Mercy College

Robotic Process Automation (RPA) and Security

A Thesis

By

Fatjet Kosi

Department of Math/Computer Sciences

Submitted in partial fulfillment of the requirements

for the degree of
Master of Science, Cybersecurity

May 1, 2019

Accepted by the Cybersecurity Program

_____, _____
Date, Dean of the Graduate School

The undersigned have examined the thesis entitled '**Robotic Process Automation (RPA) and Security**' presented by **Fatjet Kosi**, a candidate for the degree of **Master of Science in Cybersecurity** and hereby certify that it is worthy of acceptance.

Date	Advisors name
Date	committee member name

ABSTRACT

Robotic Process Automation (RPA) is a software that allows a user to create a Robot (also called a Bot or a task) to automate a repetitive manual process, such as taking data from a spreadsheet and entering it into a website. The ultimate goal of RPA is to take a manual process and automate it without any human intervention. RPA is currently used by many industries, such as telecommunications, information technology, health care, insurance, financial services, human resources, and many others to reduce operational costs, improve performance, and increase work rate output.

This thesis provides three use cases for implementation of RPA technology using the Automation Anywhere application, one of several RPA tools on the market. The first use case involves creating a Bot that retrieves stock market information from Yahoo Finance, such as stock name, last price change in terms of gains or losses, and change in terms of gains or losses in percentage, as a demonstration of how to use RPA to retrieve data from a website and insert the data into a database. The second use case involves retrieving data from the monster.com website, saving the information to an Excel file, and then sending an email with the information at a scheduled increment. The third use case involves automating some website functionalities on mercy.edu.

Finally, despite the usefulness of RPA tools and their growing presence and use in various business contexts, this technology comes with some cybersecurity concerns. This thesis goes on to discuss the cybersecurity implications of RPA technology, including a risk analysis and an analysis of security best practices.

ACKNOWLEDGMENTS

I would like to thank my thesis advisor, Dr. John Yoon, for all his support and advice during this project, throughout my masters program, and during my time as an undergraduate at Mercy College. I would also like to thank Dr. Zhixiong Chen, Dr. Narasimhaswamy Banavara, Dr. Merlinda Drini-Prelvukaj, and many other Mercy faculty members who have taught me so much and helped me develop a strong educational foundation for my career in IT.

I would also like to thank my family for supporting me as I pursued my bachelors and masters degrees. I thank my parents for being there to listen to my ideas and for always encouraging me to do my best and to try new things. I thank my mother- and father-in-law, Linda and Stephen Lauria, for warmly welcoming me into their family and for supporting me as I pursued my education and began my career. Finally, I want to thank my wife, Karen, who has always seen something special in me, even when I did not, and pushed me to be the best version of myself possible.

I would also like to thank my employer, Verizon, for encouraging me to continue my education by providing tuition assistance that enabled me to pursue my graduate degree.

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

CHAPTER I: Introduction

Robotic Process Automation (RPA) is a software that allows a user to create a Robot (also called a Bot or a task) to automate a repetitive manual process. RPA can be used to automate almost any type of process that follows a repetitive pattern, such as taking data from a spreadsheet and entering it into a website, to provide but one example. The ultimate goal of RPA is to take a manual process and automate it without any human intervention. In this project, I will demonstrate how to create a Bot (automate a process), some of the common uses of this technology, and its cybersecurity implications. RPA is a new and emerging technology that is currently being used and deployed by many large corporations worldwide. These companies use RPA programs to automate many of their business processes. I selected RPA as a topic for my Master's project because there is a demand for this line of work and because there is a shortage of RPA developers. RPA jobs are well paying; depending on your location, an automation job could pay more than $100,000 per year, which makes this area an important one to consider for students.

While there are currently more than 20 RPA tools available, three tools dominate the industry. These three tools are easy to use, customizable, scalable, and are more advanced, while the rest offer far less functionality. These three leading RPA tools are Automation Anywhere, BluePrism, and UiPath. These three tools offer essentially the same features, but there have some important differences.

Three Leading RPA Tools

BluePrism: BluePrism does not come with a free trial version, but this tool has a very user-friendly design and can be used by anyone.

UiPath: UiPath offers a free community edition, which can be used to learn and to test the tool out. This tool has a similar design as BluePrism and offers roughly the same capabilities.

Automation Anywhere: Automation Anywhere comes with a 30-day free trial. This tool requires advanced programming skills but, for an advanced developer, this is the best tool because it allows the user to program and define his or her own logic.

If the user is not a strong programmer, he or she would be best served using either BluePrism or UiPath because these two tools are user-friendly and do not require a lot of programming knowledge. However, if a user enjoys programming and is strong in this area, then Automation Anywhere is the best all-around tool and it is the tool I decided to use for this project.

Automation Anywhere Architecture

Automation Anywhere architecture is made of the following elements:

Control room: The control room is a code repository manager. All of the RPA code resides in the control room. When a developer writes new code, he or she commits all of the changes in the control room.

Bot creator: A Bot creator is the Automation Anywhere application software that is used by developers to automate a process. A Bot creator has a suite of tools, which I will discuss at greater length later in this project.

Bot runner: After the automation script has been created by a developer using a Bot creator software platform, that code can be moved into production and deployed in a Bot runner. A Bot runner is a software that is installed and resides in a virtual machine and is designed to run in unattended mode, according to a schedule specified by the user. For example, the Bot runner can be set to start processing customer invoices at 8:00 AM. The Bot wakes up and starts processing invoices the way it was programmed. A Bot runner is only used to run written code and the Bot cannot be used to edit or create a new Bot.

Virtual Machine: A virtual machine is the host environment where the Automation Bot runner lives. A virtual machine is an operating system that behaves like a regular PC. It has all of the PC applications such as a browser, file system, etc.

