Reproducing Article: "Detecing TCP ACK storm attack: a state transition modelling approach"

KAR CHUN TEONG, HONG XU LI, TIAN YI HU, ZHONG YU HUANG

Abstract
In this project, the authors reproduce the experiment of an article named "Detecing TCP ACK storm attack: a state transition modelling approach". The main work includes the implementation of state transition model, capture and analysis of TCP packets, simulation of ACK storm on a local machine, and simulation of the whole process of using the state transition model to detect the locally generated ACK storm.

Keywords
TCP, ACK storm, State Transition Model

Contents

Introduction

0.1 ACK storm attacks

ACK storm attacks are injection attacks against an Transmission Control Protocol (TCP) connection. This attack requires sending two packets by the adversary with acknowledgement number greater than the sequence number used in each direction and the two end hosts will attempt to resynchronise the sequence numbers by sending duplicate acknowledgement and enter a loop, thus rendering the connection useless, the process is shown in Figure 1.

0.2 Transmission Control Protocol

TCP is the majorly used transport layer protocol in today's Internet, it is vulnerable to trivial man-in-the-middle (MitM) attacks. This design flaw makes it vulnerable to ACK storm attacks. As many applications use TCP as transport layer protocol, these design flaws of TCP can virtually impact any application used in the Internet today.

0.3 State Transition Model

To detect the ACK storm attack in TCP, the authors proposed a state transition model, the state transition model is defined as shown in Figure 2. The states included:
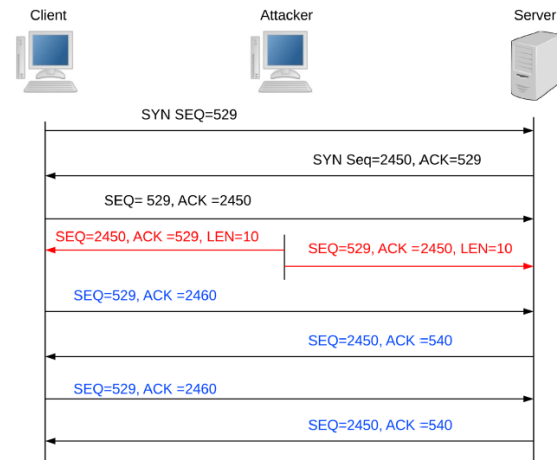

Figure 1. Simple ACK Storm Attack

1. q0: there are no unacknowledged bytes in the TCP connection

2. q1: there are bytes to be acknowledged in the TCP connection

3. q2: abnormal acknowledgement packet received

4. Storm: acknowledgement storm detected

The transition table for each states is shown in Figure 3, the variable c shown in both figures indicates the counter of instances of $ACK > SEQ$, as the counter surpasses the preset threshold α, the state transition model deducts that an ACK storm occurs.
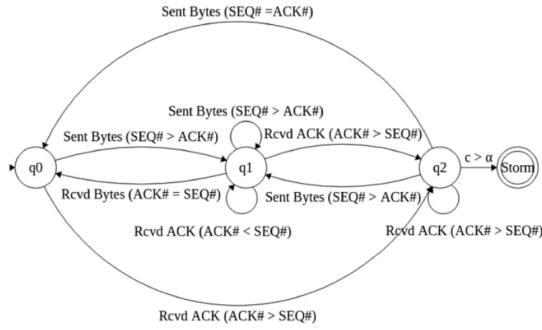
1. Implementation

Figure 2. State Transition Model

Figure 3. Transition Table

1.1 Capture and Analysis of TCP Packets

We use libpcap library to capture, filter, and analyse the TCP packets, the capture and analysis result of a TCP packet is shown in Figure 4. To achieve accurate analysis, we need to fully grasp the structure of TCP header, which is shown in Figure 5.
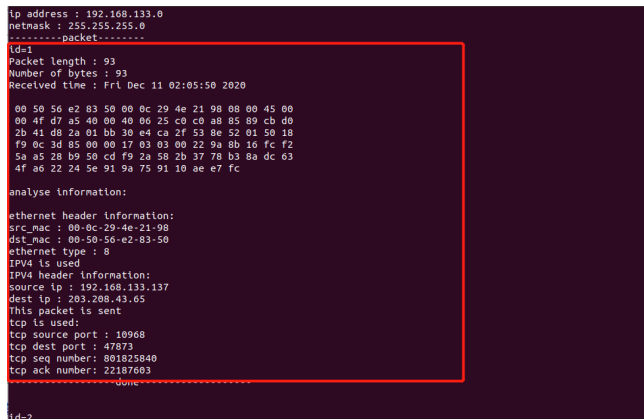

