TEAM:GPT Pros Lab1

Case 1: SPOOFING Report

The full code of this spoofer is located on the CASE1-Spoofing branch of our LAB 1 repo.

<https://github.com/CYSE587/LAB1/tree/CASE1-Spoofing>

A graph with lines and numbers

AI-generated content may be incorrect.A graph of colored lines

AI-generated content may be incorrect.A diagram of a diagram

AI-generated content may be incorrect.A graph with lines and numbers

AI-generated content may be incorrect.

After implementation of gradual spoofing, the graphs remain largely unchanged as gradual spoofing will have no effect on the throughput, packet loss, SNR, or latency since this is looking at a positional improvement only. The only difference highlighted above was the implementation of using only spoofing probabilities and having the chance of jamming be null.

The gradual implementation comes in the form of spoofer.pyA screen shot of a computer code

AI-generated content may be incorrect.

A screen shot of a computer code

AI-generated content may be incorrect.

Where the offset will gradually grow in lat/lon by 0.05 in the positive direction to slowly move it away from its original position and keep the altitude incrementally moving by -0.05 and 0.05 which may gradually move it away but uniformly will keep it near a 0 increment. I had to chose a direction in the lat/lon since if I chose -0.05 to 0.05 it would not have gradually grown it would have zeroed out.

These numbers are then added to the current lat/lon/alt which slowly increases them given every spoofed message.

I modified the n\_scen\_stat code to show these gradual spoofs within lat/lon and the differences with alt.

A graph of a graph

AI-generated content may be incorrect.

This is with a 1 probability of spoofing (hence the steady trend with only spoofing/spoofing and jamming) and a 0.7 probability within aggressive spoofing such that every 0.3 it will bottom to its original position. This trend can be made more gradual with a smaller increase. With altitude it proves my original theory that using a uniform distribution of a -0.05 and 0.05 it will never stray too far from the original creating no gradual nature.

To show maximum effect of this gradual nature, the GitHub has the changes where spoofing is probability 1 and jamming at 0. There is no modification to the nature of the tests, only the graph output in n\_scen\_stat.

Case-2: Jamming Report

A graph with different colored lines

AI-generated content may be incorrect.

In this case, the most packet loss was caused by aggressive spoofing and jamming and spoofing, indicating how well they act to block a network connection. Despite causing significant packet loss, Only Spoofing and Only Jamming fell below the disruption levels seen in combined attacks. The packet loss rate was kept constant and low by No Attack. The requirement for strong detection systems that can spot unexpected spikes in packet loss as possible signs of an ongoing attack is made clear by this. In compromised environments, the impact of packet loss can be reduced by putting error correcting systems and other routing strategies into place. The results show that packet loss is a measurable factor in intrusion detection and that setting a predetermined limit can function as an early warning system.

A graph with colorful lines

AI-generated content may be incorrect.

The most noticeable latency variations occurred in Jamming & Spoofing and Aggressive Spoofing, where irregular spikes indicated serious interference. Whereas Only Jamming and Only Spoofing produced periodic spikes but fell short of critical levels, No Attack scenarios kept latency constant. Based on these results, it may be possible to identify active jamming attempts by keeping an eye on real-time latency shifts. To reduce the impact of interference-induced latency changes, additional techniques such as path rerouting might be implemented. The results demonstrate that unstable latency is mostly caused by jamming-based attacks.

A graph with lines and numbers

AI-generated content may be incorrect.

All attack scenarios showed throughput degradation, with Aggressive Spoofing causing the most significant decrease. Jamming & Spoofing and Only Jamming showed moderate decreases in throughput, whereas No Attack scenarios maintained the best throughput. Network congestion brought on by jamming attacks severely reduces data transmission capacity, which has an impact on the system's capacity to maintain reliable communication. Adaptive congestion control mechanisms and packet prioritizing strategies should be used to increase network endurance and ensure that important messages are delivered even in difficult circumstances. According to the results, attack-induced overcrowding is a significant cause of decreased throughput, and early detection techniques must concentrate on examining patterns of throughput degradation.

A graph with blue rectangular shapes

AI-generated content may be incorrect.

Signal quality is severely compromised by jamming-based attacks, as shown by the SNR deterioration under Only Jamming and Aggressive Spoofing. The SNR remained constant in the No Attack situations, however only Spoofing had a moderate effect. According to these results, real-time SNR monitoring can be used as a means of early interference-based attack identification. Network security measures should include adaptive filtering techniques to reduce SNR loss. SNR variations are accurate indicators of attack activity. Security systems can identify and stop attacks before they have a major effect on performance by examining these patterns.