Hindawi
Security and Communication Networks
Volume 2023, Article ID 1348393, 16 pages
https://doi.org/10.1155/2023/1348393

*Research Article*

# HTTP Cookie Covert Channel Detection Based on Session Flow Interaction Features

**Wenxin Yuan**,[1,2] **Xingshu Chen**,[1,2] **Yi Zhu**,[2] **Xuemei Zeng**,[2] **and Yawei Yue**[1]

[1]*School of Cyber Science and Engineering, Sichuan University, Chengdu 610065, China*
[2]*Cyber Science Research Institute, Sichuan University, Chengdu 610065, China*

Correspondence should be addressed to Xingshu Chen; chenxsh@scu.edu.cn

Received 15 December 2022; Revised 18 March 2023; Accepted 18 May 2023; Published 26 June 2023

Academic Editor: Zhiyuan Tan



HTTP cookie covert channel is a covert communication method that encodes malicious information in cookie fields to escape regulatory audits. It is difficult to detect this kind of covert channel according to the cookie content because cookie fields are mainly encoded in custom modes. To effectively identify the HTTP cookie covert channel, this paper proposes a detection method based on the interaction features of the session flow. First, we split the HTTP session flow into fine-grained "interaction process" subflows to comprehensively describe the communication process of the cookie. Then, we compare and analyze the differences between HTTP cookie covert channels and normal cookie communications based on the interaction process, design three types of 7-dimensional features, and build the detection model combined with the machine learning algorithm. Experimental results show that our method can effectively detect HTTP cookie covert channels, and the detection rate can reach 99%. We also prove that our method has advantages in stability and time performance compared with the existing detection methods through experiment and analysis. In addition, our method has certain practicability in the simulation environment with imbalanced data.

## 1. Introduction

Covert channel refers to the secret transmission of messages over channels that are not designed to transmit information, which is achieved by using resources shared by the sender and the receiver to encode covert information [1]. Shared resources that can be used to construct covert channels include data packets in the network, and data transmission time [2, 3]. In some new scenarios, even resources such as cache [4] and optical frequency comb [5] can be used to encode covert information. At present, the application layer protocol HTTP that still exists in a large number in the network is widely used to design covert channels due to its own flexibility. The HTTP cookie covert channel delivers information by writing covert data in custom encoding modes in the cookie field of the HTTP protocol. Typically, standard intrusion detection systems will not inspect the cookie content. In addition, the encoding formats of cookies are usually specified separately by different servers [6], making HTTP cookie covert channels well concealed.

The HTTP cookie covert channel is a covert channel that directly manipulates network protocol data. In fact, many covert channels manipulate random values or strings in the protocol in some way [7]. Although the random value mode covert channel has good concealment, the transmission efficiency is low due to the limited utilization of resources. URL is another common string used to construct covert channels in the HTTP protocol. However, compared with the cookie, the URL contains more semantic information, resulting in relatively poor covertness. In recent years, many attack groups have used HTTP cookie covert channels to carry out network attacks, seriously endangering cyber security. In 2016-2017, after implanting the malicious program into the victim host, the APT10 organization [8] would collect the hostname, process identifier, current working directory, window resolution, and Microsoft Windows version of the victim host and would encrypt and encode them in the cookie field to send them to the command and control (C2) server. In 2016-2017, the backdoor program implanted into the victim host by the APT15 organization

[9] would transmit relevant parameters of the victim host and query commands to the malicious server through the cookie field. The data transmitted in the cookie would first be encrypted by AES-CBC and then encoded by Base64. In 2019–2021, in order to establish a secure session with the C2 server, the backdoor implanted into the victim host by the APT29 organization [10] would hard-code the commands that request the session key into the cookie field and would send it to the C2 server.

Therefore, effective detection of HTTP cookie covert channels can detect and block related network attacks in time and can also provide a basis for attack source tracing, which is of great significance to protecting cyber security. Currently, there are many types of research on covert channel detection, which according to different analysis methods, can be classified into covert channel detection based on packet payload features, covert channel detection based on flow behavior statistical features, and covert channel detection based on deep learning automatic feature mining. In most cases, HTTP cookie covert channels and normal HTTP cookie communications behave similarly in terms of packet payload and flow behavior, so detection based on packet payload features and flow behavior statistical features cannot identify HTTP cookie covert channels well. Moreover, detection based on deep learning and automatic feature mining often results in large model training overheads.

To solve the above problems, we propose an HTTP cookie covert channel detection method based on the interaction features of session flow in this paper. We extract features based on the interaction principles of HTTP cookie session flow to achieve effective detection of the HTTP cookie covert channel. The main contributions of this paper are as follows:

(1) On the basis of aggregating the HTTP session flow, the HTTP session flow is split into "interaction process" subflows according to cookie interaction behavior characteristics so that the effective features of the HTTP cookie covert channel can be extracted.

(2) Through the network behavior analysis method, three types of behavior features of the HTTP cookie covert channels are extracted and combined with the traditional machine learning algorithm to build a detection model so as to achieve effective detection of HTTP cookie covert channels.

(3) Experiments and theoretical analysis prove that the detection method proposed in this paper has advantages in stability and time performance compared with the existing detection methods. Furthermore, in the simulation environment with imbalanced data, our method still has certain practicability.

The rest of this paper is organized as follows. Section 2 introduces current covert channel detection methods. Section 3 introduces the framework of the detection method proposed in this paper. Section 4 details our detection method. In Section 5, we conducted experiments to verify the effectiveness and superiority of the detection method proposed in this paper. Section 6 concludes the paper with a prospect of future research.

## 2. Related Work

In this section, we introduce existing covert channel detection methods. The covert channel detection method is divided into two main steps: feature extraction and classification.

*2.1. Covert Channel Feature Extraction.* Unlike stealing cookies, the cookie field of the HTTP cookie covert channel is constructed by the attacker according to a custom encoding method and conforms to the standard format rather than being stolen or intercepted from legitimate users. Therefore, the critical point of feature extraction lies in the portrayal of covert behavior rather than the content analysis of the cookie. The existing covert channel feature extraction methods are divided into two types depending on the different perspectives of portraying covert behavior.

The first type of method extracts features based on packet payloads and portrays covert behavior at the packet level. Several works [11–13] extracted information such as field difference, entropy value, and semantic content from the specific positions in HTTP packets as features to characterize HTTP covert channels. However, such methods can only extract features for specific types of covert channels and cannot effectively characterize the HTTP cookie covert channel. The second type of method extracts behavior statistical features by aggregating network flows and portrays covert behavior at the network flow level. Some works [14–16] mainly extracted the time information of the network flow as the feature to detect covert timing channels that use the time to encode information. Such methods obviously cannot detect HTTP cookie covert channels which do not use the time to encode information. Several works [17–20] constructed multidimensional features from raw data such as packet size, packet interval, transport layer flag information, and ports in the network flow to portray covert behavior from multiple perspectives. Such methods can effectively characterize traditional covert communication behaviors. However, these general flow features are not directly related to the encoding of the cookie, so such methods are unstable when facing the problem of detecting HTTP cookie covert channels, they may be at risk of being bypassed.

