COMP40780 - Final project
Literature survey

Andrea Barberio
May 2016

ForeWork - what is it?

Etymology[edit]

From *fore-* + *work*.

Noun[edit]

forework (*plural* **foreworks**)

1. A fortifying structure erected for defense, such as an embankment, wall, or tower.

Verb[edit]

forework (*third-person singular simple present* **foreworks**, *present participle* **foreworking**, *simple past and past participle* **foreworked**)

1. (*transitive*) To work beforehand or in preparation for; make ready; prepare. [quotations ▼]

[Sources: https://en.wiktionary.org/wiki/forework :: http://images.travelpod.com/tw_slides/ta00/c8f/192/forework-of-stirling-castle-stirling.jpg]

ForeWork - what is it?

It is an open source forensic framework, aimed at **triaging and analysing forensic artifacts**, automatically or semi-automatically.

Its goals include **ease of use**, **flexibility** and **scalability**, and to run equally on a **single machine** or on a **distributed infrastructure**.

Nothing new under the sun?

There are other forensic systems with similar goals. Is ForeWork **original**? And **how to verify** that?

Several sources:

- Google Scholar
- UCD One Search
- Google Search
- And a lot of time..

The others

- **The Sleuth Kit and Autopsy**
- **OCFA**
- **XIRAF**
- **Hansken**
- **Distributed FTK** and **EnCase Processor Manager**

Non-competitors:

- **GRR Rapid Response**, **Volatility**, **Rekall**

Autopsy and The Sleuth Kit, the in
s companions

Pros: modular, open source, relatively easy to use, able to deal with many different filesystems, commercially supported. Autopsy Hadoop exists

Cons: focused on the sole extraction of the artifacts (no deep analysis)

http://ieeexplore.ieee.org.ucd.idm.oclc.org/xpls/abs_all.jsp?arnumber=4076922&tag=1

[http://www.thisiswhyimbroke.com/at-at-star-wars-dog-costume]

OCFA, the old glory

Pros: it is fast, written in C++, modular, open source, reproducible, stable, has an UI

Cons: it is abandoned, has a higher entry bar for developers, written in C++, untested scalability, designed with data locality in mind, and it was relatively difficult to use

http://link.springer.com/chapter/10.1007/978-1-4419-5803-7_4

[© Disney]

XIRAF, the father of Hansken

A proof of concept, soon **abandoned** in favour of Hansken.

It provides XQuery, a powerful **query language** on the analysed artifacts.

Closed source, only available to the Dutch Police. **Not scalable** (runs on a single machine). XML and XSLT are heavy weights.

http://www.sciencedirect.com/science/article/pii/S1742287606000776

[© Disney]

Hansken, the cool kid

It is modular, it can do distributed analysis on top of Hadoop, and can do distributed search. It has a nice scalability model and a modular infrastructure.

It is not open source, nor available for public use (only available to the Dutch Police). It is not suitable for single-host analysis, for triaging tasks, and for real-time analysis (streaming vs. batch analysis, but things are changing)

http://www.sciencedirect.com/science/article/pii/S1742287615000857

[© Disney]

Distributed FTK, EnCase Process Manager, the darksiders

Pros: de facto standards in the industry

Cons:

- Closed source
- Non-verifiable
- Unproven scalability

[© Disney]

GRR, Volatility, Rekall, the outsiders

Tools that can be integrated:

- GRR, developed by Google, is a live system analysis and imaging tool
- Volatility is a live memory forensics tool
- Rekall is Google's version of Volatility for cloud-oriented memory analysis

Network forensics is a thing, too

[© CBS Interactive, Inc.]

