A Novel Approach to Clustering Malware Behaviour to Improve Malware Detection

Authors
Rebecca Kelly Merriman, MSc (Royal Holloway, 2019)
Daniele Sgandurra, ISG, Royal Holloway

Abstract

This article is based on my thesis - A Novel Approach to Clustering Malware Behaviour to Improve Malware Detection - a study to evaluate the accuracy of clustering-based malware detection to see whether it improves malware detection. Clustering malware behaviour can be very useful, but it is unknown how accurate clustering algorithms are when dealing with malware. In this article I will describe an attempt to measure the accuracy of these algorithms. My results show that the accuracy of clustering-based malware detection is highly subjective as it depends on many factors such as the type of methods used and the features selected. [a]

[a]This article is published online by Computer Weekly as part of the 2020 Royal Holloway information security thesis series `https://www.computerweekly.com/ehandbook/Royal-Holloway-A-novel-approach-to-clustering-malware-behaviour-to-improve-malware-detection`. It is based on an MSc dissertation written as part of the MSc in Information Security at the ISG, Royal Holloway, University of London. The full thesis is published on the ISG's website at `https://www.royalholloway.ac.uk/research-and-teaching/departments-and-schools/information-security/research/explore-our-research/isg-technical-reports/`.

Introduction

Machine Learning

Machine learning is a set of mathematical techniques used for information mining, pattern discovery and data inferencing. There are two main types of machine learning, specifically, supervised and unsupervised learning. Supervised learning is used to predict outputs (label on future data) from one or more inputs to learn a mapping between the two. For example, document classification and face detection. Unsupervised learning is used to learn relationships, structure or interesting patterns from the data. Unsupervised methods are useful due to the inexistence or expense of dataset labelling.

Why use Machine Learning

Machine Learning and Artificial Intelligence will not only make cyber-attacks more powerful but will also make cyber defence just as powerful.

Due to huge amounts of data being available now (the so called "big data") it is not practical to use traditional techniques or humans to analyse information. To this end, the rise in computational capacity and tools available to process and analyse data in real time have allowed us to use machine learning techniques to perform these tasks automatically. These techniques can be used in many domains, such as the health or financial sectors. For example, by analysing large databases of insurance policies, these algorithms can provide insurers with more personal and relevant information, resulting in more focused sales and reduced operational costs. Machine learning can also be used in the cyber-security domain, for example, to identify trends and meaningful patterns of cyber-attacks and to help predict, defend and respond to these attacks. Machine learning techniques are important as they can perform jobs that for humans would be impractical, less accurate and time consuming.

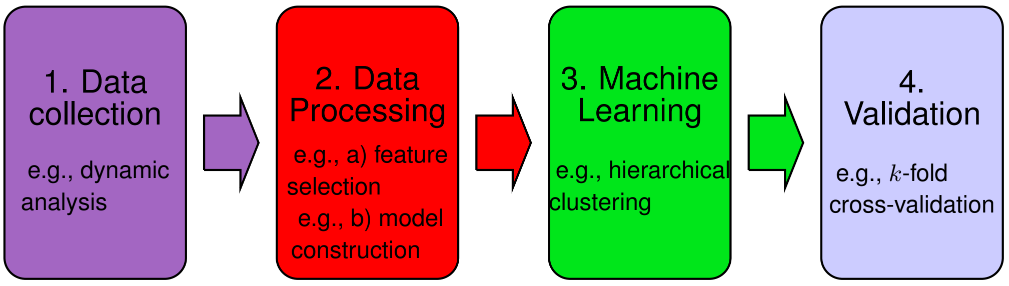
1. Data collection
e.g., dynamic analysis
2. Data Processing
e.g., a) feature selection
e.g., b) model construction
3. Machine Learning
e.g., hierarchical clustering
4. Validation
e.g., k-fold cross-validation

Figure 1: The process of a machine learning algorithm.

My project

In this article I focus on hierarchical clustering, an unsupervised machine learning technique. At each stage of the discussion I will use fruit to better illustrate the clustering process before summarising my contributions. In my MSc Project, I implemented hierarchical clustering for the behaviour of three different types of *malware*: 1) Ransomware 2) Backdoor and 3) Trojan. The goal was to evaluate the accuracy of clustering-based malware detection and conclude whether clustering malware behaviour improves (or not) malware detection.

