

Initial Input

The program accepts either 3 or 4 arguments when attempting to run it from the command line. The command to run the program is:

IDS <events file> <stats file> <days> <(optional) username>

The events file and stats file are self-explanatory. **Days** is used to generate the appropriate number of entries in the log file, and the **username** is the name of the person the statistics is based on. If the username field is left out, the system will default to using “Peter” as the user. Thus, the completed command, with proper arguments looks like this:

IDS Events.txt Stats.txt 1000 Isaac

Storage

The Events.txt and Stats.txt have specific formats that have to be followed. This is the Events.txt file:

```
5
Logins:D:0::3:
Time online:C:0:1440:2:
Emails sent:D:0::1:
Emails opened:D:0::1:
Emails deteled:D:0::2:
```

Figure 1: The Events.txt file

The first line is merely a counter, which shows the number of entries in the file. Each line after that is an event, with the following format:

<Event Name>:<Continuous(C) or Discrete(D)>:<Min>:<Max>:<Weight>:

A continuous event is one where it occurs over a period of time (*i.e. time online*), whereas a discrete event is one where each occurrence is a separate, or discrete, event (*logging in, sending emails, etc*).

There is also a minimum and maximum occurrences allowed, and the weight, which is used to determine if there are any anomalies.

The stats file is as follows:

```
5
Logins:4:1.5:
Time online:150.5:25.00:
Emails sent:10:3:
Emails opened:12:4.5:
Emails deteled:7:2.25:
```

Figure 2: The Stats.txt file

Similar to the event file, the first line is a counter, with the following lines of the format:

<Event Name>:<Mean>:<Standard Deviation>:

The mean and standard deviation are derived from a set of event files, with the relevant calculations performed on them.

We have created Stat, Event and Log classes to help facilitate the storage of the information. The Event class is the base class for the DiscreteEvent and ContinuousEvent classes. After reading the information from the events and stats files, we will dynamically create instances of those classes and store them into a vector.

```
vmw_ubuntu@vmwubuntu:~/Desktop/src$ IDS Events.txt Stats.txt 1000 IsaacY

Starting Learning Phase
#####
Reading events from Events.txt!
Reading stats from Stats.txt!
DONE!
~~~~~
Total number of Events read          -> 5
Number of events in statistical data  -> 5
Data is CONSISTENT!
~~~~~
Generating logs for 1000 days..
DONE!
Writing logs to IsaacY_1000_Daily.log..
DONE!
~~~~~

Starting Analysis Phase
#####
Reading IsaacY_1000_Daily.log
```

Figure 3: The output after the LEARNING PHASE is completed

Data Inconsistency or Missing Files

In the event that the statistics and events file are not consistent (*i.e. the number of entries do not tally*), or any of them are missing, the system will alert the user and will not proceed.

```
Starting Learning Phase
#####
Reading events from Events.txt!
Reading stats from Stats.txt!
DONE!

Total number of Events read      -> 5
Number of events in statistical data -> 4
ERROR! The number of entries in Events.txt and Stats.txt do not match!
Data is INCONSISTENT!
Please make sure the files are present and consistent.
vmw_ubuntu@vmwubuntu:~/Desktop/src$
```

Figure 4: The system will alert the user and exit if the files are missing or inconsistent with each other.

The system will also alert the user and not proceed if it finds that the information in the events and statistics file do not tally (*i.e. different names, or the names do not match*):

```
vmw_ubuntu@vmwubuntu:~/Desktop/src$ IDS Events.txt Stats.txt 1000 IsaacY

Starting Learning Phase
#####
Reading events from Events.txt!
Reading stats from Stats.txt!
DONE!

Total number of Events read      -> 5
Number of events in statistical data -> 5
ERROR! The event: "Emails opened" does not exist in Stats.txt
Data is INCONSISTENT!
Please make sure the files are present and consistent.
vmw_ubuntu@vmwubuntu:~/Desktop/src$
```

Figure 5: The system will also alert the user if the contents in each file are inconsistent.

After this is done, the program will use the **days** and **username** argument provided in the command used to start the program to generate a series of log entries.

The log entries are of the format:

<Event Name>:<C or D>:<Value>:<Weight>:

These log entries will be used to generate a new statistics file which will serve as the baseline for the comparison of data in the next part. Using these two arguments, it will create a file called <username>_<days>_Daily.log, which will contain the aggregated log entries for each day, as shown on the right.

Analysis Engine and Reading Logs

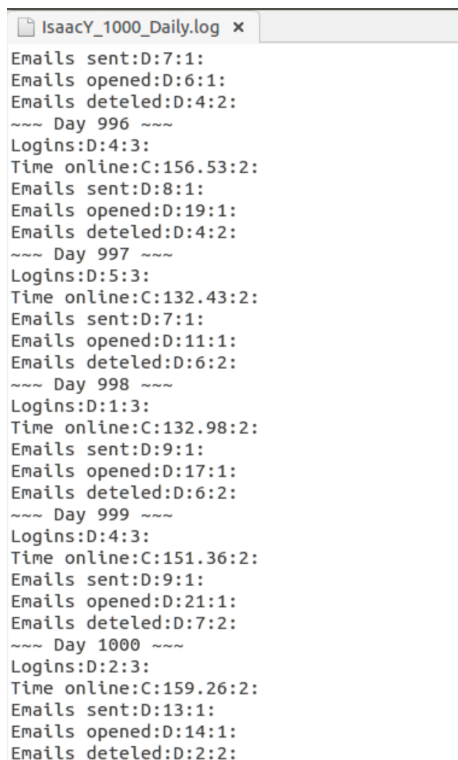
The activity engine reads the logs generated in the previous step. This is probably not done in a real IDS system, where the information would be captured and stored through actual usage. In this simulation, however, we generate the logs randomly because we don't have any actual information.

Using these logs, the program can generate a new statistics file, which will be used as our baseline to determine if the entered information is anomalous or not.

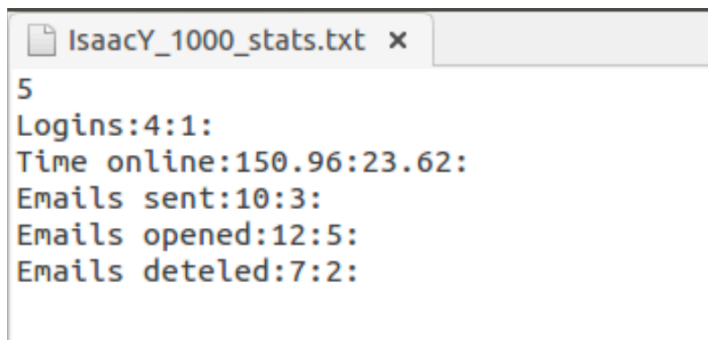
***Name of log file generated in directory: [username]_xx_Daily.log**

***Name of statistical text file generated in directory: [username]_xx_stats.txt**

Figure 6 & 7: Daily logs (figure 6) & stats (figure 7)



