Ubuntu Artifacts Generated by the Gnome Desktop Environment

Brian Nishida

Ubuntu Artifacts Generated by the Gnome Desktop Environment

GIAC (GCFA) Gold Certification

Author: Brian Nishida, brian.nishida@student.sans.edu
Advisor: Johannes Ullrich

Accepted: November 19, 2020

Abstract

This research identifies Gnome Desktop Environment (GDE) artifacts and demonstrates their utility in Linux forensic examinations. The classic Linux forensic examination is tailored to computer intrusions of victim servers because the enterprise's critical Linux systems are typically web servers, mail servers, and database servers. However, the emphasis on intrusions and servers has two shortcomings. First, in addition to network intrusions, digital forensic labs examine specimens from various investigations: e.g., child exploitation, homicide, and financial crimes, to name a few. Second, the majority of Linux users run GUI-based desktop versions rather than command-line server versions. In these cases, the GDE may be used to install applications, run applications, open files, join Wi-Fi networks, and upload files. These point-and-click actions have been overlooked in the classic Linux examination; therefore, they will be explored in this research. Lastly, the importance of these GDE artifacts will be demonstrated in three practical scenarios.

1. The Classic Linux Forensic Examination

In cybersecurity, it is best practice to patch and update systems continuously, and so it appears that the "system" of Linux Forensics is due for an upgrade. For more than ten years, the classic Linux forensic examination has remained unchanged. Its goals are to find evidence of attacker commands, malware, lateral movement, and persistence. To achieve these goals, the forensic community recommends examining: processes, services, network connections, user accounts, SUID/SGID files, scheduled jobs, shell histories, logs, and more (Pogue, Altheide, & Haverkos, 2008; Altheide & Carvey, 2011; and Le, 2019). Fung (2013) provides a detailed example of this classic Linux examination.

The classic Linux examination is tailored to investigations of computer intrusions on Linux servers. However, the classic examination may overlook important artifacts if the specimen is a Linux *desktop* system or if the analysis is for a *non-intrusion* investigation. These are quite common occurrences.

First, many digital forensic examinations in computer crime labs are for non-intrusion violations. In the FBI's Regional Computer Forensics Laboratories, the most common digital forensic examinations involve child exploitation, homicide, fraud, assault, and narcotics (FBI, 2018). Even commercial forensic shops will encounter non-intrusion policy violations such as inappropriate use or insider data theft. In these non-intrusion forensic examinations, there will be no need to search for rogue accounts, scheduled tasks, timestomping, or malware.

Also, many users run the Linux desktop version with a Graphical User Interface (GUI) rather than the command-line server version. For the popular Ubuntu distribution, 85% of users run the desktop version (Davies, 2020). This means that most Linux users perform a combination of command-line and point-and-click actions. Artifacts arising from the use of the GUI have been, for the most part, neglected in classic Linux examinations. However, some GUI artifacts have been identified.

Brian Nishida brian.nishida@student.sans.edu

The most commonly documented GUI artifact is the `recently-used.xbel` file, which has been noted by Pomeranz (2012), Patil & Meshram (2015, 2016), and Nikkel (2018) and will be discussed later in this paper. Moreover, in 2018, Nikkel provided a comprehensive list of modern Linux forensic artifacts, many of which arise from GUI usage, although not explicitly stated.

This research aims to identify the artifacts resulting from GUI use and demonstrate their utility in Linux forensic examinations. Taking a cue from the *SANS Evidence Of* philosophy (Lee, 2019), this paper will focus on the following actions:

- Desktop App Installation
- Desktop App Execution
- File Opening/File Existence
- USB Flash Drive Insertion
- Wi-Fi Network Join
- Google Drive Upload

Other GUI actions such as browser usage and email will not be covered in this paper because the data structures for these artifacts are similar to those found on Windows systems, and they have been thoroughly analyzed and documented (Lee, 2017).

In Section 2, the Linux artifacts for the above-listed categories will be described. In Section 3, three scenarios will be performed:

Network Reconnaissance: subject installs a GUI-based scanner and scans another system.

Inappropriate use: subject attaches a LUKS encrypted USB flash drive and opens files on the flash drive.

Theft of Data: subject mounts a Google Drive and uploads files.

Afterward, a post-mortem analysis of each scenario will be conducted, and the benefits and limitations of these Linux artifacts will be discussed.

Brian Nishida brian.nishida@student.sans.edu

To narrow the scope of this research, a specific Linux distribution and desktop environment must be selected. According to Distrowatch (2020), the most popular Linux distribution is *MX Linux*. However, DistroWatch's webpage explains: "The DistroWatch Page Hit Ranking statistics are a *light-hearted* way of measuring the popularity of Linux distributions and other free operating systems among the visitors of this website" (2020). DistroWatch analyzed the number of visits to a Linux distribution webpage, not downloads. For this reason, Vaughan-Nichols (2018) looked at Google Trends statistics and found that the Linux distribution that most people wanted to know about was, by far, Ubuntu Linux.

Hence, this paper will examine artifacts on the latest stable version of Ubuntu Linux, 20.04 LTS, and its default Gnome Desktop Environment (GDE). The majority of the artifacts described are also applicable to Ubuntu 16, Ubuntu 18, and other Debian distributions. Following this paper's guidance, a forensic analyst can systematically locate artifacts for other Linux distributions and desktop environments.

2. Gnome Desktop Environment (GDE) Artifacts Overview

Over the years, Gnome developers incorporated additional convenience and performance features into the GDE. For example, Gnome will remember recent files, save Wi-Fi network profiles, and create an index of the data on the local drive. The files associated with these features, along with Linux logs, form the basis of the GDE artifacts.

The method for finding these GDE artifacts is detailed in Appendix A. In essence, a snapshot of the Ubuntu system is taken, a user action is performed, and another snapshot is taken. The difference in the snapshots identifies files that changed from both the user's activities and normal operating system functions. Consequently, each candidate file is examined to determine if it is a genuine GDE artifact, i.e., a result of the user's actions.

2.1. Recurring Log files

Several Linux log files appear repeatedly and are described below. First, these three log files record vital Linux messages (Canonical Ltd., 2020):

- `/var/log/auth.log` sudo commands, password prompts, remote logins
- `/var/log/kern.log` kernel logs
- `/var/log/syslog` system logs

These logs are text files that may be exported and viewed in a text editor. Moreover, Linux performs log rotations so that multiple versions of these logs will exist, e.g., `kern.log, kern.log.1, kern.log.2`.

Next, there are three text-formatted logs associated with application installations:

- `/var/log/dpkg.log`
- `/var/log/apt/history.log`
- `/var/log/apt/term.log`

Finally, there are several journal logs (Canonical Ltd., 2019):

- `/var/log/journal/<long number>/system.journal`
- `/var/log/journal/<long number>/user-<uid>.journal`

These journals are binary files and contain some of the same messages as `kern.log` and `syslog`. In digital forensics, multiple copies of the same log messages are advantageous if some logs have been overwritten or deleted. These journal logs may be exported and converted into text files on an Ubuntu (forensic) system using the command `journalctl --file=<journal file>`.

This research demonstrates how to search these logs for specific artifacts by using particular keywords.

2.2. Artifacts of Desktop App Installation

Regarding desktop app installation, forensic analysts are interested in the following information (Lee, 2017):

- What desktop apps are installed?
- When were the desktop apps installed?
- Who installed the desktop apps?

This section describes how to obtain this information.