My focus was on evaluating whether clustering malware behaviour is useful in malware detection. In fact, clustering can discover features that might be useful in malware family classification, or find commonalities between features within the same family, and that can be used to improve detection. In particular, if the algorithm performs with high accuracy in this context, then the selected features can be used to detect these malware families at run time in the future. For example, if there is a new family of Ransomware, then these features can be used to detect the new family more easily.

Malware

Malware stands for malicious software. It undermines the security of users by performing unwanted functions without the user's permission, e.g., stealing personal data.

Due to the rise of the digital age, malware attacks are on the increase and, therefore, we need to find novel ways to detect and prevent these attacks from occurring in the future.

The Method

To perform any machine learning algorithm, data needs to be processed to remove all irrelevant and redundant elements and to produce it in a form ready for input into the algorithm. After data processing, the results are checked for accuracy. This four-stage process is summarised in Figure 1. As previously mentioned, at each stage I will use fruit to illustrate the clustering process before summarising my contributions.

1. Data Collection

   Data collection is the process of gathering the data to run the machine learning algorithm on. At this stage, the *distinguishing factors* are chosen.

   Fruit example: If one wants to separate a bag of fruits containing apples, strawberries and bananas into each fruit type (i.e. perform clustering on the fruit families) then the first stage is to gather all the different types of fruit. The distinguishing factors of the fruit include type, colour, shape and size.

My project: I used a dataset and performed *dynamic analysis* on the three malware types listed above. The distinguishing factor was the behaviour of the malware as each malware performed different actions (e.g., file access).

Table 1 shows some distinguishing factors for fruit, cars and malware.

Distinguishing Factors

They define what distinguishes one sample (e.g., a piece of fruit, a car, a malware type) from another sample of the same type.

Item	Distinguishing factors
Bag of fruits consisting of apples, strawberries and bananas	• Colour (red, yellow, green) • Size • Shape
Car	• Make • Model
Malware	• Behaviour of the malware • Keys (e.g., generation, storage)

Table 1: Some examples of distinguishing factors for fruit, cars and malware.

2. Data Processing

Dynamic Analysis

The execution of a malware sample is monitored in a controlled environment (called a sandbox) whilst the program is being executed.

Data processing removes irrelevant or redundant information to increase the accuracy of the result and the performance of the algorithm. This is composed of two phases: feature selection and model construction.

(a) *Feature Selection*

After the features have been processed, they must be converted into a form suitable for the machine learning algorithm. In addition, the data is represented as a matrix (vector). To form a feature vector, the best (most) distinguishing factor is chosen.

Feature Selection

This removes the irrelevant and redundant features and selects the most relevant data that distinguishes one family (e.g., banana) from another family (e.g., apple).

Fruit example: I identified earlier (Table 1) that the distinguishing factors of the fruit are colour, size and shape. Then, the most distinguishing factor is identified. Colour cannot be the most distinguishing factor as both apples and bananas can be green, an apple can be green or red and a banana can be green or yellow. Fruit size (e.g., S, M, L) cannot be the most distinguishing factor as they are not unique for each piece of fruit. Therefore in my example, the *shape* of the fruit becomes the most distinguishing factor.

My project: The system calls of the malware (operations) were used to represent my feature vectors because the other information (e.g., key generation) was irrelevant and did not contribute to the predictive accuracy. Two feature selection methods were used: 1) *Category* - category of the system call and 2) *Full Representation* – category and action of the system call. Both selection methods were used because this made the malware families more distinct.

(b) *Model Construction*

Many types of feature vectors can be constructed. These include bit vectors (where a feature is either present (1) or absent (0)), frequency vector (the number of times the feature was observed), or n-grams (combinations of feature sequences). The different types of the distinguishing factor form the dimensions of the feature vector.

Model Construction

This is the process of constructing the feature vectors.

Fruit example: I identified shape as the most distinguishing factor. Next the feature vector is constructed using the different types of shapes: 1) banana - curve, 2) apple - circle and 3) strawberry - cone. As each fruit only has one shape, *bit vectors* are the most relevant feature vector. The bit vector for each fruit will consist of three dimensions ([curve, circle and cone]) and for each dimension (shape) the value “1” will be used to indicate the presence of that feature and “0” to indicate absence. For example, a banana is a curve shape and so its feature vector will have a 1 in the curve position and 0 in the circle and cone positions (i.e., the feature vector will be [1,0,0]). See Table 2 for the other corresponding bit vectors.

