Network Security Term Project Final Phase
ICMP Sequence Number as Covert Channel

e2237824 - Ahmet Kürşad Şaşmaz

1 Covert Channel

The problem is using sequence numbers in ICMP packets as covert channel. In order not to be detected, our design should provide *normal-like behavior* of any ICMP packet sequence (which is normally sent by **ping** command).

1.1 Requirements

- Ping flooding is prohibited.
- Ping sequence numbers must in the increasing order.
- Due to possibility of monitoring by MITM at any level of routing, not sending any packet in the sequence is not possible.
- For successfully replied sequences, it is abnormal to wait until high sequence numbers.
- Consecutive **ping** commands shouldn't give any information about covert channel, so maximum sequence number in a group of **ping** session must be pseudo-random.

1.2 Design

1.2.1 Sending Bit 0

N : Number of packets to sent

K : Maximum value of k

k : A uniform random value

$N = 2k, k > 1, k < K$

- ping seq = 1, do not care success

- ping seq = 2, do not care success
- ping seq = 3, if fails and N = 4, then N $\leftarrow$ N+2
- ping seq = 4, if N = 4 and fails, then N $\leftarrow$ N+2, else halt.
- ping seq = 5, if fails and N = 6, then N $\leftarrow$ N+2
- ping seq = 6, if N = 6 and fails, then N $\leftarrow$ N+2, else halt.
- ...

1.2.2 Sending Bit 1

N : Number of packets to sent

K : Maximum value of k

k : A uniform random value

$N = 2k + 1, k \geq 1, k < K$

- ping seq = 1, do not care success
- ping seq = 2, if fails and N = 3, then N $\leftarrow$ N+2
- ping seq = 3, if N = 3 and fails, then N $\leftarrow$ N+2, else halt.
- ping seq = 4, if fails and N = 5, then N $\leftarrow$ N+2
- ping seq = 5, if N = 5 and fails, then N $\leftarrow$ N+2, else halt.
- ...

1.2.3 Sending Empty Information

$N = 1|2$

- do:
- ping seq = 1, if N=1, halt
- ping seq = 2, if N=2, halt
- while: at least one of the packet successes;

1.2.4 Receiving

- If source IP does not exist, create record for session
- If latest maximum sequence number is greater than current ICMP sequence number:
- If latest maximum sequence number is greater or equal than 3:
- Add element (latest maximum sequence number mod(2)) to bit array
- latest maximum sequence number ← current ICMP sequence number
- Else if latest maximum sequence number is less than current ICMP sequence number:
- latest maximum sequence number ← current ICMP sequence number

