Bryan Kline
CS 445
HW 1
02/09/18

Project Description

Denial of Service, DoS, attacks, both direct and distributed, are a serious hindrance to the establishment of a secure Internet and they are one of the major forms of attacks against which there remains little to no defense. However, there has been some research conducted on how to trace the source of DoS attacks in order to prosecute attackers or to at least learn more about the structure of an attack. For example, Savage et al. describe a method whereby routers in between the attacker or attackers and the victim can probabilistically mark packets with their addresses so that the victim can deduce the path of attack from the relative abundance of packets from different routers that are received during the attack [Savage et al. , 2000]. Given the fact that so many more packets are generated by an attacker than from normal users, the difference in marked packets allows the victim to differentiate the path to an attacker from the paths to normal users as packets have a greater chance of being marked the closer to the victim the marking router is, and so these counts can be used to reconstruct the path of attack. Additionally, there must be different strategies for marking and reconstructing of the path in the case of one attacker versus multiple attackers.

These marking and reconstruction algorithms are described in full in *Practical Network Support for IP Traceback* by Savage et al., and they are implemented programmatically in this project. The code allows for network topologies of arbitrary size and structure to be generated, end hosts to be specified as either normal users or as attackers, for the specification of the victim, and it allows the baseline number of packets sent from each host to be specified, before a simulation is carried out where packets are sent through the system from the attackers and normal users to the victim. The attackers send the baseline number of packets multiplied by some factor, either 10, 100, or 1000, and routers on the path from an attacker or normal user to the victim have some probability of marking each packet, either 0.2, 0.4, 0.5, 0.6, or 0.8. The attack simulation involves two different marking methods, both happening simultaneously, Node Sampling and Edge Sampling, which essentially involves marking a packet with one router, specifying a point, in the case of Node Sampling, or two adjacent routers, specifying a line, in the case of Edge Sampling, the latter being crucial for identifying more than one attacker. After the simulation is run, the victim then reconstructs the putative paths for each attack and then that is checked against the actual paths to the attacker or attackers and the accuracy of the attack, for that topology, and for that attack multiplier and marking probability, is recorded. This paper describes the results of multiple runs of the simulation, both for Node Sampling and Edge Sampling, for both one and two attackers, for all combinations of attacker multipliers and marking probabilities, on three different topologies each containing 20 routers, and compares the accuracy of the algorithms.

Program Description

The program which simulates the DoS attacks and implements the Node Sampling and Edge Sampling attack reconstruction algorithms is written in C++ and is run from the Linux command line. The program allows for the user to first specify the name of the victim node and the baseline number of packets that a normal user sends in the simulation. The user then selects an existing node, starting with the victim, and adds any number of nodes connected to it, which is done continually until the map is

complete. Once the network topology is built, the user can then specify which leaf nodes, end hosts, are normal users and which are attackers. The attack is then run 15 times each for both the Node Sampling and Edge Sampling reconstruction techniques, corresponding to each combination of the attack multipliers and marking probabilities. This is achieved by iterating through every end host, normal user and attacker, for each attack multiplier and probability combination, and for each host the baseline number of packets is sent through the system. This is accomplished due to the fact that the topology is built from Router class objects which have a method for accepting a packet, either a packet in the Node Sampling case or the Edge Sampling case, and generating a random number to determine whether or not to mark it with its address. Additionally, the attack is made possible by the fact that each Router object also has a vector of Link structures, where a Link contains pointers to Routers, representing both ends of the connection between Routers. Using this vector of Links, as well as a boolean flag signaling which end of the Link leads to the victim, it is possible to quickly route packets to the victim in the simulation. The victim node stores the packets as they arrive in two vectors, one for the Node Sampling attack and another for the Edge Sampling attack, and for each different attack multiplier and probability combination, the packets that were marked are added to another structure that keeps track of counts of packets that were marked at a particular Router, and, in the case of Edge Sampling, the distance from the victim.

Once the attack simulation is complete, the victim node can then begin to reconstruct the attacks by sorting the structures containing the marked packets received, the packets in the Node Sampling attacks being sorted simply by their counts, and the Edge Sampling attacks sorted by the distance away from the victim. In the case of Node Sampling, a path is reconstructed by moving through the list of Router marks based on the counts from that Router and a simple threshold is applied to each subsequent one in the list so that if there are packets from normal users they are likely below this threshold and are removed from the list. This is the final, putative attack path for that particular attack. In the case of Edge Sampling, no threshold is applied so that all edges are preserved and a tree is constructed from them based on their distances from the victim. These are stored in a new structure for preserving the results for each attack. Once the paths and trees are constructed, these are then tested for accuracy by moving through them to ensure they correctly map onto the actual structure of the topology. If the paths and trees successfully lead back to all attackers, then that result is marked as accurate, and all other identifying information is also added so as to allow the results to then be summarized. Finally, the program prints to both the screen and a file all 30 attack results, corresponding to each attack multiplier and probability combination for both Node Sampling and Edge Sampling. The user also has the choice to print out all intermediate structures used in the construction of the topology and the attack runs, however these are mainly for debugging purposes. The program writes the results to a file called output.txt by default, however an additional command line argument can be added after the executable to specify a different name for the output file. Example output of the final results for Node Sampling and Edge Sampling on the topology labeled Network1 are shown below:

