Software Requirements Specification

3 May 2017

MALT: Malicious Login Tracker

Version 1.0

CPSC 5373 Final Submission Packet

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 Purpose

This software requirements specification (SRS) will give a detailed description of the MALT network traffic analytics system. It is intended for both developers and stakeholders, and will serve as the definitive record of the purpose, features, behaviors, and interfaces of the software system. This document will provide an agreed-upon basis for the creation of the system between the client and developers. Changes to the proposed functionality of the system during development by either the client or the developers will result in update to this SRS.

1.2 Scope

This software system, MALT, will be a network traffic analytics web app for UA Little Rock Information Security Services. MALT will alleviate the current manual data-processing workflow for visualization of suspicious login data. Access to our automated data storage, processing, and visualization tool will increase the efficiency of analytics tasks.

Specifically, our solution will extract data from human-readable, Google-generated malicious login alert emails, save the data in a database, process the data, and visualize appropriate metrics with an online dashboard. Our proposed system supersedes current commercial products as it integrates all the steps of the data analysis workflow, while avoiding the need to directly write code to gain understanding of the data. Upon software delivery, UA Little Rock Information Security Services will possess a scalable, automated analytics system, deployed to the school network, which can be easily modified by later developers.

1.3 Glossary

Term	**Definition**
Administrator	Person responsible for maintaining MALT and its data storage upon deployment of the web application.
Analyst	Person using MALT to investigate current or historical potentially malicious login attempts to the UA Little Rock network
App	The web application portion of MALT. This component is responsible for accessing and visualizing MALT's data.
Dashboard	The Analyst interface component of MALT. The dashboard displays visualizations of the malicious login alert data in a

	JavaScript-enabled web browser.
DBMS	Database Management System, such as MySQL [1]
Gmail	Gmail™ email service provided by Google, Inc.
MALT	The proposed software system for analyzing UA Little Rock network traffic
Software Requirements Specification (SRS)	A document which describes the complete functionality of a proposed software system for both developers and stakeholders
Stakeholder	Any person with an interest in the system who is not a developer. For example, a user, client, or administrator.
UML	Unified Modeling Language v2.x
UpdateRecords	The data-gathering portion of MALT. This component is responsible for accessing data contained in Gmail messages, processing the data, and storing in with a DBMS for access by App.

1.4 References

[1] MySQL™ Oracle Corporation <https://dev.mysql.com>

[2] Unified Modeling Language™ (UML®) <http://www.omg.org/spec/UML/Current>

[3] Leaflet JavaScript Library <http://leafletjs.com/>

[4] Plotly API Libraries <https://plot.ly/python>

[5] Python Web Server Gateway Interface v1.0.1 <https://www.python.org/dev/peps/pep-3333/>

[6] MySQL™ 5.7 Reference Manual. Chapter 12 Data Types <https://dev.mysql.com/doc/refman/5.7/en/data-types.html>

1.5 Document Overview

This SRS has three main sections: Introduction, Overall Description, and Requirements Specification. The Overall Description section is intended to illustrate the functionality and potential uses of MALT. It provides the necessary context for understanding of the technical requirements in the third section, Requirements Specification. The Requirements Specification section describes MALT's technical specifications fully. It is intended to assist developers in creation of the system.

2 Overall Description

This Overall Description section of the SRS will describe the MALT system and its interfaces with other systems using UML [2].

2.1 System Environment

The MALT system is composed of two components which act independently to render a webpage with data visualizations (referred to as the dashboard hereafter). The system interacts with four actors: an Analyst, an Administrator, a Gmail inbox repository, and a database.

The Gmail inbox is the source of malicious login alert data and serves as the source-of-truth for MALT. The Update Records component of the MALT system is executed by the external operating system with a given frequency to gather malicious login alert data from the Gmail inbox and store it to MALT's database. The database serves as the working repository of data for the App component of MALT. App queries the database and generates visualizations which are rendered on the dashboard.

2.2 Functional Requirements Overview

This section describes the functional requirements for all users of the system. The Analyst is the most frequent user of the system, while the Administrator has maintenance responsibility. The Gmail inbox and database are only passive actors; thus, they have no associated use cases.

The section is separated into Structural, Object, and Dynamic models. Each model has a UML diagram and written description. The diagram serves as the primary model, while the written description elaborates and reiterates the system design.

