Computers, Materials & Continua
DOI:10.32604/cmc.2022.022540
Article

Tech Science Press

ICMPTend: Internet Control Message Protocol Covert Tunnel Attack Intent Detector

Tengfei Tu[1,2], Wei Yin[3], Hua Zhang[1,2,*], Xingyu Zeng[1], Xiaoxiang Deng[1], Yuchen Zhou[1] and Xu Liu[4]

[1]State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, 100876, China
[2]State Key Laboratory of Cryptology, Beijing, 100878, China
[3]National Computer Network Emergency Response Technical Team/Coordination Center of China, 100029, China
[4]Pennsylvania State University, State College, 16801, USA
[*]Corresponding Author: Hua Zhang. Email: zhanghua_288@bupt.edu.cn
Received: 10 August 2021; Accepted: 16 September 2021

Abstract: The Internet Control Message Protocol (ICMP) covert tunnel refers to a network attack that encapsulates malicious data in the data part of the ICMP protocol for transmission. Its concealment is stronger and it is not easy to be discovered. Most detection methods are detecting the existence of channels instead of clarifying specific attack intentions. In this paper, we propose an ICMP covert tunnel attack intent detection framework ICMPTend, which includes five steps: data collection, feature dictionary construction, data preprocessing, model construction, and attack intent prediction. ICMPTend can detect a variety of attack intentions, such as shell attacks, sensitive directory access, communication protocol traffic theft, filling tunnel reserved words, and other common network attacks. We extract features from five types of attack intent found in ICMP channels. We build a multi-dimensional dictionary of malicious features, including shell attacks, sensitive directory access, communication protocol traffic theft, filling tunnel reserved words, and other common network attack keywords. For the high-dimensional and independent characteristics of ICMP traffic, we use a support vector machine (SVM) as a multi-class classifier. The experimental results show that the average accuracy of ICMPTend is 92%, training ICMPTend only takes 55 s, and the prediction time is only 2 s, which can effectively identify the attack intention of ICMP.

Keywords: Internet control message protocol; support vector machine; covert tunnel; network analysis

1 Introduction

Internet Control Message Protocol (ICMP) covert tunnel is used to transmitting special information to processes or users prevented from accessing the information. It is more hidden and more difficult to detect than malware traffic. The purpose of using covert channels is to send data in the network while ensuring that the sending is unnoticed by a third party and without alerting any firewalls or Intrusion Detection Systems (IDS) on the network. Studies have shown that a large website may have 26 gigabyte (GB) of information illegally stolen by covert tunnels in a year, assuming that an ICMP packet only carries 1 bit of data [1–3].

Several researchers have oriented their research axes to detect covert channel attacks using multiple methods and techniques. Currently, covert tunnel detection is mainly studied in terms of both traffic behavior and signature.

The detection method based on traffic behavior uses behavior characteristics such as the maximum, minimum, average time interval, message size, and the ratio of the number of request and response messages within a specified time window as the detection basis. This method takes all traffic within a specified time window as a detection object, and can only determine whether a covert tunnel has been established at both ends of the communication within a certain time window, and cannot locate specific malicious traffic [4–8]. On the other hand, the data features containing specific attack commands are not extracted to clarify the specific attack intent of the covert tunnel. All these have brought inconvenience to security personnel in taking targeted defensive measures [9–12]. In order to determine the attacking intent of the covert tunnel more accurately, it is necessary to analyze the detection of the ICMP covert channel from the perspective of data.

Signature-based detection [12,13] detects attacks by detecting signatures in the data part of the ICMP traffic. It does not detect unknown attacks, and its generalization ability is weak. For example, the ICMP covert tunnel tool icmptunnel [14] generates covert tunnel traffic containing the keyword "TUNL" by default, so the way to detect such covert tunnels is to identify the signature of "TUNL", but when the attacker deliberately modifies the keyword or does not use the keyword, the detection is invalid. Because signature-based detection relies on expert knowledge to extract keywords and perform strict matching.

Through the analysis of a large number of ICMP covert tunnel traffic, we found that ICMP covert tunnel traffic has obvious and specific attack intentions in the data part, such as shell attacks, access to sensitive directories and other illegal behaviors. Corresponding shell commands, sensitive directories, communication protocol keywords, tunnel reserved words, and common network attack keywords often appear in the data part of the malicious traffic of ICMP covert tunnel. For example, the Hypertext Transfer Protocol (HTTP) keyword "www", the sensitive directory "User" in the Windows operating system, the reserved word "TUNL" in the tunnel tool ptunnel, the shell command "docker pull nginx; /bin/sh shell.sh". With these types of keywords as features, the attack intention of the covert tunnel can be effectively detected, and targeted defensive measures can be taken.

A large number of studies have proved that machine learning methods have good generalization in traffic detection. Among them, SVM [15] is a classification model that shows many unique features in solving small and medium-sized data samples, non-linear and high-dimensional pattern recognition. It solves the problem of linearly indistinguishable data sets by mapping linearly indistinguishable data to a high-dimensional feature space through a kernel function. It divides the data set by a hyperplane related to only a small number of support vectors, so it requires only a small amount of data to build a model and is insensitive to noisy data. The flow of ICMP covert tunnels happens to be high-dimensional, linearly indistinguishable, and there are a lot of noise data, so we believe that the SVM model is an effective solution for ICMP covert tunnel detection.

In this paper, we propose ICMPTend, an ICMP covert tunnel attack detector, by extracting the corresponding keyword features for common ICMP covert tunnel attack intent and using SVM as a classifier algorithm.

In summary, we make the following contributions in this paper:

- We propose a systematic ICMP covert tunnel attack intent detection framework ICMPTend, which consists of five steps: data collection, feature lexicon construction, data preprocessing, model construction, and attack intent prediction. It can detect a variety of attack intentions, such as shell attacks, sensitive directory access, communication protocol traffic stealing, filling tunnel reserved words, and other common network attacks.
- We build a multi-dimensional malicious feature lexicon containing keywords for shell attacks, sensitive directory access, communication protocol traffic theft, filling tunnel reserved words, and other common network attacks.
- The experimental results show that the average accuracy of ICMPTend reaches 92%, the training time is only 55 s, and the prediction time is only 2 s, which can effectively identify the attacking intention of ICMP.

2 Preliminaries

With the rapid development and progress of network technology, our daily work is increasingly dependent on the network. While network technology brings us convenience, it also brings hidden security threats. Many researchers have begun to study the application of artificial intelligence technology in network attack detection [16] and intrusion detection system construction [17]. As a typical network attack method, the ICMP covert channel has attracted the attention of network attackers and security researchers. At present, common network attack detection methods are based on behavior statistics and signature-based methods.

