Performance and Maintainability Evaluation of Anti-Spyware System

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 somewhat of a silent killer, because its essential task is secretly and quietly monitoring or sending victim's sensitive information to a separate third party. Unfortunately, existing anti-spyware systems lack the ability to cope with the rapid changes in the spyware signatures and programs.

The main challenge in recent anti-spyware systems is to design efficient system that able to detect new and unknown spywares in a reasonable time. Furthermore, lack of interest in existing anti-spyware systems reusability. This paper introduces an adaptive anti-spyware system that able to deal with unpredictable discovered spywares on run time and improves the detection accuracy. The proposed system adopts design patterns approach in detecting and classifying spyware, in the sense that, reuse existing systems components in detecting new or unknown spywares without performing changes on these systems' designs. The proposed

anti-spyware system can be considered as an engineering product that needs to be verified in terms of performance and maintainability. The aim here is to guarantee the performance of the designed system by defining evaluation methods for assessing the performance and maintainability of this design. Thus, the performance of the proposed system has been evaluated through the adopted data mining evaluation metrics. While, amount of reuse and reusability metrics have been defined to evaluate the proposed system maintainability.

General Terms
Computer security.

Keywords
Spyware, Data mining, Design patterns.

1. INTRODUCTION
Recently, the frequent emerging of malicious activities required fast update of existing anti-spyware systems. Commercial products are still suffering from two challenges: (1) the frequent update of spyware signatures database and (2) the frequent upgrade of the current programs design versions in order to accommodate the new and unknown spyware families. These challenges significantly impacts the adopted detection approaches accuracy in detecting zero-day-attack.

Currently existing methods for tackling spyware detection challenges are primarily based on two complementary approaches; signature-based and behavior-based approaches. These approaches are classified according to the type of features that use for discovering spyware activity. It is worth mentioning that, adoption of data mining based techniques helps in addressing the frequent update of the signatures database challenge. While, using of design pattern approach contributes to solving the frequent upgrade challenge.

1.1 Detection Techniques
Spyware detection techniques are used to monitor and detect different types of spyware and hence preventing the infection of computer systems [1]. These techniques can be categorized into either one of four types of detection; signature based detection, behavior based detection, specification based detection, and data mining based detection [2]. The first method, signature based detection detects spyware by comparing the spyware signature to the database [3].

The second method, behavior based detection, the main function is to analyze the behavior of either known or unknown spyware [4]. The third method, specification based detection, the process of monitoring programs is involved in executions and detecting the degree of difference from the behavior from the specification [5]. Data mining is the main focus of many researchers in the field of new and unknown spyware detection [2]. The results from this method of analysis can then be summarized into information that is useful and can be used for prediction. Data mining is capable of detecting new or unknown spyware with high detection rate compared to signature, behavior, and specification based detection methods.

1.2 Design Patterns
Design patterns are a general reusable solution to a commonly occurring problem within a given context in software design [6]. Design patterns have been used in various research fields as an adaptive solution for software design. Recently, securing the design pattern has gained a momentum attention in software design approaches. A security pattern encapsulates security design expertise that addresses recurring information security problems in the form of a credentialed solution. According to Fernandez definition "a security pattern is a design response to a perceived threat and a pattern is not intended to repair vulnerability but to stop or mitigatea threat" [7].

This work integrates the data mining technique with the design pattern approach in designing and implementing an adaptive anti-spyware system that able to detect new and unknown spywares. The main objective is to introduce an accurate anti-spyware system with the capability of reusing existing design. This reusability feature benefits the maintainability of the running systems such that reduces its possibility to failure along with its high detection accuracy. This paper is organized as follows: related work, the proposed anti-spyware system design and implementation, experimental results, and conclusions.

2. RELATED WORK

2.1 Data Mining

Raja K. et al. [8] detected spyware by using data mining method with a byte sequence mining approach.The experiment applied on a data set contained 137 files.Out of these, 18 files were spyware and 119 files were benign. Feature sets generated by Common Feature Based Extraction (CFBE) selection method produced better results than feature sets generated by Frequency Based Feature Extraction (FBFE). The overall Accuracy (ACC) was 90.5% and the Area under Receiver Operating Characteristic Curve (AUC) score of 0.83. Raja K. et al. [9] detected adware by using data mining method with an opcode sequence extraction mining approach. The experiment applied on a data set contained600 files. Out of these, 300 files were spyware and 300files were benign and AUC was equal to 0.949.

Zongqu Z. et al. [10] detected malware by using data mining method based on the control flow of software mining approach. The experiment applied on a data set contained 9398 files. Out of these, 4828 files were malware and 4570 files were benign. The results were ACC equal to 97%, AUC score of 0.993 and low false positive rate equal to 3.2%.

2.2 Design Patterns

J. Yoder and J. Barcalow [11] described the seven different architectural patterns for applications' security. These seven patterns work together to provide a security framework for building applications. Schumacher et al. [12] envisioned the possibilities of security engineering with patterns; they redefined a template for security patterns and showed the benefits of this approach with a real world scenario as an example. Hafiz et al. [13] analyzed various classification schemes for security patterns and proposed a new one.

Their classification scheme uses the threat model and application context to partition patterns. It further organizes the patterns in a hierarchy. The proposed work differs from the aforementioned related work in different points. First, the proposed system integrates the data mining approach with the design pattern in designing an accurate and adaptiveanti-spyware system. The proposed system outperforms the related work performance detection accuracy. Moreover, an evaluation for the reusability of the proposed design has been carried out that differentiates this work.