2.3 Structural Models

2.3.1 Use Case Diagram

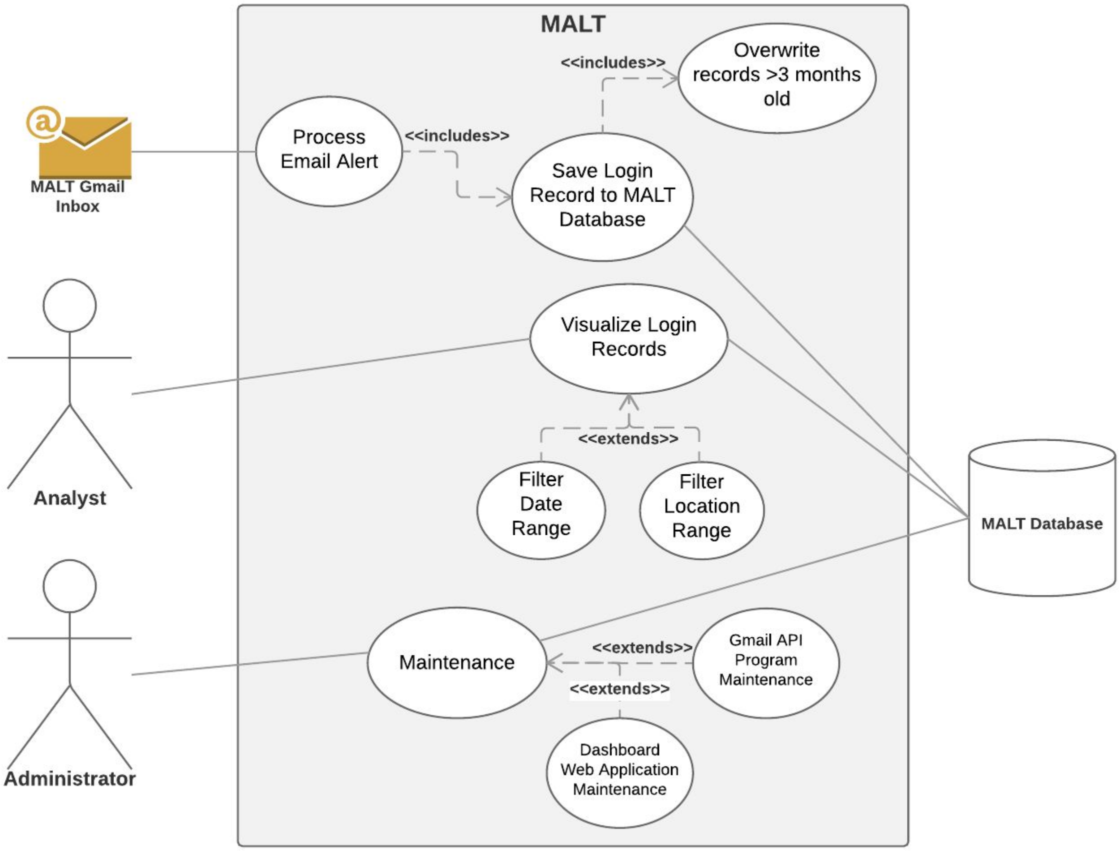
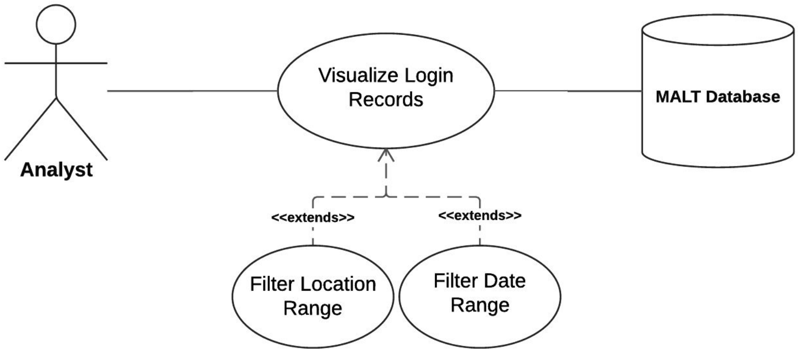


Fig. 1. Use-Case Diagram.

2.3.2 Analyst Use Cases

2.3.2.1 Use Case: Visualize Login Records

Diagram:



Brief Description:
Analyst will visualize the login records, which are loaded from the database. The analyst may optionally filter the records by a date range and/or by a location range.

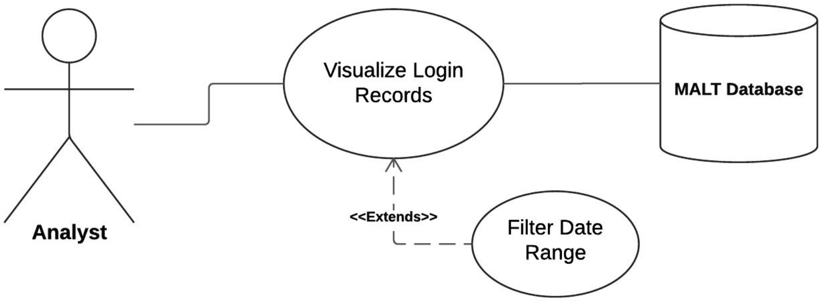
Initial Step-By-Step Description:
Before this use case can be initiated, the analyst must be connected to the online dashboard site.

1. Online dashboard loads all records from the MALT database
2. Analyst views all metrics of the data

Cross-Reference: 3.2.1

2.3.2.2 Use Case: Filter Date Range
This use case extends the **Visualize Login Records** use case.

Diagram:



Brief Description:
Analyst filters data records by date. This use case can be combined with **Filter Location Range** use Case.

Initial Step-By-Step Description:
Before this use case can be initiated, the Analyst has already entered the **Visualize Login**

Records use case.

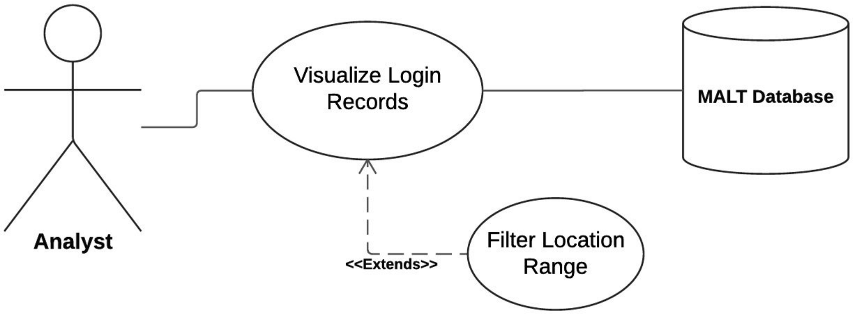
1. Analyst chooses a date range.
2. Dashboard removes data records not meeting the filter criteria.
3. Analyst views subset of data records

Cross-Reference: 3.2.2

2.3.2.3 Use Case: Filter Location Range

This use case extends the **Visualize Login Records** use case.

