**TigerScan – Email Security Scanner**

**IMPLEMENTATION PLAN**

Version 0.5

12/18/2016

**VERSION HISTORY**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Version #** | **Implemented**  **By** | **Revision**  **Date** | **Approved**  **By** | **Approval**  **Date** | **Reason** |
| 0.1 | Nick Schillaci | 10/14/2016 | Matt Paule | 10/14/2016 | All features planned in the sprint backlog were completed and no bugs existed in the program. |
| 0.2 | Nick Schillaci | 10/28/2016 | Matt Paule | 10/28/2016 | Functionality successfully improved according to sprint backlog. |
| 0.3 | Nick Schillaci | 11/25/2016 | Matt Paule | 11/25/2016 | Implemented proper algorithm for scoring, and further ironed out database functionality. Also integrated Lucene for file-searching. To-do successfully completed. |
| 0.4 | Nick Schillaci | 12/7/2016 | Matt Paule | 12/7/2016 | Integrated Maven, fixed a large number of bugs from the previous build. Met all current goals. |
| 0.5 | Nick Schillaci | 12/15/2016 | Matt Paule | 12/16/2016 | Extra testing has been provided, ensuring that the program functions exactly as intended. Current build is stable. |

***Notes to the Author***

[This document is a template of a **Project Implementation Plan** document for a project. The template includes instructions to the author, boilerplate text, and fields that should be replaced with the values specific to the project.

* Blue italicized text enclosed in square brackets ([text]) provides instructions to the document author, or describes the intent, assumptions and context for content included in this document.
* Blue italicized text enclosed in angle brackets (<text>) indicates a field that should be replaced with information specific to a particular project.
* Text and tables in black are provided as boilerplate examples of wording and formats that may be used or modified as appropriate to a specific project. These are offered only as suggestions to assist in developing project documents; they are not mandatory formats.

**When using this template, the following steps are recommended:**

1. Replace all text enclosed in angle brackets (e.g., <Project Name>) with the correct field document values. These angle brackets appear in both the body of the document and in headers and footers. To customize fields in Microsoft Word (which display a gray background when selected) select File->Properties->Summary and fill in the appropriate fields within the Summary and Custom tabs.

After clicking OK to close the dialog box, update all fields throughout the document selecting Edit>Select All (or Ctrl-A) and pressing F9. Or you can update each field individually by clicking on it and pressing F9.

These actions must be done separately for any fields contained with the document’s Header and Footer.

1. Modify boilerplate text as appropriate for the specific project.
2. To add any new sections to the document, ensure that the appropriate header and body text styles are maintained. Styles used for the Section Headings are Heading 1, Heading 2 and Heading 3. Style used for boilerplate text is Body Text.
3. To update the Table of Contents, right-click on it and select “Update field” and choose the option - “Update entire table”.
4. Before submission of the first draft of this document, delete this instruction section “Notes to the Author” and all instructions to the author throughout the entire document.

**Table of Contents**

[1 Introduction 6](#_Toc467462910)

[1.1 Purpose 6](#_Toc467462911)

[1.2 System Overview 6](#_Toc467462912)

[1.2.1 System Description 6](#_Toc467462913)

[1.2.2 Assumptions and Constraints 6](#_Toc467462914)

[1.2.3 System Organization 6](#_Toc467462915)

[1.3 Glossary 6](#_Toc467462916)

[2 Management Overview 6](#_Toc467462917)

[2.1 Description of Implementation 6](#_Toc467462918)

[2.2 Points-of-Contact 7](#_Toc467462919)

[2.3 Major Tasks 7](#_Toc467462920)

[2.4 Roll-Out Schedule 8](#_Toc467462921)

[2.5 Security and Privacy 8](#_Toc467462922)

[2.5.1 System Security Features 8](#_Toc467462923)

[2.5.2 Security Set Up During Implementation 9](#_Toc467462924)

[3 Implementation Support 9](#_Toc467462925)

[3.1 Hardware, Software, Facilities, and Materials 9](#_Toc467462926)

[3.1.1 Hardware 9](#_Toc467462927)

[3.1.2 Software 9](#_Toc467462928)

