



---

Annual ADFSL Conference on Digital Forensics, Security and Law

2009  
Proceedings

---

May 21st, 10:00 AM

## Analysis of the 'Db' Windows Registry Data Structure


Damir Kahvedžić

Centre for Cyber Crime Investigation, University College Dublin, Ireland, [damir.kahvedzic@ucd.ie](mailto:damir.kahvedzic@ucd.ie)

Tahar Kechadi

Centre for Cyber Crime Investigation, University College Dublin, Ireland, [tahar.kechadi@ucd.ie](mailto:tahar.kechadi@ucd.ie)

Follow this and additional works at: <https://commons.erau.edu/adfsl>

 Part of the [Computer Engineering Commons](#), [Computer Law Commons](#), [Electrical and Computer Engineering Commons](#), [Forensic Science and Technology Commons](#), and the [Information Security Commons](#)

---

### Scholarly Commons Citation

Kahvedžić, Damir and Kechadi, Tahar, "Analysis of the 'Db' Windows Registry Data Structure" (2009). *Annual ADFSL Conference on Digital Forensics, Security and Law*. 11. <https://commons.erau.edu/adfsl/2009/thursday/11>

This Peer Reviewed Paper is brought to you for free and open access by the Conferences at Scholarly Commons. It has been accepted for inclusion in Annual ADFSL Conference on Digital Forensics, Security and Law by an authorized administrator of Scholarly Commons. For more information, please contact [commons@erau.edu](mailto:commons@erau.edu).

**EMBRY-RIDDLE**  
Aeronautical University™  
SCHOLARLY COMMONS

(c)ADFSL



## **Analysis of the ‘Db’ Windows Registry Data Structure**

**Damir Kahvedžić**

Centre for Cyber Crime Investigation,  
University College Dublin, Ireland,  
Tel: +353 1 716 2485  
Email: [damir.kahvedzic@ucd.ie](mailto:damir.kahvedzic@ucd.ie)

**Tahar Kechadi**

Centre for Cyber Crime Investigation,  
University College Dublin, Ireland,  
Tel: +353 1 716 2478  
Email: [tahar.kechadi@ucd.ie](mailto:tahar.kechadi@ucd.ie)

### **ABSTRACT**

The Windows Registry stores a wide variety of data representing a host of different user properties, settings and program information. The data structures used by the registry are designed to be adaptable to store these differences in a simple format. In this paper we will highlight the existence of a rare data structure that is used to store a large amount of data within the registry hives. We analyse the manner in which this data structure stores its data and the implications that it may have on evidence retrieval and digital investigation. In particular, we reveal that the three of the most popular digital investigation suites fail to recognise this structure and do not allow the investigator to view the contents of the structure.

**Keywords:** Windows Registry, Data Structure

### **1. INTRODUCTION**

The Windows registry, a very extensive database in the Windows Operating system, is designed to hold a wide variety of information used by the operating system and installed software to configure settings and enhance user experience. It is designed to be extensible and scalable with respect to the amount and type of data that it holds. Numerous data structures provide this functionality, the exact specification of which has recently been published (B. D. (2009), Morgan, T. D. (2008)). Implications of these structures with respect to data hiding and forensics have also only recently been explored in Morgan, T. D. (2008).

The db data structure is a registry data structure that has not been described in these publications. It is used by the Windows NT registry to store a value of a key that exceeds a certain (very large) length and as such is rarely found in the registry. Because of its rarity and lack of specification, the structure seems not to be supported by any of the large registry analysis software vendors in the Digital Forensics field. Programs such as Forensic Toolkit (FTK), Encase and X-Ways do not support it and fail to show the data accurately. This report will document the db data structure, how it is used in the registry and what relevance it has to digital forensic investigations.

### **2. THE BASIC STRUCTURE OF THE REGISTRY**

Each registry hive is organised into a set of specific data structures each storing specific parts of the registry. The biggest and most general one of these are called hbin blocks. They are usually 4096 bytes, 4kb, (but can be a multiple of it) and store all the other smaller data structures. The most important structure is the nk structure which stores all necessary information to define a key. Amongst other things it stores pointers to a set of vk data structures. This structure holds particular values for a

key. Therefore if a key called 'Microsoft' has a value 'Version : REG\_SZ : 1.1' then the value would be stored in a vk. 'Version' would be the name of the value and 'REG\_SZ' the datatype and '1.1' the data.