Diagram:



Brief Description:
Analyst can filter data records by geographical location. This use case can be combined with **Filter Date Range** use case.

Initial Step-By-Step Description:
Before this use case can be initiated, the Analyst has already entered the **Visualize Login Records** use case.

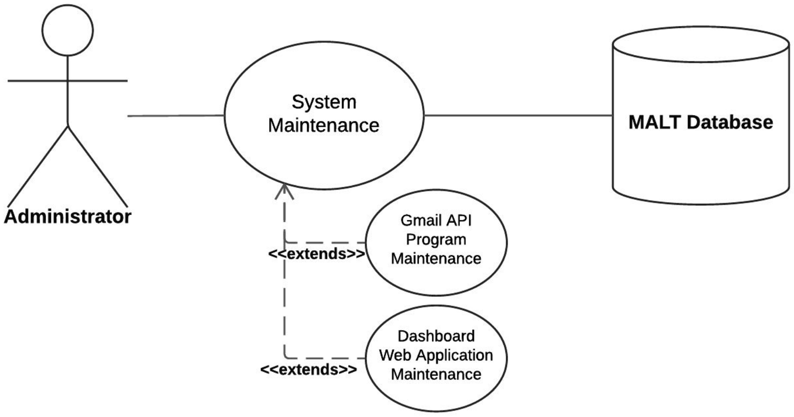
1. Analyst chooses a range for geographical location.
2. Dashboard removes data records not meeting the filter criteria.
3. Analyst views subset of data records

Cross-Reference: 3.2.3

2.3.3 Administrator Use Cases

2.3.3.1 Use Case: Maintenance

Diagram:


Administrator
System Maintenance
MALT Database
<<extends>>
Gmail API Program Maintenance
<<extends>>
Dashboard Web Application Maintenance

Brief Description:
Administrator will maintain the MALT web application system. She will be responsible for maintaining the continuous data pipeline from the system's Gmail inbox to the MALT database, ensuring that the system can properly access the database (e.g. permissions administration), and correcting issues that arise on the dashboard.

Initial Step-By-Step Description:

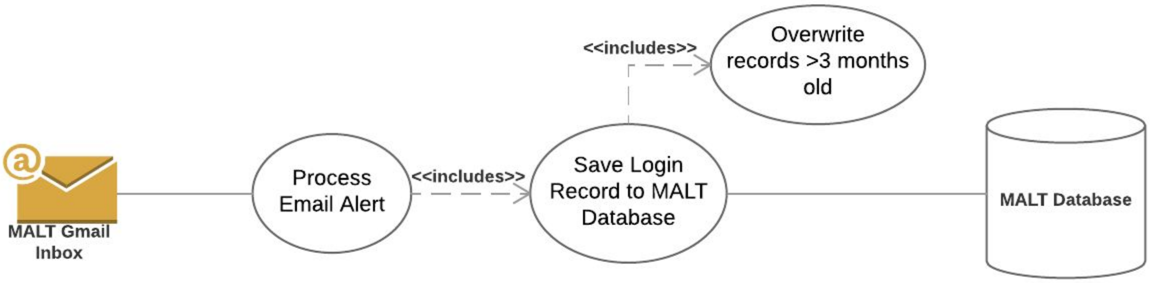
1. Administrator is notified of an issue with MALT
2. Administrator can directly access all the records both in MALT's Gmail inbox and in its database.
3. Administrator has read/write permission for the application's executable files and static files in the server environment.
4. Administrator can view data on the dashboard and plan necessary corrective action.

Cross-Reference: 3.2.4

2.3.4 Gmail Inbox Use Cases

2.3.4.1 Use Case: Process Email Alert

Diagram:



Brief Description:
MALT's associated Gmail inbox will interface with the system through the Process Email Alert

use case. MALT will connect to Gmail, retrieve the malicious login alert data from the email messages and save them to the MALT database. While MALT is connected to the database, it will delete records which are more than 3 months old.

Initial Step-By-Step Description:
Before this use case can be initiated, MALT's Gmail account has already received email alerts.

1. MALT connects to Gmail and reads all unread emails.
2. MALT parses the data in each email and marks it as "processed."
3. MALT structures the data as records matching the MALT database schema.
4. MALT opens a connection to it's database and table.
5. All new data records are INSERTed in to the MALT database.
6. The database is instructed to delete records which are more than 3 months old.
7. MALT closes the database connection.

Cross-Reference: 3.2.5

2.4 Object Model

The following diagrams model the logical structure of the MALT system.

2.4.1 Class Diagram

The MALT System Class Diagram (shown below in Fig. 2) describes the relationships among the system objects. App and UpdateRecords are the two main components of the system, which are connected by DBConnector.

UpdateRecords gets EmailRecord from Gmail and sends them to DBConnector, which requires credentials stored in the MySQLCredentials class. DBConnector serves as the access point to the database for App.

App generates a dataframe data structure in memory from the records sent through DBConnector, and, from the dataframe, generates charts. These charts are used to create the DashboardWebPage. It is composed of zero or more Charts and one FilterModule. The Charts may be of type Bar, Table, or Map.

