In4120 science fair

# Comparing Open Source Search Engine Functionality, Efficiency and Ef-fectiveness with Respect to Digital Forensic Search

# Introduction

- In cyber crime investigations, keyword searches are a key component.
- The underlying algorithms are often not revealed by the developers in many of the common tools used.
- Difficult to measure accuracy/efficiency.
- Therefore, this study looked to open-source search engines as an alternative, and if these are applicable to these kinds of investigations.
- Why is it relevant to the course?

# Main parts

- First part of the study
- Thorough literature review and comparison of supported functionality documented and a survey of available digital forensic datasets
- Second part:
- Solr and Elasticsearch selected and tested/compared by functionality and efficiency in searching, indexing and effectiveness of search results (with respect to digital forensic search using relevant datasets)
- Should help those in the digital forensic community as to what tools to use

# Background

- Big data landscape provides digital forensic investigations with ever-growing, large amounts of data, structured and unstructured
- How can investigators search through this data in a reasonable amount of time?
- How should it be stored?
- Relational databases found to be inappropriate for digital investigations – unstructured data and therefore requires other approaches
- Search engines

# What is needed in the search engine

- Data processing should be reliable, forensically sound
- Efficiency – ideally algorithms with low memory usage and time complexities
- Forensic search – restrictions on keyword search process, lack of standardized approach to data preprocessing/formatting, different file types
- New data formats put demand on quality of the search results, and should also follow Digital Forensics Process guidelines

# Choice of search engines/forensic tools

- Availability
- Underlying algorithms hidden in licensed search engines
- How were they chosen?
- Google Trends to get the most popular ones
- Then narrowed down based on documentation and tool category

Table 1: Comparison of search capabilities and functionality
**Source:** [2, 6, 10, 19, 20, 23, 25, 27]

| Capability | Sleuthkit | Volatility | Mozilla Invest-Gator | Hachoir | Elasticsearch | Solr | Sphinx |
|---|---|---|---|---|---|---|---|
| Regular expression | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Decide/Insensitive case | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | |
| Concurrent search | ✓ | | | | ✓ | | ✓ |
| Automate search, with respect to keywordlist | ✓ | | | | | | |
| Import keywords | ✓ | | | | | | |
| Export keywords | ✓ | | | | | | |
| Periodical search | ✓ | | | | | | |
| Substring matching | ✓ | | | | ✓ | ✓ | ✓ |
| Export search results | ✓ | | | | ✓ | ✓ | |
| Match highlighting | ✓ | | | | ✓ | ✓ | ✓ |
| UTF-8 Encoding support | ✓ | | ✓ | ✓ | ? | ✓ | ✓ |
| UTF-16 Encoding support | ✓ | | | ✓ | ? | | |
| ISO-8859-1 Encoding support | | | | ✓ | ? | | |
| Deduplication support | ✓ | | | | | ✓ | |
| Approximate hash based matching | ✓ | | | | | | |
| Orphan/deleted file search | ✓ | | | | | | |
| RAM search | ✓ | ✓ | ✓ | | | | |
| Matching memory structures (pre-made) | | ✓ | | | | | |
| Hash database lookups | ✓ | | | | | | |
| Wildcard | | ✓ | | | ✓ | ✓ | ✓ |
| Binary search | ✓ | ✓ | | | | | |
| HTML renderer for search results | | ✓ | | | | | |
| Support for masking sensitive fields | | | ✓ | | | | |
| Exact hash matching | | | ✓ | | | | |
| System provided keyword suggestions | | | | | | ✓ | |
| AND, OR, NOT, GROUP boolean operators | | | | | ✓ | ✓ | ✓ |
| + boolean operator (term must exist) | | | | | | ✓ | |
| File search filter | | | ✓ | | | | |
| Retrieval of documents not matching filters | | | ✓ | | | | |
| Set max search hits | | | ✓ | | ✓ | ✓ | |
| Stripping senstive metadata | | | ✓ | | | | |
| Increase search priority of important indexes | | | | | ✓ | | |
| Terminate search after a given elapsed time | | | | | ✓ | ✓ | |
| Sorting search results | | | | | ✓ | ✓ | ✓ |
| Customized message/ post-search action | | | | | ✓ | ✓ | |
| Aggregated summary of search results | | | | | ✓ | | |
| Narrow search results with post filter | | | | | ✓ | ✓ | |
| Set relevancy weight for field | | | | | ✓ | ✓ | |
| MoreLikeThisQuery | | | | | ✓ | ✓ | |
| Search result clustering | | | | | | ✓ | ✓ |
| Minimum matching criteria | | | | | | ✓ | |
| Fixed relevancy score | | | | | | ✓ | |
| Field collapsing | | | | | ✓ | ✓ | |
| Support for TF-IDF | | | | | ✓ | ✓ | ✓ |
| Language detection on index time | | | | | | ✓ | |
| Fuzzy matching | | | | | ✓ | ✓ | ✓ |
| Faceted search | | | | | ✓ | ✓ | |
| Phonetic search | | | | | ✓ | | |
| Geospatial search | | | | | ✓ | ✓ | |
| Streamed search | | | | | | ✓ | |

