Detecting Hidden Encrypted Volumes

Christopher Hargreaves[1] and Howard Chivers[1],

[1] Centre for Forensic Computing, Cranfield Univeristy, Shrivenham, SN6 8LA, UK
{c.j.hargreaves, h.chivers}@cranfield.ac.uk

Abstract. Hidden encrypted volumes can cause problems in digital investigations since they provide criminal suspects with a range of opportunities for deceptive anti-forensics and a countermeasure to legislation written to force suspects to reveal decryption keys. This paper describes how hidden encrypted volumes can be detected, and their size estimated. The paper shows how multiple copies of an encrypted container can be obtained from a single disk image of Windows Vista and Windows 7 systems using the Volume Shadow Copy feature, and how the changes between shadow copies can be visualised to detect hidden volumes. The visualisation assists in the presentation of this information to a court, and exposes patterns of change which allows the size and file system of the hidden volume to be determined.

Keywords: Forensic Computing, Encryption, Hidden Volumes, RIPA, TrueCrypt

1 Introduction

This paper examines the problem of hidden encrypted volumes during digital forensic investigations. A hidden encrypted volume is a feature of certain encryption systems that allows two keys to be created for a volume; one key decrypts the true contents (hidden volume) and the other key (the 'duress key') decrypts some pre-arranged innocent content (cover volume). This could pose a challenge in digital investigations since one of the approaches to gaining access to encrypted evidence is the use of legislation to force a suspect to provide decryption keys. If the suspect provides the 'duress key' and the data is decrypted, since it is not possible to tell if there is any additional hidden content, the effectiveness of this legislative approach is reduced. It is also possible that a password is provided more subtly with the intent to deceive the forensic analyst; e.g. the password to the cover volume is written on a post-it-note stuck to the bottom of a keyboard, making the investigator believe that they have access to the all the encrypted data.

This paper provides a practical solution to the problem of identifying the existence of hidden encrypted volumes and furthermore places it in a forensic computing context, which includes the difficulty of demonstrating the existence of such a hidden volume to a court. In addition the paper also shows how information about the hidden volume (such as its size) can be inferred through an examination of the changes in the free space of the cover volume.

The paper is organised as follows: the remainder of this section details the problem of encrypted evidence and possible means of gaining access. It discusses the legislative approach in the UK, i.e. making the failure to provide decryption keys on

request an offense (Part 3 of the UK Regulation of Investigatory Powers Act). Limitations of this legislation are also discussed in terms of technical measures, such as hidden volumes, that can be used to frustrate this approach. A particular implementation of hidden volumes is discussed (TrueCrypt), which is used throughout the later examples. Section 2 provides a summary of related work, and Section 3 describes the methodology for the research, including how multiple copies of an encrypted container can be obtained from a single disk image using the Volume Shadow Copy feature of Windows Vista and Windows 7. Section 3 also shows how the changes that occur between multiple versions of the container can be visualised and how information about the hidden volume can be extracted. Section 4 evaluates the approach and Sections 5 and 6 provide a discussion of future work and the conclusions.

1.1 Encrypted Digital Evidence

In a digital investigation there are a number of approaches that can be used to attempt to gain access to encrypted evidence. These are discussed in [1], [2], [3] and [4], and can be summarised as:

- Persuading/forcing suspect to provide the decryption key
- Locating copies of unencrypted data
- Locating keys or passphrases
- Intelligent password attacks
- Exhaustive key search
- Exploiting implementation vulnerabilities
- Hardware or software surveillance

While any of these approaches can be used, legislation has been passed in the UK to make it an offence for a suspect to fail to provide means to access the encrypted information. This makes the option of forcing the suspect to provide decryption keys more viable. This legislation is contained in Part 3 of the Regulation of Investigatory Powers Act 2000 (RIPA) [5] and the requirements of responding to a RIPA notice are explained in detail in [6]. On receiving a RIPA Part 3 notice, the person concerned is required to either provide the electronic information in intelligible form or to disclose the key to enable the data to be put into intelligible form. While this legislative approach can be used to prosecute those who do not provide decryption keys, there are technical solutions that can be used as a countermeasure to this approach. One such countermeasure is the use of hidden volumes.

