International Journal for Housing Science and Its Applications
Publish August 6, 2025. Volume 46, Issue 3 Pages 7900-7913

https,//doi.org/10.70517/ijhsa463676

Research on Malware Behavior Detection and Protection Based on Bayesian Inference Modeling

Xia Wu[1,*]
[1] Department of Information Engineering, Henan Vocational College of Water Conservancy and Environment, Zhengzhou, Henan, 450008, China
Corresponding authors: (e-mail: wx19750101@yeah.net).

Abstract With the increasing cyber security threats, malware attacks pose a serious challenge to the security of information systems. This study proposes a malware behavior detection and protection method based on Bayesian inference model. Accurate detection and protection against malware behavior is achieved by constructing a state space model with Markov chain process, combined with Bayes theorem. Experimental results show that the method in this paper has high accuracy in malware attack detection. In the simulation experiments, using the Markov chain algorithm for parameter estimation, the relative error of the final malware location parameter is 0.0366%. Meanwhile, in the algorithm error adaptation analysis, the parameter estimation interval gradually increases with the increase of the total error standard deviation, but through data cleaning, the error adaptation is significantly improved and the parameter estimation interval is narrowed. Experiments show that the method can effectively improve the recognition accuracy of malware attacks, especially in the case of concurrent attacks, the decision rate and detection rate are better than the traditional method. The study proves that the proposed Bayesian inference-based malware behavior detection and protection method has high accuracy and robustness, and can effectively identify and defend against malware attacks.

Index Terms bayesian inference, malware, behavior detection, state space model, Markov chain, data cleaning

I. Introduction

With the continuous development and popularization of the Internet, computer viruses, Trojan horses, worms, malware, and other hacking attacks continue to emerge, and gradually expand from traditional personal computers to the field of embedded devices such as mobile devices, industrial control systems, and so on [1]-[4]. These malware threaten people's system security, and may also lead to serious consequences such as property or personal information leakage [5], [6]. In order to prevent these threats, the research and application of malware behavior detection techniques are getting more and more attention [7]. And malware detection and protection based on Bayesian inference modeling is becoming a hot research topic and application trend because of its effectiveness and efficiency [8], [9].

Bayesian inference model is a statistical inference method based on Bayes' theorem, which makes inferences and predictions about unknown parameters by utilizing prior knowledge and observed data [10], [11]. The model has been widely used in various fields, including natural language processing, machine learning, and artificial intelligence [12]. The basic principle of Bayesian inference models is based on Bayes' theorem, which describes how to update the estimation of the probability of occurrence of an event based on observed data, given the prior probability of that event occurring [13]-[15]. In malware behavior detection and protection, we can use known attack samples and observed data as a training set, calculate the probability that an unknown sample belongs to a malware attack through a Bayesian inference model, and take protective measures to improve user information security [16]-[19].

With the rapid development of the Internet, malware has become one of the most serious threats in network security. Malware not only steals user data and personal privacy, but also causes serious consequences such as system crash and network paralysis. Traditional malware detection methods usually rely on virus libraries and rule matching techniques, however, these methods have the limitation of not being able to effectively deal with new types of malware, especially when the behavior of malware becomes more hidden and complex, which is often difficult to be detected by existing techniques. To address this challenge, researchers have proposed a detection method based on behavioral analysis, which is capable of identifying malicious attacks by analyzing the behavioral characteristics of the software with better adaptability and robustness.

Bayesian inference, as a probability-based inference method, can provide a more accurate judgment basis for malware detection by combining a priori information and observation data. Bayesian inference method has unique

advantages in dealing with uncertainty and dynamic updating, especially in the case of rapid and complex changes in malware behavior, which can provide effective decision support. In this paper, a new malware behavior detection method is proposed based on the Bayesian inference model, combined with the state space model and Markov chain process. With this method, the system is able to recognize and prevent malware attacks in real time in a dynamically changing network environment.

In this study, a malware behavior detection and protection model is constructed based on the Bayesian inference framework. Firstly, a state space model is established using Bayes theorem, and malware behavior is represented as a state sequence with Markov chain attributes; secondly, a parameter evaluation mechanism based on the distance function is designed, and equivalent sampling is combined with the MCMC method to realize the accurate estimation of malware parameters; thirdly, a higher-order cumulative volume slice function is proposed to identify the data features of malware attack behavior, and FIR filtering is used for data interference suppression; finally, based on the squared prediction error method to realize autonomous protection against malware attack behaviors. Through theoretical analysis and experimental verification, the effectiveness and superiority of the proposed method in malware behavior detection and protection are proved.

II. Bayesian inference and methods

II. A.Bayes' Theorem

Bayes' theorem describes how prior information can be combined with sample information in a probabilistic manner. Z is an observation of X, and when given Z, the posterior probability density function of X can be computed by Bayes' theorem:

$$p(X \mid Z) = \frac{p(Z \mid X)p(X)}{p(Z)} \tag{1}$$

where $p(X \mid Z)$ is the conditional probability of X given Z, also known as the posterior probability, since $p(X \mid Z)$ is related to the value of the specific Z [20]. $p(Z \mid X)$ is the conditional probability of Z given X, also known as the likelihood probability. $p(X)$ and $p(Z)$ are the marginal probability densities of X and Z.

II. B.Bayesian reasoning

II. B. 1) State space models

A state space model is a complete description of a system that fully characterizes all the dynamics of the system. It realizes the unification of different forms of system description and is suitable for describing complex dynamic systems. Its appearance promotes the development of control theory, realizing the transition from classical control theory to modern control theory.

A set of independent variables that can fully characterize the dynamics of a system is called the state variables of the system, which are the internal variables of the system. The column vector $X(t) = [x_1(t), x_2(t), \ldots, x_n(t)]^T$ consisting of the state variables is called the state vector, and the space in which the state vectors take values is called the state space. The first order differential equations consisting of the state variables out of the system are organized as state equations. The state model is composed of state equations and output equations:

$$\dot{x} = f(x,u,t), t \geq t_0 \tag{2}$$

$$y = g(x,u,t), t \geq t_0 \tag{3}$$

In the state equation, $x(t)$ is the state vector of the system and $u(t)$ is the input vector. In the observation equation, $y(t)$ is the observation vector.

