

Title: Xbox Forensics

Authors:

Paul K. Burke

paulkburke@gmail.com

407-823-0965

J. Philip Craiger

philip@craiger.net

407-823-3527

Fax:

407-823-3162

Address:

National Center for Forensic Science

University of Central Florida

Post Office Box 162367

Orlando, Florida 32816-2367

# XBOX FORENSICS

**Paul Burke and Philip Craiger**

National Center for Forensic Science  
Department of Engineering Technology  
University of Central Florida

Microsoft's Xbox game console is little more than a low-end personal computer; with little effort it can be modified to run additional operating systems, enabling it to store gigabytes worth of non-game related files in addition to allow it to run various computer services. Little has been published, however, on the proper forensic procedures to determine whether an Xbox has been modified, and if so, how to create a forensic duplicate and conduct a proper digital forensics investigation. Given the growing popularity of these devices, it will be important to understand how to identify, image, and examine these systems while reducing the potential of corrupting the media. We approach Xbox forensics from an applied research methodology, providing a set of forensically sound procedures to be followed during the acquisition and subsequent analysis of an Xbox.

Keywords: Digital forensics, computer forensics, Xbox, Xbox consoles, game consoles

## Introduction

The fine line between personal computers and video game consoles was blurred with the release of Microsoft's Xbox gaming system in 2001. Hobbyists have expanded the uses for the Xbox by loading the Linux operating system on the Xbox, functionally transforming it into a low-end personal computer (PC). With this modification the Xbox has the potential to provide services provided by any PC: a file server, a Web server, or a multimedia hub for a television and stereo system.

A modified (or "modded") Xbox can provide a great challenge in a computer crime investigation. Visually it is difficult to determine if an Xbox has been modified to run additional operating systems or if is storing non game-related files (which may be of probative value in a criminal or civil case). Unfortunately, there are no established procedures as to a) how to identify whether an Xbox has been modified; b) how to create a forensic duplicate of the storage media; and c) how to differentiate known-good files from files of notable content that may reside on Xbox storage. Below we provide guidelines for an investigator to aid in determining if an Xbox contains evidence and how to extract such evidence in a forensically sound manner. We first describe the history and internal workings of the Xbox, moving on to said procedures and noting their limitations.

## Background

Microsoft launched the Xbox gaming console on November 15, 2001 (Wikipedia, 2007). Cumulative sales of the Xbox system have reached 24 million worldwide (Microsoft, 2006). Unlike its console competitors, the PlayStation 2 and the GameCube, the Xbox largely relies on stock PC hardware modified for use as a game console. Every Xbox contains similar equipment to what one would expect in a PC: a hard drive, a DVD drive, dedicated graphics hardware with TV-out, Ethernet and USB (albeit via a custom connector).

As with many popular electronics consumer devices, computer hobbyists have modified the original purpose of the Xbox to include uses not intended by the developers. One of the most significant efforts was to modify the Xbox to run an operating system other than the one built in to the system. In doing so the Xbox functionally becomes identical to a low-end personal computer.

Microsoft implemented several security measures within the Xbox to thwart would-be hardware hackers from running non-approved software. These measures work together to create a chain of trust between each step of software execution. The exact methods are described in Steil's "17 Mistakes Microsoft Made in the Xbox Security System" (Steil, 2005). Of greatest concern to the investigator are two elements in the system: the hard drive password protection and the file system used on Xbox media. The Xbox employs an obscure option offered within the ATA specification: the Security Feature Set (Bögeholz, 2005). The Security Feature Set allows the computer to lock the hard drive using two 32-byte passwords. Once locked, the hard drive will not respond to requests to access its data and will cause Windows and Linux to generate errors if one attempts to access data on the disk. In the case of the Xbox, the drive is locked in a manner where only one password is required to unlock the disk. This password is cryptographically generated from a unique string located in the Xbox ROM coupled with the serial and model numbers of the hard drive in the Xbox (Speedbump, 2002). There are several ways around the

locking mechanism, some of which will be described later. It is important to note that in most cases the Xbox will refuse to boot with an unlocked drive in it and that a locked drive from one Xbox will not function on another.