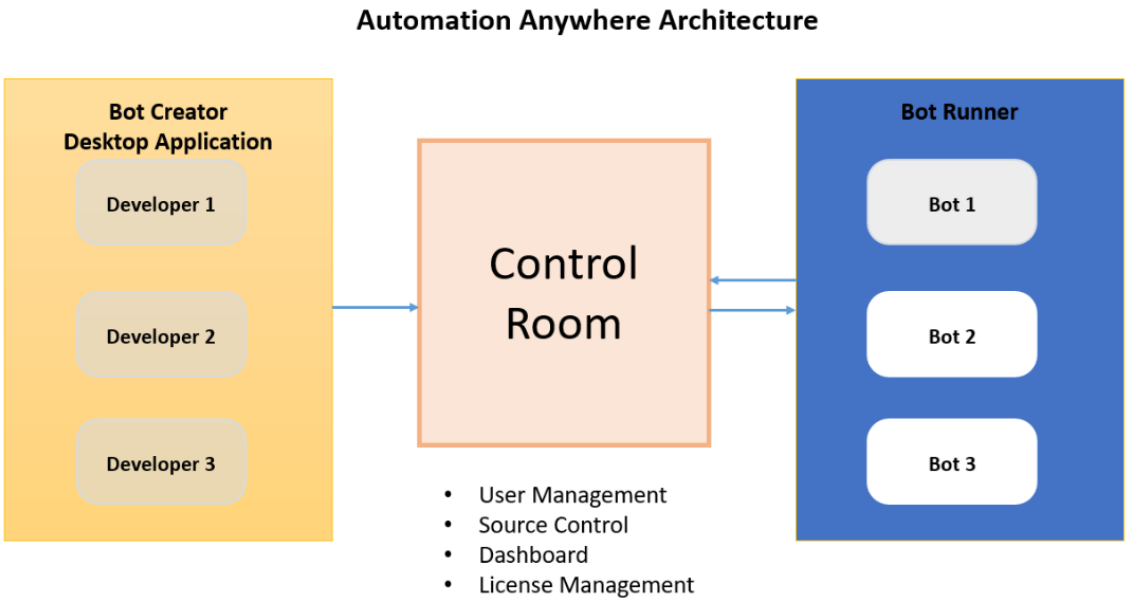

Figure 1: Automation Anywhere Architecture

CHAPTER II: History of and Applicable Uses for RPA

In its early days, RPA was initially used as a tool to test the functionality of a user graphic interface application by performing screen scraping. For example, prior to code being deployed in production, a developer could create a Bot to test the functionality of each screen, gather the results of the test in a report, and finally send the report to the development team to let them know if the application has any issues. In the past few years, RPA tools have been enhanced with new capabilities that go beyond screen scraping functions (Kappagantula, 2018). Some of the new and most common capabilities include integration with databases, APIs, Excel sheets, and other file types; working with image recognition, PDF files, and PGP (Pretty Good Privacy) for encrypting and decrypting files or any information; running external scripts such as Java and VB scripts; and integration with email and XML operations and manipulations.

RPA is currently being used by many industries, such as telecommunications, information technology, health care, insurance, financial services, human resources, and many others ("4 Industries that are Killing it With Robotic Process Automation," 2016). RPA is used by these industries to reduce operational costs, improve performance, and increase work rate output. In the following paragraphs, I detail some examples of how different industries can employ RPA tools in their specific business contexts.

Human resources is one area that can benefit from RPA tools. HR professionals must routinely update customer and employee personal information. Performing this task manually can be time consuming and take time away from employees' more important work functions. Instead, HR professionals can use RPA to automate this process by

creating a Bot to read new personal information from email, a file, a website, a database, or an API and then perform the update against the destination application which could be website(s), database(s), API, etc.

RPA is also useful for data validation. Most data validation is performed within a database, but there are instances in which data will need to be cross checked against a publicly available website or database repository. For example, if an employee needs to validate addresses in a contact database or Rolodex to make sure the addresses exist and are valid, he or she can use RPA to launch the U.S. Postal Service website (or call a U.S. Postal Service API, if available) and compare his or her list of addresses against USPS address validation website/API and determine which addresses are correct and which addresses are incorrect.

RPA is also an excellent tool when it comes to extracting information from many different file types such as Excel, PDF, Word, CSV, etc. For instance, if a process involves extracting information from different types of files, RPA has a suite of tools that can help to completely automate the process. One tool in particular, which can be useful for this kind of task, is Optical Character Recognition (OCR), which can be used to extract selected information from a file, such as first name, last name, phone number, address, and so on.

RPA can be used to prepare complex reports that aggregate data from multiple sources. Some reports are time consuming, but a developer could create a Bot to gather the report information from various systems and then have the report emailed out to the people who

requested it. The Bot can be programmed to deliver the report daily, weekly, monthly, or at another interval, as needed.

RPA also has an email command, which can be used to send mass emails. For instance, if a user needs to send customized invoices to a large number of people, he or she can create a Bot to do that. And, besides sending emails, RPA also offers the ability to read incoming emails and then to respond or make some type of decision based on the content of the emails received.

RPA is also used widely in information technology. In IT, RPA is used to monitor servers and check for system failures. For instance, a Bot can be created to check that a server is up and running and, if there is a problem, send a text message or email to the system administrator. In this case, the system admin will be made aware of the problem and might be able to fix it without too much down time and without having to hear about the problem from multiple people. In IT, RPA is also used to test application functionality and performance, a repetitive task that can be very time consuming.

CHAPTER III: Automation Anywhere Features and Functionality

This section covers Automation Anywhere commands. A developer uses these commands to automate a process. As of 2019, Automation Anywhere has 50 commands, which can be used to program and automate a process. In this section, I will provide a summary of 23 of the most commonly used commands, including the command names and their use, as well as a brief summary for some of the most widely used commands. All of this information is available on the Automation Anywhere user reference guide website (Automation Anywhere, n.d.).

Automation Anywhere Application Commands

Read from CVS/Text Command: This command allows a user to read data from CSV or text file. Usually, this command will have to be implemented with a loop. A loop will be used to extract each record in CSV/Text file.