In the detection method based on statistical behavior, [18] counted 12 behavioral characteristics of covert tunnels by analyzing data characteristic information such as packet size, tunnel traffic type, and fixed format of data, and established an SVM machine learning model to detect covert tunnels. In [19], authors established an information entropy-based detection model by calculating the confusion level of the data portion of ICMP. Reference [20] synthesized the criteria and behavior of ICMP to build an efficient tunnel detection system for ICMP. However, there are two problems with the above studies: first, hackers can bypass this detection method by imitating the communication behavior of normal ICMP traffic; second, only the signature left by the tunnel tool is used for the data part of ICMP containing malicious data without extracting the data features containing specific attack commands, which cannot clearly conceal the specific attack intent of the tunnel. All these bring inconvenience to security personnel to take targeted defense measures. In order to determine the attacking intent of the covert tunnel more accurately, the detection of ICMP covert tunnels needs to be analyzed from the perspective of data.

In signature-based detection methods, the main focus is to match the data part with a specific signature. Some covert tunnel tools generate traffic with distinct signatures, e.g., icmptunnel generates tunnel traffic with the signature string "TUNL". Some ICMP covert channels will also be used to transmit the content of other protocols, such as HTTP and Domain Network System (DNS). Keywords "TUNL", "http://" and "DNS" can be used as typical signature features. There are two problems with detection based on data signatures: first, it needs to accumulate signatures continuously, unable detect unknown attacks, and its generalization ability is weak; second, the detection unit of this method is a single traffic flow, and it cannot detect context-sensitive covert tunnel which splits payload into multiple traffic for delivery.

Symbols used in this paper and their meanings are shown in the following Tab. 1.

Table 1: Symbols table

Symbols	Meaning
FD_i	The feature word numbered i in the feature word bank
$\|*\|$	Number of $*$ elements
v	Feature vectors v constructed from ICMP traffic
T_j	The jth ICMP traffic
$f_{j,i}$	Term Frequency-Inverse Document Frequency (TF-IDF) value of the corresponding feature v_i of the jth ICMP traffic
m	Dimension of the feature vector
d_j	The data portion of the jth ICMP traffic
y_i	ICMP traffic labels, i from 0 to 5 indicate NORMAL labels, SHELL_ATTACKS labels, ACCESS_SENSITIVE_DIRS labels, STEAL_PROTOCOLS labels, FILL_RESERVED_WORDS labels, and COMMON_CYBER_ATTACKS labels, respectively

3 Detection Framework

The detection framework of ICMPTend is shown in Fig. 1, which is divided into two phases: training and prediction. The training phase includes four steps: data acquisition, feature database construction, data preprocessing and model construction. The data in the prediction phase is predicted using the trained model after data preprocessing.

3.1 Training Phase

Step I Data Acquisition: An extensive collection of five types of malicious samples including shell attacks, accessing sensitive directories, stealing communication protocol traffic, filling tunnel reserved words, and common network attacks, and using 0–5 tags to indicate classification methods, such as obtaining them from websites such as GitHub to build concealment tunnel sandbox, etc.

Step II Feature Database Construction: Collect feature words from the perspectives of shell commands, sensitive directories, communication protocol keywords, tunnel reserved words, and common network attack keywords, and build a characteristic database.

Step III Data Preprocessing: After three steps of hexadecimal decoding, common encryption method decoding, and text feature representation, the original ICMP hexadecimal is converted into a tensor that the model can learn.

Step IV Model Construction: Construction of ICMPTend covert tunnel detection model base on SVM classifier.

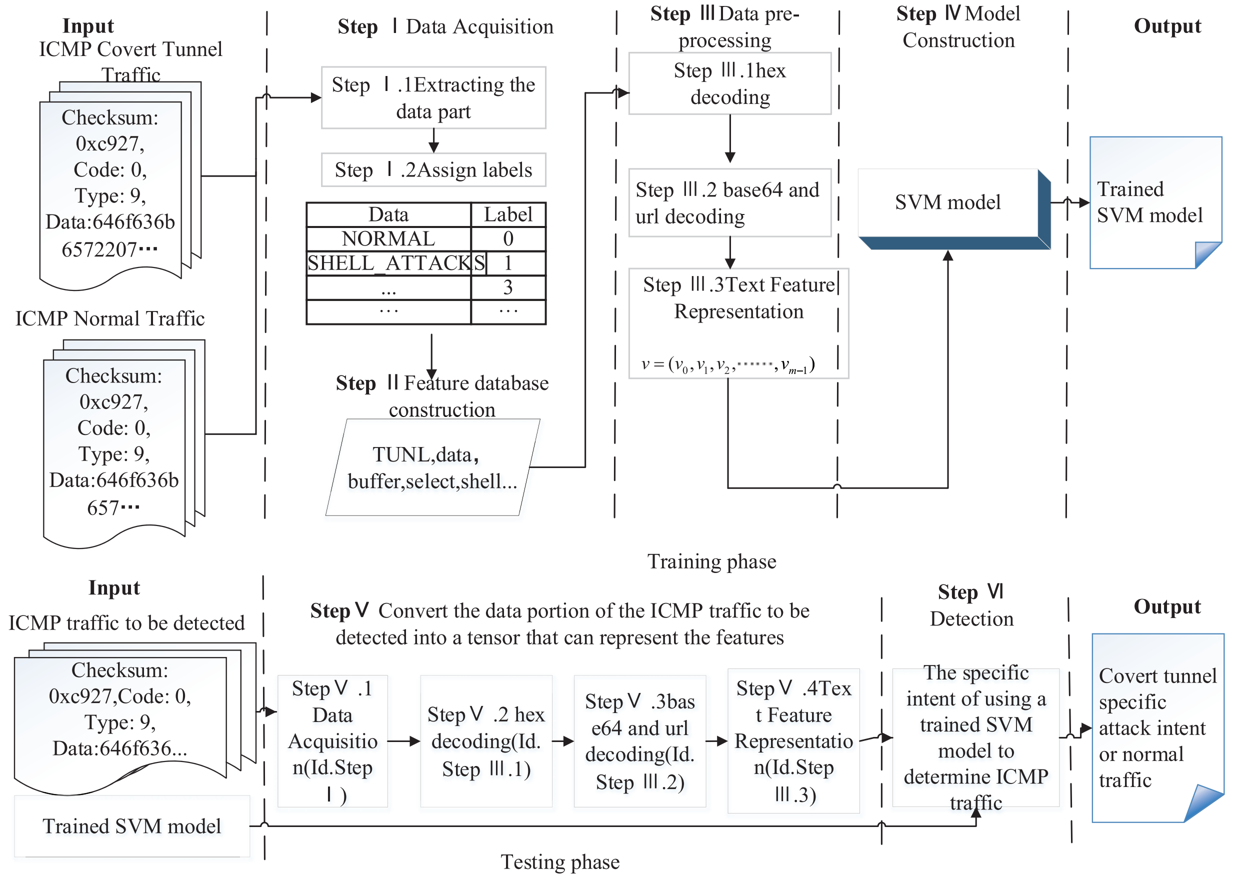
Input
ICMP Covert Tunnel Traffic
Checksum: 0xc927, Code: 0, Type: 9, Data:646f636b6572207…
ICMP Normal Traffic
Checksum: 0xc927, Code: 0, Type: 9, Data:646f636b657…
Step Ⅰ Data Acquisition
Step Ⅰ.1Extracting the data part
Step Ⅰ.2Assign labels
Data | Label
NORMAL | 0
SHELL_ATTACKS | 1
... | 3
Step Ⅱ Feature database construction
TUNL,data, buffer,select,shell...
Step Ⅲ Data pre-processing
Step Ⅲ.1hex decoding
Step Ⅲ.2 base64 and url decoding
Step Ⅲ.3Text Feature Representation
$v=(v_0,v_1,v_2,\cdots\cdots,v_{m-1})$
Step Ⅳ Model Construction
SVM model
Output
Trained SVM model
Training phase
Input
ICMP traffic to be detected
Checksum: 0xc927,Code: 0, Type: 9, Data:646f636...
Trained SVM model
Step Ⅴ Convert the data portion of the ICMP traffic to be detected into a tensor that can represent the features
Step Ⅴ.1 Data Acquisition(Id.Step Ⅰ)
Step Ⅴ.2 hex decoding(Id. Step Ⅲ.1)
Step Ⅴ.3base64 and url decoding(Id. Step Ⅲ.2)
Step Ⅴ.4Text Feature Representation(Id.Step Ⅲ.3)
Step Ⅵ Detection
The specific intent of using a trained SVM model to determine ICMP traffic
Output
Covert tunnel specific attack intent or normal traffic
Testing phase

