Report of
Assignment 3: Internet worm propagation simulation

By Piyusha Jaisinghani (UCFID - 4736715)

Table of Contents

Objective

Goal is to simulate a random-scanning worm propagation and local-preference scanning worm propagation in a medium-scale network by using discrete-time simulation.

Code and Variables used

The file ***randomWorm.c*** present under the folder malwareproject/wormpropagation has to code to simulate a random-scanning worm propagation.

The file ***localWorm.c*** present under the folder malwareproject/wormpropagation has to code to simulate a local-preference worm propagation.

Below are the variables used:

- *simulation* = Number of simulations = 3
- *simulationTime* = Number of time ticks = 1400
- *addressSpace* = Omega, address space = 100000
- *scanRate* = eeta, number of scans within the IP address space = 2
- *it[simulation][simulationTime]* = it[sim][tick] = an array to store infected IP at particular tick and for the specified simulation.

The method *randomGenerator* is used to generate a random number between 0 and 1

Simulation of a random-scanning worm propagation

To model the worm propagation, discrete-Time-Simulation technique is used. There are 1000 machines that are vulnerable from 100000 IP addresses.

I. Instructions to run

- Compiled the code using the following command

```
$ gcc randomWorm.c –o randomWorm.out
```

```
pi846531@net1547:~/malwareproject/wormpropagation$ pico randomWorm.c
pi846531@net1547:~/malwareproject/wormpropagation$ gcc randomWorm.c -o randomWorm.out
pi846531@net1547:~/malwareproject/wormpropagation$ ls
randomWorm.c  randomWorm.out
pi846531@net1547:~/malwareproject/wormpropagation$
```

- Executed the file ***randomWorm.out*** using the following command, once the code is executed an output file named ***randomWormOutput.txt*** gets generated which consists of the output:

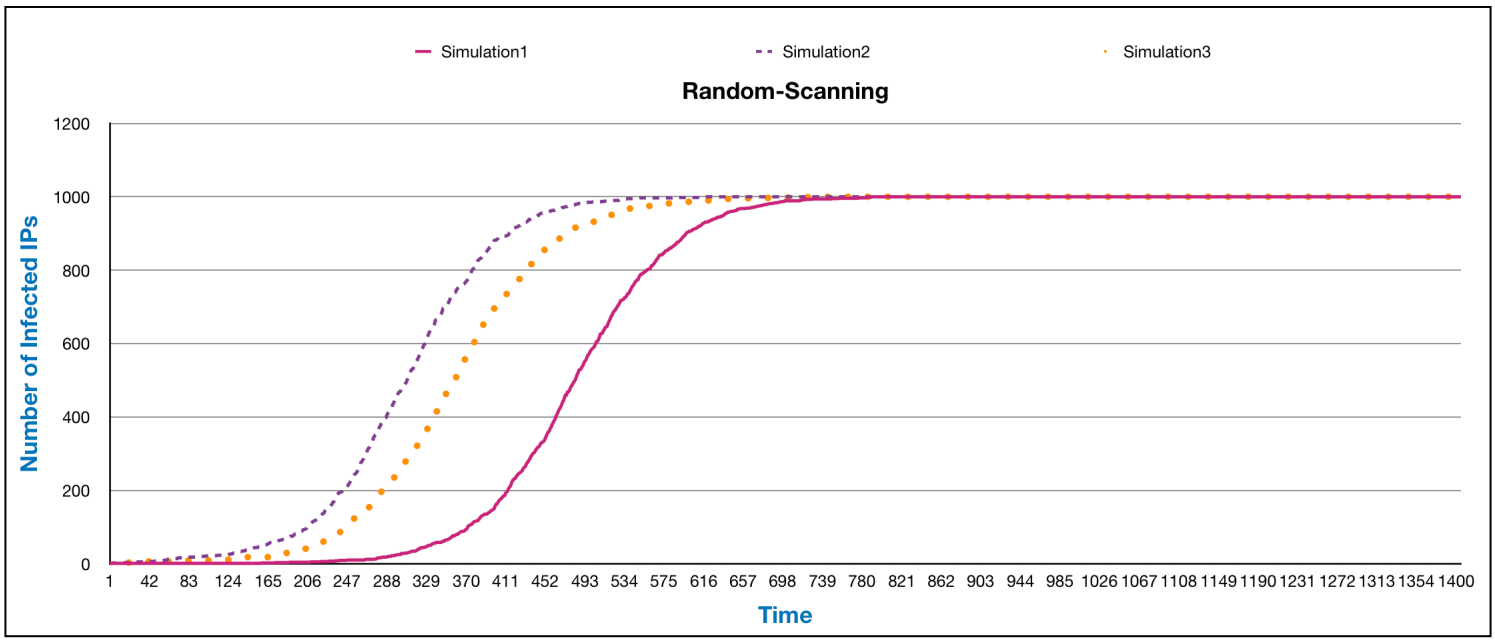
$ ./randomWorm.out

```
pi846531@net1547:~/malwareproject/wormpropagation$ ./randomWorm.out
pi846531@net1547:~/malwareproject/wormpropagation$ ls
randomWorm.c  randomWorm.out  randomWormOutput.txt
pi846531@net1547:~/malwareproject/wormpropagation$
```