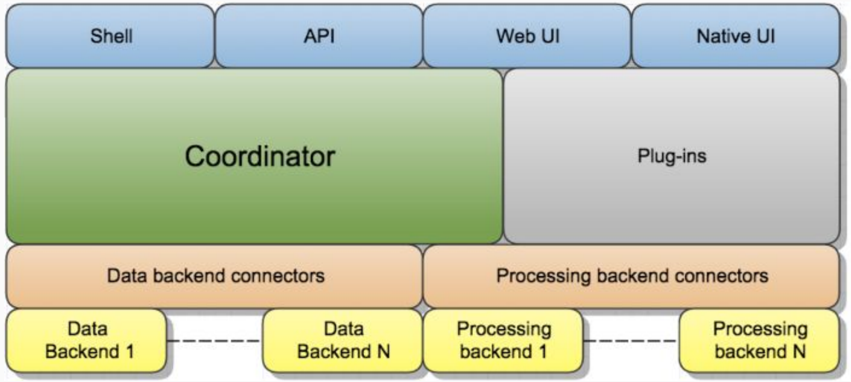
ForeWork, a new hope

Prototype. Open source, modular, scalable streaming processor of forensic artifacts.

Multiple use cases: small investigation on single machine, large distributed investigation

Component of a larger system: ingesting -> processing -> storing -> querying -> visualizing

Aimed at semi-automated analysis, can facilitate probabilistic analysis



ForeWork, a new hope

Goals (and how to achieve them):

- Simple adoption by forensic investigators (using Python, a low entry-bar language)
- Speed (components are rewritten C++ where necessary)
- Batteries included (coming with a set of basic analysis modules)
- **Scalability** (multiple processing backends, e.g. ipyparallel, Twitter Heron, Apache Storm)
- **Interactivity** (the framework is wrapped around by an IPython shell)
- Reusability (written as a set of libraries)
- Modularity (tasks are implemented as plugins)

(cool stuff in **bold**)

ForeWork, a new hope

- Integration, can use external tools (plugins and external tools)
- Interoperability, can work with other tools (import/export state in json, xml, etc)
- Reproducibility, any investigator can achieve the same results (by maintaining the chain of evidence)
- Stability (the system is able to recover from errors without stopping the analysis)
- Isolation (the results are not depending on external factors)
- **Triaging of the artifacts** (e.g. focusing on the most relevant evidence first, as the first 48 hours of an investigation are the most important ones)
- Allow for profiling, to identify the bottlenecks and the wins in the future design iterations

[© Disney]

ForeWork, a new hope

- Minimize wait time (asynchronous, streaming, non-blocking task scheduler)
- Can make the best out of the available computing resources (with automatic task load-balancing, depends on processing backends)
- **Guessing artifact properties** (e.g. entropy to identify encryption, steganography, polyglot files, etc)
- Multiple data backends (MattockFS, NFS, etc)
- Indexing and searching
- Easy automation via API
- Easy to use via GUI on top of the API
- Well documented, to minimize the need for technical support

The promises I can maintain

AKA: goals inside the project’s timeline

- Making it open source, interactive and scalable
- Making it extensible
- Provide the basic, most important plugins

The promises I’d like to maintain

AKA: goals that I can’t probably make in time

- Making it easy to use and robust
- Making it fast enough
- Provide more than the basic plugins
- Provide a nice API
- Write good documentation

The promises I cannot maintain

AKA: goals that are definitely not in this project’s timeline

- Implement indexing and searching
- Design a data store for the extracted artifacts
- Provide a GUI
- Make it bug-free :)

Questions?

[© Disney]

References

- **Digital forensics as a service: Game On**, 2015, H.M.A. van Beek et al.
- **XIRAF - XML-based indexing and querying for digital forensics**, 2006, W. Alink et al.
- **Open Computer Forensic Architecture, a Way to Process Terabytes of Forensic Disk Images**, 2009, Oscar Vermaas et al.
- **Is the Open Way a Better Way? Digital Forensics Using Open Source Tools**, 2007, D. Manson et al.
- **Open Source Software for Digital Forensics**, 2010, Eva Huebner, Stefano Zanero
- **A second generation of computer forensic analysis system**, 2009, Daniel Ayers
- **ForeWork's project page**, https://github.com/insomniacslk/forework
- **Twitter Heron**, an open-source distributed stream computation system, https://blog.twitter.com/2015/flying-faster-with-twitter-heron