*2.2. Covert Channel Classification.* The classification methods for covert channels are divided into three types. The first type of method classifies covert channels by setting thresholds for a small number of features. Relevant works [11, 12, 15] achieved the detection of covert channels by setting a threshold for a single complex feature. Such methods have good detection effects for specific types of data but are relatively prone to low robustness of the detection model due to the low dimension of the extracted features. The second type of method classifies covert channels by traditional machine learning algorithms. Relevant works

[13, 16–20] extracted multidimensional features and input them into traditional machine learning classifiers to classify covert communications. The classification ability of such methods depends on the effectiveness of feature extraction. As long as the effective features of the covert channel can be extracted, a model with good detection ability can be obtained. The third type of method uses deep learning techniques to automatically mine features from raw network traffic for classification without relying on expert knowledge. Relevant works [21–23] encoded the original network traffic and used it as the input data of the deep learning classifier to detect covert communications. On the one hand, when the payload of malicious traffic is not very different from that of normal traffic, for example, when there is no significant difference in cookie fields between the HTTP cookie covert channel and normal communication, such methods that use the original payload as input may not always have good detection effects, while the interpretability of the method is poor. On the other hand, such detection methods rely on deep learning techniques, which will cause enormous time overhead in model training.

Table 1 shows the main information of the above covert channel detection methods and summarizes their defects in detecting HTTP cookie covert channels. In summary, existing detection methods are not good at detecting HTTP cookie covert channels. To solve this problem, in this paper, we analyze the differences in session flow between HTTP cookie covert channels and normal HTTP cookie communications based on the function of the cookie and the interaction characteristics of HTTP session flow to extract effective features and achieve effective detection of HTTP cookie covert channels combined with the machine learning algorithm.

## 3. Detection Framework

The overall framework of the HTTP cookie covert channel detection method proposed in this paper is shown in Figure 1, and it consists of three phases: (1) Data processing. In this phase, we first aggregate the original network packets into session flows, and then filter the data to obtain the HTTP session flows that complete communication successfully and contain cookie fields, and finally extract "interaction process" subflows. (2) Feature extraction. In this phase, we extract three types of features based on the interaction process: cookie distribution features, response content distribution features, cookie and related data relationship features, and generate feature vectors with category labels. (3) Detection. In this phase, we use supervised machine learning algorithms to train and classify the data and determine whether the input session flow is an HTTP cookie covert channel or a normal HTTP cookie communication.

The main contributions of the above detection framework are as follows: (1) We innovatively propose the definition "interaction process," which can comprehensively describe the communication process of the cookie and is the basis of effective feature extraction. Moreover, we detail the process of extracting the interaction process. (2) The features we extract not only have good detection effects in various classification algorithms but also have strong interpretability. (3) Experiments show that our detection method is superior to the existing detection methods to a certain extent and has certain practicability.

## 4. Detection Method

In this section, we detail the detection method we propose. We first analyze why this kind of covert channel is difficult to detect, and combined with the communication principle of HTTP, propose the definition of "interaction process," which comprehensively describes the communication process of the cookie. Then, we detail the method to extract the interaction process, based on which detail the method to extract features according to the analysis of differences between HTTP cookie covert channels and normal HTTP cookie communications. Finally, we introduce the classification method we choose.

*4.1. Problem Analysis.* The cookie is a small piece of data that the server first sends to the client when using the HTTP protocol and it will be carried the next time the client sends requests to the server. The main functions of the cookie include: (1) Session state management. Enables the stateless HTTP protocol to stably record session state information. (2) User behavior tracking. Push specific content through the analysis of user behavior.

The latest standard relating to the cookie specification, RFC 6265 document [24], does not require the encoding method of the cookie, which provides conditions for attackers to arbitrarily encode the cookie field to build covert channels. Moreover, many normal servers can also customize the encoding or encryption method of the cookie.

The process of HTTP cookie covert channel generation is shown in Figure 2. First, the attacker will implant the malicious backdoor program into the victim host, and the implanting methods include file bundling, email attachments, and web page implantation. In order to bypass defense mechanisms such as firewalls and intrusion detection systems and communicate with the malicious server, the backdoor program will encode malicious messages to be delivered in the cookie field of the HTTP protocol. After decoding the cookie, the malicious server will launch further attacks on the victim host according to the information provided by the backdoor.

Table 2 shows examples of the cookie in a normal communication and a covert channel, respectively. The cookie of the covert channel is captured during the communication of the malicious program ChChes [8], and it can be decoded as "AUSER-PC ∗ 1620?3618468394?C:\\Users\\admin\\AppData\\Local\\Temp?1.4.1 (1280 × 720) ∗ 6.1.7601.24545." It can be seen that this HTTP cookie covert channel leaks relevant information about the victim host. However, the examples in Table 2 both conform to the standard cookie format defined in RFC 6265: separating key-value pairs with a semicolon. In addition, it can be inferred from the unreadable fields that either are encrypted or

Table 1: Summary of covert channel detection methods.

| Reference | Year | Extracted features | Classification method | Defects |
|---|---|---|---|---|
| [11] | 2018 | Character difference in HTTP header field | Set thresholds | Can only achieve detection of specific HTTP covert channels |
| [12] | 2020 | Relative entropy between HTTP header field probability matrices | Set thresholds | |
| [13] | 2021 | Word-level and character-level high order semantic features of HTTP request text | Traditional machine learning | |
| [14] | 2019 | Packet interval time | Deep learning | Can only detect covert timing channels |
| [15] | 2020 | Packet interval time | Set thresholds | |
| [16] | 2020 | Packet interval time | Traditional machine learning | |
| [17] | 2012 | Statistical features of packets, packet interval time, and flow duration | Traditional machine learning | Unstable, general flow statistical features are not directly related to the encoding of the cookie |
| [18] | 2015 | Uplink and downlink traffic features, small packets features, packet interval, and transport layer flag | Traditional machine learning | |
| [19] | 2019 | The number, duration, port information, dissimilarity, and average length ratio of sending to receiving of flow | Traditional machine learning | |
| [20] | 2022 | General features of network flow | Traditional machine learning | |
| [21] | 2017 | Convert traffic into images and feed them into deep learning for automatic feature extraction | Deep learning | Poor interpretability, long model training time, when the character level difference is not obvious, the detection effect may not be good |
| [22] | 2018 | Encode and aggregate traffic into matrices and feed them into deep learning for automatic feature extraction | Deep learning | |
| [23] | 2020 | Extract a certain amount of payload within the flow and feed it into deep learning for automatic feature extraction | Deep learning | |

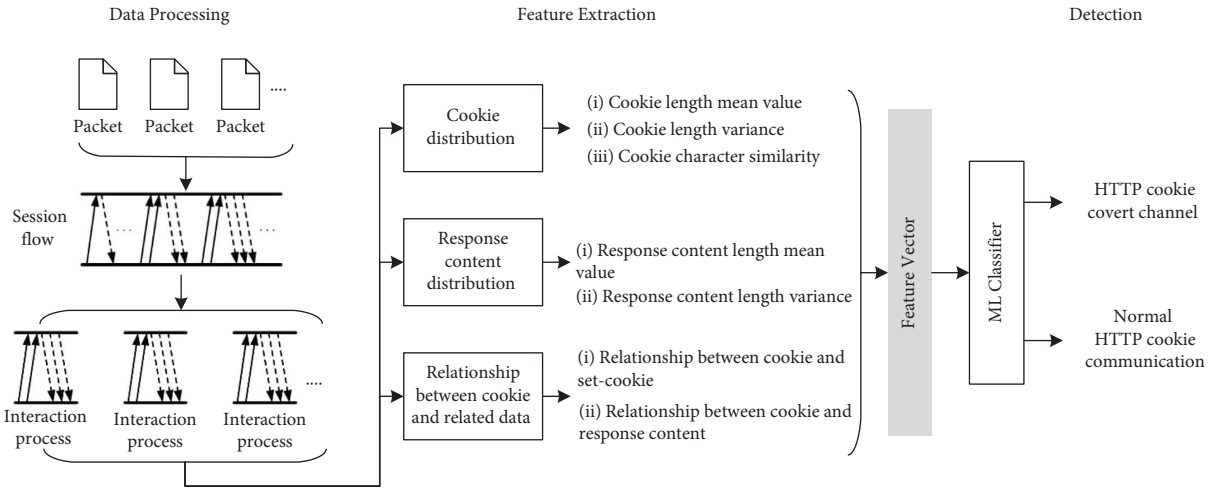


Figure 1: HTTP cookie covert channel detection framework.

