BitCuratorEdu Learning Object

Using the BitCurator Reporting Tool

Discussion questions to pair with the screencast

Authors

Cal Lee, Hannah Wang

Description

These discussion questions can be used to encourage student engagement with the BitCurator screencast, Using the BitCurator Reporting Tool: Part 1. The questions can also be used for discussion accompanying a live demonstration, a guided hands-on exercise, or independent exploration of the BitCurator Environment.

Learning object type

Lesson plan/materials

Learning objectives

This learning object might be used in a lesson to satisfy the following learning objectives:

BitCuratorEdu Learning Object

- Identify the appropriate tools to: safely acquire born-digital materials from storage media and other modes of transfer; assist in the appraisal of born-digital materials; scan for sensitive information in born-digital materials; and package born-digital materials for preservation and access.

Screencast

https://youtu.be/6A4iDWzzEWo

Discussion Questions

These discussion questions can be used to encourage student engagement with the BitCurator screencast linked above. The questions can also be used for discussion accompanying a live demonstration, a guided hands-on exercise, or independent exploration of the BitCurator Environment. Video timestamps are included in parentheses, where applicable.

For each potentially sensitive string that Bulk Extractor finds (called a "feature"), it generates a line of text in one of its reports with the following structure: offset, feature, context. If you look at the window that contains the yellow highlighted text in BeViewer (see 3:09-3:23), you can see specific offsets and features.

[For the questions below, see also the "How bulk_extractor Works" section of the Bulk Extractor User Manual and Quickstart Guide, available in the BitCurator environment in the Documentation and Help folder on the desktop and at https://github.com/simsong/bulk_extractor/wiki/Documentation.]

1. Bulk Extractor can run several different scanners. What is a scanner?
2. What is an offset, and what does it signify within a Bulk Extractor report?
3. Why does Bulk Extractor report two different offsets when it finds a feature in an encoded/compressed file?
4. What are the advantages and disadvantages of reporting features based on their offsets?
5. What else might you want to know about the location of the feature?
6. How might you determine where the feature appears in the filesystem?

Answer Key

These discussion questions can be used to encourage student engagement with the BitCurator screencast linked above. The questions can also be used for discussion accompanying a live demonstration, a guided hands-on exercise, or independent exploration of the BitCurator Environment. The questions **(in bold text)** *ask students to analyze the social and technological context of the BitCurator Environment and the tools packaged in the distribution. Example answers are given* (in regular text)*, though some questions are subjective and answers may vary, depending on the knowledge of the student and the scope of the class. Video timestamps are included in parentheses, where applicable.*

For each potentially sensitive string that Bulk Extractor finds (called a "feature"), it generates a line of text in one of its reports with the following structure: offset, feature, context. If you look at the window that contains the yellow highlighted text in BeViewer (see 3:09-3:23), you can see specific offsets and features.

[For the questions below, see also the "How bulk_extractor Works" section of the Bulk Extractor User Manual and Quickstart Guide, available in the BitCurator environment in the Documentation and Help folder on the desktop and at https://github.com/simsong/bulk_extractor/wiki/Documentation.]

Bulk Extractor can run several different scanners. What is a scanner?
Bulk Extractor runs a set of individual scanners, each of which looks for a particular pattern or set of patterns (e.g., something that looks like an email address or a phone number), and then outputs the results into a simple text file (feature file). A feature file contains rows of text (one for each feature found), with each row containing the following, separated by tabs:

- path to the feature - an offset if Bulk Extractor scanned a disk or disk image, and a directory path if Bulk Extractor scanned a directory of files
- feature - the string that it found (e.g., an email address)
- context - the feature along with some of the bytes before and after it. The default value is 16 bytes before and 16 bytes after the feature, but you can

change this by providing a different value for “Use Context Window Size” when running Bulk Extractor Viewer.

Some scanners also output additional elements to the feature file (e.g. cryptographic hashes).

What is an offset, and what does it signify within a Bulk Extractor report?
An offset is a number indicating the distance (usually in bytes) between the beginning of a data structure and a given location within that data structure. For the purposes of digital curation, the important offsets are usually within specific bistreams, including disk images and individual files. To calculate an offset, software begins at the first byte of the bitstream, reads through the bitstream to a specific location, and then counts and reports the number of bytes in between the start of the bitstream and that location.

In Bulk Extractor viewer, if you select an individual feature, you’ll see the offset between the beginning of the disk (or disk image) and the location of the feature. You can also see the offset if you open one of the individual Bulk Extractor reports in a text editor.

If you look in the large window on the right of the BE Viewer interface, you can also see the feature within the context of the whole bitstream. Each line starts with the offset of the string shown on that line. (These do not represent separate lines within the bitstream itself, which is a continuous series of bits. The software just needs to wrap to a new line periodically so the text doesn’t run off the screen to the right.)

When Bulk Extractor finds a feature within an encoded or compressed file (e.g. PDF, ZIP, DOCX), it shows two offsets: one to the beginning of the file and one to the offset of the feature within that encoded/compressed file. For example, the feature offset “11052168704 - GZIP -3437” starts with the number of bytes between the beginning of the disk or disk image and the start of a GZIP file, followed by “ - GZIP - “ and then the offset between the beginning of the GZIP file and the feature that appears within that GZIP file.

