BitCuratorEdu Learning Object

BitCurator Software: Using fiwalk to Generate Filesystem Metadata

Discussion questions to pair with the screencast

Authors 1

Description 1

Learning object type 1

Learning objectives 1

Screencast 2

Discussion Questions 3

Answer Key 6

Tools and Resources Mentioned in This Document 13

Authors

Cal Lee, Hannah Wang

Description

These discussion questions can be used to encourage student engagement with the BitCurator screencast, Using fiwalk to Generate Filesystem Metadata. 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

BitCuratorEdu Learning Object

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

- 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/TOIXkxFus9o

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.

1. At what point(s) in a digital curation workflow might you expect to use fiwalk?
2. What is Digital Forensics XML (DFXML), what types of information does it address, and why is it important for digital curation?
3. The video highlights several technical metadata elements in the DFXML (2:14-2:49). Are there specific archival principles supported by having these technical metadata elements?
4. Can you further describe these elements and why they could be important to retain?
    a. `<program>fiwalk</program>`
    b. `<version>4.6.1</version>`
    c. `<execution_environment>`
    d. `<command_line>fiwalk -f -X /home/bcadmin/Desktop/Image_1Fiwalkreports.xml /home/bcadmin/Desktop/Image_1/Image_1.E01</command_line>`
    e. `<start_time>2018-10-23T15:44:36Z</start_time>`
    f. `<source> <image_filename>/home/bcadmin/Desktop/Image_1.E01</image_filename>`
    g. `</source>`
    h. `<partition_offset>16384</partition_offset>`
        i. Can you provide an explanation for why the value of partition_offset is divisible by 4096?
    i. `<ftype_str>fat32</ftype_str>`

5. What other elements can you identify that appear before the first <fileobject> tag, and how might they serve as preservation or provenance metadata?
6. Most of the DFXML file is composed of <fileobject> elements (2:50-3:30). Are there specific archival principles supported by having the <fileobject> metadata?
7. Is it possible for a file to have an mtime (modified timestamp) that is earlier than the crtime (creation time)? If so, how would that be possible?
8. What sorts of actions can change the atime (accessed timestamp) of a file? What measures could you take to mediate the risk of overwriting the original atime?
9. What software generates the value of the <libmagic> element? How is this different from the output of DROID or JHOVE?
10. What is the meaning of the “data” value in the <libmagic> element?
11. Why wouldn’t you just rely on the file extension to determine filetype?
12. Why are there two different <hashdigest> elements?
13. What does the <byte_runs> element signify?
14. What could you conclude if fiwalk listed more than one byterun for a <fileobject>?
15. Where does the information in the DFXML come from?
16. The default fiwalk output from the BitCurator reporting tool is an XML file, but DFXML defines a set of elements and attributes that could also be represented in other ways, including JSON or fields in a relational database. What is XML and what are some of its primary uses?
17. When processing a large storage device that includes a large number of files, the XML output from fiwalk can be quite large. Why might your machine hang or crash when trying to open the XML file with the default application?

18. Do you know of other software that you could use to better view the contents of large XML files?

19. If most of the DFXML data is extracted from the filesystem, why would you also want to store it in a separate place (XML output file, JSON output file, as fields in a relational database)?

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.*

At what point(s) in a digital curation workflow might you expect to use fiwalk?
Students could provide a variety of answers. In addition to some technical and provenance metadata, the majority of fiwalk's output is "file object" elements that help to document attributes of files and directories from the filesystem. Many of these attributes (especially timestamps) can be easily changed by software or user activities, and exporting the metadata as DFXML ensures that the original values are retained.

What is Digital Forensics XML (DFXML), what types of information does it address, and why is it important for digital curation?
First introduced by Simson Garfinkel and his research and development team, DFXML is a set of metadata conventions for output from digital forensics tools. (For further explanation, see Garfinkel, S. "Digital Forensics XML and the DFXML Toolset.") From an archival perspective, DFXML is a rich source of both provenance and original order metadata. It has been adopted by the developers of a variety of open-source tools, many of which you can run in the BitCurator environment, including fiwalk (after you run the BitCurator reporting tool, see the file called fiwalk-output.xml, other reporting scripts, and HFS2DFXML. DFXML is not currently a formal standard, but the National Institute for Standards and Technology (NIST) oversees a working group that maintains the current version of the DFXML schema.

While the schema is based on XML output, one can export the same metadata elements in other forms, including:

- Microsoft Excel (generated by default by the BitCurator reporting tool)
- Attribute-Relation File Format (ARFF)
- Comma-separated values (CSV)
- Written into a relational database for more efficient querying and processing as with the Sleuth Kit, Autopsy, and Brunnhilde

Archivematica, a popular open-source digital preservation system, also uses fiwalk to generate DFXML.