Hidden volumes employ the principles of steganography, meaning that the existence of data is hidden in addition to the content. Steganography implementations often involve hiding some secret data within a ‘cover file’, for example hiding text in redundant elements of a jpeg. In the case of hidden encrypted volumes the steganography is implemented as part of the encryption system, where one password decrypts the real content, and a second password decrypts some prearranged innocent content. This is done in such a way that it is not possible to tell if there is also

additional hidden content. An example implementation is detailed in the following section and discusses the popular open source product TrueCrypt.

1.2 TrueCrypt and Hidden Encrypted Volumes

TrueCrypt is a "software system for establishing and maintaining an on-the-fly-encrypted volume" meaning that "data are automatically encrypted or decrypted right before they are loaded or saved, without any user intervention" [7]. TrueCrypt has become a popular tool for encrypting data with over 12 million downloads as of December 2009. TrueCrypt can create encrypted containers and can encrypt full volumes or disks. TrueCrypt also offers hidden volume functionality so that one password decrypts the true content and another password (a 'duress' password) decrypts some prearranged innocent content. The structures of a standard TrueCrypt volume and a 'cover volume' containing a hidden volume are shown in Figure 1.

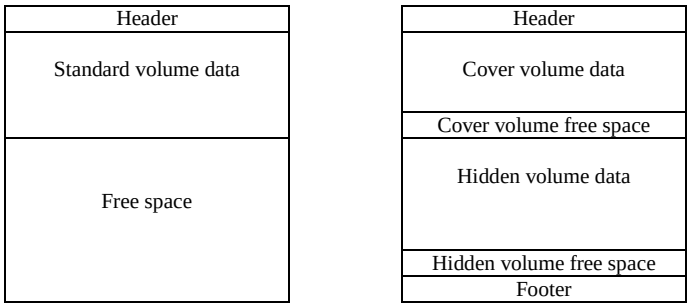

Fig. 1. The structure of a standard TrueCrypt volume (left) and one containing a hidden volume (right).

While the structures of the volumes are shown above, to an analyst encountering an encrypted volume, all that is visible is random data since the header, footer, volume data and volume free space are all encrypted. Furthermore, in a standard TrueCrypt volume the free space is filled with random data [8]. The process by which a TrueCrypt volume that contains a hidden volume is decrypted is as follows:

- The user enters a password and a key is derived from it
- That key used to attempt to decrypt the footer
- If the footer decryption is successful then the volume key is obtained from the footer and used to decrypt the hidden volume
- If the footer is not successfully decrypted then the key is used to attempt to decrypt the header
- If the header decryption is successful then the volume key is obtained from the header and used to decrypt the cover volume

The suspect can therefore use one of two passwords to access the volume. If the 'true' password is provided then it successfully decrypts the footer and access is provided to

the hidden volume; if the ‘duress’ password is provided then it fails to decrypt the footer but decrypts the header and accesses the cover volume data.

In the second case (using the ‘duress’ key), the process cannot be distinguished from decrypting a standard volume that does not contain a hidden volume, since in both cases the footer will fail to decrypt and the process moves on to attempt decryption of the header. In addition, since free space on a volume is filled with random data, free space on a standard volume is indistinguishable from a hidden container embedded in the free space of a cover volume. Therefore if a suspect is forced to provide a key, if the provided key decrypts a standard volume it is not possible to determine if there is a secondary, hidden volume that would be accessible if a second password was provided.

1.4 Summary

While there are a number of options for gaining access to encrypted data, the simplest is often to ask for the password [10]. While the suspect may choose not to cooperate, in the UK there is now legislation in place to encourage the production of decryption keys (RIPA Part 3). However, technical measures are available to counter this and can take the form of systems that use multiple keys -- one to decrypt the true content and another to decrypt some pre-arranged innocent content.

2 Related Work

Despite the careful design of TrueCrypt, in practice it may still be possible to infer the presence of a hidden volume. This is discussed in both [9] and in the TrueCrypt documentation [8].

Three threat models are described in [9] to which a hidden volume system could be subjected. These are: *one time access*, where the attacker has a single snapshot of the volume; *intermittent access*, where several versions of the volume are available over time; and *regular access*, where many versions of the volume are available taken at short intervals.

Several opportunities to infer the presence of a hidden volume are presented in [9], assuming the most restrictive model (one time access). These include data leakage through the operating system, e.g. shortcut files that are created automatically and point to data on the hidden volume; data leakage through ‘primary applications’, e.g. Microsoft Word auto-saving copies of a file from a hidden volume in an unencrypted area of the disk; and data leakage through ‘non-primary applications’ e.g. Google Desktop indexing data on hidden volumes.