[3.1.3 Facilities 10](#_Toc467462929)

[3.1.4 Materials 10](#_Toc467462930)

[3.2 Documentation 10](#_Toc467462931)

[3.3 Personnel 10](#_Toc467462932)

[3.3.1 Staffing Requirements 10](#_Toc467462933)

[3.3.2 Training of Implementation Staff 11](#_Toc467462934)

[3.4 Outstanding Issues 11](#_Toc467462935)

[3.5 Implementation Impact 11](#_Toc467462936)

[3.6 Performance Monitoring 11](#_Toc467462937)

[3.7 Configuration Management Interface 12](#_Toc467462938)

[4.0 Change Management 12](#_Toc467462939)

[[Discuss change management activities here] 12](#_Toc467462940)

[4.1 User Training 12](#_Toc467462941)

[4.2 Communication Plan 12](#_Toc467462942)

[4.3 Acceptance Criteria 12](#_Toc467462943)

[APPENDIX A: Project Implementation Plan Approval 13](#_Toc467462944)

[APPENDIX B: REFERENCES 14](#_Toc467462945)

[APPENDIX C: KEY TERMS 15](#_Toc467462946)

[APPENDIX D: System Hardware Inventory 16](#_Toc467462947)

[APPENDIX E: System Software Inventory 17](#_Toc467462948)

# Introduction

## 1.1 Purpose

The purpose of the TigerScan email security scanner is to analyze email content for potential secure/confidential terms, flag them and score them accordingly to measure how confidential the email itself is. If a scanned file yields a score above the set threshold, the email should be marked as too confidential to be sent.

## ****1.2 System Overview****

Implementing the TigerScan system means providing an efficient scanner/file analyzer to identify and calculate confidentiality in the contents of an email. In version 0.5, it exists as a runnable JAR file to check for secure content before sending an external email. It features a user authentication feature for administrators to access database settings and options as well as the event log with information about previous scans.

### **1.2.1 System Description**

The TigerScan system currently resides at version 0.5, featuring updated user authentication and updated efficiency and transparency between the user (and/or administrator) and its resulting output. This is, of course, in addition to the complete management of the SQLite database of confidential terms and their associated scores, accurate scoring algorithm and confidentiality identification, thorough text scanning for matching and stem matching via Lucene, and more. The system utilizes these features to create a module that efficiently and properly scans email content and detects confidentiality.

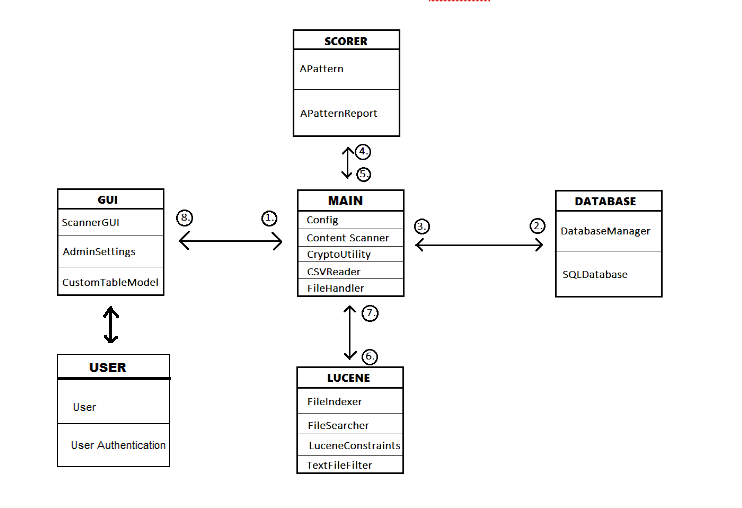
### 1.2.2 Assumptions and Constraints

The implementation process of TigerScan assumes the target operating system is supported by the system. This includes Windows, OSX and Linux systems, due to the versatile nature of Java. Otherwise, there are few constraints involved with using the TigerScan scanner:

* Java 8 installed
* File types supported (TXT, CSV, more planned for future builds)
* Administrator to manage the content of the database

