Capstone Interim Project Report

Digital Forensics & Incident Response over a Remote Network

|  |  |
| --- | --- |
| Team Name: Polaris Sight |  |
| Owen Charters | N8962375 |
| Curtis Dooley | N9152237 |
| Peijun Xue | N10115358 |
|  |  |

Table of Contents

[Introduction 3](#_Toc515833086)

[Client 3](#_Toc515833087)

[Domain 3](#_Toc515833088)

[Application Context 4](#_Toc515833089)

[Project Scope and Plans 5](#_Toc515833090)

[Project Plan 5](#_Toc515833091)

[User Story Details 5](#_Toc515833092)

[Feasibility Analysis 6](#_Toc515833093)

[Technical Description 9](#_Toc515833094)

[Preferred Approach 10](#_Toc515833095)

[Appendix 12](#_Toc515833096)

# Introduction

## Client

The project that we are developing was conceived by our client, Dr Ernest Foo.

The client explained that the idea for the project stems from his time working with a large government organisation who lamented a lack of live forensic analysis software. In particular, the ability to assess an ongoing attack, and to determine the actions and aim of the hacker as well as the extent to which the system was compromised. Learning what the hackers are doing also helps to protect against future attacks.

## Domain

The project consists of elements belonging to several discrete domains. Key amongst these are:

Incident Response

Incident Response is the discipline of detecting and managing the response to security events. This can include data logging, scans, malware analysis and other tasks as well as the procedure of coordinating systems and staff in order to achieve the best result.

Digital Forensics

Digital Forensics is the practice of investigating devices that have undergone some kind of criminal influence, intrusion, damage or other incident resulting in the Confidentiality, Integrity or Availability of data. In the context of our project, this could include things such as brute force authentication attempts, file theft or manipulation or malicious registry keys. Forensic readiness is then the practice of being able to detect and log such events, as well as taking steps to mitigate or restore damage.

Networking

Networks are connected computers that exchange data. This can include a connection to the internet or simply a local network. In the case of our project, we aim to replicate the environment of a small business by connecting several computers of varying operating systems and states within a virtual network.

## Application Context

The system is to be deployed on critical infrastructure. It should be scalable to a large network, and this can be simulated through virtual machines. It should also be suitable for small businesses, and interpretable by a layman. The primary goal is detection and analysis while maintaining forensic readiness.

In order to best emulate these requirements, the project is being developed on a Windows server within QUT’s network. This Windows server contains a virtual machine acting as the project server. Access to this server requires use of Cisco AnyConnect to authenticate through QUT firewalls using Windows Remote Desktop. The virtual machine, provided it is running, is then accessible through VirtualBox or SSH clients such as PuTTY. Several hundred ports are then open for use by the components for exchange of data with client machines, which will include both virtual and real computers.

The inputs for the project will consist of data from these client machines. Since there is no specific threat in the scenario, we will emulate several common threats such as brute force authentication, DDoS, access to dangerous websites, etc. through means of log manipulation. The goal is then to have these logged events be detected by the system, and then represent these “incidents” as visual elements.

# Project Scope and Plans

Early development of the project has focused on laying the foundations and doing adequate research. We have explored alternative approaches in both hardware and software. This was achieved through experimenting as well as a formal literature review.

Completion of the installation of the Ubuntu virtual machine and the core component GRR were also prioritised in order to help further our working knowledge of the environment, and also have the opportunity to assess any roadblocks or potential issues that might arise later in development.

Release timings have focused on iterative feedback, and also taken into consideration external elements such as university holidays, assessment periods, availability of the client etc.

## Project Plan

* Release 1 – 2 weeks into project. Initial research complete. Completed an Understanding of Requirements document, sighted by the client, in order to ensure that all parties understood what the project entailed. High level architecture diagram ([Image 1](file:///C:\Users\Curtis\AppData\Local\Temp\design\Architecture.png)) also complete and sighted.
* Release 2 – Following mid semester break. Formal literature review complete and demo of functioning GRR server to client.
* Release 3 – End of semester. Consolidation and testing of earlier work while focusing on assessment items.
* Release 4 – IFB399 following mid semester break. Completion of incomplete stories.
* Release 5 – IFB399 end of semester. Final release, automated tests & quality assessment.

