BitCurator: Requirements Document (version 0.9)

Project: BitCurator – Tools for Digital Forensics and Workflows in Real-World Collecting Institutions

Date(s): December 5, 2011 (v0.1); December 19, 2011 (v0.4); July 11, 2012 (v0.6); July 25, 2012 (v0.7); September 2, 2012 (v0.8); September 4, 2012 (v0.9)

Prepared by: Kam A. Woods and Christopher (Cal) Lee

Additional preparation: Alexandra Chassanoff

Document status: __ Draft __ Proposed _X_ Validated __ Approved

1. Introduction

This document contains the system requirements for BitCurator. These requirements are derived from several sources, including the final BitCurator grant proposal to the Andrew W. Mellon Foundation, notes prepared by the Technical Lead during initial planning of the project, information currently available at http://www.bitcurator.net/, and internal project documentation stored at the BitCurator project Basecamp site (hosted by MITH). Releases of this document following version 0.6 include information derived from current release and development materials documented at http://wiki.bitcurator.net/.

1.1 Purpose of This Document

This document is intended to guide development of BitCurator. It will move through several stages during the course of the project:

1. **Draft:** The first version, or draft version, is compiled after requirements have been discovered, recorded, classified, and prioritized.
2. **Proposed:** The draft document is then proposed as a potential requirements specification for the project. The proposed document should be reviewed by several parties, who may comment on any requirements and any priorities, either to agree, to disagree, or to identify missing requirements. Readers will include members of the project team (at both SILS and MITH), members of the Professional Expert Panel (PEP) and members of the Development Advisory Group (DAG). The document may be amended and re-proposed several times before moving to the next stage.
3. **Validated:** Once the various stakeholders have agreed to the requirements in the document, it is considered validated. Release 0.6 of this document is the first to include amendments following initial meetings with the PEP and DAG.
4. **Approved:** The validated document is accepted by representatives of each party of stakeholders as an appropriate statement of requirements for the project. The developers

then use the requirements document as a guide to implementation and to check the progress of the project as it develops.

1.2 How to Use This Document

We expect that this document will be used by a variety of groups with differing specializations and skillsets. This section explains which parts of this document should be reviewed by various types of readers.

Types of Readers

BitCurator Professional Experts Panel (PEP)

PEP readers should focus primarily on sections 1, 2, and (to a lesser extent) 4.

BitCurator Development Advisory Group (DAG)

DAG readers should focus primarily on sections 3 and 4.

BitCurator PI, Co-PI, Technical Lead, and Software Development Personnel

The BitCurator PI, Co-PI, technical lead and software development personnel will be familiar with all aspects of this document.

Technical Background Required

This document expects a broad general understanding of the issues and problems associated with handling digital collections, construction of digital archives, long-term digital preservation, and digital forensics.

A moderate technical background in operating system deployment and configuration is useful (but not required) for some 3.x subsections. Full technical specifications can be found in the supporting technical requirements documents concerning methods implemented in BitCurator for disk imaging, data forensics, and metadata acquisition and export.

Overview Sections

Section 1.3 includes broad use profiles for BitCurator. The entirety of section 2 provides a general, non-technical overview of BitCurator

Reader-Specific Sections

The document currently contains no reader-specific sections, with the exception of the previous notations regarding points of focus for PEP and DAG readers.

Section Order Dependencies

Sections in this document are organized linearly for new readers. Readers specifically interested in product use cases and the software technologies on which BitCurator depends may wish to first read sections 1.3, 3.1, and 3.3.

1.3 Scope

BitCurator is an effort to build, test, and analyze systems and software for incorporating digital forensics methods into the workflows of collecting institutions. We have designed it as a two-phase project, with the first phase occurring in years 1 and 2. The current funding from the Andrew W. Mellon Foundation is only for the first phase, but we provide some explanation in relevant places of the intended activities for the second phase (year 3). We believe that the professional engagement and outreach activities of this second phase will be essential to the uptake and sustainability of the systems and tools we build.

Phase one will focus on the following:

1. Identifying a core set of open source disk imaging, digital forensics, and metadata extraction and manipulation tools to be used in handling born-digital materials.

2. Providing a unified environment in which these tools can be used to assist in processing born-digital materials alongside existing collections management systems.

3. Extending these tools with software developed using existing APIs and scripting interfaces to improve their accessibility and utility to collections management professionals.

