Capstone Interim Project Report

Digital Forensics & Incident Response over a Remote Network

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

Table of Contents

[Introduction 3](#_Toc515820744)

[Client 3](#_Toc515820745)

[Domain 3](#_Toc515820746)

[Application Context 4](#_Toc515820747)

[Project Scope and Plans 5](#_Toc515820748)

[Project Plan 5](#_Toc515820749)

[User Story Details 5](#_Toc515820750)

[Technical Description 7](#_Toc515820751)

[Preferred Approach 9](#_Toc515820752)

[Appendix 11](#_Toc515820753)

# 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](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.

## 

## Technical Description

The project aims to unify multiple open source security software products into one centralised incident response system. Key security software products comprising this are:

Google Rapid Response (GRR)

Google Rapid Response, an effective digital forensics tool that can perform forensic tasks such as file and registry searches, memory analysis and overall client device monitoring. It is versatile and compatible with client computers that may have operating systems such as Windows, Linux or Mac.

It allows for an administrator to set up a front-end server that can be accessed via an authenticated web login. The necessary binaries are stored for the various operating systems that the administrator may encounter.

These can be installed on their client’s computers allowing for communication with GRR using a HTTP POST request. From the GUI (Graphical User Interface) within the GRR front-end server, the administrator can initiate flows and hunts that will perform previously stated forensic tasks. The output of these is in the form of security logs such as YAML files.

ELK Stack (Elasticsearch, Logstash and Kibana)

ELK Stack, a platform that combines three security software products into one to provide better capabilities in events that require incident response. ELK Stack achieves this through effective searches and filters provided by Elasticsearch, efficient storage and categorisation of security event logs provided by Logstash and ease of use via visualisation provided by Kibana.

While tools such as GRR will aim to achieve efficiency and usability in its operations, Logstash will still reduce the strain on an incident response system regarding the many security event logs it will receive. Furthermore, GRR lacks the ability to create searches within the security event logs which Elasticsearch allows.

These are automated and the administrator can tell it to monitor for key terms and types within the log files as well produce metrics relating to the logs. Finally, from these results with the use of Kibana the security logs can be visualised into various graphs, maps and tables. This is especially helpful to the laymen due to the complexity of YAML files making it hard to interpret the data within the file.

## 

## 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](demo/Browser.png)). The user can then perform a number of actions:

* View client PCs [(Image 3)](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](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](demo/NetstatResults.png), [Data 1](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 |
| Data 1 | Netstat Results (YAML) |
|  |  |