II. Design

- The Simulation is set to 3, simulation time to 1400 and the enum nodeMode defines the states of the node, empty, infected and susceptible.
- The nodeStatus of all the IPs in the address space is initialized to empty and all the infected IPs nodeStatus is set to susceptible.
- In the random worm propagation, for every time tick in a particular simulation each infected IP scans with a scanRate of 2. If the IP address are not infected already and are susceptible, those will get infected.
- The nodeStatus vector keeps track of the status of each IP i.e. infected, not infected, susceptible. The while loop handles simulation time for 1400 ticks for each simulation carried out by a for loop.
- Two new IP addresses are generated inside the while loop for each infected machine and we check if they are susceptible and not infected. If so, then infect them and increment the value in infectedIp.
- Once we check all the infected IPs for a particular time tick at scan rate 2, the updated IP counter is stored in the *it* array.
- Once all the simulations gets completed, the result is stored in randomWormOutput.txt.
- Using an excel sheet (RSheet.xls) in which the contents of the text file are imported the following chart is plotted.

On the y axis we represent the Number of infected IPs and the x- axis is for time. The plot shows the number of infected IPs at each time tick for all three simulations. At about 792nd time tick all the 1000 machines are infected



Simulation of a local-preference scanning worm propagation

To model the worm propagation, discrete-Time-Simulation technique is used. For local preference scanning worm propagation, for a particular time tick each infected machine is scanned for two local IP addresses within the range x-10 to x+10 with a 60% probability and random addresses for 100,000 ip addresses with a 40% probability. The susceptible machines are clustered together, as local preference scanning is used. So, all 1000 machines should be infected faster as compared to the previous case.

I. Instructions to run

- Compiled the code using the following command

$ gcc localWorm.c –o localWorm.out

```
pi846531@net1547:~$ cd malwareproject/wormpropagation/
pi846531@net1547:~/malwareproject/wormpropagation$ ls
localWorm.c  randomWorm.c  randomWorm.out  randomWormOutput.txt
pi846531@net1547:~/malwareproject/wormpropagation$ gcc localWorm.c -o localWorm.out
pi846531@net1547:~/malwareproject/wormpropagation$ ls
localWorm.c  localWorm.out  randomWorm.c  randomWorm.out  randomWormOutput.txt
pi846531@net1547:~/malwareproject/wormpropagation$
```

- Executed the file ***localWorm.out*** using the following command, once the code is executed an output file named ***localWormOutput.txt*** gets generated which consists of the output:

$ ./localWorm.out

```
[pi846531@net1547:~/malwareproject/wormpropagation$ ./localWorm.out
[pi846531@net1547:~/malwareproject/wormpropagation$ ls
localWorm.c  localWorm.out  localWormOutput.txt  randomWorm.c  randomWorm.out  randomWormOutput.txt
pi846531@net1547:~/malwareproject/wormpropagation$
```

II. Design

- The Simulation is set to 3, simulation time to 1400 and the enum nodeMode defines the states of the node, empty, infected and susceptible.

- The nodeStatus of all the IPs in the address space is initialized to empty and all the infected IPs nodeStatus is set to susceptible.

