BirEx JOURNAL
Budapest International Research in Exact Sciences
Medical, Biological, Argiculture, Engineering Science and other related areas
e-ISSN : 2655-7827
p-ISSN : 2655-7835
http://www.bircu-journal.com/index.php/birex

# Computer Forensics in Cloud Computing Systems

**Pedro Ramos Brandao**
*Interdisciplinary Center for History, Cultures and Societies (UID/HIS/00057/2019) – Évora University*

**Abstract** *: The challenges on traditional Computer Forensic mythology in facing with the growing adoption of Cloud Computing services and models by corporations and organizations. We analyse the results of recent research in analysis of computers' physical memory, new methodologies and tools, and how these technologies can be applied to computer forensics in this specific context.*
**Keywords** *: Computer Forensics; Cloud Computing; RAM analysis.*

## I. Introduction

Cloud Computing is a generic term that describes the evolution of technologies and processes, composed of services, applications, information and distributed infrastructures, so that they can be dynamically structured, elastically and rapidly as they are consumed. Cloud computing implies the separation of information and applications into the infrastructure that supports them. There are several descriptions for Cloud Computing-based services as well as various models to represent it. In this work, we will adopt the one described by the Cloud Security Alliance (CSA), which focuses on the model and the methodologies the perspective of the information security area. That is, Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly.

In this scenario, traditional forensic methods, methodologies and techniques may prove to be inefficient. Results of new studies and new approaches will be presented in this paper so that forensic analysis in Cloud Computing is effective.

## II. Cloud Computing, an Overview

### 2.1 The architecture

Cloud computing anticipates a strong isolation of the applications and information, infrastructure and mechanisms that are used to support it, so that the resources available can be dynamically allocated whenever requested. Most of the time, cloud computing is directly linked to virtualization technologies (the definition of which is a computer-generated abstraction), specifically for the ease of configuring and making available the integration, scalability, mobility, and dynamic storage of the resources used.

For computer forensics it is relevant to understand the impact that the distributed computing architecture and the consequent dispersion of the information, used in Cloud Computing, brings to the collection of evidence, since, for example, data can be distributed by several countries.

### 2.2 Main features

Abstraction of infrastructure:

The computing, network and storage infrastructure is isolated from information and application resources. From the point of view of the application and the services made available no matter where and by what means the data is processed, transmitted or stored.

Resources democratization:

A logical consequence of the abstraction of infrastructure, the democratization of resources, be they infrastructures, applications or information, allows them to be made available as a pool to those who have the authorization to access them.

Architecture-Oriented Services:

The abstraction of infrastructure and the democratization of resources lead to services oriented to architecture, where resources can be accessed and used in a standardized way. Thus, the focus is on the delivery of services and not on infrastructure management.

Elasticity/ Dynamics:

High levels of automation and virtualization, in conjunction with faster and more reliable connectivity, allow resource allocation to be expanded or retracted to meet requested capacity. Thus, the resources can be better utilized and the levels of services more easily achieved.

Consumption utility and storage:

The four characteristics, previously explained, combined allow the visibility at the atomic level of the resources, by service providers. This visibility allows for models of cost and usage (the contractor of a service can consume it at will, but will pay for everything that is provided and consumed) are implemented. This leads to improved environment management, with increased scale and predictable costs.

### 2.3 Models of provision of the Services

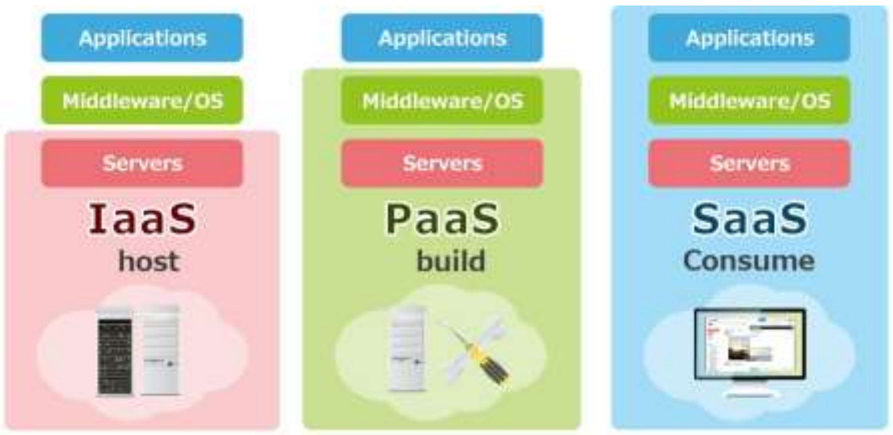


Figure 1: Cloud Computing Services Delivery Models

#### 2.3.1 Software as a Service (SaaS)

What is offered in this model are applications that run on the cloud infrastructure and can be accessed by thin clients, such as browsers. The user/ consumer does not manage or control the infrastructure used by applications (network, storage devices, operating systems, etc.), or even application settings (except for a few specific configurations).

Key points:

- Applications are supplied by the service provider.

The user does not manage or control the underlying cloud infrastructure or individual application capabilities.

- Services offered include:
  Enterprise services such as: work flow management, groupware and collaborative, supply chain, communications, digital signature, customer relationship management (CRM), desktop software, financial management, geo-spatial, and search.
- Web 2.0 applications such as: metadata management, social networking, blogs, wiki services, and portal services.
- Not suitable for real-time applications or for those where data is not allowed to be hosted externally.
- Examples: Office 365, Salesforce.com, Gmail.