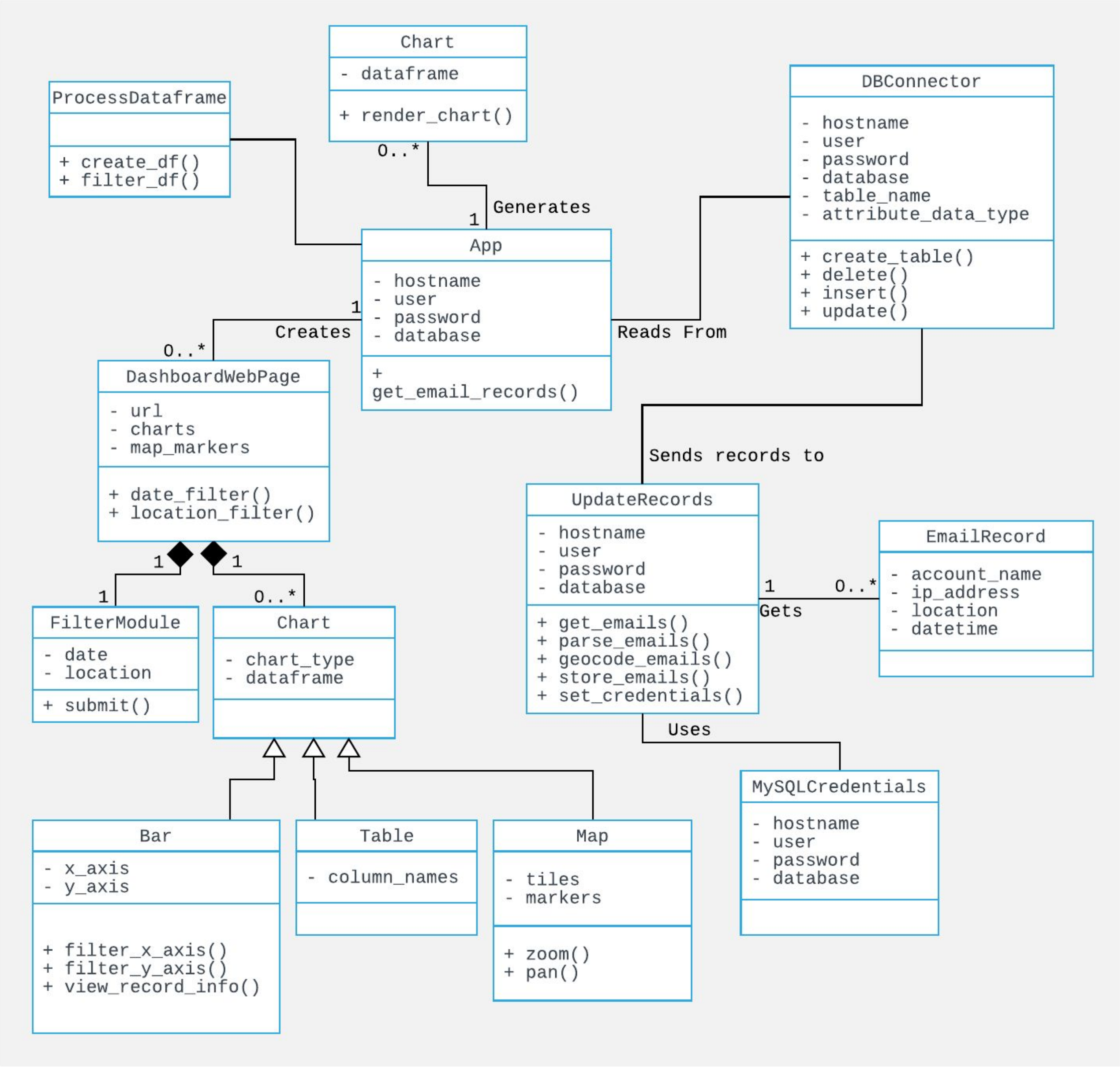


Fig. 2. MALT System Class Diagram

MALT System Class Diagram showing the relationship among the MALT system objects, of which UpdateRecords and App are primary. DBConnector and DashboardWebPage interface with external systems to deliver MALT’s functionality.

2.5 Dynamic Models

2.5.1 Sequence Diagram

The MALT System Sequence Diagram describes the temporal and logical interaction among all components of MALT.

The UpdateRecords portion of the diagram begins in the upper left and is enclosed in a loop. This portion of the system is executed by the underlying operating system on which MALT is installed (see **Detailed Nonfunctional Requirements** below) This component is responsible for storing data from the Gmail inbox to MALT's associated database.

The App component begins with the receipt of an HTTP request by the HTTP server hosting MALT. The request begins the process of generating the dashboard, which can be viewed by an analyst. The beginning and ending points correspond to interfaces to external systems, which in turn may interface with the user.

