CSP-Based General Detection Model of Network Covert Storage Channels

Hui Zhu, Tingting Liu, Guanghui Wei, Beishui Liu, Hui Li

HAL Id: hal-01480205

https://hal.inria.fr/hal-01480205

Submitted on 1 Mar 2017

CSP-based General Detection Model of Network Covert Storage Channels

Hui Zhu[1,2], Tingting Liu[1], Guanghui Wei[1], Beishui Liu[1], and Hui Li[1]

[1] State Key Laboratory of Integrated Service Networks, Xidian University, Xi'an, China
xdzhuhui@gmail.com
[2] Network and Data Security Key Laboratory of Sichuan Province

Abstract. A network covert channel is a malicious conversation mechanism, which brings serious security threat to security-sensitive systems and is usually difficult to be detected. Data are hidden in the header fields of protocols in network covert storage channels. In this paper, a general detection model based on formal protocol analysis for identifying possible header fields in network protocols that may be used as covert storage channels is proposed. The protocol is modeled utilizing the Communication Sequential Processes (CSP), in which a modified property of header fields is defined and the header fields are classified into three types in accordance to the extent to which their content can be altered without impairing the communication. At last, verification of the model in Transmission Control Protocol (TCP) shows that the proposed method is effective and feasible.

Keywords: Security modeling, Protocol analysis, Network covert storage channels, Detection, CSP.

1 Introduction & Related Work

A network covert channel is a malicious communication mechanism which can be utilized by attackers to convey information covertly in a manner that violates the system's security policy [1]. The channel is usually difficult to detect and brings serious security threat to security-sensitive systems. Consequently, there is an increasing concern on network covert channels.

There are two types of network covert channels: storage and timing channels. With the widespread diffusion of networks, many methods have been studied by attackers for constructing network covert channels using a variety of protocols including TCP, IP, HTTP and ICMP [2-4]. For example, Abad proposed [5] an IP checksum covert channels which use the hash collision. Fisk [6] proposed to use the RST flag in TCP and the payload of ICMP protocols to transfer covert message. These covert channel implementations are based on common network or application layer internet protocols. Castiglione et al presented an asynchronous covert channel scheme through using spam e-mails [7]. Moreover, Fiore et al intro-

duced a framework named Selective Redundancy Removal (SRR) for hiding data [8]. It is easy to see that the network covert channels are all based on the various protocols.

More attention has been placed on network covert channels detection. Tumoian et al used the neural network to detect passive covert channels in TCP/IP traffic [9]. Zhai et al [10] proposed a covert channel detection method based on the TCP Markov model for different application scenarios. Gianvecchio et al [11] proposed an entropy-based approach to detect various covert timing channels. The above methods are based either on the anomaly data or on unusual traffic patterns in practical network traffic. It induces that they are hard to find those potential and unknown covert channel vulnerabilities. In this paper, we establish a CSP-based general model to analyze and detect the potential exploits of protocols from the perspective of original design of protocols.

The remainder of the paper is organized as follows. Section 2 introduces the basic concepts of CSP. In Section 3, we give the CSP-based general detection model including the details of establishing and detection steps. In section 4, we test our model in Transmission Control Protocol (TCP). The conclusions are in section 5.

2   CSP (Communicating Sequential Processes)

In CSP [12-15], systems are described in terms of processes which are composed of instantaneous and atomic discrete events. The relations between processes and operations on processes are formalized with operational semantics of the algebra. Many operations on processes and their interrelationships are defined within algebraic semantics of CSP as following:

- $a \to P$(Prefixing): The process will communicate the event a and then behave as process P.
- $P \square Q$ (Deterministic Choice): This process can behave either as P or Q, but the environment decides on which process to run.
- $a: A \to P(a)$ (Prefix Choice): This represents a deterministic choice between the events of the set A which may be finite or infinite. This notation allows representing input and output from channels.
- $c?x: A \to P(x)$: The input can accept any input x of type A along channel c, following which it behaves as $P(x)$.
- $c!v \to P$: The output $c!v \to P$ is initially able to perform only the output of v on channel c, then it behaves as P.
- $P \underset{A}{\parallel} Q$ *(Parallel Composition):* Let A be a set of events, then the process behaves as P and Q acting concurrently, with synchronizing on any event in the synchronization set A. Events not in A may be performed by either of the pro-

cesses independent of the other.