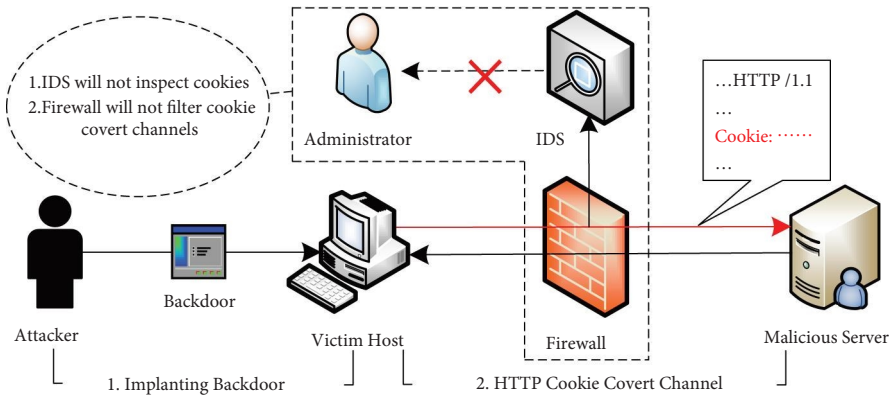


Figure 2: HTTP cookie covert channel generation process.

Table 2: Cookie examples.

| Category | Cookie content |
|---|---|
| Normal communication | lkqdidts = 1499100526; lkqdid = uZRUClndZ2M; p_0 = H4sIAAAAAAAAAJXKMQ6DMAwAwIkKRZ2YeEAXhIywk5D4NZW dOFIf0E_yE7rxA7bu3HzO949hKt9PfStR44YeYjMDREuQWSIgsVH1qiR5P H_H3rmlfw6vgKu2LAkYOUIo4kEoGWyxWsioXDT8_3zzL3f-BUl0ezPFAAAA |
| Covert channel | sKk = aiTLT6Y2GVgJJlLKk%2Bvw Ds7X%2BoR828cfLHACuRf3lRYTvn%2FQRr30OnXnpgI%3D; CMgnxG1BWA = 2xZzPwGZXPbIuaC05wEqZfKBA35vW1Sv OIl8h7BVfT%2BDDDQE4p0S; LljG5YM = 9KU9iG% 2BIkxtwXsBuBjPWYcClMkqZVS6gkQnC |

encoded in a particular way. The encoded cookie field is essentially uncertain information. At present, there are many uncertain information processing works [25, 26]. However, the cookie encoding of normal communication and covert channel can be arbitrary and irregular, so it is difficult to detect the HTTP cookie covert channel from the character-level dimension of the packet field.

One of the cookie functions is session state management, so in this paper, we consider detecting HTTP cookie covert channels at the session flow level. Common covert channel detection features at the session flow level include packet length, number of packets, and packet interval. These features can achieve coarse-grained characterization and detection of covert channels from the perspective of flow behavior without considering the construction principles. However, as the behavior of the HTTP cookie covert channel flow constructed by attackers becomes more and more similar to the normal flow, the detection effects of the above features will gradually worsen. In order to extract effective features in the session flow, we next analyze the communication process of the HTTP session flow that uses the cookie field.

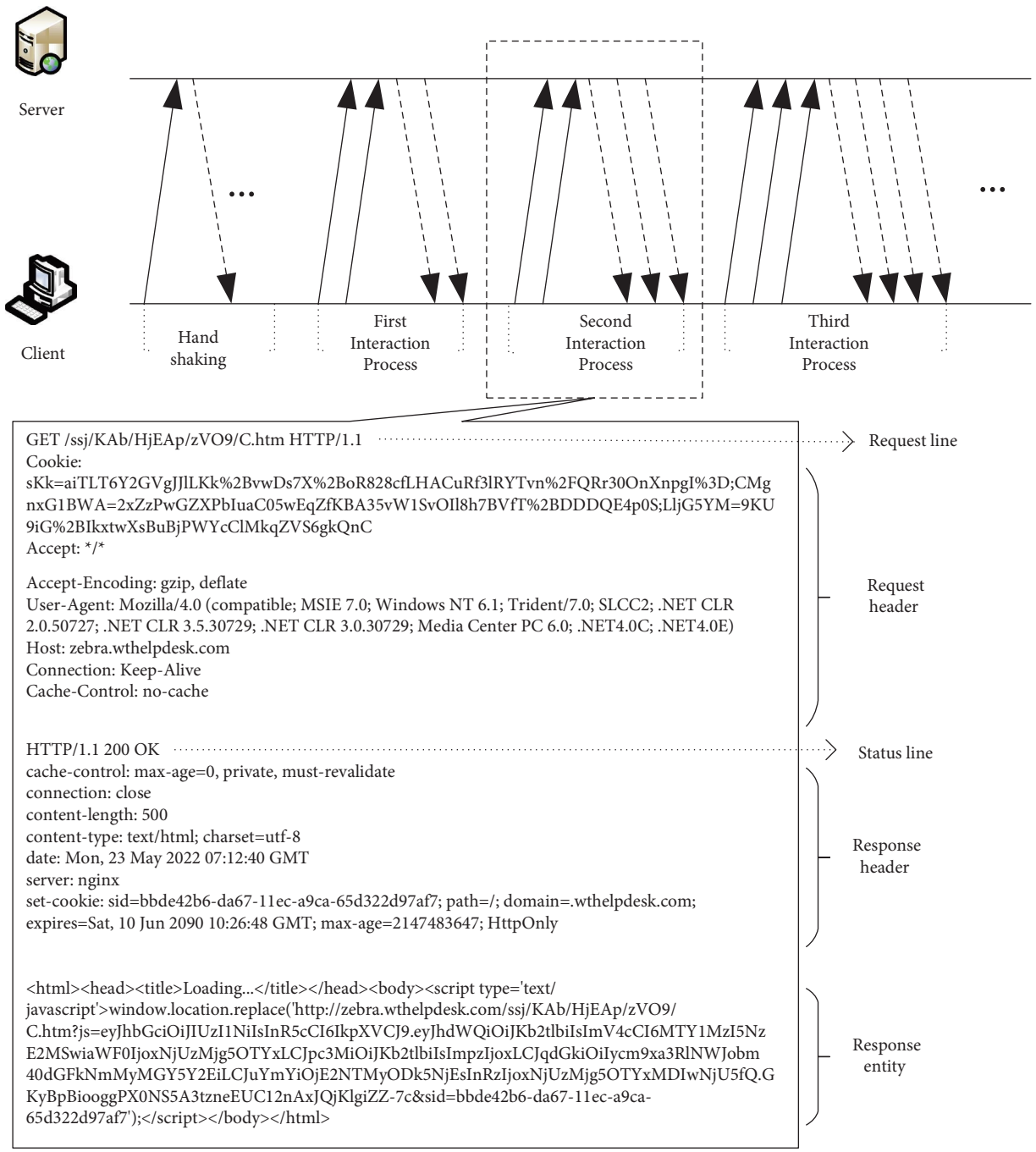


Figure 3: HTTP cookie session flow example.

