

**Department of Computer**

**Science & Engineering**

*A Project Report on*

Windows Based Malware Prediction Using Deep Learning Techniques

*Submitted in partial fulfilment of the requirements for the award of the degree of*

**Bachelor of Engineering in Computer Science & Engineering**

*By*

Devika Anil 1MS15CS040

Aravind P Anil 1MS15CS024

Aatish Kayyath 1MS15CS002

Abishek Padaki 1MS15CS005

*Under the guidance of*

Dr. S. Rajarajeswari

Associate Professor

**M S RAMAIAH INSTITUTE OF TECHNOLOGY**

**(Autonomous Institute, Affiliated to VTU)**

**BANGALORE-560054**

**www.msrit.edu**

May 2019



**Department of Computer**

**Science & Engineering**

# CERTIFICATE

Certified that the project work entitled “Windows Based Malware Prediction Using Deep Learning Techniques” carried out by Devika Anil – 1MS15CS040, Aravind P Anil – 1MS15CS024, Aatish Kayyath – 1MS15CS002 and Abishek Padaki – 1MS15CS005, a bonafide student of M.S.Ramaiah Institute of Technology Bengaluru in partial fulfilment for the award of Bachelor of Engineering in Computer Science and Engineering of the Visvesvaraya Technological University, Belgavi during the year 2018-19. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the department library.

The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree.

**Project Guide** **Head of the Department**

**Dr. S. RAJARAJESWARI Dr. ANITA KANAVALLI**

**External Examiners**

**Name of the Examiners: Signature with Date**

**1.**

**2.**



**Department of Computer**

**Science & Engineering**

# DECLARATION

We, hereby, declare that the entire work embodied in this project report has been carried out by us at M.S.Ramaiah Institute of Technology, Bengaluru, under the supervision of

**Dr. S. RAJARAJESWARI, Associate Professor,** Dept of CSE. This report has not been submitted in part or full for the award of any diploma or degree of this or to any other university.

Signature Signature

Aravind P Anil Aatish Kayyath

1MS15CS024 1MS15CS002

Signature Signature

Devika Anil Abishek Padaki

1MS15CS040 1MS15CS002

# ACKNOWLEDGEMENT

We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project. We would like to express our profound gratitude to the Management and **Dr. N.V.R Naidu** Principal, M.S.R.I.T, Bengaluru for providing us with the opportunity to explore our potential.

We extend our heartful gratitude to our beloved **Dr. Anita Kanavalli**, HOD, Computer Science and Engineering, for constant support and guidance.

We whole heartedly thank our project guide **Dr. S. Rajarajeswari,** for providing us with

the confidence and strength to overcome every obstacle at each step of the project and inspiring us to the best of our potential. We also thank her for her constant guidance, direction and insight during the project.

This work would not have been possible without the guidance and help of several individuals who in one way or another contributed their valuable assistance in preparation and completion of this study.

Finally, we would like to express sincere gratitude to all the teaching and non-teaching faculty of CSE Department, our beloved parents, seniors and my dear friends for their constant support during the course of work.

Aravind P Anil Devika Anil

Aatish Kayyath Abishek Padaki

# ABSTRACT

Malware attack is a very large domain of cybersecurity attacks. Cryptanalysts across the world has been in a long drawn out battle which has intensified in the past decade with malware. With increase in the technological capabilities of computers, there is also a distinct sharp increase in the capabilities of what malware can do and more importantly, how a malware can prevent from being detected. This project aims at developing a model which accurately predicts the probability that Windows operating system will be hit by a malware. It works on an operating systems dataset that has over 7 million recorded operating systems and their various features, generated by combining heartbeat and threat reports.  
  
In order to understand the working of a wide variety of models on such a problem, three different models will be developed and assessed for its accuracy at predicting malware. Three different models under consideration are recurrent neural network, LightGBM technique and lastly a factorization method called XDeepFM.

This approach is assessing vulnerability of the system rather than the attacker. If the attacker is constantly evolving and learning new techniques against the system’s defence, then efforts to defend against certain types of attacks are futile. Hence, predicting an attack in a more generic sense before it has even happened by assessing the system itself is the better alternative. Even though there are variants, a malware always targets a vulnerability or an exploit of the system to attack. If these weak points on the system are found and patched up before an attack happens, we can develop a very secure and malware proof security configuration.

