Design and Implementation of Anti Spyware System using Design Patterns

Mohamed Adel Sheta
Military Technical College

Mohamed Zaki
Al-Azhar University

Kamel Abd El Salam El Hadad
Military Technical College

H. Aboelseoud M.
Military Technical College

ABSTRACT
Spyware is considered as a great threat to confidentiality that it can cause loss of control over private data for computer users. This kind of threat might select some data and send it to another third party without the consent of the user. Spyware detection techniques have been presented traditionally by three approaches; signature, behavior and specification based detection techniques. These approaches were successful in detecting known spyware but it suffers from some drawbacks such as; the need for updating data describing the system behavior to detect new or unknown spywares, and the high level of false positive or false negative rate. Therefore, in this paper we introduce a proposed anti spyware system design and implementation using design patterns for detecting and classifying spyware. This proposed approach can be reusable and modifying itself for any new or unknown spyware.

General Terms
Computer security

Keywords
Spyware, Design patterns

1. INTRODUCTION
Malicious codes (malware) which were designed by hackers include multiple attacking methods such as data intercept and denial of service (DoS) attacking. These malicious codes spread using the weak spot of certain program, and zero day attack is expected. Malware contains virus, worm, Trojan horse and spyware [1]. First, virus is a contagious computer program that can copy itself and infect another computer. It spreads from one computer to another in some form of executable code when its host is taken to the target computer; for example when a user sends it over a network or the internet or carried it on a removable medium [2]. Second, worm is a self replicating malware computer program. It spreads through a computer network by sending copies of itself to other computers on the network and it may do so without the interference of any user. Unlike a virus, it does not need to be attached to any program. Worms almost always cause high network traffic, even if only by consuming bandwidth, whereas viruses almost always damage or modify files on an infected computer [3].

Third, Trojan horse is a malware that seems to perform a desirable function for the user before run or install but instead facilitates illegal access of the user's computer system. "It is a harmful piece of software that looks legitimate. Users are typically tricked into loading and executing it on their systems", as Cisco describes. The term is originated from the Trojan horse story in Greek mythology [4]. Finally, according to the Department of Computer Science and Engineering at the University of Washington, spyware is defined as "software that gathers information about use of a computer, usually without the knowledge of the owner of the computer, and relays the information across the internet to a third party location". Another definition of spyware is given as "any software that monitors user behavior, or gathers information about the user without adequate notice, consent, or control from the user" [5].

Unlike viruses, spyware is usually installed with the user's approval, since it provides some useful functionality either on its own or by another software application. That's why spyware extends beyond the boundaries of what is considered legal and illegal software and thus falls in a grey zone. The installed spyware may be capable of capturing keystrokes, taking screenshots, saving authentication credentials, storing personal email addresses and web form data, and thus may obtain behavioral and personal information about users [5]. It may also communicate system configuration including hardware and software, system accounts, location information, and information about other aspects of the system to a third party. Spyware may, e.g., show characteristics like nonstop appearances of advertisement pop-ups, open a website or force the user to open a website which has not been visited before, install browser toolbars without seeking acceptance from the user, change search results, make unexpected changes in the browser, display error messages, and the occurrence of network traffic without any request from the user.

In this paper, an anti spyware system using design patterns will be designed and implemented for detecting and classifying spyware. This proposed approach can be reusable and modifying itself for any new or unknown spyware. This paper is organized as follows; related work and design patterns are discussed in section 2 and section 3, respectively. The proposed anti spyware system design and implementation are explained in section 4 and section 5, respectively. The conclusions and future work are discussed in section 6.

2. RELATED WORK
Spyware detection techniques are used to detect the spyware and prevent the infection of the computer system. They can be categorized into signature based detection, behavior based detection, specification based detection, and data mining based detection [3] that will be discussed in the next sections.

2.1 Signature Based Detection
Signature based detection detects spyware by comparing the spyware signature to the database. These signatures are created by examining the disassembled binary code of

spyware. Disassembled code is analyzed and features are extracted. These features are used to construct the signature of a certain spyware family [2]. The main advantages of this technique is that it can accurately detect known spyware and using less amount of resources to detect the spyware because it mainly focus on signature of attack. The main disadvantage is that it is unable to detect the new spyware as there is no available signature for this new type of spyware [6].