An example of the HTTP session flow that uses cookie fields is shown in Figure 3. It is an example of a covert channel, and actually, there are no differences between normal communication and covert channel in session flow structure or packet structure. The session flow begins with the handshake phase. After the TCP connection is successfully established, the client and the server officially start to transmit HTTP packets. At this stage, the client first sends a single or a set of continuous packets to the server to request HTTP content, and then the server returns a single or a set of continuous packets to the client. Based on the above analysis, this paper proposes the definition of the interaction process.

*Definition 1* (Interaction process). In an HTTP session flow, a single or a set of continuous HTTP packets sent by the client to the server and the subsequent single or a set of continuous HTTP packets sent by the server to the client constitute an interaction process.

In an interaction process, the application layer payload in packets sent by the client to the server will constitute the HTTP request line, request header, and request data. And the application layer payload in packets sent by the server to the client will constitute the HTTP status line, response header, and response entity. In general, when the client does not cache information of the server, the first few interaction

processes will not contain cookie fields. When the server sends the client a set-cookie field defining the cookie content, subsequent interaction processes will include cookie fields. So the interaction process can comprehensively describe the communication process of the cookie.

*4.2. Interaction Process Extraction.* The HTTP session flow contains packets for establishing, maintaining, and disconnecting TCP connections, which do not constitute the actual content of the HTTP request or response, so such packets need to be filtered out when extracting the interaction process. The filter condition is: discard packets with the application layer payload length of 0 in the HTTP session flow.

Next, the remaining packets in the session flow need to be labeled, as shown in Figure 4. Each interaction process begins with a packet labeled 0 or 1, followed by a set of continuous packets labeled 2. And when there follows a packet labeled 0 or 1, it means that a new interaction process is started. The specific method for interaction process extraction is shown in Algorithm 1. In the sequence of the labeled packets, if a packet labeled 0 or 1 is found, followed by several packets labeled 2, store these packets in a subsequence, representing an interaction process. The resulting final sequence stores all interaction processes of an HTTP session flow.

*4.3. Feature Extraction.* We analyze the differences between HTTP cookie covert channels and normal cookie communications based on the interaction process and design three types of 7-dimensional features.

(1) Cookie distribution features. Some covert channels will transmit a large amount of malicious information, so the length of the encoded cookie field is relatively large. Moreover, HTTP uses the persistent connection by default, that is, multiple requests from the client to the server use a single connection. So in the same session flow, the cookies of each interaction process are strongly similar, and their differences only exist in the key-value pairs inside a few cookies. Covert channels usually use cookie fields to encode different malicious information at different stages in a session flow, so the similarity among multiple cookies is weak. The similarity can be fully characterized by the length dispersion and character-level similarity of multiple cookies in a session flow. Therefore, in a session flow, compared with normal communications, the length distribution of multiple cookies in covert channels is more discrete, and character-level differences of multiple cookies in covert channels are greater. Based on the above analysis, the extracted features are as follows:

(1) Average cookie length in a session flow. Extract the length of the cookie from each interaction process containing the cookie to get the sequence $\{cookieLen_1, cookieLen_2, ..., cookieLen_n\}$, and calculate its average value:

$$\text{MeanCookieLen} = n^{-1} \sum_{i=1}^{n} \text{cookieLen}_i, \tag{1}$$

(2) Cookie length variance in a session flow. Calculate the variance of the above cookie length sequence:

$$\text{VarCookieLen} = n^{-1} \sum_{i=1}^{n} \left(\text{cookieLen}_i - \text{MeanCookieLen}\right)^2. \tag{2}$$

(3) Adjacent cookie similarity in a session flow. Extract the field content of the cookie from each interaction process containing the cookie to get the sequence $\{cookie_1, cookie_2, ..., cookie_n\}$. To evaluate the character-level similarity of all cookies in a sequence, calculating the similarity score for each pair and then taking the average will cause a huge time overhead and is not interpretable. So, we calculate the character-level similarity score of adjacent cookies in the sequence and take the average value, which reduces the time overhead on the one hand. On the other hand, cookies may gradually change over time in the session flow of normal communications, so cookies that are separated over a long period of time in normal communications may also differ significantly. The character-level difference between adjacent cookies is theoretically smaller in normal communication than in covert channels. The calculation method is shown in Algorithm 2.

In Algorithm 2, the similarity calculation function *CaculateSimilarity* is based on the Gestalt pattern matching approach [27]. The core idea is, given two strings $S_1$ and $S_2$, first find their longest common substring, then recursively keep searching for the longest common substring at the remaining unmatched positions of the two strings until there is no common substring in each subregion of the two strings. And calculate

$$\text{Similarity} = \frac{2K_m}{|S_1| + |S_2|}. \tag{3}$$

In formula (3), $K_m$ represents the number of characters of all the longest common substrings in the whole process, and $|S_1|$ and $|S_2|$ represent the lengths

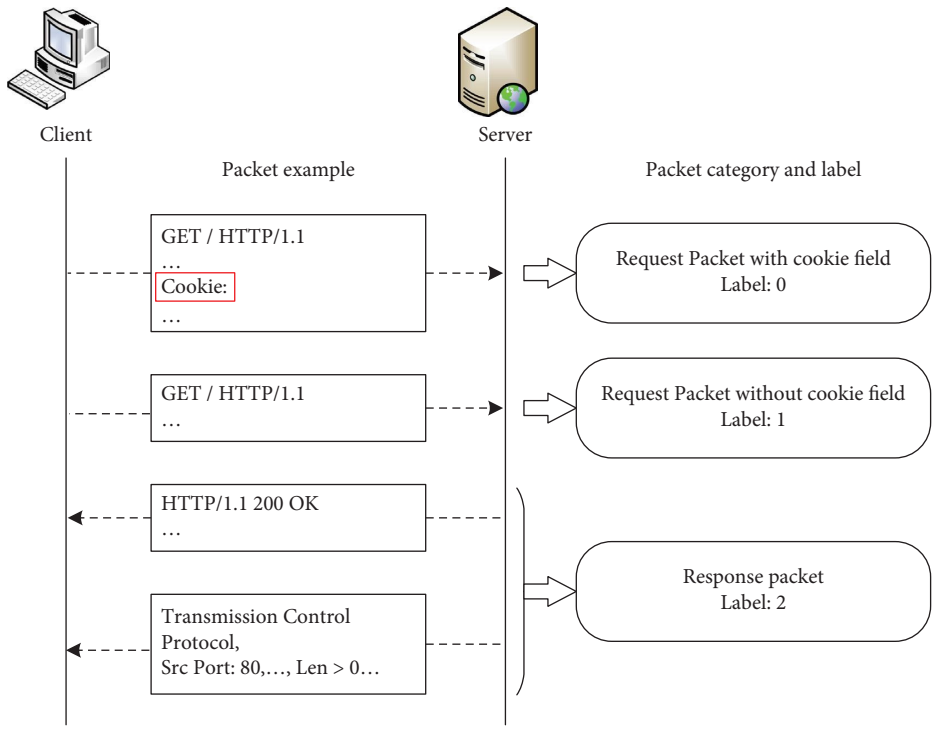


Figure 4: Packet labels.