Excel Command: This is one of the most commonly used commands. This command is used to copy data from and to excel spreadsheets. This command is also used alongside a loop to extract records from columns and rows.

Database Command: This command can be used by almost any database type, such as Microsoft SQL Server and Oracle. This command allows the programmer to connect/disconnect to a database, run SQL query, perform Data Manipulation Activity (DML) such as insert/update/delete, and run stored procedures with or without parameters.

XML Command: This command is used to process XML information created by Web Services and Cloud Computing applications. Some of the main features are Start/End XML Session, Insert Node, Delete Node/Attribute, Update Nodes, Validate XML Document, Get Node(s), Save Session Data, and Execute XPath Function.

Launch Website Command: This command is used to launch any website. The programmer can specify the URL and the website is opened in Internet Explorer or Chrome browsers. This command is usually followed by the Object Cloning command.

Object Cloning Command: Object cloning is one of the most used features of Automation Anywhere. This command is used to automate websites and desktop applications that use Flex, Silverlight, Java, and HTML technologies. The Object Cloning command can be used to automate a website or a desktop application in three different ways: by capturing website/application objects; by capturing website/application coordinates; and by image recognition.

Object Cloning Command > Capturing Website/Application Objects: The best way to automate a website or application is to do so by capturing its objects in the screen and then performing an action, such as right click, left click, double click, set text, get properties, etc. For example, if a user has launched www.mercy.edu and wants to program the Bot to log into Mercy Blackboard, he or she can achieve this by cloning the Mercy Connect hyperlink properties so that the Bot clicks on Blackboard tab hyperlink. After capturing the Mercy Blackboard hyperlink, the following properties were captured:

Property	Value
Technology	HTML
Path	7|2|1|4|1|2|1|2|1
DOMXPath	//header[@id='navbar']/div[1]/div[3]/nav/[1]/ul[2]/ul[1]/li[2]/a[1]
HTML ID	
HTML Tag	A
HTML Innertext	Blackboard
HTML HRef	https://mercy.blackboard.com/
HTML Title	Blackboard
HTML Height	38
HTML Width	108
HTML Tag Index	29
HTML Top	9
HTML Left	808

Table 1: Object Cloning Mercy.Edu

After identifying the technology as HTML, the Bot uses some of the attributes it captured to make the decision and to click on the Mercy Blackboard URL and then, eventually, capture username and password attributes and set those attributes with a user's login credentials. The most important properties captured are DOMXPath, Path, and HTML HRef. DOMXPath is a syntax for finding data elements in a website. There are two types of DOMXPath, Absolute path, and Relative path. Absolute path is more of a direct approach to finding data elements (Guru99, n.d.). In this case, the absolute path of Mercy Blackboard is *//header[@id='navbar']/div[1]/div[3]/nav/[1]/ul[2]/ul[1]/li[2]/a[1]*. In this example, we do not have a relative path, but a relative path usually starts from the

middle of the HTML DOM structure. In the table presented above, the property "Path" is generated by the Automation Anywhere Object Cloning tool. Object cloning gathers the data from the absolute DOMXpath and presents it as the Path in the following format: *7|2|1|4|1|2|1|2|1*. Another important element from the table above is HTML HRef, which is the Blackboard URL (mercy.blackboard.com). Although the Blackboard URL was captured, it is possible to click to Blackboard just by using the absolute DOMXPath alone.

If the website/application that a user is trying to automate presents the data in a table format with columns and rows, then it is possible to capture the data as a table. When capturing data as a table with column and rows, the data can be saved as a CSV file. In addition, it is also possible to get a row count, column count, selecting each row or column by its index, and, if the website is read and write, it is also possible to set columns and rows to any value desired.

Object Cloning Command > Capturing website/application coordinates: In case object cloning properties such as DOMXPath are not available, it is possible to use the "Coordinate" feature to automate. Coordinate works by finding the X and Y position of an item on the screen. For example, if a user is trying to click on a button, he or she can capture the X and Y position of the button on the screen and set the action as click. When the Bot runs, it will click on the button based on its X and Y coordinate position.

Object Cloning Command > Image Recognition: The third feature of the Object Cloning Command is Image Recognition. Sometimes GUI elements of the application are

not available and, if this is the case, a user can use image recognition to capture an item and then perform right click, left click, or double click.

Logfile Command: This command is used to log any errors encountered during runtime. Usually, this command is implemented along with the Error Handling Command. For example, if an error occurred, the Logfile command can capture the error and log the error in a text file or any file format specified.

Error Handling Command: This command is used to capture and handle any unexpected errors that might occur during runtime. This command allows a user to capture and suppress the error and move on to the next item. The command is also used to stop the current task if an error occurs, run a different task, log the error and take a screenshot of the error, and email the error screenshot.

Files/Folders Command: This command is used to copy, rename, delete, zip unzip, etc. files or folders.

If/Else Command: The logic of this command is same as in other programming languages and can be combined with AND/OR.

Loop Command: The concept of a loop in Automation Anywhere is same as in other programming languages. A user can use the Loop Command based on a condition, loop through each file in a folder, each row from a Excel sheet, each row from a SQL database query, each node in an XML dataset, and so on.

Keystrokes Command: This command is used to enter keystrokes, mouse clicks, and mouse moves.

Message Box Command: This command is used to display a message box. This command can hold variables and display data if the variable is assigned.

PGP Command: PGP stands for Pretty Good Privacy. This command is used to encrypt or decrypt files. This command supports symmetric and asymmetric encryptions. Symmetric encryption requires a passphrase to encrypt/decrypt while asymmetric requires a public/private key to encrypt and decrypt. PGP offers eight encryption methods: Idea, TripleDES, CAST5, Blowfish, AES128, AES192, AES256, and Twofish256. PGP also enables users to compress encrypted files using zip, bzip2, or zlib.

Run Script Command: This command is used to run scripts outside Automation Anywhere. This command works with Visual Basic or JavaScript and allows a user to pass parameters to the script and return a value back from the script.

