Design and Implementation of Zeitline: A Forensic Timeline Editor

By

Florian Buchholz and Courtney Falk

Presented At

The Digital Forensic Research Conference

DFRWS 2005 USA New Orleans, LA (Aug 17th - 19th)

DFRWS is dedicated to the sharing of knowledge and ideas about digital forensics research. Ever since it organized the first open workshop devoted to digital forensics in 2001, DFRWS continues to bring academics and practitioners together in an informal environment. As a non-profit, volunteer organization, DFRWS sponsors technical working groups, annual conferences and challenges to help drive the direction of research and development.

http:/dfrws.org

Design and Implementation of Zeitline: a Forensic Timeline Editor

Florian Buccholz

Courtney Falk

CERIAS, Purdue University

What is It?

- Forensic timeline editor
  - Focuses on event reconstruction such as netForensics' product without additional agent processes
  - Does not handle evidence retrieval like SleuthKit, FTK, et. al.
- Implemented in Java with a Swing GUI
- Just recently left alpha stage and entered beta development

Terms to Know

- Atomic event - discrete time event generated from a source file
- Complex event - collection of atomic events or also other complex events
- Source - file from which events were generated
- Timeline - hierarchy of events structured within a single complex event

Basic Functionality

- Deleting, copying, cutting, and pasting
- Saving/loading projects
- Searching/finding a keyword within a time range if desired
- Querying/filtering events based on a keyword and/or a time range

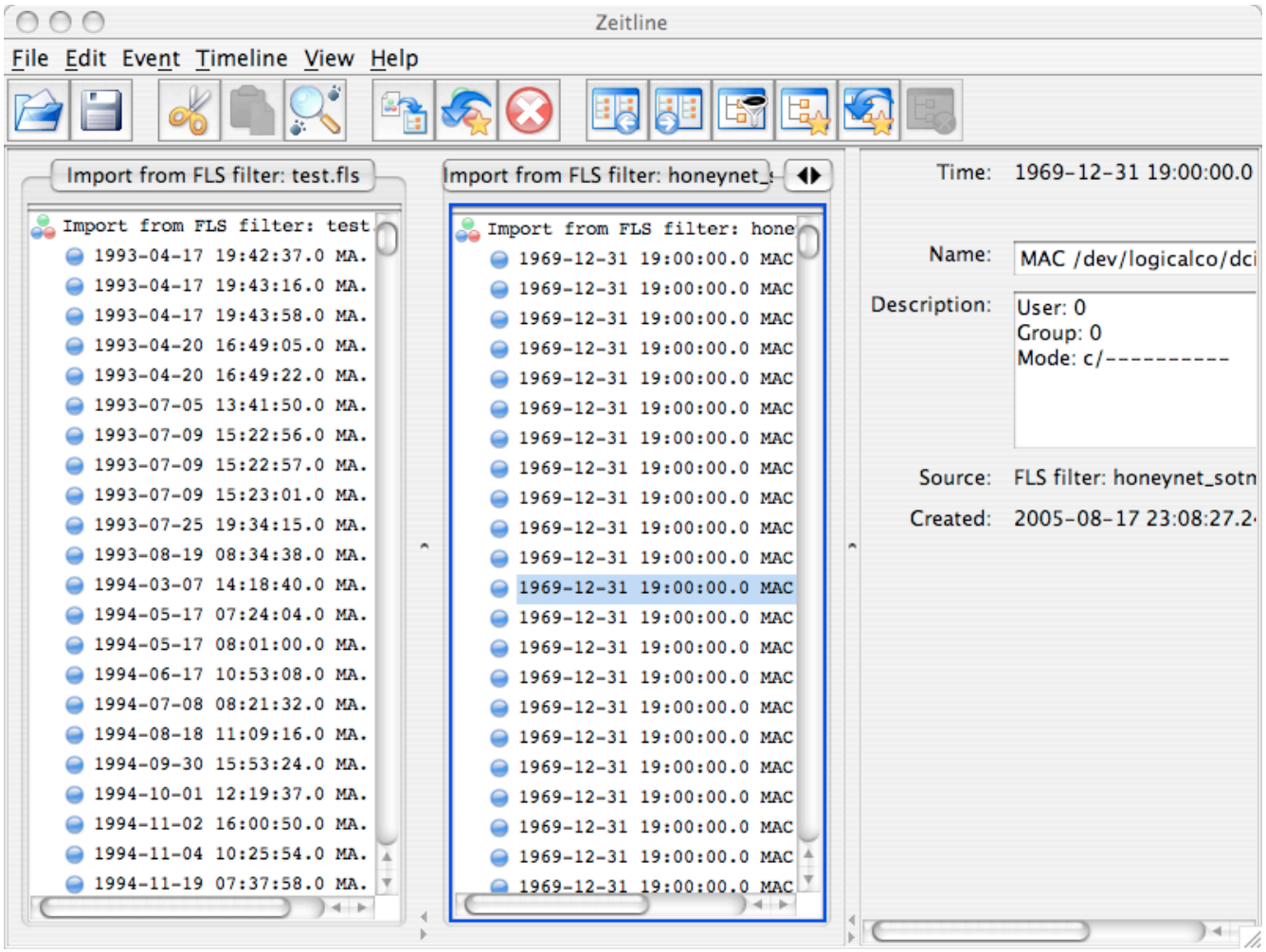
Forensic Integrity Design Considerations