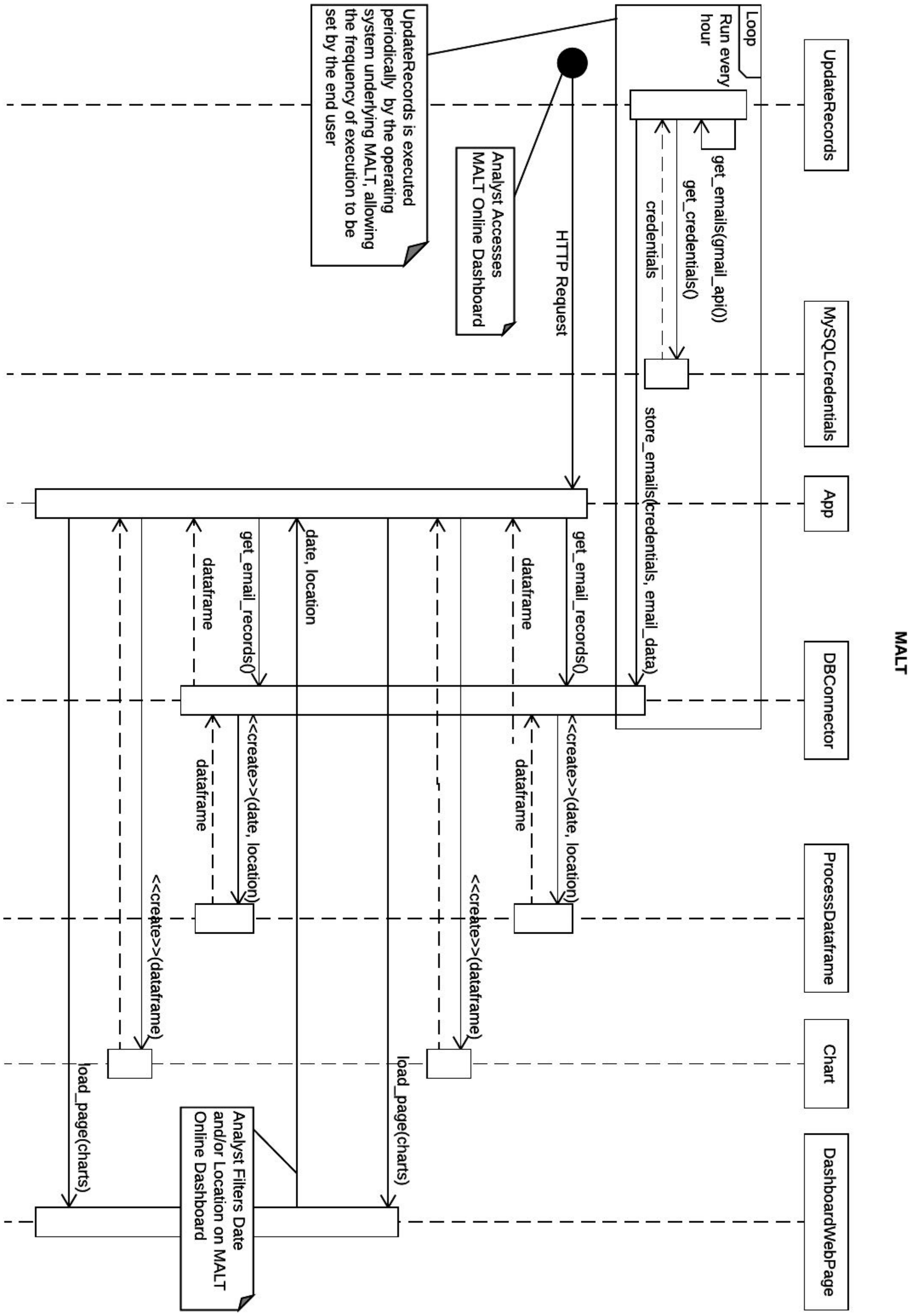

MALT
UpdateRecords
MySQLCredentials
App
DBConnector
ProcessDataframe
Chart
DashboardWebPage
Loop
Run every hour
get_emails(gmail_api())
get_credentials()
credentials
store_emails(credentials, email_data)
UpdateRecords is executed periodically by the operating system underlying MALT, allowing the frequency of execution to be set by the end user
Analyst Accesses MALT Online Dashboard
HTTP Request
get_email_records()
dataframe
<<create>>(date, location)
dataframe
<<create>>(dataframe)
load_page(charts)
date, location
get_email_records()
dataframe
<<create>>(date, location)
dataframe
<<create>>(dataframe)
load_page(charts)
Analyst Filters Date and/or Location on MALT Online Dashboard

Fig. 3. MALT System Sequence Diagram

MALT System Sequence Diagram showing the system execution sequence and logic. UML notes indicate external interfaces which interact with the system during its execution.

2.5.2 Activity Diagrams

This section contains activity diagrams describing the behavior of the two main components of MALT, UpdateRecords and App.

UpdateRecords is executed periodically by its environment (e.g. Linux cron job). It accesses the MALT Gmail inbox using the Google Gmail API, parses and geocodes the malicious login alert data, and stores it as a record in the database. UpdateRecords also marks processed emails with the “Processed” Gmail label for the benefit of the administrator. UpdateRecords accesses the database with the credentials returned by MySQLCredentials, allowing quick updates to the database configuration while keeping the credential data secure and separate. UpdateRecords creates a table in the specified database if it doesn’t already exist and inserts the records gathered from Gmail, one row per email.

App begins with the user (analyst or administrator) accessing the dashboard with a JavaScript-enabled web browser interface. The system then queries the associated database, creates data structures, and generates charts. The dashboard is rendered to the user by a web browser interface and using the Leaflet.js [3] and Plotly.js [4] JavaScript libraries.