The state equations reflect the causal relationship between the state variables and the input variables in the system, as well as the relationship between the change in each state variable with respect to time. The output equation is a functional relationship between that output and the state variables and inputs, given a specified output [21]. It reflects the causal relationship between the output variables and the state variables and inputs in the system.

Equation (4) determines the conditional transition probability i.e. the probability of predicting the current state given the state and all observations at the previous moment and equation (5) determines the likelihood probability of the current observation given the current state prediction. The expression is given below:

Conditional transition probability:

$$p(x_n \mid x_{n-1}, y_{1:n-1}) \tag{4}$$

Likelihood probability:

$$p(y_n \mid x_n) \tag{5}$$

x_n is the state of the system at moment n and $y_{1:n-1}$ is the observation at moment $1:n-1$.

II. B. 2) Markov chains

A Markov chain, also known as a Markov process, is a stochastic process that is discrete in time and state. One of the most fundamental properties of a dynamic spatial model is that it has the Markov chain property. If a discrete stochastic process $\{x\}$ has the Markov chain property, then its conditional probability has to be satisfied:

$$p(x_{k+1} \mid x_1, ..., x_k) = p(x_{k+1} \mid x_k) \tag{6}$$

That is, future states are not disturbed by past states. The Markov chain property expresses the fact that the "future" is independent of the "past", provided that the "present" is known; this property is also known as the absence of a posteriori effects.

A system model with the Markov chain property can be described as:

$$x_{k+1} \sim p_\theta(x_{k+1} \mid x_1, ..., x_k) = p(x_{k+1} \mid x_k) \tag{7}$$

where $p_\theta(x)$ denotes a set of probability density functions with parameter θ, and $p(x_{k+1} \mid x_k)$ expresses the change of the state variables during the time variation. Further, our task is to compute these system variables from the measured variables to obtain a model of the system expressed in terms of the parameter θ, i.e., the system identification problem [22].

The state variable $\{x\}$ process is an unobservable Markov process, so only the observed variables can be utilized to obtain information about these processes, and y_k satisfies the measurement model:

$$y_k \sim p_\theta(y_k \mid x_k) \tag{8}$$

In addition, the observed variables are assumed to be independent of each other in time, which can be obtained:

$$p_\theta(y_k, ..., y_N \mid x_k, ..., x_N) = \prod_{t=k}^{N} p_\theta(y_t \mid x_k, ..., x_N) = \prod_{t=k}^{N} p_\theta(y_t \mid x_t) \tag{9}$$

II. B. 3) Bayesian inference of posterior distributions

Bayesian inference can also be thought of as a dynamic process because the process starts with a priori information, collects evidence in the form of sample information, and ends with a posterior distribution. This posterior distribution can be combined with the new sample information as a new prior distribution This is the Bayesian learning model for the conversion process from prior to posterior.

The posterior distribution is the starting point and the key to Bayesian statistical inference, for the general parameter prior distribution, its posterior distribution is quite complex, it is difficult to be included in the category of currently known statistical distributions, so it is common to use computer-based methods to calculate or approximate the characteristics of the parameter posterior distribution.

The basic steps of Bayesian inference:

(1) Choose a probability density function $f(\theta)$ to represent our beliefs about a parameter θ prior to obtaining the data, which we call the prior distribution.

(2) Choose a model $f(x \mid \theta)$ to reflect our beliefs about x given a parameter θ.

(3) When the data $X_1, X_2, ..., X_n$, we update our beliefs and compute the posterior distribution $f(\theta \mid X_1, ..., X_n)$, and obtain point estimates and interval estimates from the posterior distribution.

Bayesian inference is the use of observations to correct and update the prior probability to make it closer to the true value. Suppose two events X and Y, X is unknown and Y is an observation obtained through X. The conditional probability function of Y is $p(Y \mid X)$, the prior probability of X is $p(X)$, and an X is generated from $p(X)$, which then generates sample observations $Y = (y_1, ..., y_n)$ and the posterior probability density of X:

$$p(X \mid Y) = \frac{p(Y, X)}{p(Y)} = \frac{p(Y \mid X)p(X)}{p(Y)} = \frac{p(Y \mid X)p(X)}{\int p(Y \mid X)p(X)dX} \tag{10}$$

where the joint conditional probability function for sample Y is also known as the likelihood function:

$$p(Y \mid X) = \prod_{i=1}^{n} p(y_i \mid X) \tag{11}$$

Joint probability distribution of samples Y and X:

$$p(Y, X) = p(Y \mid X)p(X) = p(X \mid Y)p(Y) \tag{12}$$

$p(Y)$ is the marginal probability distribution of Y:

$$p(Y) = \int p(Y, X)dX = \int p(Y \mid X)p(X)dX \tag{13}$$

II. B. 4) Bayesian decision making

Bayesian decision theory method is a basic method in the statistical model decision-making, its basic idea is: in the case of incomplete intelligence, the use of existing a priori probability and class of conditional probability density estimation of part of the unknown state with, and then converted to a posteriori probability with Bayesian formula and the probability of occurrence of the correction, and then finally according to the size of the a posteriori probability of making the optimal decision and classification.

Given a m pattern class $(\omega_1, \omega_2, ..., \omega_m)$ for the classification task and the statistical distribution of each class in this n for the feature space, it is a problem to distinguish to which of these m classes of samples the sample to be recognized x belongs. Suppose that a sample to be recognized is described by n attribute observations, called n features, thus forming an n-dimensional feature vector, and all possible ranges of values of this n-dimensional feature vector form an n-dimensional feature space. The statistical distribution of the feature space:

(1) $\omega_i, i = 1, 2, ..., m$ with prior probability $p(\omega_i)$.

(2) Class conditional probability density function: $p(x \mid \omega_i)$.

(3) Posterior probability: generate m conditional posterior probabilities $p(\omega_i \mid x), i = 1, 2, ..., m$ that is, for a feature vector x, each conditional a posteriori probability $p(\omega_i \mid x)$ represents the probability that the unknown sample belongs to a particular class ω_i.

