SPL 161 Assignment 1 - Cybersecurity Simulation

November 11, 2015

1 Before You Start

- It is mandatory to submit all of the assignments in pairs. It is recommended to find a partner as soon as possible and create a submission group in the submission system. Once the submission deadline has passed, it will not be possible to create submission groups even if you have an approved extension.
- Read the assignment together with your partner and make sure you understand the tasks. Please do not ask questions before you have read the whole assignment.
- Configuration files for your program will be provided in XML format. You can read about XML file format here: https://en.wikipedia.org/wiki/XML
- Skeleton classes will be provided on the assignment page, you must use these classes as a basis for your project.

2 General Assignment Definition

2.1 Network worms Vs. System Admins

In this assignment you will simulate an exciting cyber-battle between vicious **cyber-worms**, malicious programs that wreak havoc online spreading between innocent computers via network connections, and **cyber-experts**, courageous cyber-warriors braving the dark web with only one mission - free the network from the scourge of the cyber-worms!

The simulation will run in time-steps, each time-step simulates the events occurring during one day. During each time-step actions will occur according to a pre-determined set of rules specified below.

Figure 1: Vicious cyber-worm infecting a helpless pc.

2.2 Assignment goals

The objective of this assignment is to design an object-oriented system and gain implementation experience in C++ while using classes and standard STL-based data structures. You will learn how to handle memory in C++ and avoid memory leaks. You will learn to work with a 3rd party library - Boost – This library will help you work with the XML files. The resulting program must as efficiently as possible.

Also, cyber.

2.3 Simulated Objects

- Computers - these are objects which might contain a worm or a system admin at a specific moment in time. Computers are defined by:
  - Name (string)

- Operating system (string)
- Worm (complex object) - Each PC may have at most one active worm at any point in the simulation.

- Network connection - these are ”wires” that connect computer to each other and they define the actual network. Connections are defined by:
  - Point-A (computer name)
  - Point-B (computer name)
- Computer worm - these objects can infect a computer and, using its network connection, infect neighboring computers in time. Computer worms are defined by:
  - name (string)
  - dormant-period - a number that determines how much time units it takes an infected computer to become infectious (int)
  - os - type of the os the worm likes to infect (if the os of the computer does not match, it will not infect the target)
- Cyber Experts (admins) - these are objects that are capable of inspecting a computer, checking it for infections and clean them. Admins are defined by:
  - Name (string)
  - work-time - the number of time units the admin will work before taking a break (int)
  - break-time - the number of time units the admin will rest before going back to work (int)
  - efficiency - the amount of computers an cyber-expert can examine per time-unit (int)
- DNS (cyber) Server - This is a passive object that holds pointer to all computer instances in a hash-map where computer names serve as keys. Any access to computer objects should be performed by supplying a computer name to the DNS Server instance and receiving a pointer to the desired object as a return value.

2.4 Building the network

The building of the network is the first step your c++ program needs to make. In order to do that you need to read supplied input files: `computers.xml` Defines the computers sitting on the network, the file is provided in the following format

```
<computer>
   <name>[computer name]</name>
   <os>[operating system]</os>
</computer>
<computer>
   <name>[computer name]</name>
   <os>[operating system]</os>
</computer>
.
.
.
```

`network.xml` Defines the connections between computers, the file is provided in the following format:

```
<wire>
   <pointA>[computer name]</pointA>
   <pointB>[computer name]</pointB>
</wire>
<wire>
   <pointA>[computer name]</pointA>
   <pointB>[computer name]</pointB>
</wire>
.
.
```

.

Utilize c++ boost library to read the xml files and create your network before the simulation begins.

2.5 Simulation Events

Once you have set up your network, the simulation can begin. The `events.xml` file details the events which will occur throughout the simulation. In each time-step, one event is read from the file and simulation objects are updated accordingly. The format of the file is as follows:

- Hack event: this event signifies that a hacker hacked his/her way into the network and planted a worm on a computer

  ```
  <hack>
    <computer>[hacked computer name]</computer>
    <wormName>[worm name]</wormName>
    <wormDormancy>[worm dormant time]</wormDormancy>
    <wormOs>[worm favorite os]</wormOs>
  </hack>
  ```

- Clock-In event: this event signifies that an Admin clocked-in and is now starting his/her workday

  ```
  <clock-in>
    <name>[admin name]</name>
    <workTime>[admin work time]</workTime>
    <restTime>[admin rest time]</restTime>
    <efficiency>[admin efficient setting]</efficiency>
  </clock-in>
  ```

- Termination event : This event signifies the end time of our simulation, once this event is read, the simulation will continue until the specified time unit listed in the termination event.

  ```
  <termination>
    <time>[time unit to terminate at]</time>
  </termination>
  ```

  Once a termination event is read, no further events should be read from the events file.

In addition to these scripted events, each of the active "players" in the simulation (i.e. computers, cyber-experts and worms) will perform their default actions during each time step:

- **Cyber Experts** - Examine computers or rest.
- **Computers** - Connect to the network, spreading worms if they are infected.
- **Worms** - Replicate themselves to compatible computers.

2.6 Simulation Workflow

After the building of the network the simulations iterates until termination time while following the order of events (if they need to happen at the same time unit) as follows:

- Clock-In events
- Hack events
- Cleaning events
- Infection events
- Termination events

Each event should generate an output to the console as listed in the next section.