### 2.3.2 Platform as a Service (PaaS)

Consumers use the cloud infrastructure to make their applications available that must be developed in languages supported by the service provider. The user/ consumer does not manage or controls the infrastructure used by the applications (network, storage devices, operating systems etc.), but has full control and responsibility over the applications available.

Key Points:

- Allows a cloud user to deploy consumer-created or acquired applications using programming languages and tools supported by the service provider.
- The user:
  - ➢ Has control over the deployed applications and, possibly, application hosting environment configurations.
  - ➢ Does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage.
- Not particularly useful when:
  - ➢ The application must be portable.
  - ➢ Proprietary programming languages are used.
  - ➢ The hardware and software must be customized to improve the performance of the application.

### 2.3.3 Infrastructure as a Service (IaaS)

What is offered to the consumer in this model is the rental of processing, storage devices, networks and other resources considered basic. The consumer can run any software (such as various operating systems and applications), but is responsible for it.

Key Points:

- The user is able to deploy and run arbitrary software, which can include operating systems and applications.
- The user does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, deployed applications, and possibly limited control of some networking components, e.g., host firewalls.

- Services offered by this delivery model include: server hosting, Web servers, storage, computing hardware, operating systems, virtual instances, load balancing, Internet access, and bandwidth provisioning.

## 2.4 Implementation Models

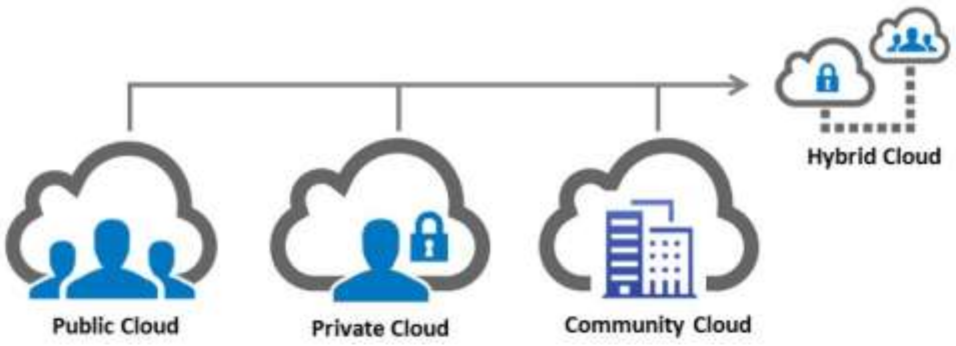


Figure 2: Cloud Computing Implementation Models

### 2.4.1 Private Cloud Computing

Also known as Internal Cloud. Its main characteristic is that it is offered by the organization itself, or by the service provider that the organization indicates, that will consume the resources. A dedicated operating environment that adds the features and characteristics of cloud computing where it is available. The physical infrastructure is the organization's own infrastructure, and may be located in your datacentre or service provider's datacentre as determined. In this way, the users of the services are considered reliable (staff/ employees, third parties and others that have some contractual relation with the organization). Untrusted users are those who are not logical extensions of the organization, even if they somehow consume the services of the organization.

### 2.4.2 Public Cloud Computing

They are offered by service providers. The operating environment offered, encompassing all the features of cloud computing, can be dedicated or shared. Consumers of services are considered unreliable.

### 2.4.3 Hybrid Cloud Computing

This system is widely used nowadays because it divides levels of security and privacy. A part of the network is in Private Cloud Computing, that is, within the perimeter of physical security of the company, another part of the network is in a system of Public Cloud Computing, that is, an external provider of cloud computing. However, both systems are interconnected and synchronized.

*Budapest International Research in Exact Sciences (BirEx) Journal*
*Volume I, No 1, January 2019, Page: 71-86*
*e-ISSN: 2655-7827 (Online), p-ISSN: 2655-7835 (Print)*
*www.bircu-journal.com/index.php/birex*
*emails: birex.journal@gmail.com*
*birex.journal.qa@gmail.com*

### 2.4.4 Community Cloud

It is a very specific type, sometimes it is not considered in the groups of implementation models, hence it adds parts of the previously mentioned models:

• The cloud infrastructure is provisioned for exclusive use by a specific community of consumers from organizations that have shared concerns (e.g., mission, security requirements, policy, and compliance considerations).

• It may be owned, managed, and operated by one or more of the organizations in the community, a third party, or some combination of them, and it may exist on or off premises.

## 2.5 Reference model

The CSA describes a reference model for cloud computing that illustrates how service delivery models are orchestrated and layered. The highest layer (SaaS) depends on the next lower layer, which supports it. The lowest layer, IaaS, serves as the basis for the others. It shall be taken into account that this is a reference model that fulfils its role of clarifying the relationships of the cloud computing service delivery models and the orchestration models of an entire IT structure.

## 2.6 Security Architecture

The security architecture will depend on the combination of the availability model and the mode of contracted consumptions. The availability model will define which controls are to be implemented and how this implementation should occur, since there are specific controls for each layer of the reference model. Example of this is that in the IaaS model there must be physical level controls (alarms, guards, etc.), whereas in SaaS they are the controls as a firewall for web applications. The mode of consumption indicates who will be responsible for administering the controls. Thus, for example, in the case of the private modality the contractor will be responsible for the administration of the security architecture. In the public mode, the cloud computing service provider is responsible for administering the security architecture.

