# Security for Windows Registry Using Carving

**Pragati Pawar**[*], **Prof. P.S. Kulkarni**[**]

[*] Department of Computer Science, Rajiv Gandhi College of Engineering, Research and Technology, Chandrapur (M.S).

***Abstract-*** This paper describes the Windows registry which stores a lot of system information and can also be used as forensic evidence. Many researchers have worked to know how the information is stored in the registry, but carving the registry files from the raw disk is not described. Till now the researchers performed the researches on how the registry files are carved when each block is not fragmented using the internal structure of registry file. It is also based on the internal structure of registry files, but in this paper, the fragmentation is performed on the multiple HBIN blocks instead of two HBIN blocks. It also recovers the Windows registry files using back up when the file system is crashed or damaged. The carving technique is used which is more effective and accurate for Windows registry files.

***Index Terms***- computer forensic; registry; carving

## I. Introduction

The Windows operating system is one of the most popular operating systems. It contains a lot of data in it and hence it is mostly attacked. As we know that the Windows registry is a database that stores configuration settings for System, users, applications, hardware and many other system parameters and as it has a lot of information contained in it, the registry can be an excellent source for potential evidential data in aiding forensic examiners on other aspects of forensic analysis. Although the Windows registry appears as a single hierarchy in registry editor such as regedit.exe, it is actually made up of a number of different binary files called hives on disk.

The Windows registry file has a lot of data which can be useful for the forensic examination hence forensic examiner shows a more interest in it. An important traditional field of computer forensic is data carving. In the data carving, the evidential files are extracted from raw images without using any file system metadata, such as the allocation information. In order to recover the data from registry files, carving is very important for computer forensic when the file system is crashed or damaged.

This paper uses the data carving technique to recover the Windows hive files. Section II briefly describes previous work on file carving and registry forensic. The internal structure of the hive will be illuminated in section III. In section IV, the process for carving the hive files from the raw disk is presented and some validation is offered to enhance. Section V describes some experimental results. In section VI, we outline future work, and in section VII, we conclude this paper.

## II. RELATED WORK

Win Hex [1] supports manual extension to other files. It doesn't support the hive file carving in its file type definition database. It has a magic number which occurs in it and can be added to the database. This magic number is referred as "regf" and its length is specified by the user as the maximum. It is difficult to understand for the new user how the fragmentation is performed. Hence it is not more efficient.[2]. Morgan has provided an algorithm to recover deleted data from the Windows registry [3]. Dolan-Gavitt introduces a way on how to forensic analysis of the Windows registry in memory while the attack modifies the cached version of the registry without altering the on-disk version [4].Mark Russinovich described the hive internal structure for the first time [5]. Jolanta Thomassen has provided extensive information of how registry information is organized into data structures on disk [6]. These are very important to the hive file carving. It helps a lot to carry out this experiment.

## III. REGISTRY INTERNAL STRUCTURE

The registry internal structure is important, in order to understand the carving as it is based on the internal structure of hive. The internal structure of Windows registry hives is shown in Fig. 1. Registry hive files start with a header, or base block, and continue with a series of hive bin blocks. The base block has a stable size of 4096 bytes and contains a magic number of "regf". This base block gives the information such as file name, its path, time stamp and other systems files. It can store the maximum length of 32 characters file name. It may be shortened by the Windows operating system.

As we know that the each base blocks has a series of hive bin (HBIN).HBIN consists of 4096 bytes, but sometimes may be any larger multiple of that size. These HBINs are linked together through the HBINs header. The size of a HBIN header is always 32 bytes, and the HBIN header contains the magic number "hbin". Each HBIN references the beginning of the next HBIN in addition to indicate its distance from the first HBIN. Information in the Registry is arranged in a tree-like system akin to folders and files.

