Copyright is owned by the Author of the thesis. Permission is given for a copy to be downloaded by an individual for the purpose of research and private study only. The thesis may not be reproduced elsewhere without the permission of the Author.

DETECTION AND CLASSIFICATION OF MALICIOUS NETWORK STREAMS IN HONEYNETS

A thesis presented in partial fulfilment of the requirements for the degree of
Doctor of Philosophy
in
Computer Science
at Massey University, Palmerston North,
New Zealand.

Fahim U.H. Abbasi

2013

Abstract

Variants of malware and exploits are emerging on the global canvas at an ever-increasing rate. There is a need to automate their detection by observing their malicious footprints over network streams. Misuse-based intrusion detection systems alone cannot cope with the dynamic nature of the security threats faced today by organizations globally, nor can anomaly-based systems and models that rely solely on packet header information, without considering the payload or content.

In this thesis we approach intrusion detection as a classification problem and describe a system using exemplar-based learning to correctly classify known classes of malware and their variants, using supervised learning techniques, and detect novel or unknown classes using unsupervised learning techniques. This is facilitated by an exemplar selection algorithm that selects most suitable exemplars and their thresholds for any given class and a novelty detection algorithm and classification algorithm that is capable to detect, learn and classify unknown malicious streams into their respective novel classes. The similarity between malicious network streams is determined by a proposed technique that uses string and information-theoretic metrics to evaluate the relative similarity or level of maliciousness between different categories of malicious network streams. This is measured by quantifying sections of analogous information or entropy between incoming network streams and reference malicious samples. Honeynets are deployed to capture these malicious streams and create labelled datasets. Clustering and classification methods are used to cluster similar groups of streams from the datasets. This technique is then evaluated using a large dataset and the correctness of the classifier is verified by using "area under the receiver operating characteristic curves" (ROC AUC) measures across various string metric-based classifiers. Different clustering algorithms are also compared and evaluated on a large dataset.

The outcomes of this research can be applied to aid existing intrusion detection systems (IDS) to detect and classify known and unknown malicious network streams by utilizing information-theoretic and machine learning based approaches.

Acknowledgements

All praises and thanks to Allah the exalted, the creator and lord of the universe, the source of all knowledge and wisdom. I thank Allah for all His blessings and especially for blessing me with good health, audacity, potential and ability to complete this manuscript.

I wish to express my profound gratitude to my supervisors Prof. Richard Harris, Dr. Giovanni Moretti and Prof. Stephen Marsland for their time, wisdom, patience, keen interest, inspiring guidance and valuable suggestions in planning, conducting and writing this manuscript. Thank you for sharing your wealth of knowledge with me and thank you for teaching me scientific methods and research. I would like to thank all my teachers who taught me in school, college and university in Pakistan and New Zealand. You all played your part that helped me to achieve my goals.

I have no words to express my gratitude to my father Mansoor-ul-Huda Abbasi and my mother Dr. Arsala Mansoor for all they have done for me. Thank you for believing in me and always encouraging me to pursue a Phd. I wish my father was alive and could be with us today to cherish this achievement. Both my parents were awarded a prestigious scholarship for Postgraduate studies overseas (in 1985) and I am glad to be their flag bearer. I would like to thank my siblings, my sister Asra, brother-in-law Irfan, and my brother Ibrahim. Thank you for your wishes and support.

I would like to thank my wife Rooshan, who has always stood firm by my side through good and difficult times during my PhD. Your constant love and support played a major role in completion of my PhD.

I would like to thank my lovely daughters Fatima and Sara. You have provided the perfect distraction that allowed me to keep my sanity and provided a good motivation to finish my thesis.

I would like to thank my fellow Postgrads at Massey University, especially my office mates Graham Jenson and Jevon Wright for the valuable discussions and distractions.

I would like to thank my fellow Pakistani students and the Pakistani Palmerston North community for all their support and help, especially Yousuf, Jawad, Shujjat, Aamir, Jibran, Ibrar and Dr. Zulfiqar Butt. Thank you for all the support, festivities and distractions that made us feel at home.

I would like to thank the Higher Education Commission (HEC) of Pakistan for funding my studies at Massey University through the MS leading to PhD scholarship. You provided a very strong motivation to persue a PhD at Massey University NZ.

I would like to thank Massey University and the School of Engineering and Advanced Technology (SEAT) for giving me the opportunity to do a PhD degree.

