Weekly Report 2.14

This week:

- Database design is changed into a simpler and more useful format.
- Database insertions and updates for Connections and Files tables are implemented.
- Working on SVM, retraining and classification of POP3 SMTP, MSN.
- Database interaction part is finished.

Next week:

- Database Query GUI will be implemented.
- Insertions and updates for the Summary table wil be implemented.
- Help files and installation guide will be prepared.
- Siemens Decoder integration.
- Decision mechanism that combines rule-based and SVM results will be implemented.

Çağla ÇIĞ: This week I spent all of my time working on the database module. The implementation of the database module was crucial to the project and needed to be done keeping in mind efficiency issues since our project is designed to operate real-time. For this purpose the database module works in a seperate thread with less priority than GUI and classification threads. This priority difference is achieved by making the database thread queue its events and dequeue and activate them when the system is idle. My part of the implementation consisted of adding the database feature to SMTP, POP3 and NNTP classification classes. First the connections are added to the database with protocol name set to UNKNOWN and if it matches with one of the protocols we are responsible of detecting, then its protocol name is changed accordingly in the connections table. In addition to this, the pathnames of the files that are transferred over these protocols are stored in the files table. Since the files transferred using YMSG protocol are encrypted, these files cannot be captured and therefore doesn't take place in the files table. Although the database module and our project is nearly finished, next week I am planning to integrate the database module to the GUI and add features like executing pre-defined and custom queries. Also if time allows I am going to start working on the documentation of our project.

Nazif İlker ERÇİN: This week, I spent my time on database module of the project. The database insertions and updates are almost finished, except the summaries of the connections. We have edited our initial database design into simpler and more useful format. We only have 3 tables. The Connections table, different from the previous design, has protocol_name and protocol_class attributes. protocol_name holds wihch connection belongs to which protocol and protocol_class is for recording the main class of a connection, i.e. email, file_transfer etc. protocol_class is planned to be used in some queries like "show me the email connections". When a connection is detected, the system initially records it having "unknown" protocol_name and unknown_class protocol_class. When the connection ends, system checks the match values, selects the maximum match value and edits the protocol_name and protocol_class attributes of the proper record in the database(in case the maximum match value

exceeds %60). Additionally, I edited the user interface and added an Options menu, which has a selection of FastMode. The FastMode can be checked or unchecked. If it is checked, the recognizers work upto some limit value of bytes(binary) or lines(text-based). Next week, I am planning to implement a query GUI which is planned to be appear when user clicks a button or selects a menu item in the main window.I will also finish implementation for the insertions and updates for Summary table of the database. Additionally, I will prepare installation guide and help files of the program.

Can Hoşgör: This week, I continued my work and I finished the core functionality of the database module. I wrote the low-level portions of the code that handles communication with Sqlite. I also made the database module multithreaded. This approach virtually eliminated all performance related issues since all requests to the database module now work asynchronously. The database module uses an internal queue for storing queries requested by other modules. It takes these queries from the queue one by one, and executes the queries them the background, allowing other threads preempt the operation when the database module is busy with lengthy I/O operations. Additionally, I helped Elvan about the SVM module. This module didn't have much work to do, but we worked on an idea suggested by our assistant. We tried to make the SVM module recognize a protocol (MSN) that doesn't have a corresponding rule-based matcher and measure its success rate. We trained it with a small dataset and evaluated the results. Although the success rate is not as high as YMSG protocol, it performed quite well compared to the size of the dataset. Besides that, I did some testing as usual and fixed a number of bugs throughout the project. By this point I can say that the program can be considered fairly stable since it doesn't have any severe bugs and it doesn't cause memory leaks. Next week, I hope that I'm going to finish decoder integration and if time allows, do some more testing for the final demo.

Elvan Gülen: This week, I worked on SVM training and classification. As you know, can was training SVM with all protocols but when we stepped into the classification part we see that SVM doesn't give good results for the protocols. We concluded that this unexpected result occurs because the number of matching protocols in all trained connections remains too few. By considering this, previous week we started to train SVM from the beginning. I and Can trained only for YMSG, on the other hand this week I started to train SVM with SMTP POP3. I needed too many pcap files so I used the ones that Sevgi gave us. For instance for SMTP, SVM doesn't give us a good result after in classification. One of the problem arose from not selecting the right source IP. For finding the right IP, I opened all pcap files with wireshark and noted the sourceIP for the related protocol's connection. It was a time consuming duty. Also I believe that it will be a drawback for the user because there will be too many connections flowing and most of the connections will come from other computers that are connected to that server so the user may not know the right IP. Anyway, another problem is that I believe that there are too many lost data, retransmission and etc.. Since we haven't got a working decoder part, we couldn't solve these problems. So this problem decreases the performance of training and classification. One last problem is that I believe that even if we solve all these problems, the performance of SVM for the text based protocols will remain low because most of the commands are same in these protocols and the unrelated data like a mail can be an important data for SVM. If we train the SVM with sending and receiving the same mail; sending uses SMTP and reading uses POP3 then the mail in both protocols can have negative effects on the SVM model. However, I'll try training these text based protocols after a working decoder. Apart from

this, since the result of SVM is very good for a binary protocol, YMSG, I gave MSN a try. Even if we didn't train with enough data, it gives some good result but I should work on it more. Also for being ready on the demo day, I try to integrate the decoder part in the next week and work on SVM more. Besides since the decision mechanism works separately for SVM and rule based classifiers, I'm planning to combine them with a proper ratio. After these works, I hope that we will be ready for the demo.