Most cybersecurity practitioners are familiar with the command-line installation of applications: `apt install`, or `dpkg -i`, or `snap install`. However, the point-and-click method of installing an application can be executed in the Ubuntu Software Center. In the Ubuntu Software Center, the user searches for an application, clicks on "Install," and is prompted for the `sudo` password. Whether installing by command-line or the Ubuntu Software Center, some of the same installation artifacts are generated.

In GDE parlance, "desktop apps" are applications that are executed by clicking on an icon. To register the application with the GDE, a `.desktop` file containing configuration data is created when the desktop app is installed (The Gnome Project, 2014). For example, `wireshark.desktop` is the `.desktop` file corresponding to the Wireshark app. Consequently, one of the fastest ways to locate all desktop apps is to search for `*.desktop` in the entire file system. The following directories are the most common locations for `.desktop` files:

- `/snap/`
- `/usr/share/applications/`

After the forensic analyst identifies apps of interest, they may search the application installation logs (`dpkg.log`, `history.log`, and `term.log`) for specific desktop app names. All of these installation logs record the installation date as well as package dependencies. Moreover, the `history.log` records the user identification (UID) of the user who installed the application.

Brian Nishida brian.nishida@student.sans.edu

The forensic analyst can also search `syslog` and `system.journal` for messages containing the words "install" or "installing" and the desktop app name. In both of these logs, the installation date and the UID are recorded.

A summary of these GDE installation artifacts is listed below:

Artifact	Search Term(s)	Install Date	Attribution
`dpkg.log`	desktop app name	yes	no
`history.log`	desktop app name	yes	yes
`term.log`	desktop app name	yes	no
`syslog`	installing or install desktop app name	yes	yes
`system.journal`	installing or install desktop app name	yes	yes

Table 1: Desktop App Installation Artifacts

The `auth.log` records that `sudo` privileges were activated during the time of the installation; however, installation messages are not recorded in this log.

2.3. Artifacts of Desktop App Execution

For desktop app execution, forensic analysts are interested in the following information (Lee, 2017):

- Who executed the desktop app?
- What desktop apps did a specific user execute?
- When was the desktop app executed?

This section describes how to locate this information.

Desktop app execution is recorded in Linux logs and GDE configuration files. Initially, some apps require `sudo` privileges to launch so that the forensic analyst can search the `auth.log` for desktop app names. The `auth.log` records the time of app execution as well as the user.

Also, the `syslog` and `system.journal` record when desktop apps are executed. When a desktop app is finished, these logs record the message “gnome-launched” with the app name. Therefore, the forensic analyst can search for "gnome-launched" and work backward through the log to ascertain the start of execution. The forensic analyst can also search for the desktop app name to find the beginning of the app execution. In both of these logs, the execution date and the UID/username are recorded.

In addition to the logs, there are GDE configuration files that indicate desktop app execution. To understand these configuration files, the desktop environment's features should be highlighted (Helmke, 2019, Chapter 3). GDE users display desktop apps by clicking on the Activities icon, which consists of a 3-by-3 matrix of dots (see Figure 1). After clicking on Activities, the user can choose the tab "All" for all apps or "Frequent" for frequently used apps. In Figure 1, the Frequent Apps tab is displayed. Finally, users may pin their favorite apps to the "Dash," which is the sidebar on the left (Figure 1).

Frequent App data is stored for each user in the text file: `/home/<username>/.local/share/gnome-shell/application_state`.

The Frequent Apps (`application_state`) corresponding to Figure 1 are displayed below.

```
<application id="org.gnome.Evince.desktop" score="0" last-seen="1601833789"/>
<application id="snap-store_ubuntu-software.desktop" score="154" last-
seen="1601832688"/>
<application id="org.gnome.Nautilus.desktop" score="2" last-seen="1601838627"/>
<application id="firefox.desktop" score="2" last-seen="1601831367"/>
<application id="org.gnome.Terminal.desktop" score="556" last-
seen="1601838624"/>
<application id="libreoffice-writer.desktop" score="4" last-seen="1601835478"/>
<application id="zenmap.desktop" score="4" last-seen="1601831364"/>
```

The last seen value (i.e., last executed date) can be parsed with the Linux command: `date-d @<number>`. The score is a relative use-value. Frequent Apps are continually changing; however, the presence of a particular desktop app with a high score indicates recurrent usage.

Figure 1: Activities, Frequent Apps, All Apps, Dash

Next, the Dash configuration data is stored in the user's dconf database, called "user," in the path below:

`/home/<username>/.config/dconf/user`

The dconf database is a binary file, but it can be exported and parsed with the strings-command. The forensic analyst can search for the keyword "forensic-apps" after the strings-command to see the apps pinned to the Dash. The Dash for Figure 1 is shown below:

```
sfavorite-apps
firefox.desktop
thunderbird.desktop
org.gnome.Nautilus.desktop
rhythmbox.desktop
libreoffice-writer.desktop
snap-store_ubuntu-software.desktop
yelp.desktop
org.gnome.Terminal.desktop
```

In the above examples, the `org.gnome.Terminal.desktop` app is both a Frequent App and a member of the user's Dash. Hence, it would be prudent to look at the shell history for this user.

The GDE artifacts for desktop app execution are summarized in the table below:

Artifact	Search Term(s)	Execution Date	Attribution
`auth.log`	desktop app name	yes	yes
`syslog`	desktop app name gnome-launched	yes	yes
`system.journal`	desktop app name gnome-launched	yes	yes
`application_state`	desktop app name	yes	yes
`dconf/user`	`strings user | grep favorite-apps -A10`	no	yes

Table 2: Desktop App Execution Artifacts

2.4. Artifacts of File Opening/File Existence

For file opening/file existence, forensic analysts are interested in the following information (Lee, 2017):

- Who opened the file?
- When was the file opened?
- Was the file ever stored on the volume?

This section describes how to acquire this information.

Forensic analysts often examine the last access timestamps of files to deduce file opening actions. However, Forensics Focus (2015) purports that the last access timestamp in ext4 is unreliable. During this research, several experiments were conducted, and it was determined that: 1) when a file is opened using the command-line (e.g., `vi file.txt`), the last access timestamp changes reliably; 2) when a file is opened with a desktop app, the last access timestamp may or may not change. The longer the file is open in the desktop app, the more likely the timestamp will change.

Fortunately, the GDE tracks recently opened files, creates thumbnails for files, and indexes documents, pictures, and multimedia files. Also, individual desktop apps may track recent files. All of these features are potential sources of GDE artifacts.

First, Recent Files will be explored. When a user opens the File Manager App, the upper left corner shows a "Recent" icon. When a user clicks on Recent, it shows icons for recently opened files (Figure 2).