Send Email Command: This command can be programmed to send emails to a list of recipient email addresses in a list, separated by a comma. This command also allows a user to include attachments. Prior to using this command, SMTP hostname and port number (25) must be specified.

String Operation Command: This is a very handy command for dealing with string manipulation and extracting part of a string. Some of the features of this command are Compare, Find, Join, Length, Upper and Lowercase, Replace, Reverse, Split, and Subtract.

Task Command: When automating a lengthy process, it is recommended to break the whole process into small processes so that these processes can be created as small scripts

or tasks and these small pieces of code can be reused when needed as in Object Oriented Programming. The Task Command allows a user to call a smaller task from a larger task, which enables a user to share variables and pass parameters from one task to another. This command also allows users to pause or stop the Bot completely if needed.

Variable Operation Command: This command is used to assign a value to a variable. It can be used to perform addition, subtraction, multiplication, division, and other mathematical functions.

Wait Command: This command is used to force the Bot to wait and not proceed to the next command if the screen of an application takes longer than usual to load.

Delay Command: This command is used to put a delay in seconds or milliseconds when executing back-to-back commands.

Windows Action Command: This command is used to maximize, minimize, resize, close, activate, and get the active window title of a web browser's window or application.

REST Web Services Command: REST, which stands for REpresentational State Transfer, is an architectural style Application Programming Interface (API) that uses HTTP to GET, POST, PUT, and DELETE data. You can use REST Web Services command to interact with applications via API if you have permission to do so.

In addition to the commands mentioned above, Automation Anywhere allows users to create variables, lists, arrays, and random numbers. It also offers some system variables, which can be used to get the date, time, OS file path, and so on.

CHAPTER IV: Automation Samples

After familiarizing myself with Automation Anywhere commands and functionality, I decided to program three automation use cases. My first automation use case retrieves stock market information from Yahoo Finance, such as stock name, last price change in terms of gains or losses, and change in terms percentage. The purpose of this automation is to demonstrate how to use RPA to retrieve data from a website and insert the data into a database and send a text message with Nasdaq stock data.

Gathering Stock Information from Yahoo Finance

First, the bot launches Yahoo Finance URL (finance.yahoo.com) using the Launch Website command (Automation Anywhere code for this project is available in Appendix A). Second, the Bot maximizes the screen using the Window Actions command. Third, since Yahoo Finance displays its data in rows and columns, the Bot captures all of the data as a table and saves it as a CSV file. It is also possible to capture the same data using other methods, such as by column and rows or by indexes.

After the data is saved in the CSV file, the Loop command loops through each record in the CSV file. Automation Anywhere has a built-in variable called *$Filedata Column$*, which is used to retrieve data in a loop from each column. The programmer specifies the variable inside of the loop with a column number in parentheses. For example, retrieving data from Column 2, would be: *$Filedata Column(2)$*. Then, the programmer assigns the *$Filedata Column(2)$* value to a variable/list/array he or she creates.

I then used the Database command to connect to an Oracle database. In the Oracle database I created a table with five columns *(Name, Last_Price, Change,*

Change_Percent, and Date_Added) and the values retrieved from Yahoo Finance were inserted into the database.

Finally, I used an IF statement to filter the Nasdaq stock prices and then used the Email command to send a text message to my mobile phone number. Depending on the cellular provider, it is possible to convert an email to text using these carrier-specific email addresses with a mobile phone number in place of "phonenumber" in each.

- AT&T: *phonenumber*@txt.att.net
- T-Mobile: *phonenumber*@tmomail.net
- Sprint: *phonenumber*@messaging.sprintpcs.com
- Verizon: *phonenumber*@vtext.com or phonenumber@vzwpix.com
- Virgin Mobile: *phonenumber*@vmobl.com

Collecting and Emailing Job Search Data from Monster.com

The second automation use case involves extracting data from Monster.com based on selected search criteria (Automation Anywhere code for this project is available in Appendix B). For this use case, I programmed the Bot to search for Cybersecurity jobs in Dobbs Ferry, NY and to collect the job title and URL of the monster.com job post in an Excel file and then sent an email with the results. This example shows that it is possible retrieve data from a website, create a report using an Excel file, and then send the report via email.

First, the Bot launched monster.com website and sets job title to "Cybersecurity" and location to "Dobbs Ferry" and clicked on the "Find Jobs" button. Next, the Bot created an

Excel sheet and opened the Excel sheet using the Excel command. After that, a loop was declared to loop through each job posting listed on the left panel of the screen.

In order to loop through each job listing, it is important to capture the first job using the Object Cloning command and then to manipulate the DOMXPath by increasing the section value by one. For this, I created a variable called *$vOC_Counter$* and then increased the variable by one for each iteration using the Variable Operation command *($vOC_Counter$ = $vOC_Counter$ +1)*. During the first iteration, job listing one is captured, during second the iteration, job listing two is captured, and so on.

Then, using the Excel command, I set the Excel cells with the values such as Job title and Job URL captured from the website. When gathering data in this fashion, it is a good idea to declare an error handling in the code because there will be some instances in which the DOMXPath does not result in a job listing and therefore there will be nothing to retrieve. I have declared error handling inside the loop to catch any unexpected results and, if the DOMXPath does not match, the error handler captures the exception, suppresses the error, and logs the exception in a log file. This is done to make sure that one exception does not make the entire process fail, but instead suppresses the error and moves to the next step. Finally, after 50 iterations, meaning after capturing 50 cybersecurity job listings, the Bot saved and closed the Excel report and, using the Email command, attached the excel report to an email and sent the email to the desired recipients.