II. C.Malware Behavior Detection Model Based on Bayesian Inference Modeling

II. C. 1) Distance function

The parameter evaluation problem of Bayesian inference can be realized by defining the distance function of the parameter to be evaluated, whose distance function generally includes the baseline output and the corrected output. The setting of the distance function of the parameters to be evaluated is crucial, and by combining the distance function into the likelihood function of Bayesian inference, the parameter evaluation problem can be converted into a maximum likelihood function solution problem [23]. Specifically, there exist four common forms of setting the distance function of the parameters to be evaluated:

(1) The distance function $D_1(x)$ defined on the basis of plausible parameters is shown in Eq. The distance function consists of the baseline output of the plausible parameters and the corrected output. The plausible parameters can often be obtained by direct measurement and can generally be considered accurate and plausible. As shown in Eq. the baseline output YB_1 of the plausible parameter is equal to the measured value of the parameter. And the calibration value YC_1 of the trusted parameter consists of the measured value of the parameter and the compensation value of the parameter, including the measured value of the parameter to be evaluated and the value of its offset constant, as shown in Eq:

$$D_1(x) = \sum_{n=1}^{N} \sum_{i=1}^{I} (YB_{1,i} - YC_{1,i})^2 \tag{14}$$

$$YB_{1,i} = S_i \tag{15}$$

$$YC_{1,i} = g_1(M_1, M_2, ..., M_p, x_1, x_2, ..., x_q) \tag{16}$$

where $D_1(x)$ is the distance function defined based on the plausible parameters. YB_1 and YC_1 are the baseline and corrected outputs of the plausible parameters, respectively. S is the plausible parameter measurement. M is the parameter measurement value. x is the parameter offset constant, including the sensor measurement bias and the magnitude of equipment failure severity level. N and I are the parameter measurement dataset and the

number of evaluated parameters, respectively. n is the serial number of the parameter measurement data set. i,p and q are the serial numbers of the evaluation parameters. g_1 is the calibration output function defined based on plausible parameters.

(2) The distance function $D_2(x)$ defined based on the parameters to be evaluated is shown in equation (17). Accordingly, the baseline output value YB_2 of the parameter to be evaluated is defined by the other parameter measurements and the corresponding offset constants. And the distance function correction output value YC_2 defined based on the parameter to be evaluated consists of the measured value of this evaluated parameter and an offset constant, which is usually equal to the cumulative value of the measured value of the evaluated parameter and the offset constant:

$$D_2(x)=\sum_{n=1}^{N}\sum_{i=1}^{I}(YB_{2,i}-YC_{2,i})^2 \tag{17}$$

$$YB_{2,i}=f_2(M_1,M_2,...,M_p,x_1,x_2,...,x_q) \tag{18}$$

$$YC_{2,i}=g_2(M_i,x_i) \tag{19}$$

where $D_2(x)$ is the distance function defined based on the parameters to be evaluated. YB_2 and YC_2 are the baseline and corrected outputs of the parameters to be evaluated, respectively. f_2 and g_2 are the benchmark output function and the corrected output function defined based on the parameters to be evaluated, respectively.

(3) The distance function $D_3(x)$ defined based on the non-evaluated parameters is shown in Eq. Different from the distance function $D_2(x)$, at this time the parameters to be evaluated are no longer the dependent variables of the benchmark output function, but the independent variables of its benchmark output function. Where the benchmark output value YB_3 of the non-evaluated parameter consists of other parameter measurements including the parameter to be evaluated and the corresponding offset constants. The corrected output value YC of the distance function defined based on the non-evaluated parameter consists of the measured value of the non-evaluated parameter and an offset constant, which is usually equal to the cumulative value of the measured value of the parameter and the offset constant.

$$D_3(x)=\sum_{n=1}^{N}\sum_{i=1}^{I}(YB_{3,i}-YC_{3,i})^2 \tag{20}$$

$$YB_{3,i}=f_3(M_1,M_2,...,M_p,x_1,x_2,...,x_q) \tag{21}$$

$$YC_{3,i}=g_3(M_i,x_i) \tag{22}$$

where $D_3(x)$ is the distance function defined based on the non-evaluated parameters. YB_3 and YC_3 are the baseline and corrected outputs of the non-evaluated parameters, respectively. f_3 and g_3 are the benchmark output function and corrected output function defined based on non-evaluated parameters, respectively.

(4) The distance function $D_4(x)$ based on the benchmark expansion is shown in Eq. This distance function is based on the available measurement data, and one or more of the above three distance functions are selectively added to expand the number of equations for the parameters to be evaluated in the distance function, increase the binding force of the distance function on the parameters to be evaluated, and thus weaken the negative impact of parameter underdetermination on the calibration results. Parameter underdetermination means that the given number of equations is not sufficient to define all the parameters to be evaluated in the distance function, resulting in the existence of multiple solutions of the equations, which leads to poor calibration results.

$$D_4(x)=\sum_{n=1}^{N}\sum_{i=1}^{I}\left[\left(YB_{1,i}-YC_{1,i}\right)^2+\left(YB_{2,i}-YC_{2,i}\right)^2+\left(YB_{3,i}-YC_{3,i}\right)^2\right] \tag{23}$$

where $D_4(x)$ is the distance function of the parameter to be evaluated based on the benchmark expansion.

II. C. 2) Fundamentals of Bayesian inference

Bayesian inference the flow of this paper's algorithm is shown in Figure 1. The method usually consists of two parts: Bayesian inference, this paper algorithm equivalent sampling.