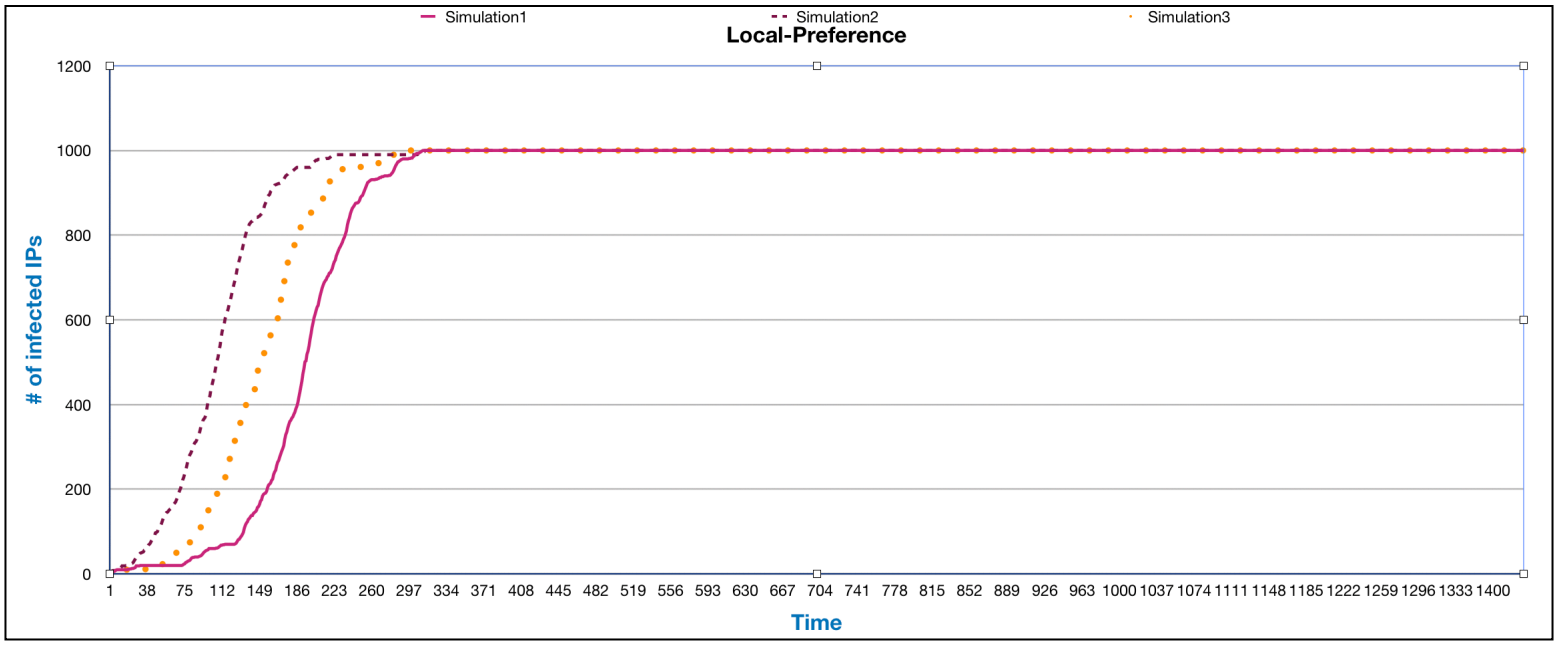
- In the local-preference worm propagation, for every infected machine generate a random number between 0 and 1.
- If the probability is less than 60% generate two new IP addresses inside the loop for each infected machine and we check if they are susceptible and not infected. If so, then infect them and increment the infectedIP.

- If the probability is less than 40% generate two random IP addresses from 100,000 IP address space then infect them and increment the infectedIP.

- Once the loop is completed the nodeStatus is stored in to priorNodeStatus and the infected IPs is stored in the *it* array.

- Once all the simulations gets completed, the result is stored in localWormOutput.txt.

- Using an excel sheet (LSheet.xls) in which the contents of the text file are imported the following chart is plotted.



On the y axis we represent the Number of infected IPs and the x- axis is for time. The plot shows the number of infected IPs at each time tick for all three simulations. At about 252nd time tick all the 1000 machines are infected.

The zip file consists of:

- A folder named **code** that consists the files randomWorm.c, localWorm.c, randomWorm.out and localWorm.out.
- A folder named Output Files consists of the output files generated from the two simulations localWormOutput.txt and randomWormOutput.txt.
- A folder named **Excel Sheets** that consists of 2 excel sheets LSheet.xlsx and RSheet.xlsx that are the two files generated from the output files.