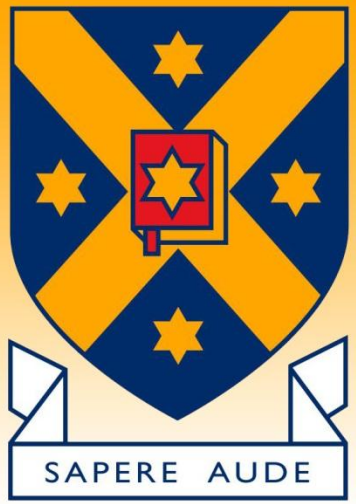


Application Footprints:

Automating the Discovery of Software Artifacts (Work In Progress)

Thomas Laurenson
Department of Information Science
School of Business
University of Otago
Dunedin, New Zealand

2013 Digital Forensic International Conference
Auckland, New Zealand

Schedule

- The Problem
- The Question
- Application Footprints - Background
- Research Methodology
- Research Challenges
- Conclusion
- Future Research

The Problem

- Automated forensic analysis of software artifacts is limited
- Current trends in DF makes manual analysis problematic
- Previous academic research has investigated forensic artefacts from a variety of different types of software
  - Anti-forensic tool artefacts [1, 2]
  - Instant Messaging artefacts [3, 4]
  - Web browser and other Internet artefacts [5, 6]
- But the resultant research does not have full impact, because…
  1. No automated techniques for forensic analysis
  2. No data abstraction to store and process this type of data

The Question

What are the challenges associated with automated software artifact detection?

Application Footprints

- In 2010 Garfinkel [7] defined application profiles as collections of:
  - **Files**, configuration changes (**registry** and **plist**) and document or network traffic **signatures** that make up an application
  - Therefore, they are a reference data set
- In 2011 Garkfinkel [8] proposed an Application Footprint XML format:
  - Means for distributing information about software artifacts using DFXML

Digital Forensic XML (DFXML)

"Digital Forensics XML is an XML language that enables the exchange of structured forensic information" [9]

- DFXML provides the ability to document:
  - Structured **metadata** tailored for the specific needs of digital forensics
  - Interchangeable, machine readable format for use between forensic tools
- Open and extensible, so perfect for research and development
- Excellent for tool composibility
  - Tool output can be used directly for other tool input
- Library is available in C++ and Python [10]

DFXML Example (fileobject)

```
<fileobject>
  <filename>Users/tlaurenson/MyDocuments/report.pdf</filename>
  <filesize>1421998</filesize>
  <alloc>1</alloc>
  <mtime>1230764913</mtime>
  <atime>1230764978</atime>
  <ctime>1230764913</ctime>
  <byte_runs>
    <run file_offset='0' fs_offset='241542144' img_offset='241542144' len='1421998'/>
  </byte_runs>
  <hashdigest type='MD5'>dede94f84fb2d00dc93ed00fda272a18</hashdigest>
</fileobject>
```

Research Methodology

1. Data Collection
2. Data Analysis
3. Application Footprint XML Development
4. Tool Development
5. Preliminary Testing

1. Data Collection

- Need to recreate the various stages in the software lifecycle:
- Data collection process based on **NIST Diskprint** methodology [11]
  - Use Virtual Machines (VMs) for data creation

Stage Number/Name	Process
1 – Ground Truth Data	Install OS
2 – Installation	Install software (default options)
3 – Execution	Execute software for a specific task
4 – Uninstallation	Uninstall software
5 – Restart	Restart OS

2. Data Analysis

- Generate DFXML metadata for each forensic image:
  - Using *fiwalk* (**F**ile and **I**node **Walk**er) [13,14]
  - Produces DFXML fileobjects from a forensic image

- **Differential forensic analysis** to identify changes between each stage [12]

- Also, used system monitor tools to record software changes:
  - Process Monitor [15], Uninstallation Tool [16]

3. Application Footprint XML Development

- Added a selection of new DFXML tags to encapsulate fileobjects

  <installer>

  <source>

  <portable>

  <install>

  <residual>

- Selection of DFXML tags to include in profile:

  <filename>

  <filesize>

  <alloc>

  <hashdigest>

Application Footprint Snippet (Truecrypt)

```
<application_footprint>
    <installer>
      <fileobject>
        <filename>TrueCrypt Setup 7.1a.exe</filename>
        <filesize>3466248</filesize>
        <alloc>1</alloc>
        <hashdigest type='MD5'>7a23ac83a0856c352025a6f7c9cc1526</hashdigest>
      </fileobject>
    </installer>
    <install>
      <fileobject>
        <filename>Program Files/TrueCrypt/truecrypt.sys</filename>
        <filesize>231760</filesize>
        <alloc>1</alloc>
        <hashdigest type='MD5'>ed5e4ce36c54f55e7698642e94d32ec7</hashdigest>
      </fileobject>
    </install>
    <residual>
      <fileobject>
        <filename>$OrphanFiles/TrueCrypt Format.exe</filename>
        <filesize>1610704</filesize>
        <unalloc>1</unalloc>
        <hashdigest type='MD5'>48538c19abe905d22e147b1c25d90880</hashdigest>
      </fileobject>
    </residual>
</application_footprint>
```

