ICT3215 Digital Forensics

U-See Bus
User Manual

PA-G 19 GonezCase
Heng Seng Hong, Royston
Bryan Toh Kai Jie
Bryce Tam Yik Man
Quah Yong Yao
Tan You Aa Michael

2301799@sit.singaporetech.edu.sg, 2103300@sit.singaporetech.edu.sg,
2301800@sit.singaporetech.edu.sg, 2302009@sit.singaporetech.edu.sg,
2302145@sit.singaporetech.edu.sg

Table of Contents

Introduction

U-See Bus is a Python-based Windows Tool that detects the insertion and removal of USB thumb drives and monitors changes made to files in selected folders. By default, the tool monitors the Downloads, Documents, and Desktop folders. It also records any executable files that are run from a USB drive. All activity is logged using cryptographic chain hashing so that any tampering with the logs can be detected. The tool produces a session log that records all events and a final digest that can be used to verify the integrity of the session log.

System Requirements

U-See Bus requires the following to function normally.

Operating System Requirements

- Windows 11
- Windows 10
    - WMIC Commands are only available on Windows.

Python Requirements

- Python version 3.8 or newer
- Pip installation manager

Github Requirements

- Git for Windows

Python Packages & Libraries

- psutil
- watchdog
- PyInstaller

Miscellaneous Requirements

- At least one existing USB port on your local device
- At least one USB drive
- Access to folders being monitored by U-See Bus
- System permission to execute Python scripts

Installation

Section Overview

This section covers how to install U-See Bus on your local device. This includes dependencies and set up of a virtual environment to freely test U-See Bus on your local device.

Github Repository

Open a Command Prompt/PowerShell session either through Terminal or the built-in terminal within Visual Studio Code and run the following command.

```
git clone https://github.com/y2git/ICT3215-U-See_Bus.git
```

Once the repository has been downloaded, pivot into the repository folder.

```
cd .\ICT3215-U-See_Bus\
```

Virtual Environment Creation

Run the command inside the project folder to create a virtual environment.

```
python -m venv .venv
```

Once the virtual environment has been created,

Installation of all required Python packages & libraries

Install all required Python packages and libraries inside requirements.txt using the following command to be used within a terminal.

```
pip install -r requirements.txt
```

Usage

Default Monitoring

Run the following command within your terminal to start U-See Bus to monitor default configured folders.

```
python main.py
```

Monitor Specific Folders

Run the following command within your terminal to start U-See Bus to monitor specific folders that you wish to focus on. Use the “--paths” argument followed by the file path using closed inverted commas.

```
python main.py --paths "C:\Users\<User>\Desktop" "C:\Evidence"
```

Disable USB Monitoring

To disable USB monitoring upon execution U-See Bus, run the following command within your terminal.

```
python main.py --no-usb-monitor
```

Custom Output Directory

Run the following command within your terminal to specify directories to store the outputted reports. The first argument is the folder that will act as the parent folder of the following folders, the second argument will store the USB_session_timestamp.json report file and the third argument will store the final_digest_timestamp.json report file.

```
python main.py --outdir <base_directory> <session_directory> <digest_directory>
```

The following command is an example of how to use the command.

```
python main.py --outdir "usb-reports" "usb-sessions" "final-
digest"
```

Custom USB Directory Mounting

If you have several USB devices connected to your local device, you can use the following command within your terminal to specify which USB device to monitor.

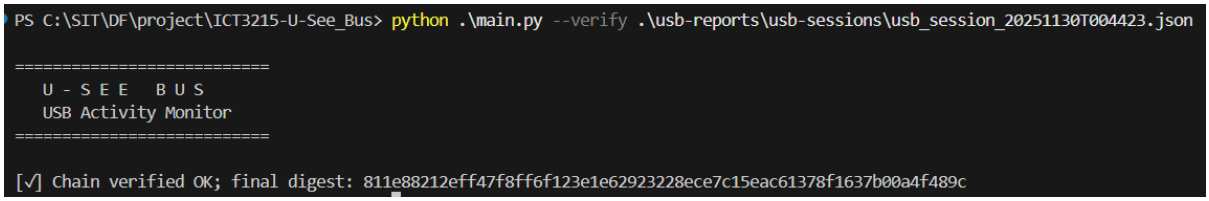
```
python main.py --usb-mount "E:\\"
```