4. Identifying common gaps in existing systems and workflows designed to process digital collections, and isolating specific cases in which technologies and methods used by BitCurator can support these systems and practices.

We believe that these areas of focus will assist collecting institutions both in improving handling of existing born-digital collections and in addressing issues we expect they will encounter as the size and complexity of individual items entering collections increases in the future.

1.4 Research and Development Case for BitCurator

While some digital forensics tools have natural applications or parallels in digital curation, they are currently not very approachable to library and archives professionals in terms of interface, operation, and documentation. Furthermore, some of the functionality required by collecting institutions is incomplete or absent from software designed for the forensics industry:

- Incorporation into the workflow of born-digital material ingest and collection management. Foremost among the outstanding issues are support for library and archival metadata conventions, and integration with existing content management and digital preservation environments. Addressing these issues may include bridging open source tools through existing APIs, export scripts (as standalone programs or modules for existing software) to crosswalk forensics and archival metadata, and modifying triage techniques to better meet the needs of archivists and librarians. Additionally, the creation of robust and user-friendly documentation is a critical aspect of tool adoption and deployment.

- Provision of public access to the data. Typical digital forensics scenarios involve criminal investigations in which the public never gets access to the evidence that was seized. In contrast, collecting institutions that are creating disk images face issues of how to provide access to the data. This includes not only interface issues, but also how to redact or restrict access to components of the disk image, based on confidentiality, intellectual property or other sensitivities. Some of the technical mechanisms available in existing forensics software – annotations, case documentation, and bookmarking of byte offsets into raw data sources – can be repurposed and repackaged to assist in providing such access.

The project has direct bearing on ongoing activities at UNC Chapel Hill and MITH, both of whom have active digital archiving programs, along with information professionals who have assisted in describing specific use cases that the BitCurator software is designed to address.

1.5 Overview of the Requirements Document

Section 1: Purpose, background, and scope of BitCurator

Section 2: General description of the project, functions, user criteria, and high-level dependencies

Section 3: User and reporting requirements, integration with existing systems and software, and security.

Section 4: High-level overview of BitCurator software and related technologies

Section 5: User support

Section 6: Appendices

Section 7: Glossary

Section 8: References

Section 9: Index

2. General Description

This document explicates the needs and requirements for BitCurator. BitCurator will define and test support for a digital curation workflow that begins at the point of encountering holdings that reside on fixed and removable media—either new acquisitions or materials that are within a repository's existing holdings—and extends to the point of interaction with an end user. BitCurator will address both client-side tools required at the point of initial data extraction and back-end tools for batch processing of disk images, which are likely to reside on a remote server. BitCurator will not address the issue of retention and analysis of materials created or stored in cloud services, unless those materials are also retained on or leave traces of activity on local media.

The requirements in this document will be used to develop prototype systems for managing disk images as objects within digital collections. This will include taking advantage of existing high-performance open source disk imaging and analysis software (including but not limited to the Advanced Forensic Format Library) using existing APIs, the development of an automated reporting tool that can be used by collecting institutions to rapidly assess the contents and use history of a disk, metadata crosswalk modules, and a live distribution based on Ubuntu.

BitCurator focuses specifically on disk images because they provide complete packaging for all contextual and technical information that may be required for future access and preservation needs. Deep inspection of these images using automated analytics tools can assist collecting institutions in making well-informed decisions about which information should be preserved (at ingest) should they choose not to retain the entire disk image.

Some of the functionality supported by the BitCurator software will require access to third-party open source software packages. We will document these dependencies and provide mechanisms for incorporating existing software. We will make all software developed as part of the BitCurator project available as source code and (when applicable) as precompiled binaries via a freely accessible online repository. We will provide an integrated virtual machine environment (as a virtual disk image), and an installable environment (as an ISO image) based on a long-term service version of a modern Linux environment (Ubuntu 12.04) that includes both in-house and third-party software. These environments will be preconfigured for ease of use and test deployment in existing production environments. Additionally, we will provide a metapackage for installation of the software on existing Ubuntu and Debian machines.

2.1 Product Perspective

Libraries, archives and museums (LAMs) are increasingly called upon to move born-digital materials that are stored on removable media into more sustainable preservation environments. This can involve media that are already in their holdings (e.g. disks stored in boxes along with paper materials), as well as materials that they are acquiring for the first time from individual donors or other producers.