3. THE PROPOSED ANTI-SPYWARE SYSTEM DESIGN

Motivated by the existing data mining-based anti-spyware model challenges, the proposed model has been designed and implemented to overcome some of the aforementioned limitations. This contributes significantly in the system reusability and helps in efficient detection of new and unknown spywares. The proposed anti-spyware framework will be consists of three layers as shown in Figure 1. It is worth mentioning that, the presented figure represents an adapted design in our earlier publication in [14]. First, the lowest layer is the design patterns which will suit the problem of spyware detection. Selection of these design patterns is mainly attributed to defining the problem precisely, in this context, spyware detection problem. The middle layer; the security pattern layer includes the data mining approach components; feature extraction and selection. Finally, the highest layer, anti-spyware layer is the interface layer for the proposed framework where the security patterns and design patterns are combined for detecting new or unknown spyware.

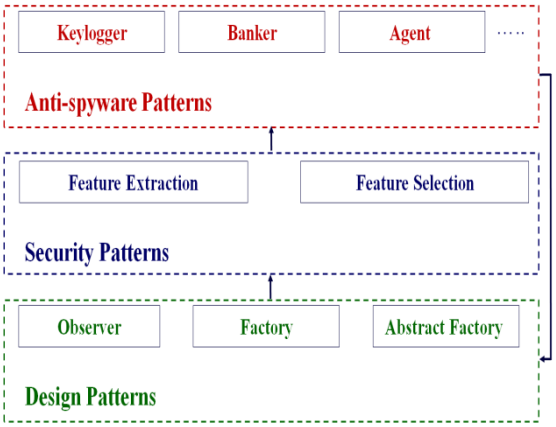

Fig 1: Proposed anti-spyware framework

3.1 Proposed Data Mining-based Security Pattern

As shown in Figure 2, mapping one-to-one between the traditional data mining-based security pattern and the proposed data mining-based security pattern is achieved. This mapping clarifies the enhanced features in the proposed model including the *Balanced* features in the used data set, *Unique* Features and *Unknown Spyware* in feature extraction, and finally the *Features Weight* and *CFFBE* in thefeature selection.

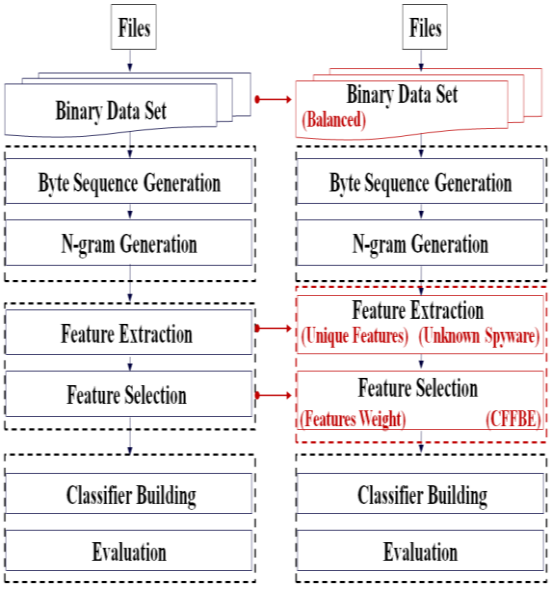

Fig 2: Security patterns (details of layer 2 of Fig 1)

3.2 Proposed Anti-spyware Patterns

Figure 3 shows the proposed anti-spyware pattern.The classifier follows the preparatory steps for the input files data representation. The classifier starts its execution by classifying the input file using the three design patterns; observer, factory, and abstract factory design patterns as follows: if the received processed file is known to the classifier then the classifier will detect its type from the predefined spyware types known to the system, this is the simplest case. However, if the input data file is a new spyware type of a predefined spyware family or unknown spyware family, then the classifier will detect this type and observer design pattern will trigger the spyware factory design pattern

to create this type related to its family in the system database to be inherited in the system.

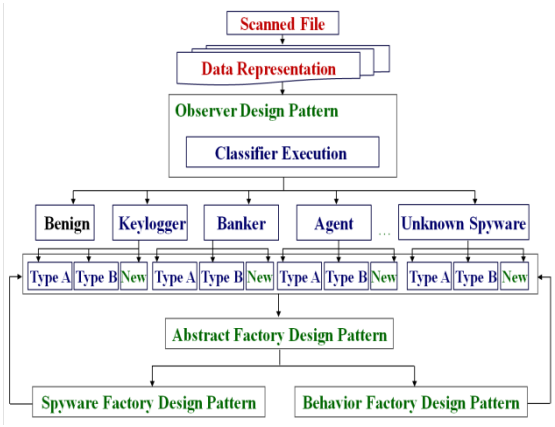

Fig 3: Anti-spyware pattern (details of layers 1&3 of Fig 1)