### **1.2.3 System Organization**

The TigerScan system is organized in modules that are defined by Java packages. This keeps each separate function of the program from becoming lost in development or testing. It also paves the way for a clear understanding of how each part of the system interacts with another part, and which modules interact with each other. The following diagram illustrates these interactions:



## ****1.3 Glossary****

*SQLite*: a local database management system embedded into the running client

*Lucene*: an Apache search engine library used for advanced file indexing and searching

*GitHub:* The official GitHub repository housing the program’s source code and other information is located at *https://github.com/nickschillaci1/tigerscan.*

# ****2 Management Overview****

The system must be managed by an administrator account, which is set on first execution once the default administrator username and password is specified. User authentication is stored locally within saved system files inside the “data” folder in the installation directory, which are salted and encrypted in such a way that ensures the security of the information. Through the user authentication system, the administrator has access to all management features, including database management (additions, removals, rename/reclassify, etc.). The administrator must keep the local database of confidential terms and their respective scores up-to-date with company standards and preferences.

## ****2.1 Description of Implementation****

TigerScan can be downloaded from the releases page on the official GitHub, as well as compiled and built from Maven. With Maven installed, simply navigate to the project’s root directory and run the command: “mvn clean package”. The program will be built and a runnable JAR file will be located in the newly created “target” directory.

When deployed to the environment, the administrator account associated with the TigerScan system should be reset (as stated above). This local information is securely stored in the “data/users.cfg” file and can be reset by removing that file from the system.

All necessary information included in the “data” directory is created by the program during first execution and requires no prior setup to ensure functionality. However, the default database will be empty and require population.

## ****2.2 Points-of-Contact****

|  |  |  |
| --- | --- | --- |
| Role | **Name** | **Contact Number** |
| Business Sponsor | Jamie Warner |  |
| Project/Program Manager | Sam Brech |  |
| Government Project Officer | Matt Paule |  |
| System Developer or System Maintainer | Nick Schillaci/Brandon Dixon |  |
| Quality Assurance Manager | Ryan Hudson/Brandon Dixon |  |
| Configuration Management Manager | Zack Flake |  |
| Security Officer | Brandon Dixon |  |
| Database Administrator | Nick Schillaci |  |
| Site Implementation Representative | Nick Schillaci |  |
| Lucene & Scanning Representative | Ryan Hudson |  |
| Interface & System Organization Manager | Zack Flake |  |

**Table 2.2 – Points-of-Contact**

## ****2.3 Major Tasks****

TigerScan can be implemented into a computing environment in two ways, both resulting in a runnable JAR file that will be called upon to perform its tasks. One way of installing the program onto a system is to download the raw Jar file from the official GitHub page and navigate to the desired installation location and place the file within.

The program can also be built and compiled locally using the source code from the GitHub page if the client computer has Maven installed. With Maven installed, simply navigate to the project’s root directory and run the command: “mvn clean package”. The program will be built and a runnable JAR file will be located in the newly created “target” directory. It can be run directly from this location, or moved to a more desirable location according to the system administrator’s preference.

All resources for the program to run properly are included in the runnable file or created by the program on first execution. This also means that if, at any time, a specific installation of the program becomes corrupt or otherwise nonfunctional due to unforeseen errors or external issues, the “data” directory can be deleted and the program will revert back to its original state.

### 2.3.1 Initialization and Population

Once the program has been set up and run, it will not catch any confidential information because it hasn’t been given any confidential information to look for. To do so, an administrator (IT technician or other manager) must click the settings button and enter the default administrator username and password. The default admin credentials are as follows:

* Username: admin
* Password: tiger2admin1676

When the administrator logs in, a new admin username and password must be created. This information is securely saved for each recurring authentication attempt, overriding the default credentials entirely. (This is, of course, if the “data” folder is deleted and the program is reverted back to its default state.)

The administrator can then populate the database of confidential terms and their associated confidential values. The same admin credentials can be used to display the event log, which keeps track of all activity within the program. This is useful for identifying the exact score of a scanned file, when the administrator was created, when certain data was written or removed to or from the database, etc.