Procedures and tools for acquiring and validating data from physical media are well established in the field of digital forensics, but their recognition and adoption within LAMs is still quite limited. There are currently a handful of innovative and promising projects to investigate the application and implications of digital forensics approaches for LAM acquisitions. Notably absent are software packages that are designed to support this class of users.

The software and expertise provided by BitCurator will attempt to fill this gap, providing these institutions with a simplified means of using powerful digital forensics tools through the extension of existing software packages and development of new software tools to facilitate digital curation activities.

2.2 Product Functions

The BitCurator architecture will support the following functions:

1) Creation of disk images from storage media. At a minimum, this will include raw hard disk images, ISO and raw images from removable optical media, floppy disk images (FDI and others), and AFF packaging for these images. BitCurator will support analysis of additional proprietary formats – e.g. Apple Disk Image format (.dmg) – when this is practical and feasible. Additionally, while we will provide some written documentation on the use of specialized hardware for disk image acquisition.

2) Association of additional metadata with the raw disk image (either automated during creation of the Advanced Forensic Format image file or in an associated metadata file). This will include provenance metadata that is generated by the forensic software (detailing the time, date, and procedures used for the capture and analytics processes), and technical metadata about the disk image, capture process, and image contents, and further information including access and distribution rights. The metadata will be machine actionable: output as XML (specifically Digital Forensics XML), with facilities to crosswalk to archival and library metadata standards. We discuss this in further detail in a later section.

3) Generation of reports that summarize and characterize the contents of a given disk or disk image. Depending on the use case, these will take the form of machine-readable XML, human-readable PDF and text reports, and visualizations that identify particular statistics of interest (for example, graphs of file format distribution within an image). These reports may include (but are not limited to):

   a) XML or plain-text summary of all file tree elements (files and directories) and associated filesystem attributes. Histograms of most frequent or customized sets of these values.

   b) Number of files of given types (with versions and format compliance when applicable)

   c) Potentially sensitive data contained within the disk image. These reports will include automated output from the flexible lexical analyzer modules in Bulk Extractor. Provisions for customized reports based on regular expression search, addition of further PII modules.

   d) User accounts associated with particular volumes on a drive -- extracted from system files (Windows Registry) or other sources.

   e) Software installed on a particular volume and log files of software installation / uninstallation events.

   f) Date ranges of file modifications, accesses and changes (based on filesystem attributes)

   g) File fragments

   h) Deleted files

i) Text documents that are stored as page images but have not be subject to optical character recognition (OCR)

j) Metadata associated with programs and equipment used for document production.

k) Hashes and fixity information found on disk.

l) README files

4) Synthesis of reports and intermediate formulations from raw metadata

5) Incorporation of the products of both functions 1 and 2 (disk images and associated metadata) into the following:

a) Existing LAM collection management systems. Collection and mapping of workflow documentation from institutions currently handling disk images to a 'master workflow' which may be used to identify relevant applications of core BitCurator technologies. This will involve multiple stages:

i) Collect notes and automated output associated with image production and analysis from external institutions.

ii) Document best practices associated with specific hardware, image types, and metadata collection based on internal trials conducted with data supplied by partners, and collected by teams at UNC Chapel Hill and MITH.

b) Existing LAM metadata schemas, including but not limited to:

i) Encoded Archival Description (EAD) Finding Aids

ii) Metadata Encoding and Transmission Standard (METS)

iii) Metadata Object and Description Schema (MODS). Transcription and export of bibliographic metadata originally encoded in DFXML (Dublin Core tags).

6) Flagging portions of a disk image that are subject to redaction

7) Redaction of portions of a disk image, either through over-writing of the data on a copy of the disk image, or use of "permission overlays" that indicate which portions cannot be accessed. Permanent or static redaction of a disk image copy is the simplest case, and reflects the application of scripts that identify private and/or sensitive content at the block level (that is, below the level of individual files), overwriting only those portions of the data that are identified as items of interest by forensic analytic techniques. Depending on the approach, this type of redaction can cause disk images to be seen as damaged by the host system or result in apparent corruption at the file level. Users may elect, as an alternate strategy, to redact or remove entire files or directory trees (based on a mapping from PII items identified to file system-level objects). Finally, for disk images being provided for public access, redaction profiles may be constructed dynamically based on derived mappings of personally identifying information (PII) in individual blocks to filesystem elements, creating permissions overlays that prevent the virtual host from allowing read access. Provision of both end user and archivist access to forensic data analysis (data generated by functions 1 and 2) with incorporation of appropriate restrictions encoded complementary to the existing metadata

(e.g., for a series in the EAD), closed until a given date. Access will include (but not be limited to) private and sensitive information extracted from raw data sources, timestamps and document production timelines, document differencing information (and sub-document-level hashes), and statistical derivations from raw data including file type histograms, file distribution on disk, and fixed and external device use.