In addition to these opportunities, [9] also highlights that if disk snapshots are available at close enough intervals then it is possible to detect the existence of hidden data since “seemingly random bytes on the hard drive will change”. However, no practical demonstration of this is provided and it is implied that intermittent or regular access is required to the disk.

The work described here shows how hidden volumes can be detected given only a single copy of a suspect’s disk, how information about the size and file system of the

hidden volume can be deduced, and demonstrates a visualisation of the results in the context of forensic computing, to allow such inferences to be explained to a court.

3 Methodology

3.1 General Methodology

Previous research has suggested that if multiple copies of an encrypted volume are available then detection of hidden volumes is possible [9], but at least intermittent access is needed to the disk. This paper provides a practical demonstration of how this is also possible from a single disk image by exploiting the Volume Shadow Copy functionality of Windows Vista and Windows 7. This feature is used to obtain multiple copies of encrypted containers produced using TrueCrypt version 6.3a. The term 'encrypted container' is used when referring to the use of Volume Shadow Copy since the technique cannot be used to obtain multiple versions of full volume or full disk encrypted data since Shadow Copies are created for files only. However, all other aspects of this research apply equally to full volume or disk encryption as long as intermittent access is available (this includes obtaining back-up copies), and in these aspects of the research the term 'volume' is used.

Once multiple copies of a volume are obtained, the paper shows how the changes can be visualised, making the presence of a hidden volume apparent in a form that is useful not only to an investigator but also to any non-expert decision makers to which the evidence needs to be presented.

Furthermore, the visualisation of changes that occur in the free space of a decrypted cover volume reveals patterns in the hidden volume which when combined with an understanding of file systems, can be used to infer information about the hidden volume. This is demonstrated by estimating the type and size of the hidden volume assuming a FAT based file system.

3.2 Obtaining Multiple Copies of an Encrypted Container

This section discusses how multiple versions of an encrypted container can be obtained by exploiting the Volume Shadow Copy feature of Windows Vista and Windows 7. This functionality extends the Restore Point feature of Windows XP so that backups are now created not just of important system files but also of user created files [11]. Shadow copies are not created every time a file is changed but when a Restore Point is created. In Windows Vista "restore points are created automatically every day, and just before significant system events such as the installation of a program or device driver" [12]. This means that previous versions of users' files may be available in addition to the current instance.

Forensic acquisition of data from Windows Vista Restore Points is discussed in detail in [13], but the process can be summarised as:

- Booting the suspect system as a clone or virtual machine
- Listing available Restore Points using the command line
- Mounting the restore points using symbolic links at the command line
- Copying out the mounted restore points to blank media

This creates copies of all files from a particular Restore Point. However, once a clone or virtualised version of the suspect system is booted, it is also possible to use the system's user interface to access previous versions of particular files of interest.

Experiments have shown that TrueCrypt encrypted containers are included in these automatic backups, but are not available through the GUI in the usual manner (shown in Figure 2). However, previous versions are available by examining previous versions of the folder in which the containers are stored (also shown in Figure 2). These folders can be restored and the multiple versions of the encrypted containers extracted.

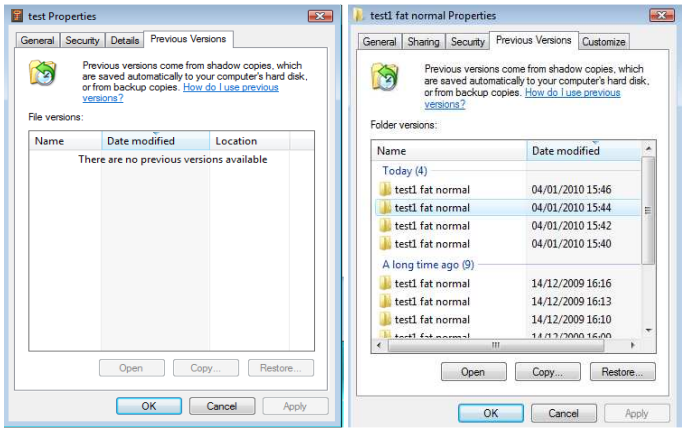

Fig. 2. While there are no Previous Versions visible for a TrueCrypt container (left), they can be accessed by examining previous versions of the containing folder (right).

Therefore, using this technique it may be possible to recover older copies of an encrypted container from a single disk image through the Previous Version functionality of Windows Vista and Windows 7. These multiple versions can then be examined further.