## User Story Details

The following is a selection of user stories completed that have significantly affected the project plan in some way. The full list of user stories is available in Appendix.

|  |  |  |
| --- | --- | --- |
| US6 | Virtual Network Deployment | 8 (Must) |

The Virtual Network Deployment was allocated to Release 1 and initially completed using Amazon Web Services EC2 which operated as a successful base for early development but was not sufficient for the strain placed as we installed more components.

Development then shifted to a QUT Windows server using VirtualBox when it was made available by the client. The configuration of this server was more difficult due to using a different Ubuntu image. The operating system has no GUI and so development consisted of entering hundreds of commands character by character. For example, long commands with multiple paths such as the one to generate the SSL Certificates (Image 6) for http access to the server,

|  |  |  |
| --- | --- | --- |
| US5 | Kibana Integration | 16 (Must) |

A core user story, Kibana provides the visual interface for the output from the other elements of the project. Kibana integration will be complete in IFB399 due to technical limitations of the development server, including a lack of administrator privileges making certain functions impossible. This has since been corrected and the development team now has administrator privileges on the server.

|  |  |  |
| --- | --- | --- |
| US3 | Industry Engagement | 4 (should) |

|  |  |  |
| --- | --- | --- |
| US2 | Client Feedback | 8 (should) |

|  |  |  |
| --- | --- | --- |
| US15 | Literature Review | 16 (should) |

These stories form our research initiative & inform our understanding of the project scope and domain. We have been in frequent contact with our client, engaged with professionals in Incident Response and performed a formal literature review in order to ensure that the project meets the needs of the client, the practical applications and requirements of industry and fills a gap in the current capabilities of incident response systems as pertains to forensic readiness.

## Feasibility Analysis

The cyber security environment nowadays is complex, and has never been simple. Because attacks evolve every day as attackers become more inventive, it is critical to properly define cyber security and identify what constitutes good cyber security. As the cyber security technology develops, tools and software like anti-various, anti-phishing and SIEM have provided protection for both individuals and enterprises. While most softwares like anti-various software just can handle the attacks in one machine at the same time, this may not efficient enough to handle massive incidents or accidents of enterprise systems, and live forensics and incident response system is designed to handle multiple incidents or attacks from different machines for enterprises.

Cybercrime is increasing on a global level and cybersecurity analytics and operations are becoming more difficult. The current biggest challenge of incident response team is most security operations have developed some form of incident response processes and procedures, but many of these existing approaches are losing in the battle against cyber attackers. In order to keep up with increasingly sophisticated hackers, a comprehensive understanding of cyber-attacks is crucial. While a truly effective response of incident requires a time-consuming level of expertise and it may not viable for some developers. Therefore, the complex cyber environment and the requirements for accurate response are both great challenges.

Live forensics and incident response system can be implemented by combining several tools. The tools that can be used to develop incident response system are multitudinous, with the development and maturity of computer and software technology, the functions of software and tools are becoming more powerful and complicated. In the past several years, as incident response tools have advanced, a single tool with several functions may achieve automation in security controls and processes, and the viable options for incident response automation are various. Choosing appropriate tools in a large scope of tools with different functions to build live forensics and incident response system can greatly increase the efficiency of the system as well as reduce the time spent on manually blocking common attacks. The system of our project will be developed by combining Google Rapid Response and ELK Stack. The scalability provided by GRR extends to other components of an incident response system to enable wider analysis. The Kibana in ELK Stack provides the platform of analytics and visualization to give a better understanding of the data.

# The Artefact

# Structure

The project aims to address the following problem:

Organisations have a need for a live forensic software that can perform the following actions:

* Real time assessment of an ongoing cyber intrusion
* Ability to determine type of intrusion and its target
* Show status of what was compromised (Integrity, Confidentiality or Availability) in the targeted system

This live forensic software needs to be scalable on critical infrastructure and a large network. It’s interface and data should be interpretable by a novice user with the versatility to deploy within a large organisation or a small business. It was found that GRR and ELK Stack fulfil these requirements after research into alternative tools and a literature review. Further changes were made to the hardware it was deployed on.

Initially GRR was tested and deployed within virtual machines on EC2 Amazon Web Services however later it was decided to deploy it on QUT desktops (16 gigabyte RAM, Intel i7 CPU) made available for testing. This was due to GRR and ELK Stack testing outweighing the monetary costs of EC2 Amazon Web Services.

