CENG519: Network Security

Term Project Phase 2

Tan Çağatay Acar

April 13, 2025

1 Covert Channel Choice

I've chosen the topic as Echo Request/Reply Payloads: Embedding data directly in ICMP echo request/reply packets. ICMP echo request/reply packets have a payload field that can be set arbitrarily. Its content and size are not defined, so we can utilize them as we see fit.

2 ICMP Covert Channel

2.1 Examples From Real World

This idea is definitely not new and probably is still widely used for establishing covert channels. A sample program that uses ICMP packets for covert chatting is called ICMP-Chat. It features bidirectional chatting with automatic key exchange and encryption. One downfall of this program is it only uses reply messages. This renders the program unusable for today's internet as modern firewalls only allow reply packets if a matching request message was sent before.

Ping Tunnel is a tool for tunneling TCP traffic with ICMP. Ping Tunnel strictly uses ICMP echo request and reply packets. It directly encapsulates them without encrypting them (might be redundant if TLS is used). However, this results in abnormally large packets that can be detected easily with the right ruleset.

3 Implementation

To mimic nominal ICMP traffic, we should always work with matching echo requests and reply messages. While this is an overhead, we can use the reply messages as acknowledgments and make our communication robust. If the message reply has not been received for 700 ms, the message is resent.

Another issue is ICMP echo traffic shape is periodic. So we need to send data at some period defined by a parameter we call as the interval. This sets our message period and we always send messages at this rate.

Message size is also critical as ICMP echo messages are usually used with payloads of 32 (Windows) or 56 (Linux) bytes. This heavily limits our achievable throughput. We introduce another configuration parameter size to denote our message length in bytes.

We also need to be careful of deep packet inspection. The messages we send will not be same as the default payload content sent from ping utilities. Therefore, we cannot conform to the existing payloads content-wise. The best we can do is to encrypt our message with AES and different IVs for each message so that we eliminate patterns in the payload that could be detected.

To run the setup, you need to run python processor, then run covert_receiver.py in insec, and then run covert_sender.py in sec. Please make sure to run from bash and not sh (venv is automatically activated that way). You can use python covert_sender.py -h for different configuration parameter options.

4 Benchmarking

All timing results are obtained by analyzing 50 average throughput calculations for each run configuration. The sender is tested with various packet sizes and sending intervals, and the plot can be seen in Figure 1, 2, 3, 4.

5 Discussion

We see that our throughput increases with increasing payload size and decreasing sending interval. This is a trivial result. But we can also see that we can get up to 2.8 Mbps with our covert channel. Whether it would be really covert or not is the subject of the next phase. But I can predict that selecting size as 64 and interval as 0.05 s is a safe bet for not being detected, and we can reach 18 kbps with this configuration.

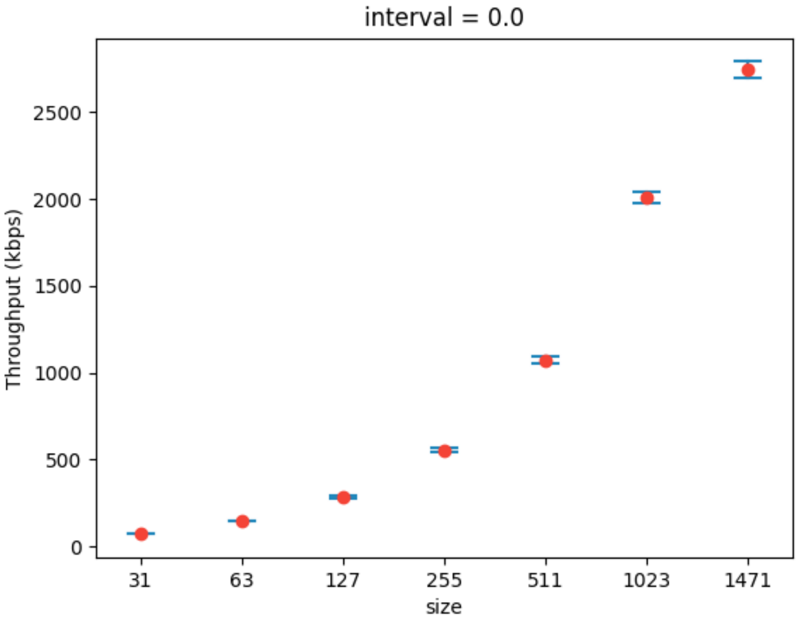

Figure 1: Sending as many packets as possible.

