I. INTRODUCTION

Federal Trade Commission Staff Report in USA defines spyware as:
"Software that aids in gathering information about a person or organization without their knowledge and that may send such information to another entity without the consumer's consent, or that influences some control over a computer without the consumer's knowledge." [1]

Spywares collect information from the user and send it to a third party. They are capable of storing personal details of the user, authentication credentials, saving screenshots, taking images and stealing the user files. Spyware does not necessarily spread in the same way as a virus or worm because infected systems generally do not attempt to transmit or copy the software to other computers. Instead, spyware installs itself on a system by deceiving the user or by exploiting software vulnerabilities. Most spyware is installed without users' knowledge, or by using deceptive tactics. Spyware may try to deceive users by bundling itself with desirable software [2].

Spyware threat has emerged as most complex and sophisticated threat over the past few years. The problems caused by Spyware are now even getting more severe[3]. A spyware application is typically difficult to remove once it has been installed on a computer system and it can seriously degrade system performance and compromise the privacy of the user [4].

II. EXISTING SOLUTION

Traditionally two approaches have been presented for the purpose of Spyware detection: Signature-based Detection and Heuristic-based Detection. These approaches perform well against known Spyware but have not been proven to be successful at detecting new Spyware[5].

A. Signature-Based Detection

Signature based methods maintain a database consisting of unique strings or specific features called Signatures. For detection it extracts specific features from binaries and compares it with existing database. This method is not good enough to detect new malicious executables.

B. Heuristic-Based Detection

Heuristic classifiers are generated by a group of virus experts to detect new malicious programs. This kind of analysis can be time-consuming and often fails to detect new malicious executables [6].

III. DESIGN

A. GOALS OF APPLICATION

This application allows us to detect whether a particular executable is spyware or not prior to their installation.
Existing antispyware programs require frequent updates to their databases from their remote server. In their contrast our software does not require updates for database from the remote server.

B. WORKFLOW

![Workflow Diagram]

IV. IMPLEMENTATION

The steps are described below in a detailed manner for easy understanding.

A. DATA COLLECTION

The data set consists of two classes of binary files:

- Benign files
- Spyware files

These files were collected from various sources.

B. BYTE SEQUENCE GENERATION

VI. TEST RESULTS

Here we extract byte sequence into byte array, then convert it into Hexadecimal dumps using Java Programming.

C. N-GRAM GENERATION

The above generated hexadecimal dumps are converted into “n” of fixed size and stored in HashMap, for later updation in the database.

D. FEATURE EXTRACTION

In this step, we extract the features by using Frequency-based Feature Extraction (FBFE) approach.

E. FEATURE REDUCTION

The features with high frequency are being considered for training the classifier. The features with low frequency are ignored.

F. MODEL TRAINING

In this step, the classifier is trained. The frequency of reduced features is read from the database created before and passed to a Java Method. This method implements maximum posterior hypothesis of Naive Bayesian Algorithm to classify the binary executables.

V. CLASSIFIER ALGORITHM

The Bayesian Classification represents a supervised learning method as well as a statistical method for classification. This Classification is named after Thomas Bayes (1702-1761), who proposed the Bayes Theorem [7].

1) Bayes Theorem: Bayesian reasoning is applied to decision making and inferential statistics that deals with probability inference. It is used the knowledge of prior events to predict future events [7].

2) Equation:

\[
P(h/D) = \frac{P(D|h)P(h)}{P(D)}
\]

P(h) : Prior probability of hypothesis
P(D) : Prior probability of training data D
P(h/D) : Probability of h given D
P(D/h) : Probability of D given h

3) Maximum A Posteriori (MAP) Hypothesis: Generally we want the most probable hypothesis given the training data hMAP = arg max P (h/D) (where h belongs to H and H is the hypothesis space) [7].

\[
h\text{MAP} = \text{arg max } \frac{P(D|h)P(h)}{P(D)}
\]

hMAP = arg max P (D/h) P (h)
Fig. 2  Homescreen

Fig. 3  GUI for Scanning Input Directory

Fig. 4  GUI for Result where Spyware is not detected

Fig. 5  GUI for Result where Spyware is Detected
VII. CONCLUSION

In this paper we have proposed and designed a spyware detection system using Data Mining approach that detects new and unseen spyware by scanning the executables. Future work involves collection of a bigger dataset of Spyware and Benign files, implementation of various Classifier Learning Algorithms.

VIII. ACKNOWLEDGMENT

We are thankful to our Principal Dr. Shrikant Kallurkar, Project Coordinator Prof. Deepali Maste and other senior faculties of Computer Department for technical assistance and feedback through discussions. Our thanks to some of our colleagues who contributed towards development of the flowchart, leading to a success of this project.

IX. REFERENCES