The Recent Files information is stored for each user in the file:

```
/home/<username>/.local/share/recently-used.xbel
```

One entry in the `recently-used.xbel` file corresponding to Figure 2 is shown below:

```
<bookmark href="file:///home/user1/Pictures/cammy.jpeg" added="2020-10-
10T16:50:19Z" modified="2020-10-10T17:05:30Z" visited="1969-12-31T23:59:59Z">
```

```
<info>
  <metadata owner="http://freedesktop.org">
    <mime:mime-type type="image/jpeg"/>
    <bookmark:groups>
      <bookmark:group>Graphics</bookmark:group>
    </bookmark:groups>
    <bookmark:applications>
      <bookmark:application name="org.gnome.Nautilus"
exec="&apos;org.gnome.Nautilus %u&apos;" modified="2020-10-10T17:05:29Z"
count="2"/>
      <bookmark:application name="Image Viewer" exec="&apos;eog
%u&apos;" modified="2020-10-10T17:05:30Z" count="2"/>
    </bookmark:applications>
```

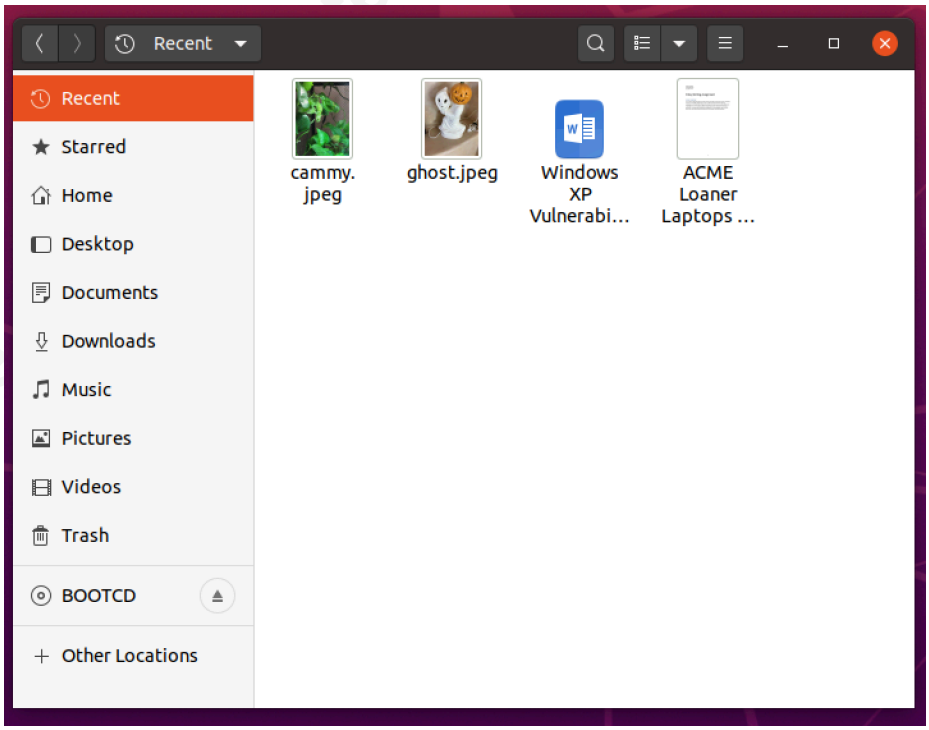


Figure 2: Recent Files

This XML file can be read with a text editor. There are several important pieces of information which are shown in bold text above and summarized below:

Source filename:	/home/user1/Pictures/cammy.jpeg
First time opened (added):	2020-10-10 16:50:19 Zulu time

Last time opened (modified):	2020-10-10 17:05:30 Zulu time
Application used:	Gnome Nautilus (a.k.a. File Manager)
Number of times opened:	2

The Recent Files settings are located in Settings > File History & Trash (see Figure 3). The default Ubuntu setting is to remember File History (i.e., Recent Files), and the default duration is forever!

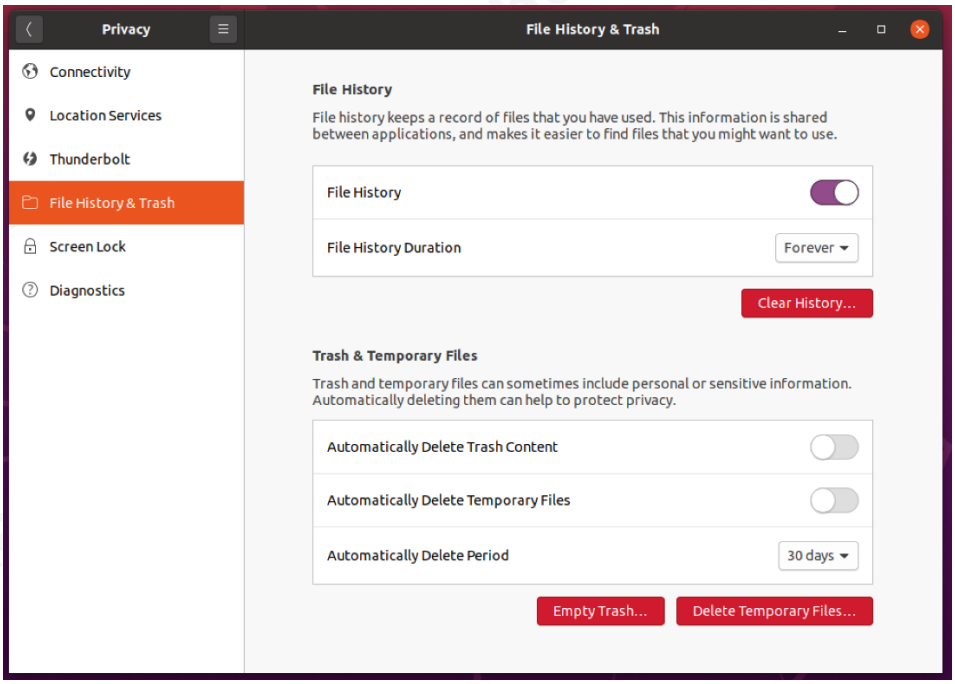


Figure 3: Privacy Settings

Moreover, if the source file is deleted, it will not appear in the Recent Files display (Figure 2). However, its entry will remain in `recently-used.xbel`.

In addition to the File Manager Recent Files, individual desktop apps may also store recent files history. For the LibreOffice suite of Apps, this information is stored in the file:

```
/home/<username>/.config/libreoffice/4/user/registrymodificatons.xcu
```

Forensic analysts can search for the term "HistoryItemRef" in `registrymodifications.xcu` to find recent files, including the full file path. This data will remain even if the source file is deleted. Other Desktop Apps will require the method described in Appendix A to locate application-specific recent file history.

Another GDE artifact is the thumbnails cache since the File Manager's default view is the thumbnails view. Thumbnails for each user are stored in the folder:

```
\home\<username>\.cache\thumbnails\
```

Whenever a user visits a folder, thumbnails of graphics and PDF files are created, even if the user never opens a file in the folder.

Moreover, the Gnome thumbnail specification (McCann, 2013) requires that the URL of the source file be placed in the thumbnail. If a forensic analyst performs a strings-command on the thumbnails and searches for the word "file," they will see the full paths of source files for the thumbnails. An example is shown below:

```
$ strings *.png | grep file
file:///home/user1/Pictures/3EA67133-0A90-4191-A762-
E87097679F26_1_105_c.jpeg
file:///home/user1/Pictures/F22%20Raptor.jpeg~6/
file:///home/user1/Pictures/Norway%20Fjords%201.jpg
file:///home/user1/Documents/Dept%20of%20Cyber%20Security%20SAP.pdfYQ
file:///home/user1/Pictures/50682B0B-D360-4D49-8D36-
9972607EDB5B_1_105_c.jpeg
file:///home/user1/Documents/ACME%20Loaner%20Laptops%20Policy.pdf
file:///home/user1/Pictures/Norway%20Fjords%203.jpg
```

Thumbnails are created for local folders and folders on external USB drives. These thumbnails will remain even if the source file is deleted. The thumbnail cache settings can be retrieved from the command-line:

```
$ gsettings list-recursively | grep "thumbnail-cache"
org.gnome.desktop.thumbnail-cache maximum-age 180
org.gnome.desktop.thumbnail-cache maximum-size 512
```

The default thumbnail cache age is set to 180 days with a cache size of 512 MB. Hence, the cache can store thousands of thumbnail files corresponding to directories that the user visited.