- **Definition 1: trace:** The trace is a model which characterizes a process P by its set traces (P): finite sequences of events which it can perform in a certain period of time.
- **Definition 2: refinement:** A relationship between two CSP processes: trace model refinement. A process P is trace-refined by a process Q, that is $P \sqsubseteq_T Q$, and only if traces (Q) $\subseteq$ traces (P).

3   The CSP-based general detection model
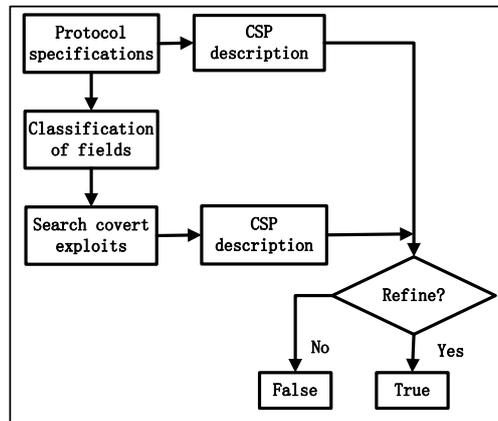
3.1   The general model framework

Fig.1. The CSP-based general model framework

We propose a general detection model focus on the original design details of protocols. The model is used to analyze and detect the covert storage channels vulnerabilities in various layers of protocols. The CSP framework diagram is shown in Figure 1, the model framework includes 5 steps as follows.

Step1: The original design specifications of protocols and the communication procedure of the protocol interacting entity are analyzed, and then a CSP-based process is established.

Step2: The header fields of protocols are classified into three types--Secure fields, exploited fields-I, exploited fields-II. The details of the classification can be obtained in section 3.2.

Step3: Based on the classification of header fields and the status of the protocol interaction system, a CSP-based search process is established to search for the covert storage channels exploits in the header fields of protocols.

Step4: Based on the hypothesis of the network covert storage channel and vulnerability which have been found in Step3, a CSP-based process of network

covert storage channels is established.

Step5: The traces (**Definition1**) of the established processes in Step1 and Step4 are detected at last. It is necessary to find whether the two traces can satisfy the refinement (**Definition2**) relationship. If the refinement (**Definition2**) relationship can be satisfied by the traces, then there are covert storage channels vulnerabilities in the header fields of protocols, vice verse.

The CSP-based general detection model reduces the existence problem of the covert channels exploits in protocol to the question whether the CSP description of covert storage channels is a refinement of the protocol specifications, which simply the detection of the network covert storage channels.

3.2   Classification of header fields

A property named modified property is defined for every header field. The header fields of protocol can be classified into three types according to the modified property as follows:

- **Secure fields**: This type of fields cannot be modified arbitrarily due to their modifications will impair the normal communications. So these fields are secure. For example, the source port field and the destination field of TCP header are secure fields. Once they are modified, the TCP connection cannot be set up.
- **Exploited fields-I:** These fields can be modified arbitrarily with its own merits of making no sense on the normal communication. Such as the reserved fields of TCP headers which are designed for future protocol improvements and the optional header fields of IP.
- **Exploited fields-II:** These fields are needed to guarantee the normal communication under some conditions, so the modifications of them have some restrictions. For example, the TCP urgent point field can be modified to convey messages when the urgent flag of the control bits field is not set.

4   The verification of model

4.1   The CSP description of TCP connection

In this section, we test and verify the effectiveness and feasibility of the general model in TCP. In order to simplify the analysis and description of TCP, we assume that our TCP interaction system consists of two hosts A and B which are only equipped with an application layer, a TCP entity and two channels between the layers as shown in Figure 2.
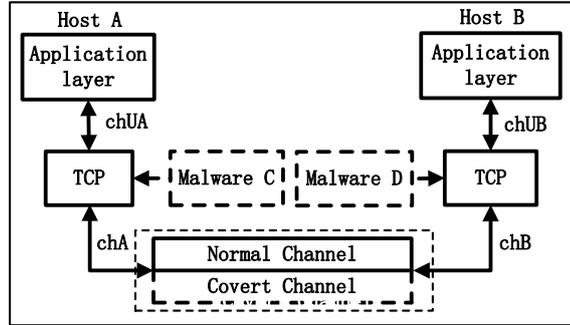
Fig.2. TCP protocol interaction system

The model of TCP describes the connection between a client and a server, and the TCP protocol state machine has six states. We regard host A as a client, and B as a server. Let *Tstate* indicates the set of states. We assume there is a set of different packet types named *PacketTypes*.