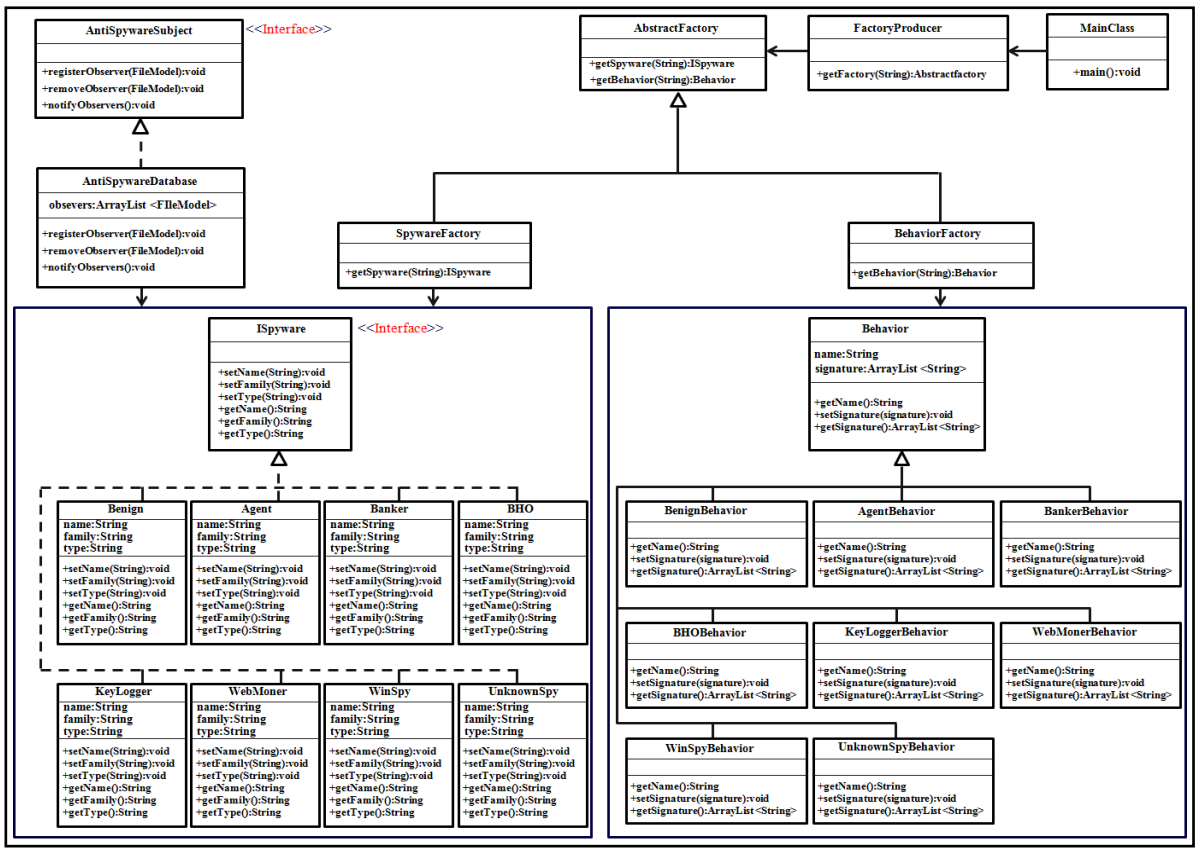

Fig 4: Proposed anti-spyware system class diagram

3.3 Proposed System Class Diagram

A design pattern system class diagram has been proposed to achieve the objective in reusability and detecting new or unknown spyware, in the sense that, Figure 4 illustrate the observer, abstract factory, and factory design patterns class diagram. Observer design pattern 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 other observers or some of them that exist in the array list for updating spyware family or the behavior using *ISpyware* interface. Abstract factory and factory design patterns start 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* respectively. One of these recent factories is chosen to create an object related to the *AbstractFactory* class order that has from the *FactroyProducer* class.

4. THE PROPOSED ANTI-SPYWARE SYSTEM IMPLEMENTATION

In this section, an illustration on the proposed system will be presented, including the data sets preparation steps, the

adopted data mining technique steps (i.e. feature extraction and selection), and the integration of the design patterns in the proposed system.

4.1 Balanced Binary Data Set Collection

As Binary spyware data set is collected from VX Heavens website [15]. Binary benign data set is collected from system directory of a fresh copy and spyware free guaranteed windows7, 32-bit operating system. The data set consists of 310 binary files out of which 270 benign and 40 spyware which is a balanced data set. The Malware File Percentage (MFP) is needed to be less or equal 15% of the total population in order to yield a high prediction performance [8].

4.2 Byte Sequence N-grams Generation

In the n-grams generation method, the data set is converted into hexadecimal format (byte-sequence). The generation method extracts byte sequences of the desired n-size. Previous researches have shown that n-grams of size 5 produced the most overall accurate results [8]. In this study it is suggested to use n-grams equal to 4, 5, and 6 to compare the performance of the proposed algorithms in each of them.

4.3 Feature Extraction

In this section, the feature extraction of the proposed method will be discussed with its steps used in this approach as shown below. The first column in the next tables is the type of spyware families or benign. The next three columns are the number of features using n-gram equal to 4, 5, and 6 for these types in each step.

Table 1 introduced the distinct features regardless the number of occurrence of each feature in all files in each class. In this step the total number of features decreased to be nearly between 4 and 5 million.

Table 1: Number of distinct features in each class

Type	N-gram = 4	N-gram = 5	N-gram = 6
BENIGN	3,043,166	4,112,403	4,360,495
SPY_AGENT	217,640	191,288	155,561
SPY _BHO	82,471	92,722	93,376
SPY_BANKER	299,153	245,885	207,627
SPY_KEYLOGGER	81,340	117,770	83,429
SPY_WEBMONER	145,096	157,943	141,521
SPY_WINSPY	66,193	82,992	86,512
Total	**3,935,059**	**5,001,003**	**5,128,521**

Table 2 shows the generation of the unique features in each class type that not exist in any other classes. This step introduces the strongest features which can be considered as a fingerprint for each class type.