4. Tool Development: Vestigium

- Vistigium – latin for footprint, trace
- Written in Python, based on Garfinkel's idifference.py tool [12,17]
- Basic principle:
  - Helps automate creation/scanning of Application Footprint XMLs
  - Create, Search, Scan modes of operation

5. Preliminary Testing

- Created Application Footprint XMLs for several anti-forensic tools
  - Truecrypt (encryption software)
  - CCleaner (privacy tool/system cleaner)
  - FileShredder (file wiper)
- Created a variety of target investigation scenarios
  - Measured success of detection/identification of artifacts
- **Preliminary findings**, so far able to:
  - Detect software artifacts
  - Classify artifact state (installed/uninstalled)
  - High recall and precision

Research Challenges

- Is DFXML file metadata sufficient for the data set?
- Are profiles from multiple platforms needed?
  - Windows vs. OSX vs. Linux
- Are profiles from multiple systems beneficial?
  - WinXP vs. Win7
  - 32bit vs. 64bit
- Are profiles from every software version beneficial?
- Are performance decreases acceptable?
  - For example, when compared to Hash Set Analysis

Conclusion

- Application Footprinting is a **next-generation** forensic technique
- Provides an **automated analysis** platform
- Removes manual analysis requirements
- DFXML is an excellent forensic data abstraction
- Potential for other researchers to use this format in their research
  - To create profiles from conducted research

Future Research

- Include registry objects (RegXML Extractor)
- Tool Testing
  - Compare results to Hash Set Analysis Techniques
- Enhanced detection methods
- Are more advanced information sources beneficial:
  - Memory, network traffic artifacts, piecewise hashing
- What about non-file based artifacts?
  - How to store/process artifacts not in a file?
  - Possible to use the feature file technique from bulk_extractor tool?

References

[1] Geiger, M., Cranor, L. (2006). Scrubbing Stubborn Data: An Evaluation of Counter-Forensic Privacy Tools. IEEE Security & Privacy, 4(5),pp. 16-25.

[2] Martin, T., Jones, A. (2011). An Evaluation of Data Erasing Tools. In Proceeding of the 9th Australian Digital Forensic Conference. pp. 84-92. Perth, Australia.

[3] van Dongen,W.S. (2007). Forensic Artefacts Left by Pidgin Messenger 2.0. Digital Investigation 4(3–4), pp. 138-145.

[4] van Dongen, W.S. (2007). Forensic Artefacts Left by Windows Live Messenger 8.0. Digital Investigation 4(2), pp. 73-87.

[5] Mee, V., Tryfonas,T., Sutherland, I. (2006). The Windows Registry as a Forensic Artefact: Illustrating evidence collection for Internet usage. Digital Investigation 3(3), pp. 166-173.

[6] Yasin,M., Cheema, A., Kausar, F. (2010). Analysis of Internet Download Manager for Collection of Digital Forensic Artefacts. Digital Investigation 7(1–2), pp. 90-94.

[7] Garfinkel, S. (2010). Digital Forensic Research: The next 10 years. Digital Investigation 7(S) pp. 64-73.

[8] Garfinkel, S. (2011). Application Footprint XML. Retrieved from http://www.forensicswiki.org/wiki/Application_Footprint_XML

[9] Garfinkel, S. (2012). Digital Forensics XML and the DFXML Toolset. Digital Investigation 8(1) pp. 161-174.

[10] Garfinkel, S. (2013). Digital Forensic XML – Forensics Wiki. Retrieved from http://www.forensicswiki.org/wiki/Category:Digital_Forensics_XML

[11] Laamanen, M & Tebbutt, J. (2012). An Application Footprint Reference Set: Tracking the lifetime of software. US DoD Cyber Crime Conference. Atlanta, GA. 26/12/2012. Available from http://www.nsrl.nist.gov/Documents/DoDCC2012DiskprintAnalysis.pdf

[12] Garfinkel, S., Nelson, A. & Young, J. (2012). A General Strategy for Differential Forensic Analysis. Digital Investigation 9(S), pp. 50-59.

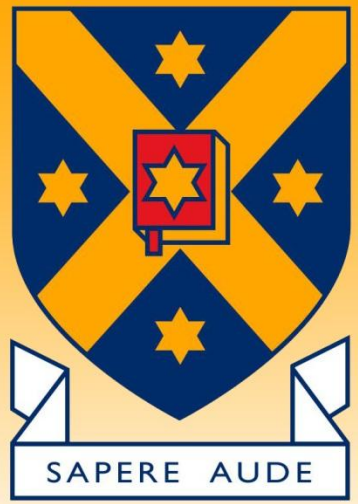
[13] Garfinkel, S. (2009). Automating Disk Forensic Processing with SleuthKit, XML and Python. *Fourth International IEEE Workshop on Systematic Approaches to Digital Forensic Engineering*.

[14] Garfinkel, S. (2013). Fiwalk – Forensics Wiki. Retrieved from http://www.forensicswiki.org/wiki/Fiwalk

[15] Windows Sysinternals. (2013). Process Monitor. Available from http://technet.microsoft.com/en-us/sysinternals/bb896645.aspx

[16] CrystalIdea. (2013). Uninstallation tool. Available from http://www.crystalidea.com/uninstall-tool

[17] Garfinekl, S. (2013). simsong/dfxml. Available from https://github.com/simsong/dfxml



Thank you for your attention
Any Questions?