Chapter 19

FORENSIC ANALYSIS OF XBOX CONSOLES

Paul Burke and Philip Craiger

Abstract Microsoft's Xbox game console can be modified to run additional operating systems, enabling it to store gigabytes of non-game related files and run various computer services. Little has been published, however, on procedures for determining whether or not an Xbox console has been modified, for creating a forensic duplicate, and for conducting a forensic investigation. Given the growing popularity of Xbox systems, it is important to understand how to identify, image and examine these devices while reducing the potential of corrupting the media. This paper discusses Xbox forensics and provides a set of forensically-sound procedures for analyzing Xbox consoles.

Keywords: Xbox consoles, forensic analysis

1. Introduction

The fine line between personal computers and video game consoles was blurred with the November 15, 2001 release of Microsoft's Xbox gaming system. Hobbyists have expanded the uses of the Xbox by loading the Linux operating system, functionally transforming it into a low-end personal computer. With this modification the Xbox can function as a file server, a Web server, or a multimedia hub for television and stereo systems.

A "modded" Xbox can provide significant challenges in a computer crime investigation. It is difficult to determine visually if an Xbox has been modified to run additional operating systems or if it stores non game-related files, which may be of probative value in a criminal or civil case. Unfortunately, no established procedures exist for: (i) identifying whether or not an Xbox has been modified; (ii) creating a forensic duplicate of the storage media; and (iii) differentiating known-good files from other files that may reside in Xbox memory. This paper focuses on the forensic analysis of Xbox consoles. It describes the internal workings of the Xbox, and provides procedures for investigators to use in determining if an Xbox contains evidence and for extracting the evidence in a forensically-sound manner. The limitations of these procedures are also discussed.

2. Background

Since Microsoft launched the Xbox gaming console on November 15, 2001, cumulative sales of the system have reached 24 million units worldwide [7]. 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 equipment similar to that in a PC: hard drive, DVD drive, dedicated graphics hardware with TV-out, Ethernet and USB (albeit via a custom connector).

As with many popular electronic devices, computer hobbyists have modified the original Xbox for uses not intended by the developers. One of the most significant is to modify an Xbox to run an operating system other than the default operating system, enabling the Xbox to become functionally identical to a low-end personal computer.

Microsoft has implemented several security measures within the Xbox to thwart would-be hackers from running foreign software. These measures work together to create a chain of trust between each step of software execution [14]. From the point of view of forensic investigations, the two most important security elements are hard drive password protection and the file system used in the Xbox.

The Xbox employs an obscure option, the Security Feature Set [1], offered within the ATA specification. The Security Feature Set allows a 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 an attempt is made to access data on the disk. In the case of the Xbox, the drive is locked so that 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 [12]. 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 not boot with an unlocked drive and that a locked drive from one Xbox will not function on another.

The data on an unmodified Xbox hard drive consists of operating system files, game cache files and saved game files. The disk does not have a partition map *per se*; it is believed that the offsets are preprogrammed into the Xbox itself [6]. Microsoft designed a derivative of the FAT file system, called FATX, for the Xbox. FATX bears a strong resemblance to FAT32, containing only minor changes to the file system layout [13]. However, these changes are significant enough to prevent FAT32 forensic utilities from reading FATX.

Vaughan [16] was the first to examine Xbox security issues and forensic recovery. He discusses Xbox modification and provides several methods for bypassing the ATA password protection on the Xbox hard drive. He also describes the procedures necessary for building a FATX-enabled kernel and performing Linux-based forensics on a FATX image. Dementiev [5] largely mirrors Vaughan's coverage, but additionally describes methods for data extraction from within Windows.

This paper complements and extends the work of Vaughan and Dementiev. Specifically, it focuses on issues regarding the forensic validity of imaging methods, and attempts to provide a straightforward, forensically-sound method for data extraction. While other approaches modify the hard drive by permanently unlocking it or install hardware to access the contents of the drive, the approach described in this paper is much less intrusive, requiring no low-level physical interaction with the Xbox unit.