Fruit dataset	Shape (feature)	Vector ([curve, circle, cone])
	Curve	[1,0,0]
	Circle	[0,1,0]
	Cone	[0,0,1]

Table 2: The shape of each fruit (the most distinguishing factor) and the corresponding bit vector.

My project: *Bit* or *frequency n-grams* were used to construct my feature vectors. 1-gram (uni-gram), 2-grams (di-grams) and 3-grams (tri-grams) were used on the malware families.

N-grams explained

Consider the Great British Athletics Team. Athletes can run, jump, throw and some can do combination of these actions . . .

- 1-grams consist of one feature per dimension. Athletes who can only run, jump or throw will fall under this category, e.g., [run, jump, throw].
- 2-grams consist of two features per dimension. Athletes who can run and jump (e.g., long jumpers, high jumpers and hurdlers), run and throw (e.g., javelin) and jump and throw (e.g., shot put) will fall under this category e.g., [run_and_jump, run_and_throw, jump_and_throw].
- 3-grams consist of three features per dimension. Athletes who can run, jump and throw (e.g., heptathletes, decathletes) will fall under this category e.g., [run_jump_and_throw].

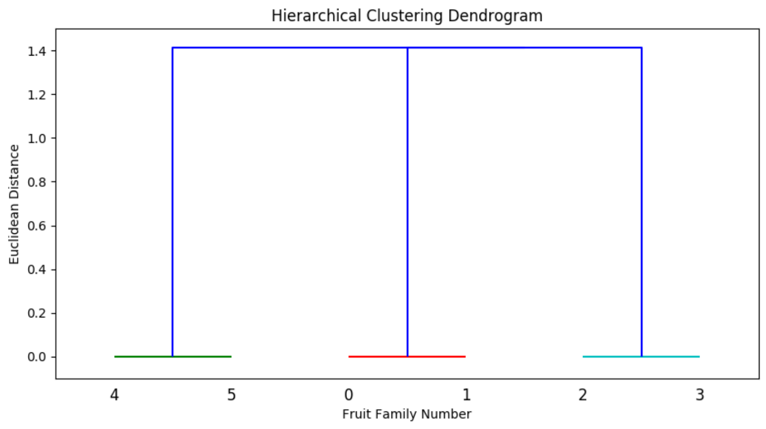

Figure 2: The dendrogram for the fruit family.

3. Machine Learning

In this project I focused on a specific machine learning algorithm called hierarchical *clustering*, as it provided more informative results in the form of a hierarchy.

Clustering

Clustering tries to group similar objects together. A cluster is a group of data points where the distance between each data point is small (objects in the same cluster are as similar to each other as possible) and the distance between data points in different clusters are large (objects in different clusters are as dissimilar to each other as possible).

Hierarchical clustering involves the following processes:

(a) Decide a dissimilarity (distance) metric to calculate the distance between each feature vector, e.g., Euclidean distance. This distance quantifies the similarity between two feature vectors.

(b) Scale the frequency feature vectors, e.g., standardisation to make sure no feature overly influences the result.

(c) Perform hierarchical clustering by constructing a *dendrogram*. To create a dendrogram the two features that are the most similar to each other (Euclidean distance is the smallest) are continually merged together to form clusters.

Fruit example: Let's assume there are six pieces of fruits in the bag, specifically two bananas, two apples and two strawberries. For simplicity, let's assign each fruit item a number:

Key

0 = banana 1	2 = apple 1	4 = strawberry 1
1 = banana 2	3 = apple 2	5 = strawberry 2

If this dataset was input into a hierarchical clustering algorithm to produce a dendrogram, using the feature selection and model construction algorithms explained previously, then fruit members 4 and 5 would be merged together to form a cluster. This is because they have the same shape (cone), and so the vectors are both [0,0,1] (Table 2) and the Euclidean distance is 0. The same would occur between the two bananas (0 and 1) and the two apples (2 and 3).