# Technical Description and Process

The client for the project specified a need for an “all-in-one” incident response system that can perform interpretable live forensics. It was found that there were various open-source forensic tools available however each could only perform individual components of an incident response system. The task as it was defined, required choosing the best open-source forensic tools and combining them into one installable tool to fulfil the need for an “all-in-one” incident response system.

The initial step was to perform a literature review where research would be done in what was an ideal incident response system and using the findings to decide which open-source forensic tools could meet the quality and capability that was required.

It was advised by the client to consider the capabilities of GRR and ELK Stack, determining whether there were other tools that provide a better service or meet additional incident response system requirements. The results from the literature review found alternative tools such as CimSweep, Osquery, TheHive and MIG did not provide features that justified the switch over from GRR and ELK Stack. A comparison table (Image 6) of the possible open source forensic tools the project could integrate was created.

Once it was decided GRR and ELK Stack would be used, the hardware to deploy and host these tools required a decision. EC2 Amazon Web Services was initially used as the underlying hardware to host virtual machines to test GRR and ELK Stack. However, the service had monetary demands that couldn’t be justified for a project with aims to be open source. QUT desktop computers reserved for research by the client were used instead to test GRR and ELK Stack.

## Preferred Approach

The approach that we decided on utilises Google Rapid Response (GRR), Elasticsearch, Logstash and Kibana (ELK Stack).

The primary reason we chose GRR is its ability to perform its functions over a network of varying operating systems. The client has versions for each operating system and simply collects the data, while the Ubuntu server handles the processing of the collected data. Thus, the generation, collection and processing of logs from a fleet that might consist of dozens of Windows clients, some Apple laptops and Linux boxes becomes a task that can be handled very simply. GRR also has useful functions that are not directly investigated in this project – for example, a stolen laptop can provide a notification to the server when it is turned on, run a Netstat and potentially be located based on the results.

The GRR system can be accessed through a web browser ([Image 2](file:///C:\Users\Curtis\AppData\Local\Temp\demo\Browser.png)). The user can then perform a number of actions:

* View client PCs [(Image 3)](file:///C:\Users\Curtis\AppData\Local\Temp\demo\ClientList.png) – The user can see a list of clients that have installed the client program, facilitating the transfer of data and commands between it and the server.
* Start new flows ([Image 4](file:///C:\Users\Curtis\AppData\Local\Temp\demo\Flows.png)) – The user can then start a flow on one of these computers, requesting certain information from that computer such as network information, memory, task management, registry keys, etc.
* Results ([Image 5](file:///C:\Users\Curtis\AppData\Local\Temp\demo\NetstatResults.png), [Data 1](file:///C:\Users\Curtis\AppData\Local\Temp\data\results_C.001a9bb007aa32f0_flows_F_C695E38\ExportedNetworkConnection\from_NetworkConnection.yaml)) – The user can download results of flows in a number of formats. For our project, we are focusing on the use of YAML logs for this output. These YAML logs can then be used by other components including Logstash/Elasticsearch.

The reason these functions are important, in combination, is that digital data can be manipulated. Most computer users know how to destroy a browser’s history, for example. A set of logs that reflects the state of a computer in the system at a given time is a key part of forensic readiness. Furthermore, data stored in RAM is inherently short lived and the ability to remotely analyse that provides significant capability to trace and record malicious processes.

The ELK stack then contributes to the liveness and interpretability of the project. Large networks have large amounts of data to protect, more users and more vulnerabilities. Text and log files are not always an effective means to assessing a threat. Kibana provides an interface that allows the user to see and analyse large amounts of data more efficiently. For example, a netstat could be displayed as a heatmap allowing the user to see the location of connections to the server. If a user is not expecting a connection from an external network, or a million requests from Russian IPs, it immediately becomes apparent that the server is likely undergoing a DDoS attack and relevant action can be taken. If resources such as CPU are reaching capacity and causing issues, it can be quickly determined what processes are causing the issue, at what time, on which PCs and load balancing or upgrades can be ordered.

# Appendix

|  |  |
| --- | --- |
| ID | Name |
| Image 1 | Architecture Diagram |
| Image 2 | Browser Connection |
| Image 3 | Client List |
| Image 4 | Flows |
| Image 5 | Netstat Results |
| Image 6 | Comparison Table |
| Data 1 | Netstat Results (YAML) |