2.3 User Characteristics

The primary users of BitCurator software products will be staff working in LAMs, who are responsible for the collections that are (partially or completely) composed of born-digital materials obtained on physical media such as complete computers, hard drives removed from acquired systems, and optical, magnetic, and solid-state removable media. We anticipate that these users will be technically adept but not necessarily familiar with traditional digital forensics software. By "technically adept" we expect that users will be:

1) Capable of employing basic hardware skills to attach external removable media to a host device (including forensic write-blockers, external media readers) and familiarity with common device types and connectors.

2) Familiar with installing pre-packaged software in a Windows or Linux environment (and have access to an administrative account on the host machine).

3) Familiar with steps required to disable automated mounting of and access to removable media on a host system.

The BitCurator project will provide documentation and instructions to assist users in performing all of the above tasks.

Note that users wishing to access lower-level features of BitCurator (for example, certain forms of batch processing or integration with existing software such as The Sleuth Kit) will require familiarity with the UNIX command-line and plugin systems on which BitCurator depends.

2.4 General Constraints

The need for rapid prototyping development and the small size of the development team for BitCurator dictate that certain platforms be prioritized, particularly Linux (to support the final bootable environment product) and Windows (for integration with existing systems). Much of the software that BitCurator is likely to depend upon can already be cross-compiled for each of these platforms.

2.5 Assumptions and Dependencies

2.5.1 Delivery

BitCurator will be delivered to relevant parties via a download portal linked from BitCurator's main site, http://www.bitcurator.net/. This portal, implemented as a Wiki at http://wiki.bitcurator.net/, will provide public access to project deliverables, documentation, and community engagement materials.

2.5.2 Technical Infrastructure

BitCurator will provide tiered access to various levels of imaging and forensic data analysis functionalities. In the simplest use case, users who cannot purchase or repurpose dedicated workstation hardware will be able to download a virtual environment in which they can explore BitCurator functionality in a virtual machine on an existing host using free, open source virtualization solutions such as Oracle's VirtualBox. The BitCurator environment will be preconfigured to support "safe" mounting of external media and software write-blocking, in recognition of the fact that many institutions may require proof-of-concept trials with these techniques prior to committing to purchasing dedicated hardware solutions.

Full functionality with the BitCurator tools and services will depend on users having access to dedicated workstation hardware with administrative-level control. Software included in and developed for BitCurator will depend upon access to specialized (but relatively inexpensive) hardware, including commercial forensic write-blockers, removable media read/write hardware, and potentially (for deep data recovery) hardware designed to read magnetic flux transitions from removable magnetic media.

BitCurator will also support and reference – as necessary – specific hardware such as DeviceSide's FC5025, Red Fox Engineering's DiscFerret, and the Kryoflux floppy acquisition device. BitCurator will provide this support in acknowledgement that legacy removable media (e.g. 3.5" and 5.25" floppies, CD-ROMs) remain a significant concern – both in volume and in terms of the technical strategies employed to read and recover them – but will emphasize tools and workflows that apply to more modern media as well.

3. Specific Requirements

This section of the document lists specific requirements for BitCurator. Requirements are divided into the following sections:

1. User requirements. This document contains user requirements primarily in narrative form. Technical detail is provided in the supporting documents for specific BitCurator modules.

2. Reporting requirements.

3. System and Integration requirements. These are detailed specifications describing the functions the system must be capable of performing.

4. Security Requirements

5. User Interface requirements. This document provides a general narrative concerning user interface requirements; ongoing interface development based on user feedback may be found in supporting technical documents and on the BitCurator wiki.

3.1 User Requirements

Users will be able to obtain the BitCurator software as source code or (for the Windows platform) as precompiled executable programs. In year two of the project, users will also have one of the following options:

1. Downloading a package file that can be installed on a Ubuntu or Debian-based workstation

2. Downloading a custom Ubuntu image as an ISO, prepared with all of the relevant BitCurator software, which can be burned to optical media or written to a bootable USB device.

Users wishing to access lower-level features of BitCurator (for example, certain forms of batch processing or integration with existing software such as The Sleuth Kit) will require familiarity with the UNIX command-line and plugin systems on which BitCurator depends.