Euclidean distance calculation

The Euclidean Distance between two feature vectors $x = [x_1, x_2, \ldots, x_n]$ and $y = [y_1, y_2, ..., y_n]$ is calculated by:

$$\sqrt{(y_1 - x_1)^2 + (y_2 - x_2)^2 + \cdots + (y_n - x_n)^2}$$

For example:

- Both strawberries (4 [0,0,1] & 5 [0,0,1]): $\sqrt{(0-0)^2 + (0-0)^2 + (1-1)^2} = 0$.
- A strawberry & a banana (4 [0,0,1] & 0 [1,0,0]): $\sqrt{(1-0)^2 + (0-0)^2 + (0-1)^2} = \sqrt{2}$.
- A strawberry & an apple (4 [0,0,1] & 2 [0,1,0]): $\sqrt{(0-0)^2 + (1-0)^2 + (0-1)^2} = \sqrt{2}$.
- A banana & an apple (0 [1,0,0] & 2 [0,1,0]): $\sqrt{(0-1)^2 + (1-0)^2 + (0-0)^2} = \sqrt{2}$

Afterwards, all three clusters (banana cluster, apple cluster and strawberry cluster) would be merged together as the distances between each of the three groups is the same ($\sqrt{2}$). Figure 2 is the resulting dendrogram to illustrate this process.

My project:

I conducted hierarchical clustering between the different families of Ransomware, between the different families of Trojan and between the different families of Backdoors.

4. Validation

The clusters obtained are validated and evaluated to verify that if an independent set of features were obtained, the same set of clusters would be produced. This measures the correctness of clusters without using external information. Many validation metrics can be used and the most common ones are Precision, Recall, Fowlkes Mallows Score (FMS), F1-Score (F1), Adjusted Rand Index Score (ARI), Silhouette Coefficient (SC).

Fowlkes Mallows Score (FMS) explained

FMS compares the labelled dataset (U) and the clusters obtained by the dendrogram cut (V) to calculate the number of true positives ($TP =$ number of pairs that are in the both U and V), true negatives ($TN =$ number of pairs that are in different clusters in both U and V), false positives ($FP =$ number of pairs that are in the same cluster in U but different clusters in V) and false negatives ($FN =$ number of pairs that are in different clusters in U but the same cluster in V) and then calculates FMS score determined by

$$\frac{tp}{\sqrt{(tp + fp) \times (tp + fn)}}.$$

To validate the clusters obtained in hierarchical clustering, the dendrogram needs to be cut at each height to produce a set of clusters for that height. The clusters obtained will be validated to test for accuracy. The best clustering is the one that yielded a validation score *closest to 1.0*.

Fruit example:

As there are only two heights in the dendrogram, it will be cut at 0 and 1.4 ($\sqrt{2}$) respectively (indicated by the two yellow lines in Figure 3) to obtain different clusters. Three clusters were obtained at height 1.4 and they were $\{\{4, 5\}, \{0, 1\}, \{2, 3\}\}$ and one cluster was obtained at height 1.

Validation metrics (such as FMS) are used to validate the clusters at each height. An example of how one validation metric can be applied to the fruit example is explained below. In the fruit

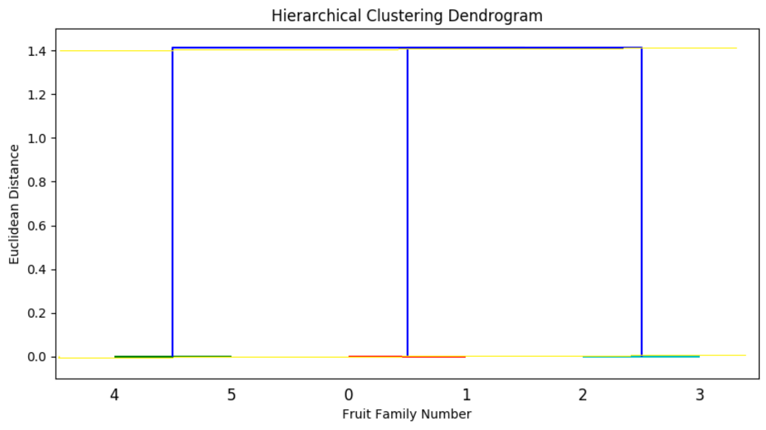

Figure 3: This figure displays where to cut the dendrogram. There are only two heights, 0 and 1.4 so the dendrogram is cut at those heights (indicated by the two yellow lines).

bag there are two bananas, two apples and two strawberries therefore the true assignments (the correct cluster assignment) for the bag of fruits is [banana, banana, apple, apple, strawberry, strawberry]. This is represented by the vector $[0, 0, 1, 1, 2, 2]$ as each type of fruit should be assigned to the same cluster.