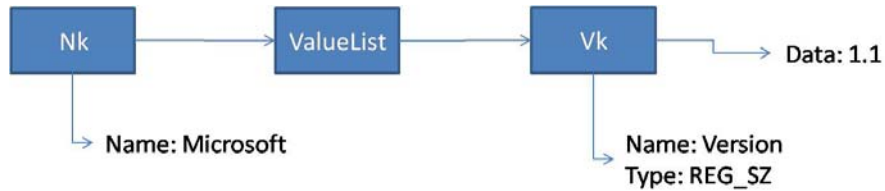


Figure 1: Typical Use of the Vk Structure

The situation is summarised in Figure 1. Further detailed specifications on the various data structures of the registries are found in (Morgan, T. D. (2008)).

The vk structure does not store the data itself but rather points to another location in the file where the actual data is stored. However, if the value is big enough the offset does not point to the actual data but to another data structure called 'db'. It is this data structure that acts as an intermediary between the vk node and the large amount of data that is stored in it.

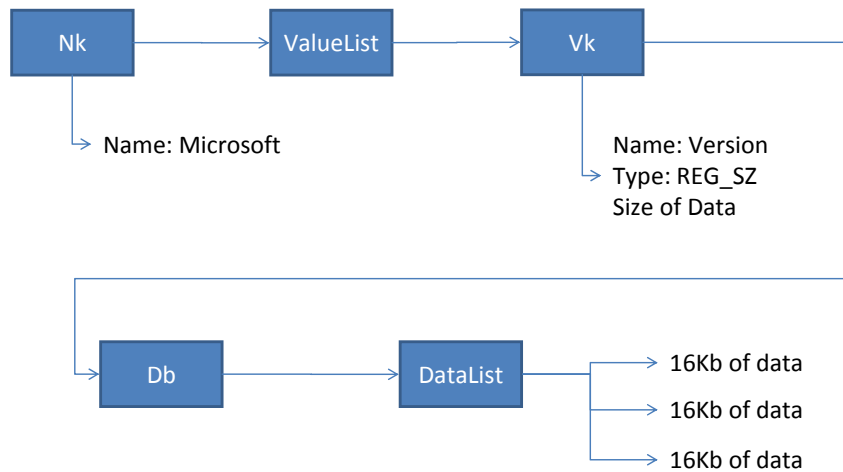


Figure 2: The Use of the Db Structure

### 3. THE STRUCTURE OF DB

The db data structure stores two offsets; one points to the set of data blocks where the data is stored, the other is unknown and appears to be useless. The first pointer points to a set of offsets that point to the actual data. The data is broken up into 16kb chunks. One large hbin block is allocated to each of these chunks. Therefore, if the value of the key goes above 16,344 bytes a db block is put into place where the actual data should be as show in Figure 2. The db data structure contains offsets that point to a DataList structure. The DataList is a data structure that does not contain an identifier and contains offsets to each of the 16kb chunks storing the value's data. The structure of the db and the DataList are shown in the Tables 1 and 2.

Structure Of Db		
Offset	Size	Description
0	dword	Size of Block
4	word	ID of Block (db)
6	dword	Number of 16kb data blocks
8	dword	Offset of DataList
12	dword	Unknown

Table 1: Configuration of the db Data Structure

Structure Of DataList		
Offset	Size	Description
0	dword	Size of Block
4+	word	Offsets of 16kb Data Blocks

Table 2: Configuration of the DataList Data Structure

#### 4. THE DB SLACK

The total size of the data distributed to the various 16kb data blocks is stored in the vk data structure. The blocks are allocated to the data regardless if the data does not fully fill the 16kb space. A marker “0x00 00 00” identifies where the data ends. Anything after this marker is not read. The data found after this marker is not part of the windows registry hierarchy and may contain remnants of previous data structures or data. If for instance, the data is shortened, the end marker is changed to reflect the new end point of the data. The excess data is not deleted and remains as *slack*. Previously stored information can be found in this place. The slack is illustrated in Figure 3.