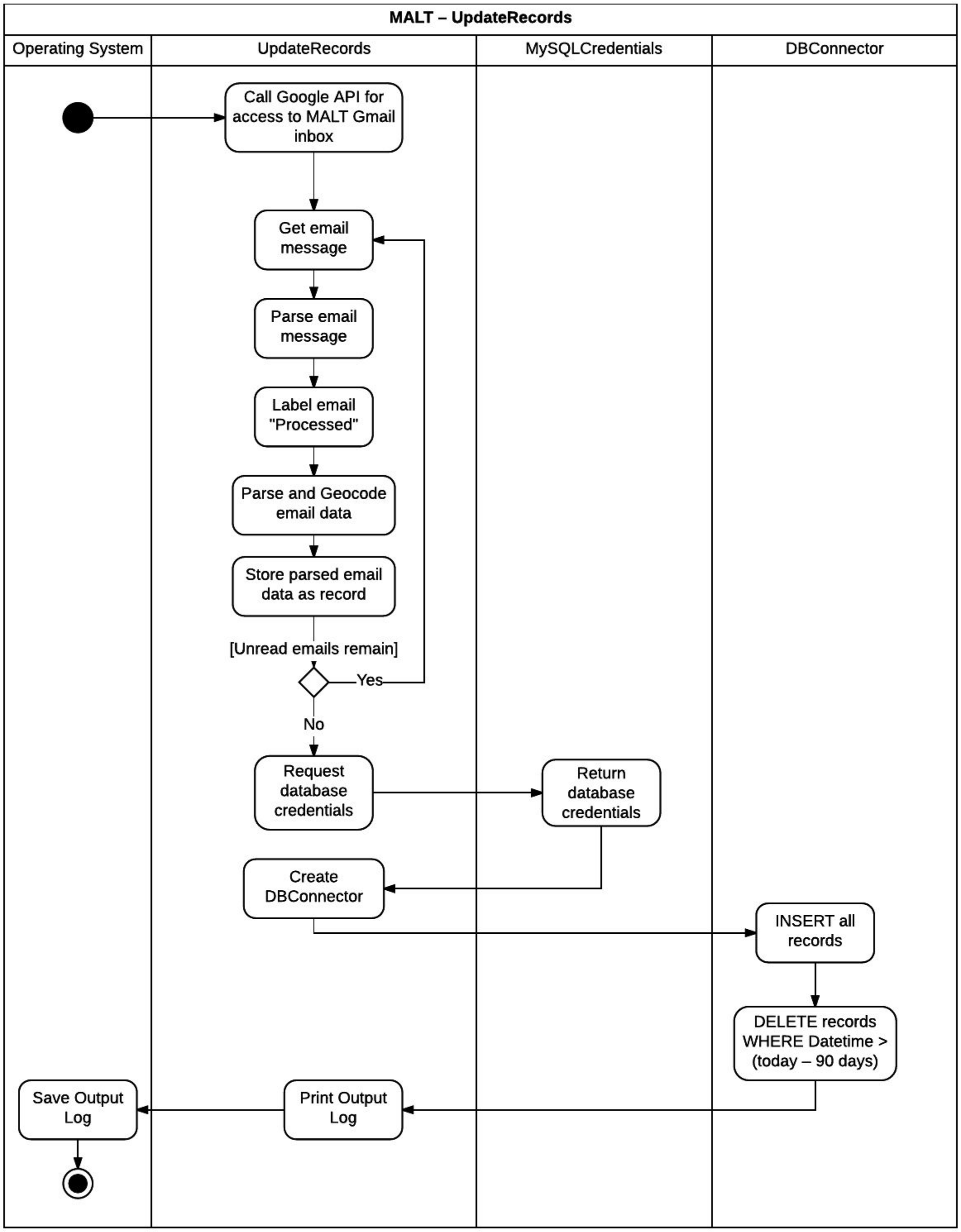


Fig. 4. UpdateRecords Activity Diagram

UpdateRecords Activity Diagram showing the system execution of the UpdateRecords component of the MALT system. UpdateRecords interfaces with the Gmail inbox to retrieve data and interfaces with a DBMS to store it.