Table 2: Number of unique features in each class

Type	N-gram = 4	N-gram = 5	N-gram = 6
BENIGN	2,959,998	4,046,441	4,330,061
SPY_AGENT	197,345	176,484	147,330
SPY _BHO	59,735	71,115	83,461
SPY_BANKER	288,577	238,972	202,972
SPY_KEYLOGGER	52,584	85,981	69,783
SPY_WEBMONER	108,182	126,482	124,972
SPY_WINSPY	43,571	63,893	74,635
Total	**3,709,992**	**4,809,368**	**5,033,214**

4.4 Feature Selection

Common Feature Frequency Based Extraction (CFFBE) has been proposed to select the features of the highest occurrence and the highest frequency in the files of each class type in the data set. This new method is proposed to select the strongest features in each class type to introduce the training data set that will be used in the classifier building. This method is hybrid of two previous methods. The first method is Common Feature Based Extraction (CFBE) which selects the features of the highest occurrence in the files of each class type in the data set. The second method is Frequency Based Feature Extraction (FBFE) which selects the features of the highest frequency in the files of each class type in the data set.

This work has been extended by applying the proposed algorithm CFFBE on the data set used, after creating a new class type for unknown spyware families. Unknown spyware is a class with features common between all of the spyware families in this data set but not in benign. These features can be assumed as a fingerprint for any new family of spyware that will be created in the future.

4.5 Classifier Building

In the learning phase, the main task is to build the classifier and formulate the primary detection model using training data set. Classifier building phase is done in offline state.

Three Attribute Relation File Format (ARFF) databases based on frequency and common features were generated. Each of them has two columns; the feature column and the type column which will be considered as the class during the operation of building the classifiers. A major part of the generated training data set feeds to the classifier for building the classification model that classifies the input as a benign or one of spyware family or type, meanwhile, the remaining part of the training data set is used in testing and verifying the performance of the generated model.

4.6 Integrated Design Patterns System

In accordance with the proposed design, integrated design patterns for anti-spyware system has been implemented to achieve the objective in reusability and detecting new or unknown spyware. The implemented design patterns layer components are observer, abstract factory, and factory patterns. The observer design patterns acts as the classifier in the proposed design patterns-based system during the execution. In the sense that, classifier rules are generated periodically, then inserted in the design patterns. While the system is running, if the system discover a pre-trained type or a new/unknown type related to pre-trained family, the abstract factory design pattern issue an order to both the spyware factory and behavior factory design patterns to create objects from the spyware family and its behavior respectively.

5. EXPERIMENTAL RESULTS

5.1 Implementation Setup

The implementation runs on Intel Core i7 with 6 GB of RAM. Eclipse Java version Juno R1 and Waikato Environment for Knowledge Analysis (WEKA) version 3.6.12 [16] are used to perform the experiments.

5.2 Performance Evaluation Metrics

In this experiment, the performance of each classifier of competing approaches is evaluated by the true positive rate, false positive rate, overall accuracy, and area under receiver operating characteristic curve that are defined as follows [8]:

True Positive Rate (*TPR*), the higher the better:

$$TPR = \frac{TP}{TP + FN} \quad (1)$$

False Positive Rate (*FPR*), the lower the better:

$$FPR = \frac{FP}{FP + TN} \quad (2)$$

The overall Accuracy (*ACC*), the higher the better:

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \quad (3)$$

Area under Receiver Operating Characteristic Curve (AUC):

It is the tradeoff between TPR and FPR, the higher the better.

5.3 Performance Analysis

In these experiments six different learning algorithms are used; ZeroR, Naïve Bayes, C4.5 Decision Tree known as (J48), Support Vector Machines with the training algorithm known as (SMO), JRip, and Random Forest as candidates for

this study. Also the proposed method CFFBE is applied using three ranges 100, 200, and 300 of the selected strongest features. These features are the strongest since they are the most common and frequent. All of the previous methods and the proposed method have been applied on the same data set due to the fact that most of the links used in the previous study provided by SpywareGuide.com were broken.

Table 3 shows the ACC results of experiments applied to six different classifiers. The experimental results show that there are peak values specify the maximum number of selected features depending on features weight in the data set.

The peak values were at the strongest 200 features in n-gram equal to 4 and 6, while in n-gram = 5 the peak values exist at the strongest 100 features. In these experiments selecting

n-gram equal to 5, the highest 100 features, and using J48 classifier resulting ACC equal to 99.98 % with reliable TPR equal to 99.9 %, FPR equal to 1.4 %, and AUC equal to 0.99. Table 4 shows ACC comparison between one of the previous studies [8] and the proposed method CFFBE [17].

Table 5 shows the ACC results of experiments for unknown spyware using CFFBE in all n-grams equal to 4, 5, and 6. Table 6 shows ACC comparison between one of the previous studies [8] and the proposed method CFFBE for unknown spyware. In these experiments after building this new class unknown spyware selecting n-gram equal to 5, the highest 100 features, and using J48 classifier resulting ACC equal to

99.91 % with reliable TPR equal to 99.9 %, FPR equal to

0.3 %, and AUC equal to 0.99. This new class will be included in an integrated anti-spyware system using design patterns so that if there will exist unknown spyware family recreated from the previous already existed spyware families the system will classify it as unknown spyware.