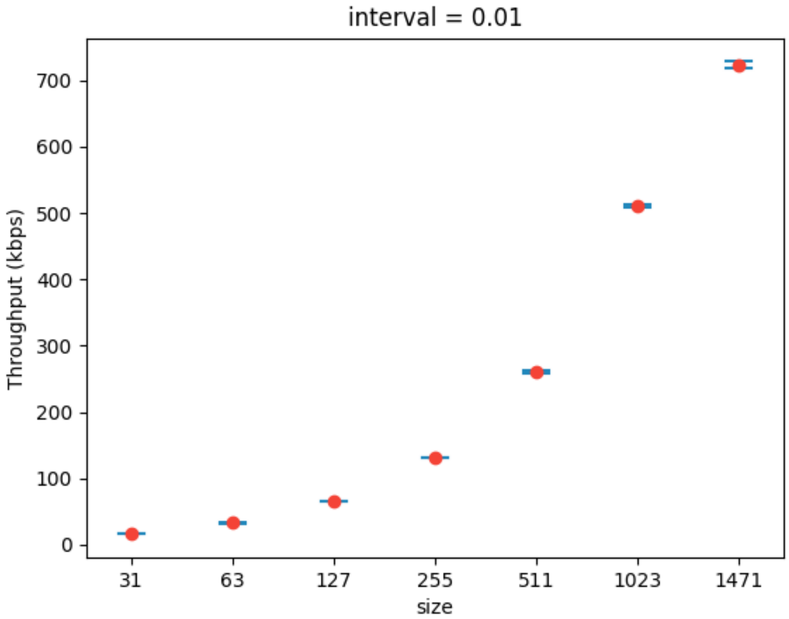

Figure 2: Sending at 100 Hz.

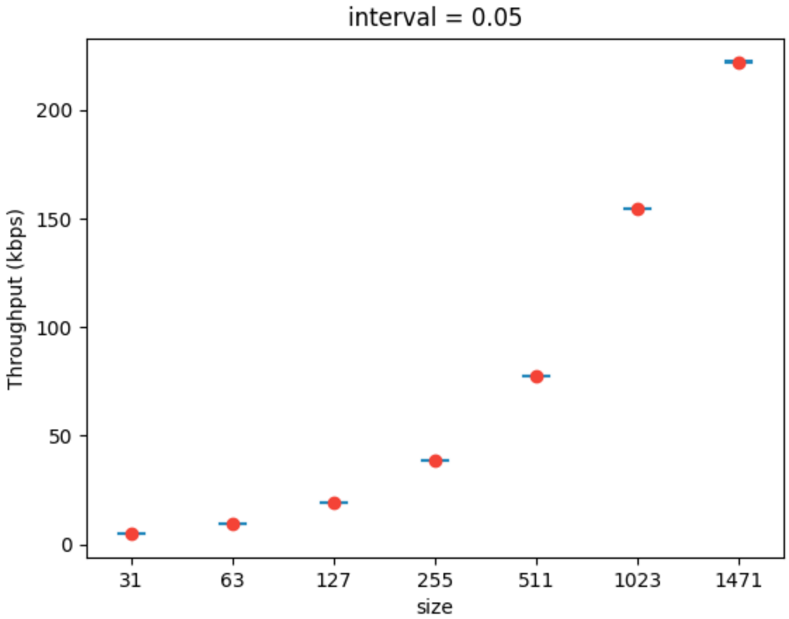

Figure 3: Sending at 20 Hz.

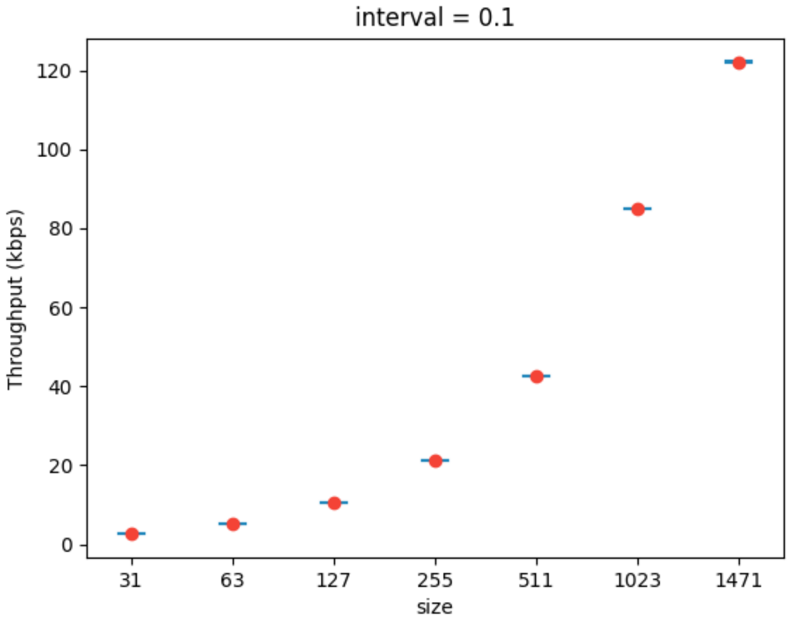

Figure 4: Sending at 10 Hz.