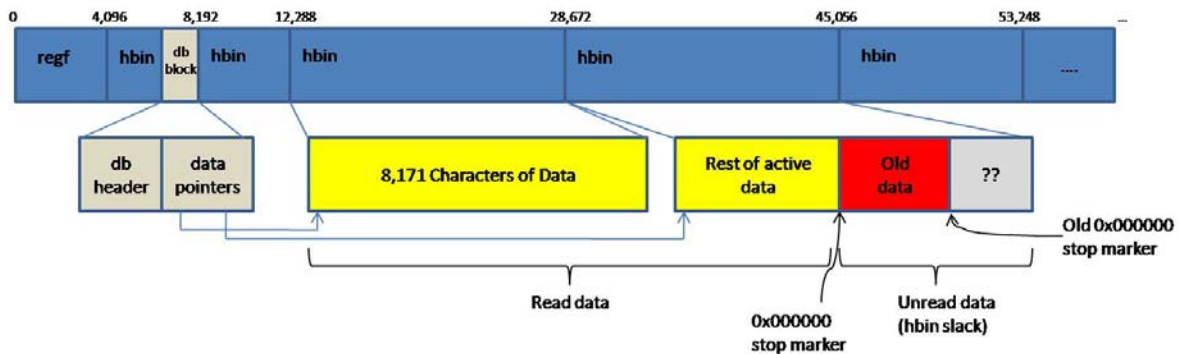


Figure 3: Db Slack

#### 5. PROBLEMS IN CURRENT FORENSIC SOFTWARE

The current leading forensic software is not aware of the db structure and fails to parse it properly. The db data structure is processed correctly in the RegEdit32 program in Windows but offline examination of values using the db data structure is not possible with the forensic tools. As such users can take advantage of this flaw and store large (possibly encrypted) files within the registry in an attempt to hide the data. As a demonstration, a large string value was placed into a registry key. The RegEdit32 window in Figure 4 shows what the data should store. This section will review the leading forensic software and how they behave with respect to this db structure.

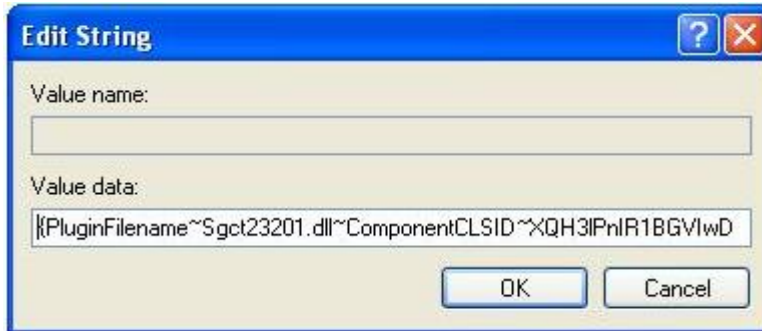


Figure 4: Normal Vk Data in RegEdit32

### 5.1 EnCase

The leading forensic software, EnCase, treats registry hive files as extension of the file system. To view a hive, the investigator must mount it as a drive and explore it like a file system. Data is show in Hex as well as ASCII. It doesn't recognise the special circumstances that create the Db data structure. In this case, Encase attempts to grab the data found at the vk address regardless if there is a db or not. As a consequence it does not follow the db offsets and reads in whatever data happens to be at the offset. As a result, it appears as if the key holds nothing but random data, thus giving a false impression. The situation is illustrated using EnCase Enterprise v.6.8 (EnCase (2009)) and is shown in Figure 5.