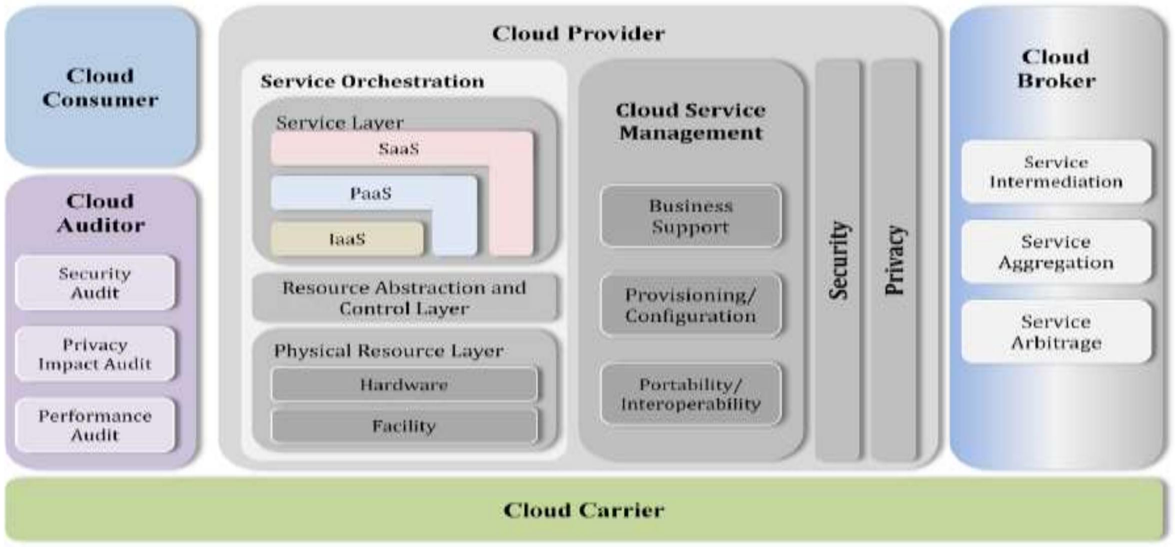


Figure 3: Orchestrated Standard Cloud Computing Models

## 3. Civil Liability by Availability and Implementation model

With regard to issues of responsibility and ownership of systems in Cloud Computing, it is necessary to pay close attention to the availability models.

If we want to analyse forensic network equipment such as switches or routers used by someone using Cloud Computing at the Software-as-Service Model level, we can never implement this investigation at the level of the suspect user; infrastructure provider, even though he is not suspected of any wrongdoing.

The same applies in the case of a suspect making use of Cloud Computing at the Platform-as-service Model level, since he does not have access to the configurations of the intermediate network equipment. In the case of a suspect use of the Software-as-Service Model and whose objective is to investigate the configurations and logs of a client application of its electronic mail, it can be directly applied to that user, since it is responsible for the content of use of this layer in said model.

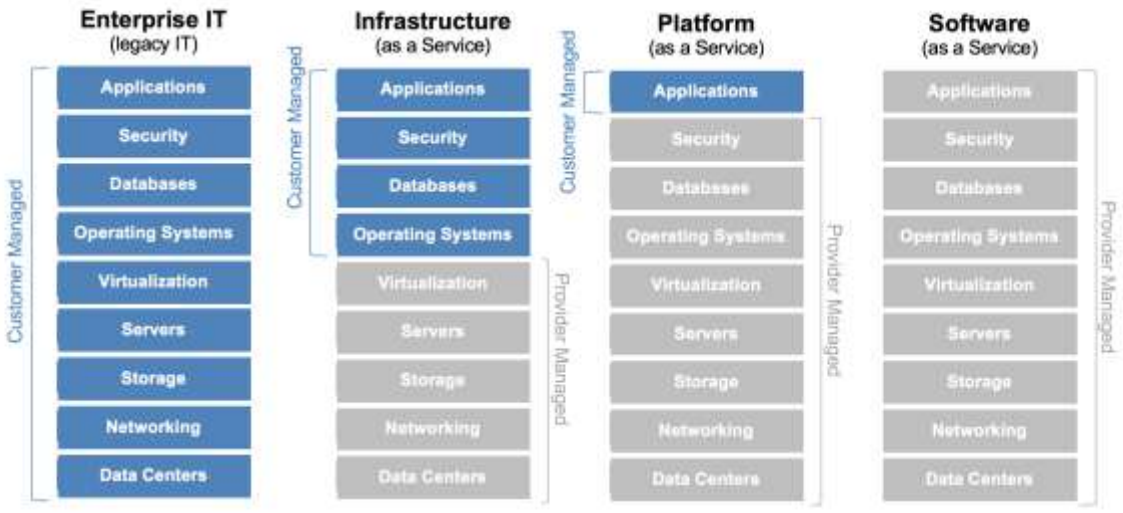


Figure 4: The heads of each service according to the Models

## 4. Computing Forensics for Cloud Computing

### 4.1 Computer forensics. Challenges in Cloud Computing Scenarios

In many cases the traditional methodology employed in computer forensics, turning off equipment and making a bit-by-bit image of the hard drives found, does not prove to be a viable option. For example, encrypted hard disks make it difficult, if not impossible, to access the data stored in them. Another problem is that often turning off an equipment for several hours to make an image of your discs can bring financial losses to the company, in addition to compromising the contracted service level agreements. Cloud computing contributes to making the analysis more complex using traditional forensic computing methodology, as it introduces variables inherent in its characteristics, models, and architecture.