3.3 Visualising Changes

The previous section showed how multiple copies of an encrypted volume can be recovered from a single disk image due to the Volume Shadow Copy feature of Windows Vista and Windows 7. This section describes the further examination of these extracted encrypted volumes.

Multiple extracted encrypted volumes can be viewed using hex editors such as WinHex which offer synchronise and compare functionality. This allows multiple volumes to be compared and the differences highlighted. However, even if differences between two encrypted volumes are seen towards the end of the volumes, from this alone it is not possible to determine if these changes are due to a hidden volume or simply data being written near the end of a standard encrypted volume. However, assuming that some keys are available – either discovered during an investigation or provided as a result of an order placed under disclosure legislation - then decrypted versions of these volumes will be available, albeit they may not be the 'true' contents of the volume. If the volumes are decrypted and then examined in WinHex, since the file system can be interpreted it can be determined if random data is changing in the free space of the volume (Figure 3). As discussed in [9], if multiple versions of a volume can be examined then existence of the hidden volume is undeniable since "seemingly random bytes in the hard drive will change".

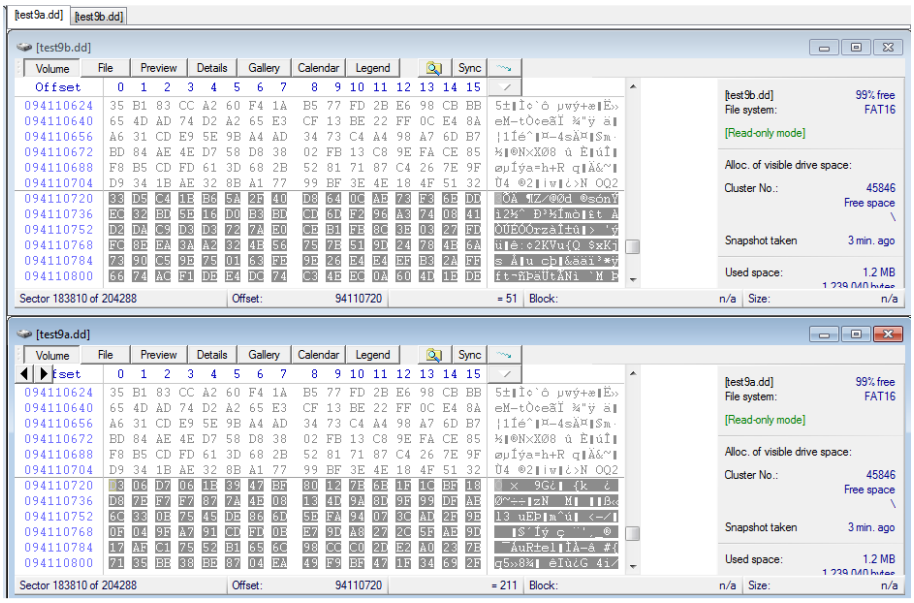

Fig. 3. WinHex highlighting the differences in the free space of two decrypted volumes.

While WinHex reports that the changes are occurring in free space of the volume, the nature of the changes is clearer if the volume is examined from a broader perspective; therefore the entire volume is visualised within a fixed space (a 100 x 100 grid). Blocks that are different between versions are highlighted. This is implemented in Python using the standard TkInter graphics library. Free space on the volume is identified using a simple entropy based test for randomness and is differently coloured; also, changes between the versions of the volume, and any changes since the first volume are highlighted. These are shown in Figure 5 and it can be clearly seen that changes are occurring in the free space of the volume.

The visualisation allows the changes to be clearly identified as being in the free space of the volume which indicates the presence of a hidden volume. This visualisation is useful not only for the analyst but also if this information needs to be presented to a court. Visualising changes between different versions of decrypted cover volumes in this way also reveals patterns in the changes to the hidden volume, from which additional information can be inferred.

3.4 Identifying the Size of the Hidden Container

Through visualisation of the decrypted volumes it is possible to infer additional information about the hidden volume. However, it is first necessary to discuss file system structures.

A FAT based file system stores data in clusters, which are the smallest unit of space on a volume which can be allocated to store data, typically 4096 bytes on a modern system. There are a number of variations of the FAT file system including FAT16 and FAT32 [14][15]. FAT file systems store data in a hierarchical structure consisting of directories that contain files and other directories, as are commonly viewed through operating systems. However, the directory entry contains only information about the file; the file content is saved elsewhere on the volume; although the directory entry does contain a pointer to the first cluster in which the file is stored.