Figure 4. Analysis of TCP packet

Figure 5. TCP header

In the process of capture and analysis of TCP packets, one of the problems we faced is the connection might be communicating with several different IP addresses at the same time, causing the TCP packets received harder to be organized. The first challenge emerged from this problem is how to identify the different packets' destinations, this can be solved relatively easily, which is by checking the source IP address of the packets.

However, as we need to record every received packet corresponding to sent packet of specific IP address to keep track of the state of connection, the other problem emerged is how to specifically locate the corresponding packets within the captured unorganized packets.

1.2 State Transition Model

To solve the previously mentioned problem and to implement the state transition model, we use a data structure named connect to store all necessary information, the structure is shown in Figure 6.

Figure 6. Data Structure

The uses of each data field are listed below:

- status: corresponds to states q0, q1, q2 and storm.

- to_q2: corresponds to the counter of instances of $ACK > SEQ$.

- ip[4]: corresponds to connecting IP addresses.

- send_packet_ack: corresponds to ACK number of sent packet.

- send_packet_seq: corresponds to SEQ number of sent packet.

- rcvd_packet_ack: corresponds to ACK number of sent packet.

- rcvd_packet_ack: corresponds to ACK number of received packet.

- rcvd_packet_seq: corresponds to SEQ number of received packet.

We create an array using this data structure, each element in the array corresponds to one connection with an IP address, and we initialize each element as shown in Figure 7.


```
193        connection[i].status = 0;
194        connection[i].to_q2 = 0;
195        connection[i].ip[0] = -1;
196        connection[i].send_packet_ack = -1;
197        connection[i].send_packet_seq = -1;
198        connection[i].rcvd_packet_ack = -1;
199        connection[i].rcvd_packet_seq = -1;
```

Figure 7. Initialization

As we initialized the array, we do the following steps for every packets the connection send and receive:

1. Check IP address

   1.1. Check ip[4] to see if the IP address is stored.

   1.2. If not stored: create a new element to store it.

   1.3. If stored: go to step 2.

2. Renew element's value

   2.1. Determine if the packet is sent or received.

   2.2. Sent packet: renew send_packet_ack and send_packet_seq with the value of ACK and SEQ.

   2.3. Received packet: renew rcvd_packet_ack and rcvd_packet_seq with the value of ACK and SEQ.

3. Renew status

   3.1. Determine if the corresponding value is renewed (for sent packet, check rcvd_packet_ack and rcvd_packet_seq; for received packet, check send_packet_ack and send_packet_seq)

   3.2. If value renewed: renew the value of status according to the transition rules.

   3.3. If value remained: skip.

At this point, the state transition model is successfully implemented with the ability to analyse any sent and received TCP packets and record the status of the connection according to the transition rules.

1.3 ACK Storm Attack
After implementing the state transition model, we need to implement the ACK storm attack to test the effectiveness of our model. The implementation steps are shown below:

1. Use nc command to create TCP connection.

   1.1. Server: nc -p 1080.

   1.2. Client: nc -p 9090 192.168.213.129 8080

2. Use wireshark to capture TCP packets, gain the ACK and SEQ number.

3. Third party uses scapy to falsify information.

The result of the ACK storm attack is shown in Figure 8.

Figure 8. ACK storm

2. Contributions

- KAR CHUN TEONG: Researching and selecting article as project topic, designing and creating PPT for mid-term and final presentation, writing final report.

- HONG XU LI: Implementing state transition model.

- TIAN YI HU: Testing and debugging.

- ZHONG YU HUANG: Implementing ACK storm attack.