- Orphan timeline - collection of all “deleted” events
- Deleting - all or none with events where either all events from a source are used or none
- Cutting - cut items are stored in a buffer, which is always saved with the project or purged to the orphan timeline if new items are copied/cut

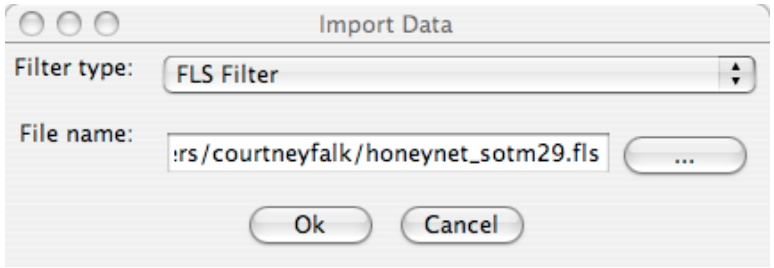
Performance

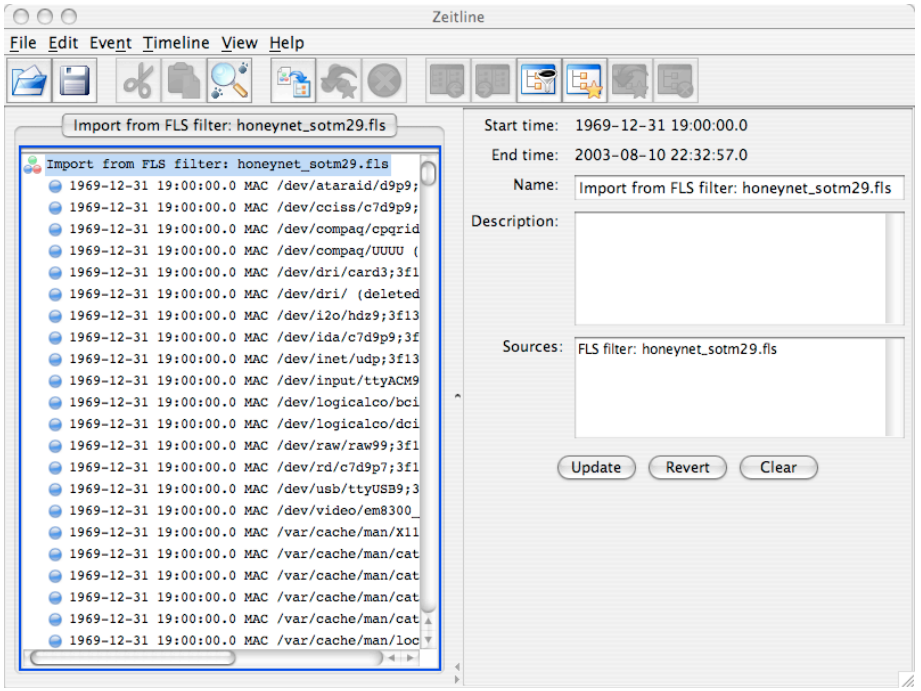
- Need to handle tens of thousands of elements
- java.util.TreeSet provided $O(n^2)$ build time
- Custom Adelson-Velksii-Landis (AVL) balanced search tree gives O(n log n) build time
- 86k events in 46.6 seconds, or 15k in 9.7s
- ~57% of processing time is graphical rendering