Since a saved file may be larger than the size of a cluster, a file may occupy multiple clusters, which may not be contiguous. The file system therefore maintains a record of the 'cluster chains', i.e. given a start cluster it is possible to look up its entry and determine which cluster (if any) should be read next. These records of cluster chains are stored in a File Allocation Table (FAT), which contains an entry for every cluster on the volume. Since the FAT is essential to reading data from the volume, two copies of it are stored, both near the beginning of the volume, one after each other. The size of each entry in the FAT depends on the file system in use; in FAT16 each entry is 16 bits (2 bytes) and in FAT32 each entry is 32 bits (4 bytes).

Therefore, when a file is written to a volume, the FATs are updated to indicate that the cluster(s) in which the file is stored are no longer free. It is this property that is used to extract additional information from the hidden volume.

Since the FATs contain an entry for every cluster in the volume, the size of a FAT is proportional to the size of the volume. Therefore, if the size of the FAT can be determined, so can the size of the volume. Since the FATs are located near the beginning of a volume and follow each other, and since data written to a volume causes both FATs to be updated, it is possible to visually identify candidates for the two FATs; the two FATs are separated by a consistent value and change when data is written to the disk. This is can be seen in Figure 5.

Since the two FATs follow each other, the size of the FAT can be estimated from the difference between the two FATs. A simple example is shown in Figure 4.

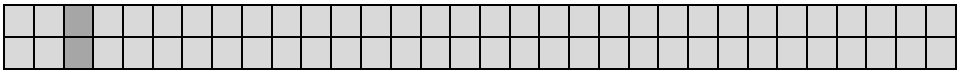

Fig. 4. A simple example highlighting the differences between two volumes.

In this example each block represents two bytes. If we assume that the two identified changes are in FAT 1 and FAT 2 then the size of the FAT is the difference between them (32 blocks = 64 bytes). Since the FAT represents all the clusters on the volume, from the size of the FAT it is possible to estimate the size of the volume. It is possible only to estimate the size of the FAT since the end of the first FAT may not align with a cluster boundary and is therefore padded. Also, since encryption is performed in sector sized blocks, changing a single bit in a sector results in the entire block changing and it is therefore not possible to determine the exact position of data changed within a block.

This idea is demonstrated by analysing a hidden volume within a cover volume, both created using TrueCrypt. In the following example a 42 Megabyte (43909120 bytes) hidden volume is created within a 100 Megabyte cover volume. Data are written to the hidden volume and a copy of the volume is made with each change. The visualisation of the changes is shown in Figure 5.

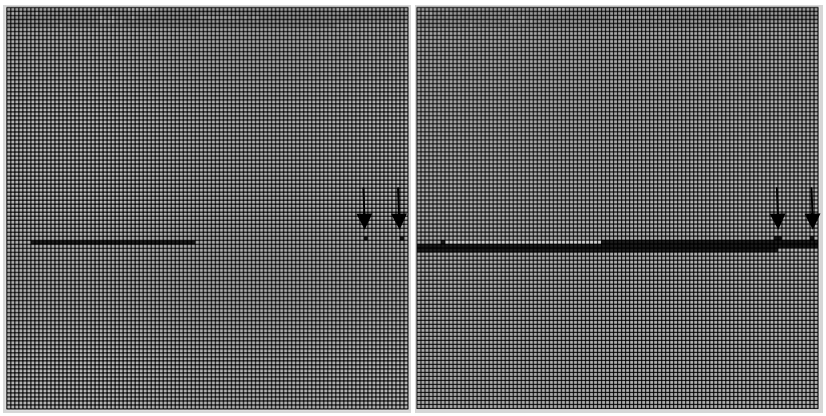

Fig. 5. A visualisation of the changes made to the test volume with the two FATs highlighted. The later contiguous blocks of changing data are the actual file contents being written to the volume.

The visualisation software allows the hypothesised positions of the two FATs to be marked and the size of the hidden volume to be estimated, as described above. However, in order to calculate the size of the hidden volume additional information is necessary.