Automating Processes on Mercy.edu

My third and final use case was created around the Mercy College website at mercy.edu (Automation Anywhere code for this project is available in Appendix C). This use case

demonstrates that it is possible to automate some of the functionalities of the Mercy College website. For example, it would be possible to set up a process to notify Cybersecurity M.S. students that registration is open and that they may register for a list of classes. First, the Bot launched the Mercy.edu M.S. Cybersecurity degrees page (mercy.edu/degrees-programs/ms-cybersecurity) and clicked on "Curriculum." Under the "Curriculum" tab, using the Object Cloning Command, the Bot captured all of the M.S. Cybersecurity classes that were listed there and saved them in a variable as one long string. It then split the classes using the String Manipulation command. Next, the Bot launched Mercy Blackboard (mercy.blackboard.com) and logged in using existing credentials (in this instance, my student username and password). For security reasons, I encrypted my credentials in a file using the PGP command.

When the Bot runs, credentials are decrypted and assigned to vUsername and vPassword variables. After logging in, the Bot clicked on My Courses > IASP-600 class > Course Email Message > Create Message; clicked on "To" recipient; selected my name (it can select all or any name); clicked on > arrow to move my name in the recipients side; wrote an email subject; typed all of the classes retrieved under M.S. Cybersecurity curriculum in the body of the email along with a friendly introduction; and sent the email.

CHAPTER V: RPA Security

Since RPA is an emerging technology, identifying RPA security vulnerabilities is not an easy task. However, RPA is not different from other software technologies, which means that the same security principles apply to RPA as apply to other applications. Prior to developing Bots, it is crucial to establish a governance framework for securing Bots. As the first step, it is important for a company to develop a risk assessment strategy to evaluate all of the potential risks that are associated with RPA. The list of cybersecurity risks below is subject to change as new risks are identified (EY, 2018).

RPA Cybersecurity Risks

1) Abuse of privilege access
    a) An attacker may be able to compromise an administrator account used by the Bot. An attacker could use the admin account to gain access to sensitive data.
    b) Before leaving employment, a former employee could program the Bot to delete important data and interrupt the business process.
2) Disclosure of sensitive data
    a) A Bot developer could mistakenly program the Bot to upload highly confidential data, such as credit card information, to a database that is accessed by the public via the web.
    b) A bot developer could use his or her account to steal intellectual property.
3) Security Vulnerabilities
    a) A security vulnerability could exist in the virtual machine environment, which is the environment where the Bot runs. Automation Anywhere Bots are deployed in

Microsoft VM Windows server 2012 R2 and if there is a security vulnerability in the virtual machine environment, the attacker could access the VM remotely and possibly access sensitive data.

b) Bot developer could program the Bot to send/receive sensitive data without encryption. This data is vulnerable and could be exploited by an attacker.

4) Denial of Service

a) Some bad programming practices could make the Bot consume all of the virtual machine system resources and cause the virtual machine to become unresponsive and therefore unable to perform any work.

b) A virtual machine could be affected by unplanned system upgrades or network maintenance, which could result in loss in an outage.

RPA Security Best Practices

RPA risks can be mitigated by implementing RPA best practices (CiGen RPA, 2017). The first security best practice is segregation of duties. This involves making sure all of the RPA user community can only perform the tasks they are assigned to and that they do not have elevated access. Access to code and the control room repository should be limited only to authorized people. For example, developers should be limited to developing or modifying code and uploading new code to the central repository. An RPA administrator's role is to publish new code to production, while a tester's access is limited to running the Bot and documenting Bot behavior (Polania, 2018).

The second security best practice is digital identity and access. Most major hacks happen after elevated credentials have been compromised. To avoid this, it is important to ensure

that the Bot has the least privileged role, the Bot can only perform the job that it was designed to, and that its user role cannot be used to perform other functions. In addition, it is important to use unique login accounts for each application the Bot will access so that, if a system is breached, it will only affect that particular system. Finally, it is important to never automate a process that requires the Bot to login with administrator credentials ("The Power and Potential of Robotic Process Automation and the Security Risks," 2018).

The third security best practice is data encryption. It is recommended to keep Bot credentials encrypted in a password vault and to have the Bot retrieve encrypted credentials via an API call. The Bot decrypts the credentials, passes the decrypted values to variables that can then be used to set a username and password and log into a website or database. Encryption of credentials limits the possibility of having the credentials exposed. Furthermore, if one of the Bot's tasks requires temporarily saving some data in a virtual machine (while waiting for another job to finish), it is important to ensure that the data or file is encrypted while at rest and deleted after the job is finished (WorkFusion, 2017).

Another security best practice is to develop data classification, data retention, data storage, and data location policies (Stephens, 2017). In theory, the Bot can access and store information from and to any medium (Microsoft shared drive, Google Drive, a database, etc.). Depending on the data classification, it is recommended to store information in the most secure repository available. For example, if a Bot deals with highly confidential data, the data should be stored in an encrypted database and not in a local machine. In addition, it is important to develop a data retention policy, which

dictates that the data produced by the Bot is disposed once it is no longer needed. Given the latest General Data Protection Regulation (GDPR) regulations introduced by the European Union, it is imperative that EU customer or personal data is not copied and transported outside of its designated geographic area.

A fifth security best practice is to monitor logs and perform auditing when internal controls are broken. When a Bot is created, it is recommended that the Bot logs all of its activity. Bot activity can be used to monitor abnormal behavior and conduct an audit in case there is a problem.

Finally, it is important to scan the code for vulnerabilities prior to publishing the code in production, which is recommended for any sort of development. There are many code vulnerability check tools available, such as Dynamic Application Security Testing (DAST) and Fortify.

CHAPTER VI: Conclusion

RPA technology is an extremely useful tool that can be applied and implemented in numerous business contexts. When used correctly, these tools can enable workers to automate repetitive manual tasks, thereby saving time, money, and allowing workers to shift their efforts to more lucrative and important work. In the three use case examples provided in this thesis, RPA made it possible to capture data from a website (Yahoo Finance) and save it to a database; to aggregate information from monster.com based on specified search criteria, save the data to an Excel file, and then send an email containing the results to a specified email address; and to automate certain tasks on the mercy.edu website. Yet, even as this technology is useful and extremely versatile in its potential applications, it is not without its risks. As this thesis has explored, there are numerous potential security weaknesses in implementing this software and RPA implementation should also be accompanied by a proper and thorough risk analysis, which establishes policies and procedures that follow RPA security best practices.