**Input:** Labeled packets in the session flow: Packet = {packet1, packet2, ..., packetN}
**Output:** Interaction process sequence
Tmp ← []
Interaction ← []
**for** $packet_i \in$ Packet **do**
**if** $packet_i$.label != 2 && $packet_{i+1}$.label == 2 **then**
Tmp.append ($packet_i$)
$j \leftarrow 1$
**while** $i + j < N$ && $packet_{i+j}$.label == 2 **do**
Tmp.append ($packet_{i+j}$)
$j \leftarrow j + 1$
**end while**
Interaction.append (Tmp)
Tmp.clear
**end if**
**end for**
**return** Interaction

Algorithm 1: Interaction process extraction algorithm.

of the two strings, respectively. The value range of the calculation result is (0, 1). And the closer the result is to 1, the more similar the two strings are; the closer the result is to 0, the more different the two strings are.

(2) Response content distribution features. Take the most common HTTP application web as an example. Due to the diversity of web page elements requested by clients, the response content returned by the server is also diverse in different interaction processes in the session flow. While in order to improve the data transmission rate, malicious servers using HTTP cookie covert channels usually only pay attention to the legality of the header structure of the HTTP response packets. So the response content of the session flow of the HTTP cookie covert channel in different interaction processes presents the characteristics of similar content and small length. Based on the above analysis, the extracted features are as follows.

(4) Average response content length. The response content length is obtained by accumulating the

**Input:** Cookie content sequence {$cookie_1$, $cookie_2$, ..., $cookie_n$}
**Output:** Adjacent cookie similarity in a session flow
  **if** $n < 2$ **then**
    **return** $-1$
  **end if**
  **for** $i = 1$ to $n - 1$ **do**
    $Similarity_i \leftarrow ||CaculateSimilarity\ (cookie_i, cookie_{i+1})||$
  **end for**
  $sum \leftarrow 0$
  **for** $i = 1$ to $n - 1$ **do**
    $sum \leftarrow sum + Similarity_i$
  **end for**
  **return** $sum/(n-1)$

Algorithm 2: Adjacent cookie similarity in a session flow algorithm.

Table 3: Data collecting and processing information.

| Data category | Data source | Filtered flows | Total flows |
|---|---|---|---|
| Normal | ISCXIDS2012 dataset | 513 | 1304 |
| Normal | CTU-normal | 791 | |
| Abnormal | Malicious program MD5: c0c8dcc9dad39da8278bf8956e30a3fc | 82 | 1045 |
| Abnormal | Malicious program MD5: 07abd6583295061eac2435ae470eff78 | 78 | |
| Abnormal | Malicious program MD5: 37c89f291dbe880b1f3ac036e6b9c558 | 181 | |
| Abnormal | Malicious program MD5: 3afa9243b3aeb534e02426569d85e517 | 182 | |
| Abnormal | Malicious program MD5: f03f70d331c6564aec8931f481949188 | 178 | |
| Abnormal | Malicious program MD5: 684888079aaf7ed25e725b55a3695062 | 175 | |
| Abnormal | Malicious program MD5: 7891f00dcab0e4a2f928422062e94213 | 169 | |

Table 4: Comparison results on different classification algorithms.

| Classifier | Accuracy | Precision | Recall | $F1$ score |
|---|---|---|---|---|
| Random Forest | **0.99858** | **1.0** | **0.99681** | **0.99840** |
| Decision Tree | 0.99007 | 0.99041 | 0.98726 | 0.98883 |
| SVM | 0.98014 | 0.96296 | 0.99363 | 0.97805 |
| KNN | 0.97588 | 0.95412 | 0.99363 | 0.97347 |
| Naive Bayes | 0.86524 | 0.99103 | 0.70382 | 0.82309 |

Bold values in Table 4 represent the best result in each indicator.

application layer payload length of all response packets for an interaction process, and the sequence {$ResConLen_1$, $ResConLen_2$, ..., $ResConLen_m$} is obtained in a session flow, calculate its average value:

$$\text{MeanResConLen} = m^{-1} \sum_{i=1}^{m} \text{ResConLen}_i. \quad (4)$$

(5) Response content length variance. Calculate the variance of the above response content length sequence:

$$\text{VarResConLen} = m^{-1} \sum_{i=1}^{m} \left(\text{ResConLen}_i - \text{MeanResConLen}\right)^2. \quad (5)$$

(3) Cookie and related data relationship features. In general, the position where the information is actually transmitted in the network traffic will store the longer content. The actual position for the HTTP cookie covert channel to transmit information is the

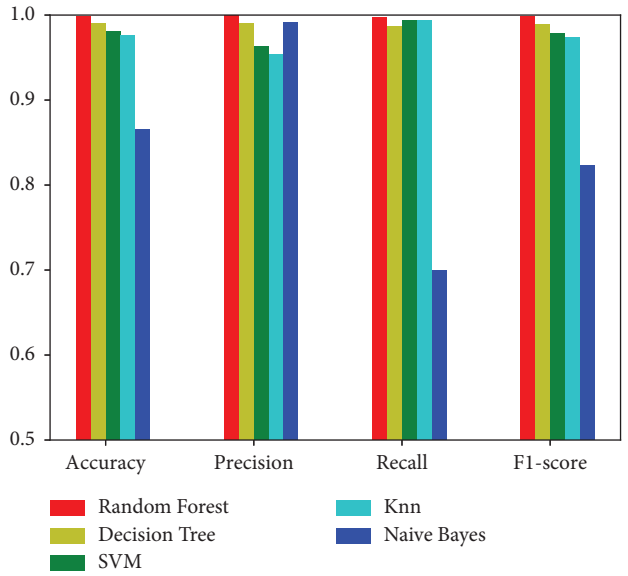


Figure 5: Comparison with different algorithms.

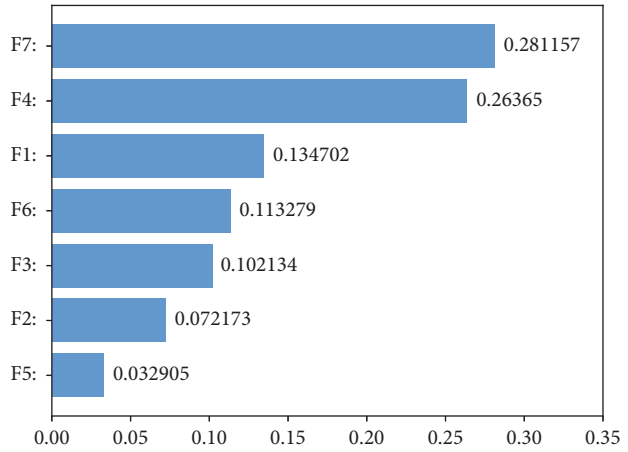


Figure 6: Contribution score of each feature.

