Sonata's Telemetry Results mapped to a DDOS environment

Hirsh Guha
Rushi Shah
Josh Gardner

December 11, 2020

1 Background

1.1 Problem & Importance

DDOS attacks are when a botnet of devices send traffic to a victim machine. The machine gets overwhelmed with all the traffic, and the distributed nature of the requests makes it hard to block the incoming traffic without taking the service down. This leads to a denial of service. We will implement a DDOS detection network telemetry utility using Sonata. This will allow network administrators to detect when a host in their network is experiencing a DDOS attack, so they can respond appropriately.

Existing detection solutions require changes to low level router software, which is inconvenient and invasive. In contrast, because our tool will be implemented in Sonata, it will be more easily integrated into other network telemetry services the administrator already has running through Sonata.

2 Plan

2.1 Proposal

The implementation will be within the Sonata framework, which is written entirely in Python. They have provided us with necessary libraries and methods to generate network traffic, and of course the tools to provide measurements or data about the ongoing traffic pattern.

As mentioned in the Problem section, we hope to implement traffic patterns that would replicate a ddos attack. One easy way to do that might be with a program like MEMCRASHED, which emulates a memcache-ddos attack by forcing a number of vulnerable IOT devices to send their memory cache

to a target. The amplification is sufficiently large for us to capture good telemetry results from. Within sonata, we can implement or adapt an existing DDOS-catching framework to count the number of unique sources sending data.

2.2 Questions

1. What does the telemetry of a DDOS attack look like, and how can it be modeled against other attacks in a network setting?
2. What is the speed with which Sonata can detect a DDOS attack, and how can it be improved?
3. How is a query-driven network language more intuitive or powerful than an alternative?

2.3 Evaluation Strategy

Our evaluation strategy is entirely based on our ability to create a Sonata application that can count unique addresses within a specific timeframe or with a common packet type and report that data back. Because Sonata is in itself an evaluation engine, our work automatically results in evaluation. We can then ascertain success by understanding whether our resulting telemetry matches the expected results(as given by empirical data), indicating both success in our code, and success in our use of sonata's abilities.