Ryan Wilson - Master's Thesis Defense - Experiments Overview Map

Enhanced Debugging Capabilities

1. Virtual Breakpoints:
This experiment demonstrates the concept of a virtual breakpoint, which helps to build the foundation of this thesis. Virtual breakpoints are set and triggered from a kernel driver on MSR access, Port I/O access and VMX instruction access.

2. Nearly Unlimited Breakpoints:
This experiment proves that more than four virtual breakpoints can be simultaneously set. Ten virtual breakpoints are set on port I/O addresses. Port I/O was used because it is comparable to a standard hardware breakpoint.

3. VMM Control over Guest OS:
This experiment shows that the VMM has control over the guest OS. The guest system state is saved into a control structure on a VM-exit, which can be viewed and modified by the VMM. This experiment demonstrates the VMM's ability to view and directly modify the guest's instruction pointer (EIP register).

Smart Filtering of Data During Dynamic Analysis

4. Processor Mode Switch:
This experiment demonstrates the ability to tailor the virtual breakpoint concept to create a breakpoint that traps on a processor mode switch, such as from protected mode to real mode.

5. Model Specific Register (MSR):
The MSR experiment tailors a virtual breakpoint to trap on access to a specified MSR. This experiment uses two scenarios to demonstrate use cases. One scenario addresses cache types of memory ranges specified through entries in the page table. The other scenario addresses setting an interrupt to occur when a thermal limit is reached. Both scenarios show how a specific MSR could be miscalculated, leading to run-time errors at a much later point in execution. It would be difficult to track where the corrupted MSR's values were written in a complex OS. The tailored use of a virtual breakpoint could quickly identify exactly where the problems occurred, decreasing the time it takes to debug the issue.

Analysis of Complex Debug Environments

6. Virtual Breakpoint on Hardware Breakpoint:
Virtual breakpoints can be configured to trap on processor exceptions. This type of virtual breakpoint does not make modifications to the guest such as hooking an interrupt. A standard hardware breakpoint causes exception 0x01 to occur, allowing a virtual breakpoint to be used to help debug standard debugging capabilities.

7. Virtual Breakpoint on Errors:
A virtual breakpoint can be used to trap on errors that cause processor exceptions, such as a divide by zero error, an undefined op-code exception, a general protection fault, or a page fault. This experiment demonstrate a virtual breakpoint that traps on a divide by zero error. This allows the user to analyze the system state before the error occurs, and to identify the location of the error.