Table 5: Features summary.

| Symbol | Statistics | Contribution ranking |
|---|---|---|
| F1 | Information quantity of cookie | 3 |
| F2 | Dispersion degree of cookie length | 6 |
| F3 | Character level difference of adjacent cookies | 5 |
| F4 | Information quantity of response content | 2 |
| F5 | Dispersion degree of response content length | 7 |
| F6 | Information quantity comparison of response content and cookie | 4 |
| F7 | Information quantity comparison of set-cookie and cookie | 1 |

cookie field, while the main position for normal HTTP communication to transmit information is the response content returned by the server to the client. So in each interaction process of a session flow, calculating the ratio of the content lengths stored in the above two locations and then calculating the average value in the session flow can describe the difference between normal communications and covert channels to a certain extent. In addition, some covert channels will also respond to the malicious information passed in the cookie field through the set-cookie field of the HTTP response packets. Therefore, in each interaction process of a session flow, calculating the ratio of the set-cookie

field length to the cookie field length and then calculating the average value in the session flow can reflect the individual difference between such covert channels and normal communications. Based on the above analysis, the extracted features are as follows:

(6) Relationship between cookie and response content. For all interaction processes that contain cookies, calculate the ratio of the response content length to the cookie length, and take the average value in the session flow:

$$\text{CookieResCon}_{\text{relation}} = n^{-1} \sum_{i=1}^{n} \frac{\text{ResConLen}_i}{\text{cookieLen}_i}. \quad (6)$$

(7) Relationship between cookie and set-cookie. For all interaction processes that contain cookies, calculate the ratio of the set-cookie length to the cookie length, and take the average value in the session flow:

$$\text{CookieSetcookie}_{\text{relation}} = n^{-1} \sum_{i=1}^{n} \frac{\text{setcookieLe}n_i}{\text{cookieLe}n_i}. \quad (7)$$

*4.4. Classification Method.* We extract a total of 7 numerical features from three perspectives, so it is impossible to use the detection method that sets a threshold for a single feature. Moreover, to obtain a detection model with a better classification effect and shorter training time in small sample data, we implement the detection of HTTP cookie covert channels based on the traditional supervised machine learning algorithm rather than the deep learning method.

## 5. Experiments and Analysis

In this section, we introduce the data used for the experiments and the experimental procedures and analyze the experimental results.

*5.1. Data Collecting and Processing.* To ensure the purity and diversity of the normal data used in the experiment, the pcap files in the ISCXIDS2012 dataset [28] and CTU dataset [29], which have been marked as collected from normal network activity, were selected as the normal data samples. There are no public datasets for HTTP cookie covert channels currently. We obtained 7 types of backdoor programs in the ChChes malware family [8] that use HTTP cookie covert channels from threat intelligence, and executed them in the virtual machine of the 32-bit Windows7 system and collected the traffic of the virtual machine. The malicious servers that communicate with these backdoor programs were still alive, but they had their own stealth mechanisms. When the backdoor was frequently executed, the malicious server would respond to the status information of "429 Too Many Requests," and the communication recovery time was unknown. So the amount of data collected is relatively limited.

Next, we aggregated the original packets through the IP addresses and ports of the two communicating parties and the transport layer protocol to obtain the bidirectional session flow. On this basis, the session flow was truncated according to the two time standards proposed in the paper [30]: (1) Truncate the session flow when its duration exceeds 1800 seconds. (2) Truncate the session flow when no new packets are generated between the two communicating parties within 120 seconds. The resulting session flows contained various protocols of the transport layer and the application layer, which needed to be filtered to obtain the HTTP session flows with the cookie field and successful communication. The logic of data filtering is as follows: first, determine whether there is at least one packet in the session flow with the HTTP application layer protocol and the cookie field. If the condition is true, retain this session flow; otherwise, discard it. Then determine whether there is at least one HTTP response packet in the session flow with a status code of 200. If the condition is true, retain this session flow; otherwise, discard it. The final data collecting and processing information in this paper is shown in Table 3.

*5.2. Feature Validity Experiment.* In order to verify the validity of the proposed features, we constructed supervised classifiers using Random Forest, SVM, Decision Tree, Naive Bayes, and KNN, respectively. After feature extraction and labeling of the filtered session flows, we divided them into a training set and a test set according to 7 : 3 while ensuring that the proportion of positive and negative samples in the two parts of data was the same. Then trained each classifier and compared the results on common evaluation indicators: Accuracy, Precision, Recall, and $F1$ score. The results are shown in Table 4 and the results are compared, as shown in Figure 5. Bold values in Table 4 represent the best result in each indicator. The experimental results show that extracted features perform well in most supervised classifiers, and the performance of Random Forest algorithm is the best.

Then, in order to evaluate the contribution of each feature, we utilized the Random Forest to get the contribution score of features F1–F7 (following the order described in Section 4.3). The higher the score is, the greater the feature's contribution is. The results are shown in Figure 6. It can be seen that the contribution of each feature is relatively average, which indicates our extracted features are comparatively effective to detect HTTP cookie covert channel. Table 5 summarizes all the features.

*5.3. Comparative Experiment.* Next, we compared our detection method with the two types of existing covert channel detection methods.

To realize the detection of various covert channels generated by malware from the perspective of flow behavior without considering the construction principle, the paper [31] extracted 28-dimensional general flow statistical features combined with existing work, as shown in Table 6. The features in Table 6 have relatively good detection results for various covert channels under several commonly used machine learning algorithms. Paper [23] extracted a certain

Table 6: General flow statistical features.

| Features | | | |
|---|---|---|---|
| Total bytes sent | Bytes sent proportion | Total bytes received | Bytes received proportion |
| Packet length mean | Packet length median | Packet length mode | Packet length variance |
| Packet length standard deviation | Packet length coefficient of variance | Packet length skew from median | Packet length skew from mode |
| Packet time mean | Packet time median | Packet time mode | Packet time variance |
| Packet time standard deviation | Packet time coefficient of variance | Packet time skew from median | Packet time skew from mode |
| Request/response time mean | Request/response time median | Request/response time mode | Request/response time variance |
| Request/response time standard deviation | Request/response time coefficient of variance | Request/response time skew from median | Request/response time skew from mode |

Table 7: Comparison results on different methods.

| Method | Accuracy | Precision | Recall | *F*1 score |
|---|---|---|---|---|
| Our method | **0.99858** | **1.0** | 0.99681 | **0.99840** |
| Paper [31] | 0.99756 | 0.99751 | **0.99751** | 0.99751 |
| Paper [23] | 0.99061 | 0.98181 | 0.99375 | 0.98759 |

Bold values in Table 7 represent the best result in each indicator.

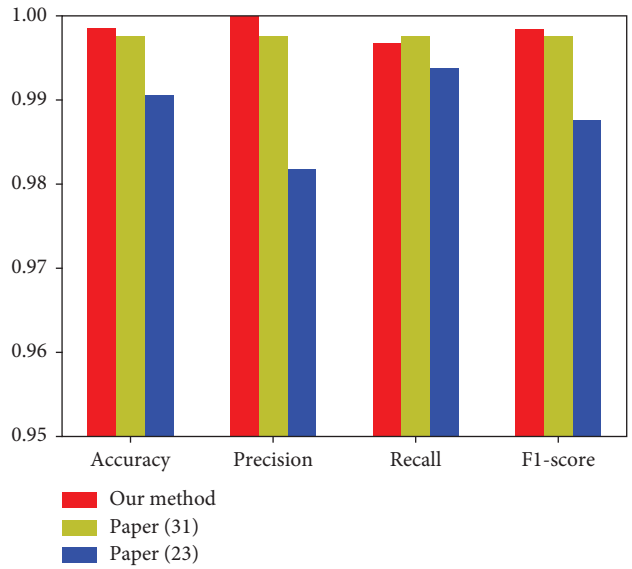


Figure 7: Comparison of different methods.