### 2.3.2 Training and Materials

No special training is necessary to get TigerScan up and running. It is a simply run program that deals with all the complex calculations and setup by itself and requires only management. As long as an administrator can properly access the database settings and understands the importance of the confidentiality scores on a scale from zero to one-hundred, no extra training is required.

Likewise, no special materials are necessary other than the required Java 8 and Maven (if you plan to build using Maven). The required third-party libraries are included in the provided JAR or downloaded by Maven and included in the compilation as necessary. Therefore, no external materials are necessary for implementation.

## ****2.4 Roll-Out Schedule****

The TigerScan system is provided as-is in working condition, but there may be updates in the future beyond the creation of this Implementation Plan. Any new builds or updates will be pushed to the official GitHub and can be updated from there. Updated installation requirements will be provided if ever necessary.

## ****2.5 Security and Privacy****

Because TigerScan is designed to deal with secure, confidential information and not display such information to common users, ensuring that all sensitive data remains secure is its top priority. It maintains secure database information, user credentials, and scanned file results.

### **2.5.1 System Security Features**

User authentication credentials are salted and encrypted passwords, salted, and encrypted again with username information. Therefore, the password is never actually decrypted, ensuring the information never passes through system memory. Database information and the event log are encrypted as well, providing information to the program for the proper analysis of confidential terms and identifying analysis results.

### **2.5.2 Security Set Up During Implementation**

Apart from local administrator setup as specified above, no further security is necessary during the implementation of the program.

# ****3 Implementation Support****

Beyond supporting the local database by keeping it updated with proper confidential terms, the administrator needs no outside support to keep TigerScan fully functional. All required modules are included and fully able to self-support for the lifetime of the system.

## ****3.1 Hardware, Software, Facilities, and Materials****

### **3.1.1 Hardware**

There is not specific hardware necessary to run TigerScan on a computer system other than the functional computer system itself. The database is saved locally on the device instead of a server, meaning no special hardware or hardware training is required for the proper execution of the program.

### **3.1.2 Software**

TigerScan only requires the support of Oracle Java 8 Runtime Environment to function, due to the fact that the system is built to run in the JRE environment. No other external software is required for the program to function over its lifetime.

### **3.1.3 Facilities**

No physical facilities or staff is required. The software is either downloaded or built in-house and supported by internal software only.

### **3.1.4 Materials**

There are no specific materials necessary to implement or support TigerScan other than the computing devices that the software will be installed on.

## **3.2 Documentation**

Documentation is well-covered in the included Design Document, stating everything necessary for proper understanding of the implementation and execution of the program over its lifetime. Also, Javadocs are available for complete understanding of how each feature in the system is programmed and what each piece of each module actually does. This provides great transparency between the customer and the development team.

## ****3.3 Personnel****

TigerScan only requires an administrator to access its management functions. Training is hardly required because each function is self-explanatory and straightforward. Regular users will not use the management functions, meaning even less understanding of how the program works is needed. Therefore, no special personnel are required.

### **3.3.1 Staffing Requirements**

The administrator is the only required position to support the TigerScan system. As mentioned previously, the administrator is necessary to access management features within the system and keep the system’s confidential terms to search for up-to-date. The administrator position is assumed to be delegated to a manager or other system administrator or IT technician who will be installing the program.

### **3.3.2 Training of Implementation Staff**

Training is not required for the TigerScan security system. Each feature is clearly labelled and functions exactly as expected from a user. The administrator will be set upon system implementation, and so the position requires the default admin credentials to be entered for the new account to be created. Otherwise, no specific instructions are needed for user to properly install or navigate the program.

## 

## **3.4 Outstanding Issues**

While there are no issues with the current build version (0.5), the program is limited to content from .TXT files and importing database terms from .CSV files. While these limitations do render the program less flexible, it remains fully functional and more filetypes can be expected to be supported in the near future.

## ****3.5****Implementation Impact

The implementation of TigerScan will have little impact on an already functioning department or ecosystem. It simply provides an avenue of checking for confidential content within an email or attachment before the file is sent over email to an external source. It affects nothing else on the client device and reaches no other hardware or software other than itself and its host environment.