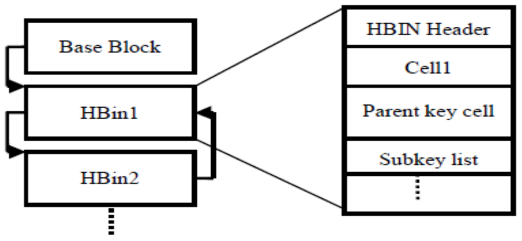

**Figure 1. Registry internal structure**

In the registry, the containers for information are called “keys”. Keys can have sub keys just as folders can have sub folders. Each HBIN has a series of variable-length cells and each cell’s total length is a multiple of 8 bytes, which is specified in the first four bytes of the cell. There are some types of cells in registry. They are: key cell, sub key lists, value lists, value cell, and security cell and class name. The key cell is the most important structure which contains a number of offset fields to other cells. It acts as a connecting link between two HBINs. This key cell is divided into some parts which contains the size of the cell, name, timestamp of the key, and offsets to parent key, sub key list, value list, security descriptor and class name. If a key does not have remaining cells, offsets are set to 0xffffffff. Key cells use sub key-lists to reference a set of other key cells. The name of data that is contained in a key is called a "value". This is something analogous to a file name. The actual data can have several formats and may be a string, a number, or a series of numbers. This cells, keys and values are arranged in tree structure as shown in fig 2.

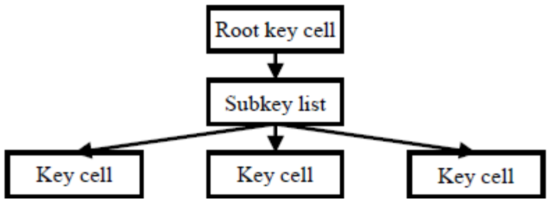

**Figure 2. Registry tree**

## IV. CARVING THE HIVE FILES

The Registry can be seen as one unified “file system” which consist of number of blocks. And this block consist of number of HBINs, again in which a number of keys and values are present. In registry, there are 5 HKEYs, its abbreviation is “handle to a key”. They are HKEY_CLASSES_ROOT (HKCR),
HKEY_CURRENT_USER (HKCU),
HKEY_LOCAL_MACHINE (HKLM),
HKEY_USERS (HKU),
HKEY_CURRENT_CONFIG (HCU).

Only two of these are actually “real”. They are HKU and HKLM. The other three are sub keys of these. Each of these five hives is composed of keys, which contain values and sub keys. Values are the names of certain items within a key, which uniquely identify specific values pertaining to the operating system, or to applications that depend upon that value. Depending on the size of the partition, the hard disk cluster can vary in size, It means that when the hive files are fragmented, the internal structure of the base blocks are not destroyed, but sometimes the HBINs’ may be fragmented when their lengths are larger than the size of cluster. In typical Windows, all hives file are illuminated in following registry path: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\hivelist. 12 hives file can be found. The six user hives locate in user home directory, they are: the ntuser and UsrClass hives for the currently logged on system user, the Local Service user, and the Network Service user; and the rest store in %Systemroot%\system32\config path, they are: default; SAM; system; SECURITY; software; and the HARDWARE hive which is generated at boot and provides information on the hardware detected in the system.. Each system user contains two hives: ntuser and UsrClass. There are some hive files with the path of %Systemroot%\repair. These hive files were created when the Windows system was installed in the first time and their base blocks have neither timestamp nor file name. The structures of the base blocks and HBINs files are intact. The unallocated space which is not used on disk sometimes includes several registry data. These unavailable data are produced when the Windows system needs delete a hive; it drops the original space, but leaves the registry data. This situation always occurs when we delete a system user, the ntuser and UsrClass are deleted. There is another probability that these data are created by last Windows system and not covered by the current system. The structures of the base blocks and HBINs in above files are intact. From the above explanation, carving method for two blocks had been proved successfully. Using this carving method, our aim is to prove that the carving can be performed successfully on multiple blocks. The carving process shows how the blocks are fragmented.

Initially select the base blocks. Based on the magic number "regf" and the file name is included the hives file name; the cluster should be the base block of a hive file.

Secondly, check whether the hive file is fragmented. If it is fragmented then it jumps to the HBIN of that base blocks. The HBINs have the magic number "hbin", the offset to the first HBIN and the length. According to the hive internal structure, all HBINs follow after the base block, and the difference of the offset to the first HBIN between two conjoint HBIN blocks is the length of the previous HBIN. If this rule is broken, the fragment must happen. There is another fragmented case. If a HBIN’s next cluster not starts with the magic number "hbin" and the current found HBINs’ total size is not greater than the base block’s total size of all HBINs. Otherwise, all the HBIN blocks can be found and the found HBIN blocks length is equal to the base block’s total size of all HBINs.it continues for next base blocks if present.

