- Always ensure that the media you are using to acquire live response data is pristine and does not contain unrelated case data, malicious code specimens, or other artifacts from previous investigations. Acquiring digital evidence on "dirty" or compromised media can taint and undermine the forensic soundness of the acquired data.

VOLATILE DATA COLLECTION METHODOLOGY

▶ Prior to running utilities on a live system, assess them on a test computer to document their potential impact on an evidentiary system.
▶ Data should be collected from a live system in the order of volatility, as discussed in the introduction. The following guidelines are provided to give a clearer sense of the types of volatile data that can be preserved to better understand the malware.

Documenting Collection Steps

▶ The majority of Linux and UNIX systems have a `script` utility that can record commands that are run and the output of each command, providing supporting documentation that is cornerstone of digital forensics.

- Once invoked, `script` logs the time and date, as shown in Fig. 1.2.

```
Script started on Tue 08 Mar 2011 02:01:19 AM EST
```

FIGURE 1.2—`Script` command time and date logging.

- `Script` caches data in memory and only writes the full recorded information when it is terminated by typing "exit." By default, the output of the `script` command is saved in the current working directory, but an alternate output path can be specified on the command line.

Volatile Data Collection Steps

- On the compromised machine, run a trusted command shell from a toolkit with statically compiled binaries (e.g., on older nonproprietary versions of the Helix CD or other distributions).
- Run `script` to start a log of your keystrokes.
- Document the date and time of the computer and compare them with a reliable time source.
- Acquire contents of physical memory.
- Gather host name, IP address, and operating system details.
- Gather system status and environment details.
- Identify users logged onto the system.

- Inspect network connections and open ports and associated activity.
- Examine running processes.
- Correlate open ports to associated processes and programs.
- Determine what files and sockets are being accessed.
- Examine loaded modules and drivers.
- Examine connected host names.
- Examine command-line history.
- Identify mounted shares.
- Check for unauthorized accounts, groups, shares, and other system resources and configurations.
- Determine scheduled tasks.
- Collect clipboard contents.
- Determine audit configuration.
- Terminate `script` to finish logging of your keystrokes by typing `exit`.

Analysis Tip

File Listing

In some cases it may be beneficial to gather a file listing of each partition during the live response using The Sleuthkit (e.g., `/media/cdrom/Linux-IR/fls /dev/hda1 -lr -m / > body.txt`). For instance, comparing such a file listing with a forensic duplicate of the same system can reveal that a rootkit is hiding specific directories or files. Furthermore, if a forensic duplicate cannot be acquired, such a file listing can help ascertain when certain files were created, modified, or accessed.

Preservation of Volatile Data

☑ ***First acquire physical memory from the subject system, then preserve information using live response tools.***

▶ Because Linux is open source, more is known about the data structures within memory. The transparency of Linux data structures extends beyond the location of data in memory to the data structures that are used to describe processes and network connections, among other live response items of interest.

- Linux memory structures are written in C and viewable in `include` files for each version of the operating system. However, each version of Linux has slightly different data structures, making it difficult to develop a widely applicable tool. For a detailed discussion of memory forensics, refer to Chapter 2 of the *Malware Forensics Field Guide for Linux Systems.*

- After capturing the full contents of memory, use an Incident Response tool suite to preserve information from the live system, such as lists of running processes, open files, and network connection, among other volatile data.
- Some information in memory can be displayed by using Command Line Interface (CLI) utilities on the system under examination. This same information may not be readily accessible or easily displayed from the memory dump after it is loaded on a forensic workstation for examination.

Investigative Considerations

- It may be necessary in some cases to capture some nonvolatile data from the live subject system and perhaps even create a forensic duplicate of the entire disk. For all preserved data, remember that the Message Digest 5 (MD5) and other attributes of the output from a live examination must be documented independently by the digital investigator.
- To avoid missteps and omissions, collection of volatile data should be automated. Some commonly used Incident Response tool suites are discussed in the Tool Box section at the end of this book. 🛠

Physical Memory Acquisition on a Live Linux System

☑ ***Before gathering volatile system data using the various tools in a live response toolkit, first acquire a full memory dump from the subject system.***

- Running Incident Response tools on the subject system will alter the contents of memory.
- To get the most digital evidence out of physical memory, perform a full memory capture prior to running any other incident response processes.
- There are a myriad of tools of and methods that can be used to acquire physical memory and many have similar functionality. Often, choosing a tool and method comes down to familiarity and preference. Given that every malware incident is unique, the right method for the job may be driven not just by the incident type but by the victim system typology. Various approaches to acquiring physical memory are provided here, and the examination of the captured data is covered in Chapter 2 of the *Malware Forensics Field Guide for Linux Systems.*