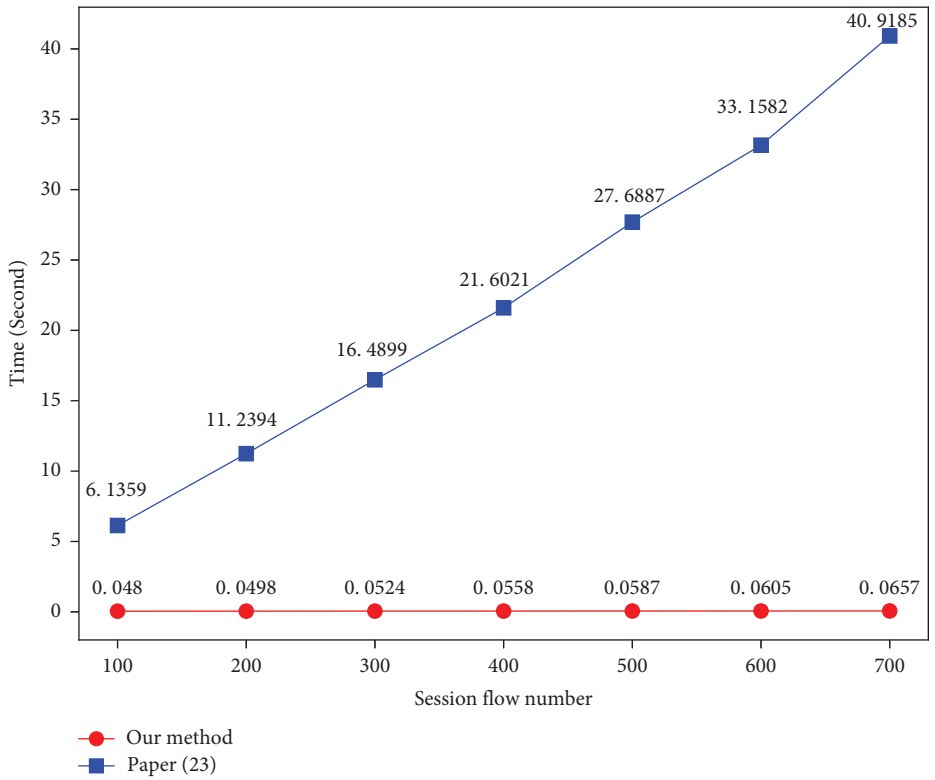


Figure 8: Model training time comparison with deep learning.

Table 8: Results of the data imbalance experiment.

| Abnormal/normal in test set | Accuracy | Precision | Recall | *F*1 score |
|---|---|---|---|---|
| 1 : 1 | 0.9876 | 0.99696 | 0.9788 | 0.98775 |
| 1 : 5 | 0.99104 | 0.99873 | 0.949 | 0.97225 |
| 1 : 10 | 0.99725 | 1.0 | 0.97 | 0.9847 |
| 1 : 50 | 0.99901 | 1.0 | 0.96 | 0.97309 |
| 1 : 100 | 0.999 | 1.0 | 0.9 | 0.9444 |

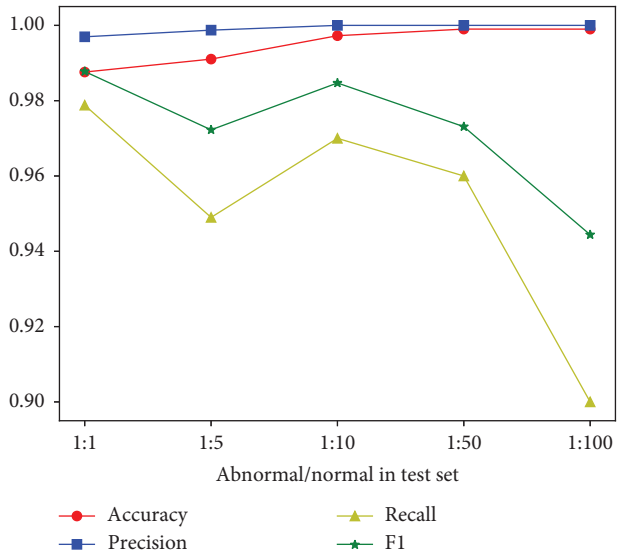


Figure 9: Comparison of data imbalance experiment results.

amount of payload within the flow and fed it into Convolutional Neural Networks for covert communication detection. To make a valid comparison, we used the same data to verify detection indicators of our method and the above two methods. Moreover, our method and the method of paper [31] both used Random Forest algorithm. The results are shown in Table 7 and the comparisons of results are shown in Figure 7. Bold values in Table 7 represent the best result in each indicator.

It can be seen that the detection indicators of our method are slightly higher than the other two methods. However, compared with paper [31], our method extracts fewer features while ensuring a high detection rate, which proves the effectiveness of our proposed features. Moreover, the paper [31] mainly extracts features from four aspects: bytes sent and received, packet length, packet time, and request/response time. These four aspects of information are not directly related to the encoding of cookie. The features extracted by our method are related to the behavior of cookie encoding, so theoretically, our features are more stable when detecting HTTP cookie covert channel, that is to say, when the general flow statistic features change, HTTP cookie covert channel can still transmit information because the cookie does not change, in this situation, our method still works because it does not use general flow statistical features. Paper [23] uses deep learning method for classification, which will result in more time overhead. Figure 8 shows the comparison of the training time for the models to reach convergence between our method and the method of paper [23] for different data sizes. Therefore, our method has better time performance than deep learning methods.

*5.4. Data Imbalance Experiment.* Typically, the covert channel accounts for only a small part of the network traffic. Therefore, to test the practicability of our method, we conducted the data imbalance experiment. We used 500 normal and 500 abnormal data points as the training set. The test set consisted of two parts; the first part was 500 normal data that were different from the training set. In the second part, we selected 500, 100, 50, 10, and 5 abnormal data that were different from the training set to test the classification effect of our method when the ratio of abnormal and normal data were 1 : 1, 1 : 5, 1 : 10, 1 : 50, and 1 : 100, respectively. And we used Random Forest as the classifier. To exclude possible individual errors in a single experiment, we repeated the experiment ten times with different data under each data ratio. Finally, the classification performance was measured based on the average value of each evaluation indicator. The results are shown in Table 8 and the comparisons of results are shown in Figure 9. It can be seen that the four indicators fluctuate within a certain range. In summary, our method

performs well under each data ratio and has certain practicability.

## 6. Conclusion

Aiming at the problem that existing covert channel detection method cannot well detect HTTP cookie covert channels, we propose an HTTP cookie covert channel detection method based on session flow interaction features. We propose to further divide the session flow into smaller subflow, which is defined as the interaction process. And we extract effective features based on the interaction process. Experimental results show that, for HTTP cookie covert channels, the detection rate of our method can reach 99%. We also prove the superiority of our method compared with the existing methods and its practicability in the simulation environment through experiments and analysis. In the future work, we plan to further expand the dataset to optimize the effect and performance of our method in practical application.