Primary functionality of the software will be provided through a custom graphic user interface (GUI). Simplified, wizard-style access to *specific* services within BitCurator will be available, particularly for dealers, donors, and other external parties who do not work directly within collecting institutions. Depending on the functionality of the underlying code, separate GUIs may be used for imaging of media and analysis of the images produced.

3.2 Reporting Requirements

The project staff will provide reports on recommendations, changes, and progress to the DAG and the PEP following scheduled meetings in both year 1 (PEP and DAG) and year 2 (DAG). Lee, as the Principal Investigator, will provide narrative and financial reports to the Mellon Foundation each year no later than December 31, 2012 and December 31, 2013, respectively. Several other project personnel will work with Lee on this reporting: Kirschenbaum, Woods, Porter Olson (MITH GA), Alexandra Chassanoff (SILS GA), Tammy Cox (SILS Director of Business Operations), and Christina Grogan (MITH Business Manager).

3.3 System and Integration Requirements

3.3.1 API(s)

BitCurator depends on a number of software packages with mature, stable APIs, including AFFLIB and The Sleuth Kit. Because of these dependencies, access to the underlying APIs will be retained, although no external API support for BitCurator is planned during the initial development phase. Note: Queries regarding RESTful web API access were raised during the PEP meeting in December, 2011. This access is out of scope at the present time.

3.3.2 Build Environment

Setting up a build environment for BitCurator will vary depending on the target platform. For those packages which are distributed as source code, we will provide build instructions for the software based on specific environmental exemplars:

1. Standard UNIX build environment tools as distributed by Canonical as part of Ubuntu 12.04
2. Native compiled Windows versions of specific applications and support tools, including Bulk Extractor, Bulk Extractor Viewer, and Python reporting scripts.
3. Potentially [lower priority than 1 and 2]: Compilation on Mac OS X (Lion or current) using additional packages from MacPorts

3.4 Security Requirements

Institutions that make use of BitCurator software will be subject to their own security requirements. The BitCurator project will develop tools and methods to help them identify and restrict access to sensitive data from disk images, but it will be the responsibility of implementing institutions not to disseminate sensitive data from their collections.

In BitCurator documentation, users will be encouraged to use write-blocking hardware when obtaining disk images, in order to prevent accidental damage to donor or archival materials.

Software produced by the BitCurator project will undergo an independent code review for functionality and security in year 2.

3.5 User Interface Requirements

The primary BitCurator interface will include both a custom GUI that provides access to basic forensic imaging, processing, reporting, and exporting services, along with the interfaces (GUI and/or command line) to existing tools. New BitCurator interface elements will be constructed in Python and will run cross-platform. BitCurator will also draw significantly on existing support code provided with bulk extractor, fiwalk, and The Sleuth Kit.

3.6 Documentation Requirements

BitCurator documentation will consist of the following:

1) Documentation for installation of the software sources. README, INSTALL, and ChangeLog for all primary releases.

2) Full product documentation. A searchable wiki entry of all BitCurator software functions (along with sample use cases) will be provided and updated for all major subreleases (0.1, 0.2, etc.). We will provide UNIX-style man pages for appropriate software, along with developer documentation included with source code repositories. Downloadable PDFs of project documentation will be provided prior to the final phase one release.

3) FAQs. The BitCurator web site will, over the course of the project, be populated with FAQ-style responses to community questions, common procedures associated with digital forensics tools, and other examples.

4) White papers, articles and conference papers.