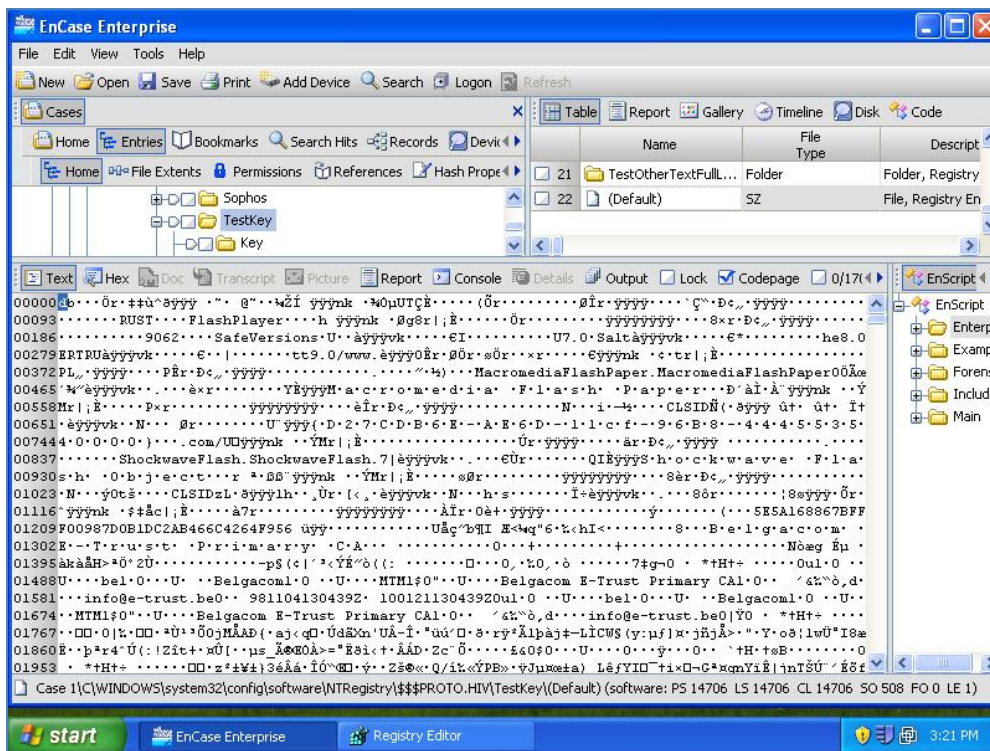


Figure 5: Encase Displaying Incorrect Data

### 5.2 X-Ways Forensics

The X-Ways Forensic v14.0 (X-Ways (2009)) program includes a separate registry viewer to view the hive files in a similar manner to RegEdit32. Similarly to EnCase above, if a registry key with the db data structure is found the data is read at the db offset. The structure is not parsed properly and attempted to be read as Unicode string values. Since the values describing the db are binary the

displayed data is nonsensical. The program throws an exception and crashes shortly after viewing the data.

### **5.3 Forensic Toolkit**

Access Data's FTK Registry Viewer v1.5.0.14 (Forensic Toolkit (2009)) program fails to find any data at the db data structure and displays nothing. It does not crash but fails to display the correct data.

## **6. SUMMARY AND DISCUSSION**

We have presented a specification for a rare registry data structure, db, used by the Windows registry to manage a key with a huge amount of data. We have described the function of this structure and analysed it with from a forensic point of view. As such we have described the structure's slack that may store remnants of past data.

The Db data structure is exceedingly rare but is created by at least one major multimedia software provider, such as Real Player (RealNetworks (2009)). As mentioned above, the db data structure is accessed as normal using the RegEdit32 program. Data being stored in it can be readily accessed using this Windows built in application. However investigators may fail to examine it if viewed offline using any of the aforementioned investigation suites. Although the data referenced may be found by brute force string searching algorithms the data would not be attributable to any active part of the registry and may bring doubt to any assertions made based on that evidence.

## **REFERENCES**

- B. D. (2009), 'Registry File Format', <http://home.eunet.no/pnordahl/ntpasswd/WinReg.txt>, visited Feb 2009.
- Encase (2009), 'Guidance Software Digital Investigations', <http://www.guidancesoftware.com/>, visited Feb 2009.
- Morgan, T. D. (2008) "Recovering Deleted Data From the Windows Registry", Digital Forensic Research Workshop, Aug 2008, Baltimore, USA.
- RealNetworks (2009), 'Real Player 11 Basic', <http://europe.real.com/player/win/>, visited Jan 2009.
- Forensic Toolkit (2009), 'Access Data', <http://www.accessdata.com/>, visited Feb 2009.
- X-Ways (2009), 'X-Ways Software Technology AG', <http://www.x-ways.net/>, visited Feb 2009.

## **ACKNOWLEDGEMENTS**

The authors would like to acknowledge "Higher Education Authority" (HEA) and "Irish Research Council for Science, Engineering and Technology" for providing funding that made this research possible.