Verifying Existing Session Or Digest File Hash

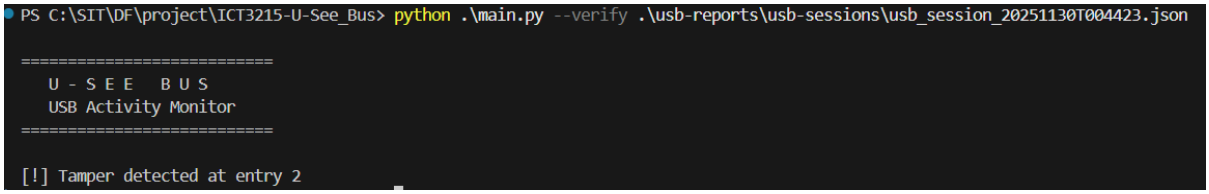
To verify if an existing session report has been tampered with or if it is still in its Golden Copy state, execute the following command within your terminal.

```
python main.py --verify path/to/usb_session_<timestamp>.json
```

If there are no modifications to the session file, the output will look like this:



If the session log was modified which resulted in a change in the hash chain, the output will look like this:

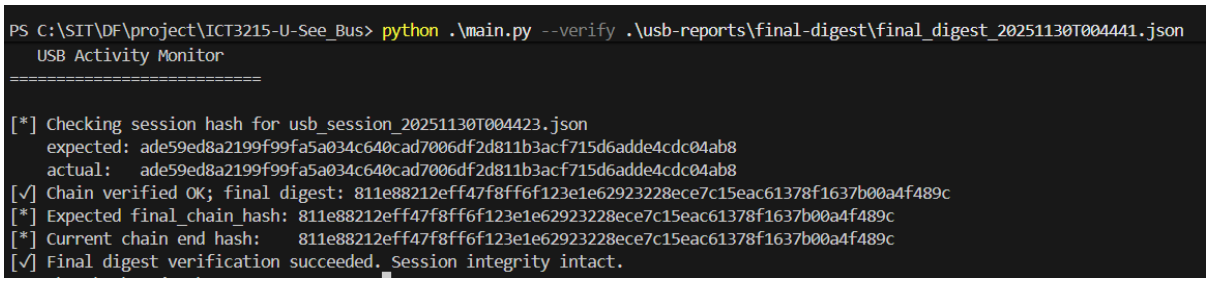


In this case, the tampering occurred at the 3rd chain entry (count starts from 0).

To verify if an existing digest report has been tampered with or if it is still in its Golden Copy state, execute the following command within your terminal.

```
python main.py --verify path/to/final_digest_<timestamp>.json
```

If there are no modifications to the session file or final digest file, the output will look like this:



If the session log was modified which resulted in a change in the hash chain, the output will look like this:

```
PS C:\SIT\DF\project\ICT3215-U-See_Bus> python .\main.py --verify .\usb-reports\final-digest\final_digest_20251130T004441.json

 ==========================
   U - S E E   B U S
   USB Activity Monitor
 ==========================

 [*] Checking session hash for usb_session_20251130T004423.json
     expected: 7e8dc722455483b92e755d472e32ec765149a4acb6e84cd5daa250409b82ae6d
     actual:   ade59ed8a2199f99fa5a034c640cad7006df2d811b3acf715d6adde4cdc04ab8
 [!] Session file hash mismatch — possible tampering!
```

If the hash related values of the final digest were modified, the output will look like this:

```
PS C:\SIT\DF\project\ICT3215-U-See_Bus> python .\main.py --verify .\usb-reports\final-digest\final_digest_20251130T004441.json

 ==========================
   U - S E E   B U S
   USB Activity Monitor
 ==========================

 [*] Checking session hash for usb_session_20251130T004423.json
     expected: ade59ed8a2199f99fa5a034c640cad7006df2d811b3acf715d6adde4cdc04ab8
     actual:   ade59ed8a2199f99fa5a034c640cad7006df2d811b3acf715d6adde4cdc04ab8
 [√] Chain verified OK; final digest: 811e88212eff47f8ff6f123e1e62923228ece7c15eac61378f1637b00a4f489c
 [*] Expected final_chain_hash: 811e88212eff47f8ff6f123e1e62923228ece7c15eac61378f1637b00a4f489c
 [*] Current chain end hash:    811e88212eff47f8ff6f123e1e62923228ece7c15eac61378f1637b00a4f489d
 [!] Chain hash mismatch — session altered after digest creation!
```

Runtime Freeze Anti-Tampering Validation

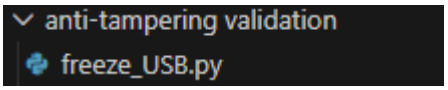
Section Overview

This section covers how to perform validation of U-See Bus's Runtime Freeze Anti-Tampering.