Figure 2: Cyber Expert protecting CyberSpace

2.7 Output

Your simulation should output a notification for the following events to the console. Information in square parenthesis should be replaced with actual values, where several options are displayed, the appropriate ones should be selected:

- PC initialization (before first time step, only once)

```
[PC name] connected to server
```

- Connection between PC's:

```
Connecting [PC A] to [PC B]
   [PC A] now connected to [PC B]
   [PC B] now connected to [PC A]
```

- Hacking Event:

```
Hack occured on [PC name]
   [PC name] infected by worm-name
```

- Worm Activity:

```
Worm [worm name] is dormant
[PC name] infected by [worm name]
Worm [worm name] successfully removed from [PC name]
```

- PC activity:

```
[PC name] infecting...
```

- Cyber Expert activity:

```
[cyber expert name] clocked in
[cyber expert name] examining [PC name]
[cyber expert name] is taking a break
[cyber expert name] is back to work
```

- Beginning of time step:

```
Day [step number]:
```

Example output files will be available on the assignment homepage.

3 Implementation requirements

3.1 Class Skeletons

The following files will be provided for you on the assignment homepage:

```
cyberexpert.h
cyberpc.h
cyberworm.h
cyberdns.h
```

While you are free to add your own functions and variables to these header files (in fact, you MUST do so in order to complete the assignment successfully), you are required to use the supplied functions in your implementation.

Any attempt to *circumvent* these functions (i.e. storing pointers to connected computers in the `CyberPC` class rather than using the supplied string vector) - will result in a significant penalty to your grade.

3.2 Use of Pointers and Memory Management

In order to simulate a computer network, you will implement the class `CyberDNS` which hold **pointers** to `CyberPC` objects in a **hash map** of **strings**. Any access to a PC object should be performed through the `CyberDNS::GetCyberPC(string)` function.

Take care not to duplicate objects unnecessarily, and be sure to implement destructors for each class which will free all memory allocated to the object. We will test your program using VALGRIND in order to ensure no memory leaks have occurred.

3.3 Parsing XML files

In order to parse XML files, you must use the BOOST `ptree` and `xml_parser` classes. Documentation for these classes can be found here: http://www.boost.org/doc/libs/1_57_0/doc/html/property_tree.html

You may assume that the root directory of your project contains the following files:

```
computers.xml
network.xml
events.xml
```

It may also be assumed that these files are valid, both in format and in content. There will be no syntax error in the input files, duplicate PC names, duplicate connections etc...

3.4 Sample Data

Sample data and the desired output will be available on the course website.

4 Submission

Your submission should be in a single zip file called "student1ID-student2ID.zip". The files in the zip should be set in the following structure:

```
src/
include/
bin/
makefile
```

src directory should include all .cpp files that are used in the assignment. include directory should include all the header (.h or *.hpp) files that are used in the assignment. bin directory where the compiled objects (the .o files and the assignment executable) should be placed when compiling your source files. There is no need to submit binary files (your "bin" directory should be empty). The makefile should compile the cpp files into the bin folder and create an executable named "cyber" and place it also in the bin folder.

Your submission will be tested by running the following commands:

```
make
./bin/cyber
```

You submission must compile **without warnings or errors** on the department computers, compiler commands must include the following flags:

```
-Wall -g
```

The submissions must be made in pairs. After you submit your file to the submission system, re-download the file that you have just submitted, extract the files and check that it compiles on the university labs. Failure to properly compile or run on the departments computers **may result in a zero grade for the assignment**.

5 Grading

Although you are free to work wherever you please, assignments will be checked and graded on CS Department lab computers - so be sure to test your program thoroughly on them before your final submission. "But it worked on my Windows/Linux based home computer" will not earn you any points if your code fails to execute at the lab.

6 Questions

All questions regarding the assignment should be published in the assignment forum. Please search the forum for similar questions before publishing a new one. Question by mail regarding the assignment will be ignored.

7 Delays

In case of reserve duty (aka Sherut Miluim), illness, etc. please send an email to the course mail majeek@cs.bgu.ac.il. The mail should include partners name and IDs, explanation and certification for your illness / Miluim. We will not answer other mail in the course mail.

8 Appeals

The policy is very simple, There are no appeals! Please make sure you made all the necessary QA on your work before submitting it. It is highly recommended to download your submission, compile and run it on one of the lab computers on a clean folder.

9 Cyber Cyber Cyber

Upon completing this assignment, you may find that constant use of the word ”cyber” causes everything to sound ridiculous. This is an important lesson: refrain from abusing the term ”cyber”.

The Webster dictionary defines ”Cyber” as ”of, relating to, or involving computers or computer networks”. Hence any internet connected cellular phone becomes a ”Cyber-Phone”, this assignment is a ”Cyber-Assignment” and e-mail becomes ”Cyber-Mail”. Since you are studying to become computer scientist and not starring in a ”hacker” movie from the 90’s, please use professional language when discussing computer security.

Good luck with your assignment!

May the cyber be with you ;)

פצצות אטום? זה כל כך אייטיז, התקדמנו לפצצות סייבר

כמעט כל מדינה בעולם יכולה להפוך כיום למעצמה צבאית בעזרת האקרים מומחים ולא ניתן לפקח על הנושא, פייסבוק מציגה לרגולטורים זועמים תירוץ יצירתי במיוחד, איך נראים בפועל הפיטורים בטוויטר ומהנדס בונה את הפטיש של ת'ור

יוסי גורביץ | 14.10.2015, 15:10

Figure 3: Don’t be this guy.