Introduction

Digital forensics is the ability to retrieve and analyze information from digital media (whether it is disk, computer memory, cell phones, or various handheld devices) to be able to introduce the results as evidence in a court of law. As storage area networks (SANs), whether iSCSI or Fibre Channel (FC), increasingly are used to store an organization’s information, the ability to quickly and efficiently capture data from storage arrays for forensic purposes is increasingly necessary. But there are significant differences between capturing data from a direct access storage device (DASD) that is part of the physical enclosure of a desktop or laptop computer, and data that is stored on a network accessible device that is physically and logically separate from the end-user computing platform.

I will present the similarities and differences encountered when collecting data from a “disk structure” that is implemented as part of a storage array. Although similar techniques can be used as for local disks, the implementation of logical volumes in the SAN, as well as the size of the underlying storage abstraction means that certain techniques are not as effective as they would be when applied to DASD devices, specifically disks stored with the computing device itself. Although these differences can be a significant impediment to a forensics examiner, storage administrators have access to different tools and techniques that can be applied to retrieve data for forensics examination. Some of these tools enable activities that are now difficult, such as creating point-in-time copies of information that represent the state of a computer’s memory and disk at a particular instance in time. This article summarizes tools and techniques as they apply both to traditional storage environments and to virtualized environments.

First, the usual disclaimer: I am not a lawyer (IANAL). Nothing in this article should be construed as legal advice. Nor is it meant as instructions on how to break the law or an exhortation to do so. When in doubt, consult your organization’s policies on incident response, and involve your legal team and management (and Human Resources if an employee is potentially involved.) If you don’t have an incident response plan: get one. Having to make these decisions while an incident is in progress wastes time and increases the likelihood that mistakes will be made that may jeopardize potential actions in the future.
That CSI Moment...

Most of us have heard of “that CNN moment,” the moment in time when someone at the CxO level in the company gets a telephone call from a local reporter asking for comment on a breaking news story on CNN indicating that the company has just suffered an enormous security breach. Fewer have heard of what Richard Austin calls “that CSI moment” when someone from law enforcement shows up at your data center with a warrant to search your 140Tbyte storage subsystem for files from a particular user [AUST07]. Austin makes the observation that with the increased use of SAN storage, it is increasingly likely that a storage administrator will be among the “1st on the scene” when it becomes time to gather digital evidence.

What exactly do we mean by digital evidence? Let’s start with the definition of a digital investigation: a process whereby we develop and test hypotheses about digital events. Digital evidence is thus a “digital object that contains reliable information that supports or refutes a hypothesis.” [CARR05, p. 4] Evidence can be used in either an investigative or a legal context.

The nature of digital evidence requires us to use technology during an investigation, so the main difference between a digital investigation and digital forensic investigation is the introduction of legal requirements A digital forensic investigation is a process that uses science and technology to analyze digital objects and that develops and tests theories, which can be entered into a court of law, to answer questions about events that occurred. In other words, a digital forensic investigation is a more restricted form of digital investigation. [My emphasis] [CARR05, p. 4]

There are two general categories of digital investigations: live and dead. Live implies that the incident in question is ongoing. Dead implies that the incident has ended (or has been ended by shutting down an affected system). One issue is volatility: investigation on a live system can make changes on that system, whereas creating a forensic copy (a bit-for-bit duplicate of a digital object) and performing analysis on that copy guarantees that the original evidence remains intact.
Two things have happened that make efficiency in detecting an incident more important. First is the existence of breach notification laws that require notification after a period of time if a security breach has occurred. Second, since much of the information now stolen has implications for financial loss (identity theft, emptying bank accounts, etc), rapidly notifying affected parties may reduce the financial impact of that breach.