Finally, find all the fragments, and reassemble them together. If the hive file is fragmented, the next fragment must start with the magic number "hbin", and its offset to the first HBIN can be made sure because the previous HBIN’ offset to the first HBIN and its size are known. If a found fragment already belong to another hive, we ignore this fragment and otherwise, try to next one while accords with our condition Try to find all the fragments until total size of all HBINs is the same as the base block’s total size of all HBINs and integrate those HBIN blocks together by the offsets to the first HBIN.

During this process it is important to backup the files of registry. There may be a possibility that the data may be lost .hence in order to recover the data, it is an essential factor. There is a high-level validation is mentioned to verify the all found fragments are the real ones. For each new carved hive, we try to build the registry tree from the root key, if the build fails, there must be some fragments don’t fit the hive. The available bases block should include the magic number, file name and timestamp. The timestamp has very important role in the carving. If some base blocks have the same file name, the timestamp can be used to select the latest base block.

There is another way to validate base blocks is checksum that can be stored at offset 0X1FC. In order to carve the hive of multi-user, first, we should carve SAM hive accurately and acquire the key with path \SAM\Domains\Account\Users\Names. The available system users' names are this key's sub keys.

## V. EXPERIMENTAL RESULTS

In our experiment of carving the multiple registry files, first we selected the blocks for fragmentation and their HBINs. After selecting the blocks, validate the fragmented blocks. Later on all the fragmented blocks are merged. At last the registry tree is built to verify that all found fragments are the real ones or not. Because of fragmentation, the data may be scatter and the related data may be in any other blocks. During this process there is a possibility of data to be lost.

Carving technique in this experiment is maximum successful, regarding carving the file. The problem with this approach is that loss of data. There are more chances of loss of data as it is multiple files are carved .it works well when compared to other techniques but still we need to work on the back up files and recovery as well. Hence our technique is performed well for carving the multiple registry files than other.

## VI. FUTURE WORK

Currently, the tool that we used for carving the registry hives file works well when multiple blocks are fragmented. In order to be more useful, in future we have to work on the recovery part of registry by using back up files so that data must not lost and can be recoverable when needed.

## VII. CONCLUSIONS

In this paper, our aim is to propose a method to carve the registry hive files based on file's internal structure. The method is successful when the multiple HBIN hive blocks are fragmented. More importance is given to the recovery as well.

## REFERENCES

[1] X-Ways Corporation, "WinHex," http://www.winhex.com/index-c.html.

[2] Wong L.W, "Forensic Analysis of the Windows Registry,"http://www.forensicfocus.com/index.php?name=Content&pid=73&page=1.

[3] Morgan T.D, "Recovering deleted data from the Windows registry,"Digital Investigation. Vol. 5, Supplement, pp. 33–41,2008.

[4] Dolan-Gavitt B, "Challenges in Carving Registry Hives from Memory".http://moyix.blogspot.com/2007/09/challenges-in-carving-registry-hive.html.

[5] Russinovich Mark, "Inside the Registry,"http://technet.microsoft.com/en-us/library/cc750583.aspx.

[6] Jolanta Thomassen, Forensic analysis of unallocated space in Windows registry hive files, University of Livepool, 2008.

[7] Morgan T.D, "the windows nt registry file format,"http://sentinelchicken.com/data/TheWindowsNTRegistryFileFormat.pdf

[8] Dolan-Gavitt Brendan, "Forensic analysis of the Windows registry in memory" Digital Investigation. Vol. 5, Supplement 1, pp. 52–57,2008

## AUTHORS

**First Author** – Pragati Pawar, B.tech(CSE), M.tech(App), Pragati_pawar42@yahoo.com,Pragati.choti@gmail.com
**Second Author** – Prof .P.S.Kulkarni, P.hd persuing, HOD(IT),RCERT,Chandapur;kulkarnips1811@gmail.com.

**Correspondence Author** – Sarika Madavi, B.Tech(CSE),Sarika_madavi@sify.com