5.4 Maintainability Evaluation Metrics

The following metrics have been adapted in order to evaluate the contribution of design patterns in the proposedanti-spyware system design. These metrics are divided into amount of reuse and reusability metrics that are defined as follows [18]:

5.4.1 Amount of Reuse Metrics

Class Reuse Level (*CRL*): This metric is defined as the ratio between the number of Reused Pattern Classes (*RPC*) and the total number of classes in the designed system.

$$CRL = \frac{\sum_{i=1}^{n} RPC(C_i)}{n} \quad (4)$$

Attribute Reuse Level (*ARL*): This metric is defined as the ratio between the number of Reused Attributes (*RAT*) of pattern classes and the total number of attributes in the designed system.

$$ARL = \frac{\sum_{i=1}^{n} \sum_{j=1}^{m_i} RAT(a_{ij})}{\sum_{i=1}^{n} m_i} \quad (5)$$

Operation Reuse Level (*ORL*): This metric is defined as the ratio between the number of Reused Operations (*ROP*) of pattern classes and the total number of operations in the designed system.

$$ORL = \frac{\sum_{i=1}^{n} \sum_{q=1}^{k_i} ROP(op_{iq})}{\sum_{i=1}^{n} k_i} \quad (6)$$

5.4.2 Reusability Metrics

Class Reusability (*CR*): This metric is defined as the ratio between the number of Identified Pattern Classes (*IPC*) ina model designed without using patterns and the number of RPC in this model when designed using patterns.

$$CR = \frac{\sum_{i=1}^{n} IPC(C_i)}{\sum_{i=1}^{n} RPC(C_i)} \quad (7)$$

Attribute Reusability (*AR*): This metric is defined as the ratio between the number of Identified Attributes (*IAT*) of pattern classes in a model designed without using patterns and the number of RAT of pattern classes in this model when designed using patterns.

$$AR = \frac{\sum_{i=1}^{n} \sum_{j=1}^{m_i} IAT(a_{ij})}{\sum_{i=1}^{n} \sum_{j=1}^{m_i} RAT(a_{ij})} \quad (8)$$

Operation Reusability (*OR*): This metric is defined as the ratio between the number of Identified Operations (*IOP*) of pattern classes in a model designed without using patterns and the number of ROP of pattern classes in this model when designed using patterns.

$$OR = \frac{\sum_{i=1}^{n} \sum_{q=1}^{k_i} IOP(op_{iq})}{\sum_{i=1}^{n} \sum_{q=1}^{k_i} ROP(op_{iq})} \quad (9)$$

5.5 Maintainability Analysis

However, prior to measure the aforementioned metrics,the proposed anti-spyware data mining-based system has been redesigned without using design patterns. It is designed by four experts who have an experience in UML. Figure 5 illustrates the proposed system without design patterns in terms of number of classes, attributes and operations.

The *Main* class constitutes the core of the system design.

The *AntispywareSubject*, *AntispywareDatabase*, *Behavior*, *ISpyware*, and *Main* classes are known as identified pattern classes that represent a specific domain (i.e., blue classes). While other classes known as specific application classes

(i.e., white classes), represent different types of spyware families that are inherited from the domain identified classes. It is worth mentioning that the increasing number in the

specific application classes as the domain grows periodically motivates us to redesign the proposed system using design pattern concepts in order to enhance the proposed system reusability. Following the object oriented concepts, in order to add new behaviors or spyware family that inherits all the identified pattern classes properties and operations, the *Main* class should be updated accordingly. Periodical updating in the core of the system impedes its performance on runtime and impacts its maintainability. Figure 6 illustrates the proposed system after applying the design pattern concept in the proposed design. In this design, the classes *AntispywareSubject*, *AntispywareDatabase*, *Behavior*, *ISpyware*, and *Main* have been reused . Moreover, the classes *AbstractFactory*, *FactoryProducer*, *SpywareFactory*, and *BehaviorFactory* have been embedded in this design, all these classes constitute specific application elements which specialize the *Behavior* and *ISpyware* classes. These classes reuse all features of *BehaviorFactory* and *SpywareFactory* pattern classes. It is clear that, the reused classes in the new design have been rearranged such that remove the *Main* class from the core of the design and allow more inheritance in the pattern classes. Furthermore, most of the operations that was carried by the *Main* class in the previous design moved to the reusable design pattern classes (Observer, Abstract Factory, and Factory classes) i.e., data set preprocessing operations and the matching operations between the generated rules and the preprocessed data. Thus, this rearrangement reduces the processing at the *Main* class. Also, defining more specific application classes (i.e. new spyware families) becomes easier in the presence of the design pattern. Furthermore,all attributes and operations of the inherited classes have been reused without any exception. Indeed, the focus in this design is on the reusability of the identified pattern classes as shown in Figure 5 in order to define and accommodate more specific application classes without affecting the proposed system performance and maintainability.

Table 7 illustrates the calculated amount of reuse and reusability metrics. In this table the values of reuse and reusability metrics obtained for the design patterns

(Observer, Factory, and Abstract factory) which are used for designing the proposed anti-spyware real-time application.