One last GDE feature of interest is the Gnome Tracker. Tracker is a file indexing and search engine framework (Tueco, 2015; Propst, 2020). When a user adds a document, photo, audio file, or video file to the local file system, Tracker automatically indexes it and adds the information to the Tracker database.

The Tracker database files associated with each user are stored in the locations:

```
/home/<username>/.cache/tracker/

/home/<username>/.local/share/tracker/data/
```

A user can view the information indexed by Tracker using the `tracker info <filename>` command. If the file is deleted, then `tracker info` will yield no results. However, Tracker uses a SPARQL database, and most databases retain deleted records for a while.

A forensic analyst can run strings against the `tracker-store.journal` file as a search for "Afile" records:

```
strings /home/<username>/.local/share/tracker/data/tracker-store.journal | grep Afile
```

This will retrieve all full-path filenames, both active and deleted files indexed by Tracker. Hence, the filenames of all documents, photos, audio files, and video files on the local drive, active or deleted, can be recovered.

As an example of analytic pivoting, there is a scenario where a user installed a desktop app, executed the app, and uninstalled the app. Using Tracker artifacts, the forensic analyst can search for all `*.desktop` files to identify desktop apps once installed and subsequently deleted.

Brian Nishida brian.nishida@student.sans.edu

File opening/file existence artifacts are summarized in Table 3.

Artifact	File Path	File Opening Date	Attribution
recently-used.xbel	yes	yes	yes
registrymodifications.xcu	yes	no	yes
thumbnail cache	yes	no	yes
tracker-store.journal	yes	no	yes

Table 3: File Opening/File Existence Artifacts

2.5. Artifacts of USB Drive Insertion

Forensic analysts are interested in the following information when a USB flash drive has been inserted into a computer (Lee, 2017):

- Manufacturer
- Make
- Model
- Serial Number
- VID & PID
- Volume Name
- Mount Point
- UID associated with the insertion
- Date/time of drive insertion
- Filenames of stored files

The section describes how to retrieve this information.

In Windows Analysis, the retrieval of this information is a multi-step process (Lee, 2017). In Linux, this process is simpler, as the `kern.log`, `syslog`, and `system.journal` each contain copies of the desired USB drive information (except for the filenames of stored files).

The forensic analyst can search each of these logs for the string "New USB device" and then read subsequent lines in the log. These "New USB device" log entries occur even if the same USB flash drive has been inserted multiple times.

Additionally, when a USB flash drive is inserted into an Ubuntu system, the default setting is to auto-mount the volume to the mount point:

```
/media/<username>/<volume name>
```

In the File Opening/File Existence Artifacts listed in Section 2.5, files' full paths are usually seen. Hence, forensic analysts can see USB drive insertion indicators and USB drive stored files if the file path contains: `/media/<username>/<volume name>`.

The USB drive insertion artifacts are summarized below:

Artifact	Search Term(s)	Insertion Date	Attribution
`kern.log`	New USB device	yes	yes
`syslog`	New USB device	yes	yes
`system.journal`	New USB device	yes	yes
File Opening Artifacts	/media/<username>	no	yes

Table 4: USB Drive Insertion Artifacts

2.6. Artifacts of Wi-Fi Network Join

Forensic analysts are interested in the following information regarding Wi-Fi network joining (Lee, 2017):

- Timestamp of connection
- SSID Name
- UID of the user making the connection
- MAC address of Wi-Fi card
- Wi-Fi security type (WEP, WPA, WPA2, etc.)
- Interface name
- Domain Name
- DNS servers
- DHCP address and other DHCP information

This section describes how to retrieve this information.

When a user joins a Wi-Fi network, several log file artifacts are created. A forensic analyst can search for the string “authenticate with” in the `kern.log` file to find timestamps with Wi-Fi MAC addresses. However, no SSID or user information is available. On the other hand, if the forensic analyst searches `syslog` or `system.journal` for the string “starting connection," all of the bullet list's desired information is presented.

In addition to these logs, a Wi-Fi configuration file is created in:

```
/etc/NetworkManager/system-connections/<SSID-name>.nmconnection
```

This file contains the Wi-Fi access point password in cleartext. Its only protection is that it is owned by root. However, in a post-mortem forensic examination, a forensic analyst can easily read the file.

Wi-Fi connection artifacts are summarized below.

Artifact	Search Term(s)	SSID Name	Attribution
`kern.log`	authenticate with	no	no
`syslog`	starting connection	yes	yes
`system.journal`	starting connection	yes	yes
`<SSID-name>.mconnection`	--	yes	no

Table 5: Wi-Fi Connection Artifacts

2.7. Artifacts of Google Drive Upload

For cloud storage, forensic analysts are interested in the following information:

- Who is the cloud provider?
- What is the cloud account name or identifier?
- What files have been uploaded?

This section describes how to locate this information.

The Ubuntu Software Center has apps for Dropbox, iCloud, and other cloud providers. However, this research will look at one of the default cloud providers in Ubuntu Linux, Google Drive.

To mount a Google Drive, users navigate to Settings > Online Accounts > Google, and then authenticate to their google account (see Figure 4).

When a user mounts their Google Drive, the configuration is stored in the file:

`/home/<username>/.config/goa-1.0/accounts.conf`

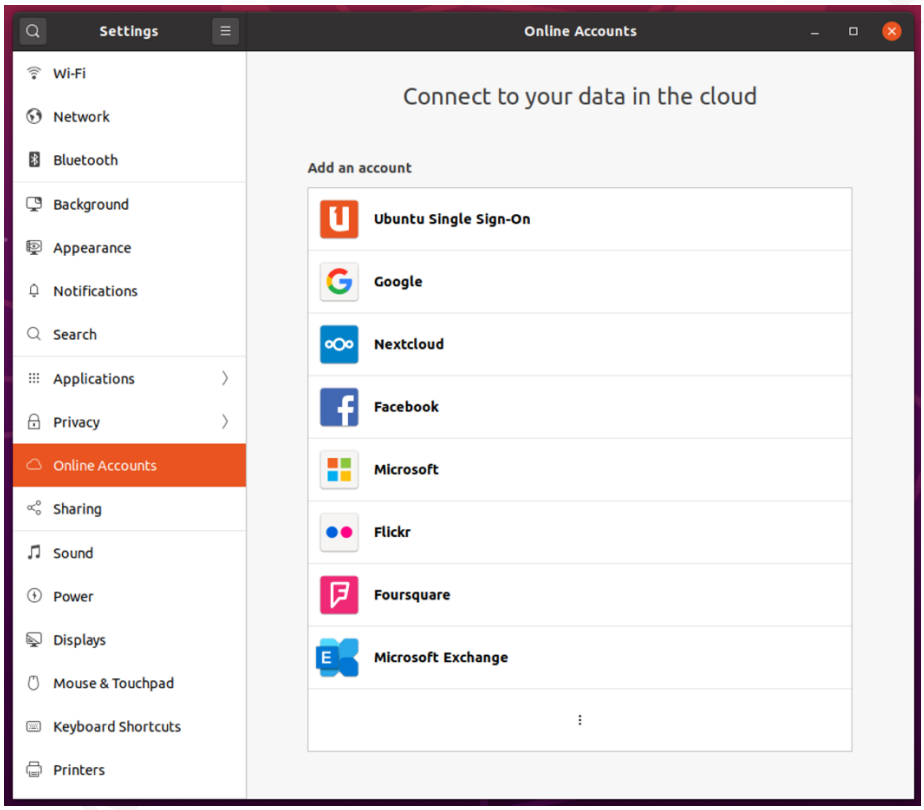


Figure 4: Google Drive set-up

In this configuration file, the Google account name is visible in plain text. The account name may be forwarded to law enforcement or a company's legal department for further action.