Table of Contents

***Declaration i***

***Acknowledgement ii***

***Abstract iii***

***List of Tables viii***

***List of Equations ix***

***List of Figures x***

**1 INTRODUCTION 1**

1.1 General Introduction 1

1.2 Problem Statement 2

1.3 Objective 2

1.4 Project Deliverables 3

1.5 Current Scope 3

1.6 Future Scope 3

**2 PROJECT ORGANIZATION 5**

2.1 Software Process Models 5

2.2 Roles and Responsibilities 6

**3 LITERATURE SURVEY 8**

3.1 Introduction 8

3.2 Related Work with Citation 8

3.2.1 Malware and Malware Obfuscation Techniques 7

3.2.2 - Malware Detection Techniques 10

3.3 Literature Survey Table of Comparison 16

**4 PROJECT MANAGEMENT PLAN 27**

4.1 Schedule of the Project 27

4.2 Risk Identification 29

**5 SOFTWARE REQUIREMENT SPECIFICATIONS 34**

5.1 Introduction 34

5.1.1 Purpose 34

5.1.2 Intended Audience and Reading Suggestions 34

5.1.3 Product Scope 34

5.2 Overall Description 35

5.2.1 Product Perspective 35

5.2.1.1 The Operating System 36

5.2.1.2 The Prediction Model 36

5.2.2 Product Features 36

5.2.3 Operating Environment 37

5.2.4 Design and Implementation Constraints 37

5.2.5 Assumptions and Dependencies 37

5.3 External Interface Requirements 38

5.3.1 User Interfaces 38

5.3.2 Hardware Interfaces 38

5.3.3 Software Interfaces 38

5.3.3.1 Jupyter Notebook 38

5.3.3.2 Python 3.6 39

5.3.3.3 Google Colab 39

5.3.3.4 Windows OS 39

5.4 Functional Requirements 39

5.4.1 Graphical User Interface Module 39

5.4.1.1 Description 39

5.4.1.2 Input / Output Sequences 39

5.4.1.3 Functional Requirements 39

5.4.2 Light GBM Model 40

5.4.2.1 Description 40

5.4.2.2 Input / Output Sequences 40

5.4.2.3 Functional Requirements 40

5.4.3 XDeepFM Model 40

5.4.3.1 Description 40

5.4.3.2 Input / Output Sequences 40

5.4.3.3 Functional Requirements 41

5.4.4 Recurrent Neural Network 41

5.4.4.1 Description 41

5.4.4.2 Input / Output Sequences 41

5.4.4.3 Functional Requirements 41

5.4.5 Data Conversion 41

5.4.5.1 Description 41

5.4.5.2 Input / Output Sequences 42

5.4.5.3 Functional Requirements 42

5.5 Non-Functional Requirements 42

5.5.1 Safety Requirements 42

5.5.2 Security Requirements 42

5.5.3 Software Quality Attributes 43

5.5.4 Business Rules 43

5.5.5 Performance Requirements 43

**6 DESIGN 44**

6.1 Introduction 44

6.1.1 Purpose 44

6.2 Architecture Design 45

6.3 Detailed Software Design 46

6.3.1 Elastic Compute Cloud and Server Hosting 46

6.3.2 Model Design 47

6.3.3 Use Case Diagram 49

6.3.4 Sequence Diagram 50

6.4 Detailed Hardware Design 50

6.4.1 Central Processing Unit 50

6.4.2 Graphical Processing Unit 51

6.4.2.1 Tesla K80 51

6.4.2.2 Nvidia GTX 1080 51

6.6 Data Flow Diagram 53

6.7 Module Diagram 54

6.8 System Security and Integrity Controls 54

**7 IMPLEMENTATION 56**

7.1 Tools and Technology Introduction 56

7.1.1 Jupyter Notebook 56

7.1.2 Python 3.6 56

7.1.3 Google Colab 56

7.1.4 Flask Server 57

7.1.5 AWS EC2 Instance 57

7.2 Feature Engineering Performed 57

7.2.1 Time Split Validation 57

7.2.2 Feature Encoding 58

7.3 Light Gradient Boosting Method 59

7.3.1 Algorithm 61

7.3.2 Console Output 62

