Security Design Documentation

<REDACTED>– <REDACTED> – James Ryan – <REDACTED>

University of Victoria

SENG 468

# Database Security Design

The DBMS we chose to use for this project is MySQL. The Python driver provided with MySQL, called mysql-connector provides protection from SQL injection attacks by parameterizing any user inputs which prevents embedded quotes or other special characters from allowing users to input their own SQL commands.

We currently do not protect against DoS or DDoS attacks. In a real-world situation this would need to be a requirement of the database, however we do not have the current knowledge between our group members to have a solution outlined in this document.

# Transaction Server Security Design

For the scope of this project database connection parameters, such as passwords, are stored in the transaction server as a hard-coded variable. This is not a good practice, however, is useful for ease of use during development. In a real-world situation, we would avoid this as much as possible and make sure to prevent this from making it into a production build of the server.

A consideration for if this were a real-world project, would be to modify the legacy quote server so that it can handle end to end encryption with the transaction server. The scope of this project prevented any changes to the quote server, however we do believe that it would be beneficial to prevent any outsiders from viewing the information passed between the transaction servers and quote server.

# User Web Client Security Design

The first and most obvious security design fix would be to install an SSL certificate within the front-end components in order to upgrade both the WebSocket connection and the HTTP connection. With an SSL certificate, the WebSocket connection would be made WebSocket Secure (wss:// instead of ws://) providing much more security as data going through that socket would be encrypted. The same goes for upgrading HTTP to HTTPS, preventing data from being breached over the connection through a man-in-the-middle attack [<https://www.computerworld.com/article/3180690/its-time-to-turn-on-https-the-benefits-are-well-worth-the-effort.html>].  
 ccc “industry standard functionality for asymmetric encryption…”[<https://www.mysql.com/products/enterprise/encryption.html>]. This includes digital signing tools to ensure that the data has not been corrupted in transit (RSA, DSA), and validating data authenticity. All of these would ensure that the database adheres to security standards.

Finally, given that our transaction servers, and the whole system, run in the lab, it is already protected by Uvic’s security. You cannot access the system from outside the lab unless you set up a port forwarding system, which can easily be protected with secret key authentication. In the real world, you would probably do something similar, not allowing access to any part of the system except through trusted channels.