3. Forensic Procedures

A forensically-sound procedure for analyzing an Xbox must ensure that the digital evidence is not tainted or modified in any way. Such a procedure has 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 the determination of whether or not the Xbox has been modified. After the Xbox is confirmed as being modified, the next step is to build an analysis and acquisition workstation and image the Xbox in a forensically-sound manner. The final step involves both logical and physical analyses of the storage media.

Linux is an ideal operating system for analyzing the Xbox as a kernel-level file system driver is available from the Xbox Linux Project [11]. This means that images of the Xbox disk partitions can be mounted and interacted with using Linux in a native manner (i.e., without using external programs). Several alternatives exist for reading FATX volumes in Windows. However, our tests have demonstrated that they generally do not support logical analysis at a level necessary for forensics; for example, they do not support file times or provide a file listing interface.

These tools may also prevent forensic suites like EnCase and FTK from properly representing the media at a logical level as they do not present the volumes as Windows drives. Furthermore, they require the disk to be removed from the Xbox, unlocked and connected to the analysis machine. As noted above, our interest is in procedures that are non-intrusive and forensically sound.

Our procedures are based on the assumption that the forensic examiner has an understanding of the Linux operating system 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 enable it to process the Xbox FATX file system.

Tests were performed on a version 1.6 Xbox, which was "soft modded" via the Ndure exploit (described below). Linux was loaded on the primary partition of the Xbox as test data. Despite the limited number of tests performed, we believe that the proposed methods should function on most modified Xbox units as the modifications are not expected to impact our methods.

3.1 Initial Assessment

The first step is to determine whether or not the Xbox being investigated has been modified to run foreign code. There are two primary modification methods. The first, called "hard modding," involves physically modifying the Xbox to override the built-in protections against executing foreign code. The second method, "soft modding," breaks the software protection using code exploits. For the purposes of this paper, the two methods have the same functional effect – a modified Xbox will execute foreign programs, including Linux. Therefore, we do not differentiate between hard modded and soft modded Xboxes as far as forensic analysis procedures are concerned.

Hardware modification usually leaves obvious traces of tampering. This method of modification requires that the Xbox console be physically opened to install circuitry, often a replacement BIOS chip. The six screws that hold the console case together are located under the case and are concealed by a combination of stickers and rubber pads (Figure 1). Attempts to open the case generally result in the removal or destruction of the stickers and rubber pads.

An Xbox's connections and peripherals are good indicators that it has been modified. For example, if the Xbox is connected to a network and is running without a connection to a TV, it is likely that it is being used as a Linux-based server. USB-to-Xbox controller adapters with tradi-

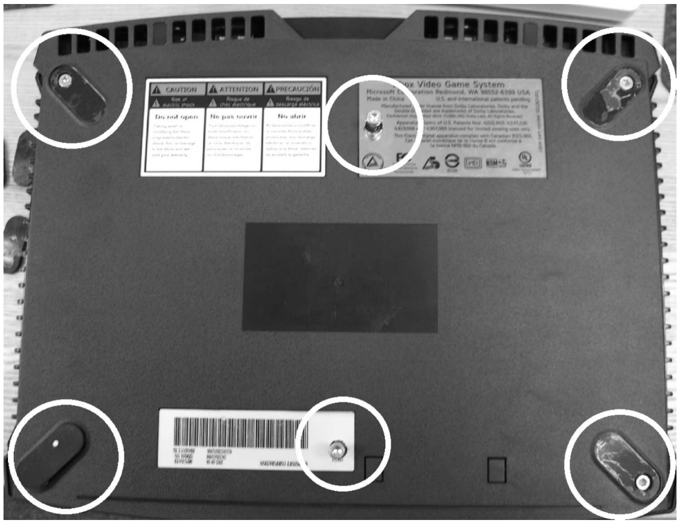

Figure 1. Xbox screw locations.

tional USB keyboards and mice are also suspect as these could allow the machine to be used like a traditional PC. The presence of a network cable 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 are more likely to attempt modifications.