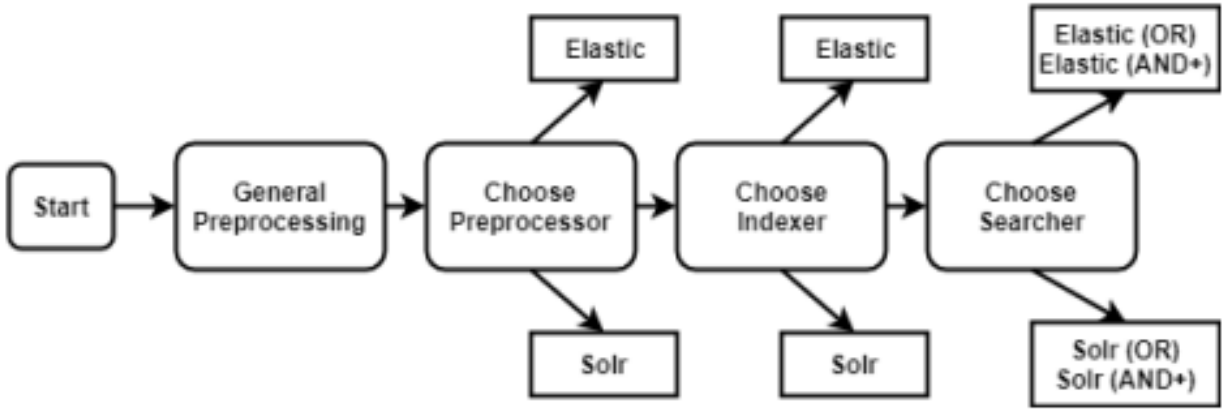
# Databases

- Overview of relevant publicly-available datasets was performed
- Not possible to obtain real-world data from crime investigation
- Using datasets created by researchers for data analysis purposes

1. **Fraud:** Enron email dataset [4].
2. **Network:** Snort IDS log file [26].
3. **Email:** Hillary Clinton emails [13].
4. **Malware:** VirusTotal and PE32 reports [21].
5. **Spam:** DITSSC [3].
6. **SMS:** NUSSC [5]

# Experiment methodology

- Experiments with fulltext searching
- Experiments performed on a set of search engines (*Solr* and *Elasticsearch*).
- Set of keywords based on domain knowledge of datasets and a search for strings that are not present in the dataset
- Searching within an index (i.e. not searching across all indexes or multiple indexes at the same time).
- Search time
- Cache temperature
- Memory measurement during search
- Search Accuracy - count of clear cut misses
- Out of box configurations (default values)



# How are the search engines compared?

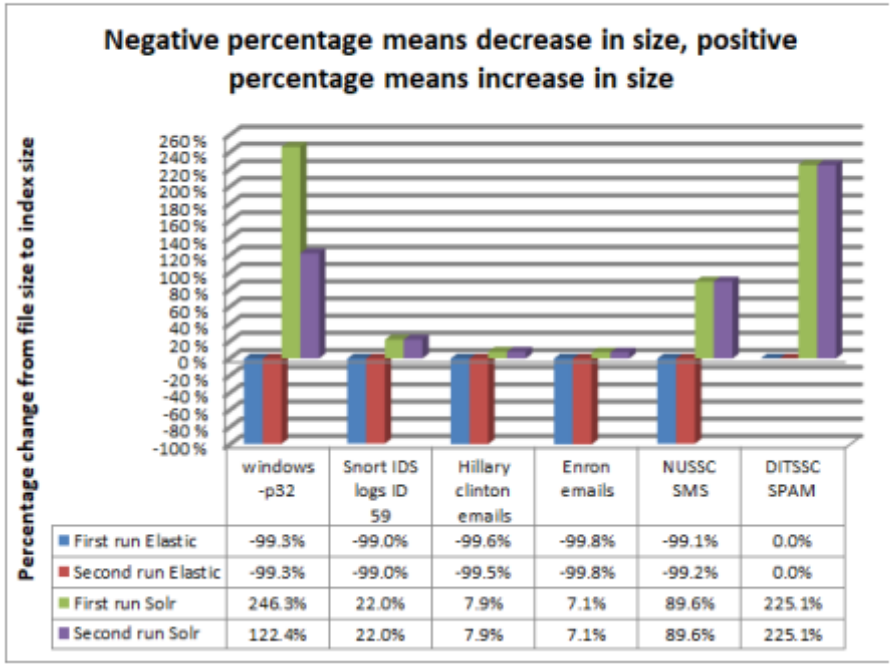
- Documented functionality put in a check list
- Experimental comparison of Solr and Elasticsearch
- Compared by: indexing, searching and memory consumption during searching