#### 4.1.1 Data location

Traditionally, the data and information were in the custody and responsibility of the companies and, in most cases, in their physical facilities (at most in the datacenter of third companies contracted for this purpose). The distributed architecture inherent in cloud computing supports geographically distributed data. In this way, an image of local hard disks may not be efficient, as these disks may not contain the same data that is in the cloud. Depending on the availability model contracted, the contracting company may choose to use thin clients, without a hard disk, with the data that is in the cloud. The geographic distribution of the data brings another factor of concern to the forensic analysts, since this data can be in different States or countries and, therefore, subject to different regulations and legislations. Access to this data becomes more difficult.

#### 4.1.2 Unknown file systems

In order to meet the dynamism and elasticity inherent in the distributed architecture of cloud computing, the file systems used in this architecture may have been redesigned, customized or even created specifically to serve certain requests or save certain types of data. People who use the traditional forensic computing methodology may face difficulties retrieving data recorded in these file systems, as their structures may not be known to anyone who is doing the analysis. In addition, the file system can be proprietary and make the task of knowing it even more complex.

#### 4.1.3 Volume of data to be audited

The capacity of mass storage systems grows rapidly [3] and are increasingly accessible to public. For forensic computing, the use of traditional methodology can be a problem, hence the time spent to generate the bit-a-bit image tends to be higher, even considering the advance of the hardware devices that make this type of copy. In addition, you have to consider the space required to store the generated images and their respective copies. Growing cloud computing services make a larger storage area critical (multiple e-mail providers offer more than 5GB of storage per user, for example), increasing the volume of data that can be analysed. To make matters worse, this huge amount that is distributed in distributed cloud systems requires the use of a more specific technology as well as a mass data analysis system. So, the analysis of this type of data in Cloud Computing requires the use of Data mining technology.

#### 4.1.4 Encryption of storage systems in Cloud Computing

CSA recommends in your guide that the cloud computing service provider use strong encryption on storage systems to prevent data from being accessed by anyone who does not have the right to do so even after storage devices, have been discarded. The encryption of storage systems makes it difficult, if not impossible, to analyse images made of these devices following the traditional methodology of computing forensics. In the face of these challenges, as the architecture and model of cloud computing make it more complex, alternative methodologies need to be found for computer forensics so that these issues can be better met.

### 4.2 Forensic Computer Live Response to address the new challenges

Previously, the challenges that the traditional methodology used by forensic computing have faced are described, and how cloud computing scenarios make these challenges even more complex. Another factor that can not be disregarded is the existence of malwares that exist only in memory while they are run. The proposal presented here to address these challenges is based on the analysis of equipment while they are connected. This technique is known as live response,

It is intended to collect volatile data from a system while it is running. The collection of evidence has to consider its order of volatility, in particular the RAM content. The range of this search is a live response on devices running Windows family operating systems. RFC 3227 [4] provides recommendations on the evidence-gathering process. It recommends that this collection follow the order of evidence volatility, from the more volatile to the less volatile. Thus, one of the first evidences that must be collected is physical memory (RAM). RAM contains all the programs that are running in a given period. The new RAM analysis techniques, which will be described a posteriori, allow to discover processes and threads that were not executed as well as the files and handles
used by them. This undoubtedly allows forensic analysts and investigators to have a more detailed and accurate view of what happened in certain systems, and can help to better address the challenges presented.

## 5 Analysis of RAM (volatile memory of computers)

### 5.1 Analysis and data collection of RAM

Following the order of volatility recommended by RFC 3227, RAM memory must be collected and analysed first. [5] Before discussing the techniques of RAM analysis, some forms of RAM collection will be presented. The purpose of collecting RAM information is to take a "snapshot" of what was running at one point and save it to a file for later analysis. There are several techniques that may be used to extract the contents of RAM to a file, but perhaps the best known and used is the dd tool (http://en.wikipedia.org/wiki/dd_(Unix)), or some of its variations such asWin32dd (http://win32dd.msuiche.net) or MDD (http://www.mantech.com/msma/MDD.asp).

### 5.2 RAM Analysis through traditional techniques

Traditionally, the analysis of the content extracted from physical memory consisted of searching the RAM file for passwords, e-mail addresses, IP addresses or any other strings that could give clues during the analysis. The problem with this approach is that it was difficult to correlate the information found with a specific process, that is, it had no context. The contextualization of the information found in RAM has become the target of further studies.

In order to promote debate, research and development of tools and techniques in this area the Digital Forensic Research Workshop (DFRWS - www.dfrws.org) in 2005 launched a challenge of RAM content analysis. The goal was achieved. There was a great advance in the understanding of the RAM structure by the community and several tools were made available.

## 5.3 Analysis of RAM through new approaches

### 5.3.1 Operating system version

In the possession of a RAM dump file, the first thing to be done is to determine the operating system of this dump file, as it is a Windows system the version is very relevant. This is because the structures used to define the threads and processes in memory vary according to the versions of Windows, suffering variations even between different service packs, as highlighted by Harlan Carvey [6]. There are several tools that do well in this role. In a simplified way, what many of them do is try to find in the RAM image file the Windows kernel, which is an executable file, and within its structure, look for the section VS_VERSION_INFO. This section contains the product name and its version. Perhaps this is, so far, the most reliable way to determine the Windows version of a RAM image.