Unfortunately, there are no artifacts that log when files are uploaded to Google Drive. Also, the thumbnail cache or the Gnome Tracker do not track files on Google Drive.

Fortunately, file opening/file existence artifacts such as `recently-used.xbel` file and the `registrymodifications.xcu` store the full paths of Google Drive files that were opened. These artifacts can provide a glimpse of some of the filenames located on Google Drive.

The Google Drive artifacts are summarized below.

Brian Nishida brian.nishida@student.sans.edu

Artifact	Google Account	Files on Google Drive	Attribution
`accounts.conf`	yes	no	yes
`recently-used.xbel`	yes	yes	yes
`registrymodifications.xcu`	yes	yes	yes

Table 6: Google Drive Artifacts

3. Three Practical Scenarios

The GDE artifacts corresponding to different user actions have now been cataloged. The artifacts originate from log files (Section 2.1), GDE configuration files, application-specific configuration files, and the Gnome Tracker database. However, it is essential to analyze several test cases to assess the value of GDE artifacts.

First, the Ubuntu 20.04 LTS operating system was installed on a laptop computer, and the operating system was updated and upgraded. Three user accounts were also created: alex, ted, and cindy.

Next, the three scenarios were performed from October 14, 2020, to October 16, 2020. After the scenarios were completed, a forensic E01 image of the laptop's hard drive was created.

Then, a Kali Linux 2020.2 system was used to:

- mount the E01 image,
- run the strings-command on thumbnails, `dconf/user`, and `tracker-store.journal`,
- run the journalctl-command on `system.journal`, and
- export the log files and the resulting text files.

Finally, a simple text editor was used to extract GDE artifacts from the text files. The results will show that GDE artifacts capture the majority of the user actions performed in these scenarios.

3.1. Scenario 1: Network Reconnaissance

On October 14, 2020, the Security Operations Center detected a scan of a company web server from an internal Ubuntu 20.04 system. The Incident Response Team created a forensic image of the suspect Linux system and requested that the Forensics Team conduct a post-mortem analysis. The IR Team requested answers to the following questions:

- Was scanning software installed? If so,
- Who installed the scanning software?
- Who used the scanning software?

The Forensics Team searched the forensic image for *.desktop files, and located three desktop scanning apps:

```
/var/lib/snapd/desktop/applications/zaproxy_zaproxy.desktop
/usr/share/applications/zenmap.desktop
/usr/share/applications/zenmap-root.desktop
```

Next, the team searched `dpkg.log`, `history.log`, `term.log`, `syslog`, and `system.journal` for "zaproxy" and "zenmap" installation messages. In `syslog.2.gz`, the installation of zaproxy is recorded:

```
Oct 14 09:10:04 TEST-LAPTOP kernel: [  217.486907] audit:
type=1326 audit(1602688204.320:9581): auid=1000 uid=1000 gid=1000
ses=3 pid=4280 comm="pool-org.gnome." exe="/snap/snap-
store/481/usr/bin/snap-store" sig=0 arch=c000003e syscall=251
compat=0 ip=0x7f0b0a629959 code=0x50000
Oct 14 09:10:08 TEST-LAPTOP snapd[620]: api.go:999: Installing
snap "zaproxy" revision unset
```

In `history.log`, the installation of zenmap is recorded:

```
Start-Date: 2020-10-14  09:22:05
Commandline: apt install ./python-gtk2_2.24.0-6_amd64.deb
./zenmap_7.80+dfsg1-1build1_all.deb
Requested-By: alex (1000)
```

Both of these scanning apps were installed by user alex (UID 1000) on October 14, 2020.

Next, in `auth.log`, there is a message for launching zenmap:

```
SHOT authorization for action com.ubuntu.zenmap for unix-
process:1744:4366 [/usr/bin/gnome-shell] (owned by unix-user:alex)
Oct 14 09:35:14 TEST-LAPTOP pkexec: pam_unix(polkit-1:session):
session opened for user root by (uid=1000)
Oct 14 09:35:14 TEST-LAPTOP pkexec[2459]: alex: Executing
command [USER=root] [TTY=unknown] [CWD=/home/alex]
[COMMAND=/usr/bin/zenmap]
```

And in `syslog.2`, there is a message for launching zaproxy:

```
Oct 14 09:44:22 TEST-LAPTOP zaproxy_zaproxy.desktop[2781]: 370
[main] INFO org.zaproxy.zap.GuiBootstrap  - OWASP ZAP 2.9.0 started
14/10/20 09:44:22 with home /home/alex/.ZAP/
```

Zaproxy is terminated after running for several hours.

```
Oct 14 11:47:38 TEST-LAPTOP zaproxy_zaproxy.desktop[2781]:
7396310 [ZAP-Shutdown] INFO org.parosproxy.paros.control.Control  -
OWASP ZAP 2.9.0 terminated.
```

```
Oct 14 11:47:38 TEST-LAPTOP systemd[1464]: gnome-launched-
zaproxy_zaproxy.desktop-2617.scope: Succeeded.
```

Recall that the "gnome-launched" message occurs at the *termination* of the app. Next, the Forensics Team looked at the Dash for user "alex", but did not find scanner apps. Finally, the Forensics Team examined Frequent Apps (`application_state`). The following entries were located:

```
      <application id="zenmap.desktop" score="30" last-
seen="1602689936"/>
          <application id="zaproxy_zaproxy.desktop" score="5" last-
seen="1602690316"/>
```

The Forensics Team ran the `date -d @<number>` command showing zenmap was last run on October 14, 2020, at 9:38:56 am MDT, and zaproxy was last run on October 14, 2020, at 9:45:16 am MDT.

The Forensics Team concluded that two scanning apps (zenmap and zaproxy) were installed by alex (UID 1000) on October 14, 2020, and launched by alex shortly after that. Hence, the GDE artifacts accurately depicted what happened in this scenario.

3.2. Scenario 2: Inappropriate Use

The organization has a strict internal policy that prohibits viewing reptile pictures on company computers. On October 15, 2020, an employee of the company observed a coworker named Ted looking at reptile photos on his Ubuntu 20.04 company laptop. The IT Team perused the laptop and found no reptile photos. The IT Team also located three USB flash drives on Ted's desk. Two of the USB drives contained no reptile photos. The third USB flash drive, a SanDisk Cruzer, was password protected with LUKS encryption and could not be viewed.

Because the IT Team was not familiar with Linux, the IT supervisor requested that the Forensics Team examine the laptop. The IT supervisor requested answers to the following questions:

- Did Ted view reptile photos on the company laptop? If so,
- How many reptile photos were there?
- In what folder(s) were they located?

