Fredrik Nordlund <fnordl@kth.se> Prakash Rajagopalan<prakashr@kth.se>

IK2213 Bouncer Report

Introduction

Even when the privacy in the internet-world is protected using encryption, there is no way to assure your anonymity unless your IP is also hidden. Not only hidden at the program level (which always is the case), but also at the network level, so no network analysis tool can unveil it.A bouncer is a program that "bounces" connections from one machine to another. Similarly to proxies, you always connect to the machine where the bouncer resides and this machine, in turn, connects to any other user's computer.

The goal of this small project has been to get a greater understanding of how to implement one such bouncer application using C programming language .The goal is also to get a deeper understanding of advance socket implementation and its usage,TCP/IP packet processing and TCP , ICMP and IP checksums

Problem

The problem of this project was, how can you use RAW socket and PCAP library to capture the necessary IP packets at network interface level and send to the designated server. Several problems that had to be solved for example,how do you store the IP , TCP and ICMP information in the bouncer , so that when server responds to the bouncer , the bouncer has to send back the response to the client. Another problem was how to extract the ICMP and TCP payload from the IP packet ,reassign the source and destination IP address and ports ,recalculate the ICMP , TCP and IP payloads.

We are targeting the medium level requirement. A working "ping" bouncer (ICMP Echo Request/Reply) and a TCP bouncer.

Solution

The bouncer is designed as two sub systems . The first one is a ICMP/IP subsystem and the second a TCP/IP subsystem.The functionalities are as follows.

Both the subsystem are implemented using pcap library to capture the packets at the network interface level in order to receive the packet address to the bouncer and RAW sockets to send the packet to the remote destination .

In case of ICMP/IP ping , the required IP packets are filtered and IP packet header are correctly validated, to make sure that only correct IP headers are bounced. All the IP header fields are be validated according to the RFCs. Also , the ICMP payload is extracted from the packet and ICMP headers are correctly validated. Then , the source and destination fields in the IP headers are reassigned to reach the remote destination . The ICMP and IP checksum are recalculated and packet is sent to the remote destination.

.In case of TCP/IP , the required IP packets are filtered and IP packet header are correctly validated, to make sure that only correct IP headers are bounced and the TCP payload is extracted from the packet and TCP headers are correctly validated. Then , the source and destination fields in the IP headers are reassigned to reach the remote destination . The TCP and IP checksum are recalculated and

packet is sent to the remote destination.

In order to handle multiple client sending ICMP/IP or TCP/IP request , we have used a linked list to store the client details such as IP address , ICMP identifier and TCP ports. When the server responds to the bouncer , bouncer check in the linked list for the appropriate client and forward the response back to the client using RAW socket .

Link to SVN Repository

https://svn.xen.ssvl.kth.se/ik2213/fnordl/