Figure 1: Framework of ICMPTend

3.2 *Prediction Phase*

Step V Convert the data part of the ICMP traffic to be detected into a tensor that can represent features as model inputs. After the **Step VI** detection phase, the output of the model is a label, which can indicate the specific attack intention of the hidden tunnel or confirm that there is no covert tunnel.

4 Implementation

4.1 *Data Acquisition*

Extracting features from the perspective of specific attack intentions of ICMP covert tunnels to identify hidden tunnels is essentially a multi-classification task of ICMP hidden tunnels based on attack intentions. In this paper, we mainly consider a large number of attack intents in covert tunnels, such as shell attacks, access to sensitive directories, stealing communication protocol traffic, filling tunnel reserved words, and common network attacks. There are five types of specific attack intentions. The benign samples come from normal ICMP traffic in the campus network of Beijing University of Posts and Telecommunications (BUPT), with a total of 1,000; the malicious samples come from the following sources:

(1) Sample ICMP tunnel traffic collected from sites such as GitHub, counting 442 entries.
(2) Rules and other ICMP covert tunnel detection models judged as malicious, and manually sampled and labeled malicious traffic in the campus network, totaling 659 items.
(3) The malicious traffic was constructed and communicated using ICMP covert tunnel tools such as icmptunnel, ptunnel, and icmpsh, and then crawled using Wireshark, counting 3,361 entries.

A total of 4,462 malicious samples with malicious attack intent were obtained, and the number of samples with specific attack intent of 5 types is shown in Tab. 2. The samples are divided into the training set and testing set in the ratio of 7:3 for experiment.

Table 2: Five types of specific attack intent and sample data distribution

Label (y_i)	Training set	Testing set	Total
y_0: NORMAL	700	300	1000
y_1: SHELL_ATTACKS	660	282	942
y_2: ACCESS_SENSITIVE_DIRS	520	222	742
y_3: STEAL_PROTOCOLS	686	294	980
y_4: FILL_RESERVED_WORDS	474	203	667
y_5: COMMON_CYBER_ATTACKS	784	336	1120
Total	3824	1638	5462

4.2 Feature Database Construction

Feature words are mainly composed of letters, numbers and special symbols, and different feature databases have different construction methods. Some commercial software constructs feature databases by directly querying the feature signature of malware [6]. In [18], the feature databases were constructed by directly cutting words. In our paper, we found that ICMP covert tunnel traffic has obvious and specific attack intentions in the data part, such as SHELL_ATTACKS, ACCESS_SENSITIVE_DIRS, etc. In order to extract keywords that can represent these attack intents, corresponding to the above attack intents, feature keywords from the perspective of shell commands, sensitive directories and their operations, communication protocols and related information, tunnel reserved words and common network attacks. The database is constructed as follows:

(1) SHELL_ATTACKS keywords: Shell attacks are essentially composed of various shell commands, and shell commands are divided into built-in commands and external commands. Therefore, this paper combines the malicious samples in the training set to collects 78 keywords of built-in shell commands and 33 common keywords. There are 111 external commands, such as the built-in command keyword “kill” for forcibly terminating the startup process and the external command keyword “sh” for starting a shell script, which constitute the keyword set for shell attacks.
(2) ACCESS_SENSITIVE_DIRS keywords: When hackers enter sensitive directories, they may add, delete, change, check, copy, upload and download files in sensitive directories. Therefore, this article combines the malicious samples in the training set to collect 241 common sensitive directories, sensitive file names, and keywords for sensitive file operations in Linux

and Windows operating systems, such as sensitive directories "etc" and "bin" in Linux. And the keyword "read()" for Python functions used to read and write the contents of a file. For example, the sensitive directories "etc" and "bin" in Linux, and the keywords "read()" and "write" of the Python function used to read and write the contents of files are used to construct the keyword set for sensitive directory access.

(3) STEAL_PROTOCOLS keywords: After some ICMP covert tunnels are established, traffic from the controlled side using any communication protocol will be sent to the control side through the covert tunnel. In this paper, we combine the malicious samples in the training set to collect the names of common communication protocols and a total of 86 keywords related to each communication protocol, such as "http://" involving HTTP protocol, "www.", ".com" and ".cn" etc. are used to construct a keyword set for the theft of communication protocol traffic.