2.2 Behavior Based Detection

The function of behavior based detection is to analyze the behavior of known or unknown spyware. It usually occurs in two phases: training phase and detection phase. During training phase the behavior of the system in the ideal state is observed and machine learning technique is used to create a profile of such normal behavior. The detection phase is the comparison of the current system behavior after attack to the normal behavior and differences are flagged as potential attacks [7]. The main advantage of this technique is that it is able to detect known as well as new or unknown instances of spyware because it focuses on the behavior of system to detect unknown attack. The main disadvantage of this technique is that it constantly needs to update the data describing the system behavior. It needs more resources like CPU time, memory and disk space as well as level of false positive rate is high [8].

2.3 Specification Based Detection

Specification based detection is derivative of behavior based detection that tries to overcome the high false positive rate associated with it. Monitoring programs are involved in executions and detecting deviation of their behavior from the specification, rather than detecting the occurrence of specific attack patterns [9]. The main advantage of this technique is that it can detect known as well as new or unknown instances of spyware and level of false positive is low. The main disadvantages of this technique are the level of false negative rate is high, not as effective as behavior based detection in detecting new attacks and development of detailed specification is time consuming.

2.4 Data Mining Based Detection

Data mining has been the main focus of many spyware researchers for detecting the new unknown spyware. They have added data mining as a fourth proposed spyware detection technique [3]. Data mining helps in analyzing the data with automated statistical analysis techniques, by identifying meaningful patterns or correlations. The results from this analysis can be summarized into useful information and can be used for prediction. Machine learning algorithms are used for detecting patterns or relations in data which are further used to develop a classifier. Data mining is capable of detecting new or unknown spyware with high detection rate compared to signature, behavior, and specification based detection methods [10][11].

3. DESIGN PATTERNS

Design patterns are a general reusable solution to a commonly occurring problem within a given context in software design. It is a software solution for problems that arise regularly during software design. It serves as readily applicable, time saving strategies for software development. Design patterns are a description or template for how to solve a problem that can be used in many different situations. The idea of a design pattern was developed by Christopher Alexander in his work on reusable strategies for architecting space and structure. Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice [12].

3.1 Design Patterns Categories

Design patterns are categorized into three main categories depending on their functions; creational, structural and behavioral patterns. Creational patterns provide instantiation mechanisms, making it easier to create objects. Structural patterns generally deal with relationships between entities, making it easier for these entities to work together. Behavioral patterns are used in communications between entities and make it easier and more flexible for these entities to communicate. In Table 1, the all types of design patterns related to their categories are summarized and it discussed with more details in [12].

Table 1: Types of design patterns

Behavioral	Structural	Creational
Factory Abstract Factory Builder Prototype Singleton	Adapter Bridge Composite Decorator Facade Flyweight Proxy	Interpreter Template Method Chain of Responsibility Command Iterator Mediator Memento Observer State Strategy Visitor

3.2 Design Pattern Selections

One challenge for the current anti spyware systems is the need of changing the structure of the programs to be able to detect new or unknown spywares. In our proposed anti spyware system, observer, factory, and abstract factory design patterns are selected to be used in this system. Observer design pattern defines a one-to-one dependency between objects so that, if one object changes state, all its dependents are notified and updated automatically [13]. Therefore, in the case of any modifications in the program, these will be inheriting in the system to be reusable.

In the factory design pattern an interface for creating an object is provided, but it leaves a choice of the object's concrete type to a subclass [13]. Therefore, when a scanned file is classified as a new spyware the system will create a new spyware type according to its family.

Finally, abstract factory design pattern provides an interface for creating families of related or dependent objects without specifying their concrete classes [13]. So, when a scanned file is classified as unknown spyware, the system will create a new spyware family and new spyware type with this new behavior to make the system update itself.

4. THE PROPOSED ANTI SPYWARE SYSTEM DESIGN

In the proposed anti spyware system design, security patterns will be combined with design patterns in an integrated anti spyware system. Security patterns analyze the detected signature, the behavior or the data depending on the used methods for classifying the scanned file. Moreover the

reusable advantage of the design patterns for building a new anti spyware patterns for each spyware family. So that, in the proposed approach, there is no need for changing the structure of the current anti spyware programs in case of a new spyware type is existed.