### 5.3.2 Basic Operation of Processes on Windows Platforms

Many of the researches have processes as sources of information pertaining to forensic analysis of RAM, so understanding the process structures and how they are created is essential. Harlan Carvey also comments in his book that most process-related concepts are valid for different versions of Windows, and that the difference lies in the very structure of the process. There are two main structures of the processes that are relevant for computer forensics: EProcess and PEB (Process Environment Block). Each Windows process is represented by an executive process, EProcess (Executive Process), which is a data structure that stores process attributes and pointers to other structures also contained in the process. It is important to note that these structures are completely dependent on the Windows version so that their sizes and even the values of the structures change over the different versions and service packs. Fortunately, you can see how this structure and the other frameworks of a process are, with the Microsoft Debugging Tools tool and the correct symbols for the operating system and service pack, which can be downloaded for free from Microsoft's own website. To see the EProcess structure, you must, after installing the tool and the correct symbols, open the command prompt and type livekd -w. In the window that will open, the graphic interface of the debugger, just type dt -a -b -v _EPROCESS. The entire EProcess structure will be listed. Below are two excerpts from this structure as examples of variations it may have on different versions of Windows operating systems:

- Windows XP SP3

kd> dt -a -b -v _EPROCESS
ntdll!_EPROCESS
**struct _EPROCESS, 107 elements, 0x260 bytes**
+0x000 Pcb : struct _KPROCESS,
29 elements, 0x6c bytes
+0x000 Header : struct
_DISPATCHER_HEADER, 6 elements, 0x10 bytes

- Windows 7 RC

nt!_EPROCESS

```
struct _EPROCESS, 133 elements, 0x2c0 bytes
+0x000 Pcb : struct _KPROCESS,
34 elements, 0x98 bytes
+0x000 Header : struct
_DISPATCHER_HEADER, 30 elements, 0x10 bytes
```

Purposely the line of the structure that informs its elements and its size in the two stretches was highlighted. Notice that the EProcess structure of Windows 7 has more elements and is larger. The complete output of this command is larger (0x2c0 - 704 - bytes) and therefore has not been fully played. There is another important element referenced in the EProcess structure, the PEB (Process Environment Block). For computer forensics the following elements of the PEB are highlighted:

- Pointer to the loader data structure (PPEB_LDR_DATA), which has pointers or references to the DLLs used by the process;
- Pointer to the address of the base image, where it is expected to find the beginning of the executable file;
- Pointer to the process parameter structure, which contains the path to the DLL, the path to the executable file, and the command line used to start the process.

Now that you have an overview of the structure of a process, is necessary to understand how they are built into Windows operating systems. Mark Russinovich and David Solomon do an excellent job in describing the flow of process creation, which can be summarized as follows [7]:

a) Validation of parameters; conversion of Windows subsystem flags and options to native form; do parsing, validation and conversion of the attribute list to its native form.
b) Open the image file (.exe) to be run inside the process.
c) Create the Windows Executive Process object.
d) Create the initial thread.
e) Perform post-process tasks, and initialize the Windows subsystem.
f) Start execution of the initial thread.
g) In the context of the new process, complete the initialization of the address space (to load the necessary DLLs, for example) and begin the execution of the program.

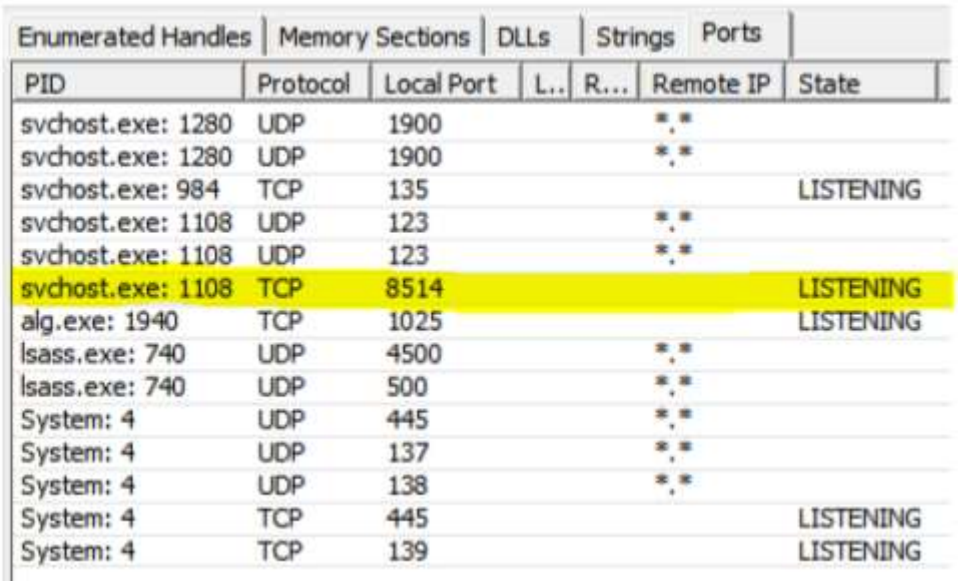
From now on the newly created process uses memory according to the EProcess structure (and other structures) and can start to be used more as it is running.