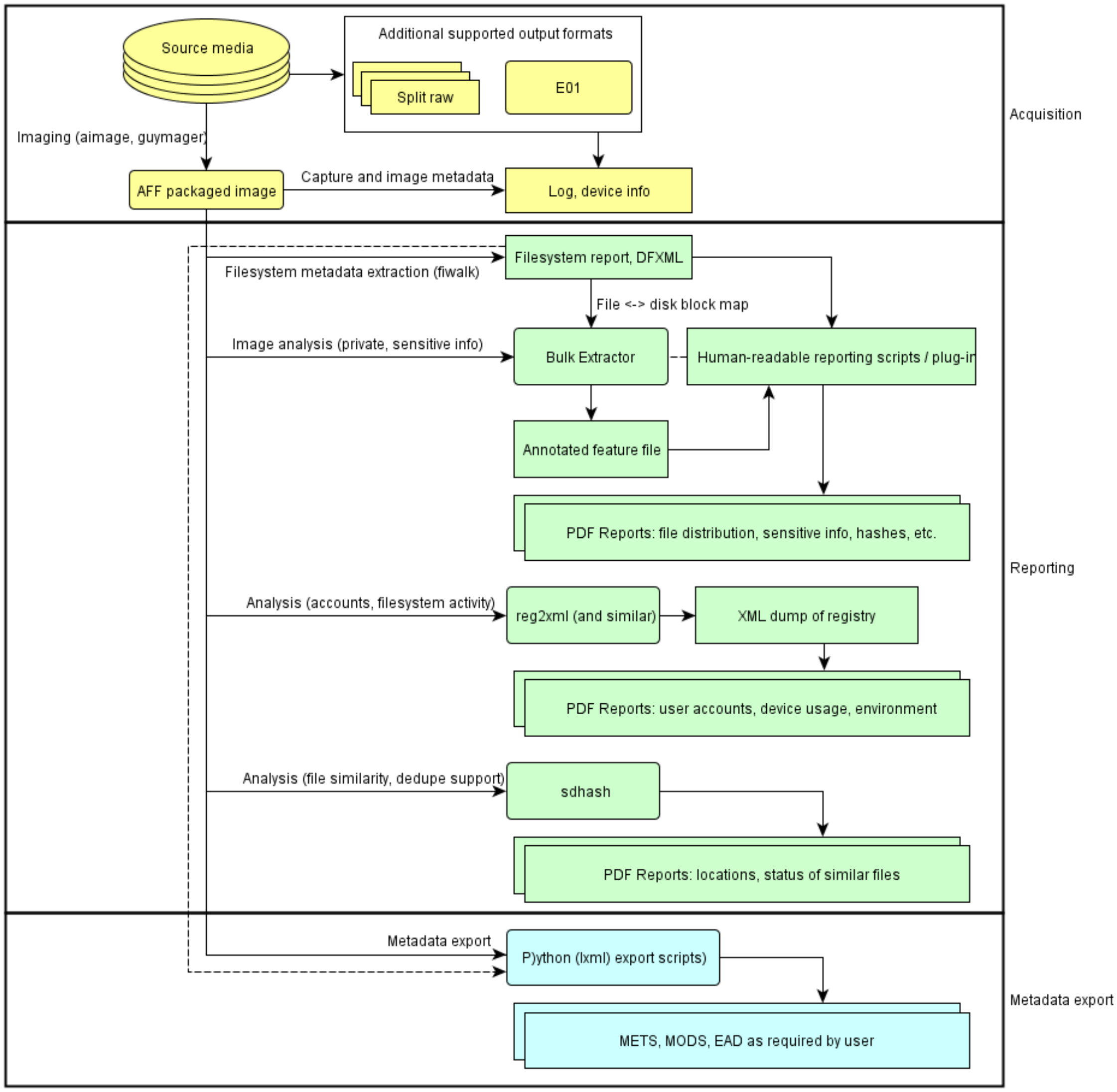
4. High-Level Technology Architecture

BitCurator will constructed using a series of cross-platform software tools and services to provide LAM professionals with simplified access to digital forensics methods.

The following table provides a partial list of open source libraries and software that will provide back-end support for the BitCurator interface and processing modules. Note that this may be subject to substantial change based on discussions with and recommendations of the Professional Expert Panel (PEP) and Development Advisory Group (DAG).

Forensics software	Current Version	Purpose
Data Processing		
Advanced Forensic Format Library (AFFLIB)	3.7.1	Support for AFF packaging of disk images
fiwalk	0.6.16 (or equivalent functionality in current SleuthKit Release)	Fast file and inode walks, DFXML output
Bulk Extractor	1.3.0	Identification of private/sensitive information
sdhash	2.0	Fuzzy hashing, file similarity
SleuthKit	3.2.3, 4.0 when available	Accessing and processing filesystem data
reg2xml, regXMLParse.py	0.1	Extraction and processing of Windows Registry data (including XML output)
Metadata		
lxml (libxml2, libxslt)	2.3	Python XML library, metadata handling and DFXML reprocessing
User Interface		
wxPython	2.8.12	BitCurator cross-platform interface development
Packaging		
Ubuntu packaging tools – build-essential, devscripts, ubuntu-dev-tools, debhelper, dh_make, diff, patch, cdbs, quilt, gnupg, fakeroot, lintian, pbuilder	Varies (see Ubuntu packaging docs)	Support for apt packaging of BitCurator sources, generation of the BitCurator virtual environment, and construction of a bootable ISO image

The diagram included below provides a basic, high-level overview of how these technologies will be integrated into the BitCurator architecture. For the sake of clarity and space, this diagram does not include microservice hooks or GUI-specific actions.