The data on a normal Xbox hard drive consists of the operating system files, game cache files, and free space for saved game files. The disk itself does not have a partition map *per se*; it is believed that the offsets are preprogrammed into the Xbox itself (de Quincey, 2002). Microsoft designed a derivative of the FAT file system, called FATX, for use on the Xbox. FATX bears a strong resemblance to FAT32, containing only minor changes to the file system layout (Steil, 2003). These changes are significant enough to prevent forensic utilities designed to read FAT32 from reading FATX, however.

## Prior Work

There are two notable papers previously written regarding forensic issues dealing with an Xbox. Both papers describe the basic background knowledge required for Xbox forensics, which we have covered above. “Xbox security issues and forensic recovery methodology (utilising Linux)” by Chris Vaughan (Vaughan, 2004) covers in detail the background behind Xbox modification and provides several methods of bypassing the ATA password protection on the Xbox hard drive. Vaughan also describes the procedures necessary for building a FATX-enabled kernel and performing Linux-based forensics on a FATX image. “Defeating Xbox (utilizing DOS and Windows tools)” by Daniel Dementiev (Dementiev, 2006) largely mirrors this coverage, instead providing methods for data extraction from within Windows.

Our goal is to compliment and extend these works. Specifically, we deal with issues regarding the forensic validity of imaging methods and attempt to provide the most straightforward method of data extraction. Previous works suggested modifying the state of the hard drive (permanently unlocking it) or installing hardware in the Xbox in order to access the contents of the disk; our methods provide the least-

intrusive path known and require no low-level physical interaction with the machine. Our goal is to provide a forensically sound set of procedures that can be followed by an investigator when acquiring and analyzing an Xbox. We approach the topic of Xbox forensics from an applied research method; what follows are our discoveries related to the Xbox in a forensic context and a series of recommendations to follow when dealing with these devices.

## Forensic Procedures

As stated above, our research is aimed at providing forensically sound procedures for an investigator to follow upon the discovery of an Xbox. These procedures assure that the digital evidence acquired has not been tainted or modified in any way. We have separated the procedures into three primary steps: initial assessment, creating a forensic duplicate of the Xbox storage media, and analysis of the storage media. Initial assessment covers the beginning stages of the investigation, including how to determine if an Xbox has been modified. Once the Xbox is confirmed as being modified, we discuss how to build an analysis and acquisition workstation and how to image the Xbox in a forensically sound manner. Finally, methods of analyzing the acquired data from both a logical and a physical approach are discussed.

Linux is an ideal operating system to perform analysis of an Xbox as there is a FATX file system driver available from the Xbox Linux project. This means that images of the Xbox disk partitions can be mounted and interacted with in Linux natively, without using external programs. Several alternatives for reading FATX volumes exist in Windows, but our tests demonstrated that they by and large do not support logical analysis at a level necessary for forensics, for instance, not supporting file times or providing a file listing interface. Such tools may also prevent forensic suites such as EnCase and The Forensic Toolkit from properly representing the media at a logical level, as they do not present the volumes as proper Windows drives. Furthermore, these forensics suites require the disk to be extracted from the Xbox, unlocked, and connected to the analysis machine. As noted above the methods we

employ and demonstrate here do not require any interaction with hardware. Our procedures are based on the assumption that a modified Xbox is involved and that the examiner has an understanding of the Linux operating environment. The acquisition process involves booting the Xbox from a Linux-based CD and extracting the partition data over a local network to the analysis machine. The analysis machine is then modified to be able to process the Xbox FATX file system. These procedures are discussed in more detail below. We first deal with methods for determining if an Xbox console has been modified.

It should be noted that the test base for analyzing modified Xboxes was quite small for this research and no effort was made to be complete and cover procedures for individual types of modifications. Tests were performed on a version 1.6 Xbox compromised via the Ndure exploit (or “soft modded,” see below). Linux was loaded on the primary partition of the Xbox as test data. Despite the number of tests performed we believe that these methods should function on the majority of modified Xbox units in the wild as such modifications should behave similarly relative to our methods.

## Initial Assessment

Before an examiner can begin an analysis, he or she must first determine if the Xbox has been modified to run non-Microsoft programs. There are two primary modification methods; one can either physically modify the Xbox (called “hard modding”), overriding the built-in protections against the execution of foreign code, or break these software protections using code exploits (“soft modding”). For the purposes of this paper the two options functionally have the same effect: once modified, the Xbox will execute programs regardless of their origin, including Linux. Therefore, we do not differentiate between these two methods in latter sections of the paper.



Figure 1. Xbox screw locations.