1.2.5 State Diagrams

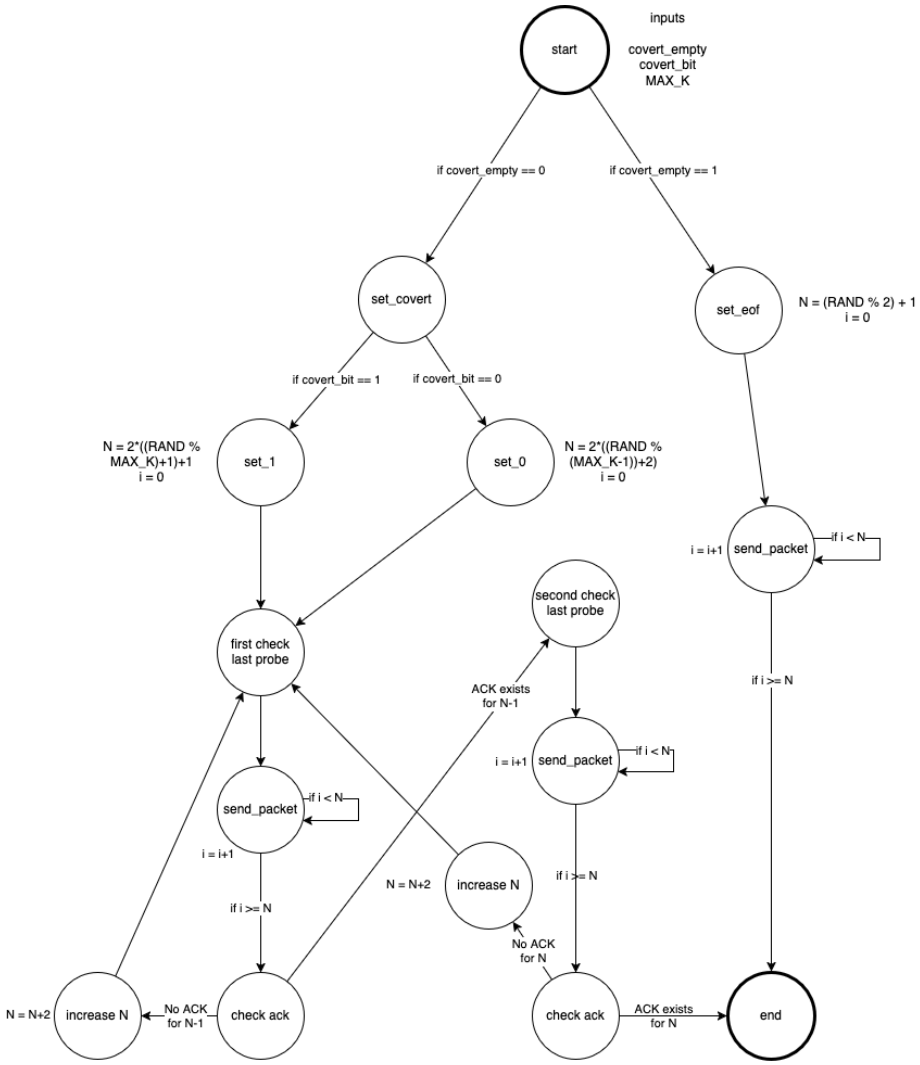

Figure 1: Cover channel sender state diagram

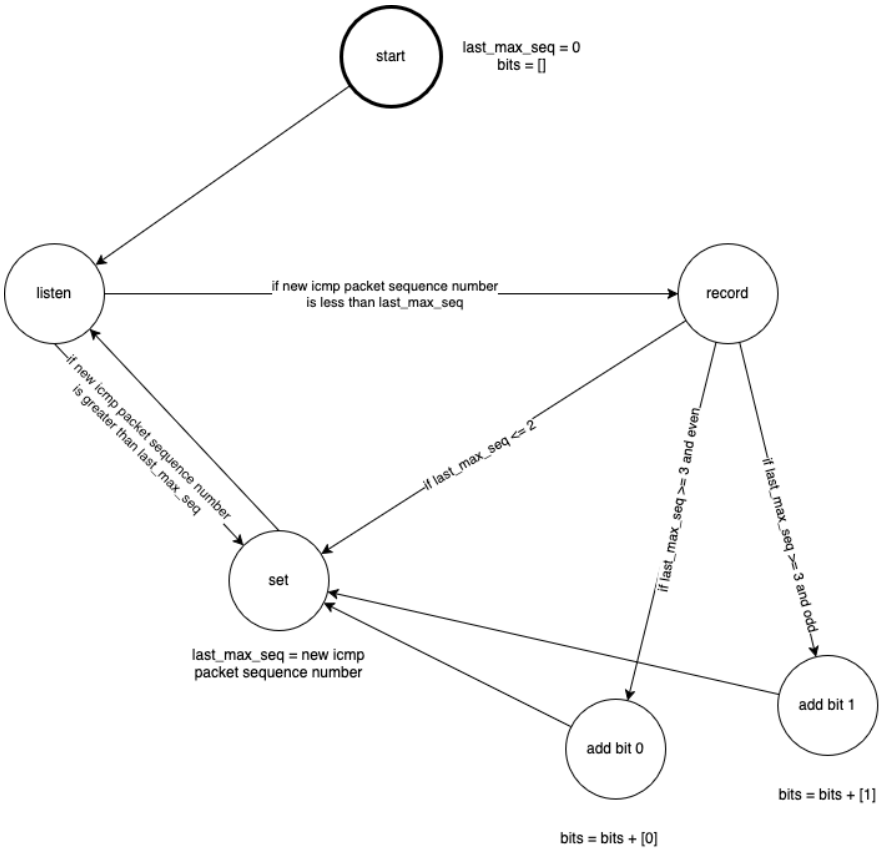

Figure 2: Cover channel receiver state diagram

1.3 Implementation

1.3.1 Sender

The original repository of **ping** from **iputils** is updated for covert channel.

1.3.2 Receiver

The command below prints all ICMP echo packets arrived.

tcpdump -i eth0 icmp and icmp[icmptype]=icmp-echo -n -l

Above command piped to our python script that parses the output of the command.

It parses and creates session if needed. Then it parses bit and adds it to the array, writes to file if file output is chosen.

1.4 Experimentation

1.4.1 Scenarios

- Send bit-0 N times
- Send bit-1 N times
- Send k random bits N times

1.4.2 Design & Implementation

A manager experimentation bash script is written to control sequence of starts and stops of docker scripts and gathering and comparing the results.

1.4.3 Parsing & Plotting

Parsing is done by human hands using Visual Code shortcuts. (Sometimes faster than parsing text file in Python). Plotting is done by online Python plotter.

1.5 Results

1.5.1 Scenario 1 : Send bit-0 N times

	TX	RX
Total	780	596
Mean	7.8	5.96
Std	2.04	1.61
%95	0.4	0.32

1.5.2 Scenario 2 : Send bit-1 N times

	TX	RX
Total	854	634
Mean	8.54	6.34
Std	2.77	1.72
%95	0.54	0.37

1.5.3 Scenario 3 : Send k random bits N times

	TX 2	RX 2	TX 4	RX 4
Total	1372	1032	2639	1975
Mean	13.72	10.32	26.39	19.75
Std	2.85	1.89	4.70	3.13
%95	0.56	0.37	0.92	0.61

1.6 Conclusions

- Drop probability of **MITM** is 0.05. So that probability of arrival of reply packet is 0.95*0.95 = **0.9025**. Expected received packet count is one tenth of transmitted packet count. We didn't achieved this expectancy so well.
- All the data is transmitted correctly even in the packet drop scenario.
- It looks like sending k bits requires approximately 7.5 transmitted packets.