The video highlights several technical metadata elements in the DFXML (2:14-2:49). Are there specific archival principles supported by having these technical metadata elements?
While students might have additional answers, they should talk about the principles of provenance and chain of custody. The technical metadata indicates what tools and commands were used to generate the output. Note that the PREMIS output from several of the tools in the BitCurator environment also helps to advance the principles of provenance and original order.

Can you further describe these elements and why they could be important to retain?

`<program>fiwalk</program>`
The name of the XML-generating program. In this case, it’s fiwalk, a core element of the BitCurator reporting tool.

`<version>4.6.1</version>`
The specific version of the XML-generating program.

`<execution_environment>`
This element can include various details about the execution environment used to generate the XML file.

```
<command_line>fiwalk -f -X
/home/bcadmin/Desktop/Image_1Fiwalkreports.xml
/home/bcadmin/Desktop/Image_1/Image_1.E01</command_line>
```

This shows the specific fiwalk command that generated the XML file. The BitCurator reporting tool runs this command automatically, but you would get the same results if you typed this command in the terminal (Linux command-line interface). If you're not very familiar with the command line, seeing this text can be a helpful way to learn the different elements of Linux/Unix commands. It first says what program to run (fiwalk), then includes some switches that set parameters for how the program should run.

- The "-f" switch tells fiwalk to report the output of the 'file' command (https://en.wikipedia.org/wiki/File_(command)) for each file.
- The "-X" switch tells fiwalk to output XML to the location indicated: /home/bcadmin/Desktop/Image_1Fiwalkreports.xml.
- Finally, the command indicates what file should be used as input, which in this case is a disk image: /home/bcadmin/Desktop/Image_1/Image_1.E01.

```
<start_time>2018-10-23T15:44:36Z</start_time>
```

This is the time when the command started. The timestamp follows the form of YYYY-MM-DD then "T" indicating "time" and then HH:MM:SS. The "Z" indicates that this is "Zulu" time, also known as Universal Time Coordinated (UTC) or Greenwich Mean Time (GMT). This conforms to ISO 8601 (https://en.wikipedia.org/wiki/ISO_8601).

```
<source>
<image_filename>/home/bcadmin/Desktop/Image_1.E01</image_fi
lename>
</source>
```

This is the name and location of the disk image that served as input to fiwalk.

```
<partition_offset>16384</partition_offset>
```

This is the offset (in bytes) of the partition from the beginning of the image file (see https://en.wikipedia.org/wiki/Disk_partitioning).

Can you provide an explanation for why the value of partition_offset is divisible by 4096?
4096 (4K) is a common sector size. This means the computer can't allocate storage in increments smaller than 4K.

`<ftype_str>fat32</ftype_str>`
The filesystem on the disk or disk image (FAT32 in this case).

What other elements can you identify that appear before the first <fileobject> tag, and how might they serve as preservation or provenance metadata?
Students can provide a variety of options. If you'd like to see an example of DFXML, you can run Bulk Extractor and the BitCurator reporting tool and then open the DFXML output file (by default: fiwalk-output.xml).

Most of the DFXML file is composed of <fileobject> elements (2:50-3:30). Are there specific archival principles supported by having the <fileobject> metadata?
Students could offer a variety of answers. Two principles to consider would be provenance and original order.

Is it possible for a file to have an mtime (modified timestamp) that is earlier than the crtime (creation time)? If so, how would that be possible?
The crtime reflects when the file was first written to the storage volume. This is not the same as when the file was first created. Someone could have modified the file and then copied it to this disk, in which case the crtime can be later than the mtime.

What sorts of actions can change the atime (accessed timestamp) of a file? What measures could you take to mediate the risk of overwriting the original atime?
The access time is when a user or software most recently accessed the file. This can include conscious actions by people but also background actions taken by software such as virus scans and backups.

What software generates the value of the <libmagic> element? How is this different from the output of DROID or JHOVE?
This output comes from running the utility in Linux called file - https://en.wikipedia.org/wiki/File_(command). This is a standard Unix/Linux command-line tool. It runs more quickly and can identify a larger number of file types than DROID or JHOVE, but doesn't afford as much verification or metadata output.