<table>
<thead>
<tr>
<th>“Dead” Forensics</th>
<th>“Live” Forensics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security Breach by Attacker or Malware</td>
<td>Security Breach by Attacker or Malware</td>
</tr>
<tr>
<td>Identification of Security Incident</td>
<td>Identification of Security Incident</td>
</tr>
<tr>
<td>Notification to Users/Management</td>
<td>Hard Drive Analysis begins</td>
</tr>
<tr>
<td>Hard Drive Acquisition</td>
<td>Notification to Users/Management</td>
</tr>
<tr>
<td>Hard Drive Analysis begins</td>
<td>Hard Drive Acquisition (if necessary)</td>
</tr>
</tbody>
</table>

Table: “Dead” vs. “Live” Forensics [SAWY07]

Most of the writings on digital investigation presume that we are working in a dead environment. The compromised system is acquired (in some case, physically seized), the disk(s) are mounted externally, potentially using hardware-based write blockers, and an exact duplicate of the disk is created. This disk is then used for analysis.

We’ve made several assumptions in the above scenario. First, it makes sense to physically acquire the affected system as it will provide access to all storage components (memory, persistent storage, etc.) We are working at the file system and the raw device (block) level. We can access the data in a reasonable time: hours, days at the outside. We can make forensically sound copies (to the bit level) of the particular device (we’ll refer to this as cloning a device, rather than copying a device, since cloning implies block-level access while copying implies file level access). The physical devices that we wish to examine are contained in a single enclosure or in several enclosures but are in reasonable physical proximity to each other. Each disk can be taken as a whole (that disk is mountable on another device and the device will be able to read data from the device in an understandable fashion.) Although this doesn't strictly apply if we are using software RAID using internal disks, we can assume that if we have the right enclosure we could mount these disks in that rack and, using the same software that was used to create the RAID, read and organize that data stored on those disks.
In SCSI terms, our unit of analysis is the Logical Unit (LU), identified by its LUN. At one level of abstraction, the LUN corresponds to a disk spindle (OK, not true if we’re using striping). The LUN can be broken down into other structures such as partitions. In the Linux world, I’m used to seeing entities like sda and sdb, corresponding to two SCSI disk drives, with partitions named sda1, sdb2, etc. Each partition can host a file system (NTFS, FAT32, ext3, etc.) Usually, a partition will be created that covers an entire disk. Why? So an entire disk can be copied from one system to another using a single command, as in the following example.

```
dd if=/dev/sda5 of=/dev/sdb5 bs=1024
```

Example: using the `dd` command to copy a partition from one device to another.

There is one final note. If the attacker has sufficient skills, it may be possible to hide data on the local disk by accessing those parts of the file system or the underlying disk format that are not normally made available to users at the file system level. These places can include the Host Protected Area (HPA) of the disk drive itself, slack space available in an NTFS file system, or in “bad blocks” areas.

**Digital Investigations in the SAN**

The disks that are made available via a SAN (whether an IP or a FC SAN) are very much like attached disks, but there these differences can be key to a digital investigation.

First, the storage presented to the individual host is frequently in the gigabyte or terabyte range, and this presents problems of obtaining suitably sized storage and the time it takes to make (and search) forensic copies.

LUNs made available via the SAN potentially are accessible via multiple systems. Our miscreant doesn’t need to access one particular system anymore; many systems may access the same LUN on the same storage array. In the SAN, I have two targets for
forensic investigation: the machine that accessed the SAN storage, and the storage array itself.

When RAID is in use, a single physical disk doesn't much matter, especially if you're using striping, or if you're using a technology that spreads individual blocks of data (files, for want of a better word) across EFD (Enterprise Flash Drives), FC (Fibre Channel), and SATA drives. Consider EMC FAST (Fully Automated Fast Tiering) that utilizes a "LUN-less" or "below-LUN" storage solution for adjusting to differences in application workload by moving files (in this case, blocks) between different kinds of storage based on policy. Given that the disks in use will vary based on application load, it would be extremely difficult to determine which physical disks were in use at any given moment in time. This is generally true of any storage system that takes advantage of usage patterns to move lightly used data to different physical structures. This time-varying use of physical storage is another example of how SANs differ from a fixed local disk.

So how are SAN disks similar to local disks?

We’ve already mentioned that our unit of analysis is the LUN, and demonstrated one common way to create a clone of a particular LUN using standard Linux tools. This same technique will work regardless of whether that raw disk structure is created on a local disk or whether it has been created on storage in the SAN. At the LUN level, we can treat data in the SAN exactly as we treat data from a directly attached local disk, regardless of how long the cable is or how many intermediate steps there are along the way. This is true even if encryption is present, either at the host HBA, within the SAN, or within the storage array. If the data is in cleartext when it’s made available via file system access, then I can read the contents of the LUN as a file system and the contents of the files as well.