2 Detector

2.1 Dataset Collection

- 100K packets are captured while while using covert channel (Covert ping behavior sending random number of random bits continuously)
- 100K packets are captures while not using covert channel (Normal ping behavior started in a loop and terminated with random timeout)

2.2 Model Architecture

- Recurrent neural network layer
- Fully connected layer
- Softmax layer

2.3 Training

Cross entropy is used as loss function. Different hidden layer size and different window size is trained for five epochs. Dataset is splitted 30-70 fashion, test and train percentage respectively.

2.4 Results

Hidden Size	16 Window	32 Window	64 Window	128 Window
16	0.48	0.51	0.69	0.70
	0.63	0.51	0.70	0.51
	0.48	0.51	0.51	0.48
	0.51	0.48	0.48	0.51
	0.48	0.48	0.48	0.51
32	**0.90**	0.48	0.54	0.62
	0.72	0.68	0.69	0.48
	0.72	0.51	0.73	0.48
	0.51	0.51	0.71	0.70
	0.51	0.51	0.73	0.71
64	0.79	0.67	0.68	0.69
	0.51	0.72	0.67	0.71
	0.72	0.72	0.71	0.69
	0.72	0.72	0.48	0.51
	0.51	0.72	0.68	0.74
128	0.76	0.72	0.75	0.70
	0.74	0.73	0.74	0.72
	0.76	0.72	0.71	0.72
	0.76	0.72	0.72	0.71
	0.75	0.73	0.72	0.71

Table 1: Accuracy values with respect to hidden layer size, window size and epoch number of training

Interestingly beside this hyper-value grid; hidden layer size 32, window size 32 and batch size 256 configuration gave us **0.99** accuracy in second epoch.

2.5 Conclusions

Considering not being an ML expert and the simplicity of the problem, I think detecting the covert channel is not so hard. As the captured ping packets gets larger, we get better idea about the distribution and the pattern on maximum sequence numbers. Also sending one or two packets rarely means that a covert channel is likely to exist. In addition, according to the design, sender must ensure that the last two packets is received by the receiver. If there are last two ping requests that didn't replied successfully and sequence number does not increase meaning there is no covert channel.

The best models luckily found a way to learn these rules and distinguish normal and covert channel ping sequence. As a result, I think it is very normal to be such accurate for a specific model.

3 Mitigator

3.1 Idea and Examination

The man in the middle shouldn't show itself. So that changing the packet contents are suspicious and needs much effort to stay undetected. But some fake reply packets does not reveal the existence of the man in the middle unless there is another communication channel beside ping packets.

Idea is that sending fake reply packets to the source sometimes while does not forwarding request packets to the destination. That means the covert channel will continue to serve if it is there, but there will be bit errors on some conditions.

Because bit construction is built on the evenness of the maximum sequence number, if the real last packet does not arrive to the receiver and the sender gets reply for that packet, the sender won't be suspicious and starts a new ping sequence for the new bit. The receiver will start reading the new bit sequence when the packet with smaller sequence number received, but there will be bit error due to the evenness of the last received packet's sequence number.

This occurs when MITM sends fake reply for the last packet that the sender sends for that bit with a certain probability. However, with a small probability, MITM may send fake reply for both the last two packets. If specified N is greater than 4, this won't cause bit error. Also fake replies for the middle ping packets does not affect the covert channel.

The good part is that bit error in the whole constructed bit string occurs more likely as the communication continues.

3.2 Experimentation

3.2.1 Design and Implementation

Only the python-processor is updated for sending fake replies with a predefined probability.

Experimentation is done similar to covert channel experimentation. Only sending K bits for N time scenario is used.

3.2.2 Results

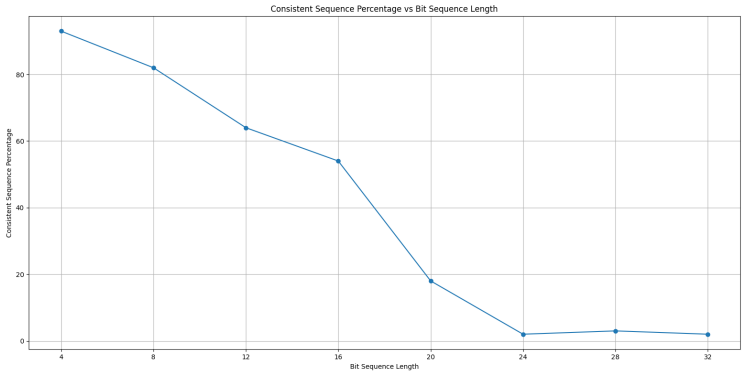

Figure 3: Consistent sequence percentage vs bit sequence length while using mitigator

3.3 Conclusions

The results show us the idea is working and the expected bit error occurrence is true. Increasing the fake reply probability may completely broke the covert channel. With less fake replies, we hide in the small probabilities from revealing ourselves as MITM.