On one hand, the values obtained for reuse metrics show that more than half of the classes, the attributes and the operations of anti-spyware are instantiated from the implemented design patterns. For example, the values of reuse metrics obtained show that 73% of classes (CRL = 0.73), 50% of attributes (ARL = 0.50) and 69% operations (ORL = 0.69) belonging to the model fragment relative to the anti-spyware framework are instantiated from these patterns. Thus, it is deduced a good level of reuse of the design pattern elements in the modeling of real-time applications i.e. anti-spyware application. Indeed, the values of reusability metrics obtained for anti-spyware

Table 3: ACC results of experiments using CFFBE

N-grams	Selected Features	ZeroR	Naïve Bayes	J48	SMO	JRip	Random Forest
4	100	94.40 %	99.89 %	99.92 %	99.90 %	98.70 %	99.92 %
	200	95.66 %	99.90 %	**99.95 %**	99.93 %	98.85 %	99.95 %
	300	95.16 %	99.88 %	99.91 %	99.89 %	98.60 %	99.91 %
5	100	95.38 %	99.95 %	**99.98 %**	99.97 %	99.40 %	99.98 %
	200	92.94 %	99.90 %	99.96 %	99.95 %	99.30 %	99.96 %
	300	91.09 %	99.61 %	99.94 %	99.93 %	99.22 %	99.94 %
6	100	92.13 %	99.78 %	99.89 %	99.85 %	98.32 %	99.89 %
	200	95.75 %	99.92 %	**99.95 %**	99.90 %	98.40 %	99.95 %
	300	95.44 %	99.47 %	99.86 %	99.80 %	98.35 %	99.86 %

Table 4: ACC comparison between the previous methods and the proposed CFFBE

The Classifier	N-grams	CFBE Raja et al. [8]	FBFE Raja et al. [8]	CFFBE Mohamed et al. [17]
ZeroR	4	86.25 %	90.13 %	95.66 %
	5	86.72 %	91.13 %	95.38 %
	6	85.54 %	92.79 %	95.75 %
Naïve Bayes	4	69.19 %	87.43 %	99.90 %
	5	67.29 %	95.13 %	99.95 %
	6	77.72 %	96.38 %	99.92 %
J48	4	86.25 %	94.74 %	99.95 %
	5	**86.72 %**	**95.13 %**	**99.98 %**
	6	85.54 %	97.84 %	99.95 %
SMO	4	83.41 %	96.50 %	99.93 %
	5	83.64 %	96.80 %	99.97 %
	6	82.70 %	97.19 %	99.90 %
JRip	4	86.25 %	94.73 %	98.85 %
	5	86.72 %	95.13 %	99.40 %
	6	85.78 %	97.89 %	98.40 %
Random Forest	4	82.46 %	92.31 %	99.95 %
	5	83.17 %	94.39 %	99.98 %
	6	81.75 %	95.44 %	99.95 %

Table 5: ACC results of experiments for unknown spyware using CFFBE

N-grams	Selected Features	ZeroR	Naïve Bayes	J48	SMO	JRip	Random Forest
4	100	79.60 %	98.96 %	99.52 %	99.52 %	96.72 %	99.52 %
	200	83.40 %	99.34 %	99.75 %	99.75 %	97.20 %	99.75 %
5	100	80.40 %	99.45 %	99.91 %	99.91 %	97.50 %	99.91 %
	200	78.41 %	98.49 %	99.83 %	99.83 %	96.80 %	99.83 %
6	100	79.98 %	98.81 %	99.79 %	99.79 %	96.70 %	99.79 %
	200	90.86 %	98.89 %	99.81 %	99.81 %	97.50 %	99.81 %

Table 6: ACC Comparison between the previous methods and the proposed CFFBE for unknown spyware

The Classifier	N-grams	CFBE Raja et al. [8]	FBFE Raja et al. [8]	CFFBE (unknown spyware)
ZeroR	4	86.25 %	90.13 %	83.40 %
	5	86.72 %	91.13 %	80.40 %
	6	85.54 %	92.79 %	90.86 %
Naïve Bayes	4	69.19 %	87.43 %	99.34 %
	5	67.29 %	95.13 %	99.45 %
	6	77.72 %	96.38 %	98.89 %
J48	4	86.25 %	94.74 %	99.75 %
	5	**86.72 %**	**95.13 %**	**99.91 %**
	6	85.54 %	97.84 %	99.81 %
SMO	4	83.41 %	96.50 %	99.75 %
	5	83.64 %	96.80 %	99.91 %
	6	82.70 %	97.19 %	99.81 %
JRip	4	86.25 %	94.73 %	97.20 %
	5	86.72 %	95.13 %	97.50 %
	6	85.78 %	97.89 %	97.50 %
Random Forest	4	82.46 %	92.31 %	99.75 %
	5	83.17 %	94.39 %	99.91 %
	6	81.75 %	95.44 %	99.81 %