4.1 The Proposed Framework

The proposed anti spyware framework will be consists of three layers as shown in Figure 1. First, the lowest layer is the selected design patterns which will be suitable in the case of anti spyware containing the observer, factory, and abstract factory design patterns. Second, the middle layer is the security patterns which analyze signature in the case of using signature based detection techniques. It also analyze behavior when using behavior/specification based detection techniques, or data when using data mining techniques of the scanned file. Finally, the highest layer is the new anti spyware patterns for each family of spyware using design patterns for detecting new or unknown spywares.

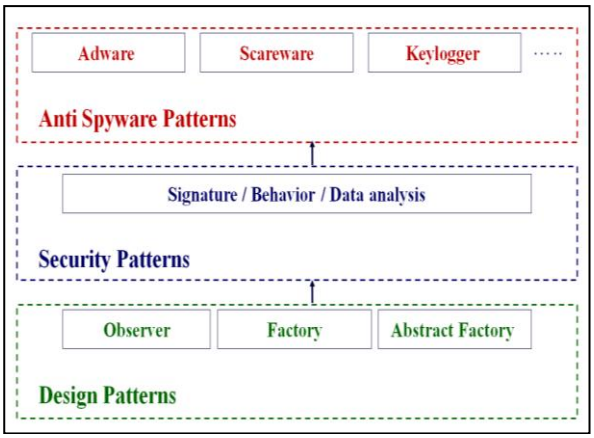


Fig 1: The proposed anti spyware framework

Figure 2 shows the proposed anti spyware block diagram that uses the design patterns approach. It starts with the problem description for the incoming data then choosing the corresponding design patterns to this problem and after that building a new design patterns system to classify the scanned file into benign or spyware.

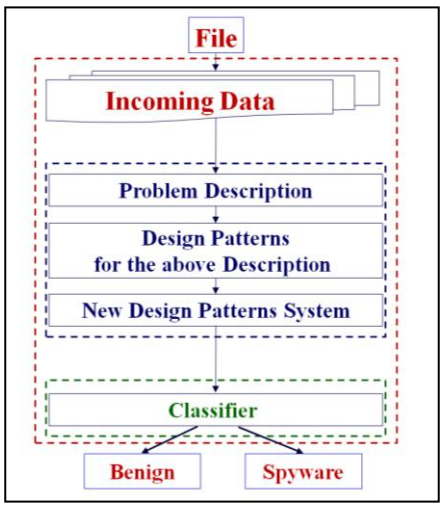


Fig 2: The proposed anti spyware block diagram

4.2 The Proposed System Architecture

Figure 3 shows the proposed anti spyware system architecture based on design patterns. The classifier in this architecture used to classify the input file after representation of the data in the format that is predefined to the classifier before. This architecture makes the system ready for detecting new spyware type or new spyware family. In this section, we will introduce the different blocks of this proposed architecture as follows.

4.2.1 Observer Design Pattern

The observer design pattern jobs are discussed as follows. First, if the input data file is known to the classifier then the classifier will detect its type from the predefined spyware types known to the system. Second, if the input data file is a new spyware type of a predefined spyware family then the classifier will detect this type and observer design pattern will trigger the system to create this type under its family. Finally, if the input data file is a new spyware type of a new spyware family then the observer design pattern will trigger the system to create this new family and its new behavior.

4.2.2 Factory Design Pattern

If the input data file is a new spyware type of a predefined spyware family then the classifier will detect this type and the factory design pattern will create this type related to its family in the system database to be inherited in the system.

4.2.3 Abstract Factory Design Pattern

If the input data file is a new spyware type of a new spyware family. The abstract factory design pattern will trigger the spyware factory and the behavior factory to create the new family of the spyware and its new behavior in the system database to be inherited in the system.

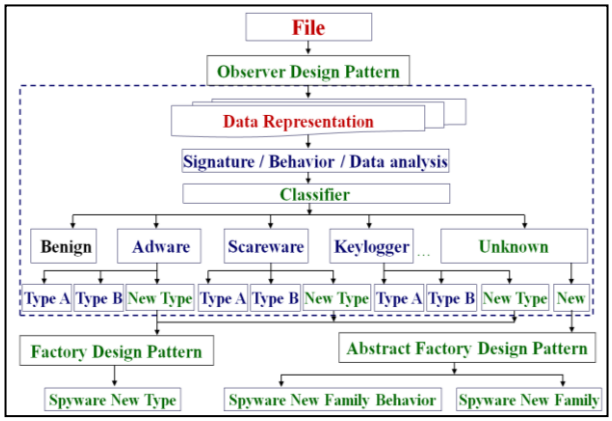


Fig 3: The proposed anti spyware system architecture

5. THE PROPOSED ANTI SPYWARE SYSTEM IMPLEMENTATION

In this section the system class diagram will be discussed in section 5.1, and the implementation setup and results are shown in section 5.2.