(4) FILL_RESERVED_WORDS keywords: Some ICMP covert tunnel tools [19,20] and some hackers deliberately fill in some reserved words in ICMP covert tunnel traffic as their identities. In this paper, we collect 76 reserved words, such as "TUNL", "tun0" and "signature" from the malicious samples in the training set, and build the tunnel reserved word keyword word set.

(5) COMMON_CYBER_ATTACKS keywords: After the ICMP covert tunnels are established, some attackers send common network scripts such as SQL injection, command execution, cross-site scripting attacks to the controlled end through the covert tunnel, and the controlled end launches corresponding attacks on the target server, thus evading the security personnel's tracking by means of this intermediate bridge. In this paper, we collect a total of 150 common network attack keywords with malicious samples in the training set, such as "select", "union" and "from" frequently used in SQL injection, and "<script>", "alert" and "<img>", frequently used in cross-site scripting attacks.

The final set of these five types of keywords are combined into a feature database (FD) containing 637 unique feature words. The composition and description of the feature database are shown in Tab. 3.

Table 3: Composition and instances of feature database

Keyword	FDi	Feature word instances
SHELL_ATTACKS	111	rm, cd, mkdir, wget, cat, echo, kill, sh
ACCESS_SENSITIVE_DIRS	241	etc., admin, bin, read, write, db, C:\Program Files
STEAL_PROTOCOLS	86	http, www, .com, .cn, .gov, https, ssh, scp, irp, if
FILL_RESERVED_WORDS	76	TUNL, tun0, signature, tunnel, DataBuffer
COMMON_CYBER_ATTACKS	150	select, from, union, insert, alert, <script>, <img>,
TOTAL (unique words)	637	

4.3 Data Preprocessing

Data preprocessing is the process of converting the hexadecimal data of ICMP data part into tensors that can be recognized by the machine learning model after hexadecimal decoding, string decoding, and text feature representation. The specific process is as follows:

Step 1: Hexadecimal Decode

The data field of the original ICMP traffic stores data in the form of a hexadecimal stream. In order to extract the text features of the transmitted content, the hexadecimal data needs to be decoded. The decoding function is shown in Eq. (1).

$$\mathrm{decode}(\mathrm{HEX}) = \mathrm{encoded\,String} \tag{1}$$

As shown in Fig. 2, the ICMP covert tunnel is to transmit a shell attack statement-"docker pull nginx; L2Jpbi9zaCBzaGVsbC5zaA==", but the actual after hexadecimal encoding, what is passed is "646f636b6572 ......", where "docker" is encoded as "646f636b6572", "pull" is encoded as "707566c6c", and "nginx" is encoded as "6e67696e78". Both the training model and the prediction stage need to encode the ICMP data.

Step 2: String Decode

With the continuous development of various encryption technologies, attackers use Uniform Resource Locator (URL) encoding [21], BASE64 encoding [22] and other encoding methods [23] to encode attack traffic to evade the detection of security detection system and thus hide their information. Attack intent, normal URL decoding and BASE64 decoding of text can effectively restore the original traffic and improve the detection efficiency. The decoding function is shown in Eq. (2).

$$\mathrm{decod}e_{\mathrm{base64,URL}}(\mathrm{encoded\,String}) = \mathrm{String} \tag{2}$$

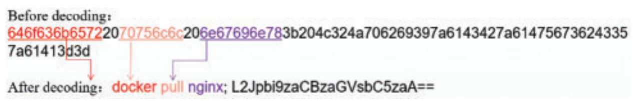
Before decoding:
646f636b657220707566c6c206e67696e783b204c324a706269397a6143427a61475673624335
7a61413d3d
After decoding: docker pull nginx; L2Jpbi9zaCBzaGVsbC5zaA==

Figure 2: The instance of ICMP covert tunnel traffic before and after decoding

As shown in Fig. 3, the hexadecimal decoded text "docker pull nginx; L2Jpbi9zaCBzaGVsbC5 zaA ==" is decoded by BASE64 and becomes "docker pull nginx; /bin/sh shell.sh", the key part of the shell attack "/bin/sh shell.sh" is restored.

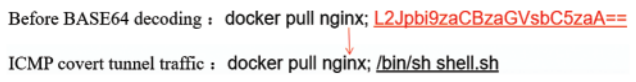
Before BASE64 decoding: docker pull nginx; L2Jpbi9zaCBzaGVsbC5zaA==
ICMP covert tunnel traffic: docker pull nginx; /bin/sh shell.sh

Figure 3: The instances of the traffic before and after BASE64 decoding

Step 3: Text Feature Representation

In this paper, we use word frequency-inverse document frequency (TF-IDF) [24] for text feature representation, through which the text content can be converted into a feature-representing tensor, which can be input into the model for learning. TF-IDF is a statistical method used to evaluate the importance of a word to a sample in the training set. The core idea is that the importance of a word increases in proportion to the number of times it appears in the sample, but it is not in the sample. The number of occurrences is inversely proportional to the frequency in the training set. The algorithm flow of text feature representation is as follows. The text feature representation of the example ICMP covert tunnel traffic is shown in Fig. 4.

1) Each word in the FD is numbered, which corresponds to the index in the feature vector with a latitude size of, m = |FD|$\vec{v} = (\overrightarrow{v_0}, \overrightarrow{v_1}, \cdots, \overrightarrow{v_{m-1}})\ \overrightarrow{v_i} = 0$ for $i = 0, 1, \ldots, m$ (637 in this paper).
2) Initialize a vector for each ICMP traffic T_j

$$\vec{v} = (\overrightarrow{v_0}, \overrightarrow{v_1}, \cdots, \overrightarrow{v_{m-1}})\ \overrightarrow{v_i} = 0\ for\ i = 0, 1 \ldots m. \tag{3}$$

vi in Eq. (3) represents the mapping of the corresponding numbered words in the feature database. The corresponding TF-IDF value is then calculated for this flow. $\vec{v} = (\overrightarrow{v_0}, \overrightarrow{v_1}, \cdots, f_{j,i}, \cdots, \overrightarrow{v_{m-1}})$