Node Sampling:

```
Baseline packets: 1
Attack multiplier: 10
Total packets: 10
Probability : 0.2
Accurate: false
Nodes in path:
     R1:7 R4:2 R8:2 R16:2 R14:1 R18:1 R20:1 R10:1
```

Edge Sampling:

```
Baseline packets: 1
Attack multiplier: 10
Total packets: 10
Probability : 0.2
Accurate: false
Attackers: 2
Tree:
[V-R1]:5 {[R1-R2]:1 {} [R1-R4]:2 {[R4-R10]:1 {[R10-R20]:1 {}}}}
```

Topologies Used

The user interacts with the program via the command line as described above, however a more convenient way to load network topologies and run attacks with various different values for baseline packets is to redirect a configuration file into the program upon execution. Included in the software package are three such configuration files which build different topologies and run attacks on them, and it is these three different topologies which are used to gather the data used to compare Node Sampling and Edge Sampling techniques described in this paper. Figure 1 shows the topology labeled Network1, included in the software package as network_1.txt, Figure 2 shows the topology labeled Network2, included in the software package as network_2.txt, and Figure 3 shows the topology labeled Network3, included in the software package as network_3.txt. The dashed lines between nodes in each topology represent links in the topologies which may vary in that the nodes they lead to may have been omitted depending on whether the attack was run on one or two attackers and one or two normal users, where N denotes normal users and A denotes attackers.

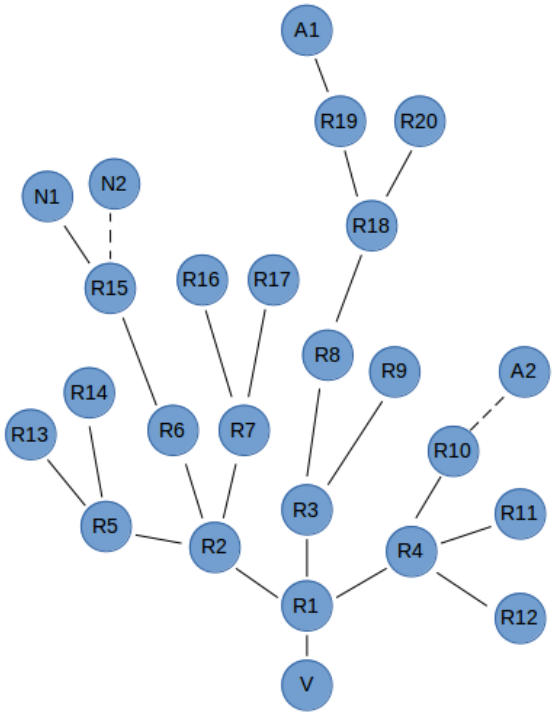

Figure 1: Topology created by configuration file titled network_1.txt.

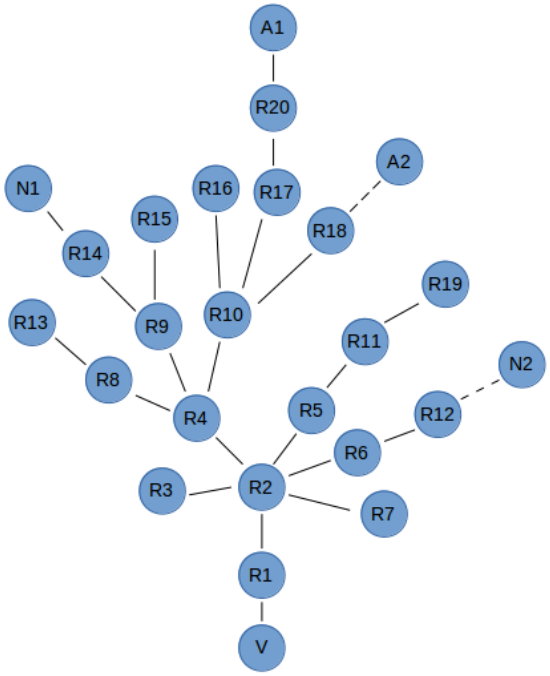

Figure 2: Topology created by configuration file titled network_2.txt.