I would like to the thank the SEAT staff especially Michele Wagner, Linda Lowe, Dilantha Punchihewa and Gayle Leader for sorting out numerous administrative and financial issues during the tenure of my PhD. and especially the SEAT support staff: Patrick Rynhart, Colin Plaw, Mike Lusby and Nick Look. Your technical expertise and discussions have helped me during the thick of things.

Finally I would like to thank Dr Ian Welch, Dr Peter Komisarczuk, Dr Felix Leder and Dr. Tillmann Werner for their support and advice.

Contents

Abstract **iii**

Acknowledgements **v**

1 Introduction **1**

1.1 Background 1

1.2 Motivation 3

1.3 Network-based Intrusion Detection Research Challenges 7

1.4 Scope of study 9

1.4.1 Collection of malicious network streams for ground truth 10

1.4.2 Detection of malicious network streams and their variants: Supervised Learning 11

1.4.2.1 Similarity measurement between malicious network streams: The use of information-theoretic and string metrics based measures 12

1.4.3 Detection of new or novel malicious network streams: Unsupervised Learning 12

1.4.4 Stream Classification 12

1.5 Research Goals 13

1.6 Contribution of this thesis 13

1.7 Limitations 14

1.8 Thesis Structure and Outline 14

2 Literature Review on Intrusion Detection 17

2.1 Intrusion Detection Systems 17

2.1.1 A Brief History and Review Of IDS 18

2.1.2 A Taxonomy of Intrusion Detection Systems 22

2.1.3 Categorization by Analysis Method 25

2.1.3.1 Misuse-based Detection Systems: 27

2.1.3.2 Anomaly-based Detection Systems: 28

2.1.3.3 Hybrid Intrusion Detection Systems: 30

2.1.4 Categorization by Location in the network: 31

2.1.4.1 Host-based IDS (HIDS) 31

2.1.4.2 Network-based Intrusion Detection Systems or (NIDS) 32

2.1.5 Categorization by Information source 33

2.1.6 Categorization by Detection Methodology: 33

2.1.6.1 Statistical Methods for Intrusion Detection 34

2.1.6.2 Knowledge-based detection 34

2.1.6.3 Machine learning approaches for Intrusion Detection . . 34

2.1.6.4 Information-theoretic approaches for Intrusion Detection 37

2.1.6.5 Worm Detection Systems 38

2.1.7 Categorization by Processing time: 40

2.1.8 Categorization by Architecture: 40

2.1.9 Categorization by Response: 41

2.1.10 Gaps in Intrusion Detection Systems 41

2.2 Classifying Computer Attacks 42

2.2.1 Types of Attacks 42

2.2.2 Why is it important to understand Computer Attacks? 43

2.2.3 A Brief Taxonomy Of Computer Attacks 43

2.3 Data Collection for Intrusion Detection 45

2.3.1 Honeypots and Honeynets 47

2.3.1.1 Honeypots 47

2.3.1.2 Honeynet 48

2.4 Summary 49

3 Architectural Design, Honeynets and Datasets **51**

3.1 Problem Description 51

3.2 Design Overview 52

3.2.1 The Malicious Stream Profiler 52

3.2.2 Malicious Stream Processor 53

3.3 Honeynet Architectures and Deployment Strategies 55

3.3.1 Virtual Honeynet 56

3.3.2 Data capture and analysis tools On the Gateway 57

3.3.3 Data capture tools on the Honeypot 57

3.4 Proposed Honeynet Architecture and Deployment Challenges 58

3.4.1 Problem Identification and Proposed Solution 58

3.4.2 Design Details and Discussion 60

3.4.3 Results and Discussion 62

3.4.4 Honeynet Architecture Summary 64

3.5 Dataset 66

3.6 Summary 68

4 Similarity Metrics **69**

4.1 Problem Description 69

4.2 Methodology 71

4.2.1 Feature Extraction: Packet/Stream Profiles 71

4.2.2 Measuring similarity between packet/stream profiles? 72

4.3 Similarity Metrics 73

4.3.1 String Metric or Edit distance based measures 73

4.3.2 Information-theoretic measures 77

4.3.2.1 Information Theory and Security 77

4.3.2.2 Similarity using Spamsum 78

4.3.2.3 Similarity with Normalized Compression Distance (NCD) 82

4.3.3 Hybrid approach 86

4.4 Experiments 87

4.4.1 Experimental Objectives 88

4.4.2 Experiment 1: Simulation 88

4.4.3 Experiment 2: Mixed Internet and worm traffic 89

4.5 Results 90

4.5.1 Result 1: Simulation Results 90

4.5.2 Validation of Simulation 96

4.5.3 Result 2: Mixed Internet Traffic 96

4.6 Limitation 97

4.7 Comparison with Inverse Compression Ratio 97

4.8 Conclusion 101