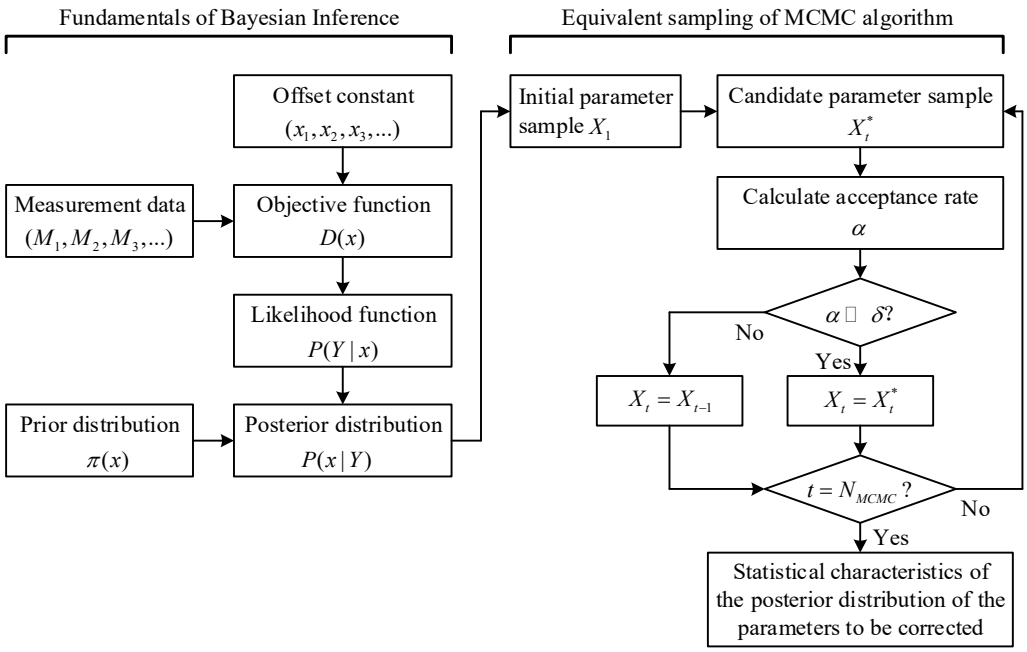

Figure 1: The process of the Bayesian Inference MCMC algorithm

The basic principle of Bayesian inference is to derive a set of values such that there is as close a match as possible between the corrected value of the parameter to be corrected and the true value of the parameter to be corrected. The posterior distribution $P(x|Y)$ of the offset constant x of the parameter to be corrected is jointly defined by the full probability function $P(Y)$, the prior distribution $\pi(x)$, and the likelihood function $P(Y|x)$, whose basic mathematical expression is shown in Eq. Based on the central limit theorem, the prior distribution $\pi(x)$ of each offset constant is defined to obey a normal distribution. Where the full probability function $P(Y)$ is a normalized constant as shown in Eq. The likelihood function $P(Y|x)$ is usually set to a normally distributed probability density function with zero mean. The distance function $D(x)$ in Eq. represents the difference between the baseline function and the calibration function.

$$P(x|Y) = \frac{P(Y|x) \times \pi(x)}{P(Y)} \tag{24}$$

$$P(Y) = \int P(Y|x)\pi(x)dx \tag{25}$$

$$P(Y|x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{1}{2\sigma^2} D(x)\right] \tag{26}$$

where x is the offset constant of the parameter to be corrected. Y is the observed value. $P(x|Y)$ is the posterior distribution function of the offset constant. $P(Y)$ is the full probability function. $P(Y|x)$ is the likelihood function of the offset constant. $\pi(x)$ is the prior distribution function for the offset constant. σ is the standard deviation. $D(x)$ is the distance function of the offset constant.

II. C. 3) Equivalent sampling of the algorithm in this paper

In this paper, the MCMC method is used to replace the complex integration problem, and the information of the parameters to be corrected is obtained by equivalent sampling of the MCMC method. Usually, the execution steps of equivalent sampling of the algorithm in this paper are as follows:

Step 1: Select the initial parameter samples X_1 of the Markov chain. The sample of the initial parameters consists of the means of the prior distributions of the parameters to be corrected.

Step 2: Assume that the probability density function $f(X_t^*|X_{t-1})$ adopts a Gaussian probability density function which is centered on the previous sample of parameters X_{t-1} and whose covariance matrix consists of the standard deviations of the parameters to be corrected. In the t th step iteration, a candidate parameter sample X_t^* is selected from the hypothesized probability density function. The candidate parameter sample X_t^* is obtained by adding 1 random variable ε to the previous candidate sample X_{t-1}, as shown in Eq:

$$X_t^* = X_{t-1} + \varepsilon, \varepsilon = [-g, g] \tag{27}$$

where g is a random number.

Step 3: Calculate the acceptance rate α of the candidate parameter sample according to Eq.

$$\alpha = \min\left\{1, \frac{P(X_t^* \mid Y, \pi(x))}{P(X_{t1} \mid Y, \pi(x))} \times \frac{f(X_{t1} \mid X_{t1}^*)}{f(X_t^* \mid X_{t1})}\right\} \tag{28}$$

where, $P(X_t^* \mid Y)$ is the posterior distribution function of the candidate parameter sample X_t^*. $P(X_{t-1} \mid Y)$ is the posterior distribution function of the previous parameter sample X_{t-1}. $f(X_{t-1} \mid X_t^*)$ denotes the Gaussian probability density function centered at X_t^*. $f(X_t^* \mid X_{t-1})$ denotes the Gaussian probability density function centered at X_{t-1}.

Step 4: Select a new sample of parameters based on the acceptance rate α. In each iteration, compare the acceptance rate α with a random number δ, which is randomly generated between 0 and 1. If the acceptance rate α is greater than or equal to the random number δ, the parameter sample is X_t^* in the t th iteration. Otherwise, the parameter sample is X_{t-1}.

Step 5: Repeat steps 2 through 4 until the prescribed sampling setpoint Nacwc is reached to obtain equivalent samples of the posterior distribution.

Step 6: Perform statistics on the parameter samples to obtain the statistical characteristics of the parameter to be corrected, including the mean and standard deviation.

III. Methodology for autonomous protection against malware attacks

III. A. Higher-order cumulant slice function

The Fourier transform method is adopted on the data to obtain the following equation for the oscillation decay law of the data:

$$S(t) = mt + \sqrt{am} B^h(t) \tag{29}$$

where a and $B^h(t)$ denote the inter-domain coefficient of variance of the data as well as the data identification function about the malware attack behavior, respectively, and mt denotes the single-component transmission information.

The information is obtained by utilizing the oscillatory decay law of the data:

$$\begin{cases} \omega(t) = \bar{\omega}(t) + iy(t) = a(t)e^{\psi} + \xi(t) \\ \bar{\omega}(t) = \dfrac{ma}{B^k(t)} \\ m = \dfrac{1}{i}\sum_{i=1}^{n} x_i \end{cases} \tag{30}$$

where $\bar{\omega}(t)$ and $iy(t)$ denote the real part of the data time series and the imaginary part of the data time series during information transmission, $a(t), e^{\psi}$ and $\xi(t)$ denote the phase randomly varying amplitude, the interference vector, and the filter tap coefficient, respectively.