# Solr and Elasticsearch benchmark testing

- Setup used: Virtual Machine (6 cores, 40GB RAM and 2TB storage) with Ubuntu 16.04.3 LTS, Openjdk 1.8.0 131, Elasticsearch 6.0.1 and Solr 7.1.0 and Solr Cloud.
- Limitations: Single virtual machine, default configurations

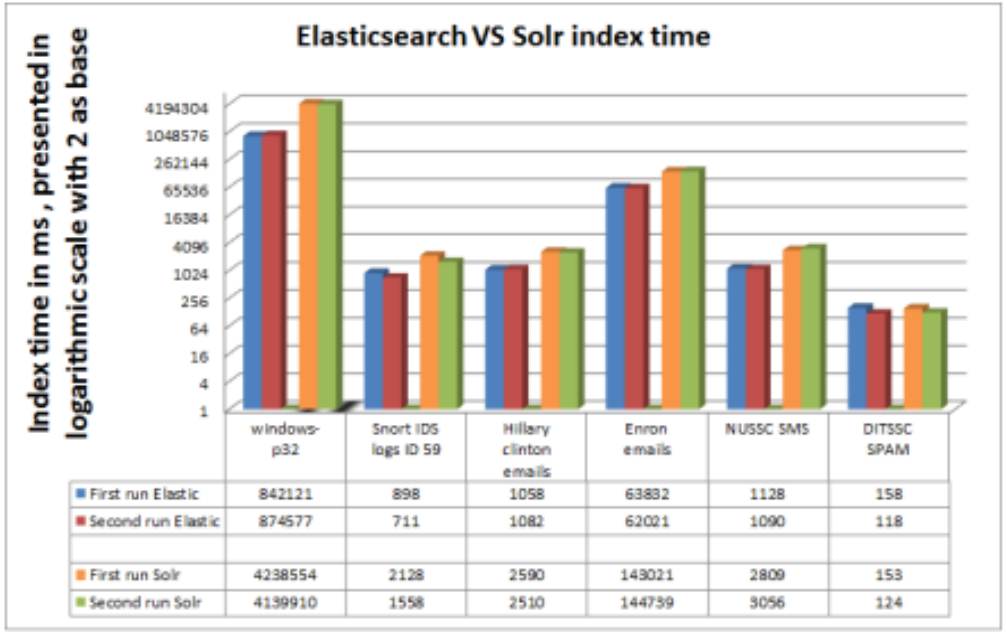
# Indexer performance

Figure 3: The change in index size



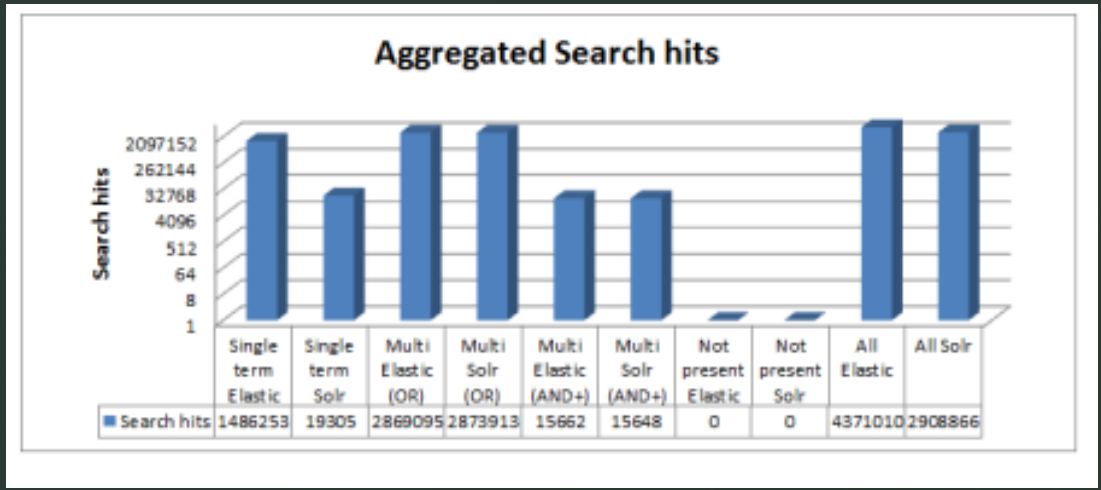

| | windows -p32 | Snort IDS logs ID 59 | Hillary clinton emails | Enron emails | NUSSC SMS | DITSSC SPAM |
|---|---|---|---|---|---|---|
| First run Elastic | -99.3% | -99.0% | -99.6% | -99.8% | -99.1% | 0.0% |
| Second run Elastic | -99.3% | -99.0% | -99.5% | -99.8% | -99.2% | 0.0% |
| First run Solr | 246.3% | 22.0% | 7.9% | 7.1% | 89.6% | 225.1% |
| Second run Solr | 122.4% | 22.0% | 7.9% | 7.1% | 89.6% | 225.1% |