5 Clustering and Classification for Intrusion Detection 103

5.1 Problem Description 104

5.2 Background 104

5.2.1 Machine Learning 104

5.2.2 Machine Learning for Intrusion Detection 104

5.3 Evaluation 106

5.3.1 ROC 107

5.4 Methodology and Experiments 108

5.4.1 Information visualization and analysis through Multi-dimensional scaling MDS 108

5.4.2 Experimental Objectives 109

5.4.3 Preparing Test Datasets for experiments 110

5.4.4 Experiment 1: Determination of the best Clustering and Classification algorithm 110

5.4.5 Experiment 2: Determination of the threshold for T-NN 114

5.4.6 Experiment 3: Determining the best Similarity Metric 115

5.5 Results and Discussion: 115

5.5.1 Result 1: The best clustering algorithm for our domain 115

5.5.2 Result 2.1: T-NN with Fixed Threshold Clustering using different Similarity Metrics 115

5.5.3 Result 2.2: T-NN Adaptive vs T-NN Fixed Threshold 120

5.5.4 Threshold Results: Using T-NN with Adaptive threshold 120

5.5.4.1 AUC vs Accuracy vs Zero False Positive (FP). 120

5.5.5 Result 3: Best similarity metric using T-NN with adaptive threshold 123

5.5.6 Comparative Results: Cluster comparisons 123

5.6 Limitations 126

5.7 Conclusions 128

6 Exemplar Learning for Intrusion Detection 129

6.1 Introduction 130

6.2 Problem Description 130

6.3 Methodology and Algorithm Design 131

6.3.1 Exemplar Learning for Intrusion Detection 131

6.3.2 Model Creation using exemplar-based learning 133

6.3.2.1 Algorithm 1: Exemplar Selection (Choosing Exemplars from TR) 133

6.3.3 Algorithm 2: Lifelong Learning (Detecting and adding novel input classes) 134

6.4 Experiments and Validation 137

6.4.1 Validation of the Exemplar Selection Algorithm 137

6.4.1.1 Experiment 1: Exemplar Selection (Supervised) 137

6.4.1.2 Results of Experiment 1 138

6.4.1.3 Discussion 138

6.4.2 Validation of the Novelty Detection Algorithm 139

6.4.2.1 Experiment 2: Novelty Detection (Supervised) 141

6.4.2.2 Results of Experiment 2 142

6.4.2.3 Experiment 3: Novelty Detection (Unsupervised) 142

6.4.2.4 Results of Experiment 3 142

6.4.2.5 Discussion 143

6.5 Comparative work 145

6.5.1 Experiment 4: Instance Selection 145

6.5.1.1 How best to select the exemplars or instances required per class for classification? 145

6.5.2 Experiment 5: Threshold Selection 145

6.5.2.1 How to select the optimal threshold required per instance per class? 145

6.5.3 Comparative Results 146

6.5.4 Comparison with other instance selection algorithm 148

6.5.4.1 Difference between ENN and our proposed algorithm . 148

6.5.4.2 Experiment 6: Comparison with other Instance Selection Algorithms (ENN) 149

6.5.4.3 Results of Experiment 6 150

6.6 Conclusion 150

7 Conclusions and Future Work **151**

7.1 Summary of the Research and Significant findings 151

7.2 Implementation summary 155

7.3 Contribution of this thesis 156

7.4 Future Research and Open Questions 156

Bibliography **159**

A A declaration of previous work **173**

List of Tables

2.1 Proposed IDS Taxonomy 26

3.1 Project Summary: November, 2008 65

4.1 Comparison of MD5 with CTPH 79

4.2 Comparison of Similarity metrics: Levenshtein, Damerau Levenshtein, Hamming, Jaro Distance, Jaro-Winkler, NCD, NCD with Spamsum, Spamsum, Spamsum Reverse using Strings: strA and strB. Scaled such that 0 means perfect match and 1 means complete miss match 88

4.3 Comparison of IDS: Snort vs Similarity metric 91

6.1 Comparison of methods random exemplars vs selected exemplars 138

6.2 Learning with Algorithm 2 142

6.3 Comparison of algorithms using incremental numbers of exemplars . . . 148

6.4 Comparison with ENN 150

List of Figures

1.1 Intruder knowledge vs Attack sophistication (Lipson, 2002) 5

1.2 CERT Incident reports received from 1988-2003 (Hansman and Hunt, 2005) 6

2.1 Time-line of a few notable Intrusion Detection Systems 23

2.2 Taxonomy of Intrusion Detection Systems as per the proposed seven category criteria 24

2.3 A Honeypot comic by XKCD(http://xkcd.com/350/), showing several honeypots connected together in a network, running samples of malware 48