In the information transmission, the higher-order cumulant posterior search method is selected to search the network output parsing model to identify the malware attack behavior data features, and the higher-order cumulant posterior processing operator is utilized to identify the energy aggregation and noise suppression characteristics contained in the malware attack behavior data features, in which the higher-order cumulant slicing function is as follows:

$$\hat{c}_{4s}(n,\tau) = \hat{c}_{4s}(n,\tau) + vr\sum_{j=0}^{\infty} h(j)h^3 e \tag{31}$$

After the energy support vector let X^{-} represent the set of negative class operators: X^{+} represents the set of positive class operators: s_{i1} represents the affiliation function with the following expression:

$$s_{i1}=\begin{cases}\dfrac{d_i^{*}+\delta}{\max d_i^{*}}, y=+1, x_i^{*}\in X^{*}\\ \dfrac{d_i^{-}+\delta}{\max d_i^{-}}, y=-1, x_i^{-}\in X^{-}\end{cases} \tag{32}$$

where δ is the smaller positive number. d_i^{-} represents the distance that exists between the negative class center and the negative class sample. d_i^{+} represents the distance that exists between the positive class center and the positive class samples, which are calculated as follows, respectively:

$$d_i^{-}=|x_i^{-}-m^{-}| \tag{33}$$

$$d_i^{+}=|x_i^{+}-m^{+}| \tag{34}$$

where v_r and $h(j)$ denote the malware attack behavior data feature bee degree and the corresponding diagonal slicing operator, h^3e and $\hat{c}_4$, (n,τ) denote the transfer function of the e-stage filters as well as the alternative data consisting of the test data components via autocorrelation feature matching theory, respectively.

The network filtered data output from the post-focused search using higher-order cumulative volume features can accurately identify the data features of malware with attack behaviors during the information transmission process.

III. B. Data interference suppression based on FIR filtering

Separation of malware attack behavior data features by higher order cumulative volume back path is formulated as follows:

$$\begin{cases}v_x(t)=v_0+2\beta t\\ R_p(u)=X_p(u)+vr(v_0+\beta t)\end{cases} \tag{35}$$

where $v_x(t)$ and $R_p(u)$ denote the output malware attack behavior data feature frequency cross terms as well as the time scale impulse response of the malware attack behavior data transmission, v_0 and $Y_p(u)$ denote the frequency cross terms at the initial stage and the malware attack behavior The data identifies the output center distance, and βt denotes the inclusion of noise in the malware attack behavior data time series.

When the malware carries out the attack behavior the attack behavior data time series contains noise as Gaussian voice, the solution equation can be obtained as follows:

$$\hat{c}_{4s}(n,\tau)=\hat{c}_{4s}(n,\tau)+\hat{c}_{4s}(n,\tau) \tag{36}$$

If the malware carries out the attack behavior when the attack behavior data time series contains noise as non-Gaussian noise, it can be known that $\hat{c}_{4w}(n,\tau)$ is the data time series constrained directionality feature.

The malware attack behavior data contained in the data transmission is linearly correlated time series, and the data interference suppression function based on FIR filtering is:

$$x_n=c_0+\sum_{i=1}^{M_{AR}}c_i x_{n-i}+\sum_{j=0}^{M_{MA}}b_j\eta_{n-j} \tag{37}$$

where c_0 and c_i denote the initial sampling assignment of the data during the information transmission process and the sampling assignment of the data at a specified moment, η_{n-j} and M_{MA} denote the time-varying transient frequency of the data features of the malware attack and the short-time window function of the time series of the data in the subdomain of the time-frequency feature space, M_{AR} and X_{n-i} denote the multivariate quantity value function of the data time series and its associated mean-value function, respectively. window function, M_{AR} and x_{n-i} denote the multivariate quantity-valued function of the data time series and the scalar time series of the data with its associated mean-variance, respectively, and b_j denotes the oscillatory assignments of the data in the information transmission process.

III. C. Autonomous protection against malware attacks

Using $(v_1, v_2, \cdots, v_m)$ to represent the sequence of data feature vectors identified in the previous subsection as malware with attack behavior during the transmission of the information, which is protected by the Malicious Protection Center.

The first r sequences in the sequence of feature vectors are used as the principal components of the normal space of the data packet, and the columns of the matrix of the data packet P' that have the same size as the sequence of feature vectors $(v_1, v_2, \cdots, v_r)$ as the $m^* r$, which are obtained from the computed data $\hat{y}$ and $\bar{y}$ formulas, respectively:

$$\hat{y} = P'P'y' = By' \tag{38}$$

$$\bar{y} = (I - P'P')y' = \tilde{B}y' \tag{39}$$

Among them, $B = P'P'$ and $\tilde{B} = I - P'P'$ are packet matrices, and the packet subspace is represented by Q, and $\hat{y}$ and $\bar{y}$ are mapped to the packet normal subspace and the packet exception subspace respectively, and when the packet has an abnormal condition, $\bar{y}$ will change dramatically. The squared prediction error method is used to protect the malware attack behavior, and the self-protection function of the malware attack line is as follows:

$$SPE = (\| \bar{y} \|^2 + \| \tilde{B}y' \|^2)x_n \tag{40}$$

When the data does not exist malware attack behavior $SPE \le \zeta_a^2$, when the data exists malware attack behavior $SPE > \zeta_a^2$, where ζ_a^2 denotes the confidence level of $1-\alpha$ when the malware attack behavior of the squared prediction error threshold value, when $SPE > \zeta_a^2$ when utilizing Eq. for autonomous protection against malware attack behavior.

IV. Experimentation and analysis

IV. A. Simulation analysis

IV. A. 1) Algorithm Validation

Set the number of Markov chain iteration steps as 40000 steps, the first 20000 steps are burning period, used for sample convergence, and only the samples of the last 2000 steps are counted. After 20000 steps every 20 steps to take a sample for statistical malware parameters, the standard deviation of the total error of the posterior probability distribution function is 1. In order to visualize the convergence process of this paper's algorithm and draw the sample extraction process of the malware location parameters, the sample extraction process of the malware location parameters of this paper's method is shown in Figure 2. From the figure, it can be intuitively observed that the samples extracted by this paper's algorithm are gradually close to the real malware location parameters, and finally reach convergence near the real malware location parameters.