Tstate= {CLOSED, LISTEN, SYN-SENT, SYN-RECEIVED, ESTABLISHED, FIN_WAIT1}

PacketTypes={packet.{syn},packet.{syn_ack},packet.{ack},packet.{rst}, packet.{fin}}

```
Host A:                               Host B:
TCP_A_SM=                             TCP_B_SM=
  Tstate==CLOSED  &                     Tstate==LISTEN  &
  (chUA!!command→                       (chB??packet→
  if(command==connect.B)                （if(packet.SYN==1)
then (chA!!packet.{syn}→              then (chB!!packet.{syn_ack}→
Tstate:=SYN_SENT))                    Tstate:=SYN_RCVD)
else（chUA!!Error）→TCP_A_SM           else(discard.packet)) →TCP_B_SM
  □Tstate==SYN_SENT  &                  □Tstate==SYN_RCVD  &
  (chA??packet→                         (chB?? packet→
  If((packet.SYN==1) ∧（packet.ACK==1   if(packet.SYN==1) ∧（packet.ACK==1）
  ∧（packet. ackNo==seqNo+1））         ∧（packet.ackNo==seqNo+1）
then (chA!!packet.{ack}→              then(Tstate:=ESTABLISHED)
Tstate:=ESTABLISHED)) → TCP_A_SM     else (discard.msg) →TCP_B_SM
  □Tstate==ESTABLISHED  &               □Tstate==ESTABLISHED  &
  (chUA??packet→(if(packet.FIN==1)      (chuB??packet→(if(packet.FIN==1)
then (Tstate:=FIN_WAIT1;chA!!packet)  then (Tstate:=FIN_WAIT1;
else (chA!!packet) ) →TCP_A_SM        chB!!packet)
|(chA??packet→(if(packet.FIN==1)      else (chB!!packet) ) →TCP_B_SM
  then (Tstate:=FIN_WAIT2;chUA!!packet) □Tstate==FIN_WAIT1  &
else (chUA!!packet) →TCP_A_SM         (chB??packet→(if(packet.FIN==1)
  □Tstate==FIN_WAIT1  &                then(Tstate:=FIN_WAIT2)→TCP_B_SM
  (chA??packet→(if(packet.FIN==1)      □Tstate==FIN_WAIT2   &
then(Tstate:=FIN_WAIT2)→TCP_A_SM       (wait for close)
  □Tstate==FIN_WAINT2   &             →TCP_B_SM
  (wait for close)→TCP_A_SM
```
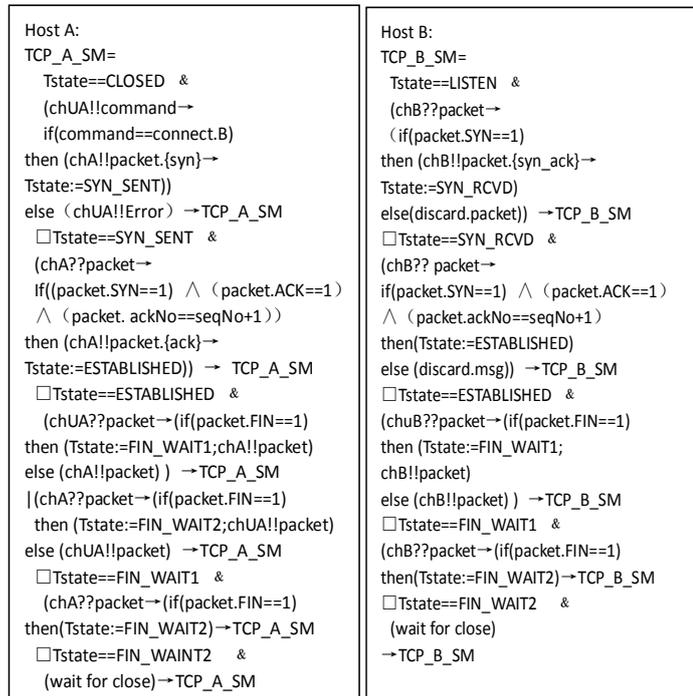
Fig.3. The CSP descriptions of TCP in Host A and Host B