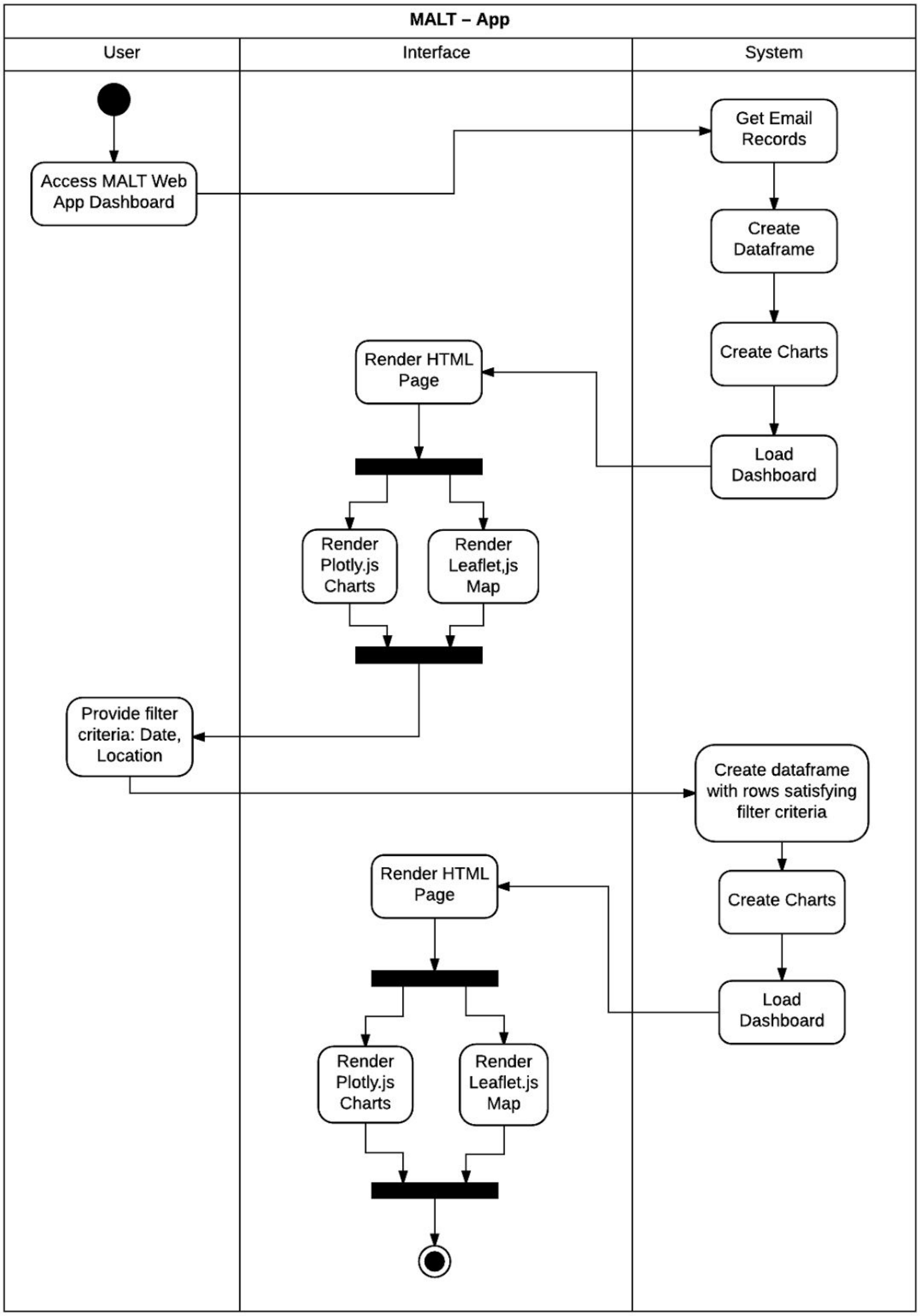


Fig. 5. App Activity Diagram

App Activity Diagram showing the system activity when the dashboard is accessed by an analyst.

2.6 User Characteristics

There are two target users of the MALT system: analyst and administrator.

Analyst
The analyst is expected to be able to navigate a web application, and have an understanding of UA Little Rock's Information Security Services context, in which MALT operates. They will understand UA Little Rock's expected network traffic and be able to identify malicious or other noteworthy login patterns.

Administrator
The administrator will be familiar with the system's design, implementation, and interfaces. She will handle exceptions, either notifying the development team or solving them autonomously through her solution-domain knowledge.

2.7 Nonfunctional Requirements Overview

MALT will be hosted on a UA Little Rock private web server running a Linux operating system. A production HTTP proxy server will serve the MALT dashboard's static files and redirect HTTP requests to a local port where a python WSGI server [5] will host the App component of the MALT system. A DMBS to which MALT has access will be hosted locally as well. A scheduled job will be created to run the UpdateRecords component of MALT.

The web application dashboard interface will be used for all Analyst interaction with MALT. The web server will be protected by an upstream network firewall and the dashboard must only accessible from itself or other servers in the same network group to preserve the security of the data therein.

3 Requirements Specification

3.1 External Interface Requirements

The MALT web app interfaces with two external systems: a web server and a DBMS.

The web server is in turn comprised of two systems: A WSGI server will host the App component of MALT, and a scalable HTTP proxy server will serve static files and redirect HTTP requests to the WSGI server. The system is designed to protect the WSGI server from excessively large workloads with the HTTP proxy server while still using python's data manipulation features.

A web browser with JavaScript enabled is required to access the dashboard interface.

The DBMS connection allows fast and efficient persistent storage of the malicious login alert data. The DBMS server hosts a database on the Linux server system environment which the MALT system connects to and reads/writes data at runtime. The interface between the App and the DBMS combines persistent storage with manipulation of the data in memory to preserve speed during execution and a small resource footprint while the App is not in use.

3.2 Detailed Functional Requirements

This section contains the detailed functional requirements describing the use cases listed in section 2.3.1

3.2.1 Visualize Login Records

Use Case Name	Visualize Login Records
XRef	Section 2.3.2.1
Trigger	The Analyst accesses the MALT webpage
Precondition	Analyst must be connected to online dashboard site
Basic Path	1. On Malt homepage, the analyst will be given the dashboard of charts, maps, and the table. By utilizing the panel of selection criterias, the analyst can filter the records by time frame, specific location, and the radius of the result. 2. The analyst will be able to view all the data, including location distribution, alert locations map, IP addresses of used today, Time-of-Day distribution, Account distribution, and the record table that has columns of account name, datetime, IP address, city, state, country, latitude, and longitude.
Alternative Paths	1. The analyst wants to optionally filter the records by location range, date range, or both. 2. The analyst click refresh button on the top of the panel. 3. When the criteria is applied, click the refresh button. The results will be displayed on the webpage.
Postcondition	The records information are displayed on the webpage, ready for analysis
Exception Paths	The Reader may go back to the default page abandon the search at any time after apply the optional filtering.
Other	The records are automatically updated and rendered every hour.

3.2.2 Filter Date Range

Use Case Name	Filter Date Range
XRef	Section 2.3.2.2
Trigger	The analyst intends to filter the record results by date
Precondition	Analyst must be on the dashboard website
Basic Path	1. On Malt homepage, the analyst will be given the dashboard of charts, maps, and the table. 2. By utilizing the panel of selection criterias, the analyst can filter the records by time and date frame to see the filtered result. 3. Dashboard removes data records that are not meeting the filter criteria 3. The analyst will be able to view all the data based on the criteria applied. Visualization including location distribution, alert locations map, IP addresses of used today, Time-of-Day distribution, Account distribution, and the record table that has columns of account name, datetime, IP address, city, state, country, latitude, and longitude.
Postcondition	Data records are shown based on the date range that analyst selects
Exception Paths	The analyst may goes back to the default records at any time.
Other	This use case can be combined with filter location range use case

3.2.3 Filter Location Range

Use Case Name	Filter Location Range
XRef	Section 2.3.2.3
Trigger	The analyst intends to filter the records result b geographical location.
Precondition	Analyst must be on dashboard website
Basic Path	1. On Malt homepage, the analyst will be given the dashboard of charts, maps, and the table. 2. By utilizing the radius(km), latitude, and longitude selection on the panel, the analyst can filter the records by geographic location to see the filtered result. 3. Dashboard removes data records that are not meeting the filter criteria