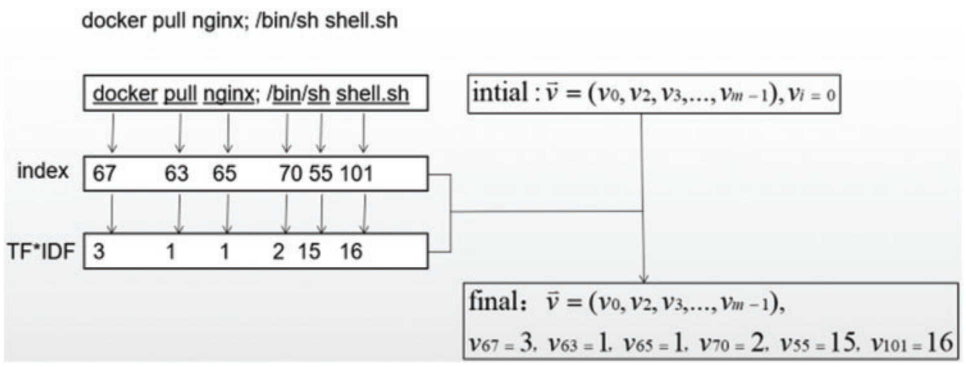
docker pull nginx; /bin/sh shell.sh
docker pull nginx; /bin/sh shell.sh
intial : $\vec{v} = (v_0, v_2, v_3, \ldots, v_{m-1}), v_i = 0$
index
67 63 65 70 55 101
TF*IDF
3 1 1 2 15 16
final: $\vec{v} = (v_0, v_2, v_3, \ldots, v_{m-1})$,
$v_{67} = 3,\ v_{63} = 1,\ v_{65} = 1,\ v_{70} = 2,\ v_{55} = 15,\ v_{101} = 16$

Figure 4: The instances of textual feature representation of ICMP covert tunnel traffic

4.4 Model Construction

In order to clarify the specific attack intentions of ICMP covert tunnel, ICMP traffic flow needs to be classified into multiple categories. There are six categories of multi-classification, namely, normal traffic, shell attack, sensitive directory access, communication protocol traffic stealing, filling tunnel reserved words, and common network attacks as shown in the aforementioned Tab. 1. ICMPTend model construction process is shown in Fig. 5, which is divided into two phases: training and testing. In the training phase, the input is pre-labeled benign and malicious sample data in the training set, and the output is a multi-class model with attack intention prediction capabilities. In the prediction phase, the input is the actual data part of the unlabeled ICMP traffic, and the output is the possible attack intent of the traffic. The purpose is to obtain a multi-classification model ICMPTend, which can predict the attack intention of ICMP traffic in real situations.

ICMPTend uses SVM as the classification algorithm. SVM has maintained its unique advantage in solving classification problems for small and medium samples, high-dimensional, and linearly indistinguishable datasets. The ICMP covert tunnel dataset constructed in this paper happens to be small-sample, high-dimensional, and linearly indistinguishable, so theoretically SVM is suitable for the situation in this paper.

The ICMPTend receives the data part of ICMP traffic as input, and outputs the label of the category to which the traffic belongs. The label corresponds to the specific attack intent. Suppose the training set contains the data part of ICMP traffic.

$\{(d_1, y_1), (d_2, y_2), \ldots, (d_T, y_T)\}, d_t(t = 1, 2, \ldots, T)$ denotes the data portion of a flow. $y_t \in \{0, 1, \ldots, 5\}$ indicates the label of ICMP traffic, 0 to 5 indicate normal traffic class, shell attack class, sensitive directory access class, communication protocol traffic stealing class, filling tunnel reserved word class, and common network attack class, respectively. The model first needs to obtain the feature representation of the data part, i.e., $d_t \rightarrow v_t, v_t \in R^n$. Next, a classification function needs to be fitted, assuming that the predicted label is $\hat{y}_t$. Then $\hat{y}_t = f(v_t)$ gives the result that the data portion of an ICMP traffic is predicted to be a certain class. $\mathrm{P}(\hat{y}_t = f(v_t)), t = 0, 1, \ldots, 5$ indicates the probability that the data part is predicted to be a certain category, Finally, the prediction label corresponding to $max(\mathrm{P}(\hat{y}_t = f(v_t))), t = 0, 1, \ldots, 5$ is selected as the final prediction output, which corresponds to the specific covert tunnel attack intent. During training, the model needs to minimize the loss function $\sum_{1}^{T} L_{\hat{y}_t \neq y_t}$ over the entire training set, and L denotes the function that calculates the loss in case of classification errors.

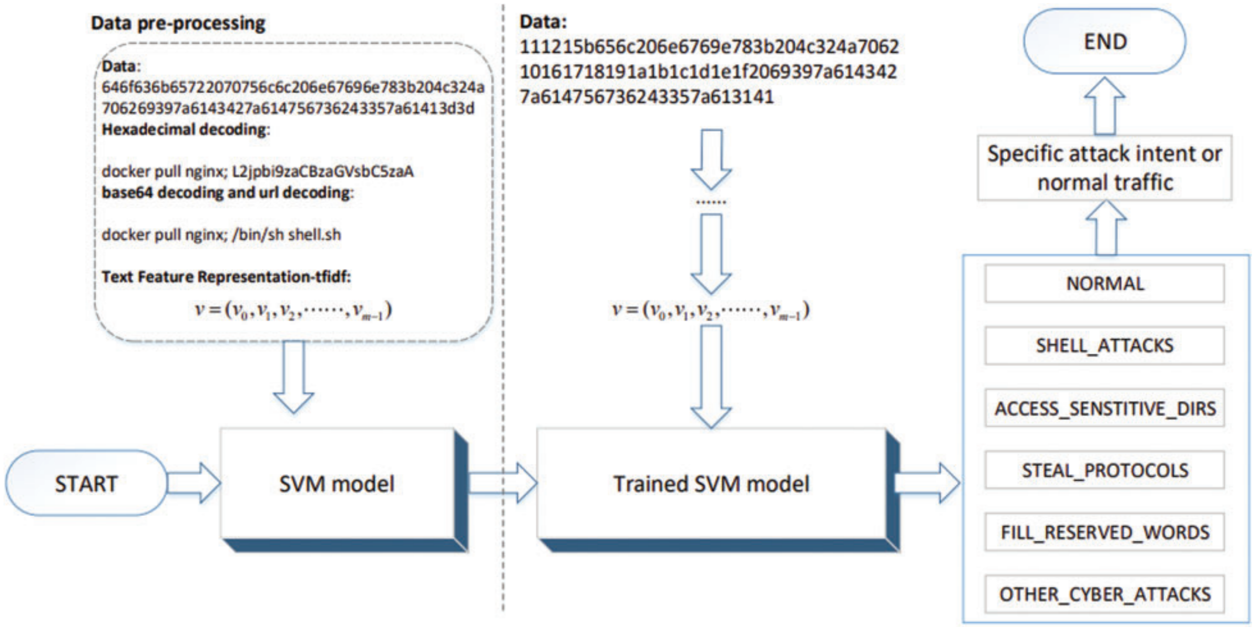
Data pre-processing
Data:
646f636b65722070756c6c206e67696e783b204c324a
706269397a6143427a614756736243357a61413d3d
Hexadecimal decoding:
docker pull nginx; L2jpbi9zaCBzaGVsbC5zaaA
base64 decoding and url decoding:
docker pull nginx; /bin/sh shell.sh
Text Feature Representation-tfidf:
$v = (v_0, v_1, v_2, \cdots\cdots, v_{m-1})$
Data:
111215b656c206e6769e783b204c324a7062
10161718191a1b1c1d1e1f2069397a614342
7a614756736243357a613141
......
$v = (v_0, v_1, v_2, \cdots\cdots, v_{m-1})$
START
SVM model
Trained SVM model
END
Specific attack intent or normal traffic
NORMAL
SHELL_ATTACKS
ACCESS_SENSTITIVE_DIRS
STEAL_PROTOCOLS
FILL_RESERVED_WORDS
OTHER_CYBER_ATTACKS