## Data Availability

The data used to support the findings of this study are available from the first author upon request.

## Conflicts of Interest

The authors declare that they have no conflicts of interest.

## Acknowledgments

This work was supported by the National Natural Science Foundation of China (no. U19A2081), Fundamental Research Funds for the Central Universities (no. SCU2021D048), and Science and Engineering Connotation Development Project (no. 2020SCUNG129).

## References

[1] B. W. Lampson, "A note on the confinement problem," *Communications of the ACM*, vol. 16, no. 10, pp. 613–615, 1973.

[2] O. Darwish, A. Al-Fuqaha, G. B. Brahim, I. Jenhani, and M. Anan, "Towards a streaming approach to the mitigation of covert timing channels," in *Proceedings of the 2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC)*, pp. 255–260, Limassol, Cyprus, June 2018.

[3] H. Seong, I. Kim, Y. Jeon, M.-K. Oh, S. Lee, and D. Choi, "Practical covert wireless unidirectional communication in IEEE 802.11 environment," *IEEE Internet of Things Journal*, vol. 10, no. 2, pp. 1499–1516, 2023.

[4] Y. Guo, X. Xin, Y. Zhang, and J. Yang, "Leaky way: a conflict-based cache covert channel bypassing set associativity," in *Proceedings of the 2022 55th IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pp. 646–661, Chicago, IL, USA, October 2022.

[5] F. Fu, B. Lu, X. Yan, M. Deng, L. Zhu, and A. Wang, "Low-cost covert wireless communication assisted by optical frequency comb for deep denoising," in *Proceedings of the 2022 IEEE International Topical Meeting on Microwave Photonics (MWP)*, pp. 1–4, Orlando, FL, USA, October 2022.

[6] K. G. Kogos, E. I. Seliverstova, and A. V. Epishkina, "Review of covert channels over HTTP: communication and countermeasures," in *Proceedings of the 2017 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*, pp. 459–462, Moscow, Russia, February 2017.

[7] S. Wendzel, S. Zander, B. Fechner, and C. Herdin, "Pattern-based survey and categorization of network covert channel techniques," *ACM Computing Surveys*, vol. 47, no. 3, p. 26, 2015.

[8] J. Miller-Osborn and J. Grunzweig, "Menupass returns withnew malware and new attacks against japanese academics and organizations," 2017, https://unit42.paloaltonetworks.com/unit42-menupass-returns-new-malware-new-attacks-japanese-academics-organizations/.

[9] Z. Hromcova, "Okrum and ketrican: an overview of recent ke3chang group activity," 2019, https://www.welivesecurity.com/wp-content/uploads/2019/07/ESET_Okrum_and_Ketrican.pdf.

[10] R. Nafisi and A. Lelli, "GoldMax, GoldFinder, and sibot: analyzing NOBELIUM's layered persistence," 2021, https://www.microsoft.com/en-us/security/blog/2021/03/04/goldmax-goldfinder-sibot-analyzing-nobelium-malware/.

[11] F. Liu, D. Li, and Y. Zhao, "The covert communication detection model based on key field of header in HTTP protocol," *Fire Control and Command Control*, vol. 43, no. 11, pp. 40–45, 2018.

[12] G. Shen, J. Zhai, and Y. Dai, "HTTP parameter sorting covert channel detection method based on markov model," *Computer Engineering*, vol. 46, no. 2, pp. 154–158, 2020.

[13] J. Wu, Z. Yang, and W. Liu, "Multiscale feature fusion for malicious HTTP request detection," *Application Research of Computers*, vol. 38, no. 03, pp. 871–874, 2021.

[14] O. Darwish, A. Al-Fuqaha, G. Ben Brahim, I. Jenhani, and A. Vasilakos, "Using hierarchical statistical analysis and deep neural networks to detect covert timing channels," *Applied Soft Computing*, vol. 82, 2019.

[15] S. Al-Eidi, O. Darwish, and Y. Chen, "Covert timing channel analysis either as cyber attacks or confidential applications," *Sensors*, vol. 20, no. 8, 2020.

[16] S. Al-Eidi, O. Darwish, Y. Chen, and G. Husari, "SnapCatch: automatic detection of covert timing channels using image processing and machine learning," *IEEE Access*, vol. 9, pp. 177–191, 2021.

[17] Y. F. Wang, Y. L. Yang, and M. L. Rao, "Research of http tunnel detecting technique based on c4.5," *Computer Engineering and Design*, vol. 33, no. 2, pp. 493–497, 2012.

[18] L. I. Wei, L. Li, L. I. Jia, and S. Lin, "Characteristics analysis of traffic behavior of remote access trojan in three communication phases," *Netinfo Security*, vol. 5, pp. 10–15, 2015.

[19] X. Chen, J. Chen, and G. Shao, "A covert communication behavior detection method based on session flow aggregation," *Journal of University of Electronic Science and Technology of China*, vol. 48, no. 3, pp. 388–396, 2019.

[20] H. Li and D. Chasaki, "Network-based machine learning detection of covert channel attacks on cyber-physical systems," in *Proceedings of the 2022 IEEE 20th International Conference on Industrial Informatics (INDIN)*, pp. 195–201, Perth, Australia, July 2022.

[21] W. Wang, M. Zhu, X. Zeng, X. Ye, and Y. Sheng, "Malware traffic classification using convolutional neural network for representation learning," in *Proceedings of the 2017 International Conference on Information Networking (ICOIN)*, pp. 712–717, Da Nang, Vietnam, January 2017.

[22] S. Z. Lin, Y. Shi, and Z. Xue, "Character-level intrusion detection based on convolutional neural networks," in *Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, Rio de Janeiro, Brazil, July 2018.

[23] G. Mar'in and P. Casas, "Deepmal - deep learning models for malware traffic detection and classification," 2020, https://arxiv.org/abs/2003.04079.

[24] A. Barth, "HTTP state management mechanism," 2011, https://www.rfc-editor.org/rfc/rfc6265.

[25] Y. Tang, Y. Chen, and D. Zhou, "Measuring uncertainty in the negation evidence for multi-source information fusion," *Entropy*, vol. 24, p. 1596, 2022.

[26] Y. Tang, S. Tan, and D. Zhou, "An improved failure mode and effects analysis method using belief jensen–shannon divergence and entropy measure in the evidence theory," *Arabian Journal for Science and Engineering*, vol. 48, 2022.

[27] J. W. Ratcliff and D. Metzener, "Pattern matching: the gestalt approach," *Dr. Dobb's Journal*, vol. 46, 1988.

[28] S. Ali, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Computers and Security*, vol. 31, no. 3, pp. 357–374, 2012.

[29] Stratosphere, "Stratosphere laboratory datasets," 2015, https://www.stratosphereips.org/datasets-overview.

[30] Z. Aouini and P. Adrian, "Nfstream:A flexible network data analysis framewok," *Computer Networks*, vol. 204, Article ID 108719, 2022.

[31] M. Montazerishatoori, "Detection of DoH tunnels using time-series classification of encrypted traffic," in *Proceedings of the 2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech) IEEE*, Calgary, AB, Canada, August 2020.