	3. The analyst will be able to view all the data based on the criteria applied. Visualization including location distribution, alert locations map, IP addresses of used today, Time-of-Day distribution, Account distribution, and the record table that has columns of account name, datetime, IP address, city, state, country, latitude, and longitude.
Postcondition	Data records are shown based on the location range that analyst selects
Exception Paths	The analyst may goes back to the default records at any time.
Other	This use case can be combined with Filter Date Range use case

3.2.4 Maintenance

Use Case Name	Maintenance
XRef	Section 2.3.3.1
Trigger	System needs to be maintained
Precondition	The administrator has the proper credentials to access the system.
Basic Path	1. Administrator is notified of an issue with MALT 2. Administrator gains the read/write permission for the application's executable files and static file in the server environment. 3. Administrator views data on the dashboard and check the email that is linked to the system and then plans the actions needed.
Postcondition	Action is taken. The system gets re-installed and runs.
Other	Additional information regards to source code could be found at github.com/andrew-pyle/malt/tree/master/malt

3.2.5 Process Email Alert

Use Case Name	Maintenance
XRef	Section 2.3.4.1
Trigger	Operating system on which MALT is installed executes the program with a set frequency.
Precondition	MALT has an associated Gmail inbox and access permissions to read and modify.
Basic Path	1. MALT connects to Gmail and reads all unread emails.

	2. MALT parses the data in each email and marks it as “processed.” 3. MALT structures the data as records matching the MALT database schema. 4. MALT opens a connection to it’s database and table. 5. All new data records are INSERTed in to the MALT database. 6. The database is instructed to delete records which are more than 3 months old. 7. MALT closes the database connection.
Postcondition	New malicious login alert data is stored in the database and records more than 3 months old are removed from the database.
Other	Output of the program should be saved in an accessible log on the operating system underlying MALT.

3.3 Detailed Nonfunctional Requirements

3.3.1 Web Server Environment

The MALT system will be installed on a UA Little Rock private web server running a Linux operating system. The MALT system will be installed with its own user account to be used by the Administrator. The account must have the necessary permissions to run scheduled cron jobs and start/stop system-wide processes.

The UpdateRecords component of MALT must be scheduled to run periodically, as determined by the Administrator. The Administrator user account must be authorized by the Google API to access the MALT Gmail inbox. A passphrase file is included in the installation.

The App component of MALT is served to a local port by a WSGI server. A production HTTP server will be configured as a proxy server to that local port and to serve the dashboard’s static files.

A DBMS will be installed on the Linux web server and will provide an account for MALT to access its database and table of malicious login alert records.

3.3.2 Logical Structure of MALT Data

MALT is a dashboard for visualization of malicious login attempt records for the UA Little Rock network. The attributes of each record which comes from Google are shown below in Fig. 6.

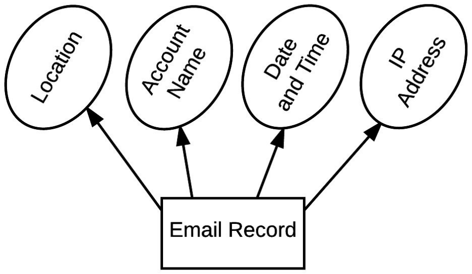


Fig. 6. Gmail Record Attributes

Table 1. Gmail Malicious Login Alert Data Entity

Data Item	Type	Description	Comment
Account Name	Text	UA Little Rock Active Directory account name	
IP Address	Text	IP address as reported by Google	IPv4 and IPv6 intermingled
Approximate Location	Text	**Format:** Neighborhood, City, State, Country	Not all records possess all fields of the Location Item
Date and Time	Text	**Format**: Tuesday, May 2, 2017 at 12:16:12 PM Central Daylight Time	

MALT processes the data and augments it to achieve the attributes shown below in Fig. 7 through geocoding and parsing the data present in the original Gmail record. This augmented record structure allows for visualizations with greater relevance and finer granularity. All data types in Table 2 are MySQL defined types [6].

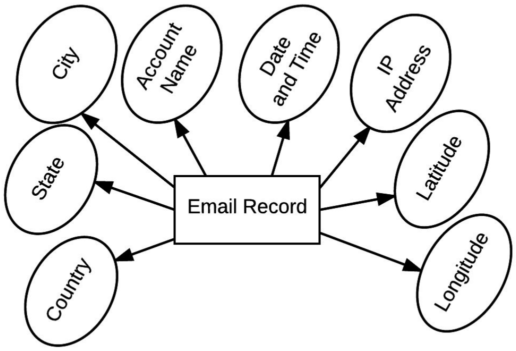


Fig. 7. MALT Record Attributes

Table 2. MALT Record Data Entity

Data Item	Type	Description	Comment
Account Name	varchar(15)	UA Little Rock Active Directory account name	
IP Address	varchar(40)	IP address as reported by Google	IPv4 and IPv6 intermingled
Date and Time	Datetime	YYYY-MM-DD HH:MM:SS	
City	varchar(20)		
State	varchar(20)		
Country	varchar(25)		
Latitude	float(20)	Approximate latitude and longitude based on Google-reported location.	Most records are resolved at the city level. Maximum resolution is neighborhood-level.
Longitude	float(20)		

3.3.3 Security

The MALT system resides on a Linux server managed by UA Little Rock Information Security Services. It’s internal firewall allows incoming connections only on port 22 (for SSH) and port 80 (for HTTP). The upstream firewall configuration is managed by Information Security Services and prevents access from the UA Little Rock network and from the Internet. The only way to access the MALT dashboard is with a web browser on MALT system server itself or from another server within the same network level, behind other upstream firewalls (not managed by MALT). The dashboard is served using a modern web browser and the HTTP protocol (port 80).