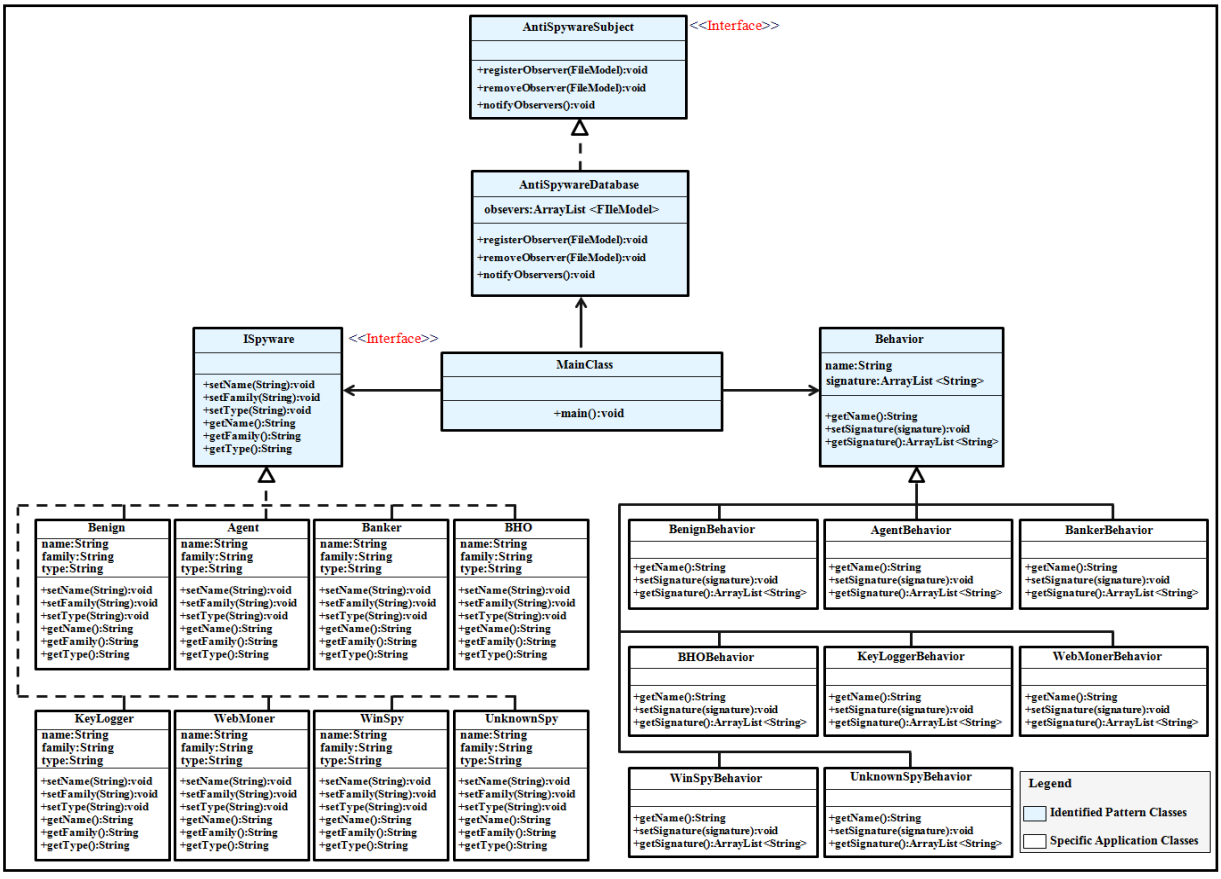

Fig 5: Proposed anti-spyware system class diagram without using design patterns

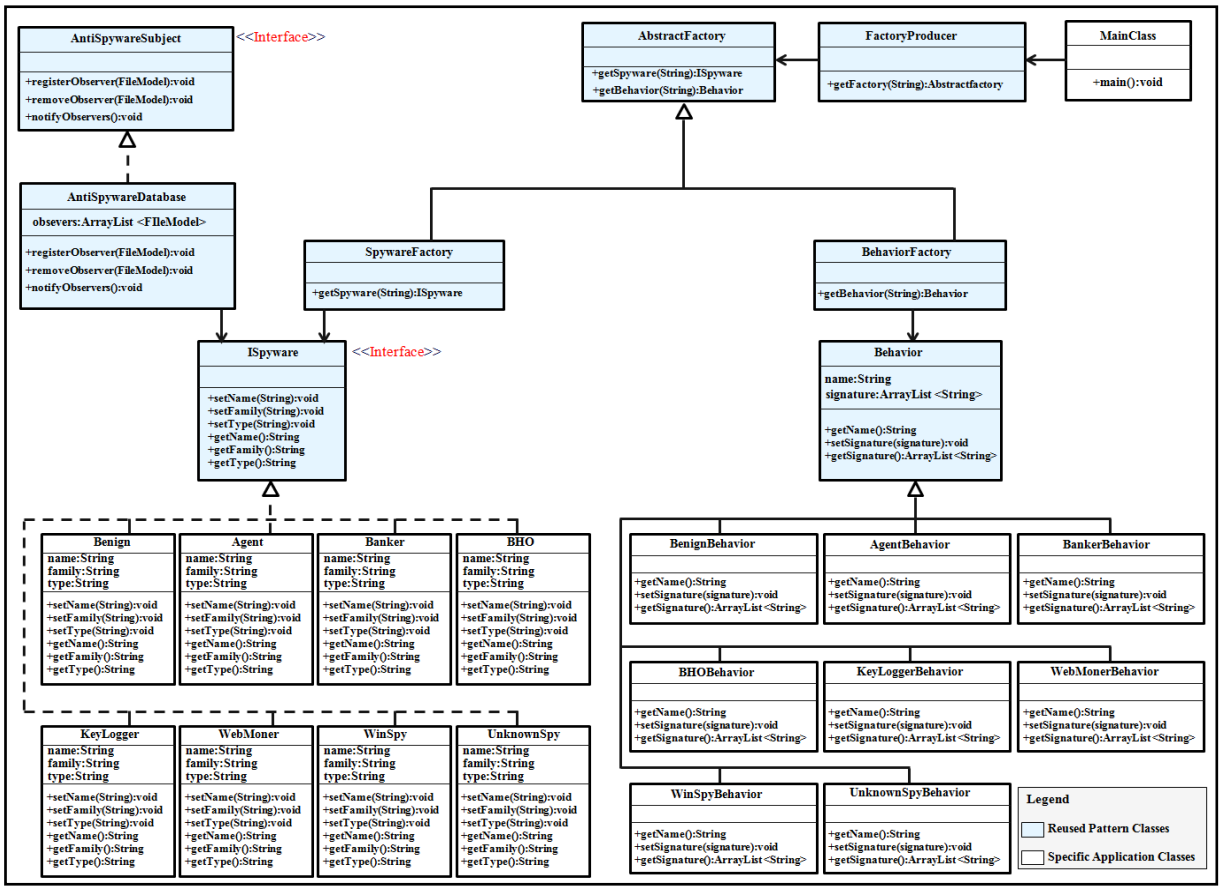

Fig 6: Proposed anti-spyware system class diagram using design patterns

application show that 63% of the reused classes are identified (CR = 0.63), all reused attributes of the design patterns are identified (AR = 1), and 75% of the reused operations are identified (OR = 0.75). This means that the reuse of this pattern is interesting in the real-time domain because it adequately represents the concepts of data acquisition functionality. The reusability metric values presented approach to 1, it indicates that the majority of pattern participants (classes, attributes and operations) are recognized in the application model; this means that the adopted design patterns cover the anti-spyware data mining-basedsystem concepts.

Table 7: Calculated reuse and reusability metrics

	Metrics	Value
Reuse metrics	Class Reuse Level (CRL)	**8/11 = 0.73**
	Attribute Reuse Level (ARL)	**3/6 = 0.50**
	Operation Reuse Level (ORL)	**20/29 = 0.69**
Reusability metrics	Class Reusability (CR)	**5/8 = 0.63**
	Attribute Reusability (AR)	**3/3 = 1**
	Operation Reusability (OR)	**15/20 = 0.75**

6. CONCLUSIONS

The main challenge in existing anti-spyware systems is to guarantee the performance and maintainability of these systems. In this paper, a data mining-based anti-spyware system using design patterns was proposed. Adoption of data mining based techniques proved its efficiency in detecting new types or unknown families of spyware. Moreover, integrating design pattern approach in the proposed system design contributes significantly in the proposed system maintainability. Experimental results and findings have shown that, the adopted methods to be more effective by a staggering 99.98% overall detection accuracy. Furthermore, in these experiments, a new class named as *unknown spyware* has been presented, this class acts as a general finger print for any spyware. These experiments show that all the new families fall into this class of spyware at an overall percentage detection accuracy of 99.91%. In accordance to the performance of the design patterns contribution in the proposed system, the conducted experiments evaluate the quality of the design patterns and determine whether the used patterns represent the concepts that suit an anti-spyware domain. This assessment achieved by computing the amount of reuse metrics and reusability metrics of the design patterns. As expected, adoption of design patterns achieved the system reusability in terms of amount of reuse metrics and reusability metrics with values approach to 1 that indicates the proposed design patterns cover the anti-spyware data mining-based system concepts.