REFERENCES

4 Industries that are Killing it With Robotic Process Automation. (2016, March 3). Retrieved April 28, 2019, from https://ayehu.com/4-industries-that-are-killing-it-with-robotic-process-automation/

Automation Anywhere. (n.d.). Commands. Retrieved April 30, 2019, from https://docs.automationanywhere.com/bundle/enterprise-v11.3/page/topics/aae-client/bot-creator/commands/commands.html

CiGen RPA. (2017, November 22). Security Risks in Robotic Process Automation (RPA): How You Can Prevent Them. Retrieved April 28, 2019, from https://medium.com/@cigen_rpa/security-risks-in-robotic-process-automation-rpa-how-you-can-prevent-them-dc892728fc5a

EY. (2018). How do you protect the robots from cyber attack?. Retrieved April 16, 2019, from https://www.ey.com/Publication/vwLUAssets/ey-how-do-you-protect-robots-from-cyber-attack/$FILE/ey-how-do-you-protect-robots-from-cyber-attack.pdf

Guru99. (n.d.). XPath in Selenium WebDriver: Complete Tutorial. Retrieved April 16, 2019, from https://www.guru99.com/xpath-selenium.html

Kappagantula, S. (2018, December 6). Robotic Process Automation | What is Robotic Process Automation? Retrieved April 27, 2019, from https://www.edureka.co/blog/robotic-process-automation/#Emergence%20of%20RPA

Polania, A. (2018, July 19). Security Risks in Robotic Process Automation (RPA) and How to Prevent Them. Retrieved April 15, 2019, from

https://www.elevateconsult.com/security-risks-in-robotic-process-automation-rpa-and-how-to-prevent-them/

The Power and Potential of Robotic Process Automation and the Security Risks. (2018, March 7). Retrieved April 16, 2019, from https://www.cyberark.com/blog/power-potential-robotic-process-automation-security-risks/

Stephens, G. (2017, January 30). Information technology - Three levels of data classification proposed. Retrieved April 25, 2019, from https://www.uab.edu/it/home/about-uab-it/announcements/item/819-three-levels-of-data-classification-proposed

WorkFusion. (2017, August 23). How safe are your bots? The top 3 RPA security risks and most common mistakes enterprises make. Retrieved April 27, 2019, from https://blog.workfusion.com/how-safe-are-your-bots-the-top-3-rpa-security-risks-and-most-common-mistakes-enterprises-make-eec774a87a31

APPENDIX A

Automation Anywhere code for gathering stock market information from Yahoo Finance.

```
Begin Error Handling; Action: Continue; Options: Log to File,  Task Status: Fail
   If Folder Does Not Exist ("$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project") Then
      Create Folder "$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project"
   End If
   If Window Exists ("Major World Indices - Yahoo Finance - Internet Explorer") Then
      Comment: Please enter the conditional commands here.
      Close Window: "Major World Indices - Yahoo Finance - Internet Explorer"
      Delay: (5 sec)
   End If
   If File Exists ("$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project\stock_market.csv") Then
      Comment: Please enter the conditional commands here.
      Delete Files "$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project\stock_market.csv"
   End If
   If Window Does Not Exist ("Major World Indices - Yahoo Finance - Internet Explorer") Then
      Comment: Please enter the conditional commands here.
      Launch Website "https://finance.yahoo.com/world-indices"
      Wait for Window to Open ("Major World Indices - Yahoo Finance - Internet Explorer") (Wait up to 5 seconds - For Window to Open)
      Keystrokes: [CTRL DOWN]0[CTRL UP] in "Major World Indices - Yahoo Finance - Internet Explorer" with delay: 50 ms
      If Window Exists ("Major World Indices - Yahoo Finance - Internet Explorer") Then
         Comment: Please enter the conditional commands here.
         Maximize Window: "Major World Indices - Yahoo Finance - Internet Explorer"
      End If
      Delay: (10 sec)
      Object Cloning: Get Table(Extract Table) "" to csv and store at "$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project\stock_market.csv" from window 'Major World Indices - Yahoo Finance - Internet Explorer'; Source: Window; Play Type: Object
      Object Cloning: Get Total Rows of Table "" from window 'Major World Indices - Yahoo Finance - Internet Explorer'; Assign to variable "$vTableRows$"; Source: Window; Play Type:Object
      Object Cloning: Get Total Columns of Table "" from window 'Major World Indices - Yahoo Finance - Internet Explorer'; Assign to variable "$vTableColumns$"; Source: Window; Play Type:Object
      Variable Operation: $vTableRows$-1 To $vTableRows$
      Message Box: "Table rows: $vTableRows$Table columns: $vTableColumns$"
      Delay: (2 sec)
      PGP: Decrypt Files using Passphrase; Source: "$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project\encrypt.txt.enc"; Destination: "$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project\encrypt.txt"
      Read From Text File: "$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project\encrypt.txt" Delimiter: 'NewLine' Header: 'No' Trim Leading Space: "No" Trim Trailing Space: "No" Session: 'Default'
      Start Loop "Each row in a CSV/Text file of Session: Default"
         If $Counter$ Equal To (=) "3" Then
            Variable Operation: $Filedata Column(1)$ To $vDB_Username$
         Else If $Counter$ Equal To (=) "4" Then
            Variable Operation: $Filedata Column(1)$ To $vDB_Password$
         End If
      End Loop
      Start Loop "Each row in a CSV/Text file of Session: Default"
         If $Counter$ Equal To (=) "3" Then
            Variable Operation: $Filedata Column(1)$ To $vDB_Username$
         Else If $Counter$ Equal To (=) "4" Then
            Variable Operation: $Filedata Column(1)$ To $vDB_Password$
         End If
      End Loop
      Delete Files "$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project\encrypt.txt"
      Read From CSV file: "$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project\stock_market.csv" Delimiter: "Comma" Header: "Yes" Trim Leading Space: "Yes" Trim Trailing Space: "Yes" Session: 'stock_market_csv_file'
      Connect to "Provider=MSDASQL.1;Password=$vDB_Password$;Persist Security Info=True;User ID=$vDB_Username$;Data Source=X" Session:'RPA-DB'
      Start Loop "Each row in a CSV/Text file of Session: stock_market_csv_file"
         Comment: Use $Filedata Column$ variable for each record in File.
         Variable Operation: $Filedata Column(2)$ To $vName$
         Variable Operation: $Filedata Column(3)$ To $vLastPrice$
         Variable Operation: $Filedata Column(4)$ To $vChange$
         Variable Operation: $Filedata Column(5)$ To $vChangeInPercent$
         Message Box: "$vName$: $vLastPrice$, Change:$vChange$, % Change: $vChangeInPercent$"
         Execute SQL Statement: 'INSERT INTO RPA_TEST (NAME, LAST_PRICE, CHANGE, CHANGE_PERCENT, DATE_ADDED)VALUES('$vName$', '$vLastPrice$', '$vChange$', '$vChangeInPercent$', SYSDATE),' Session: 'RPA-DB'
         If $vName$ Equal To (=) "Nasdaq" Then
            Comment: Please enter the conditional commands here.
            Send Email: Subject "MS Project"
         End If
      End Loop
      Disconnect from database Session:'RPA-DB'
   End If
End Error Handling
```

