High Speed Search Using Tarari Content Processor in Digital Forensics

By

Jooyoung Lee, Sungkyong Un, Dowon Hong

Presented At

The Digital Forensic Research Conference

DFRWS 2008 USA Baltimore, MD (Aug 11th - 13th)

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
High-speed Search using Tarari Content Processor in Digital Forensics

Jooyoung Lee
joolee@etri.re.kr
Contents

I. Overview

II. Proposed Mechanism

III. Conclusion

Background/Related Works

Architecture/Model/Evaluation

Summary/Further Works
Overview (1/3)

- Requirement for high-tech tools against high-tech crimes has been increasing steadily

- “Speed” is one of the hot issue in DF
  - 500GB HDD costing about $0.18/GB
  - Recent technology implementing areal density of 1 Tb/in²
  - Plan to commercialize 4 TB HDD for desktop PC by 2011 (Hitach GST)

- It means
  - 14 hours to search 1 TB of data with normally used forensic tools
  - “Size” is a serious problem in DF
Hardware forensic tools on the market

- Evidence cloning, password cracking aiming to acceleration

<table>
<thead>
<tr>
<th>Forensic Tool</th>
<th>Manufacture</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>HardCopyII</td>
<td>Voom technology</td>
<td>- H/W based imaging tools with writing protect</td>
</tr>
<tr>
<td>Shadow 2</td>
<td></td>
<td>- Up to 5.5 GB/min (ATA Drive)</td>
</tr>
<tr>
<td>Instant Recall</td>
<td></td>
<td>- 2nd generation instant recovery tool</td>
</tr>
<tr>
<td>TACC1441</td>
<td>Tableau</td>
<td>- Accelerating password recovery</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Attacks for algorithms WinRar, PGP, Winzip by a factor of 6-30 times with PRTK</td>
</tr>
<tr>
<td>T35e</td>
<td></td>
<td>- Write Blocker</td>
</tr>
<tr>
<td>OmniClone</td>
<td>Logicube</td>
<td>- Hard drive duplication system</td>
</tr>
<tr>
<td>Sonix</td>
<td></td>
<td>- at peek rate of 3.5 GB/min (SATA Drive)</td>
</tr>
</tbody>
</table>
Forensic Search Tools

- Main requirements
  - to present all the matching results without missing when an investigator gives a query
  - Forensic search needs more time than traditional search because it has to perform bitwise operations on the whole disk in the physical level

- Traditional approach to forensic search
  - Stream based search using bitwise comparison
  - Index-based search
  - Search based on distributed processing using multiple systems
Our approach

- **Design and develop a high-speed search engine with a Tarari CP**

- **Goals**
  - To get high-speed in forensic search
  - To be practical and scalable method
  - To apply hardware-based approach to the field of forensic search and analysis
  - To meet domestic requirements
    - support document files by domestic word processors
    - support Korean and English Language in the documents

- **Evaluation**
  - Compare performance and advantages to those of a popular forensic tool on the market – Encase
allows a user to develop applications that exploit Tarari RegEx Agent which provides an arbitrary content identification and characterization

- Enables applications to analyze fixed or variable patterns in a data stream at speeds up to 1 Gb/s

Applications

- Intrusion detection and prevention
- Anti-SPAM
- Content filtering
- MIME and XML parsing
- Anti-virus
- Real time message routing
- Protocol emulation/modeling
Architecture

- **Client**
  - Installed on a Windows system
  - Presents GUI to a user
  - Sends commands and receives its results

- **HSSB**
  - Server on a Linux system
  - XDR Translator
    - Network communication module
  - Preprocessor/Postprocessor
  - Search Engine
    - Search using Tarari board

- **Storage**
  - NAS connected with NFS
Internal process of search engine
- Initialization
- Compilation
- Loading
- Scan
Load balancing process

- Loads keyword(s) or regular expressions to agents
- 4 agents used
- Automatic load balancing model used
- Before loading, the keyword(s) must be compiled into Tarari image by a compiler

Flowchart:

1. Start
2. \texttt{rgxInitialize}
3. NonBlocking mode?
   - Yes: Initialize Poll counter
   - No: \texttt{rgxBuildHwStateMachine}
4. Compile
5. Load same images into each agent
6. End
Programming Model Used

- **Scanning Process**
  - The forensic image is scanned for keywords by the agents
  - A single threaded model used
  - When the jobs are completed,
    - Searched pattern
    - Starting point
    - End point