If you cut the dendrogram in Figure 3 at cut 0.0, the predictive assignments (clusters obtained by the dendrogram cut) are $[0, 0, 0, 0, 0, 0]$. The number of $TP = 3$, $TN = 0$, $FP = 0$, $FN = 12$ and $FMS = 0.4$. At cut 1.4, the predictive assignments are $[0, 0, 1, 1, 2, 2]$. The number of $TP = 3$, $TN = 12$, $FP = 0$, $FN = 0$ and $FMS = 1.0$.

Table 3 shows the FMS, ARI, F1 and SC scores at the different heights. The heights that yield a score closest to 1 is the best clustering method. All metrics agree that the cuts obtained at the height 1.4 were the best clustering (score of 1.0) as they are identical to the original classification. This means that the feature selection (shape) and model construction (bit vector) method was the best clustering method.

Metric	Height	Score
FMS	0.0	0
	1.4	1
ARI	0.0	0
	1.4	1
F1	0.0	0.1
	1.4	1
SC	0.0	-1
	1.4	1

Table 3: The validation scores for each height of the dendrogram.

My project:

I used FMS, F1, ARI and SC to evaluate the similarity of each cluster obtained to the original dataset. This was done for all malware families using each feature selection and model construction method at each height. The clusters obtained at each cut were compared to the original dataset used in the dynamic analysis (1. Data Collection). The cut that was most similar to the true assignments (labelled dataset) was the best clustering.

The Main Results

Table 4 shows the experiments which produced the best clustering for each malware family. It was evident that the evaluation metrics did not agree on the best n-gram (1-gram, 2-gram or 3-gram), system call representation (full representation or category) and vector representation (bit vector or frequency vector).

Metric	Malware	N-gram, System Call Representation and Vector Representation
FMS	Ransomware	Uni-gram, Category, Frequency Vector
	Backdoor	Tri-gram, Full Representation, Frequency Vector
	Trojan	Tri-gram, Category, Frequency Vector
F1	Ransomware	Di-gram, Full Representation, Frequency Vector
	Backdoor	Uni-gram, Full Representation, Frequency Vector
	Trojan	Tri-gram, Category, Frequency Vector
ARI	Ransomware	Uni-gram, Full Representation, Frequency Vector
	Backdoor	Di-gram, Full Representation, Bit Vector
	Trojan	Uni-gram, Full Representation, Frequency Vector
SC	Ransomware	Uni-gram, Category, Bit Vector
	Backdoor	Uni-gram, Full Representation, Bit Vector, Uni-gram, Category, Bit Vector, Di-gram, Full Representation, Bit Vector, Di-gram, Category, Bit Vector and Tri-gram, Full Representation, Bit Vector
	Trojan	Uni-gram, Category, Bit Vector

Table 4: The experiments that produced the best clustering for each malware according to FMS, F1, ARI and SC.

I then ranked the clustering methods from best to worst for each metric and calculated a total score by adding the scores for each metric together. The one with the lowest total score was the best clustering method for each malware family (Table 5).

Malware	N-gram, System Call Representation and Vector Representation
Ransomware	Uni-gram Category Frequency Vector
Backdoor	Di-gram Full Representation Frequency Vector
Trojan	Tri-gram Category Frequency Vector

Table 5: The best clustering methods for each malware family.

Limitations

Running times of the experiments need to be considered. On a standard laptop computer, this algorithm run on these datasets may requires several days. This is mainly due to the large number of system call dimensions in the vectors, the large distances between two feature vectors and the large number of dendrogram heights which indicates that the feature vectors are not similar to each other. Therefore, as the running times were not considered, the best clustering found in this project may not reflect what the best clustering method actually is. Future experiments should run the algorithm on larger datasets, perform additional feature selection or use different vector representation methods to produce more accurate results. In this case, the best clustering methods should be based on both the evaluation metrics and running times because if it takes too long to run, people will be reluctant to use it to detect malware.