5.1 System Class Diagram

Figure 4-(a) shows the observer design pattern class diagram that contains two main classes. The first class is AntiSpywareSubject which is the interface between the system and the outside environment. Its update will be reflecting on the spyware or spyware behavior database. The second class is AntispywareDatabase that can register, or remove observers in the array list. It can also notify all of observers or some of them that exist in the array list for

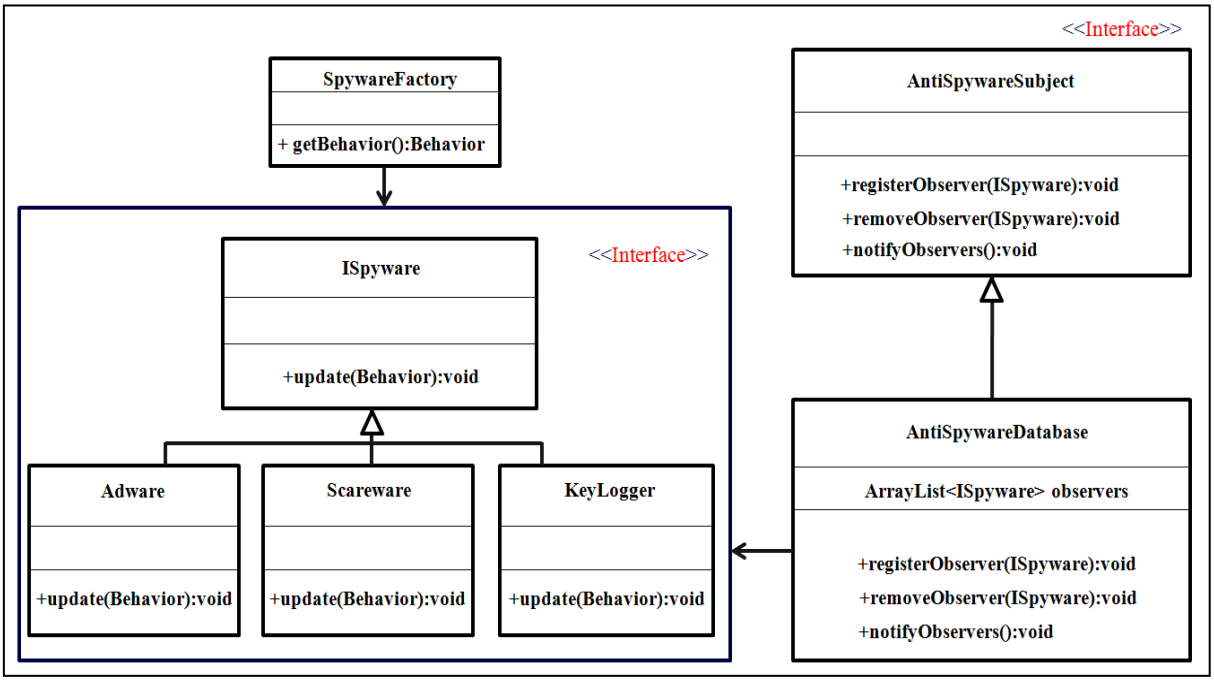
updating spyware family or the behavior using ISpyware interface.

Figure 4-(b) shows the abstract factory and factory design patterns class diagram that starts with the two main classes, ISpyware and Behavior classes. These two classes get their order for creating new objects of spyware families and behaviors from their own SpywareFactory and BehaviorFactory. One of these recent factories is chosen to create an object related to the AbstractFactory class order that has from the FactroyProducer class.