3.1 Malicious Stream Profiler: Create and label malicious streams 54

3.2 Malicious stream processor: Capture and compare network profiles . . . 55

3.3 Honeywall Roo Logical Design 59

3.4 Virtual Honeynet Proposed Design 61

3.5 Top 50 Attackers by IP Address 63

3.6 Top 50 Attackers by Country 64

4.1 NCD results for 50 fuzzed packets. 50 packets were synthesized from an original slammer worm packet instance. Each synthesized packet contained 10 bytes of random fuzz added to its predecessor instance in a cumulative manner. 90

4.2 NCD results for 100 fuzzed packets. 100 packets were synthesized from an original slammer worm packet instance. Each synthesized packet contained 10 bytes of random fuzz added to its predecessor instance. . . 91

4.3 NCD results for 200 fuzzed packets. 200 packets were synthesized from an original slammer worm packet instance. Each synthesized packet contained 10 bytes of random fuzz added to its predecessor instance. . . 92

4.4 Comparative results using various similarity metrics (both edit distance and information theoretic) on the 50 fuzz packet dataset. The y-axis shows distance or similarity scores without normalization. The x-axis represents each of the 50 fuzzed packet. 93

4.5 Comparative results on the 50 fuzz packet dataset (similarity scores are normalized and scaled between 0-1). Here 0 = identical and 1 = dissimilar 94

4.6 Sensitivity test of distance with incremental fuzz 95

4.7 NCD and Jaro results for fuzzed Slammer worm packets 96

4.8 Mixed Internet traffic, containing malicious and benign samples from the 500 dataset compared with a reference Slammer worm exemplar/instance. 98

4.9 Condensed results: Mixed Internet traffic, containing malicious and benign samples from the 500 dataset 99

4.10 Comparison of similarity metrics on fuzzed worm packets, with reference Slammer worm instance 100

5.1 ROC Curve: the points on the curve represent the performance of the classifier. The ideal model is one that obtains a True Positive Rate of one and a zero False Positive Rate (i.e., TPrate = 1 and FPrate = 0, point A in this figure. A worst case scenario would be point B, coordinates (1,0), where TPR is zero and FPR is a maximum. A model that makes a random guess should reside along the line connecting the points (TPrate = 0, FPrate = 0), where every instance is predicted as a negative class, and (TPrate = 1, FPrate = 1), where every instance is predicted as a positive class. 108

5.2 A 2D visualization of a small dataset of different types of packet data, using MDS. The natural clusters forming are highlighted and partitioning between different packets types is clearly visible. 109

5.3 MQTC on a small dataset of mixed 100 packets of benign and worm packets. The algorithm took a long time to calculate a stable tree even for such a small dataset. The clusters are highlighted here in this figure. 112

5.4 UPGMA on a small dataset @ T = 0.55. The rectangular boxes show the clusters, the dark oval nodes represent similarity and dissimilarity percentages, while the light coloured flat oval nodes represent the packets. In a very short span of time (less than a minute), UPGMA was able to correctly cluster the dataset according to its labels. UPGMA results were visualized using a modified version of Ero Carreras script (http://blog.dkbza.org/2005/07/using-complexity-theory-to-measure.html) 113

5.5 NCD on a small dataset @ T=0.55. The different galaxies represent the various clusters, the connections are represented by edges with their similarity scores. Almost all nodes seem to be part of a cluster, which shows the effectiveness of NCD @ T=0.55. 116

5.6 Jaro Distance calculated on a small dataset @ T=0.55 suggests that this fixed threshold is not optimal for Jaro. The big dense galaxy forming indicates misclassification at this threshold value for the Jaro algorithm. A separate experiment was conducted to determine the optimum threshold value for each similarity metric. All nodes in the dataset seem to have connections/linkages with other nodes. 117

5.7 Levenshtein Distance calculated on a small dataset @ T=0.55, which is nearly half-way between 0 and 1. The results are not very good at this threshold value, as many nodes seem to be misclassified into a larger class. Very few nodes are correctly classified. 118

5.8 Hamming Distance on a small dataset @ T=0.55, which is nearly half-way between 0 and 1. The results are not very good at this threshold value, as many nodes seem to be misclassified into a larger class, very few nodes are correctly classified, while others are lying independently in the visualization. 119