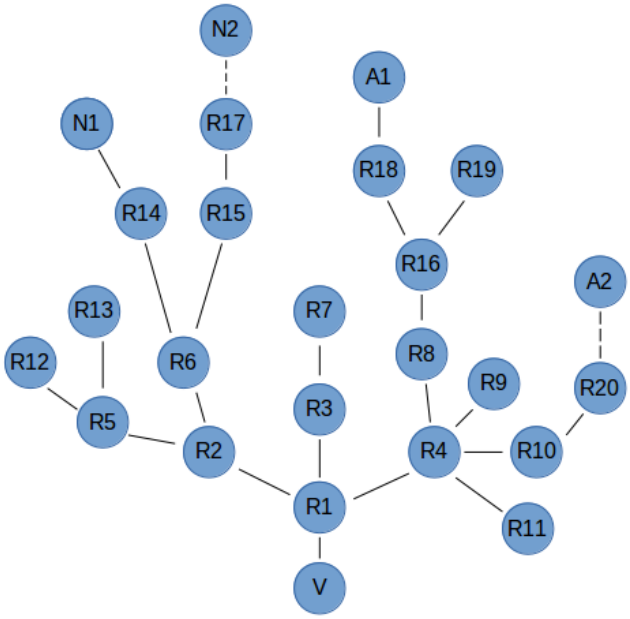

Figure 3: Topology created by configuration file titled network_3.txt.

Results

Upon running the attack simulation a number of times on each of the topologies shown above, each with an increasing number of packets, from 10 up to 10,000, each with every combination of attacker multipliers 10, 100, and 1000, and Router marking probabilities 0.2, 0.4, 0.5, 0.6, and 0.8, the results of all the combined runs are show in the plots below. Additionally, each round of attacks was carried out with both one attacker and two attackers, shown on each plot as separate trend lines. Figure 4 shows

the combined results for the three different topologies for Node Sampling and Figure 5 shows the combined results for the three different topologies for Edge Sampling. In both Node Sampling and Edge Sampling, the x-axis represents the different router marking probabilities and the y-axis represents the number of packets necessary in order for the victim to be able to accurately reconstruct the attack paths. If any particular attack resulted in a complete inability to reconstruct the attack path, then it is shown as a zero, and so zero packets should interpreted as a failure to traceback the attack.

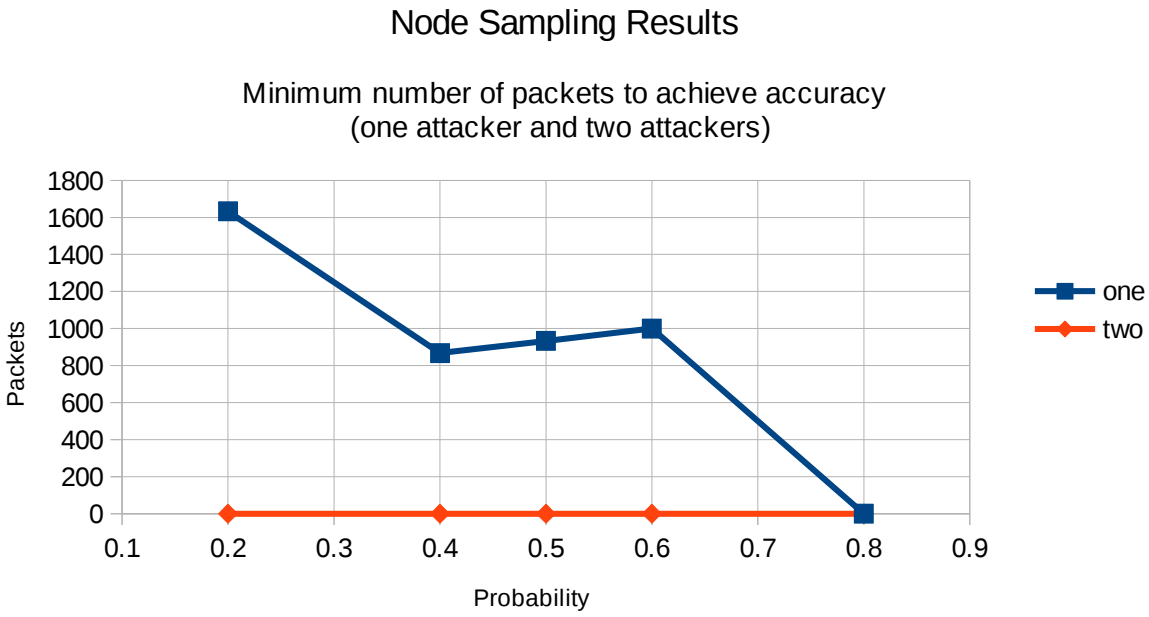

Figure 4: Results for Node Sampling traceback for the combined topologies for one and two attackers.