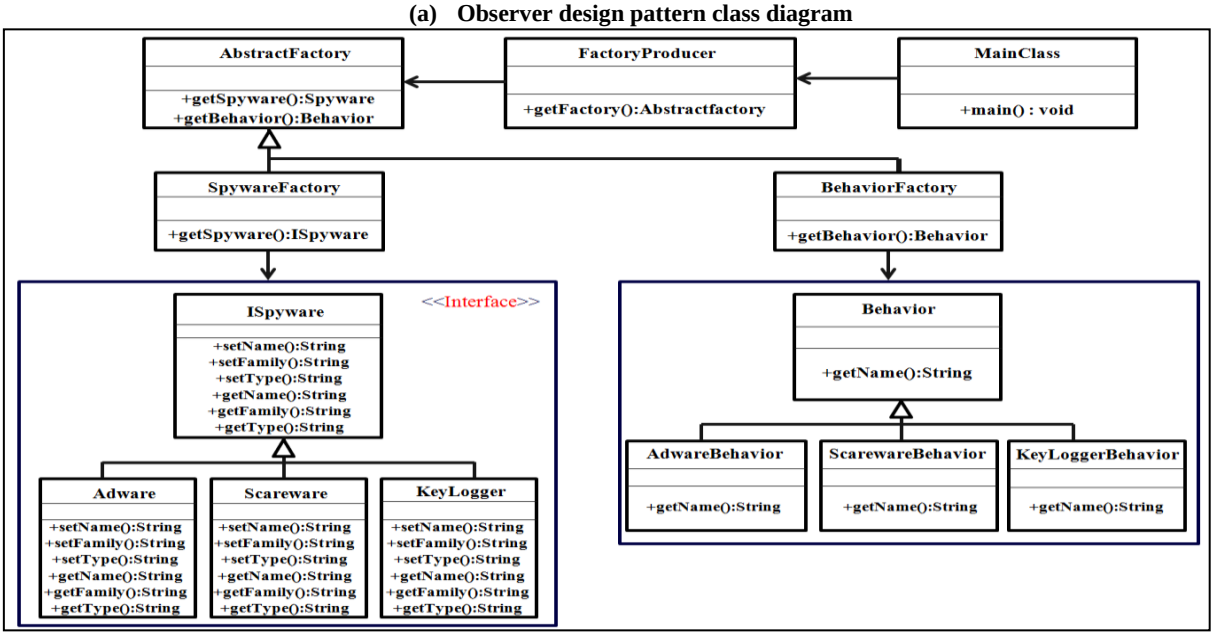
5.2 Implementation Setup and Results

Our implementation runs on Intel Core i7 with 4GB of RAM using Eclipse Java version Juno R1. Figure 5 shows the Eclipse Java software interface with its different modules and simulation output results. The results show some of new spyware families and behaviors that created from the simulation program. As shown in Figure 5, the Adware, Adawra1, and Type1 are some of the output results of spyware family, name, and type, respectively when a new object is detected by the proposed simulated system.

The simulated results also show updating all spyware behaviors (i.e., Adware, Scareware, and KeyLogger) or updating only the behavior of KeyLogger.



