False Reality for Resilient Cloud Management

Ryan Sinner, Daniel Le, William Anderson, Chengxing Zhuang

Objective:

Using the concepts of migration and false reality, we plan to implement a novel algorithm that will both prevent and react to DoS attacks on a video streaming service successfully. We will then measure the differences in performance between different variations on the algorithm.

Approach:

GENI Topology

- 3 nodes running Apache server, 2 for video streaming, 1 for Dummy VM
- 1 node in a separate slice acting as the SDN controller
- 2 nodes used as switches

SDN and OpenFlow setup

- OpenFlow
  - SDN Protocol
  - Server controls network switches
  - Routing rules operate on network layer / intervene in packet routing
    - Diff. attacker from user at this stage, route accordingly
  - Supported in GENI

Differentiating between attacker and regular user

For our project, we want to make sure that we only initiate false reality when needed, on a real attacker. To minimize false positives, we will use a packet/traffic analysis library (either wireshark or ntopng) to analyze packet and connection information. The average number of normal connections for an IP address will be used as a threshold to determine suspicious behavior. If the number of connections for a particular host goes over threshold, mark as suspected attacker. Then we will monitor the suspicious user, counting the number of packets per connection sent, if passed another threshold, we will mark the user as an attacker, migrate, and initiate false reality. We will do a comparison of more tests/heuristics for detection as we develop the algorithm.

Proactive Migration Scheme

As of now we will periodically migrate from an existing pool of **n** VMs. The migration destination will be randomly chosen for now. We will do live migration to minimize down time. We will copy the resources to the destination, then have the controller redirect traffic to the new VM.

Reactive Migration Technique

If an attack is detected we will redirect traffic and responses to and from the false reality to make the attacker think their attack was successful.

False Reality

Because we are trying to protect against Slowloris we want our dummy VM to send output traffic that is similar to traffic sent out by a real server instance successfully attacked by slowloris. Thus we will test slowloris on a real server video resource, then attempt to use the server as a legitimate user. Whatever performance behavior the “legitimate user” experiences during the attack will be what the Dummy VM mimics and sends out to trick the attacker than an attack has been successful.

After a brief testing, we found out that service drops after around 450 “socket” or open tcp connections by slowloris. It doesn’t “take down” the server or crashes it, but just affects video streaming latency. We can simulate service degradation on the dummy VM by imposing traffic uploading limits as seen in Lab 1.

Attacker/Cloud Provider UI

We will be using a third party library or UI framework for the GUI of both the attacker and the cloud provider. The main parameters we will be using for now are the number of sockets and time between connections. (slowhttptest or slowloris) As for the Cloud Provider, we can change the time between the proactive migrations, the severity of throttling in the false reality, and different changes in the SDN logic.

Visualization/Metrics

For now, we plan to measure playback quality for the user (downtime, skipping in the video, download rate, etc.), time to react to an initiated slowloris attack, success rate of detection of an attacker. We will likely find more aspects which can be measured.

Responsibilities/Tasks

- Ryan
  - Implement attack detection
  - Set up traffic monitoring/packet analysis
- Daniel
  - Implement Proactive Migration Trigger AI
  - Implement False reality implementation.
- William
  - SDN/OpenFlow traffic redirection
- Cheng
  - Implement reactive migration