Source media
Additional supported output formats
Split raw
E01
Acquisition
Imaging (aimage, guymager)
AFF packaged image
Capture and image metadata
Log, device info
Filesystem report, DFXML
Filesystem metadata extraction (fiwalk)
File <-> disk block map
Image analysis (private, sensitive info)
Bulk Extractor
Human-readable reporting scripts / plug-ir
Annotated feature file
PDF Reports: file distribution, sensitive info, hashes, etc.
Reporting
Analysis (accounts, filesystem activity)
reg2xml (and similar)
XML dump of registry
PDF Reports: user accounts, device usage, environment
Analysis (file similarity, dedupe support)
sdhash
PDF Reports: locations, status of similar files
Metadata export
P)ython (lxml) export scripts)
METS, MODS, EAD as required by user
Metadata export

5. User Support

The BitCurator team will provide input and guidance to PEP and DAG members who are testing the BitCurator software. This will include information on configuring and using pre-release versions of the BitCurator virtual environment and standalone tools, along with step-through guides for specific data acquisition and analytic strategies.

The BitCurator team will also provide implementation suggestions and accept bug reports from other interested individuals who test the software. The BitCurator team will not provide live user support, nor will they provide support for products on which the BitCurator software depends (such as those in the AFFLIB repository, The Sleuth Kit, and existing open source libraries that form the main dependency tree).

BitCurator support will be provided primarily through “How-To” write-ups and a Frequently Asked Questions section published on the BitCurator wiki (http://wiki.bitcurator.net/)

The BitCurator team will additionally provide documentation on building and installing related applications on each of the supported platforms (Ubuntu 12.04 and Windows 7), configuration and use of the BitCurator virtual environment, and hardware guidelines.

6. Appendices

This version of the requirements document is supported by a series of external technical documents covering the specifics of disk imaging, forensics, and metadata extraction.

7. Glossary

Advanced Forensics Format (AFF) – an open packaging architecture for disk images and associated metadata

BitCurator Team – the projects dedicated staff, composed of the PI, Co-PI, Technical Lead and project graduate research assistants at MITH and SILS

Collection management system – an application or set of integrated applications used by a LAM for management of its collections and associated descriptive information; this includes repository management packages (e.g. ContentDM, DSpace, Fedora) and integrated library systems (e.g. Millenium)

Development Advisory Group (DAG) – one of the two project advisory groups, composed of individuals who have significant experience with development of software

Digital Forensics XML (DFXML) – a set of XML encoding conventions for forensics data, developed by Simson Garfinkel

Disk Image – an exact, sector-by-sector copy of the contents of a storage medium

LAM – library, archive or museum

Professional Expert Panel (PEP) – one of the two project advisory groups, composed of individuals who are at various levels of implementing digital forensics tools and methods in their collecting institution contexts

8. References

“BitCurator: Tools for Digital Forensics Methods and Workflows in Real-World Collecting Institutions.” Grant Proposal to the Andrew W. Mellon Foundation, 2011.

Garfinkel, Simson L. 2006. AFF: A New Format for Storing Hard Drive Images. *Communications of the ACM* 49 (2):85-87.

9. Index

access, 3, 5, 6, 7, 8, 9, 10, 12
AFFLIB, 10, 12, 14
BitCurator, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 15
Bulk Extractor, 6, 10, 12
Development Advisory Group, 1, 2, 12, 15
Digital Forensics XML, 6, 15
disk image, 5, 6, 7
EAD, 7
forensics, 2, 3, 6, 8, 11, 12, 15

ISO, 5, 6, 9, 12
Linux, 5, 8, 10
metadata, 2, 3, 5, 6, 7, 8, 12, 15
MITH, 1, 4, 7, 9, 15
PEP. *See* Professional Experts Panel
Principal Investigator, 9
redaction, 7
Technical Lead, 1, 2, 15
The Sleuth Kit, 9, 10, 14
UNC, 4, 7
Windows, 6, 8, 9, 10, 12, 14
XML, 6, 12, 15