Initially, the Forensics Team looked at user ted's Recent Files (`recently-used.xbel`), but only the filenames are logged. Below are three JPEG files referenced as recent files:

```
      <bookmark href="file:///media/ted/LUKS/IMG_0853.jpeg"
added="2020-10-15T14:00:43Z" modified="2020-10-15T14:00:44Z"
visited="1969-12-31T23:59:59Z">
         <info>
           <metadata owner="http://freedesktop.org">
             <mime:mime-type type="image/jpeg"/>
…
<bookmark href="file:///media/ted/LUKS/IMG_1013.jpeg" added="2020-
10-15T14:00:51Z" modified="2020-10-15T14:00:52Z" visited="1969-12-
31T23:59:59Z">
    <info>
      <metadata owner="http://freedesktop.org">
        <mime:mime-type type="image/jpeg"/>
…
<bookmark href="file:///media/ted/LUKS/IMG_1019.jpeg" added="2020-
10-15T14:00:56Z" modified="2020-10-15T14:00:57Z" visited="1969-12-
31T23:59:59Z">
    <info>
      <metadata owner="http://freedesktop.org">
        <mime:mime-type type="image/jpeg"/>
```

Because the JPEG filenames were non-descriptive, the Forensics Team moved on. All three recent photos were located on an external device with a volume name of LUKS. The Forensics Team searched the `system.journal` for string "New USB device" to gather additional information about the device.

```
      Oct 15 08:00:14 TEST-LAPTOP kernel: usb 1-3: New USB device found,
idVendor=0781, idProduct=5575, bcdDevice= 1.00
      Oct 15 08:00:14 TEST-LAPTOP kernel: usb 1-3: New USB device strings:
Mfr=1, Product=2, SerialNumber=3
```

```
Oct 15 08:00:14 TEST-LAPTOP kernel: usb 1-3: Product: Cruzer Glide
Oct 15 08:00:14 TEST-LAPTOP kernel: usb 1-3: Manufacturer: SanDisk
Oct 15 08:00:14 TEST-LAPTOP kernel: usb 1-3: SerialNumber: XXXXXXXXXXXXX
Oct 15 08:00:14 TEST-LAPTOP kernel: usb-storage 1-3:1.0: USB Mass Storage
device detected
…
Oct 15 08:00:15 TEST-LAPTOP kernel: sd 5:0:0:0: [sdb] Attached SCSI
removable disk
Oct 15 08:00:22 TEST-LAPTOP udisksd[795]: Unlocked device /dev/sdb as
/dev/dm-0
Oct 15 08:00:22 TEST-LAPTOP systemd[1]: Finished Clean the
/media/ted/LUKS mount point.
Oct 15 08:00:22 TEST-LAPTOP udisksd[795]: Mounted /dev/dm-0 at
/media/ted/LUKS on behalf of uid 1001
```

The USB device was identified as the LUKS encrypted SanDisk Cruzer USB flash drive. It was attached by user ted (UID 1001) on October 15, 2020.

Still, the contents of these JPEG files were not known. Next, the Forensics Team looked at the Gnome thumbnail cache for user ted. Three reptile thumbnails were observed (Figure 5). Looking at the metadata in each of these png files, the Forensics Team identified the source file names, including the full paths (Table 7). The thumbnails were then matched to files in `recently-used.xbel`.

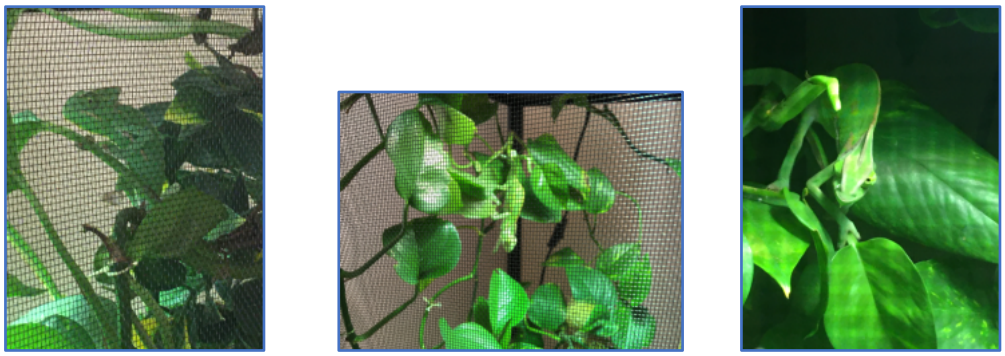


Figure 5: Thumbnails of reptile photos

Thumbnail File	Metadata (Source File Name)
9d6e5069a0a3087ea0a5f0d7e47aa7ed.png	file:///media/ted/LUKS/IMG_0853.jpeg
47ef5f6fa15914b07a7ed067a32085a3.png	file:///media/ted/LUKS/IMG_1019.jpeg
eba58a8b530105e68b2210d420af14f2.png	file:///media/ted/LUKS/IMG_1013.jpeg

Table 7: Gnome thumbnails and metadata

In summary, the Forensics Team determined that a LUKS-encrypted Sandisk USB flash drive was mounted on the Ubuntu system by user ted on October 15, 2020. Three reptile photos were located on the encrypted Sandisk USB flash drive, and the user ted opened all three of the reptile photos. Furthermore, user ted has knowledge of the LUKS password. Once again, the GDE artifacts accurately tracked what happened in this scenario.

3.3. Scenario 3: Theft of Data

On October 16, 2020, law enforcement advised the company that some of their proprietary data was for sale on a Dark Web forum. The files posted were: *Top Secret ACME Customer List.xlsx, Top Secret ACME Formulas.docx*, and *Top Secret ACME Strategic Plan.pdf*. Event logs and inventory records indicated an SMB connection between the Top Secret data folder and an Ubuntu 20.04 company system.

The Incident Response Team requested that the Forensics Team examine this Linux computer. The questions posed are:

- Were the stolen files on this Ubuntu system? If so,
- Who was responsible?
- Where were the files transferred to? USB flash drive? Cloud storage?

The Forensics Team initially searched the system for the stolen files, but they were not located in the file system.

Next, the Forensics Team examined the `recently-used.xbel` files for each user. Although there were numerous recent files, the stolen files were not among them. This indicates that the files were not opened on the system.

Then, the Forensics Team looked at the thumbnail cache for each user because PDF files generate thumbnails. The metadata for one of the Gnome thumbnails referenced one of the stolen files:

```
file:///home/cindy/Documents/Top%20Secret%20ACME%20Strategic%20
Plan.pdf
```

Thumbnails, unfortunately, will not be captured for documents and spreadsheets. However, the Forensics Team concluded that the stolen PDF was located in the Documents folder for user "cindy."

Next, the Forensics Team ran the Gnome Tracker (`tracker-store.journal`) through the strings-command and searched for Afile records. All three stolen files were located:

```
Afile:///home/cindy/Documents/Top%20Secret%20ACME%20Customer%20List.xlsx
Afile:///home/cindy/Documents/Top%20Secret%20ACME%20Formulas.docx
Afile:///home/cindy/Documents/Top%20Secret%20ACME%20Strategic%20Plan.pdf
```

This indicates that all of the stolen files were once on the Ubuntu system in cindy's Documents but have been deleted. At this point, the Forensics Team can data carve for the stolen files for additional confirmation.

Unfortunately, there are multiple exfiltration avenues. The Forensics Team can search the `system.journal` for USB flash drives; look for cloud storage apps, and examine browser history for cloud storage sites. However, for demonstration purposes, the Forensics Team looked at Wi-Fi and Google Drive artifacts.

The Forensics Team searched `syslog` and `system.journal` for the string "starting connection." The following Wi-Fi hotspot connection was located:

```
Oct 16 07:08:56 TEST-LAPTOP NetworkManager[762]: <info>
[1602853736.8752] device (wlp2s0): Activation: starting connection 'Verizon-
791L-XXXX' (9c5f64e2-edd3-452f-95c6-a11ed711e8db)
Oct 16 07:08:56 TEST-LAPTOP NetworkManager[762]: <info>
[1602853736.8754] audit: op="connection-add-activate" uuid="9c5f64e2-edd3-452f-
95c6-a11ed711e8db" name="Verizon-791L-XXXX" pid=1691 uid=1002 result="
…
```

The SSID for this connection is Verizon-791L-XXXX. If this connection is saved, it will be stored in `/etc/NetworkManager/system-connections/Verizon-791L-XXXX.nmconnection`.