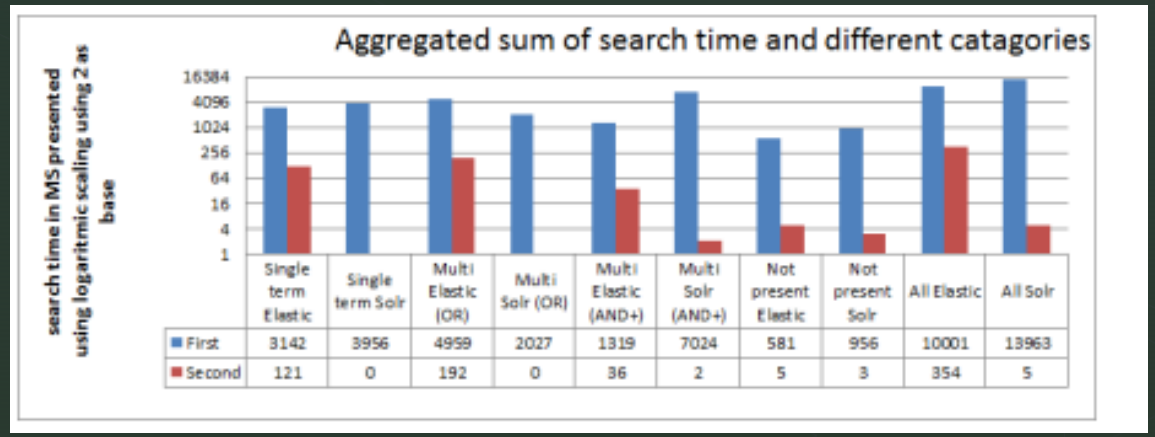
Figure 4: Index time Took and QTime




| | windows-p32 | Snort IDS logs ID 59 | Hillary clinton emails | Enron emails | NUSSC SMS | DITSSC SPAM |
|---|---|---|---|---|---|---|
| First run Elastic | 842121 | 898 | 1058 | 63832 | 1128 | 158 |
| Second run Elastic | 874577 | 711 | 1082 | 62021 | 1090 | 118 |
| | | | | | | |
| First run Solr | 4238554 | 2128 | 2590 | 143021 | 2809 | 153 |
| Second run Solr | 4139910 | 1558 | 2510 | 144739 | 3056 | 124 |

# Search performance results



# Memory usage

Table 3: Memory stats Elasticsearch and Solr during search

| **Elasticsearch** | | | | | **Solr** | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Virtual memory(VIRT): Size in GiB | | | | | Virtual memory(VIRT): Size in GiB | | | | |
| Average | Max | Min | Delta | Mode | Average | Max | Min | Delta | Mode |
| 42.831 | 42.831 | 42.831 | 0 | 42.831 | 29.144 | 29.144 | 29.144 | 0 | 29.144 |
| Physical memory (not swappable) - RES:size in GiB | | | | | Physical memory (not swappable) - RES: size in GiB | | | | |
| 2.807 | 2,898 | 2.666 | 0.232 | 2.898 | 2.936 | 2.964 | 2.881 | 0.083 | 2.964 |
| shared memory (SHR): size in GiB (rounded up) | | | | | shared memory (SHR): size in GiB | | | | |
| 0.34 | 0.43 | 0.2 | 0.23 | 0.43 | 2.296 | 2.323 | 2.241 | 0.082 | 2.323 |

# Discussion/conclusion

- For index size and index creation time, Elasticsearch is found to be favorable
- Some distinctions on functionality, but Solr has more capabilites useful for search in large-scale datasets
- For the chosen search test categories, Elasticsearch performed better on the first run, but Solr was better for the second run (where the second run should be more like a real operating environment with a warm cache etc.)
- The number of clear cut search misses were similar
- Memory consumption a lot higher for Elasticsearch – 13GiB more

# Future improvement of the study

- Suggestions to how the study can be improved on
- Compare specific search algorithms on the specific indexing and search methods used by Elastic and Solr.
- Do more than just one virtual machine – experiment in a multi-virtual machine environment
- ++

# Source

- https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2584227