Figure 5: ICMPTend model architecture diagram

5 Evaluation

In order to verify the effectiveness and practicability of the database-based SVM covert tunnel attack intent detection model proposed in this article, this section answers the following four questions through related experiments:

Question 1 (Q1): Are the features constructed in this article effective?

Question 2 (Q2): Is it appropriate to choose SVM as a classifier?

Question 3 (Q3): What are the advantages of building a feature lexicon based on specific attack intent?

Question 4 (Q4): Compared with the anomaly detection capabilities of two categories, is the attack intent detection of multiple categories acceptable?

The software environment used in this paper is Python 3.7, Scikit-Learn 0.21.3, Wireshark 3.2.7.0, the operating system is Ubuntu 16.04, and the hardware environment is Intel(R) Core(TM) i7-8550U @ 1.80 GHz central processing unit (CPU), 8 GB random-access memory (RAM). The goal of this article is to measure the effectiveness of the model, which is essentially a standard multi-class model. Therefore, precision, recall, F1 score, accuracy and macro average are used as evaluation indicators to evaluate the experimental results of the multi-class model. This is shown in the following Tab. 4.

Table 4: Evaluation indicators and their meaning

Evaluation indicators and formulas	Meanings in this paper
$\text{Precision} = \frac{TP}{TP + FP}$ [24]	The percentage of samples correctly identified by the model as ICMP covert tunnel intentions out of the total number of samples identified by the model itself.
$\text{Recall} = \frac{TP}{TP + FN}$ [24]	The percentage of attacks that are correctly classified as ICMP covert tunnels among all samples in this category of data set.
$\text{F1} - \text{score} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$ [24]	When determining the accuracy rate, the larger the F1 score, the larger the proportion of the ICMP covert tunnel traffic correctly classified by the model to the total number of malicious traffic samples in the data set. When determining the recall rate, the larger the F1 score, the larger the proportion of the ICMP hidden tunnel traffic correctly classified by the model to the total number of malicious traffic samples identified by the model.
$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$ [25]	The percentage of traffic that the model correctly judges to the total traffic. The higher the accuracy of the model, the more effective the model.
$\text{MacroP} = \frac{1}{n}\sum_{i=1}^{n} \text{Precision}_i$ [25] $\text{MacroR} = \frac{1}{n}\sum_{i=1}^{n} \text{Recall}_i$ [25]	In the multi-class evaluation index, the higher the macro mean, the better the model.

5.1 Answer to Q1 and Q2

In order to answer Q1 and Q2, we build a feature dictionary based on the attack intent of shell attacks, access to sensitive directories, stealing communication protocol traffic, filling tunnel reserved words, and common network attacks, and construct feature vectors based on the feature dictionary as input to the model. In order to verify the effectiveness of the feature construction method in this paper, the feature vector is input into a separate model for training and prediction, and the effectiveness of the detection is evaluated.

The results of the comparison experiments using SVM, logistic regression (LR), and Naive Bayesian (NB) models are shown in Tab. 5. It can be found that inputting the feature vectors constructed by the feature construction method in this paper into multiple models for testing has achieved good results in terms of accuracy, recall, F1 score, and accuracy. Even in the NB model, which has the worst combined effect, the precision, recall, F1 score and accuracy reach at least 0.86, 0.80, 0.83, and 0.89, respectively. This indicates that the method of constructing features in this paper is effective.

Table 5: Evaluation result metrics

Model	Class	Precision	Recall	F1	Accuracy
SVM	NORMAL	0.91	0.90	0.90	0.92
	SHELL_ATTACKS	0.89	0.89	0.89	
	ACCESS_SENSITIVE_DIRS	0.92	0.90	0.91	
	STEAL_PROTOCOLS	0.93	0.88	0.90	
	FILL_RESERVED_WORDS	0.91	0.90	0.90	
	OTHER_ CYBER_ATTACKS	0.89	0.86	0.87	
LR	NORMAL	0.89	0.89	0.89	0.91
	SHELL_ATTACKS	0.87	0.88	0.87	
	ACCESS_SENSITIVE_DIRS	0.90	0.88	0.89	
	STEAL_PROTOCOLS	0.91	0.86	0.88	
	FILL_RESERVED_WORDS	0.91	0.90	0.90	
	OTHER_ CYBER_ATTACKS	0.88	0.86	0.87	
NB	NORMAL	0.87	0.87	0.87	0.89
	SHELL_ATTACKS	0.89	0.89	0.89	
	ACCESS_SENSITIVE_DIRS	0.86	0.80	0.83	
	STEAL_PROTOCOLS	0.89	0.88	0.88	
	FILL_RESERVED_WORDS	0.88	0.87	0.87	
	OTHER_ CYBER_ATTACKS	0.86	0.84	0.85	

Comparing the detection effect of the SVM model with that of the LR and NB models, the SVM model is also superior to other models in all aspects. This show that the SVM model is more appropriate in discerning the specific attack intent of the covert tunnel. The feature construction method in this paper is effective and SVM can be used as a classifier for covert tunnel specific attack intent detection [26,27].

Observation 1: (1) The keywords of shell attacks, access to sensitive directories, stealing communication protocol traffic, filling tunnel reserved words, and common network attacks are often found in the data portion of malicious traffic in ICMP covert tunnels, so extracting these types of keywords to construct features would be effective; (2) The SVM is suitable for high-dimensional, linearly indistinguishable data, and ICMP traffic happens to be high-dimensional and linearly indistinguishable, so SVM is more appropriate than other machine learning models.