Hardware modification leaves the most obvious trace that tampering has occurred. This method requires that the Xbox console be physically opened in order to install the necessary circuitry, often a replacement BIOS chip. The six screws that hold the case together are located on the bottom of the case and are concealed by a combination of stickers and rubber pads; the removal or destruction of either is immediately evident when one observes the bottom of the unit (see Figure 1). Another simple method to determine if the Xbox has been modified is to observe its surroundings: if it is connected to a network and running without a connection to a TV it is likely that it is being used as a server. USB-to-Xbox controller adapters with traditional USB keyboards and mice nearby are also suspect, as these allow the user to interact with the machine like a traditional PC. The presence of a network cable connected does not necessarily imply modification, as a legitimate Microsoft service called Xbox Live utilizes the Internet for gaming. Other modifications such as custom cases do not necessarily indicate lower-level hardware modification, although those with the expertise to make such customizations may be more likely to attempt such modification.

A general method to determine if an Xbox has been modified, either through soft or hard modding, is by attempting to boot a Linux-based boot CD on the unit. Our tests have shown that the hash of the hard drive does not change when doing so, provided Linux is executed immediately; while portions of the Xbox software are often loaded before Linux (such as the routine to temporarily unlock the hard drive) they do not appear to modify the disk contents in any way. Our tests indicated that only read operations are performed at that point of the boot process. That said, we did note that the Xbox would reformat any

memory units attached to the controllers at startup if nonstandard data is found on them; we therefore advise removing all memory units before booting the Xbox. We describe these memory units in more detail below.

It should be noted that if the Xbox has been unplugged from a power outlet for a short amount of time the internal clock will reset as the Xbox uses a capacitor instead of a battery for its internal PC clock. Reports indicate that the capacitor charge will last at least an hour; if the Xbox must be removed from its environment during a seizure it should be immediately plugged into an outlet as soon as possible. A reset of the internal clock will not modify the hard drive, but it has the potential to interfere with some soft modifications available (Xbox-Linux 1, 2006). This may prevent loading Linux even if the Xbox has been modded. If the clock has been reset, a dialog box will appear at boot before any Xbox or Linux software is loaded. If immediately dismissed (by pressing the A button on the controller) the clock will not be set and the dialog box reappears at next boot; this is the advisable course of action if the capacitor was accidentally drained.

In developing and testing our procedures we used a Xebian (Xbox-Linux 1, 2007) boot CD (Xbox version) specially designed for the Xbox. According to the Xbox Linux project, the CD/DVD drives on Xboxes are frequently of inferior design and often have difficulty reading CD-Rs (Xbox-Linux 2, 2006). When burning a boot CD it is recommended that the examiner burn the CD ISO to a DVD-R instead, as it forces the Xbox to rely on a different laser to read the disc. The boot CD will function normally despite the difference in media. We recommend testing the burned DVD-R on a modified Xbox before analysis to make certain that the disc was burned properly. If one attempts to boot an unmodified Xbox with Xebian or any other non-Microsoft disc they are presented with the error:

*“Your Xbox can’t recognize this disc. Make sure it is an Xbox game, DVD movie, or audio CD. Also check to see if the disc is dirty or damaged. Remove the disc to continue.”*



Linux 2, 2007), under the “Linux” category.

The latest Linux kernels can be downloaded from kernel.org; make sure the version of the kernel matches the patch downloaded from the Xbox-Linux site. With both downloaded, decompress the kernel tarball and apply the patch:

```
# tar jxf linux-2.4.32.tar.bz2
# cd linux-2.4.32
# zcat ../linux-2.4.32-xbox-patch.gz | patch -p1
```

Depending on the distribution chosen the process for building a custom kernel can vary, but most distributions support the traditional method of compiling the kernel. The process for compiling the 2.4 kernel is as follows:

```
# make xconfig      (or make menuconfig)
# make dep
# make clean
# make bzImage
# make modules
# make modules_install  (must be performed as root)
# make install        (must be performed as root)
```

When configuring the kernel, make sure that CONFIG\_FATX\_FS and CONFIG\_BLK\_DEV\_LOOP are either built-in to the kernel or modularized. As with any kernel upgrade, make sure your bootloader has been updated to load the new kernel.