APPENDIX B

Automation Anywhere code for collecting and emailing job search results from Monster.com.

```
If File Exists ("$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project\Indeed_Jobs.xlsx") Then
    Delete Files "$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project\Indeed_Jobs.xlsx"
End If
If File Exists ("$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project\logs.txt") Then
    Delete Files "$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project\logs.txt"
End If
Launch Website "https://www.monster.com/jobs/advanced-search/"
Wait for Window to Open ("Advanced Search - Monster.com - Internet Explorer") (Wait up to 5 seconds - For Window to Open)
Delay: (3 sec)
Object Cloning: Set Text of TextBox "q" in window 'Advanced Search - Monster.com - Internet Explorer'; Value:"Cybersecurity"; Source: Window; Play Type: Object
Object Cloning: Set Text of TextBox "where" in window 'Advanced Search - Monster.com - Internet Explorer'; Value:"Dobbs Ferry, NY"; Source: Window; Play Type: Object
Object Cloning: Click On PushButton "" in window 'Advanced Search - Monster.com - Internet Explorer'; Click Type: Click; Source: Window; Play Type: Object
Delay: (10 sec)
If File Does Not Exist ("$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project\Indeed_Jobs.xlsx") Then
    Copy Files "$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project\Indeed_Jobs_Template.xlsx" to "$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project\Indeed_Jobs.xlsx"
    Excel: Open Spreadsheet "$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project\Indeed_Jobs.xlsx". ActiveSheet: "Default". Contains Header. Session: Output
Else
    Excel: Open Spreadsheet "$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project\Indeed_Jobs.xlsx". ActiveSheet: "Default". Contains Header. Session: Output
End If
Excel: Activate Sheet by index = 1. Session: Output
Loop While $vOC_Counter$ Less Than(<) "50"
    Begin Error Handling; Action: Continue; Options: Log to File, Variable Assignment,  Task Status: Fail
        If $Counter$ Equal To (=) "1" Then
            Continue
        End If
        Variable Operation: $vOC_Counter$+1 To $vOC_Counter$
        Object Cloning: Get Property 'HTML InnerText' of Link "" from window 'Cybersecurity Jobs in Dobbs Ferry, NY. Dobbs Ferry, NY Cybersecurity Jobs. | Monster.com - Internet Explorer'; Assign to variable "$vJob_Title$"; Source: Window; Play Type:Object
        Object Cloning: Get Property 'HTML HRef' of Link "" from window 'Cybersecurity Jobs in Dobbs Ferry, NY. Dobbs Ferry, NY Cybersecurity Jobs. | Monster.com - Internet Explorer'; Assign to variable "$vJob_URL$"; Source: Window; Play Type:Object
        Delay: (50 ms)
        Excel: Set value of Cell "A$Counter$" with "$vJob_Title$". Session: Output
        Excel: Set value of Cell "B$Counter$" with "$vJob_URL$". Session: Output
        Message Box: "$vJob_Title$$vJob_URL$$vOC_Counter$"
        Delay: (1 sec)
        If $vOC_Counter$ Equal To (=) "24" Then
            Object Cloning: Click On Link "" in window 'Cybersecurity Jobs in Dobbs Ferry, NY. Dobbs Ferry, NY Cybersecurity Jobs. | Monster.com - Internet Explorer'; Click Type: Click; Source: Window; Play Type: Object
            Delay: (5 sec)
        End If
    End Error Handling
End Loop
Excel: Save Spreadsheet. Session: Output
Excel: Close Spreadsheet. Session: Output
If File Exists ("$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project\Indeed_Jobs.xlsx") Then
    Send Email: Subject "Mercy College MS IASP 600 Project" with Attachment(s).
End If
```