Remembering that the FAT is an index of all the clusters on the disk, by matching up changes in the FAT with changes they represent in the data area of the hidden volume, it is possible to calculate the ratio between the size of the FAT entries and the size of the data that they index. For example in Figure 6, F_1 and F_2 represent points in FAT1 and using the visualisation tool it is possible to see the corresponding changes in the data area, marked D1 and D2. The visualisation tool can be used to identify the offsets of these:

$F_1 = 60556267$	$F_2 = 60557047$	$F_{diff} = 780$
$D_1 = 60743662$	$D_2 = 61147117$	$D_{diff} = 403455$

Therefore, 780 bytes in the FAT represents 403455 bytes of data, meaning 1 byte in the FAT represents approximately 517.25 bytes of data. Since each entry in the FAT represents one cluster and the default cluster sizes are powers of 2 (e.g. 512, 1024, 2048... bytes) [16], the actual ratio of FAT bytes to data bytes is 512.

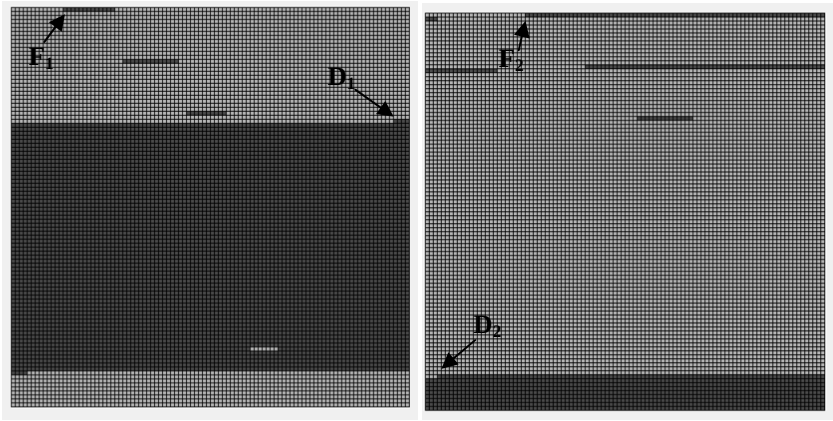

Fig. 6. A visualisation of the changes made to the test volume with points in the hidden volume's FAT and data area marked.

The size of the FAT can be estimated using the visualisation; in this case the difference between the entries in the two FATs is approximately 85kB (see Figure 7, below). The size of the volume can be calculated by multiplying the size of the FAT by the ratio of FAT bytes to data bytes (85514 * 512 = 43783168 bytes).

In addition to the size of the volume it is also possible to determine the FAT version (FAT16 or FAT32) and the cluster size in use. Since FAT16 and FAT32 have 2 and 4 bytes per FAT entry respectively, there are therefore either 42757 (FAT size/2) or 21378.5 (FAT size/4) FAT entries.

Since there are the same number of FAT entries as there are clusters, the possible cluster sizes can be calculated as volume size divided by the possible number of clusters, giving either 1024 (43783168/42757) or 2048 (43783168/21378.5). From [16] it is possible to determine which of these two possibilities are valid for the volume size identified; in this case it is FAT16 with a cluster size of 1024 bytes.

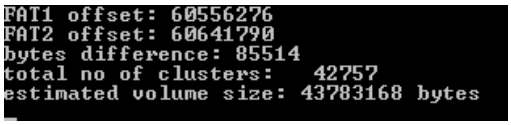

Fig. 7. Successful calculation estimating the size of the hidden volume embedded in a decrypted cover volume (actual size is 43909120 bytes)

4 Evaluation

While the previous section has shown that the existence and the size of the hidden volume can be determined there are limitations to this approach.

In this paper, the multiple copies of volumes have been obtained using the Volume Shadow Copy feature of Windows Vista and Windows 7. Firstly this is only an option for encrypted containers and cannot be used for encrypted volumes or disks. Also, if the System Restore feature is turned off then these will not be available; although this obviously removes all the benefits that System Restore provides. However, the developed visualisation technique does not necessarily require Shadow Copies of encrypted volumes and multiple versions of the volumes from any source can be used e.g. from external backups. Therefore, if external backups are available then the visualisation technique could also be used with full volume or full disk encryption.

By visualising the changes that occur in the free space of a decrypted cover volume and using knowledge of file system structures is has been possible to identify the FATs, estimate the size of the hidden volume and derive the FAT version in use and the cluster size. In future it should also be possible to identify hidden volumes formatted using NTFS although the patterns of changes are different due to the use of a Master File Table rather than FATs.