## Creating a Forensic Duplicate

Once the investigator has determined that an Xbox has been modified, the next step is to create a forensic duplicate of the hard drive. After the Xbox has successfully booted into Xebian and the analysis workstation is prepared, connect the two using a crossover Ethernet cable. The default IP address of Xebian is 192.168.0.10/24; set the IP address of the analysis workstation to anything within that subnet. You can now use SSH to connect as root to the Xbox using the aforementioned IP address (The password for root is “xebian”). Once at the prompt information about the individual Xbox can be extracted by

running `xbox_info -a`.

The Xbox partition structure appears slightly different from a normal PC when viewed in Linux. Partitions are enumerated from minor node 50 up; a factory-default Xbox will have the partition layout described in Table 1.

<b>Device Name</b>	<b>Size (Megabytes)</b>	<b>Size (Bytes)</b>
/dev/hda50	4882 MB	5120024576 B
/dev/hda51	500 MB	524288000 B
/dev/hda52	750 MB	786432000 B
/dev/hda53	750 MB	786432000 B
/dev/hda54	750 MB	786432000 B

Table 1. Xbox Partition Layout

One can create forensic duplicates of the individual partitions from the analysis machine over SSH using commands similar to the following:

```
# ssh root@192.168.0.10 "dd if=/dev/hda50" > xbox-50.dd
```

We recommend imaging individual partitions instead of the entire disk to make mounting the partitions easier as Linux does not have native support for mounting partitions embedded within an image. Once the acquisition is complete hashing can be performed with the `md5sum` utility on both the analysis machine and the Linux CD running on the Xbox. The command `ls -l /dev/hda*` can be used to determine if any other (user added) partitions exist on the hard drive. Some Xbox hard drives are shipped with 10 GB (as opposed to the original 8 GB) drives; some users have partitioned this extra 2 GB for use in Linux. Multiple extra partitions may also show if a new, larger hard drive was installed in the Xbox. The latter instance requires a hardware modification of the Xbox to work properly.

The Xbox memory units which store saved games can provide a challenge to the investigator. Memory units are small (typically 8 MB) flash storage devices that are designed to plug in to the Xbox controller and provide portable storage for saved games. Our tests revealed that the Xbox will format memory units at boot that appear to have non-save game data (e.g, personal documents or pictures) on them. This

format appears to be what is commonly referred to as a “quick format”, as not all data is zeroed out. Regardless, the operation is destructive. Theoretically this allows for a booby-trap to be set: Data can be placed on the memory unit and left plugged in; if another party attempts to access it they may unwittingly destroy the saved information before it can be recovered. The formatting of the memory unit appears to completely wipe the file allocation tables and directory entries of the previous FATX file system; at least some of the data area is left intact, however. In a scenario where such a formatting has occurred we have used tool such as foremost (Foremost, 2007) to successfully extract the remaining file data. This formatting behavior should only be present in Xboxes which have not been hardware modified to load a replacement Dashboard at boot, but this has not been tested.

As noted by Dementiev (Dementiev, 2006), if a memory unit is placed in an attached controller before boot, Xebian Linux will acknowledge the device and present it as a USB mass storage device. These are enumerated through SCSI emulation so they will appear as /dev/sda, /dev/sdb, etc. and can be imaged in the same manner as a partition. However, as we have noted above, if the Xbox is booted in this way the units are subject to possible formatting. Furthermore, attempts to hot plug Microsoft memory units in after booting into Linux have failed in our testing. Despite our efforts we were not able to find a forensically sound procedure for imaging a memory unit. Due to the formatting issues above, we attempted imaging the memory units using a USB adapter included with Datel’s Action Replay for the Xbox, which revealed several issues. One issue stems from the fact that Microsoft-branded memory units do not present themselves as proper USB storage devices. Most USB mass storage devices transmit a bInterfaceSubClass field of 0x06; this denotes that a SCSI transparent command set should be used to interface with the device (USB Implementers, 2003). Instead, Microsoft memory units transmit a code of 0x66. This alone will confuse most operating systems that attempt to interface with the unit, as no transport protocol is officially assigned to the transmitted code.