## 5.4 Analysis of RAM with Conficker in action

Conficker is a malware that employs the characteristics of cloud computing to install and upgrade itself. Version B, which will be used for analysis, has an algorithm that generates a daily list of domains that can be used to download your updates. This mechanism provides an efficient and highly mobile update service - the locations are recalculated every day by the infected machines [8]. It is a malware that operates in a distributed way over the Internet, with several clients (infected micros) and servers (machines that host the malicious program and its new versions) distributed over the Internet. It can be understood as a new type of service delivery: Virus as a Service (VaaS) or Malware as a Service (MaaS), in which contaminated machines are offered as networks that can be used by spammers, phishers, paedophiles and groups that perform malicious activity over the Internet. Thus, in order to demonstrate how the analysis of RAM in live response cases can help researchers and analysts in the area of computer forensics, Conficker B has been installed in a "virtual" laboratory with the following specifications:

- Virtualization: Virtual Box (www.virtualbox.org);
- Operational system: Windows XP SP3 with all patches up to 7/20/2009 applied and 256 MB configured for RAM;
- RAM Image Collection: Helix Pro (www.efense.com);
- Memory File Analysis: Memorize and Audit Viewer (www.mandiant.com).

This is not an in-depth analysis of Conficker, but of how to use the new RAM analysis methodologies to identify a process responsible for suspicious activity. After configuring the described lab environment, it was necessary to infect the machine with a malicious program, in this case Conficker. With the certainty that the test machine was contaminated, an image of your RAM was obtained using Helix Pro. The next step was to analyse the image with Memorize and Audit Viewer. The Audit Viewer was used to interpret the results and display them in an easy-to-navigate graphical interface. Figure 5 shows the ports that were in use and the processes that used them. With this view you could see that the ID 1108 process, svchost.exe, was listening on port 8514.

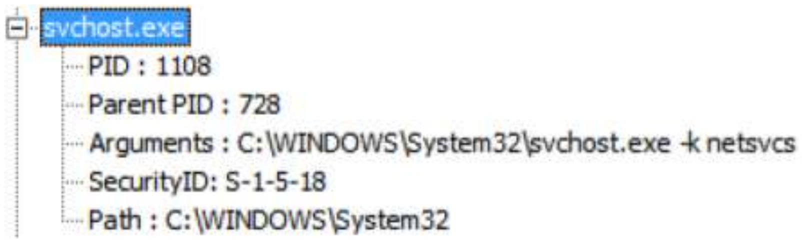


Enumerated Handles | Memory Sections | DLLs | Strings | Ports

| PID | Protocol | Local Port | L... | R... | Remote IP | State |
|---|---|---|---|---|---|---|
| svchost.exe: 1280 | UDP | 1900 | | | *.* | |
| svchost.exe: 1280 | UDP | 1900 | | | *.* | |
| svchost.exe: 984 | TCP | 135 | | | | LISTENING |
| svchost.exe: 1108 | UDP | 123 | | | *.* | |
| svchost.exe: 1108 | UDP | 123 | | | *.* | |
| svchost.exe: 1108 | TCP | 8514 | | | | LISTENING |
| alg.exe: 1940 | TCP | 1025 | | | | LISTENING |
| lsass.exe: 740 | UDP | 4500 | | | *.* | |
| lsass.exe: 740 | UDP | 500 | | | *.* | |
| System: 4 | UDP | 445 | | | *.* | |
| System: 4 | UDP | 137 | | | *.* | |
| System: 4 | UDP | 138 | | | *.* | |
| System: 4 | TCP | 445 | | | | LISTENING |
| System: 4 | TCP | 139 | | | | LISTENING |

Figure 5: Process of listening on a suspicious port through the Audit Viewer

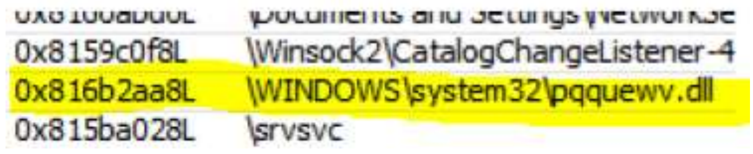
The suspect svchost.exe process is named like a legitimate Windows operating system process. The legitimate process has the role to check in a registry key the group of services it should load and is located in% SYSTEMROOT% \ system32 [9]. The Audit Viewer shows the path from where the process was called, as shown in Figure 6. One can then conclude that this is a legitimate process, but one that could have been used to load malicious code.



```
svchost.exe
  PID : 1108
  Parent PID : 728
  Arguments : C:\WINDOWS\System32\svchost.exe -k netsvcs
  SecurityID: S-1-5-18
  Path : C:\WINDOWS\System32
```

Figure 6: Path through which the process was executed, viewed through Audit Viwer.

Based on this information it may be interesting, in certain situations, to capture the image of the process to analyse it in more detail. You can do this from within the Audit Viewer itself by right-clicking on the desired process and then clicking on Acquire Process. Navigating the Handles tab of the Audit Viewer for the process in question, you may notice that there is a handle pointing to the c: \ windows \ system32 \ pqquewv.dll dll, as illustrated in Figure 7.



```
0x8159c0f8L    \Winsock2\CatalogChangeListener-4
0x816b2aa8L    \WINDOWS\system32\pqquewv.dll
0x815ba028L    \srvsvc
```

Figure 7: Handles for process ID 1108, viewed through Audit Viwer.