Two CSP descriptions for Host A and Host B are shown in Figure 3. The CSP model of the TCP connection is composed of TCP_A_SM and TCP_B_SM.

$$TCP\_CSP=TCP\_A\_SM \mid\mid\mid TCP\_B\_SM$$

4.2   Classification of header fields in TCP

According to the rules mentioned in section 3.2, the header fields of TCP are classified into three types as shown in Table 1.

Table 1. The classification of TCP header

type	number	header field		value
Secure fields	1	Source port		0
	2	Destination port		0
	5	TCP header length		0
	7	URG		0
	8	ACK		0
	9	PSH		0
	10	RST		0
	11	SYN		0
	12	FIN		0
	13	Window size		0
exploited fields-I	6	Reserve field(6 bit)		1
exploited fields-II	3	SeqNo	SYN==1,ACK==0	V=1
	4	AckNo	SYN==1,ACK==0	V=1
			ACK==1	V=0
	14	checksum		V=1
	15	Urg_p	URG==0	V=1
			URG==1	V=0

The process C_exploit(X) search for the covert storage channels exploits in TCP according to the rules based on the modified property and the specifications of TCP connection. Figure 4 shows the CSP process C_exploit(X). After searching for the network covert storage channel exploits through the process C_exlpoit(X), we have obtained three results based on the different states of TCP connection.

- When the TCP connection is beginning to establish (Tstate=CLOSED):
  C_exploit(X)={seqNo, ackNo, Reserve_field, checksum, Urg_p}
- When the TCP entities are not in closed state and the urgent flag of the control bits is not used:
  C_exploit(X)={Reserve_field, checksum, Urg_p}
- When the TCP entities are not in closed state and the urgent flag of the control bits is set:
  C_ exploit(X)={Reserve_field,checksum}

```
C_exploit (X) = learn? Packet:Packet.i→C_ exploit (Judge(X∪{i}))
        □ (if X=∅ then overt→C_ exploit (X)
                         else  covert→C_ exploit (X))
Subprocess Judge (X∪{i} is as follow:
Judge (X∪{i})=(if(Tstate==CLOSED)∧(cmd.connect.B)→add.seqNo);
            (if(Tstate==SYN-SENT)∧(SYN==1)∧(ACK==0) →add.ackNo);
            (if(Tstate≠CLOSED) →add.Reserve_field);
            (if(packet. Window_size≠null→add.checksum);
            (if(packet.URG==0)→add.Urg  p)
```

Fig.4. The CSP process C_exploit(X)

4.3   The CSP description of covert storage channels

As have been shown in Figure 2, there are two malwares C and D who hide inside the TCP interaction system.
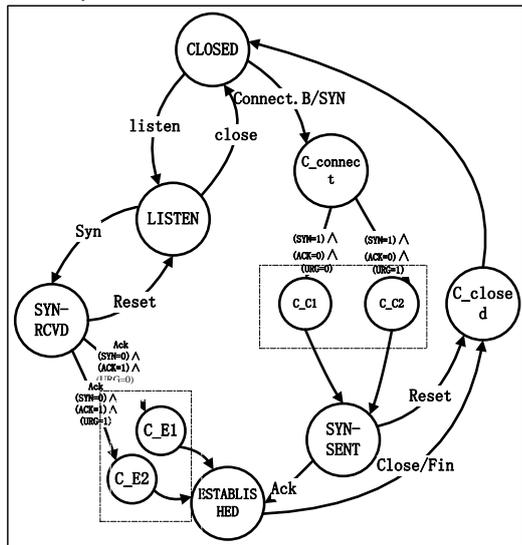

Fig.5. The finite state diagram of TCP interaction system

Assume that the malwares C and D set up a covert storage channel, of which C is the sender of the covert channel, D is the receiver. Let the set **Cstate** indicate the states of covert storage channels. The set *IS_state* indicates the global states of TCP interaction system which include the states of covert channel. Then, *IS_state=Tstate ∪Cstate*. Figure 5 depicts the state transition diagram of the whole TCP interaction system.