Attempts to bypass this behavior revealed that these memory cards also do not respect the established

storage protocols. Forcing Linux to interact with the device using the SCSI transparent command set failed; our discussions with a Linux kernel developer revealed that the Microsoft memory units are not standard mass storage devices. Some research has already gone into the problem, as the Xbox-Linux 2.4 kernel patch includes some basic modifications to the USB stack for these memory units to register. However, our attempts to use the patch on a device other than the Xbox resulted in the device not properly registering. Some memory units (such as the one included with Datel's Action Replay) will act as proper devices and can be imaged through the Action Replay USB adapter or by hot plugging the memory unit into a controller after boot. All Xbox memory units formatted by the Xbox have no partition layout.

## Logical Analysis

Once the partitions have been acquired and transferred to the analysis machine they can be mounted for analysis. Each partition can be mounted through the loopback device read-only as follows:

```
# mount -t fatx -o ro,loop xbox-50.dd /mnt/xbox-50
```

The partition can then be treated as a local file system, as demonstrated below. The file listing is representative of an Xbox partition that has had the GentooX Linux distribution (GentooX, 2007) installed on it.

```
# cd /mnt/xbox-50
# ls -l
total 4365792
drwxr-xr-x  2 root root      16384 Nov 15  2001 CACHE
drwxr-xr-x  4 root root      16384 May  8  2004 TDATA
drwxr-xr-x  4 root root      16384 May  8  2004 UDATA
drwxr-xr-x  3 root root      16384 Jul 22  2006 apps
drwxr-xr-x  6 root root      16384 Jul 22  2006 backup
drwxr-xr-x  2 root root      16384 Jul 22  2006 dash
drwxr-xr-x  2 root root      16384 May 15  2005 gentoox
-rwxr-xr-x  1 root root    337024 May 15  2005 gentooxx.xbe
-rwxr-xr-x  1 root root    236295 May 15  2005 initrd.gz
-rwxr-xr-x  1 root root       285 May 15  2005 linuxboot.cfg
drwxr-xr-x  3 root root      16384 Jul 22  2006 ndts
-rwxr-xr-x  1 root root 4194304000 May 15  2005 rootfs
-rwxr-xr-x  1 root root    4827622 May 15  2005 sprkrd.gz
-rwxr-xr-x  1 root root   268435456 May 15  2005 swap
-rwxr-xr-x  1 root root    2256805 May 15  2005 vmlinuz
```

At this point traditional Linux forensics tools can be employed to examine the images for content (Craiger, 2006). For example, one can generate a timeline of file access events using The Sleuthkit and macrobber (Carrier, 2006):

```
# cd /mnt/xbox-50
# mac-robber . > ~/mac-robber-hda50
# mactime -b ~/mac-robber-hda50 > ~/mactime-hda50.txt
```

The output of the mactime program contains a list of file Modified, Accessed and Created (MAC) times, as demonstrated below.

```
# cat ~/mactime-hda50.txt
[Output truncated]
Sun May 15 2005 13:33:42 236295 mac -rwxr-xr-x ./initrd.gz
337024 mac -rwxr-xr-x ./gentooxx.xbe
Sun May 15 2005 14:45:48 285 mac -rwxr-xr-x ./linuxboot.cfg
Sun Jun 04 2006 18:05:46 68096 ma. -rwxr-xr-x ./CACHE/LocalCache02.bin
33792 ma. -rwxr-xr-x ./CACHE/LocalCache01.bin
Sat Jul 22 2006 14:56:08 16384 ..c drwxr-xr-x ./UDATA
512 ..c -rwxr-xr-x ./CACHE/LocalCache00.bin
16384 ..c drwxr-xr-x ./TDATA/fffe0000
68096 ..c -rwxr-xr-x ./CACHE/LocalCache02.bin
33792 ..c -rwxr-xr-x ./CACHE/LocalCache01.bin
512 ..c -rwxr-xr-x ./CACHE/LocalCache08.bin
16384 ..c drwxr-xr-x ./CACHE
16384 ..c drwxr-xr-x ./TDATA
```

The first column of output contains a timestamp that correlates to activity on one or more files. The second column lists each files size in bytes. The third column shows any combination of m, a, or c to correspond with the modified, accessed and created times; if a letter is present it means that the timestamp that it represents was changed at the date in the first column. The fourth column shows the permissions of the file (which are not applicable for a FATX file system) in addition to if the file is a directory or not (the leading 'd' in the string of characters). Columns five, six, and seven are not applicable in analysis of a FATX file system (and were removed from the listing to save space). The final column lists the file path and name.