It should be emphasised that this technique is dependent on identifying patterns in the underlying file system. While this is possible for the current version of TrueCrypt there are other steganography encryption systems where this would not be possible, e.g. Rubberhose [17] which does not store the file system in a linear manner.

The value of this work to a forensic investigation is that this technique allows investigators who believe they have recovered encryption keys to assess the likelihood of deliberate deception. In the UK there is legislation in place to encourage suspects to provide decryption keys; therefore demonstrating the existence of a hidden container is sufficient to allow non-technical measures to be used to further an investigation. Also, the estimation of the size of a volume can be used to give an idea of how much data could be hidden; while this not essential for a RIPA prosecution, it may add weight to a case against a defendant for refusing to supply decryption keys.

5 Future Work

In addition to estimating the size of the volume, since the Shadow Copies have associated dates and times, it may be possible to determine the amount of data written to a hidden volume between two dates. This may then be correlated with other sources of digital evidence, for example records from Internet Service Providers.

This visualisation technique can be extended to examine other file systems including NTFS, EXT3 and HFS+ and it may be possible to identify other information about the hidden volume in addition to determining the size of the hidden volume.

It is also desirable to determine additional information on FAT file systems, for example the position of the boot sector. This is particularly useful since it is used in known-plaintext-based decryption key recovery approaches such as [18].

6 Conclusions

There are several approaches to addressing the problem of encrypted evidence, and in the UK, legislation has been passed to encourage decryption keys to be provided by the suspect. However, there are technical measures that can be employed to counter this legislative approach, including the use of hidden volumes. This paper has demonstrated how these technical measures can be overcome by acquiring multiple copies of an encrypted container from a single disk image using the Volume Shadow Copy feature of Windows Vista and Windows 7. It has also shown how these multiple copies can be used to detect the presence of hidden volumes within a standard encrypted volume. A visualisation of the volume and the changes to that volume can be used to infer additional information about the hidden volume, for example to estimate its size. Demonstrating the existence of hidden volumes is useful in an investigation since it allows investigators who believe they have recovered encryption keys to assess the likelihood of deliberate deception, potentially motivating further non-technical investigative measures.

References

1. Casey, E.: Practical Approaches to Recovering Encrypted Digital Evidence. International Journal for Digital Evidence **1** (2002)
2. Wolfe, H.B.: Encountering Encrypted Evidence (Potential). Proceedings of the 4th Conference on Information Technology Curriculum (2002)
3. Wolfe, H.: Encountering Encryption. Computers and Security **22** (2003) 388-391
4. Wolfe, H.: Penetrating Encrypted Evidence. Digital Investigation **1** (2004) 102-105
5. United Kingdom: Regulation of Investigatory Powers Act 2000. HMSO (2000)
6. Home Office: Investigation of Protected Electronic Information: Code of Practice. (2007)
7. TrueCrypt: TrueCrypt Documentation. http://www.truecrypt.org/docs/ (2009)
8. TrueCrypt: TrueCrypt Documentation: Hidden Volume. (2009)
9. Czeskis, A., Hilaire, D.J.S., Koscher, K., Gribble, S.D., Kohno, T., Schneier, B.: Defeating encrypted and deniable file systems: TrueCrypt v5. 1a and the case of the tattling OS and applications. (2008)
10. Craiger, J.P., Pollitt, M., Swauger, J.: Law Enforcement and Digital Evidence. http://ncfs.org/craiger.delf.revision.pdf (2005)
11.Microsoft: Learn about the features: Shadow Copy. http://www.microsoft.com/windows/products/windowsvista/features/details/shadowcopy.mspx (2007)
12. Microsoft: System Restore: frequently asked questions. (2008)
13. Titheridge, D.: Microsoft Windows Vista Registry. MSc. Cranfield University (2009)
14. Carrier, B.: File System Forensic Analysis. Addison Wesley (2005)
15. Sammes, T., Jenkinson, B.: Forensic Computing: A Practitioners Guide, Second Edition. Springer (2007)
16.Microsoft: Default cluster size for NTFS, FAT, and exFAT http://support.microsoft.com/kb/140365 (2009)
17. Assange, J., Weinmann, R.P., Dreyfus, S.: Rubberhose. http://iq.org/~proff/marutukku.org/ (Undated)
18. Hargreaves, C., Chivers, H.: Recovery of Encryption Keys from Memory Using a Linear Scan. The International Workshop on Digital Forensics, Barcelona, Spain (2008)