---

**Scanning Algorithm**

```plaintext
initialize
getHWConfiguration
read in rexFile
compileAndSave
loadImageToAgent
initializeThread
for total job {
    read in dataFile
    scanNonBlock
}
while(!JobListCompleted) {
    if (jobCompleted) {
        getResults
        printResults
        freeJob
    }
} 
freeJobList
deinitializeThread
shutdown```

---
## System Setup

### HSSB

<table>
<thead>
<tr>
<th>Platform</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>Intel Xeon 5149 2.33Ghz</td>
</tr>
<tr>
<td>Memory</td>
<td>1GB DDR2 667Mhz ECC</td>
</tr>
<tr>
<td>Disk</td>
<td>500GB 7.2K rpm SATA</td>
</tr>
<tr>
<td>Interface</td>
<td>PCI-X slot</td>
</tr>
<tr>
<td>Pattern Matching Board</td>
<td>Tarari Grand Prix 3200</td>
</tr>
<tr>
<td>OS</td>
<td>Linux Fedora Core 6</td>
</tr>
<tr>
<td>Compiler</td>
<td>gcc</td>
</tr>
</tbody>
</table>

### GUI Module

<table>
<thead>
<tr>
<th>Platform</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>Intel Core™2 2.4Ghz</td>
</tr>
<tr>
<td>Memory</td>
<td>3GB DDR2</td>
</tr>
<tr>
<td>OS</td>
<td>Microsoft Windows XP Professional SP2</td>
</tr>
<tr>
<td>Compiler</td>
<td>Visual Studio 2005</td>
</tr>
</tbody>
</table>
Objective
- To measure time to take for searching keywords

1 GB forensic image made with dd command of Linux

Keywords
- Single keyword
  - “홍길동”(Korean)
- Multiple keywords
- A Regular expression
  - [0-9][0-9][0-1][0-9][0-3][0-9] *- *[0-4][0-9] ][0-9 ]][0-9 ]][0-9 ]][0-9 ]][0-9 ]}
Evaluation 1 (2/2)

Search speed for keywords using the proposed method is faster over 5 times than that of EnCase.

The number in the parenthesis indicates the hit number of keywords:
- EnCase finds fewer patterns
- It is caused by the fact that EnCase could not extract texts in a structured format by a domestic word processor, Hangul.
Evaluation 2 (1/2)

- **Objective**
  - To measure speeds according to size variation of forensic images

- **4 forensic images made with *dd* command of Linux**
  - 274 MBytes
  - 552 MBytes
  - 1.1 GBytes
  - 2.03 Gbytes

- **Keywords**
  - “홍길동” (Korean)
  - [0-9][0-9][0-1][0-9][0-3][0-9] *- *[0-4][0-9] ][0-9 ][0-9 ][0-9 ][0-9 ][0-9 ]
This result shows the proposed method is so scalable that we can apply it to a very large scale of evidence practically.

<table>
<thead>
<tr>
<th></th>
<th>274 MB</th>
<th>552 MB</th>
<th>1100 MB</th>
<th>2030 MB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposed (keyword)</td>
<td>6</td>
<td>7</td>
<td>15</td>
<td>22</td>
</tr>
<tr>
<td>EnCase (keyword)</td>
<td>33</td>
<td>72</td>
<td>100</td>
<td>184</td>
</tr>
<tr>
<td>Proposed (regex)</td>
<td>5</td>
<td>7</td>
<td>11</td>
<td>22</td>
</tr>
<tr>
<td>EnCase (regex)</td>
<td>30</td>
<td>71</td>
<td>122</td>
<td>181</td>
</tr>
</tbody>
</table>

![Graph showing the comparison of proposed and EnCase methods over varying evidence sizes]
Summary

- **High-speed search in physical level**
  - Search a string in ADS (Alternative Data Stream) and hidden files
  - Support searching at 100 MB/sec
- **Supported file formats**
  - MS Office
  - PDF
  - HWP (Domestic word processor popularly used)
  - ...
- **Encoding**
  - ASCII, Unicode, UTF-7, UTF-8
- **Query keyword format**
  - Text in Korean and English
  - Regular Expressions
We have proposed a forensic searching method using hardware as a solution to those trends and requirements.

Our results show that search using a Tarari board can be performed over 5 times faster than tools currently on the market.

- same results with even a set of regular expression

It is feasible and practical approach for getting high speed in search and analysis of digital forensics.
Further Works

Problem

- Over-analysis or misanalysis requiring the investigators to spend time for filtering unnecessary data

To research methods to decrease over-analysis or misanalysis rate, keeping recall ratio 100%

- Presenting relatively fittest information to the investigator's intention in the front parts of the result list
- But, required a way to evaluate the satisfaction degree of the investigators

Web-based GUI

- Allow investigators an access to HSSB remotely for convenience