A general method for determining if an Xbox has been modded is to attempt to boot a Linux-based boot CD from the unit. Our tests have shown that the hash of the hard drive does not change during this process, provided Linux is executed immediately. In fact, although portions of the Xbox software are loaded before Linux, e.g., the routine to temporarily unlock the hard drive, they do not appear to modify the disk contents in any way. Our tests also 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. Therefore, we recommend removing all memory units before booting the Xbox.

It should be noted that the internal clock of the Xbox is reset when the unit loses power even for a short amount of time; this is because the Xbox uses a capacitor instead of a battery to power its clock. The capacitor charge lasts for about one hour. Therefore, if an Xbox has to be removed from its environment during a seizure, it should be plugged into an outlet as soon as possible. An internal clock reset does not modify the hard drive, but the reset has the potential to interfere with certain soft modding exploits [17], including preventing Linux from being loaded. A dialog box appears at boot before any Xbox or Linux software is loaded. However, if the dialog box is dismissed immediately (by depressing the A button on the controller) the clock is not reset and the dialog box reappears at the next boot. This is the recommended course of action if the capacitor is accidentally drained.

In developing and testing our procedures we used a Xebian [19] boot CD specially designed for the Xbox. According to the Xbox Linux project, Xbox CD/DVD drives are frequently of inferior quality and have difficulty reading CD-Rs [18]. When burning a boot CD, it is recommended that the examiner burn the CD ISO to a DVD-R instead, as this forces the Xbox to rely on a different laser to read the disk. 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 disk was burned properly.

Any attempt to boot an unmodified Xbox with Xebian or any other non-Microsoft disk will display an error message:

> Your Xbox can't recognize this disk. Make sure it is an Xbox game, DVD movie or audio CD. Also check to see if the disk is dirty or damaged. Remove the disk to continue.

If the Xbox successfully boots into Linux it can be surmised that the unit has been modified in some manner and may contain Linux. The error message listed above is also displayed if the boot CD/DVD cannot be read; therefore, it is recommended to try multiple types of media (CD–RW, DVD+/–R, DVD+/–RW). Once the Xbox has booted into the Xebian environment, its hard drive and associated media are ready for acquisition and analysis.

3.2 Preparing an Analysis Machine

Preparing a workstation for FATX analysis is relatively simple. We recommend using a dedicated forensic computer. However, a machine with the latest hardware may not be appropriate as it must be compatible with the 2.4 Linux kernel. We chose to use Debian, but any other Linux distribution should work. The only requirements are adequate hard drive space to hold the acquired images along with the availability of basic compilation tools and an SSH client.

It is necessary to apply a patch from the Xbox Linux Project [11] to build a Linux kernel that can process the FATX file system. At the time of this writing, the Xbox Linux patch for kernel 2.6 was still in an experimental stage and not suitable for use in analysis. However, the patch for kernel 2.4 is still available and works well. After downloading the source for the latest 2.4 kernel from `kernel.org` one can apply the patch available from the Xbox Linux Project. When building the kernel, it is important to ensure that `CONFIG_FATX_FS` and `CONFIG_BLK_DEV_LOOP` are either built into the kernel or are modularized.

3.3 Creating a Forensic Duplicate

Having 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, the two systems should be connected using a crossover Ethernet cable. The default IP address of Xebian is 192.168.0.10/24; the IP address of the analysis workstation should be set to an address within this subnet. Next, SSH is used to connect to the Xbox as root using the aforementioned IP address. Note that the password for root is "xebian." At the prompt, information about the individual Xbox may be extracted by running the command `xbox_info -a`.

Table 1. Sample Xbox partition layout.

Device Name	Size	Size
/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

The Xbox partition structure is slightly different from that of a normal PC when viewed in Linux. Partitions are enumerated from minor node 50 up. Table 1 presents the partition layout of a factory-default Xbox.