What is the meaning of the "data" value in the <libmagic> element?
This is essentially the "miscellaneous" category. According to the man page (https://man7.org/linux/man-pages/man1/file.1.html), "The type printed will usually contain one of the words text (the file contains only printing characters and a few common control characters and is probably safe to read on an ASCII terminal), executable (the file contains the result of compiling a program in a form understandable to some UNIX kernel or another), or data meaning anything else (data is usually "binary" or non-printable). Exceptions are well-known file formats (core files, tar archives) that are known to contain binary data."

Why wouldn't you just rely on the file extension to determine filetype?
File extensions can be a quick and convenient way to identify file types. For example, Windows operating systems rely heavily on file extensions to determine what application should be used to open a file when you double-click on it. However, there are several reasons why it can be problematic to assume the file extension is an accurate representation of a file's format. These include:

- Anyone who has permission to change the name of a file also has permission to change the file extension, so someone can switch it to something that no longer reflects that format. This was especially common before the introduction of long file names in Windows 95. Previous to 1995, Windows users (and anyone exchanging files with Windows users) could only use eight characters in the main file name and three characters in the extension (often referred to as the "8.3" limitation). So people would often jam parts of the file name into the extension. For example, changing "november.doc" to "november.rpt" might have made it more clear that the document was a report from November, but this also broke the connection between the file name and its format.

- Similarly, those with malicious intent can change a file extension in an effort to trick people or software into thinking that a virus is just a normal file. For example, someone could change the file name of malicious code written in javascript from "evilstuff.js" to "goodstuff.jpg." Digital curation workflows and systems should include other ways to verify the file format, so viruses aren't silently deposited into repositories or the computers used to process collections.
- There is no central authority for registering or approving new file extensions, so the same string of characters can designate many different types of files. Consider the numerous uses of extensions like "bar" (https://www.file-extensions.org/search/?searchstring=bar) or "rip" (https://www.file-extensions.org/search/?searchstring=rip).

Why are there two different <hashdigest> elements?
By default, fiwalk run through the BitCurator Reporting Tool will generate two different hashes (checksums) for each file. One is based on running the file's bitstream through the md5 algorithm, and the other is based on running the file's bitstream through the SHA1 algorithm.

What does the <byte_runs> element signify?
This indicates where the file was located on the storage medium – and thus also where it's located on a disk image generated from that storage medium – in terms of:

- Logical offset within the file (number of bytes from the start of the file),
- Physical offset within the original disk or disk image (number of bytes from the start of the disk), and
- Length (number of bytes).

What could you conclude if fiwalk listed more than one byterun for a <fileobject>?
This would mean the file is fragmented, i.e. parts of the file are stored in different parts of the disk. Most files are not fragmented, so most will have only one byterun.

Where does the information in the DFXML come from?
It's generated by fiwalk (https://confluence.educopia.org/display/BC/Fiwalk). Most of the information in the DFXML is copied directly out of the filesystem of the original storage medium. However, fiwalk also generates some additional information such as running the Linux tool called file (https://en.wikipedia.org/wiki/File_(command)) to identify file type and running checksum tools to generate the md5 and SHA1 hash of the file.

The default fiwalk output from the BitCurator reporting tool is an XML file, but DFXML defines a set of elements and attributes that could also be represented in other ways, including JSON or fields in a relational database. What is XML and what are some of its primary uses?
XML standard for Extensible Markup Language (https://www.w3.org/TR/1998/REC-xml-19980210.html), but this name is deceptive, because XML is not a markup language. Instead, XML lays out a set of syntax rules for how to create new markup languages. These markup languages – along with their associated schemas – define the elements that are allowed, how those elements relate to each other hierarchically, and what attributes can be identified for those elements.

When processing a large storage device that includes a large number of files, the XML output from fiwalk can be quite large. Why might your machine hang or crash when trying to open the XML file with the default application?
Many applications will try to parse the whole XML file and load it into memory. This works fine for the small files usually involved with tasks such as editing web content, but it is not the right approach to access larger files.

Do you know of other software that you could use to better view the contents of large XML files?
What you want instead to view the XML file is a text editor that progresses line by line and only loads into memory the parts that you need to see, e.g. Emacs (Windows, macOS, Linux), Notepad++ (Windows). You can also use command-line

tools that allow you to scroll up and down within the text, e.g. less (MacOS, Linux) and more (MacOS, Linux, Windows).