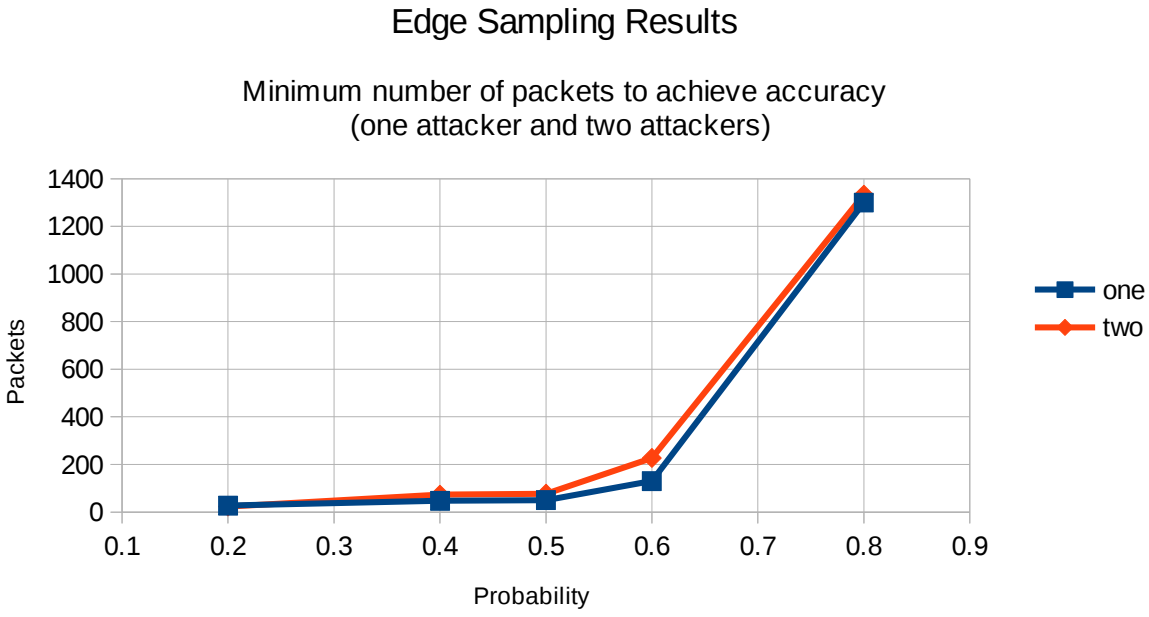

Figure 5: Results for Edge Sampling traceback for the combined topologies for one and two attackers.

Analysis

In the Node Sampling case for one attacker, as shown by the blue trend line in Figure 4, if the probability for each router along the attack path is too low it takes more packets in order to accurately traceback the attack. For example, for probability 0.2 it requires over 1600 packets to locate the attacker, and that number declines as the probability for each router to mark a packet with its address increases. This is consistent with the findings of Savage et al. who recommend a probability of 0.5 because if the probability a router will mark a packet is too low, then very distal routers in the topology will mark fewer packets and all subsequent routers are likely to overwrite those distal markings that do take place. The accuracy with fewer packets improves as the marking probabilities increase up to a point where the accuracy falls off completely. When the marking probability rises to 0.8 then the attack path cannot be accurately reconstructed at all, both because distal router markings are lost due to the high probability of their markings being overwritten by subsequent routers, and because packets coming from normal users are numerous enough to cause routers which do not participate in the attack to be included in the final attack path. In the case of two attackers, as shown by the red trend line in Figure 4, Node Sampling is not at all able to reconstruct a path back to the attacker for any of the attacks. This is because the Node Sampling technique identifies a point, whereas a line is needed if there are more than one attacker given the fact that any router along the paths are simply recored and ordered by their counts and so if there are divergent paths leading to two attackers these are erroneously combined into one path.

In the Edge Sampling case for one attacker, as shown by the blue trend line in Figure 5, the accuracy is much better than in the Node Sampling case. It takes many fewer packets to accurately construct a tree which leads to the attacker given the fact that the relative counts of the packets coming through do not contribute to the reconstruction of the paths, and so distal routers' markings are not lost and normal users' packets do not confound the paths. Instead, all packets received contribute to a tree with different paths and the branches of the tree with high counts can be followed to the attacker. The same holds for the case with two attackers, as shown by the red trend line in Figure 5. However, in both the one attacker and two attacker cases, accuracy requires many more packets as the marking probability rises to 0.8. In this case distal router's markings do get overwritten by the high probability of more proximal routers to mark the packets. In reality, the high accuracy of the lower probabilities shown in Figure 5 is due to the small size of the topologies used, and as pointed out by Savage et al., in general many thousands of packets are needed in order to ensure accuracy, and so even the more than 1200 packets in the 0.8 case is fairly low. As the number of packets increases above this all variations of the attack produce accurate traceback to the attacker or attackers.

Resources

S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Practical network support for IP traceback," *ACM SIGCOMM Computer Communication Review*, vol. 30, no. 4, pp. 295–306, Jan. 2000.