Since a LUN appears as a mounted disk, we can assume that our wrongdoer can utilize any and all tricks of the trade in creating files on that remote file system. Hidden files

\[\text{\textsuperscript{1}}\]

\[\text{\textsuperscript{1}}\] Note that files that appear hidden under one OS might not be hidden on another. Whenever I access a Windows share on my home NAS, I see the .dot files created by my Mac OS/X system. Not so on my Mac OS/X machine.
alternative data streams on NTFS file system, hiding data in plain sight (c:\winnt\system32\io.sys.dll [notice the 0?]). Consider such tools as TrueCrypt [TRUE10]. TrueCrypt allows you to specify a file and then to mount and process it as an encrypted file system. Administrative privileges are not required. Not only that, but TrueCrypt allows a second level of encrypted file system within the first. This allows a user to surrender the password for the 1st file system without revealing the contents of the 2nd file system.

Let's assume that the contents of the file system would be of interest to an investigator, and let's label this file evidence.tc. Remember: this file represents anything that is of investigative interest. It could be something illegal such as child pornography, something that is against company policies, or any unauthorized activity on a particular computing system.

```
%> truecrypt -t -c       # text mode, prompt for info
%> truecrypt evidence.tc /mnt/tc # will prompt for password
%> cp badstuff /mnt/tc
%> shred -u badstuff # secure removal of badstuff
```

Although this seems almost foolproof, the TrueCrypt user guide, at over 100 pages in length, goes into detail to explain when even these efforts won't preserve data from prying eyes (for good or bad).

When it comes time to delete a file, SAN environments don't work quite the same as a local disk. Consider the following disclaimer that is displayed as part of the help option for the Linux shred command.

CAUTION: Note that shred relies on a very important assumption: that the file system overwrites data in place. This is the traditional way to do things, but many modern file system designs do not satisfy this assumption. The following are examples of file systems on which shred is not effective, or is not guaranteed to be effective in all file system modes:

- log-structured or journaled file systems, such as those supplied with AIX and Solaris (and JFS, ReiserFS, XFS, Ext3, etc.)
• file systems that write redundant data and carry on even if some writes fail, such as RAID-based file systems
• file systems that make snapshots, such as [some vendor’s NAS servers]
• file systems that cache in temporary locations, such as NFS version 3 clients
• compressed file systems

In the case of ext3 file systems, the above disclaimer applies (and shred is thus of limited effectiveness) only in data=journal mode, which journals file data in addition to just metadata. In both the data=ordered (default) and data=writeback modes, shred works as usual. Ext3 journaling modes can be changed by adding the data=something option to the mount options for a particular file system in the /etc/fstab file, as documented in the mount man page (man mount).

In addition, file system backups and remote mirrors may contain copies of the file that cannot be removed, and that will allow a shredded file to be recovered later.

Listing from shred Linux manual page.

I think that we can agree that data stored in SANs presents forensic investigators with a different set of problems. The tools available to manipulate data stored in the SAN are different as well. A storage administrator has software tools to perform data replication of existing LUNs. LUNs can be cloned by mirroring (creating complete snapshots of a LUN), or by replication piecemeal, based on copy-on-write techniques (sparse clones). Copies of a LUN can be made available to separate systems almost instantly to separate systems that can then begin parallel processing on identical copies of the data.

You can use this technique, originally used to provide business continuity and disaster recovery, when you believe that a security breach is in process. Creating replicas of the file systems provides an organization with the ability to begin disk analysis sooner without taking a particular system out of production.
Network Attached Storage (NAS)

Network-attached storage (NAS) provides access to file-system-like storage systems that provide access either to DASD devices or to storage systems. Some NAS devices may be constructed as JBOD (just a bunch of disks), others may access directly attached storage arrays or storage arrays accessible via a SAN. Common access protocols in use are Sun's Network File System (NFS), Microsoft's Common Internet File Systems (CIFS), or File Transfer Protocol (FTP).