Cstate= {C_closed, C_listen, C_connect, C_c1, C_c2, C_e1, C_e2}

The malwares monitor the status of TCP connection and utilizes the ex-

ploited fields of TCP to transmit the covert message. The CSP process of malware C is shown in Figure 6. The CSP description of malware D is similar to the malware C, so we omit its description here. The process IS_CSP indicates the interaction system of TCP protocol entities and two malwares.

```
C_Channel=(listen??chUA)→
            (if(command==connect.B)∧(Tstate==CLOSED)
              then(Cstate:=C_connect))→C_Channel_modu(Cstate)
              else if(Tstate==SYN_SENT)then(Cstate:=C_established)
              else if(Tstate==ESTABLISHED)then(Cstate:=C_established)
              else if(Tstate==FIN_WAIT1)then(Cstate:=C_closed)
Subprocess C_Channel_modu(Cstate) is as follow:
C_Channel_modu(Cstate)=
        Cstate=C_connect  &
        (if(SYN==1)∧(ACK==0)∧(URG==0)
        then(Cstate:=C_c1;modu(packet.{syn})→chA!!packet. {syn} .(covert_msg))
        else if((SYN==1)  ∧(ACK==0)∧(URG==1)
        then(Cstate:=C_c2;modu(packet.{syn})→chA!!packet. {syn} .(covert_msg)) )
□Cstate=C_ established  &
   (if(URG==0)
   then (Cstate:=C_e1;modu(packet.{syn_ack})→chA!!packet.{syn_ack}.(covert_msg))
   else (Cstate:=C_e2;modu(packet.{syn_ack})→chA!!packet.{syn_ack}.(covert_msg))
```

Fig.6. The CSP process of malware C

The description of network covert storage channels is
Covert_channel= C_Channel |||D_Channel
Then, the following IS_CSP is obtained.
IS_CSP=TCP_CSP |||Covert_channel
          =TCP_A_SM |||TCP_B_SM||| C_Channel |||D_Channel

4.4   Analysis of the verification

In this section, we analyze and verify the existence of network covert storage channels under the normal communication of TCP using trace model refinement (**Definition 2**). It means that whether the traces of IS_CSP are subsets of traces of TCP_CSP which allow precisely the valid traces.

```
tr(TCP_CSP)=
<cmd.connect.B,send.A.B.packet.{syn},receive.B.A.packet.{syn_ack},send.A.B.packet.{ack},receive.B.A.packet…..>
tr(IS_CSP)=
<cmd.connect.B,send.A.B.packet.{syn}.(seqNo=covert_msg),receive.B.A.packet.{syn_ack}.(urgent_pointer=covert_msg),…>
```

Fig.7. An example of traces of TCP_CSP and IS_CSP

The CSP model can detect whether the network covert storage channels

are possible under the normal communication of TCP. Figure 7 shows an example of traces of TCP_CSP and IS_CSP. As we can see that tr(IS_CSP) is a subset of tr(TCP_CSP):

$$tr(IS\_CSP) \subseteq tr(TCP\_CSP)$$

From the trace of IS_CSP, we can see that the malware C modulates the initial sequence number to convey the covert message which has been utilized and found by the previous researchers. For example, Rutkowska [16] implemented a network storage channel utilizing the initial sequence number named NUSHU. On basis of the definition 2, we come to the conclusion that the CSP model IS_CSP refines TCP_CSP.

$$TCP\_CSP \sqsubseteq_T IS\_CSP$$

tr(TCP_CSP)=
<cmd.connect.B,send.A.B.packet.{syn},receive.B.A.packet.{syn_ack},send.A.B.packet.{ack},.....>
tr(IS_CSP)=
<cmd.connect.B,send.A.B.packet.{syn}.(reserved_fields=covert_msg),receive.B.A.packet.{syn_ack}.(reserved_fields=covert_msg),....>

Fig.8. Another example of traces of TCP_CSP and IS_CSP

Figure 8 depicts other similar traces as well. The traces show us that the malwares C and D utilize the reversed field to convey covert message. This covert storage channel have been found and studied by Handel [17].