User Interface

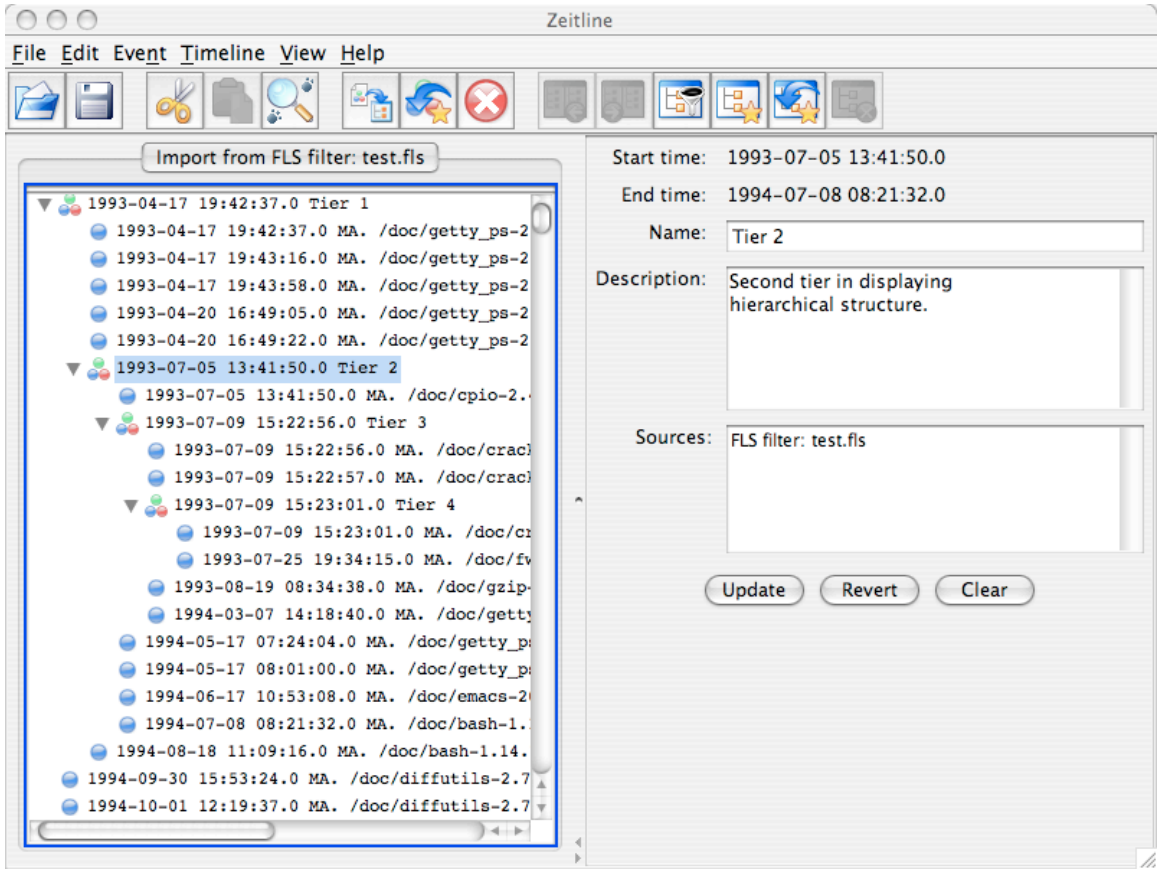


Importing Events

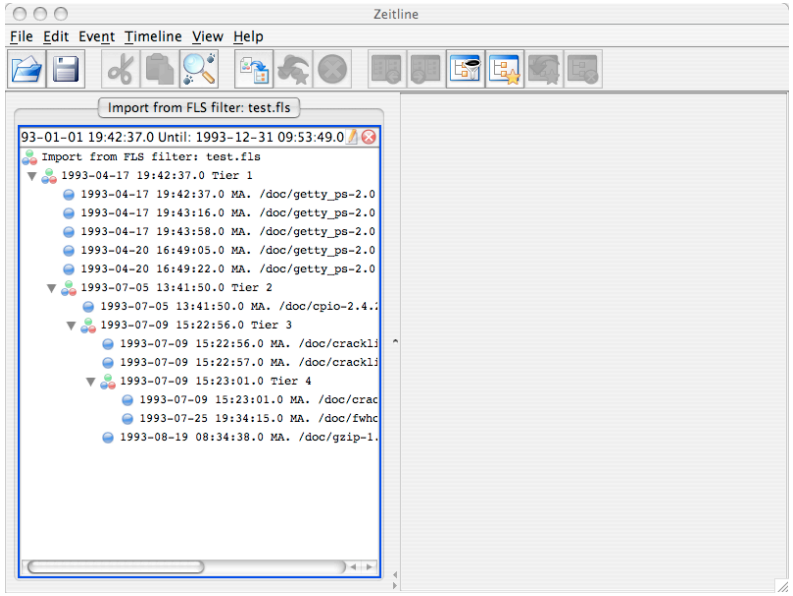


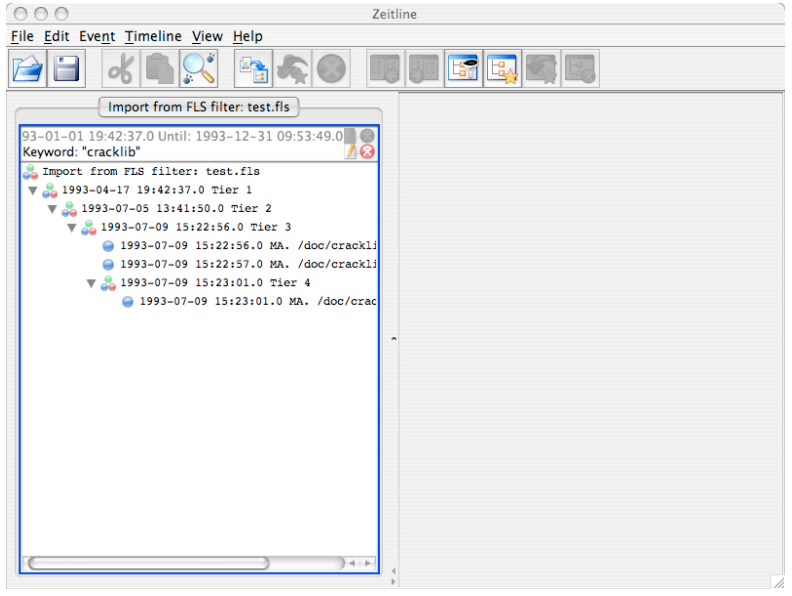


Hierarchical Structure



Queries


Zeitline
File Edit Event Timeline View Help
Import from FLS filter: test.fls
93-01-01 19:42:37.0 Until: 1993-12-31 09:53:49.0
Import from FLS filter: test.fls
1993-04-17 19:42:37.0 Tier 1
1993-04-17 19:42:37.0 MA. /doc/getty_ps-2.0
1993-04-17 19:43:16.0 MA. /doc/getty_ps-2.0
1993-04-17 19:43:58.0 MA. /doc/getty_ps-2.0
1993-04-20 16:49:05.0 MA. /doc/getty_ps-2.0
1993-04-20 16:49:22.0 MA. /doc/getty_ps-2.0
1993-07-05 13:41:50.0 Tier 2
1993-07-05 13:41:50.0 MA. /doc/cpio-2.4.
1993-07-09 15:22:56.0 Tier 3
1993-07-09 15:22:56.0 MA. /doc/crackli
1993-07-09 15:22:57.0 MA. /doc/crackli
1993-07-09 15:23:01.0 Tier 4
1993-07-09 15:23:01.0 MA. /doc/crac
1993-07-25 19:34:15.0 MA. /doc/fwhc
1993-08-19 08:34:38.0 MA. /doc/gzip-1.


Zeitline
File Edit Event Timeline View Help
Import from FLS filter: test.fls
93-01-01 19:42:37.0 Until: 1993-12-31 09:53:49.0
Keyword: "cracklib"
Import from FLS filter: test.fls
1993-04-17 19:42:37.0 Tier 1
1993-07-05 13:41:50.0 Tier 2
1993-07-09 15:22:56.0 Tier 3
1993-07-09 15:22:56.0 MA. /doc/crackli
1993-07-09 15:22:57.0 MA. /doc/crackli
1993-07-09 15:23:01.0 Tier 4
1993-07-09 15:23:01.0 MA. /doc/crac

Dynamic Filters

- All input filters, including those provided with the package, are dynamically loaded
- Overwhelming task of writing filters for all possible input sources
- Writing new filters is as easy as implementing a single abstract class and doesn’t require recompiling the entire project

InputFilter Class

```
public abstract class InputFilter
{
        public abstract Source init(String location,
                        Component parent);
        public abstract AtomicEvent getNextEvent();
        public abstract FileFilter getFileFilter();
        public abstract String getName();
        public abstract String getDescription();
        public abstract long getExactCount();
        public abstract long getTotalCount();
        public abstract long getProcessedCount();
}
```

Future Features

- Increased host information: user mappings, hardware and software details
- Events with no time value (logical events)
- Events with multiple hosts: clock synchronization, drift, and time zones
- Typing system for events
- Exportation such as XML output