A forensic duplicate of the individual partitions from the analysis machine may be created over SSH using the commands:

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

We recommend imaging individual partitions instead of the entire disk to simplify the task of mounting the partitions; this is because Linux does not have native support for mounting partitions embedded within an image. After the acquisition is complete, hashing can be performed using the `md5sum` utility on 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; many users partition the extra 2 GB for use with Linux. Multiple extra partitions may also show if a new, larger hard drive was installed (this requires a hardware modification of the Xbox).

3.4 Xbox Memory Units

Xbox memory units that provide portable storage for saved games can provide a challenge to investigators. These memory units are small (typically 8 MB) flash storage devices designed to plug into the Xbox controller. Our tests have revealed that the Xbox will format memory units at bootup that contain non saved-game data (e.g., personal documents or pictures). This formatting appears to be what is commonly referred to as a "quick format," where not all data is zeroed out. Regardless, the operation is destructive. Theoretically, this allows 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, the saved information is destroyed before it can be recovered. Formatting the memory unit appears to completely wipe out the file allocation tables and directory entries of the previous FATX file system; however, at least some of the data area is left intact. In a scenario where such formatting has occurred, a tool such as `foremost` [10] can be used to successfully extract the remaining file data. It appears that this formatting behavior is only present in Xboxes that have not been hardware modified to load a replacement Dashboard at bootup, but this has not been tested.

As noted by Dementiev [5], if a memory unit is placed in an attached controller before bootup, Xebian Linux will acknowledge the device and present it as a USB storage device. The devices 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 noted above, an Xbox is booted in this way renders the memory units subject to possible formatting. Furthermore, we have observed that attempts to hot plug Microsoft memory units after booting into Linux have always failed.

Despite our best efforts, we were unable to devise a forensically-sound procedure for imaging memory units. Due to the formatting issues mentioned above, we attempted to image memory units using a USB adapter included with Datel's Action Replay for the Xbox. But this approach revealed several problems. One problem 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`, which denotes that a SCSI-transparent command set should be used to interface with the device [15]. Instead, Microsoft memory units transmit a code of `0x66`. This confuses 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 the memory units also do not respect the established storage protocols. Forcing Linux to interact with a unit using the SCSI-transparent command set also failed; our subsequent discussions with a Linux kernel developer revealed that Microsoft memory units are not standard mass storage devices. Some research has already focused on the problem, as the Xbox-Linux 2.4 kernel patch includes certain basic modifications to the USB stack to enable these memory units to register. However, our attempts to use the patch on a device other than the Xbox resulted in the device not registering properly.

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.

3.5 Logical Analysis

Xbox partitions can be mounted for analysis after they have been acquired and transferred to the analysis machine. 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
```

At this point, traditional Linux forensics tools can be employed to examine the images for content [4]. For example, one can generate a timeline of file access events using Sleuth Kit and `macrobber` [3]. Both these tools create a linear timeline based on MAC times extracted from the file system. If individual files are identified as being of interest, they can be copied from the mounted file system.

It is also possible to use the Samba daemon [9] to share the contents of a mounted partition's directory over a network. 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 FTK 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 will not locate any content in unallocated space on a partition.

We have collected a set of file hashes that have been found to be consistent between Xbox units (version 1.6); the hash set is posted at [2].

The hash values should help eliminate some of the files from consideration in a logical analysis; these known-good files can be ignored when processing the set of files found on the partitions.

3.6 Physical Analysis

A logical analysis cannot detect evidence in the form of deleted files, stray data and information in slack space. Therefore, it is necessary to analyze each partition image physically as well. Linux utilities such as `xxd`, `strings`, `grep` and `foremost` may be used to identify case-related information.

Some information about the file system can be extracted using a hex editor (e.g., the Linux `xxd` hex viewer, which is usually included with the ubiquitous UNIX editor `vim` [8]). For example, searching for deleted file markings (`0xE5`) in the directory should make deleted entries apparent. Steil's analysis of the FATX file system [13] describes the binary structure of these entries.

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 may be in English. This 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'
```

Note that this example will not identify words or phrases that span two lines.

Finally, `foremost` may be employed to extract specific file types from the image based on their file signatures. The utility relies on scanning an image for file information unique to that type of file and then extracting the data following that information. Since this is independent of any file system, the utility will recover any deleted files that are present. While it is not perfect, `foremost` can recover data that would otherwise only be retrievable using a hex editor.