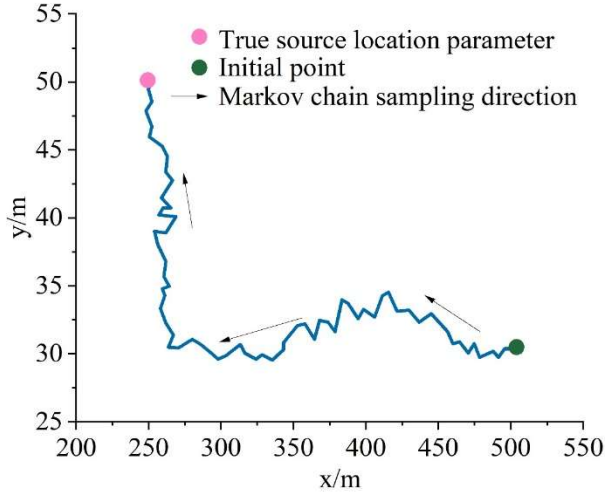

Figure 2: Location parameter sample extraction process

The statistical results of the malware sample parameters of this paper's algorithm are shown in Table 1.The results of the malware samples obtained using this paper's algorithm approximate the true values with small relative errors, which further validates the accuracy of this paper's algorithm in determining the malware. This paper's algorithm will eventually obtain the estimated interval of the malware parameters, and the range size of the interval is related to the initial variance selected by the proposed distribution of this paper's algorithm.

Table 1: Data of source sample parameters

Leakage source parameter	True value	Sample mean	Relative error(%)	Estimation interval
Location parameter x	220	250.094392	0.0366	(253.06,251.16)
Location parameter y	70	49.991205	0.01641	(49.22,50.39)
Strength parameter Q	1550	1641.445392	0.7125	(1310.11,2322.6)

IV. A. 2) Algorithm error adaptation analysis

In this section of experiments, the error adaptation of this paper's algorithm is analyzed to study the effect of the standard deviation of the total error σ on the performance of this paper's algorithm. Before carrying out the algorithm operation of this paper, data cleaning was carried out and the effect of data cleaning on the error adaptability of this paper's algorithm was analyzed at the same time. Using this paper's algorithm to run independently 100 times and take the average, the results of this paper's algorithm under different total error standard deviation when data cleaning is not performed on the data are shown in Table 2. The results of this paper's algorithm are shown in Table 3 under different total error standard deviations when data cleaning is performed on the data. Among them, the estimation interval of the parameters is the interval constituted by the extracted samples. It can be seen that with the increase of the total error standard deviation of this paper's algorithm, the width of the estimation interval of the parameters of the acquired samples gradually increases, which indicates that the increase of the total error will make part of the samples extracted by this paper's algorithm gradually away from the true value. At the same time, the results of this algorithm are more adaptive and the width of the parameter estimation interval is narrower under different total error standard deviation after data cleaning than without data cleaning. This is due to the data cleaning will be less than 0.000002g/m³ data taken as 0, reducing the impact of very small values on the sampling results of this algorithm. At the same time, in the data cleaning, for less than 0.1/m³ data to take the logarithmic processing, not only did not change the nature of the data and the relative relationship, but also make the data smoother, which is conducive to the algorithm of this paper to converge to the real malware parameters.

Table 2: The results of the different total error standard deviation of the data are analyzed

Total error standard deviation	The parameters (x, y, Q) mean	Estimation interval			The length of the interval is estimated at (x, y, Q)
		x	y	Q	
0.1	(253.32,47.86,1608.65)	(248.22,250.59)	(50.83,49.04)	(125.75,1686.5)	(3.21,4.86,157.34)
0.5	(246.2,48.82,1637.74)	(233.9,265.63)	(47.21,54.17)	(154.47,2032.82)	(28.74,9.96,775.53)
1	(256.04,51.49,1243.05)	(184.93,330.64)	(38.73,60.39)	(109.32,2384.12)	(146.13,15.97,2273.66)
1.5	(247.61,48.56,192.61)	(155.73,324.46)	(40.48,59.72)	(283.24,2742.13)	(171.55,22.32,2457.26)
2	(266.97,51.54,908.89)	(37.95,516.58)	(22.17,80)	(0.61,2904.62)	(469.1,53.11,2902.35)
2.5	(208.72,56.59,486.79)	(-120.33,584)	(-3.7,97.13)	(0.27,3612.3)	(701.09,105.13,3610.72)

Table 3: The logarithm of the different total error standard deviation is the result

Total error standard deviation	The parameters (x, y, Q) mean	Estimation interval			The length of the interval is estimated at (x, y, Q)
		x	y	Q	
0.1	(247.93,48.88,1626.53)	(246.21,254.84)	(48.1,46.89)	(1402.81,1853.3)	(3.96,0.85,453.69)
0.5	(251.7,45.79,1620.69)	(249.98,252.2)	(52.06,51.05)	(1212.34,2104.17)	(3.19,1.69,888.38)
1	(253.79,54.3,1664.98)	(242.64,257.5)	(48.89,52.33)	(875.54,2674.32)	(10.52,5.65,1800.23)
1.5	(249.06,48.45,111.93)	(243.38,261.7)	(47.39,50.35)	(707.64,3262.65)	(22.05,6.91,2558.66)
2	(251.01,50.73,160.39)	(239.96,263.94)	(47.94,51.69)	(563.79,331.71)	(28.68,9.68,3262.8)
2.5	(250.23,47.98,2070.17)	(233.97,264.38)	(43.83,58.82)	(451.33,4281.76)	(32.62,11.79,3835.15)

IV. A. 3) Effect of initial point parameters on the algorithm

In this paper's algorithm, because the posterior probability distribution function is complex and multidimensional, the selection of the initial point determines where the algorithm starts iterating, which has an important impact on the subsequent iterative process of this paper's algorithm. Using this paper's algorithm to run 100 and take the average value, the results of this paper's algorithm under different initial point parameters are shown in Table 4. Among them, the success rate represents the proportion of the average malware parameter calculation results of this operation with a straight-line distance of less than 2m from the real malware parameters, and the number of first successful iteration steps represents the number of iteration steps in this operation when the first sampling is obtained with a straight-line distance of less than 2m from the real malware parameter samples, which is used to characterize the time for the algorithm to reach convergence. It can be seen that the success rate of this paper's algorithm gradually decreases as the initial point is gradually far away from the real malware, which indicates that the initial point far away from the real malware parameters makes it difficult for this paper's algorithm to converge to the real value, and the calculation results have a larger error with the real malware parameters. At the same time, as the initial point is gradually away from the real malware, the first successful iteration step of this paper's algorithm gradually increases, indicating that the time for this paper's algorithm to successfully converge to the real malware parameters becomes longer.