7.4 XDeepFM 63

7.4.1 Factorization Machine 64

7.4.2 Compressed Interaction Network 64

7.4.3 Algorithm 67

7.4.4 Console Output 68

7.5 Recurrent Neural Networks 69

7.5.1 Algorithm 70

7.5.2 Console Output 71

7.6 Pseudo Code (Main Module) 73

7.7 Output 75

**8 TESTING 77**

8.1 Introduction 77

8.2 Testing Tools and Environment 77

8.3 Test cases 78

**9 RESULTS AND PERFORMANCE ANALYSIS 81**

9.1 Result Snapshots 81

9.2 Comparison Tabular Results 83

9.3 Performance Analysis Graph 86

**CONCLUSION AND FUTURE SCOPE 88**

**REFERENCES**  **89**

List of Tables

|  |  |  |
| --- | --- | --- |
| *Table Number* | *Title* | *Page Number* |
| 2.1 | Roles and Responsibilities of team members | 6 |
| 3.1 | Literature survey comparison | 16 |
| 4.1 | Schedule Plan and Timeline of the Project | 27 |
| 4.2 | Risk Identification and mitigation of steps | 29 |
| 6.1 | GPU Comparison between Tesla K80 and GTX 1080 | 52 |
| 8.1 | Unit test cases for various modules of the project | 78 |
| 8.2 | Functional/Integration testing | 79 |
| 9.1 | LightGBM Results | 85 |
| 9.2 | XDeepFM Results | 85 |
| 9.3 | Recurrent Neural Network Results | 86 |

List of Equations

|  |  |  |
| --- | --- | --- |
| *Equation Number* | *Title* | *Page Number* |
| 7.1 | Null hypothesis used for One Hot Encoding | 59 |
| 7.2 | Alternate Hypothesis used for One Hot Encoding | 59 |
| 7.3 | Z- value equation used in One Hot Encoding | 59 |
| 9.1 | True Positive Rate Equation | 83 |
| 9.2 | False Positive Rate Equation | 83 |

List of Figures

|  |  |  |
| --- | --- | --- |
| *Figure Number* | *Title* | *Page Number* |
| 4.1 | Gantt Chart illustrating the entire project schedule | 28 |
| 6.1 | High Level Architecture Design of EC2 Module | 45 |
| 6.2 | EC2 Instance Architecture of the System | 46 |
| 6.3 | Model Development Environment Architecture of the system | 47 |
| 6.4 | Use Case Diagram -Windows malware prediction system | 49 |
| 6.5 | Sequence Diagram - Windows Malware Prediction System | 50 |
| 6.6 | Data Flow Diagram of Operations in the models | 53 |
| 6.7 | Module Diagram of models and Flask server | 54 |
| 7.1 | LGBM Process Flow illustrating the flow of data in the model | 60 |
| 7.2 | Pseudocode of the LightGBM Model | 61 |
| 7.3 | Console Output of LGBM Model showing AUC scores for each fold | 62 |
| 7.4 | XDeepFM Model | 64 |
| 7.5 | Compressed Interaction Network in XDeepFM | 65 |
| 7.6 | Pseudocode that illustrates the working of the xDeepFM model | 67 |
| 7.7 | Console Output of XDeepFM showing Validation Accuracy | 68 |
| 7.8 | Pseudocode that illustrates the working of the recurrent model | 70 |
| 7.9 | Pseudocode that illustrates the working of the final integrated project | 71 |
| 7.10 | Pseudocode that illustrates the working of the final integrated project | 73 |
| 7.11 | Console Output of Main Module on Flask Server | 75 |
| 9.1 | Landing Screen of the Malware Predictor | 81 |
| 9.2 | Scanning screen of the Malware Predictor | 82 |
| 9.3 | Result screen showing vulnerability of the Operating System | 83 |
| 9.4 | ROC-AUC curve | 84 |
| 9.5 | Model Behaviour Comparison Plot | 87 |