Firstly, run U-See Bus as per normal using the following command.

```
python main.py
```

Secondly, run the freeze_USB.py script provided in the anti-tampering validation folder. It is recommended to run this script in a standalone PowerShell or Command Prompt session.



Upon execution of the script, it should begin suspending U-See Bus for 6 seconds to simulate live tampering of U-See Bus's execution.

After 6 seconds, U-See Bus will print the following response.

```
 ==========================
   U - S E E   B U S
   USB Activity Monitor
 ==========================

 U-See Bus is executed. Press Ctrl+C to stop.
 [*] Watching local path: C:\Users\bryce\Downloads
 [*] Watching local path: C:\Users\bryce\Documents
 [*] Watching local path: C:\Users\bryce\OneDrive\Desktop
 [!] Ignored NON-USB device at D:\ (not removable USB thumbdrive or is external storage or is internal storage)
 [!] Runtime freeze detected — stopping all threads immediately
 [√] Forced digest written: usb-reports\final-digest\forced_digest_20251130T151148.json
 [√] Session frozen at tamper moment.
```

If the output in the screenshot above is observed, then the Runtime Freeze detection feature is working as intended.

Termination Tampering Detection with Marker

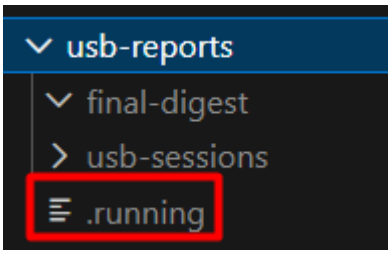
Section Overview

This section covers how U-See Bus uses a marker to detect any tampering with it's termination.

Firstly, run U-See Bus as per normal using the following command.

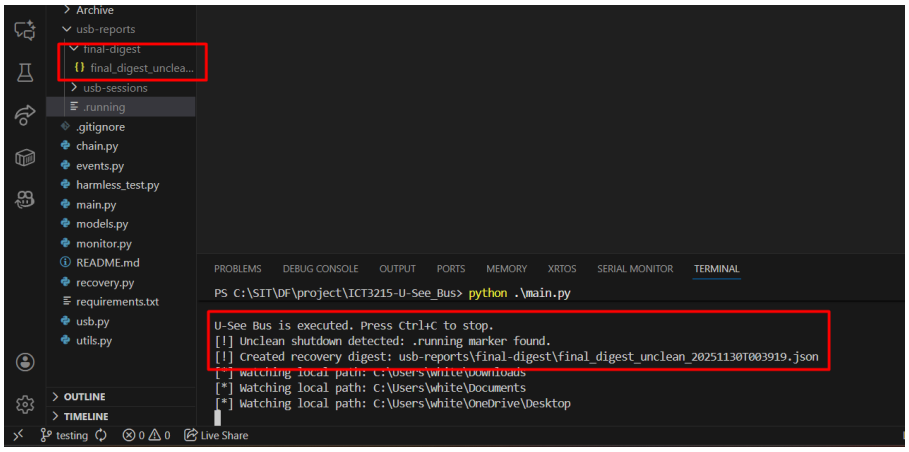
```
python main.py
```

When U-See Bus is executed, a ".running" file will be generated in the directory that contains the folders for both the final digest and the USB session logs.



When U-See Bus is terminated normally with CTRL+C, it will delete the ".running" file. However, if U-See Bus is forcefully terminated such as if the computer is forced shut down, the ".running" file will remain in the folder.

If this happens, and if U-See Bus is executed once more, a different final digest file will be generated along with a message indicating that the ".running" file was not removed from the last termination, implying that U-See Bus stopped running unnaturally.



This will indicate to the analyst or investigators that there might be an issue with the computer, and they should investigate it if something suspicious had recently occurred.