5.2 Answer to Q3

The general method of using keywords for classification in machine learning is to use the collection of all words in the training set after sample word separation to form a vocabulary to form a large-dimensional vocabulary, which often requires further dimensionality reduction. In order to verify the effectiveness of this dimensionality reduction method, a comparative experiment before and after dimensionality reduction was constructed. The pre-dimensionalization experiment is to split the data part of each traffic in the training set with spaces and special symbols, and use the set of all words obtained after splitting as a feature dictionary, which contains about 30,000 words. The dimensionality reduction experiment adopts the feature dictionary construction method of this article. The other experimental steps are the same.

As shown in Tab. 6, after dimensionality reduction, the precision, recall, and F1 scores of each category are higher than those before dimensionality reduction by at least 0.02, 0.01, and 0.02, the accuracy rate is improved by 0.05, and the training time after dimensionality reduction is also reduced to about 1/8 of that before dimensionality reduction. This show that the dimensionality reduction method used in this paper is effective, not only improves the training speed, but also improves the evaluation indicators of each category. Before and after dimensionality reduction, the accuracy rate can be improved, and the time efficiency can be significantly improved.

In terms of CPU resource consumption, as shown in Fig. 6, after dimensionality reduction, the CPU usage during training is significantly lower than before. Only a single core is required to meet the training requirements, while before dimensionality reduction, an eight-core CPU is required to meet the training requirements. This shows that the dimensionality reduction method used in this paper can effectively reduce the consumption of CPU resources.

In terms of memory resource consumption, as shown in Fig. 7, the memory utilization during training after dimensionality reduction is 45.4% lower than that before dimensionality reduction. This shows that the dimensionality reduction method used in this paper can make more effective use of memory resources.

Table 6: Comparative experimental results before and after dimensionality reduction

Dimensions	Class	Precision	Recall	F1	Accuracy	Train time
About30000	NORMAL	0.86	0.80	0.83	0.87	4 min
	SHELL_ATTACKS	0.87	0.81	0.84		
	ACCESS_SENSITIVE_DIRS	0.89	0.88	0.88		
	STEAL_PROTOCOLS	0.80	0.80	0.80		
	FILL_RESERVED_WORDS	0.87	0.86	0.87		
	OTHER_ CYBER_ATTACKS	0.86	0.85	0.85		

(Continued)

Table 6: Continued

Dimensions	Class	Precision	Recall	F1	Accuracy	Train time
637	NORMAL	0.91	0.90	0.90	0.92	0.5 min
	SHELL_ATTACKS	0.89	0.89	0.89		
	ACCESS_SENSITIVE_DIRS	0.92	0.90	0.91		
	STEAL_PROTOCOLS	0.93	0.88	0.91		
	FILL_RESERVED_WORDS	0.91	0.90	0.90		
	OTHER_CYBER_ATTACKS	0.89	0.86	0.87		

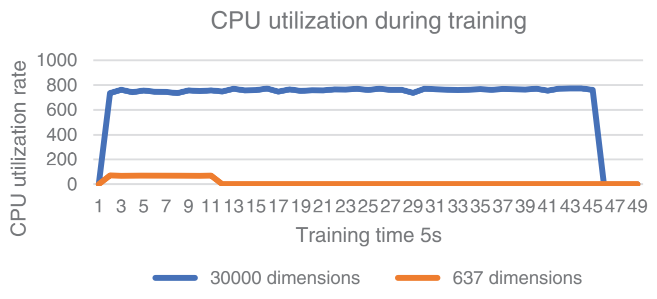

Figure 6: CPU utilization during training

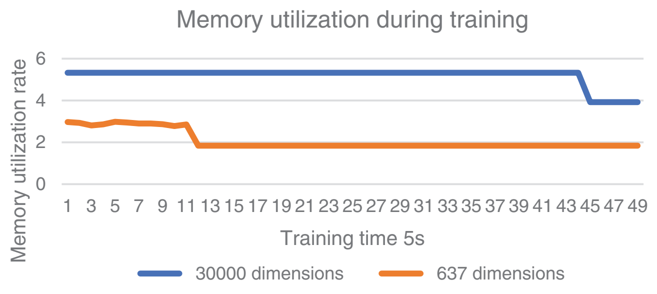

Figure 7: Memory utilization during training

Observation 2: Using the feature lexicon constructed in this paper, the dimension of the final feature vector is reduced from more than 30,000 to 637 compared to the dictionary constructed after direct word segmentation. Therefore, dimensionality reduction can undoubtedly improve the efficiency of the model. At the same time, choosing an appropriate dimensionality reduction method can filter out noise or irrelevant information, thereby helping the model to better learn the main features, and the model can obtain better detection results.

5.3 Answer to Q4

In order to clarify the gap of detection capability between attack intention detection based on multi classification and anomaly detection based on binary classification, we aggregate all types of malicious samples into malicious samples, and conducts anomaly detection experiments based on two classifications. And compare the results of the anomaly detection experiment with the results of the attack intention detection experiment.

The experimental results are shown in Tab. 7. The macro-average accuracy, macro-average recall, and accuracy of attack intent detection based on multi-classification are 0.05, 0.06, and 0.05

lower than that of anomaly detection based on two-classification, and the gap is controlled within 0.1. At the same time, the lowest accuracy rate and the lowest recall rate in the multi-classification also reached 0.89 and 0.86 respectively, achieving a better multi-classification effect.

Table 7: Comparison experiments of multi-classification-based attack intent detection and binary classification-based anomaly detection

Model	Category	Precision	Recall	Macro-Precision	Macro-Recall	Accuracy
Multi-Classification	NORMAL	0.91	0.90	0.91	0.89	0.92
	SHELL_ATTACKS	0.89	0.89			
	ACCESS_SENSITIVE_-DIRS	0.92	0.90			
	STEAL_PROTOCOLS	0.93	0.88			
	FILL_RESERVED_-WORDS	0.91	0.90			
	OTHER_ CYBER_-ATTACKS	0.89	0.86			
Binary classification	NORMAL	0.95	0.94	0.96	0.95	0.97
	ABNORMAL	0.96	0.96			

Observation 3: Compared with anomaly detection based on binary classification, attack intention detection based on multi-classification has better detection capabilities.

6 Conclusion

This paper uses ICMP data as the starting point to extract malicious attack intention keywords from five perspectives: shell commands, sensitive directories, communication protocol keywords, tunnel reserved words, and common network attack keywords, and build an ICMPTend detection model. Compared with the use of dictionary suffix cutting to construct feature vectors, it reduces noise interference and greatly reduces the dimensionality of feature vectors, which can clarify the attack intention of malicious traffic contained in the data part.

