database of virus signatures to crosscheck the existence of a virus in the file.



```
----------------------------
Registry: \Device\HarddiskVolume1\Documents and Settings\Administrator\NTUSER.DAT
Key name: Count
Last updated: 2010-08-15 19:17:23 UTC+0000

Subkeys:

Values:

REG_BINARY    UEME_CTLSESSION :
0x00000000  d7 c8 59 0e 02 00 00 00                           ..Y.....

REG_BINARY    UEME_RUNPIDL:%csidl2%\MSN.lnk :
ID:            1
Count:         14
Last updated:   2010-06-10 16:10:27 UTC+0000
0x00000000  01 00 00 00 13 00 00 00 56 24 cf 70 b7 08 cb 01   ........V$.p....

REG_BINARY    UEME_RUNPIDL:%csidl2%\Windows Media Player.lnk :
ID:            1
Count:         13
Last updated:   2010-06-10 16:10:27 UTC+0000
0x00000000  01 00 00 00 12 00 00 00 56 24 cf 70 b7 08 cb 01   ........V$.p....
```

Fig. 4. Registry activity during the RAM capture

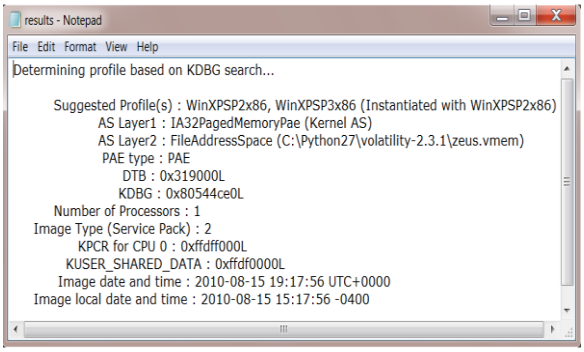
Information on external connection of the processes during the Ram capture is listed in this report. The IP address of the target and the process ID of the process interacting with it is present in this report.



```
  Offset(P)       Local Address          Remote Address        Pid
------------ ------------------------ -------------------- --------
0x02214988   172.16.176.143:1054    193.104.41.75:80     856
0x06015ab0        0.0.0.0:1056      193.104.41.75:80     856
```

Fig. 5. External connections of the processes during the RAM capture

This report gives the basic information about the operating system of the user whose RAM was captured.



```
Determining profile based on KDBG search...

          Suggested Profile(s) : WinXPSP2x86, WinXPSP3x86 (Instantiated with WinXPSP2x86)
                     AS Layer1 : IA32PagedMemoryPae (Kernel AS)
                     AS Layer2 : FileAddressSpace (C:\Python27\volatility-2.3.1\zeus.vmem)
                      PAE type : PAE
                           DTB : 0x319000L
                          KDBG : 0x80544ce0L
          Number of Processors : 1
     Image Type (Service Pack) : 2
                KPCR for CPU 0 : 0xffdff000L
             KUSER_SHARED_DATA : 0xffdf0000L
           Image date and time : 2010-08-15 19:17:56 UTC+0000
     Image local date and time : 2010-08-15 15:17:56 -0400
```

Fig. 6. System footprint

V. CONCLUSION

Since the data is precious, the loss of confidentiality, integrity or accessibility of the data is high threat to the cyber world. Capturing the footprints of the attacker from a volatile device is a challenge. By extracting the evidence from RAM and analyzing them we have contributed towards the cyber investigation process.

REFERENCES

[1] http://www.uscert.gov/sites/default/files/publications/forensics.pdf

[2] http://www.evidencemagazine.com/index.php?option=com_content&task=view&id=116&Itemid=49

[3] Nikkel, B. (2006) the role of digital forensic with a corporateorganisationwww.digitalforensics.ch/nikkel/06a.pdf

[4]]http://digitalforensics.sans.org/blog/2009/09/12/best-practices-in-digital-evidencecollection [5]http://www.wisegeek.com/what-is-an-evidence-log.htm

[5] http://www.wisegeek.com/what-is-an-evidence-log.htm

[6] http://www.sans.org/reading-room/whitepapers/incident/computer-forensics-weve-incident-investigate-652

[7] http://www.forensicfocus.com/enhanced-digital-investigation-model The Enhanced Digital Investigation Process Model Venansius Baryamureeba and Florence Tushabe barya@ics.mak.ac.ug, tushabe@ics.mak.ac.ug Institute of Computer Science, Makerere University P.O.Box 7062, Kampala Uganda www.makerere.ac.ug/ics May 27, 2004

[8] http://www.porcupine.org/forensics/forensic-discovery/appendixB.html

[9] [Page 4], Ref: 2.1 Order of Volatility, http://www.ietf.org/rfc/rfc3227.txt

[10] Privacy Protection and Computer Forensics, Michael A. Caloyannides, second edition

[11] Collecting Evidence from a Running Computer: A Technical and Legal Primer for the Justice Community- By Todd G. Shipley, CFE, CFCE and Henry R. Reeve, Esq.

[12] http://www.dfinews.com/articles/2011/06/memory-forensics-where-start#.Uw14QfmSzfI

[13] Volatile Data Collection Methodology, Page No: 94, Module 3: Collecting Volatile Data, CERT Training and Education handbook, First Responders Guide to ComputerForensic

[14] http://www.acpo.police.uk/documents/crime/2011/201110-cba-digital-evidence-v5.pdf

[15] http://forensic.belkasoft.com/en/live-ram-forensics

[16] http://ru.belkasoft.com/en/live-ram-forensics

[17] Lest We Remember: Cold Boot Attacks on Encryption Keys, J. Alex Halderman, Seth D. Schoen, Nadia Heninger, William Clarkson, William Paul, Joseph A. Calandrino, Ariel J. Feldman, Jacob Appelbaum, and Edward W. FeltenFebruary, 21, 2008, Proc. 2008 USENIX Security Symposium

[18] http://www.gfi.com/blog/top-20-free-digital-forensic-investigation-tools-forsysadm