The verification of TCP proves that the CSP-based general model might yield several very similar covert storage channels in other network protocols in terms of the modified property of header fields and the specifications of network protocols.

5 Conclusion

In this paper, we propose a CSP-based general detection model for analyzing and detecting the network covert storage channels in network protocols. In our model, we describe the protocol interaction system based on the original design specifications of protocols. Besides, we define a modified property for every header field, and classify the header fields into three types based on this property. We establish a search process for searching the potential covert exploits in the header fields of protocols. Then we establish a network covert storage channel based on the hypothesis of network covert storage channels, and verify the covert channels based on trace refinement. Finally, the model of this CSP formal method is illustrated and verified in TCP. The result of the verification shows that the general model is effective and feasible in finding the covert storage channels.

The CSP-based general detection model is modular, so it can be easily extended to describe the other network protocols and detect the covert channels hidden in them. In the future, we will try to establish a formalized method for detecting and analyzing the covert timing channels.

References

1. Snoeren, A., Partridge, C., Sanchez, L.: Single Packet IP Trace back. ACM/IEEE Transaction on networking. 10(6):721-734(2002)
2. Zander, S., Armitage, G., Branch, P.: A Survey of Covert Channels and Countermeasures in Computer Network Protocols. IEEE Communications Surveys and Tutorials, 9(3):44-57(2007)
3. Ahsan, K., Kundur, D.:Practical Data Hiding in TCP/IP. ACM WKSP Multimedia, pp. 7-14, (2002)
4. Cauich, E., Gardenas, R. G., Watanabe R.: Data Hiding in Identification and Offset IP Fields. In: 5th International Symposium (2005)
5. Abad, C.: IP checksum covert channels and selected hash collision. Technical report,(2001)
6. Fisk, G., Fisk, M., Papadopoulos, C., Neil, J.: Eliminating steganography in Internet traffic with active Wardens. In: 5th Internetional Workshop on Information Hiding, pp.18-35(2002)
7. Castiglione, A., Santis, A.D., Fiore, U., Palmieri, F.: An asynchronous covert channel using spam. Computers and Mathematics with Applications 63(2): 437-447(2012)
8. Fiore, U.: Selective Redundancy Removal: A Framework for Data Hiding. Future Internet 2(1): 30-40 (2010)
9. Tumoian,E., Anikeev, M.: Network based detection of passive covert channels in TCP/IP. In: Proceedings of the IEEE Conference on Local Computer Networks 30th Anniversary.802-809(2005)
10. Zhai,J., Liu,G., Dai,Y.: A covert channel detection algorithm based on TCP Markov model. In: Proceedings of Second International Conference on Multimedia Information Networking and Security,pp.893-897(2010)
11. Gianvecchio, S., Wang, H.: An Entropy-Based Approach to Detecting Covert Timing Channels. IEEE Transactions on dependentable and secure computing. 8(6), 785-797 (2011).
12. Hoare, C.A.R.: Communicating Sequential Processes. ACM, Communications of the ACM,pp.666-677(1978)
13. Brookes, S.D., Hoare, C.A.R., Roscoe,A.W.: A theory of Communicating Sequential Processes. Journal of the ACM,Vol.31,pp.560-599(1984)
14. Roscoe, A.W.: The theory and practice of concurrency. s. l.: Prentice Hall(1998)
15. Schneider, S. A.: Concurrent and real-time systems: the CSP approach. s. l . : John Wiley(1999)
16. Rutkowska,J.: The implementation of passive covert channels in the Linux kernel. Available: http://invisiblethings.org/papers.html.
17. Handel,T.G., Sandford,M.T.:Hiding data in the OSI network model. Lecture Notes in Computer Science.1174,23-38(1996)