APPENDIX C

Automation Anywhere code for automating processes on Mercy.edu.

```
1  Begin Error Handling; Action: Continue; Options: Log to File, Variable Assignment,  Task Status: Fail
2      Launch Website "https://www.mercy.edu/degrees-programs/ms-cybersecurity"
3      Wait for Window to Open ("Cybersecurity | Mercy College - Internet Explorer") (Wait up to 5 seconds - For Window to Open)
4      If Window Exists ("Cybersecurity | Mercy College - Internet Explorer")  Then
5          Maximize Window: "Cybersecurity | Mercy College - Internet Explorer"
6      End If
7      Keystrokes: [CTRL DOWN]0[CTRL UP] in "Cybersecurity | Mercy College - Internet Explorer" with delay: 50 ms
8      Delay: (2 sec)
9      Object Cloning: Click On Link "" in window 'Cybersecurity | Mercy College - Internet Explorer'; Click Type: Click; Source: Window; Play Type: Object
10     Delay: (5 sec)
11     Object Cloning: Get Property 'HTML InnerText' of ListView "" from window 'Cybersecurity | Mercy College - Internet Explorer'; Assign to variable "$Clipboard$"; Source: Window; Play Type:Object
12     Message Box: "$Clipboard$"
13     String Operation: Split "$Clipboard$" with delimiter "IASP" and assign output to $my-list-variable$
14     Start Loop "List Variable $my-list-variable$"
15         If $my-list-variable$ Not Equal To (<>) "" Then
16             Message Box: "IASP$my-list-variable$"
17         End If
18     End Loop
19     PGP: Encrypt Files using Passphrase; Source: "$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project\encrypt.txt"; Destination: "$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project\encrypt.txt"
20     PGP: Decrypt Files using Passphrase; Source: "$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project\encrypt.txt.enc"; Destination: "$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project\encrypt.txt"
21     Read From Text File: "$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project\encrypt.txt" Delimiter: 'NewLine' Header: 'No' Trim Leading Space: "No" Trim Trailing Space: "No" Session: 'Default'
22     Start Loop "Each row in a CSV/Text file of Session: Default"
23         If $Counter$ Equal To (=) "1" Then
24             Variable Operation: $Filedata Column(1)$ To $vUsername$
25         Else If $Counter$ Equal To (=) "2" Then
26             Variable Operation: $Filedata Column(1)$ To $vPassword$
27         End If
28     End Loop
29         Delete Files "$AAApplicationPath$\Automation Anywhere\My Docs\MS_Project\encrypt.txt"
30     Object Cloning: Click On Link "" in window 'Cybersecurity | Mercy College - Internet Explorer'; Click Type: Click; Source: Window; Play Type: Object
31     Delay: (5 sec)
32     Object Cloning: Set Text of TextBox "user_id" in window 'Blackboard Learn - Internet Explorer'; Value:"$vUsername$"; Source: Window; Play Type: Object
33     Object Cloning: Set Text of TextBox "password" in window 'Blackboard Learn - Internet Explorer'; Value: "************" ; Source: Windows; Play Type: Object
34     Object Cloning: Click On PushButton "login" in window 'Blackboard Learn - Internet Explorer'; Click Type: Click; Source: Window; Play Type: Object
35     Delay: (4 sec)
36     Object Cloning: Click On Link "" in window 'Welcome, Fatjet – Blackboard Learn - Internet Explorer'; Click Type: Click; Source: Window; Play Type: Object
37     Delay: (2 sec)
38     Object Cloning: Click On Link "" in window 'My Courses – Blackboard Learn - Internet Explorer'; Click Type: Click; Source: Window; Play Type: Object
39     Delay: (2 sec)
40     Object Cloning: Click On Link "" in window 'Homepage – 201910 Spring IASP-600-DLA (CRN-9294.201910) - Internet Explorer'; Click Type: Click; Source: Window; Play Type: Object
41     Delay: (2 sec)
42     Object Cloning: Click On Link "" in window 'Course Messages – 201910 Spring IASP-600-DLA (CRN-... - Internet Explorer'; Click Type: Click; Source: Window; Play Type: Object
43     Delay: (2 sec)
44     Object Cloning: Click On PushButton "to_b" in window 'Compose Message – 201910 Spring IASP-600-DLA (CRN-... - Internet Explorer'; Click Type: Click; Source: Window; Play Type: Object
45     Delay: (2 sec)
46     Object Cloning: Select Item By Text "Fatjet Kosi" of ListView "to_ss" in window 'Compose Message – 201910 Spring IASP-600-DLA (CRN-... - Internet Explorer'; Source: Window; Play Type: Object
47     Delay: (1 sec)
48     Object Cloning: Click On Graphics "" in window 'Compose Message – 201910 Spring IASP-600-DLA (CRN-... - Internet Explorer'; Click Type: Click; Source: Window; Play Type: Object
49     Delay: (1 sec)
50     Object Cloning: Set Text of TextBox "subject_t" in window 'Compose Message – 201910 Spring IASP-600-DLA (CRN-... - Internet Explorer'; Value:"Cybersecurity test"; Source: Window; Play Type: Object
51     Delay: (2 sec)
52     Object Cloning: Click On Link "" in window 'Compose Message – 201910 Spring IASP-600-DLA (CRN-... - Internet Explorer'; Click Type: Click; Source: Window; Play Type: Object
53     Delay: (5 sec)
54     Delay: (1 sec)
55     Start Loop "List Variable $my-list-variable$"
56         If $my-list-variable$ Not Equal To (<>) "" Then
57             Object Cloning: Set Text of TextBox "htmlSource" in window 'HTML code view - Internet Explorer'; Value:"<br>$my-list-variable$"; Source: Window; Play Type: Object
58         End If
59     End Loop
60     Object Cloning: Click On PushButton "insert" in window 'HTML code view - Internet Explorer'; Click Type: Click; Source: Window; Play Type: Object
61     Delay: (3 sec)
62     Object Cloning: Click On PushButton "bottom_Submit" in window 'Compose Message – 201910 Spring IASP-600-DLA (CRN-... - Internet Explorer'; Click Type: Click; Source: Window; Play Type: Object
63 End Error Handling
```

BIBLIOGRAPHY

Fatjet Kosi

Candidate for the Degree of

Master of Science

Thesis: ROBOTIC PROCESS AUTOMATION (RPA) AND SECURITY

Major Field: Cybersecurity

Biographical:

Personal Data:

Education: Bachelor of Science, Computer Information Systems (2013)

Completed the requirements for the Master of Science in Cybersecurity at Mercy College in Dobbs Ferry, NY in May 2019.

ADVISER'S APPROVAL: Dr. John Yoon