```
IsaacY_1000_Daily.log x
Emails sent:D:7:1:
Emails opened:D:6:1:
Emails deteled:D:4:2:
~~~ Day 996 ~~~
Logins:D:4:3:
Time online:C:156.53:2:
Emails sent:D:8:1:
Emails opened:D:19:1:
Emails deteled:D:4:2:
~~~ Day 997 ~~~
Logins:D:5:3:
Time online:C:132.43:2:
Emails sent:D:7:1:
Emails opened:D:11:1:
Emails deteled:D:6:2:
~~~ Day 998 ~~~
Logins:D:1:3:
Time online:C:132.98:2:
Emails sent:D:9:1:
Emails opened:D:17:1:
Emails deteled:D:6:2:
~~~ Day 999 ~~~
Logins:D:4:3:
Time online:C:151.36:2:
Emails sent:D:9:1:
Emails opened:D:21:1:
Emails deteled:D:7:2:
~~~ Day 1000 ~~~
Logins:D:2:3:
Time online:C:159.26:2:
Emails sent:D:13:1:
Emails opened:D:14:1:
Emails deteled:D:2:2:
```



```
IsaacY_1000_stats.txt x
5
Logins:4:1:
Time online:150.96:23.62:
Emails sent:10:3:
Emails opened:12:5:
Emails deteled:7:2:
```

Alert Engine & Reading Inconsistencies

Following this part of the program, after generating both the log files & statistical file for the user entered,

Program prompts for stats file as such:

Please enter the live data stats file [-1 to quit]:

Enter the new statistical file to generate “**live data** ” to be checked for inconsistencies/anomalies against the baseline statistics for each of the days “live data” is generated for.

EG. Figure 8

```
Please enter the live data stats file [-1 to quit]: IsaacY_1000_stats.txt
Number of days of data to be generated from IsaacY_1000_stats.txt: 10
Generating 10 days of LIVE DATA..
DONE!
=====
```

EG. Figure 9 shows the live data generated and inconsistencies/anomalies against the baseline statistics detected for each of the days generated

```
Day 9
-----
Event : Logins
Event : Time online
Event : Emails sent
Event : Emails opened
Event : Emails deteled
~~~~~
Threshold:      18
Current:       12.96
No anomalies detected.
=====

Day 10
-----
Event : Logins
Event : Time online
Event : Emails sent
Event : Emails opened
Event : Emails deteled
~~~~~
Threshold:      18
Current:       9.32
No anomalies detected.
=====
~~~~ ANOMALIES FOUND: 2 ~~~~
```

Figure 10 shows the `live_data.log` file generated saved in the program's directory



```
live_data.log x
Emails sent:D:9:1:
Emails opened:D:8:1:
Emails deteled:D:5:2:
~~~ Day 6 ~~~
Logins:D:3:3:
Time online:C:164.07:2:
Emails sent:D:5:1:
Emails opened:D:6:1:
Emails deteled:D:6:2:
~~~ Day 7 ~~~
Logins:D:3:3:
Time online:C:152.27:2:
Emails sent:D:14:1:
Emails opened:D:8:1:
Emails deteled:D:8:2:
~~~ Day 8 ~~~
Logins:D:5:3:
Time online:C:153.61:2:
Emails sent:D:10:1:
Emails opened:D:16:1:
Emails deteled:D:8:2:
~~~ Day 9 ~~~
Logins:D:4:3:
Time online:C:138.65:2:
Emails sent:D:12:1:
Emails opened:D:7:1:
Emails deteled:D:10:2:
~~~ Day 10 ~~~
Logins:D:4:3:
Time online:C:159.90:2:
Emails sent:D:8:1:
Emails opened:D:7:1:
Emails deteled:D:7:2:
```