5.9 NCD on the 500 dataset @ T = 0.55. Since this is an unbalanced dataset, we see clusters of differing sizes forming. The dense dark galaxy represents the dominant class in this dataset. The subsequent smaller clusters represent the other classes as seen by the classifier. Very few nodes seem to be independent or misclassified. ROC curve analysis (discussed in 5.3) revealed that the best classification can be achieved at a threshold value of 0.75. 121

5.10 Adaptive vs fixed threshold: The adaptive threshold technique has a clear advantage over the fixed threshold technique, in terms of accuracy. Here applying an adaptive threshold gives 87% accuracy, while using a fixed threshold gives only 64% accuracy. 122

5.11 Accuracy vs AUC vs Zero FP across all metrics. This figure shows that NCD with AUC yields a high number of TP values with very few FP values compared to Accuracy. This tradeoff helps decide a suitable threshold (T) using the adaptive thresholding method for T-NN clustering. 123

5.12 FP results for NCD, Jaro, Levenshtein and Hamming distance metrics - AUC vs Accuracy. Here NCD is showing the lowest number of FP samples compared to the others. 124

5.13 FP results for NCD and Jaro – AUC vs. Accuracy. Here NCD with AUC is showing the least number of FP samples. 124

5.14 ROC: Hamming vs Levenshtein vs Jaro vs NCD, while step-wise varying the threshold. Here Hamming and Levenshtein distance performs poorly with many false positives at the start and getting worse towards the end. Jaro distance, compared to the previous two metrics performs much better, with many TP results until at the end a few FP results start to appear. As is evident from the ROC curve analysis, NCD has outperformed other similarity metrics by comparison. 125

5.15 An ROC curve comparing Hamming, Levenshtein, Jaro, NCD and Spamsum similarity measures, while step-wise varying the threshold. It can be seen that Hamming and Levenshtein distance measures perform poorly, with many false positives at the start and getting worse towards the end. Jaro distance metric is much better, with true positive results until, at the end, a few false positives results start to appear. NCD and Spamsum outperform the other similarity metrics. 126

5.16 Comparative work to determine the best clustering algorithm for our domain. Adaptive thresholding performs the best, followed by fixed thresholding and k-NN for k = 3 and 5. 127

6.1 For categories a, b, c, d, each with a set of samples such as $a_1 \dots a_n$, a model can be defined as a list of tuples, where each tuple comprises of an exemplar a_ϵ selected from the samples and a corresponding distance threshold value a_τ. If there are significant variations within one class, there may be several exemplars from the same class. 131

6.2 The model written out to a csv file. Each row can be read as: The labelled exemplar ϵ , and the threshold τ. There can be several exemplars from the same class (e.g. class 30 in this case has several exemplars), indicating rich variations within the class samples or candidate sub-classes. Otherwise if all samples in the class are quite similar, then they can be represented by a single exemplar. 132

6.3 Novelty detection 135

6.4 An aerial view of the classification. We can see multiple clusters forming, with one large cluster in the center representing the majority class. While the minority classes are shown in the sides. This graphic gives a clear picture and idea of the different sizes of classes. We used 49 exemplars to successfully cluster this test set 139

6.5 The core of the large cluster is visible here. It has 1 exemplar, represented as green node and multiple samples of the same class, represented as pale blue nodes. 140

6.6 Enlarging previous figure: The core of the large cluster is visible here. It has 1 exemplar, represented as green node and multiple samples of the same class, represented as blue nodes. 140

6.7 A small cluster representing a small class, having 2 exemplars, represented as green nodes and multiple samples of the same class, represented as blue nodes. 141

6.8 Original model, with highlighted samples that will be removed from the model for testing the novelty detection algorithm 143

6.9 List of outliers found by the novelty detection algorithm 144

6.10 New Exemplar ϵ_{novel} and Threshold τ_{novel} pairs proposed by the algorithm. The highlighted samples/exemplars are the samples from the same class that we initially removed from the original model 144

6.11 A comparison of classification methodologies showing accuracy results for: (1) Fixed threshold (2) Optimum threshold using step-wise thresholding (3) Nearest Neighbour (4) k-Nearest Neighbour and (5) for exemplar learning. Results are calculated at varying number of exemplars for techniques 1-4. It can be seen here that exemplar learning results using Algorithm 1 (first green triangle) and then Algorithm 2 (second green triangle) outperforms other techniques by yielding better accuracy with least number of exemplars. 147