Why does Bulk Extractor report two different offsets when it finds a feature in an encoded/compressed file?

Bulk Extractor's default process is to read through a disk or disk image as one long bitstream without paying any attention to files or folders. This allows it to operate very quickly and efficiently, because it doesn't need to make lots of calls through the filesystem (finding the boundaries of a given file and then opening it). It also allows Bulk Extractor to find features on disks or disk images that have unknown or corrupted filesystems. However, specially encoded and compressed files require additional processing before Bulk Extractor can read the text within them. Reporting the two different offsets allows Bulk Extractor and other software to find the relevant file and then also find the feature within the individual file. It's important to note that Bulk Extractor isn't a universal file parser. There are many file types that it does not pre-process before scanning for features; for those types of files, it will only find features if they appear within the disk or disk image as plain (ASCII or Unicode) text.

What are the advantages and disadvantages of reporting features based on their offsets?

This allows the scanners to run very quickly, because the software progresses through the raw data byte-by-byte rather than having to work through the filesystem. It also makes it fast and easy to jump to the spot on the disk (or disk image) where the feature is located. A major disadvantage is that it doesn't tell the user where the feature appears in the filesystem (file or folder).

What else might you want to know about the location of the feature?

Location within the filesystem (file or folder), including whether it's in an allocated or unallocated (deleted) part of the disk.

How might you determine where the feature appears in the filesystem?

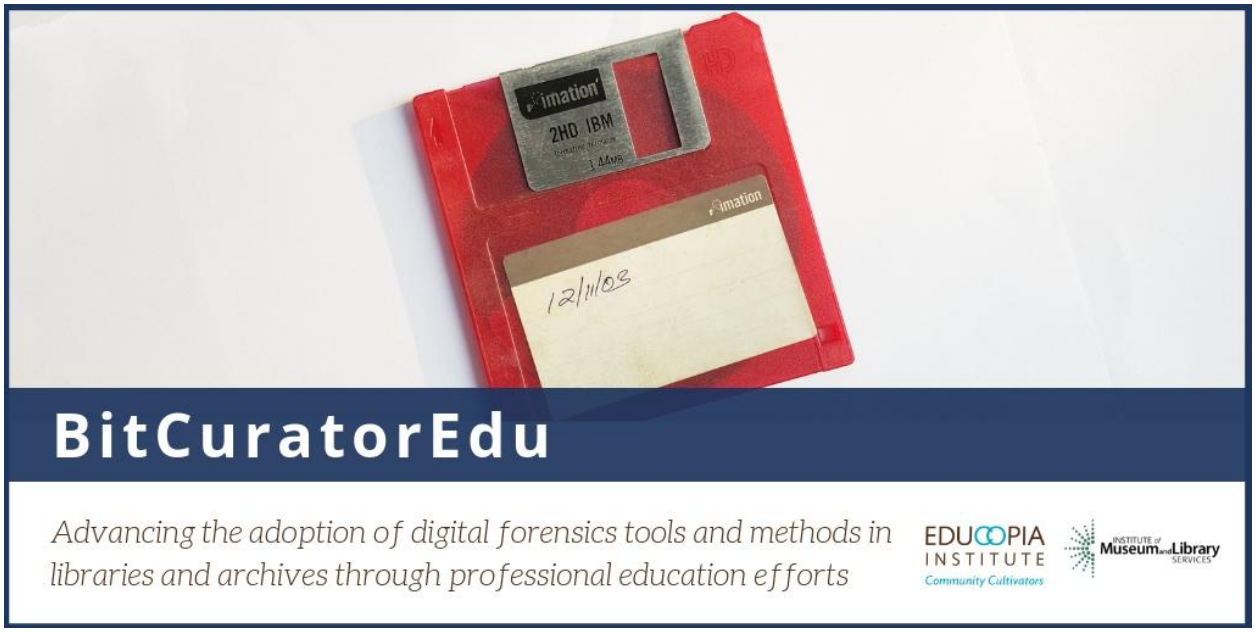
Fiwalk outputs information from the filesystem. Its digital forensics XML (DFXML) output includes "byte runs," which indicate the specific offsets where a file begins and ends. (A fragmented file will have multiple byte runs.) The default behavior of the BitCurator reporting tool is to run fiwalk (generating DFXML and then running

some Python code that determines where each Bulk Extractor features is located within the filesystem by calculating whether the feature falls within the byte run of a specific file or folder.

Tools and Resources Mentioned in This Document

Bulk Extractor documentation:
https://github.com/simsong/bulk_extractor/wiki/Documentation

BitCuratorEdu Learning Object



Most resources from the BitCuratorEdu project are intentionally left with basic formatting and without project branding. We encourage educators, practitioners, and students to adapt these materials as much as needed and share them widely.

The BitCuratorEdu project is a three-year effort (2018-2021) funded by the Institute of Museum and Library Services (IMLS) to study and advance the adoption of digital forensics tools and methods in libraries and archives through professional education efforts. This project is a partnership between Educopia Institute and the School of Information and Library Science at the University of North Carolina at Chapel Hill, along with the Council of State Archivists (CoSA) and several Masters-level programs in library and information science.