Network file systems provide a different kind of challenge for the forensic investigator. First, there is no notion of a disk partition or a raw disk presented to the host system. Rather, there is a directory hierarchy. Access to entities within this hierarchy is at the file level, not the block level. And as we've seen, access at the file level will not produce a block-for-block copy of the actual disk structure. On the other hand, we can expect that the attacker won't have access at the block levels of the NAS device either, so certain kinds of data hiding won't happen.

One question to ask concerning access to NAS file-system is what directories (folders) can I access on a particular share? Consider that the attacker might be able to create a personal level directory or a project-level directory at the top level of a file system hierarchy: say, S:\clbrooks, or S:\project_name (where S: is the drive letter of my shared file system.) This, combined with other anti-forensics techniques (changing file extensions, changing file signature on the disk) can make it more difficult for the investigator to perform a disk analysis.

After all, what's the goal for the attacker? Aside from the initial activity, a secondary goal is to force the investigator to take more time to search for evidence of malfeasance. The more places the attacker can create files, and the larger those places, the more time will be required to search and analyze the contents of that space. Time increases the expense of the investigation as well as the risk that evidence will be lost, spoiled, or that interest in pursuing the investigation will wane (perhaps because a new incident has occurred). Of course, if the attacker wishes to cover their tracks more fully, there are also more places where they need to delete the history of their actions. And getting to
those places may be correspondingly more difficult in a NAS environment than in a local or SAN environment.

**Digital Forensics in Virtualized Environments**

Virtualized computing environments are increasingly common, and create yet a different kind of problem for the investigator.

Consider how virtual machine images are constructed. VMware, for example, keeps all information about a particular VM in a set of associated files. VMware servers (running the ESX or ESXi hypervisors) can store files associated with a particular guest VM using NFS, Raw Disk Maps (RDM), or VMware proprietary file systems called VMFS [HALE09]. While 3rd party drivers for reading these file systems exist, accessing these file systems isn’t as well integrated into forensic tools as other file system formats.

If the VMware server is using SAN storage or NAS to store the virtual machine images, the same tools that are used to replicate volumes in the SAN can replicate data from a particular guest VM for investigation. There are, however, particular steps that need to be taken to preserve the most information for an investigator. For example, VMware HA should be shut off [HALE09, p. 415], since leaving it running could cause the guest VM to reboot. Likewise, the VM should be shut down from within the VM kernel, since this is the only way to prevent shutdown code from running. This kind of detail (and we have barely scratched the surface) indicates that someone who is intimately familiar with virtualized installations should be part of the incident response team.

Certain tasks are made easier in virtual environments. One issue for the forensic investigator is how to gain access to the memory image of a particular system. Memory for VMs is stored in files in the file system. If a VM is quiesced, the contents of memory are preserved, along with the state of the file system (this is achieved by taking a snapshot). You can make this snapshot available to the investigator.
Digital Forensics in the Cloud

Forensics in the Cloud represents an even more difficult job for the investigator. Let’s consider two different scenarios. In the first instance, I'm trying to track down the actions of an individual from a physical computer that I've obtained; in the second instance, I'm working with a Cloud Service Provider (CSP) to determine what that person might have done using a compute service.

The initial activity requires identifying exactly what the subject has had access to during the period of interest. Let's use an example of Amazon Web Services. There are multiple services available: the ones in which we're most interested are the Simple Storage Service (S3) and the Elastic Compute Cloud (EC2).

S3 allows a user to store a particular file (henceforth, blob, recognizing that the contents are not interpreted by the S3 service) in a specified bucket that is stored somewhere within a particular geographic location (say “us1-east-usa – Northern Virginia.”). Amazon doesn't provide more detail than that in terms of physical location. Any manipulation of the data within that blob (compression, encryption) is the responsibility of the user and not the provider. Once the data is stored in the S3 cloud, the original blob can be deleted from the local system, along with any file-system meta-data associated with the original.

```bash
%> s3cmd put /tmp/evidence.tc s3://s3.clbrooks.com/20100222-archive.tgz
%> shred -z -n 7 -u -random-source=/dev/urandom /tmp/evidence.tc
# overwrite 7 passes of random data, one final pass of zeroes, and truncate and delete the file
```

Example N: Storing a file in the cloud and then securely deleting it from the local disk

Let's consider our TrueCrypt volume again, evidence.tc. Once we've deleted that file, and securely wiped the directory information, we haven't left much for an investigator to go on. This is especially true since there is no necessary correspondence between the identifier chosen on the local machine and the identifier in the Cloud. While there are other ways that an experienced investigator might determine file system activity
(lastcomm and history on Linux come to mind), searching for securely deleted files in a file system isn’t one of them (one of the problems of dead (post-hoc) analysis).