## ****3.6 Performance Monitoring****

The event log produced from program activity acts as the main performance monitor and is accessible from administrator access on the main interface window. It shows each and every activity that occurs within the system, even that which isn’t shown to regular users. If an unexpected error occurs within the program, it will be noted in the log as such.

## ****3.7 Configuration Management Interface****

New build versions of TigerScan will be made available on the official GitHub page when new features are implemented. There is no planned schedule for future updates, so please keep an eye out on the page and post an issue/request if a feature is desired and should be implemented in the next version.

## 4.0 Change Management

Communication between managers and users is important to ensuring the program is utilized daily as designed. To keep the program lightweight and easy to manage, there is no built-in functionality to update managers in the current build version (0.5).

## 4.1 User Training

User training is not necessary, as stated above. However, if training to some capacity is desired by your organization, users can be prompted on how to add files and scan them with the system, as well as what its results mean. Of course, this information is straightforward enough, needing only two labelled button clicks and a quick glance at the message displayed when scanning your file(s). Therefore, user training really is extraneous.

## 4.2 Communication Plan

Communication between stakeholders and the TigerScan development team will remain a transparent, two-way street. We allow and encourage feedback and feature requests on our official GitHub page and will update the page with future build versions including new updates and improvements. We plan to inform stakeholders of security updates or important upgrades to system stability to ensure the performance and integrity of the program itself will not be compromised in the working environment. Be sure to check the GitHub page for any other future build improvements.

The tables in the following sections provide an overview of the communications for each phase of the project. Each table provides details on communication objectives and key messages at a summary level. Additionally, the tables provide detailed information on specific communications to be made during each phase including the audience, delivery mechanism, message detail, and expected due date/frequency of the message.

|  |  |  |  |
| --- | --- | --- | --- |
| **Communication Objectives** | | Keep stakeholders and system administrators up-to-date on TigerScan integrity and key functionality improvements | |
| **Key Messages** | | Ensuring stability and integrity, necessary updates to improve functionality | |
| **Audience** | **Delivery Mechanism** | **Message Detail** | **Expected Due Date/Frequency** |
| * Administrators (management) * Stakeholders | Email, GitHub release update | * Identify importance of improvement * Specify how the features have been changed * Specify how responsibilities of the system administrator have changed | * As required, not currently scheduled |

## 4.3 Acceptance Criteria

The TigerScan team identifies the project acceptance criteria on the front of implementation to include clearly conveying the implementation and installation instructions within this document as well as all other system documentation and their respective topics. This includes the Design Document and other writing deliverables for the program written throughout the development of the project. An administrator for a properly implemented system will understand the limitations and requirements of the system as described within this document and understand what the program’s management and support features are. This includes support pertaining to future updated versions as released on the official GitHub page.

APPENDIX A: Project Implementation Plan Approval

The undersigned acknowledge that they have reviewed the **TigerScan Implementation Plan** and agree with the information presented within this document. Changes to this **Project Implementation Plan** will be coordinated with, and approved by, the undersigned, or their designated representatives.

|  |  |  |  |
| --- | --- | --- | --- |
| Signature: | Nicholas Schillaci | Date: | 12/18/16 |
| Print Name: | Nicholas Schillaci |  |  |
|  |  |  |  |
| Title: | TigerScan Developer |  |  |
| Role: | Development Team Member |  |  |

APPENDIX B: REFERENCES

[Insert the name, version number, description, and physical location of any documents referenced in this document. Add rows to the table as necessary.]

The following table summarizes the documents referenced in this document.

|  |  |  |
| --- | --- | --- |
| **Document Name** | **Description** | **Location** |
| Design Document | Document describing and detailing the internal design of the system and how its features have been implemented. | The design document is located within the same folder as this document (“doc”). |
| GitHub | The official GitHub page where the source code is located and documentation can be found. | https://github.com/nickschillaci1/tigerscan |

APPENDIX C: KEY TERMS

The following table provides definitions and explanations for terms and acronyms relevant to the content presented within this document.

|  |  |
| --- | --- |
| **Term** | **Definition** |
| JRE | Java Runtime Environment: required for Java applications to function on a device |