Project Part 2 – Mid-Term Project Report
Christopher Zimmer
3/31/2009

Some of the goals that needed to be accomplished to fulfill this work are presented below.

Week 1
In this work I have reviewed literature that applies to security in the general purpose environment. Since this work primarily deals in code injection I reviewed such concepts as stack shielding, address stack layout randomization, and compiler based canary values. This approach is most similar to compiler based canary values as it will utilize intra task instrumentation to best facilitate the needs.

Week 2
The Simulator environment I am working in has been modified with a new system call that allows the scheduler to set up a periodic timer interrupt. This is a programmable interrupt made from the scheduler. My current scheduler is currently being interrupted every 2 us in simulation time. The frequency of the scheduler is clocked at 1 Ghz so that's roughly equivalent to every 2000 cycles. I feel that interrupts every 2000 cycles will be a good starting point in evaluation to determine cost and trade-off regarding overhead of this project vs detecting timing anomalies.

Week 3 - Current
I have modified the scheduler and the simulator to support new system calls that enable the scheduler to determine the last running PC of the last task in a single processor environment. A test real-time task set has been configured using two simple clab benchmarks CNT and LMS. The tasks have been analyzed in both a regular timing analyzer tool set and a parametric timing analyzer tool set.

Remaining Challenges
1. The aim of this work was to attempt to create a security check that was completely orthogonal to the running task. But to do this in a practical manner on a uni-processor system it is going to be necessary to relax that constraint until this work can be moved into a multi-core system. For now I am working implementing a system of in-loop counters that the scheduler can reference quietly during the periodic security interrupts to determine the amount of times back-edges have been utilized for a set of instruction addresses. Utilizing these numbers it will be possible to utilize the cost of preceding code segments, and parametric loop equations to determine if the current program is operating within normal boundaries. My expectation is that the instrumentation will be finished tomorrow.
Expected Finish end of this week.

From there the final challenge left for the next week is to design a partitioning scheme for the output data from the timing analyzer tool set for which to compare the scheduler security values against.

2. Two ideas to solve for this is to use the verbose timing metrics received from the timing analyzer to create an actual timing end value for the each of the instructions. Though this method will only work for the sequential code and may require significant memory overhead.

This will then be coupled with parametric timing values for PC's that come from loop bodies. The structure of the loops are known and the secure scheduler will have to utilize this to determine nesting and probable timing values for a given PC.
The other approach I'm going to attempt to utilize is a partitioned approach in which the last PC is stored between periods and I'm going to insure using timing data and the known period and frequency of the processor to determine if the current PC value is in a valid range for the amount of cycles executed between the periods. For example, the period is every two microseconds or 2000 cycles between periods at this frequency. If there are no preemptions between these two times, if the PC I land on in the timing tree has 20 instructions left in its partition and 10 cycles left for the WCET of that partition (assuming partitions are centered around basic blocks) we have just detected an attack. The inverse works as well using the known period. If the current PC's BCET is 4000 cycles away from the last PC and there were no preemptions, we may have detected a timing anomaly.
Expected Finish end of next week 4/10

The remainder from this point will be one half composing task sets using the benchmarks and testing this approach by itself and then in conjunction with two other approaches created prior to this project in this realm of work. The other half will be performing the write up of the results and approach into a report.
Expected Finish 4/21