There are two others aspects that arise when storage is retrieved from the Cloud: the encryption problem and the parsing problem. Both of these problems are typified by our TrueCrypt example. TrueCrypt on Linux gives me four choices for file system format: none, FAT, ext2 and ext3 (the latter two are Linux file systems). As an investigator, I have four choices about how I might need to mount this particular file on a file-system mount point – and I know how to parse all of the above options. If the file is encrypted, I have a different problem – recovering the encryption key. Truecrypt allows passwords up to 64 characters in length – chances are I'm not going to brute force these passwords. Unless I'm clever and lucky (the user has chosen a bad password), there is only a small chance that I will be able to decrypt the file and apply my parsing skills. And without decrypting the contents, I can’t determine what data is contained on that blob.

Location poses another problem for the investigator. Amazon provides availability zones (data centers) in the United States and the EU. At least one reason for this is the restriction on the use of private information within the EU. Private, individual information cannot be stored outside the geographic boundaries of the EU. Is the data that an attacker has stored in the Cloud, under the auspices of an account that she owns, considered to be “in her possession”? Does she have “plausible deniability” based on the steps that she’s taken to secure access to that data? (The fewer steps she’s taken to secure access to the data moves ever closer to the SODDI defense ["Some other dude did it"]). Is the blob that has been stored in an S3 bucket really what was originally stored? As an investigator, how do I prove that it is?

Now let’s consider EC2. In this offering, Amazon enables a customer to create an instance of an Amazon Machine Image (AMI). An AMI is a pre-configured server image (possibly supplied by a customer) containing particular software packages intended to offer an Internet-accessible service. Creating an instance of that image is thus akin to provisioning a physical server with a “gold master.” Note that S3 is used internally by Amazon to store both AMIs and the components of an EC2 instance.
EC2 supports three kinds of storage. They are:

- instance storage. This is provided as part of the AMI, and is presented as the usual mounted partitions (/home, /usr, etc. on a Linux system)
- Elastic Block Storage (EBS) This storage is similar to SAN storage. The following example demonstrates how EBS is created and attached to a running instance.

```
%> e2-create-volume –size 10 –z us-east-1a
# response is: vol-9a5b43hf
%> ec2-attach-volume vol-9a5b43hf –i i-fb4732dd –d /dev/sdh
%> mkdir /mnt/datastore
%> mkfs –t ext3 /dev/sdh
%> mount –t ext3 /dev/sdh /mnt/database
# … use the filesystem
%> unmount /mnt/datastore
%> ec2-detach-volume vol-9a5b43hf –I i-fb4732dd
# volume is available for mounting via another instance
```

Example: creating an EBS volume and attaching to a local mount point

First, create a volume of 10 Gigabytes in the us-east-1a zone. The response to this command identifies the volume identifier. The volume (vol-9a5b43hf) is then attached to a specific EC2 instance (in this case, i-fb4732dd) as /dev/sdh. A mount point is created (/mnt/datastore) and a filesystem of type ext3 is created on the raw device, upon which the device is mounted on the allocated mount point (/dev/datastore). At that time, the volume can be accessed as an ext3 filesystem.

Once the file system is no longer needed, it can be unmounted and the volume released for use by another instance.
• S3. Instances can access objects stored via S3 (assuming possession of appropriate credentials.)

```bash
$> s3cmd get s3://s3.clbrooks.com/20100222-archive.tgz /tmp/evidence.tc
$> truecrypt -mount /tmp/evidence.tc /mnt/documents
#prompt for passwords
```

Example: extracting S3 data to a particular instance

These three types of storage offer different capabilities in terms of complexity, speed, and cost. From a security perspective, they offer different guarantees concerning reliability, durability, and data remanance. Instance-based storage is claimed to be zeroed out when an instance is terminated. Both S3 storage and EBS have a lifespan longer than a specific instance. An instance could terminate in one availability zone, another could be instantiated in a second availability zone, and the EBS could be mounted on this second instance.