Table 4: Results of this algorithm with different initial point parameters

Initial parameter(x, y, Q)	Success rate(%)	The first successful iteration of steps
(350,45,2000)	100	89
(500,40,3000)	97	223
(600,35,4000)	94	494
(700,30,5000)	87	903
(800,25,6000)	39	3006
(900,20,7000)	28	9889
(1000,15,8000)	16	10689

IV. B. Simulation Data Analysis and Research

A Pentium42.8GHz,4G RAM simulation platform is used in the simulation, and the operating system model is XP SP2.In order to make a fair performance comparison between method 1 (FFD-based local anomaly data mining algorithm in large-scale high-dimensional datasets), method 2 (quantitative data mining algorithm based on improved multilevel fuzzy association rules) and the method of this paper, a Workstation5.0 virtual machine,and the driver system was introduced into the actual system to install the malware attack analysis system. When the installation of all the software is completed, the virtual machine is photographed and the photos are stored to facilitate the system state rollback. In this paper, the experimental design of malware attack types and attack time as shown in Table 5.

Table 5: Malicious software attack type and attack time

The time of the attack/s	Attack type	Duration of the attack/s
0	Normal	100
135	Unauthorized access	15
150	Buffer overflow	15
165	Normal	100

The number of packet loss of method 1, method 2 and the method of this paper is shown in Fig. 3~Fig. 5. As can be seen from the figure, the model constructed in this paper can effectively analyze the time when the malware attack behavior occurs. It corresponds exactly to the time of launching attack designed in the table. When when the attack event occurs in 135s and 150s, the number of received data table of this paper's method decreases significantly, indicating that the attack behavior occurs in this time period.

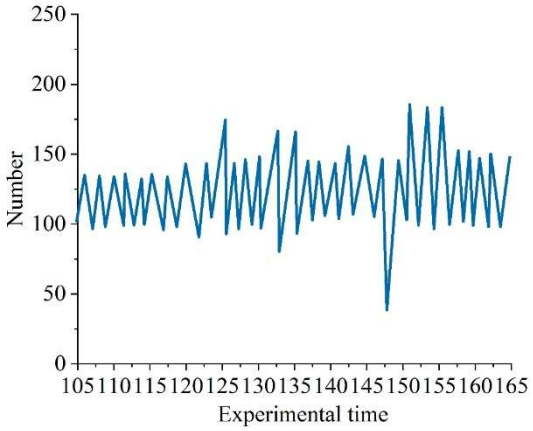

Figure 3: Method 1 number of lost bags

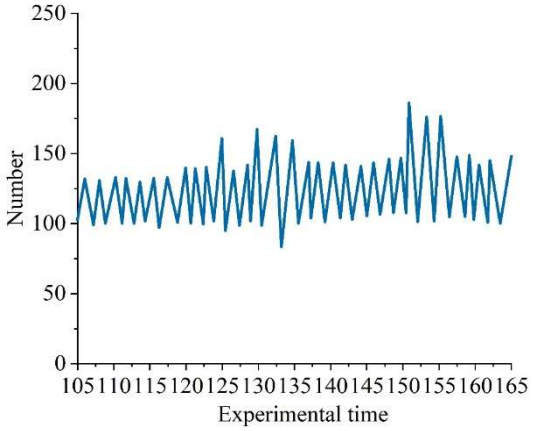

Figure 4: Method 2 number of lost bags

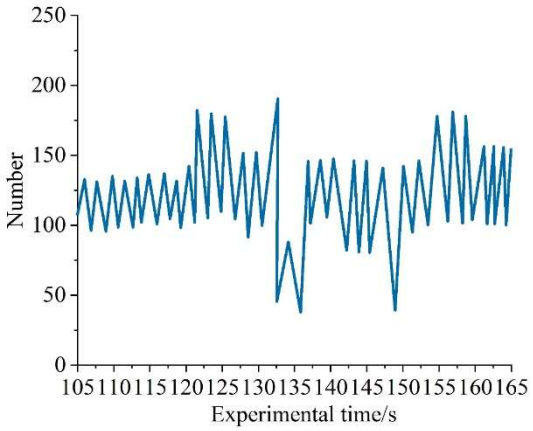

Figure 5: The number of bags lost in this article

In this simulation, the total simulation time is 165 seconds, and the malicious attack is launched against the software, which is calculated using three methods. The decision rate and detection rate test results of the three methods are shown in Fig. 6 and Fig. 7, respectively. As can be seen from the figures, the decision rate of method 1 decreases faster, which is due to the fact that with the increase in the number of concurrent inputs to the system, multiple active events occur at the same moment, resulting in the method not being able to make accurate decisions. In addition, the detection rates of all three methods show a decreasing and then stabilizing trend with the increase of the number of concurrent inputs. The decision rate and detection rate are higher than the two methods when using this method for software malicious attack detection. It shows that the method in this paper has fully grasped

the behavioral characteristics of malicious attacks after a period of detection, and improved the accuracy of attack recognition.

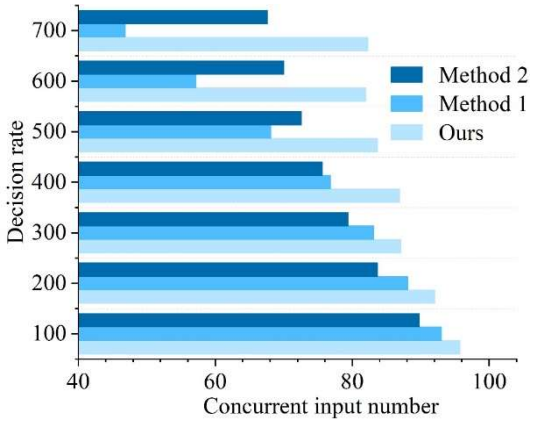

Figure 6: Comparison of decision rate of different methods

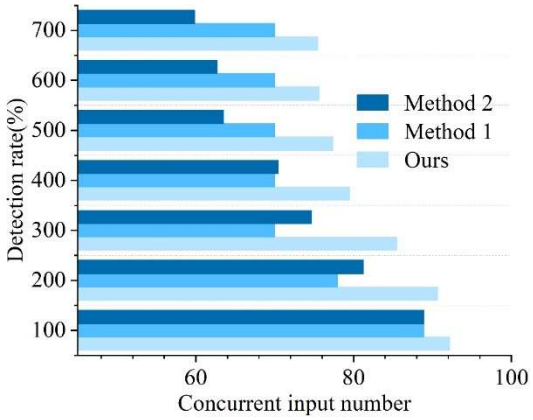

Figure 7: Comparison of different method detection rates

V. Conclusion

By comparing the experimental results, the malware behavior detection method based on Bayesian inference model proposed in this paper shows significant advantages. In the experiments, the relative error of malware location parameter is 0.0366%, which proves the high accuracy of the method in malware detection. Meanwhile, the strong adaptability of this paper's method to the error is proved by data cleaning and error adaptation analysis, and the parameter estimation interval gradually increases with the increase of the standard deviation of the total error, but the parameter estimation interval is significantly narrowed after the data cleaning process, which shows the robustness of the method in the complex data environment.

