**7ET023 Dissertation Literature Review**

Shar, L (2013) describes the different aspects of SQL Injection such as insecure coding practices, SQL Injection defences, defensive coding, SQL Injection vulnerabilities (SQLIV) detection, runtime SQL Injection attack (SQLIA) prevention and tool support. Why SQLIA is a problem as it is a type of attack that makes most front end security ineffective by bypassing it and gaining access to the database directly in order to perform malicious actions. The source provides a list of several different SQLIA defences which is similar to the work of Dorai, R (2011) as both deal with the issue of database security for example the first source provides details on SQLIA detection software/tools as well as the different features that may be included such as whether vulnerability locating is automatic or manual, tips to modifying code for the better and more which is very useful as the tools it shows may be used in the practical section of this paper, meanwhile the second source discusses how the database is vulnerable to this attack such as with escape characters, types inappropriately dealt with or the database server itself which may become part of the testing of the practical section of this paper, the source also shows several tools for the detection and prevention of SQL Injection such as GreenSQL an open source software for this purpose, due to it being open source it has the advantage in terms of cost over paid software however due to this it will undoubtedly be less effective than the paid software such as perhaps some of the tools outlined in the first source that will impose less of an overhead to the systems overall performance at runtime with query and transaction times, therefore this is an important point to be considered when designing the artefact and so will help answer the research question.

Dorai, R (2011) also discusses SQL Injection but also shows how this relates to database security as a whole by detailing the large amount of attacks on websites that were reported in a certain time frame in order to outline the gravity of the issue. The source discusses several SQLIA methods such as those that take advantage of types and escape characters being handled incorrectly in order to gain information about the system or the records it holds, the database server itself as well as intentionally causing error messages to appear with the injection of SQL code in order to take advantage of this and again access to information they normally would be restricted to this is useful as this may be used as part of a testing suite on the finished artefact for this paper. The source also outlines methods for the prevention of SQLIAs which is similar to the work of Kim, M (2014) both discuss SQL Injection and the complexities the topic brings with it such as how the application is coded and if there are any vulnerabilities that attackers can take advantage of such as error, type mishandling and more. The first source specialises in prevention of SQLIAs with its tool GreenSQL that is essentially a firewall for the database that attempts to block malicious SQL statements which is an improvement on simply detecting attacks however this may cause runtime issues such as queries that are not malicious in nature being blocked impeding a company's productivity as well as the overhead such a tool will impose on the queries, transactions or any action on the database. Meanwhile the second source outlines its design for detecting SQLIAs based on data mining that is broken up into several different phases such as the data collection, pre-processing, training with the use of support vector machins(SVMs) which is a machine learning technique where the tool will learn from the data it is given in order to better perform its task which is the last phase of actual detection of SQLIAs, which is very useful considering the numerous malicious statements that must be learnt an automatic detection system will drastically increase the amount of attacks that are known of and in so doing improve security however this will still be after the attack has taken place instead of stopping it which is why prevention is more useful however both will no doubt impact the design and testing of this papers practical section and in so doing help answer the research question.

Kim, M (2014) discusses SQL Injection and the details of its design for detecting SQLIAs which it shows to be sectioned into several phases that add up to a tool that provides the user with the ability of sensing attacks of this kind in order to learn from them and take steps to secure their system against it. The phases are the collection of data, its pre-processing, and then the training of the tool via support vector machines(SVMs) which is a part of machine learning where the tool is given key data that acts as an anchor for the tool to draw on so that further data given to it is analysed correctly in order to perform its task so in a sense the tool will learn and improve over time without major alterations to its structure, coding etc and from that detect SQLIAs. This is similar to the work of Shar, L (2013) as both sources deal with the issue of SQL Injection by detailing tools to help secure the database against this, the first source focuses on detection which while this is important as it is necessary to learn the vulnerabilities in a system it is not perfect as damage can still be done in order to learn of the problems with a system meanwhile the second source also shows tools for detection but also tools that show how to improve coding in the system to make it less vulnerable as well as tools that take this further and attempt to prevent SQLIAs entirely which if implemented properly would greatly improve database security and as such will be a basis for further research in this paper as well as the possible implementation of one or more tools to test against the artefact however the issue with these is that they are to be used at runtime and may affect performance and come after all the design considerations have been taken and implemented and therefore a better solution may be a hybrid of detection and prevention which will also be a focus of this paper and should help answer the research question.

Pinzón, C (2013) discusses SQL Injection but also provides its solution as an example of how to safeguard against this kind of attack by first detecting it and then prevent similar occurrences from happening with its proposed hybrid tool that comprises a classifier agent using as case based reasoning engine similar in nature to machine learning where the tool learns from past problems which in the case of SQLIAs would be a formidable ally to have due to the myriad methods available for attackers having a tool to automatically learn from attacks will drastically increase security. The tool also includes a visual agent in order to physically see suspicious queries and determine if they are malicious in nature or not which is useful as it allows for the human eye to be a factor in the defence of the database as software can always make mistakes that the human eye will be able to see however this does mean that a company using this tool will need to employ personnel to perform this task which decreases possible performance and its cost effectiveness of an automatic system. This is similar to the work of Kim, M (2014) as both provide solutions to SQLIAs but their methods and the techniques involved are different as while the first source has its hybrid tool of detection and prevention the second source focuses solely on detection and also uses SVMs instead of cased based reasoning which due to its nature if implemented properly with the right data for the task may outperform cased based reasoning in terms of precision due to the amount of data that can be provided for it giving the second sources tool a strong platform to learn and deal with the issue of SQLIAs and therefore both sources tools will be considered when designing the artefact and may play a part in the practical stage of this paper in order to answer the research question.