CISC 499 Projects

Contact: Prof. Mohammad Zulkernine (mzulker@cs.queensu.ca)

(Joint Work with Marwa Elsayed and Shahrear Iqbal)

Project 1: An Execution Zone Aware CPU Scheduler

Linux uses "Completely Fair Scheduler (CFS)" implemented by Ingo Molnar as the default CPU scheduler. It replaced the previous O(1) vanilla scheduler and merged in Kernel 2.6.23. The CFS scheduler models an "ideal, precise multi-tasking CPU" on real hardware. The CFS maintains the amount of time a task has been permitted to execute on the CPU. It uses a time-ordered red-black tree to maintain fairness in providing processor time to tasks. CFS does not use priorities directly but instead uses them as a decay factor for the time a task is permitted to execute. Lower-priority tasks have higher factors of decay, where higher-priority tasks have lower factors of delay. In this project, the student will modify the existing CFS algorithm to accommodate execution zones with different priorities. An executions zone enforces constraints on the applications reside in that zone. Notably, processes may be moved from one zone to another dynamically based on their behavior. We can call it "Zone Aware Completely Fair Scheduler".

Project 2: Knowing Malware from the Web

Web Mining is the technique used to extract useful information from the internet. Nowadays, a massive amount of important information can be found over the internet in the form of natural language. For example, often malware are reported in blogs and news sites before any signature available for them in anti-virus software. In this project, the student will use existing data mining techniques to mine websites and make a program that can indicate whether an application is malicious according to user opinions/news found on the Internet. This operating system can use this information to restrict an application's behavior until the anti-virus database is updated.

Project 3: A Comparison of Code Analysis Tools for Cloud Applications

By design, cloud SaaS application development relies mostly on existing web and internet technologies. Following the service-oriented architecture (SOA), applications can be composed as a service from other services. RESTful web services are the most preferred way of exposing SaaS (e.g., Google, Amazon, Yahoo, Facebook, and Twitter). Code vulnerabilities in SaaS application open a front end universally accessible from the internet. Successful exploits of such vulnerabilities can lead to breach the integrity and confidentiality of sensitive data. In this project, the student will develop a security benchmark suite as cloud applications that utilize RESTful web services written in JAVA. The suite should contain benign and vulnerable applications. This benchmark will be used for looking for security vulnerabilities using static or dynamic analysis

tools (e.g., Soot, YASCA, Indus, and Lapse Plus). The benchmark should demonstrate vulnerabilities like NoSQL injection with its many vectors JSON, JavaScript, scheme, and view - injection, cross-site scripting (XSS), and information leakage to unauthorized parties. The application should also reflect inter-app communication, lifecycle callback, implicit and explicit information flows, and other dynamic features. Then, the student will conduct experiments to compare between static and dynamic analysis tools to detect vulnerabilities in the developed benchmark suite.