7. REFERENCES

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] E. B. Fernandez "A Methodology for Secure Software Design", International Conference on Software Engineering Research and Practice, USA, 2004.

[8] 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.

[9] 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.

[10] Zongqu Zhao, Junfeng Wang, and Jinrong Bai1 "Malware detection method based on the control-flow construct feature of software", International Journal of The Institution of Engineering and Technology (IET) on Information Security, Vol. 8, pp. 18-24, England, 2013.

[11] J. Yoder and J. Barcalow "Architectural patterns for enabling application security", In Proceedings of the 4th Conference on Patterns Language of Programming (PLoP'97), USA, 1997.

[12] Schumacher and U. Roedig "Security engineering with patterns", In Proceedings of the 8th Conference on Patterns Language of Programming (PLoP'01), USA, 2001.

[13] M. Hafiz, P. Adamczyk, and R. E. Johnson "Towards an Organization of Security Patterns", IEEE International Conference on Software, Vol. 24, pp. 52-60, USA, 2007.

[14] Mohamed Adel Sheta, Mohamed Zaki, Kamel AbdEl Salam El Hadad, and H. Aboelseoud M. "Design and Implementation of Anti Spyware System using Design Patterns", International Journal of Computer Applications, Vol.123, No.2, pp.9-13, USA, 2015.

[15] VX Heavens, http://vx.netlux.org, accessed 01-10-15.

[16] Ian H. Witten, Eibe Frank, and Mark A. Hall, Data Mining: Practical Machine Learning Tools and Techniques, 3rd edition, San Francisco, CA, Morgan Kaufmann Publishers, Inc., USA, 2011.

[17] Mohamed Adel Sheta, Kamel Abd El Salam El Hadad, and H. Aboelseoud M. "Data Mining-basedAnti-spyware System Using a Hybrid of Common Feature-based Extraction And Frequency-based Feature Extraction Approaches", International Journal of Applied Engineering Research (IJAER), Vol. 10, No. 24, pp. 45597-45605, India, 2015.

[18] Saoussen Rekhis, Hela Marouane, Rafik Bouaziz, Claude Duvallet, and Bruno Sadeg "Metrics for Measuring Quality of Real-time Design Patterns", In the 8th International Conference on Software Engineering Advances (ICSEA), pp. 163-168, France, 2013.