Finally, the Forensics Team searched for Google Drive artifacts. The Google Drive configuration file is located in: `/home/cindy/.config/goa-1.0/accounts.conf`. They located this file, and it contained the following information:

```
[Account account_1602853944_0]
Provider=google
Identity=cindy.insidertheft@gmail.com
PresentationIdentity=cindy.insidertheft@gmail.com
MailEnabled=true
CalendarEnabled=true
ContactsEnabled=true
DocumentsEnabled=true
PhotosEnabled=true
FilesEnabled=true
PrintersEnabled=true
```

In conclusion, the Forensics Team determined that the three stolen files were once on the Ubuntu system in the Documents folder of the user cindy. On October 16, 2020,

the cindy account (UID 1002) joined a Verizon Wireless hotspot network. Next, the user cindy mounted a Google Drive with account `cindy.insidertheft@gmail.com`. However, the Forensics Team cannot determine that the stolen files were actually uploaded to Google Drive.

4. Conclusion

The three scenarios demonstrate how GDE artifacts can enrich the classic Linux examination. For example, a forensic analyst can determine the contents of graphics files on an encrypted USB drive using the Gnome thumbnail cache, thumbnail metadata, and Recent Files. Alternatively, a forensic analyst can identify unopened, deleted files on a system using thumbnail metadata, and Gnome Tracker. Furthermore, the only tools required for this analysis are several Linux commands and a text editor.

However, there are some limitations to using GDE artifacts. First, Linux logs are kept for only four to five weeks (Pomeranz, 2012), an inconveniently brief window between the time of an incident and the forensic analysis. Also, GDE artifacts cannot determine if files were uploaded to Google Drive or perhaps any cloud storage. Finally, although it was successful, using the strings-command to parse `dconf/user` and `tracker-store.journal` is somewhat unpolished. Consequently, an area for future research might be to develop tools to decode the dconf database and Gnome Tracker for post-mortem examinations. Despite these limitations, GDE artifacts can significantly enhance the classic Linux examination, and they deserve a place in every forensic analyst's repertoire.

References

Altheide, Cory, and Harlan Carvey (2011). *Digital Forensics with Open Source Tools*, Chapter 5: Linux Systems and Artifacts. Elsevier, Waltham, MA.

Canonical Ltd. (2019). *Ubuntu Manuals: journalctl.* Retrieved from: http://manpages.ubuntu.com/manpages/focal/man1/journalctl.1.html

Canonical Ltd. (2020). *Viewing and monitoring log files*. Retrieved from: https://ubuntu.com/tutorials/viewing-and-monitoring-log-files#1-overview

Davies, Rhys (April 22, 2020). *Ubuntu 20.04 Survey Results*. Retrieved from: https://ubuntu.com/blog/ubuntu-20-04-survey-results

DistroWatch (October 4, 2020). *DistroWatch Page Hit Ranking.* Retrieved from: https://distrowatch.com/dwres.php?resource=popularity

Federal Bureau of Investigation (2018). *2018 Annual Report: Regional Computer Forensics Laboratory, Top 5 Crimes for each RCFL*. Retrieved from: https://rcfl.gov/downloads.

Finke, Jens (May 2012). *Thumbnail Managing Standard*. Retrieved from: https://specifications.freedesktop.org/thumbnail-spec/thumbnail-spec-latest.html

Forensics Focus (2015). *Linux Timestamps, Oh Boy!*. Retrieved from: https://www.forensicfocus.com/articles/linux-timestamps-oh-boy/

Fung, James (2013). *Dead Linux Machines Do Tell Tales*. SANS Institute, Bethesda, MD.

Helmke, Matthew; Hudson, Andrew; Hudson, Paul (2019). *Ubuntu Unleashed: 2019 Edition*. Pearson Education, Indianapolis, IN.

Le, Tho (2019). *Linux Forensics — Some Useful Artifacts*. Retrieved from: https://medium.com/@tho.le/linux-forensics-some-useful-artifacts-74497dca1ab2

Lee, Rob (2019). *SANS DFIR Windows Forensic Analysis Poster*. SANS Institute, Bethesda, MD.

Lee, Rob (2017). *SANS FOR500: Windows Forensic Analysis*. SANS Institute, Bethesda, MD.

Mader, Robert (2019). *Gitlab - Gnome*. Retrieved from: https://gitlab.gnome.org/GNOME/gnome-shell/-/blob/master/src/shell-app-usage.c

McCann, William Jon (December 2, 2013). *Gnome Project: Thumbnail management DBus specification.* Retrieved from: https://wiki.gnome.org/action/show/DraftSpecs/ThumbnailerSpec?action=show&redirect=ThumbnailerSpec

Nikkel, Bruce (2018). *Forensic Artifacts in Modern Linux Systems*. Bern University of Applied Sciences, Division of Computer Science (September 10, 2018).

Nishida, Brian (2019). PowerShell Snapshots: Windows, macOS, and Linux. Retrieved from: https://www.github.com/briannishida/win_snap

Patil, Dinesh N., and Meshram, B. B. (2015). *Forensic Investigation of User Activities on Windows7 & Ubuntu 12 Operating System*. International Journal of Innovations in Engineering and Technology (IJIET), Volume 5, Issue 3, Madhya Pradesh, India.

Patil, Dinesh, and Mehram, Bandu B. (2016). *Digital Forensic Analysis of Ubuntu File System*. International Journal of Cyber-Security and Digital Forensics (IJCSDF), Hong Kong, China.

Pogue, Chris; Altheide, Cory; and Haverkos, Todd (2008). *Unix and Linux Forensic Analysis DVD Toolkit*. Elsevier, Inc., Burlington, MA.

Pomeranz, Hall (2012). *Linux Forensics (for Non-Linux Folks)*. Retrieved from: http://www.deer-run.com/~hal/LinuxForensicsForNon-LinuxFolks.pdf

Propst, Oliver (June 25, 2020). *Gnome Project: Tracker*. Retrieved from: https://wiki.gnome.org/Projects/Tracker

The Gnome Project (2014). *Desktop files: putting an application in the desktop menus*. Retrieved from: https://developer.gnome.org/integration-guide/stable/desktop-files.html.en

Tueco, Francesco (October 10, 2015). *Gnome Project: What is Tracker?* Retrieved from: https://wiki.gnome.org/Projects/Tracker/WhatIsTracker

Vaughan-Nichols, Steven J. (April 23, 2018). *What's the most popular Linux of them all?* Retrieved from: https://www.zdnet.com/article/whats-the-most-popular-linux-of-them-all/

Appendix A: Methodology for Identifying GDE Artifacts

This research uses PowerShell snapshots and Python difference code to identify a computer system's changes when a user interacts with it (Nishida, 2019). A PowerShell script captures the state of the system in the form of text files. A Python script computes the differences between these text files.