(a) Observer design pattern class diagram



(b) Abstract factory and factory design pattern class diagram

Fig 4: System class diagrams

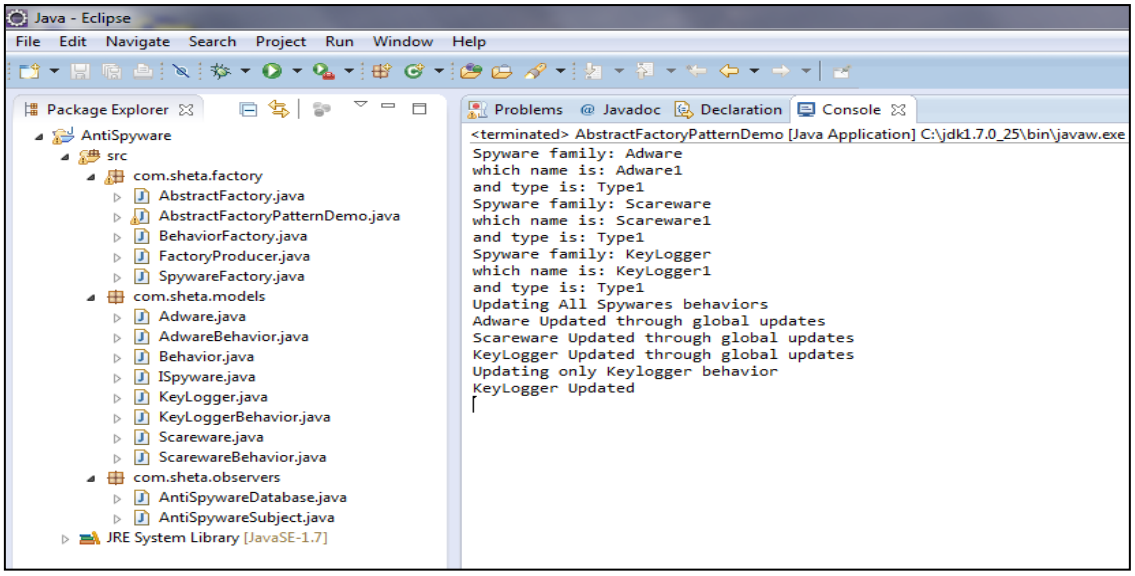


Fig 5: Simulation Output Results

6. CONCLUSTIONS AND FUTURE WORK

In this paper we proposed anti spyware system based on design patterns for detecting and classifying spywares. The classification of new or unknown spyware will create a new object according to its type so the system will update itself. The proposed anti spyware system is reusable in which it can modify itself in case of new spyware so that new as well as unknown spyware will be inherited in the system to be detected. The simulated results show updating all spyware behaviors or updating any one of these types. In future work, the proposed system based on design patterns can be combined with any of the traditional spyware detection methods.

7. REFERENCES

[1] G. Padmavathi, and S. Divya "A Survey on Various Security Threats and Classification of Malware Attacks, Vulnerabilities and Detection Techniques", The International Journal of Computer Science & Applications (TIJCSA), Vol. 2, pp. 66-72, India, 2013.

[2] Donghwi Lee, Won Hyung Park, and Kuinam J Kim "A Study on Analysis of Malicious Codes Similarity Using N-Gram and Vector Space Model", IEEE International Conference on information and applications (ICISA), pp. 1-4, Republic of Korea, 2011.

[3] Jyoti Landage, and Wankhade "Malware and Malware Detection Techniques: A Survey", International Journal of Engineering Research & Technology (IJERT), Vol. 2, pp. 61-68, India, 2013.

[4] Mohamad Fadli Zolkipli, and Aman Jantan "A Framework for Malware Detection Using Combination Technique and Signature Generation", IEEE International Conference on Computer Research and Development, pp. 61-68, Malaysia, 2010.

[5] Raja Khurram Shazhad, Syed Imran Haider, and Niklas Lavesson "Detection of Spyware by Mining Executable Files", IEEE International Conference on Availability, Reliability and Security (ARES), pp. 295-302, Sweden, 2010.

[6] Kai Huang, Yanfang Ye, and Qinshan Jiang "ISMCS: An Intelligent Instruction Sequence based Malware Categorization System", IEEE International Conference of Anti-counterfeiting, Security, and Identification in Communication, pp. 509-501, China, 2010.

[7] Mohammad Wazid, Avita Katal, R.H. Goudar, D.P. Singh , and Asit Tyagi "A Framework for Detection and Prevention of Novel Keylogger Spyware Attacks", IEEE International Conference on Intelligent Systems and Control (ISCO), pp. 433-438, India, 2012.

[8] Karan Sapra, Benafsh Husain, Richard Brooks, and Melissa Smith "Circumventing Keyloggers and Screendumps", IEEE International Conference on Malicious and Unwanted Software, pp. 103-105, USA, 2013.

[9] Raihana Md Saidi, Siti Arpah Ahmad, Noorhayati Mohamed Noor, and Rozita Yunos "Windows Registry Analysis for Forensic Investigation", IEEE International Conference on Technological Advances in Electrical, Electronics and Computer Engineering (TAEECE), pp. 132-136, Malaysia, 2013.

[10] Raja Khurram Shahzad, Niklas Lavesson, and Henric Johnson "Accurate Adware Detection using Opcode Sequence Extraction", IEEE International Conference on Availability, Reliability and Security (ARES), pp. 189-195, Czech Republic, 2011.

[11] Raja Khurram Shahzad and Niklas Lavesson "Detecting scareware by mining variable length instruction sequences", IEEE International Conference on Information Security South Africa (ISSA), pp. 1-8, South Africa, 2011.

[12] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software, Boston, Massachusetts, Addison-Wesley Longman Publishing Co., Inc., USA, 1995.

[13] Eric Freeman, Elisabeth Freeman, Bert Bates, and Kathy Sierra, Head First Design Patterns, O'Reilly Publishing Co., Inc., USA, 2008.