8. CONCLUSION

The ability to download information and to submit data without the constraint of location and time made Internet a boom. This is made possible with trillions of Web applications developed for every unique purpose. These applications come with many security issues and vulnerabilities. The vulnerabilities in these applications are exploited by hackers for personal information gathering and material benefits, which make fixing web application flaws inevitable and the need for the hour. The focus of this thesis was on improving the security of the Web by implementing strategies that explore the vulnerabilities in the Web applications and mitigate them.

8.1 SUMMARY

The four strategies, namely Data Cleansing Algorithm, Amendment capture Algorithm, Permission Enforcer Algorithm, and the Inference Rules are innards of this thesis which sought to identify and alleviate certain Web application vulnerabilities.

Data cleansing algorithm facilitated mitigating the input validation attacks. The SQL Injection and cross site scripting attacks were addressed by the input data cleansing algorithm. Four test bed applications were taken as input and a cheat sheet was used as an input to these applications. The algorithm was able to detect 100% of the vulnerable input submitted to the applications. This system with intrusion prevention proxy proved to be effective in detecting the SQL injection attacks and primitive XSS attacks and preventing these attacks from penetrating the web application. In SQL injection, attacks caused using tautologies, union queries and piggy back queries could be detected and mitigated. The source code of the application was not changed and the system was fully automated. A server side solution was provided and hence user intervention was not needed in mitigation of these attacks. The security check imposed an increase in response time by
3.49% which is not very significant in an Internet scenario. This could be handled by increasing the number of proxy servers, such that the web application could handle any number of requests without increase in latent time and still can protect the application from SQL injection and XSS attack. The order of complexity of the algorithm was \( O(1) \) which made the system simple to implement and execute, when compared with the other standard systems whose order of complexity is \( O(3) \).

The evolution of Smartphones caused the migration of application from the desktop environment to Smartphones. A variety of platforms for Smartphone were evolved which led to different attacking scenarios leading to new types of vulnerabilities. This thesis underscored the importance of behavior monitoring and updates check, to mitigate personal information theft in Smartphones. A system named amendment capture service, which implemented the Amendment Capture Algorithm, was developed. Fifty applications were used to train and test the system using features monitoring. The system achieved an accuracy of 94%. The update trigger and the behavior pattern trigger were tested using 18 applications and the accuracy was found to be 93.5%. A malicious application exhibiting a genuine feature measure was not identified, which was the limitation of this system. In future work a rectification for this issue will be addressed. This system deals with update monitoring, which is a unique feature that has not been addressed in other standard counterparts.

Another major issue with the smartphone operating system is the excess permissions requested by the applications. Permission enforcer algorithm was proposed and implemented to handle the attack caused by excess permissions. 1194 applications were downloaded from play store and these applications were categorized into 7 categories. The permissions were segregated into white list, black list and red list for each category. Both system defined permission and developer defined permissions were considered. The evaluation of the system was done using case studies. The accuracy of the system was found to be 100%.
WebViews in Smartphone platform provide great feature for the development of hybrid-web based applications. But they also pose a great threat as they create the portal for launching malicious attacks on Smartphones and web pages. An automated static analysis system, for scanning WebViews vulnerable to attacks, was developed and implemented. Five attacks were considered. In addition, a new attack named, the Supplementary Event-Listener Injection attack, was explored. This attack could be launched through JavaScript Injection mechanism. The system provided a detailed report on the WebView that was reviewed. The system classified malicious applications with an accuracy of 85%. Attacks caused through WebView are critical as they result in loss of sensitive private information from web pages and hence causing insecurity to users.

By providing an overall defense for major security issues in Web application, in both the desktop and smartphone environments, this thesis has contributed its mite to enhance the security of the Web as a whole.

8.2 SUGGESTIONS FOR FUTURE WORK

Security issues in Smartphone applications are a great concerns due to the ever-growing nature of the Web technology. In particular, for Android systems, most of these issues involve the permission feature. Therefore, there are more research avenues such as, providing flexibility in selecting permissions from provided list of permissions for a given application. It would certainly restrict the functionality of the application.

Smartphone applications performing different functionalities, tend to exhibit different behavior. Applications can be categorized based on their functionalities. Since each category performs more or less the similar functions, the behavior of the applications within a particular category can be analyzed. Taking this into consideration, the behavioral analysis described in chapter 5 could be enhanced by performing a behavioral analysis based on application category.
Vulnerability analysis at runtime can unearth certain issues that were not discoverable during static analysis, thus improving the efficiency of the system. The system that statically analyzed the WebView to detect vulnerabilities could be expanded to include dynamic analysis of the application for discovering such vulnerable holes and for effectively mitigating the attacks caused by WebView.

Apart from the above mentioned issues, intents also pose a great security threat. Intents are the components that are used for inter process communications among the applications. There are two types of intents, namely implicit intents and explicit intents. The Permission Enforcer Algorithm described in chapter 6 could be enhanced to include the analysis of the flow of control using these intents and the coordination between the intents used in the application and permissions requested for those applications.