If most of the DFXML data is extracted from the filesystem, why would you also want to store it in a separate place (XML output file, JSON output file, as fields in a relational database)?
Students could provide many different answers to this, including but not limited to:

- Less dependence on having software that can read the original filesystem directly
- Efficiency of processing (searching, querying and manipulating the XML rather the original filesystem)
- Efficiency of transfer (keeping these metadata elements closer to where they're needed, rather than having to move around the whole bitstream of the disk image)
- One can use external characterizations to "bootstrap" effective use of bitstream content.
- Redundancy eliminates a single point of failure (in this case, corruption of filesystem information could make a file unfindable or unreadable).
- Change of XML metadata doesn't require a change to the bitstream

Tools and Resources Mentioned in This Document

Attribute-Relation File Format (ARFF): https://www.cs.waikato.ac.nz/ml/weka/arff.html

Autopsy: https://www.sleuthkit.org/autopsy/

Brunnhilde: https://github.com/tw4l/brunnhilde

DFXML Schema : https://github.com/dfxml-working-group/dfxml_schema

Disk partitioning: https://en.wikipedia.org/wiki/Disk_partitioning

Emacs: https://www.gnu.org/software/emacs/

File (command):

- https://en.wikipedia.org/wiki/File_(command)
- https://man7.org/linux/man-pages/man1/file.1.html

Fiwalk:

- https://forensicswiki.xyz/wiki/index.php?title=Fiwalk
- https://confluence.educopia.org/display/BC/Fiwalk

Forensic disk images | Documentation (Archivematica 1.13.2) | Archivematica: Open-source digital preservation system. (n.d.). Retrieved June 14, 2022, from https://www.archivematica.org/en/docs/archivematica-1.13/user-manual/transfer/forensic/

Garfinkel, S. (2012). *Digital forensics XML and the DFXML toolset.* https://calhoun.nps.edu/handle/10945/44286

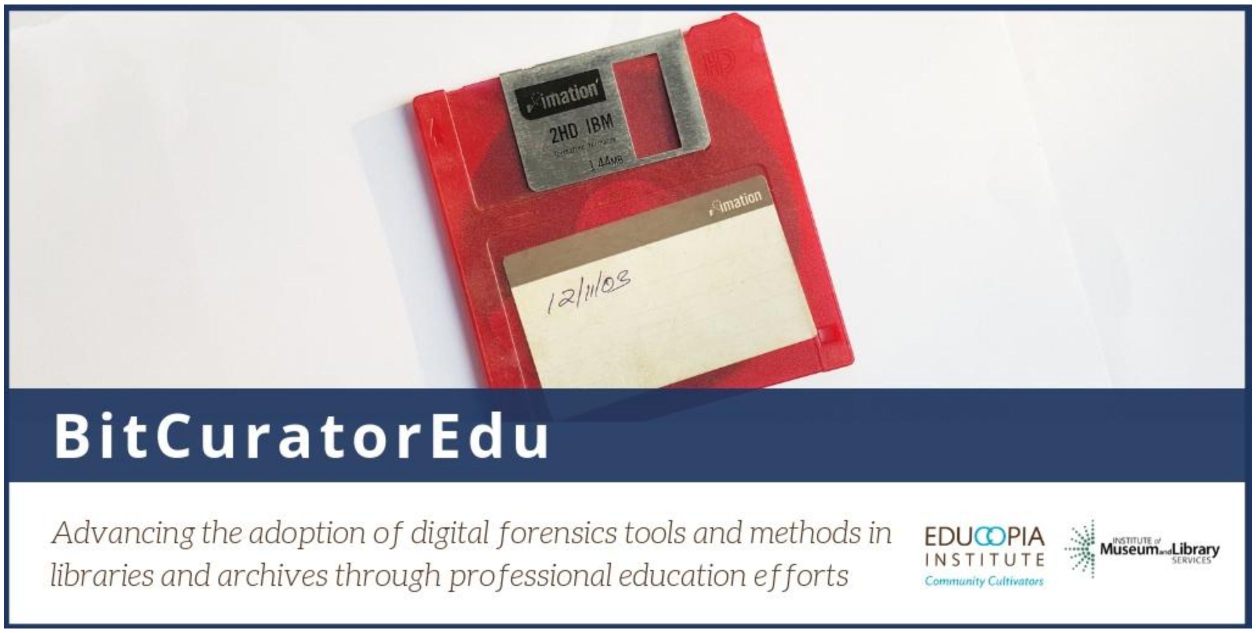
HFS2DFXML: https://github.com/cul-it/hfs2dfxml

ISO 8601: https://en.wikipedia.org/wiki/ISO_8601

Notepad++: https://notepad-plus-plus.org/

The Sleuth Kit: https://www.sleuthkit.org/

BitCuratorEdu Learning Object


BitCuratorEdu
Advancing the adoption of digital forensics tools and methods in libraries and archives through professional education efforts
EDUCOPIA INSTITUTE Community Cultivators
Institute of Museum and Library Services

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.