4. Conclusions

Since an Xbox contains hardware and software approaching that of a personal computer, it should be considered to be an evidentiary item at a crime scene. Nevertheless, Xbox forensic analysis is still in its infancy. As of early 2007, no major forensic tool suite supports the

Xbox's FATX file system and partition layout, and no standard methods exist for extracting electronic evidence from an Xbox hard drive.

The Xbox forensic analysis procedures presented in this paper are by no means exhaustive. Due to the myriad Xbox models and the abundance of modification methods, there is a strong possibility that the procedures may not work as intended. Still, they should be useful to examiners. We also hope that they will stimulate the digital forensics research community to develop more sophisticated methods for Xbox forensics.

Acknowledgements

We wish to thank Ron Jewell of the Forensic Science Center at Marshall University, Huntington, West Virginia for donating the Xbox consoles used in our research. We also wish to thank Jonathan Medefind for his assistance in modifying Xbox consoles.

References

[1] H. Bögeholz, At your disservice: How ATA security functions jeopardize your data (www.heise.de/ct/english/05/08/172/), 2005.

[2] P. Burke and P. Craiger, Xbox media MD5 hash list, National Center for Forensic Science, Orlando, Florida (www.ncfs.org/burke .craiger-xbox-media-hashlist.md5), 2006.

[3] B. Carrier, The Sleuth Kit (www.sleuthkit.org).

[4] P. Craiger, Recovering evidence from Linux systems, in *Advances in Digital Forensics*, M. Pollitt and S. Shenoi (Eds.), Springer, New York, pp. 233–244, 2005.

[5] D. Dementiev, Defeating Xbox (utilizing DOS and Windows tools), unpublished manuscript (personal communication), 2006.

[6] A. de Quincey and L. Murray-Pitts, Xbox partitioning and file system details (www.xbox-linux.org/wiki/Xbox_Partitioning_and_File system_Details), 2006.

[7] Microsoft Corporation, Gamers catch their breath as Xbox 360 and Xbox Live reinvent next-generation gaming (www.xbox.com/zh-SG /community/news/2006/20060510.htm), May 10, 2006.

[8] B. Moolenaar, Vim (www.vim.org).

[9] Samba.org, The Samba Project (www.samba.org).

[10] SourceForge.net, Foremost version 1.4 (foremost.sourceforge.net).

[11] SourceForge.net, The Xbox Linux Project (sourceforge.net/projects /xbox-linux).

[12] SpeedBump, Xbox hard drive locking mechanism (www.xbox-linux.org/wiki/Xbox_Hard_Drive_Locking_Mechanism), 2002.

[13] M. Steil, Differences between Xbox FATX and MS-DOS FAT (www.xbox-linux.org/wiki/Differences_between_Xbox_FATX_and_MS-DOS_FAT), 2003.

[14] M. Steil, 17 mistakes Microsoft made in the Xbox security system (www.xbox-linux.org/wiki/17_Mistakes_Microsoft_Made_in_the_Xbox_Security_System), 2005.

[15] USB Implementers Forum, Universal Serial Bus Mass Storage Class Specification Overview (Revision 1.2) (www.usb.org/developers/devclass_docs/usb_msc_overview_1.2.pdf), 2003.

[16] C. Vaughan, Xbox security issues and forensic recovery methodology (utilizing Linux), *Digital Investigation*, vol. 1(3), pp. 165–172, 2004.

[17] Xbox Linux Project, Clock loop problem HOWTO (www.xbox-linux.org/wiki/Clock_Loop_Problem_HOWTO), 2006.

[18] Xbox Linux Project, Xbox Linux boot CD/DVD burning HOWTO (www.xbox-linux.org/wiki/Xbox_Linux_Boot_CD/DVD_Burning_HOWTO), 2006.

[19] Xbox Linux Project, Xebian (www.xbox-linux.org/wiki/Xebian), 2006.