Funding Statement: This research was supported by National Natural Science Foundation of China (Grant Nos. 61972048, 62072051).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] Y. B. He, Y. F. Zhu and W. Lin, “HTTP tunnel trojan detection model based on deep learning,” *Journal of Physics: Conference Series,* vol. 1187, no. 4, pp. 1–11, 2019.
[2] M. Chen, X. J. Wang, M. S. He, L. Jin, K. Javeed *et al.,* “A network traffic classification model based on metric learning,” *Computers, Materials & Continua,* vol. 64, no. 2, pp. 941–959, 2020.

[3] C. L. Du, S. H. Liu, L. Si, Y. H. Guo and T. Jin, "Using object detection network for malware detection and identification in network traffic packets," *Computers, Materials & Continua,* vol. 64, no. 3, pp. 1785–1796, 2020.

[4] M. Aiello, M. Mongelli and G. Papaleo, "DNS tunnel detection through statistical fingerprints of protocol messages and machine learning," *International Journal of Communication Systems,* vol. 28, no. 14, pp. 1987–2002, 2015.

[5] C. L. Li, X. M. Sun and J. H. Cai, "Intelligent mobile drone system based on real-time object detection," *Journal on Artificial Intelligence,* vol. 1, no. 1, pp. 1–8, 2019.

[6] M. Crotti, M. Dusi, F. Gringoli and L. Salgarelli, "Detecting http tunnels with statistical mechanisms," in *Proc. 2007 IEEE Int. Conf. on Communications*, Glasgow, Scotland, pp. 6162–6168, 2007.

[7] P. Engelstad, B. Feng and T. van Do, "Detection of DNS tunnel in mobile networks using machine learning," in *Proc. Int. Conf. on Information Science and Applications*, Macau, pp. 221–230, 2017.

[8] J. K. Liu, S. H. Li, Y. S. Zhang, J. Xiao, P. Chang *et al.,* "Detecting DNS tunnel through binary-classification based on behavior features," in *Proc. 2017 IEEE Trustcom/BigDataSE/ICESS*, Sydney, Australia, pp. 339–346, 2017.

[9] A. Almusawi and H. Amintoosi, "Dns tunnel detection method based on multi-label support vector machine," *Security and Communication Networks,* vol. 2018, no. 2018, pp. 1–9, 2018.

[10] Y. J. Ding and W. D. Cai, "A method for HTTP-tunnel detection based on statistical features of traffic," in *Proc. 2011 IEEE 3rd Int. Conf. on Communication Software and Networks*, Xian, China, pp. 247–250, 2011.

[11] M. Dusi, M. Crotti, F. Gringoli and L. Salgarelli, "Tunnel hunter: Detecting application-layer tunnels with statistical fingerprinting," *Computer Networks,* vol. 53, no. 1, pp. 81–97, 2009.

[12] K. Borders and A. Prakash, "Web tap: Detecting covert web traffic," in *Proc. 11th ACM Conf. on Computer and Communications Security*, Washington DC, USA, pp. 110–120, 2004.

[13] T. Sohn, J. Moon, S. Lee, D. H. Lee and J. Lim, "Covert channel detection in the ICMP payload using support vector machine," in *Proc. Int. Sym. on Computer and Information Sciences*, Berlin, Heidelberg, pp. 828–835, 2003.

[14] Y. Insu, D. Kapil and T. Kim, "Automatic techniques to systematically discover new heap exploitation primitives," in *Proc. 29th USENIX Security Sym. (USENIX Security '20)*, pp. 1111–1128, 2020.

[15] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning,* vol. 20, no. 3, pp. 273–297, 1995.

[16] N. V. Sharma and N. S. Yadav, "An optimal intrusion detection system using recursive feature elimination and ensemble of classifiers," *Microprocessors and Microsystems,* vol. 85, pp. 104293, 2021.

[17] N. V. Sharma and G. Agarwal, "Network attacks and intrusion detection system: A brief," in *Proc. 2nd Int. Conf. on Intelligent Communication and Computational Techniques (ICCT)*, Manipal University Jaipur, pp. 280–283, 2019.

[18] H. Q. Lin, G. Liu and Z. Yan, "Detection of application-layer tunnels with rules and machine learning," in *Proc. Int. Conf. on Security, Privacy and Anonymity in Computation, Communication and Storage*, Georgia, United States, pp. 441–455, 2019.

[19] X. D. Xu, C. A. Wang and S. R. Zhu, "Covert channel detection in ICMP payload based on information entropy SVM," *Journal of Computer Applications,* vol. 2009, no. 2009, pp. 1–7, 2009.

[20] S. Sayadi, T. Abbes and A. Bouhoula, "Detection of covert channels over ICMP protocol," in *Proc. 2017 IEEE/ACS 14th Int. Conf. on Computer Systems and Applications*, Hammamet, Tunisia, pp. 1247–1252, 2017.

[21] B. L. Zhang, S. Zhou, L. Yang, J. H. Lv and M. J. Zhong, "Study on multi-label classification of medical dispute documents," *Computers, Materials & Continua,* vol. 65, no. 3, pp. 1975–1986, 2020.

[22] V. N. Vapnik, "An overview of statistical learning theory," *IEEE Transactions on Neural Networks,* vol. 10, no. 5, pp. 988–999, 1999.

[23] Z. T. Li, J. W. Kang, R. Yu, D. D. Ye, Q. Y. Deng *et al.,* "Consortium blockchain for secure energy trading in industrial internet of things," *IEEE Transactions on Industrial Informatics,* vol. 14, no. 8, pp. 3690–3700, 2017.

[24] Z. T. Li, J. Zhang, K. H. Zhang and Z. Y. Li, "Visual tracking with weighted adaptive local sparse appearance model via spatio-temporal context learning," *IEEE Transactions on Image Processing,* vol. 27, no. 9, pp. 4478–4489, 2018.

[25] Z. Li, W. Li, F. Lin, Y. Sun, M. Yang *et al.,* "Hybrid malware detection approach with feedback-directed machine learning," *Science China Information Sciences,* vol. 63, no. 139103, pp. 1–3, 2020.

[26] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science,* vol. 349, no. 6245, pp. 255–260, 2015.

[27] C. Schuldt, I. Laptev and B. Caputo, "Recognizing human actions: A local SVM approach," in *Proc. of the 17th Int. Conf. on Pattern Recognition*, Cambridge, United Kingdom, pp. 32–36, 2004.