It is interesting to note that although it is a dll, it does not appear in the list of dlls that the program loads by itself, ie it is very likely that this dll does not appear in the import table2 of the svchost.exe executable. This fact is also very suspect. Why would a dll need to be loaded if it does not appear in the import table of an executable? As mentioned, the svchost.exe executable reads from a registry key the information about a particular service group that must be loaded into memory. It is enough to see in the appropriate registry key the group of services that is called and which services that are part of this group. The key is HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\SvcHost. You should then right-click on the netsvcs entry and then click Modify. In the list that opens, go to its end and look for a random service name [10]. In the case under study, the unknown service name is aebri. The next step is to go to HKLM\SYSTEM\CurrentControlSet\Services\ 'name of the malicious service', I'll hit this case. The problem is that Conficker is an intelligent malicious code that tries to hide its presence to the maximum. Therefore, this registry key had its permissions changed so that it was not visible at first. Permissions must be restored before the key is seen. It contains the parameters that are used to call Conficker. The value found in the lab machine's registry was% SystemRoot% \ system32 \ pqquewv.dll. Conficker is a malicious code that, after its installation, searches for updates to its code on several servers on the Internet and also listens on certain ports so that it can serve as a repository for other Confickers to update themselves. [11] [12]

## 6. Guidelines for Evidence Collection and Archiving [13]

This topic is practically a direct application of what is stated in the above RFC 3227, given its operational importance we decided to highlight it in this topic.

### 6.1 Guiding Principles during Evidence Collection [13]

- Adhere to your site's Security Policy and engage the appropriate Incident Handling and Law Enforcement personnel.

- Capture as accurate a picture of the system as possible.
- Keep detailed notes. These should include dates and times. If possible, generate automatic transcript. (e.g., On Unix systems the 'script' program can be used, however the output file it generates should not be to media that is part of the evidence). Notes and print-outs should be signed and dated.
- Note the difference between the system clock and UTC. For each timestamp provided, indicate whether UTC or local time is used.
- Be prepared to testify (perhaps years later) outlining all actions you took and at what times. Detailed notes will be vital.
- Minimise changes to the data as you are collecting it. This is not limited to content changes; you should avoid updating file or directory access times.
- Remove external avenues for change.
- When confronted with a choice between collection and analysis you should do collection first and analysis later.
- Though it hardly needs stating, your procedures should be implementable. As with any aspect of an incident response policy, procedures should be tested to ensure feasibility, particularly in a crisis. If possible, procedures should be automated for reasons of speed and accuracy. Be methodical.
- For each device, a methodical approach should be adopted which follows the guidelines laid down in your collection procedure. Speed will often be critical so where there are a number of devices requiring examination it may be appropriate to spread the work among your team to collect the evidence in parallel. However, on a single given system collection should be done step by step.
- Proceed from the volatile to the less volatile (see the Order of Volatility below).

### 6.2 Order of Volatility [13]

When collecting evidence, you should proceed from the volatile to the less volatile. Here is an example order of volatility for a typical system.

- Registers, cache;
- Routing table, arp cache, process table, kernel statistics, memory;
- Temporary file systems;
- Disk;
- Remote logging and monitoring data that is relevant to the system in question;
- Physical configuration, network topology;
- Archival media.

### 6.3 Things to avoid [13]

It's all too easy to destroy evidence, however inadvertently.

- Don't shut-down until you've completed evidence collection. Much evidence may be lost and the attacker may have altered the start-up/shut-down scripts/services to destroy evidence.
- Don't trust the programs on the system. Run your evidence gathering programs from appropriately protected media (see below).
- Don't run programs that modify the access time of all files on the system (e.g., 'tar' or 'xcopy').

**6.4 Privacy Considerations [13]**

- Respect the privacy rules and guidelines of your company and your legal jurisdiction. In particular, make sure no information collected along with the evidence you are searching for is available to anyone who would not normally have access to this information. This includes access to log files (which may reveal patterns of user behavior) as well as personal data files.
- Do not intrude on people's privacy without strong justification. In particular, do not collect information from areas you do not normally have reason to access (such as personal file stores) unless you have sufficient indication that there is a real incident.
- Make sure you have the backing of your company's established procedures in taking the steps you do to collect evidence of an incident.

**6.5 Legal Considerations [13]**

Computer evidence needs to be:

- Admissible: It must conform to certain legal rules before it can be put before a court.
- Authentic : It must be possible to positively tie evidential material to the incident.
- Complete : It must tell the whole story and not just a particular perspective.
- Reliable: There must be nothing about how the evidence was collected and subsequently handled that casts doubt about its authenticity and veracity.
- Believable: It must be readily believable and understandable by a court.

**6.6 The Collection Procedure [13]**

The collection procedures should be as detailed as possible. As is the case with your overall Incident Handling procedures, they should be unambiguous, and should minimize the amount of decision-making needed during the collection process.

**6.6.1 Transparency**

The methods used to collect evidence should be transparent and reproducible. You should be prepared to reproduce precisely the methods you used, and have those methods tested by independent experts.