If individual files are identified as being of interest, they can be easily copied off of the mounted file system from within Linux. If one is feeling particularly inventive it is possible to use the Samba daemon

(Samba, 2007) to share the contents of a mounted partition's directory over the network. The Samba configuration (smb.conf) can be modified to add the directory in question as a read-only share, as shown below.

```
[xbox-hda50]
browseable = yes
read only = yes
guest ok = yes
path = /mnt/xbox-50
```

The analysis computer may then be attached to a Windows computer, which will see the partition as a network drive. Tools such as EnCase and The Forensics Toolkit running on the Windows computer can then be used to import and analyze the information on the partitions. It should be noted that this is a logical analysis so operations such as string searches within these tools will reveal not locate any content in unallocated space on the partition.

With this paper we are publishing a set of file hashes we found to be consistent between two Xbox units (version 1.6). These should help eliminate some of the files from consideration in a logical analysis; they are known-good files and can be ignored when processing the value of the file set found on the partitions. The hash list can be downloaded from [ncfs.org](http://ncfs.org) (Burke, 2006).

## Physical Analysis

Because the above analysis cannot detect evidence in the form of deleted files, stray data or information in slack space, it is necessary to analyze each partition image physically as well. Within Linux one can employ utilities such as xxd, strings, grep, and foremost to identify case-related information.

Some information about the file system can be extracted within a hex editor; searching for 0xE5 (the marking for a deleted file; see Figure 3) among the directory entries should make deleted entries apparent. Steil's analysis of the FATX file system (Steil, 2003) describes the binary structure of these entries. xxd

is one such hex viewer available in Linux, usually included with the ubiquitous UNIX editor vim (Moolenaar, 2007).

```
00025f0: 0000 0000 0000 0000 0000 0000 0000 0000 .....
0002600: e559 4649 4c45 2020 5458 5420 1814 2056 .YFILE TXT .. V
0002610: e532 e532 0000 d154 e532 0200 9334 0000 .2.2...T.2...4..
0002620: 0000 0000 0000 0000 0000 0000 0000 0000 .....
```

Figure 3. xxd showing a deleted file entry.

The strings utility is included in the GNU binutils package and should be installed by default on all Linux systems. When run on the Xbox partition image it will return printable character content that maybe in English. It can provide a good starting point to determine if any evidence exists on the partition in ASCII form.

GNU grep allows the user to search for strings located in a file based on regular expressions. When coupled with xxd, it can be used to locate the area of the file where the string occurs:

```
# xxd xbox-50.dd | grep -i 'credit cards'
```

It should be noted that this example will not identify words or phrases that span two lines of console output.

Finally, foremost may be employed to extract specific file types from the image, based on their file signature. Foremost relies on scanning an image for file information unique to that type of file and then extracting the data following that information. This is independent of any file system, and the utility will recover deleted files if present. While not perfect, foremost is capable of recovering data which otherwise would only be retrievable using a hex editor.

## Conclusions

Xbox forensic analysis is still in its infancy. With the hardware of a personal computer, the Xbox should be considered a valid device for potential evidence at a crime scene. Yet despite this no major forensic

suite, as of early 2007, is able to support the FATX file system and no simple method exists to extract the information from an Xbox hard drive in all instances.

The procedures presented here are by no means exhaustive; due to the numerous revisions of Xbox models along with the abundance of modification methods there is a strong possibility of these methods not working as intended. Our research was focused on soft-modded Xboxes that also limited the test base. Finally, the logical analysis as demonstrated prevents a simple extraction of deleted files and slack space, greatly limiting the ease with which an investigation is performed.

Even as Xbox forensics is being recognized, new devices are coming to market. Microsoft's Xbox 360 and Sony Entertainment's PlayStation 3 are two new consoles that build on the success of their predecessors; while there are currently efforts underway to run Linux on the Xbox 360, the PlayStation 3 supports Linux out of the box. Both consoles will provide additional challenges to investigators as common users become increasingly aware of the unconventional potential of these devices.

## References

Bögeholz, H. (2005). At your disservice: How ATA security functions jeopardize your data. *c't Magazine*, 172. (<http://www.heise.de/ct/english/05/08/172/>).

Burke, P. & Craiger, P. (2006). Xbox media MD5 hash list. Retrieved February 27, 2007, Web site: <http://www.ncfs.org/burke.craiger-xbox-media-hashlist.md5>

Carrier, B. (2007). The Sleuth Kit. Retrieved February 27, 2007, Web site: <http://www.sleuthkit.org/>

Craiger, P. (2006). Recovering evidence from a Linux system. *Advances in digital forensics*, M. Pollitt and S. Sheno (Eds.), New York: Springer.