Best practices for storage within the Cloud, such as EC2, counsel that all data should be encrypted, and that encryption keys should not be stored with the data. If instance level data is zeroed upon instance termination, and EBS and S3 storage is encrypted, an investigator now must retrieve a set of encryption keys to gain access to the data for analysis – a complicating step that may delay investigations, or expose the fact of an ongoing investigation.

On the other hand, there are potential benefits for performing incident response/investigations in the Cloud. Craig Balding, in his blog post entitled *Seven Technical Security Benefits of Cloud Computing* [BALD07], lists the following

• **Forensic readiness.** Pre-build forensic machine images, ready to be instantiated at a moment’s notice. Multiple instances could be created to support multiple investigators.

• **Decrease evidence acquisition time.** Clone a compromised server and make the compromised system available for analysis – memory, disks, and all
• **Eliminate or reduce service downtime.** No need to take a compromised system out of service. Re-instantiate the server and move on.

• **Decrease evidence transfer time.** Bit-for-bit copies are very fast, and may reside in the same activation area.

• **Eliminate forensic image verification time.** Some Cloud storage may provide an MD5 of SHA hash code as part of creation of the storage object.

• **Decrease time to access protected documents.** Utilize increased number of forensic servers to perform tasks in parallel.

Seen from this perspective, the Cloud provides opportunities for more resources to be brought to bear on an investigation which may increase our ability to diagnose and prevent further exposure.

**Conclusion**

SANs, whether IP, FC or NAS, complicate the work for the forensic investigator. There are more places to store data, a larger search space, physical systems not necessarily co-located with particular hosts, different rules for data manipulation (unused data migrated to offline storage, deleted from primary), time-varying location of data on physical driver, all of which imply more time needed to complete an investigation. RAID demonstrates that there is no longer a single physical object that can be duplicated and the original held as evidence. Instead, capturing a single device might require that multiple disks be removed from a storage array, and in certain storage environments, these disks may not be dedicated to a single system (consider “LUN-less storage”).

That said, storage subsystems provide other tools that can aid the investigator. Replication software can be used to clone a particular storage volume for analysis without removing that volume from production. Snapshot technologies can be used to dynamically track changes to a storage volume as they occur. RAID devices typically provide a large file cache that could yield information about file usage. One might suggest creating a clone and then removing the original disks for forensic purposes as an alternative (consider it a case of all disks failing at once for a particular LUN).
Although accessing those original disks may require specific software as well as particular positioning of those disks within a given storage array, having the physical disks available would help support a true forensics examination that requires physical possession and “chain of custody” for the physical media.

Cloud-based storage adds another layer of complication. Object-based (symbolic access) to data storage creates a more extreme disconnection between what an object is named and what might be stored in that object. Encryption is advocated as a best practice for storing data in the cloud. As a result, an investigator can count on having to obtain encryption keys to further an investigation.

On the other hand, virtualization, an enabling technology for the Cloud, does permit access to all elements of a running VM, including memory. Snapshot capabilities within the hypervisor can be used to create a moment-in-time image of a running machine including both disk and memory images. Unless the OS or application in question encrypts data in memory, decrypted information can be found in these files, including passwords as well as decrypted contents of file blocks.

In summary, then, SANs, in concert with virtualization technologies and cloud computing, represent a new challenge for forensic investigations. Two improvements will ensure that investigators can keep pace with the wrongdoers: better integration and manipulation of SAN and virtualization data structures with existing forensics and investigative tools, and increasing co-operation between service providers and the security investigators in order to make forensic duplicates available for analysis early on in a security breach investigation.
Appendix A - Sources


CARR05  Carrier, Brian. *File System Forensic Analysis.* Pearson Education, 2005


TRUE10  *Truecrypt* help command output, version 6.3a for 32-bit Debian systems.