In addition, this paper's method outperforms traditional methods in terms of decision rate and detection rate under different attack scenarios. Especially in the concurrent scenario of malware attacks, the method in this paper successfully identifies the attack behavior and accurately locates the time when the attack occurs, thus effectively avoiding the risk of misjudgment and omission. This shows that the malware behavior detection and protection method based on Bayesian inference proposed in this paper has high practicality and adaptability, and can provide strong technical support for network security protection.

References

[1] Gan, C., Feng, Q., Zhu, Q., Zhang, Z., Zhang, Y., & Xiang, Y. (2020). Analysis of computer virus propagation behaviors over complex networks: a case study of Oregon routing network. Nonlinear Dynamics, 100, 1725-1740.

[2] Bhunia, S., & Tehranipoor, M. (2018). The hardware trojan war. Cham,, Switzerland: Springer.

[3] Saeed, M. A. H. (2020). Malware in computer systems: Problems and solutions. IJID (International Journal on Informatics for Development), 9(1), 1-8.

[4] Akinde, O. K., Ilori, A. O., Afolayan, A. O., & Adewuyi, O. B. (2021). Review of computer malware: detection and preventive strategies. Int. J. Comput. Sci. Inf. Secur.(IJCSIS), 19, 49.

[5] Or-Meir, O., Nissim, N., Elovici, Y., & Rokach, L. (2019). Dynamic malware analysis in the modern era—A state of the art survey. ACM Computing Surveys (CSUR), 52(5), 1-48.
[6] Tahir, R. (2018). A study on malware and malware detection techniques. International Journal of Education and Management Engineering, 8(2), 20.
[7] Alenezi, M. N., Alabdulrazzaq, H., Alshaher, A. A., & Alkharang, M. M. (2020). Evolution of malware threats and techniques: A review. International journal of communication networks and information security, 12(3), 326-337.
[8] Mat, S. R. T., Ab Razak, M. F., Kahar, M. N. M., Arif, J. M., & Firdaus, A. (2022). A Bayesian probability model for Android malware detection. ICT Express, 8(3), 424-431.
[9] Yerima, S. Y., Sezer, S., & McWilliams, G. (2014). Analysis of Bayesian classification-based approaches for Android malware detection. IET Information Security, 8(1), 25-36.
[10] Turner, B. M., & Van Zandt, T. (2018). Approximating Bayesian inference through model simulation. Trends in cognitive sciences, 22(9), 826-840.
[11] Doan, B. G., Nguyen, D. Q., Montague, P., Abraham, T., De Vel, O., Camtepe, S., ... & Ranasinghe, D. C. (2024, September). Bayesian learned models can detect adversarial malware for free. In European Symposium on Research in Computer Security (pp. 45-65). Cham: Springer Nature Switzerland.
[12] Jamadi, Z., & Aghdam, A. G. (2024). Enhanced Malware Prediction and Containment Using Bayesian Neural Networks. IEEE Journal of Radio Frequency Identification.
[13] Ashfaq, A. B., Abaid, Z., Ismail, M., Aslam, M. U., Syed, A. A., & Khayam, S. A. (2018). Diagnosing bot infections using Bayesian inference. Journal of Computer Virology and Hacking Techniques, 14, 21-38.
[14] Sharma, A., Gupta, B. B., Singh, A. K., & Saraswat, V. K. (2023). A novel approach for detection of APT malware using multi-dimensional hybrid Bayesian belief network. International Journal of Information Security, 22(1), 119-135.
[15] Zimba, A. (2022). A Bayesian attack-network modeling approach to mitigating malware-based banking cyberattacks. Int J Comput Netw Inf Secur, 14(1), 25-39.
[16] Meng, W., Li, W., Xiang, Y., & Choo, K. K. R. (2017). A bayesian inference-based detection mechanism to defend medical smartphone networks against insider attacks. Journal of Network and Computer Applications, 78, 162-169.
[17] Doan, B. G., Yang, S., Montague, P., De Vel, O., Abraham, T., Camtepe, S., ... & Ranashinghe, D. C. (2023, June). Feature-space bayesian adversarial learning improved malware detector robustness. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 37, No. 12, pp. 14783-14791).
[18] Hou, Y., He, R., Dong, J., Yang, Y., & Ma, W. (2022). Iot anomaly detection based on autoencoder and bayesian gaussian mixture model. Electronics, 11(20), 3287.
[19] Perusquía, J. A., Griffin, J. E., & Villa, C. (2022). Bayesian models applied to cyber security anomaly detection problems. International Statistical Review, 90(1), 78-99.
[20] Hu Xiao,Zheng Jun,Su Ningxin,Fan Tian,Yang Chunliang,Yin Yue... & Luo Liang. (2021) .A Bayesian inference model for metamemory.. Psychological review,128(5),824-855.
[21] Ao LIU,Ben Qi,Tao Liu & Liguo Zhang. (2023). AN ACCIDENT DIAGNOSIS METHOD OF HTR-10 BASED ON BAYESIAN INFERENCE MODEL. Proceedings of the ... International Conference on Nuclear Engineering. Book of abstracts : ICONE,30,1235-1235.
[22] Loevenich Johannes F.,Rettore Paulo H.L.,Lopes Roberto Rigolin F. & Sergeev Aleksandr. (2022). A Bayesian Inference Model for Dynamic Neighbor Discovery in Tactical Networks. Procedia Computer Science,205,28-38.
[23] Alan Benson & Nial Friel. (2020). Bayesian Inference, Model Selection and Likelihood Estimation using Fast Rejection Sampling: The Conway-Maxwell-Poisson Distribution. Bayesian Analysis,16(3),905-931.