School of Computing & Information Technology

CSCI262 System Security
SIM-2017-S2

Assignment 3 (14 marks, worth 14%)

Due 22 May 2017 23:59 (Singapore Time)

The groups

Students can form groups of size 2-3.

Overview

Each group needs to design and implement an Email system event modeller & intrusion detection system in accordance with the system descriptions below. The implementation is to be in C, C++ or Java. While there are concrete details on the form of the initial input, and certain inputs along the way, the format of intermediate data is up to each group.

You need to provide a report in a file `Report.pdf` covering the various points through this assignment where information is required. This report should be broken into sections associated with the components as follows:

- Initial input.
- Activity engine and the logs.
- Analysis engine.
- Alert engine.

Initial Input

You only need command line options at the setup phase, some user input is required later.

```
IDS Events.txt Stats.txt Days
```

`Events.txt` and `Stats.txt` define the formats and the distributions of the events to be modelled. `Days` is an integer used in the next section.

Here goes an example `Events.txt` file. This file describes the events and some of their parameters.

```
4
Logins:D:0::2:
Time online:C:0:1440:3:
Emails sent:D:0::1:
Emails opened:D:0::1:
```

The first line contains the number of events being monitored. Each subsequent line is of the form

`Event name:[CD]:minimum:maximum:weight:`

C and D represent continuous and discrete events respectively. Discrete events must take integer values and occur one at a time, continuous events don't need to take an integer value and an occurance of that event may be of any value. The minimum and maximum specify the allowed range for that event type across a day. Continuous events need be recorded to two decimal places. The weights are used in the alert engine and will always be positive integers.

The file `Stats.txt` contains the distributions to be modelled for the events. Here goes an example `Stats.txt` file.

```
4
Logins:4:1.5:
Time online:150.5:25.00:
Emails sent:10:3:
Emails opened:12:5:
```

The first line again contains the number of events being monitored. Each subsequent line is of the form

`Event name:mean:standard deviation:`

Your program should appropriately report events and statistics read in, as evidence this phase works. You should include in your report a description of:

1. How you are going to store the events and statistics internally.
2. Potential inconsistencies between `Events.txt` and `Stats.txt`. You should attempt to detect those inconsistencies. If there are inconsistencies you are aware of but haven't attempted to detect them, note this in your report.

Activity Simulation Engine and the Logs

Once the intial setup has taken place, and you have read in the base files, the activity engine should start generating and logging events. Your program should give some indication as to what is happening, without being verbose.

You are attempting to produce statistics approximately consistent with the statistics specified in the file `Stats.txt`. You should log for the number of `Days` specified at the initial running of `IDS`. You can, if you like, store the events in distinct files for each day, or in a single log file. This collection of events forms the baseline data for the system.

You should include in your report a description of:

1. The process used to generate events approximately consistent with the particular distribution. This is likely to differ between discrete and continuous events. If you can handle some but not all of those note this.

2. The name and format of the log file, with justification for the format. You will need to be able to read the log entries for subsequent parts of the program. The log file needs to be human readable.

Analysis Engine

Your program should indicate it has completed event generation and is going to begin analyis. You can now measure that baseline data for the events and determine the statistics associated with the baseline.

Produce totals for each event for each day, store that in a data file, and determine the mean and standard deviation associated with that event across that data. Report what is happening as you consider appropriate.

Even if you are unable to produce data consistent with a given distribution you can still have the analysis engine reading and reporting on the log file.

You should include in your report the name and format of the file containing the daily totals and statistical data for the events.

Alert Engine

The alert engine is used to check consistency between "live data" and the base line statistics. Once this phase is reached you should prompt the user for a file, containing new statistics, and a number of days. The new statistics file has the same format as `Stats.txt` from earlier but will generally have different parameters for the events. You should run your activity engine and produce data for the number of days specified. Use the analysis engine to produce daily totals, those are used in alert detection.

For each day generated you need to report on whether the there is an intrusion detected by comparing an anomaly counter with a threshold. You calcualte the anomaly counter by adding up the weighted number of standard deviations each specific tested event value is from the mean for that event, where the standard deviation and mean are those you have generated from the base data and reported, and the weight is taken from the original `Events.txt` file.

For example, if the mean number of logins per day is 4 and the standard deviation is 1.5; then if we get 1 login in a day we are 2 standard deviations from the mean. Referrring back to the weight of the login event we see it was 2 so the login event contributes 4 to our overall anomaly counter.

The threshold for detecting an intrusion is 2∗(Sums of weights) where the weights are taken from `Events.txt`. If the anomaly counter is greater or equal to the threshold you should report this as an anomaly.

You should output the threshold, and give the anomaly counter for each day as well as stating each day as okay or flagged as having an alert detected.

Once the alert engine part has finished you should return to the start of this phase, so another set of statistics and number of days can be considered. An option to quit should be provided.

Notes on submission

1. Submission is via Moodle. Everything will need to be uploaded in a single zip file. Please don't put subdirectories in your submission. In addition to addressing the specified points, you can include anything of significance in your report. You should report on any parts not completed.
2. One person from each group needs to make a submission. If multiple people make submissions the last one received will be marked.
3. Include the compilation instructions with your submission (i.e., provide a readme.txt file).
4. Late submissions will be marked with a 25% deduction for each day, including days over the weekend.
5. Submissions more than three days late will not be marked, unless an extension has been granted.
6. If you need an extension apply through SOLS, if possible **before** the assignment deadline.
7. Plagiarism is treated seriously. Students involved will likely receive zero.