**6.6.2 Collection Steps**

- Where is the evidence? List what systems were involved in the incident and from which evidence will be collected.
- Establish what is likely to be relevant and admissible. When in doubt err on the side of collecting too much rather than not enough.
- For each system, obtain the relevant order of volatility.
- Remove external avenues for change.
- Following the order of volatility, collect the evidence with tools as discussed in Topic 6.2.
- Record the extent of the system's clock drift.
- Question what else may be evidence as you work through the collection steps.
- Document each step.
- Don't forget the people involved. Make notes of who was there and what were they doing, what they observed and how they reacted.

Where feasible you should consider generating checksums and cryptographically signing the collected evidence, as this may make it easier to preserve a strong chain of evidence. In doing so you must not alter the evidence.

### 6.7 The Archiving Procedure [13]

Evidence must be strictly secured. In addition, the Chain of Custody needs to be clearly documented.

#### 6.7.1 Chain of Custody

We should be able to clearly describe how the evidence was found, how it was handled and everything that happened to it. The following need to be documented:

- Where, when, and by whom was the evidence discovered and collected.
- Where, when and by whom was the evidence handled or examined.
- Who had custody of the evidence, during what period. How was it stored.
- When the evidence changed custody, when and how did the transfer occur (include shipping numbers, etc.).

#### 6.7.2 Where and how to Archive

If possible, commonly used media (rather than some obscure storage media) should be used for archiving. Access to evidence should be extremely restricted, and should be clearly documented. It should be possible to detect unauthorized access.

### 6.8 Important Tools [13]

We should have the programs we need to do evidence collection and forensics on read-only media (e.g.,a CD). You should have prepared such a set of tools for each of the Operating Systems that you manage in advance of having to use it.

#### 6.8.1 Tools

The set of tools should include the following:

- A program for examining processes (e.g., 'ps').
- Programs for examining system state (e.g., 'showrev', 'ifconfig', 'netstat', 'arp').
- A program for doing bit-to-bit copies (e.g., 'dd', 'SafeBack').
- Programs for generating checksums and signatures (e.g., 'sha1sum', a checksum-enabled 'dd', 'SafeBack', 'pgp').
- Programs for generating core images and for examining them (e.g., 'gcore', 'gdb').
- Scripts to automate evidence collection (e.g., The Coroner'sToolkit [FAR1999]).

The programs in our set of tools should be statically linked, and should not require the use of any libraries other than those on the read-only media. Even then, since modern rootkits may be installed through loadable kernel modules, we should consider that in our tools might not be giving you a full picture of the system.

## VII. Conclusions

The use of cloud computing services by companies poses new challenges for forensic computing, in addition to those arising from the natural evolution of technologies. The tendency is that more and more infected computer networks will be offered as services, in which malware, due to the current technological sophistication, will reduce their tracks in these machines, and may only exist in RAM.

The traditional methodology employed by forensic analysts and researchers performs well, but the new techniques of RAM analysis can help to overcome these difficulties as they add more information and context to the analysis process. These techniques, which are able to contextualize the information found, linking them to specific processes, mapping the activities and threads of the process, allow malicious programs to be analysed when running, since many use advanced techniques to hide, which makes it difficult to before they are executed. Conficker was used as an example of how these new techniques can be applied, yet its applications are many. For example, information presented to the community after the 2005 DFRW allowed Jeff Bryner to develop tools capable of invoking in a RAM image file such as contacts, last accessed records etc., GMail and Yahoo!, pdgmail and pdymail, respectively [11].

Still on RAM content analysis, Andreas Schuster states in his blog [12] that there is an area that deserves to be studied more deeply, the persistence of data in memory after a reboot or after a crash dump, which is a sophisticated problem in Cloud Computing since the memory in these systems is emulated and dynamic. The fact that the virtual machine objects in Cloud Computing are an abstraction makes forensic research very difficult. All components of computers or virtual servers, which are those used today in cloud computing, are pure abstractions of host server hardware where virtualization is available, so forensic analysis can not be done in the traditional model. It is necessary to have complex data analysis tools with linked systems that can analyse dynamic virtual memories.

## References

[1] Cloud Security Alliance "Security Guidance for Critical Areas of Focus in Cloud Computing" 2009.
[2] Ghemawat, S., Gobioff, H., Leung, S-T, Chen W.-K, The Google File System.2003.
[3] Kingston. http://www.kingston.com/flash/dt300.asp?id=1.
[4] RFC 3227. http://www.faqs.org/rfcs/rfc3227.html.
[5] Farmer, D. Venema, W. "Forensic Computer Expertise. Theory and Applied Practice" Pearson Prentice Hall, 2007.
[6] Carvey, H. "Windows Forensics Analysis DVD toolkit", 2nd Ed., Syngress, 2009.
[7] Russinovich, M., Solomon, David A. "Microsoft Windows Internals", 5th Ed., Microsoft Press, 2004.
[8] SRI International. http://mtc.sri.com/Conficker/.
[9] Microsoft. "A description of Svchost.exe in Windows XP Professional Edition". http://support.microsoft.com/kb/314056.
[10] Microsoft. "Virus alert about the Win32/Conficker.B worm". http://support.microsoft.com/kb/962007
[11] Jeff Bryner. Pdgmail – http://www.jeffbryner.com/code/pdgmail. Pdymail - http://www.jeffbryner.com/code/pdymail.
[12] Andreas Schuster. "Data Lifetime". http://computer.forensikblog.de/en/2006/04/data_lifetime.html
[13] Network Working Group; Request for Comments: 3227; BCP: 55.