The advantage of using PowerShell in Linux is that certain file attributes can be easily specified to create tailored file listings. In this research, the following state files were saved:

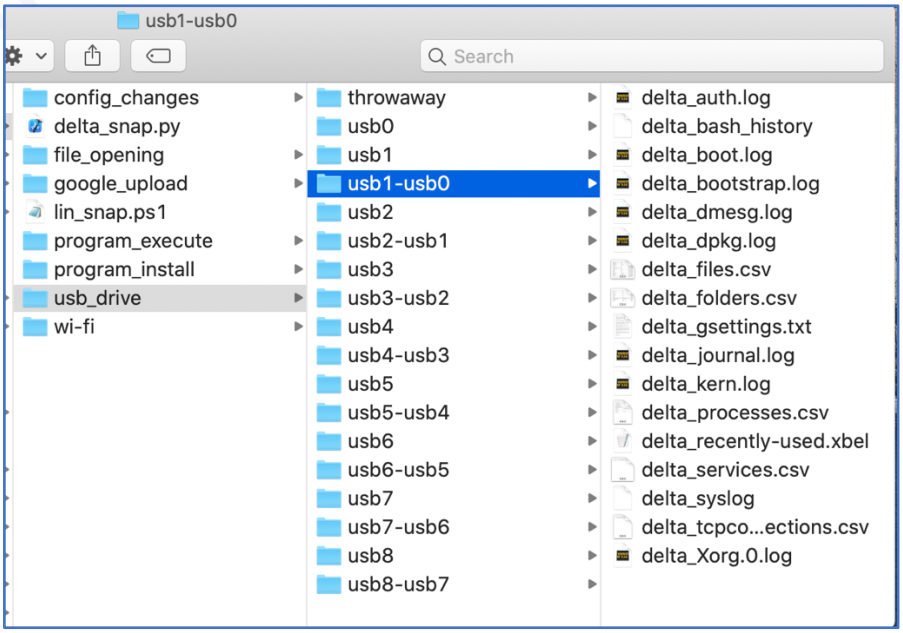
- `auth.log`
- `boot.log`
- `bootstrap.log`
- `dmesg.log`
- `dpkg.log`
- `kern.log`
- `syslog`
- `Xorg.0.log`
- `recently-used.xbel`
- `journalctl > journal.log`
- `gsettings list-recursively > gsettings.txt`
- File Listing (path, filename, size, creation date, modification date)
- Folder Listing (path, folder name, creation date, modification date)
- Not used (`bash_history`, processes, services, network connections)

As an example, the USB drive insertion experiment is shown in the table below:

Timestamp	Action
	Restore VM snapshot: Programs Installed
09/20/2020 @ 14:20	Powershell snapshot: throwaway
09/20/2020 @ 14:22	Powershell snapshot: usb0
09/20/2020 @ 14:25	Insert Transcend USB flash drive
09/20/2020 @ 14:30	Powershell snapshot: usb1
09/20/2020 @ 14:32	Eject Transcend USB flash drive

09/20/2020 @ 14:37	Powershell snapshot: usb2
09/20/2020 @ 14:41	Insert Sandisk USB flash drive
09/20/2020 @ 14:45	Powershell snapshot: usb3
09/20/2020 @ 14:48	Eject Sandisk USB flash drive
09/20/2020 @ 14:51	Powershell snapshot: usb4
09/20/2020 @ 14:54	Login as user2
09/20/2020 @ 14:58	Powershell snapshot: usb5
\09/20/2020 @ 15:00	Eject Transcend USB flash drive
09/20/2020 @ 15:04	Powershell snapshot: usb6
09/20/2020 @ 15:07	Login as user3
09/20/2020 @ 15:11	Powershell snapshot: usb7
09/20/2020 @ 15:14	Eject Sandisk USB flash drive
09/20/2020 @ 15:17	Powershell snapshot: usb8
	Copy off Powershell snapshots
	Image VM: usb.e01
	Re-baseline VM: Programs Installed

The PowerShell snapshots are copied off, and then the Python difference-calculating script is run on the snapshot folders:



There are no USB drive insertion artifacts in `auth.log`, `boot.log`, `bootstrap.log`, `dmesg.log`, `dpkg.log`, or `Xorg.0.log`.

However, in `kern.log`, `syslog`, and `system.journal`, the following artifacts are computed by the Python script:

```
Sep 20 14:25:19 test-system kernel: usb 3-2: New USB device found, idVendor=8564,
idProduct=1000, bcdDevice=11.00
Sep 20 14:25:19 test-system kernel: usb 3-2: New USB device strings: Mfr=1, Product=2,
SerialNumber=3
Sep 20 14:25:19 test-system kernel: usb 3-2: Product: Mass Storage Device
Sep 20 14:25:19 test-system kernel: usb 3-2: Manufacturer: JetFlash
Sep 20 14:25:19 test-system kernel: usb 3-2: SerialNumber: XXXXXXXXXXXXXX
Sep 20 14:25:19 test-system kernel: usb 3-3: new full-speed USB device number 4 using
xhci_hcd
Sep 20 14:25:19 test-system kernel: usb 3-3: New USB device found, idVendor=0e0f,
idProduct=0002, bcdDevice= 1.00
…
Sep 20 14:25:21 test-system kernel: scsi 33:0:0:0: Direct-Access     JetFlash Transcend
16GB   1100 PQ: 0 ANSI: 0 CCS
Sep 20 14:25:21 test-system kernel: scsi 33:0:0:0: Attached scsi generic sg3 type 0
Sep 20 14:25:21 test-system kernel: sd 33:0:0:0: [sdc] 31703040 512-byte logical blocks:
(16.2 GB/15.1 GiB)
Sep 20 14:25:21 test-system kernel: sd 33:0:0:0: [sdc] Write Protect is off
Sep 20 14:25:21 test-system kernel: sd 33:0:0:0: [sdc] Mode Sense: 43 00 00 00
Sep 20 14:25:21 test-system kernel: sd 33:0:0:0: [sdc] No Caching mode page found
Sep 20 14:25:21 test-system kernel: sd 33:0:0:0: [sdc] Assuming drive cache: write through
Sep 20 14:25:21 test-system kernel: sd 33:0:0:0: [sdc] Attached SCSI removable disk
Sep 20 14:25:21 test-system systemd[1]: Finished Clean the /media/user1/Transcend mount
point.
Sep 20 14:25:21 test-system udisksd[803]: Mounted /dev/sdc at /media/user1/Transcend on
behalf of uid 1000
```

Clearly, this data shows that a specific USB drive was inserted at the correct date/time. Therefore, these logs are USB drive insertion artifacts.

Next, observe the changes to the file listing:

```
/home/user1/.cache/event-sound-cache.tdb.45ed5e4032f441659631bcdead7b8595.x86_64-pc-linux-
gnu
/home/user1/.cache/tracker/meta.db
/home/user1/.cache/tracker/meta.db-shm
/home/user1/.cache/tracker/meta.db-wal
```

/home/user1/.config/dconf/user
/home/user1/.local/share/tracker/data/tracker-store.journal
…
/var/lib/udisks2/mounted-fs
/var/lib/systemd/timesync/clock
/var/lib/NetworkManager/timestamps
/var/log/auth.log
/var/log/kern.log
/var/log/syslog
/var/log/journal/45ed5e4032f441659631bcdead7b8595/system.journal
/var/log/journal/45ed5e4032f441659631bcdead7b8595/user-1000.journal

The changed files in the folders `/run` or `/tmp` result from doing a live analysis. These files will not be seen in a post-mortem forensic examination, and they can be ignored. On the other hand, notice that specific logs and the Gnome Tracker files have changed. Each of these candidate files is then examined in detail to determine if they contain evidence that a Transcend USB drive was inserted.

The advantage of a snapshot analysis is that it can be executed in a timely way. For example, this experiment was conducted in under an hour. The disadvantage is that there is usually a lag between when a file changes and when Ubuntu updates the file metadata. If the file metadata has not changed, the snapshot will not flag the file as altered.

Therefore, after all the snapshots experiments are completed, a forensic image is created. A forensic tool such as AD Lab or X-Ways is used to find artifacts through string searches. For example, in the USB drive insertion experiment, the forensic analyst would conduct string searches for "Transcend" and "Sandisk." This analysis will identify additional artifacts that the snapshots may have missed.