Dementiev, D. (2006). Defeating Xbox (utilizing DOS and Windows tools), Unpublished (acquired via personal communication).

Foremost. Retrieved February 27, 2007, Web site: <http://foremost.sourceforge.net/>

GentooX. Retrieved February 27, 2007, Web site: <http://gentoox.shallax.com/>

Microsoft Corporation, (2006, May 10). News - 20060510. Retrieved February 27, 2007, from Xbox.com Web site: <http://www.xbox.com/zh-SG/community/news/2006/20060510.htm>

Moolenaar, B. (2007). vim online. Retrieved February 27, 2007, from vim.org Web site: <http://www.vim.org/>

de Quincey, A. & Murray-Pitts, L. (2002). Xbox partitioning and filesystem details. Retrieved February 27, 2007, from Xbox-Linux Web site: [http://www.xbox-linux.org/wiki/Xbox\\_Partitioning\\_and\\_Filesystem\\_Details](http://www.xbox-linux.org/wiki/Xbox_Partitioning_and_Filesystem_Details)

The Samba Project. Retrieved February 27, 2007, Web site: <http://www.samba.com/>

SpeedBump, (2002). Xbox hard drive locking mechanism. Retrieved February 27, 2007, from Xbox-Linux Web site: [http://www.xbox-linux.org/wiki/Xbox\\_Hard\\_Drive\\_Locking\\_Mechanism](http://www.xbox-linux.org/wiki/Xbox_Hard_Drive_Locking_Mechanism)

Steil, M. (2003). Differences between Xbox FATX and MS-DOS FAT. Retrieved February 27, 2007, from Xbox-Linux Web site: [http://www.xbox-linux.org/wiki/Differences\\_between\\_Xbox\\_FATX\\_and\\_MS-DOS\\_FAT](http://www.xbox-linux.org/wiki/Differences_between_Xbox_FATX_and_MS-DOS_FAT)

Steil, M. (2005). 17 mistakes Microsoft made in the Xbox security system. Retrieved February 27, 2007, from Xbox-Linux Web site: [http://www.xbox-linux.org/wiki/17\\_Mistakes\\_Microsoft\\_Made\\_in\\_the\\_Xbox\\_Security\\_System](http://www.xbox-linux.org/wiki/17_Mistakes_Microsoft_Made_in_the_Xbox_Security_System)

USB Implementers Forum, (2003). Universal Serial Bus Mass Storage Class Specification Overview (Rev. 1.2). Retrieved February 27, 2007, from [http://www.usb.org/developers/devclass\\_docs/usb\\_msc\\_overview\\_1.2.pdf](http://www.usb.org/developers/devclass_docs/usb_msc_overview_1.2.pdf)

Vaughan, C. (2004). Xbox security issues and forensic recovery methodology (utilising Linux). Digital Investigation. 1, 165-172.

Xbox. (2007). In Wikipedia [Web]. Retrieved February 27, 2007, from <http://en.wikipedia.org/wiki/Xbox>

Xbox-Linux Project (1), (2006). Clock loop problem HOWTO. Retrieved February 27, 2007, from Xbox-Linux Web site: [http://www.xbox-linux.org/wiki/Clock\\_Loop\\_Problem\\_HOWTO](http://www.xbox-linux.org/wiki/Clock_Loop_Problem_HOWTO)

Xbox-Linux Project (1), (2007). Xebian. Retrieved February 27, 2007, from Xbox-Linux Web site: <http://www.xbox-linux.org/wiki/Xebian>

Xbox-Linux Project (2), (2006). Xbox Linux boot CD/DVD burning HOWTO. Retrieved February 27, 2007, from Xbox-Linux Web site: [http://www.xbox-linux.org/wiki/Xbox\\_Linux\\_Boot\\_CD/DVD\\_Burning\\_HOWTO](http://www.xbox-linux.org/wiki/Xbox_Linux_Boot_CD/DVD_Burning_HOWTO)

Xbox-Linux Project (2), (2007). Linux on the Microsoft Xbox. Retrieved February 27, 2007, from Sourceforge.net Web site: <http://sourceforge.net/projects/xbox-linux>