In my experiments the true assignments and testing labels were known (from the dataset). In real scenarios these might not be known due to new malware or variants of previously seen malware (e.g., due to polymorphism or metamorphism). This makes detection hard as one may not be able to identify what malware family the sample belongs to. This means that validation techniques, other than FMS, F1 and ARI (like Silhouette Coefficient, SC), should be used because in these validation metrics the ground truth is unknown. However, using SC can introduce problems. For example, Table 4 shows that when SC was used no malware family agreed on the best clustering method and Backdoor produced five different best clustering methods. This is not beneficial as it means that a programmer can use any of these experiments despite them having different vector dimensions and features selected. To verify that the best clustering methods according to SC were valid, I looked at the FMS, F1 and ARI scores. However, they were low (between 0.0001 and 0.453). Therefore, even though the SC scores were high, as the FMS, F1 and ARI scores were closer to 0 rather than 1; these results suggest that the cluster assignments produced by the dendrogram cuts were not similar to the labelled dataset. Therefore, if either FMS, F1 or ARI scores were high then the SC scores were low (and vice-versa).

These limitations suggest that clustering-based malware detection has a low validity and accuracy. However, this conclusion is problematic as some researches have reached a contradictory conclusion:

- Bayer et al.[1] produced results with a precision of 98.4% and a recall of 93.0%.
- Hamid et al.[2] produced accuracy results of 98.1% and 74.7%.
- Zhang et al.[3] produced accuracy results in the range of 46% – 91%.

The differences can be explained by the analysis, feature selection, model construction and machine learning methods as they all differed between the papers (Table 6).

Research	Techniques for Analysis	Feature Selection and Model Construction	Machine Learning Algorithms
MSc	Static and dynamic analysis	Operation/ category of operation bit/ frequency vector n-grams (1-gram, 2-gram and 3-gram)	Hierarchical unsupervised machine learning method
Bayer et al.	Dynamic analysis	Behavioural profiles	Hierarchical unsupervised machine learning method
Hamid et al.	-	-	K-means unsupervised machine learning method
Zhang et al.	Static analysis	TF-IDF n-grams (1gram, 2-gram, 3gram and 4-gram)	Supervised machine learning methods (Decision Tree, Random Forest, K-Nearest Neighbour, Naive Bayes, and Gradient Boosting Decision Tree)

Table 6: All four projects/research papers used different techniques for analysis, feature selection and model construction methods and machine learning algorithms.

Conclusion

The accuracy of clustering-based malware detection is highly subjective as it depends on many factors including the type of machine learning algorithm, the selected features, the feature selection methods, the model construction methods and evaluation metrics. This was illustrated in my results (Table 5) and when comparing my results to other papers (Table 6) , as I noticed that different methods of feature selection and model construction yielded different results. Therefore, there is a discrepancy over the accuracy of clustering-based malware and whether clustering improves malware detection.

[1] U. Bayer, P. M. Comparetti, C. Hlauschek, C. Kruegel, and E. Kirda, “Scalable, behavior-based malware clustering”, 16th Annual Network and Distributed System Security Symposium, vol. 9, pp. 1–18, 2009.

[2] I. R. A. Hamid, N. S. Khalid, N. A. Abdullah, N. H. A. Rahman, and C. C. Wen, “Android malware classification using k-means clustering algorithm”, IOP Conference Series: Materials Science and Engineering, vol. 226, 2017.

[3] H. Zhang, X. Xiao, F. Mercaldo, S. Ni, and F. Martinelli, “Classification of ransomware families with machine learning based on n-gram of opcodes”, Future Generation Computer Systems, vol. 90, pp. 211–221, 2019.

Future research should be conducted to find out all the reasons that may affect the accuracy of clustering malware and discover the best methods in terms of accuracy and a good run time to improve malware detection. For example, researchers should conduct a validation study where they use the same dataset that I used in this project using the feature selection, model construction and machine learning methods that Bayer et al., Hamid et al. and Zhang et al. used and see if they produce similar high accuracy results. This will help to conclude whether clustering improves malware detection or not.

Biographies

Rebecca Kelly Merriman graduated from Royal Holloway, University of London with a first-class honours in BSc Computer Science (Information Security) in 2018 and a distinction in MSc Information Security (GCHQ accreditation) in 2019. She now works for BAE Applied Intelligence as a graduate Software Engineer.

Daniele Sgandurra is a Senior Lecturer in Information Security at Royal Holloway, University of London in the Information Security Group (ISG), where he leads the Systems and Software Security Research Lab (S3Lab). His current research interests lie within systems and software security, focusing on malware analysis, sandbox evasion and virtualization security.

Series editor: S.- L. Ng