Jeremy Holloway
CPSC-3220-001
6/6/2019

Pre-class Assignment #14

1. Define the following terms:
   - TLB: A small hardware table containing the results of recent address translations.
   - superpage: A set of contiguous pages in physical memory that map a continuous region of virtual memory, where the pages are aligned so that they share the same high-order(superpage) address.
   - tagged TLB: A translation lookaside buffer whose entries contain a process ID; only entries for the currently running process are used during translation. This allows TLB entries for a process to remain in the TLB when the process is switched out.
   - TLB shootdown: A request to another processor to remove a newly invalid TLB entry.

2. Name a reason why you would need to do a TLB flush.
   To perform a context switch, the flush will change the hardware page table register to point to the new process’s page table by discarding the contents of the previous TLB.

3. Consider the following two approaches to zero matrix "a". Assume that matrix a is stored in row-major order, that a double-precision value is 8 bytes, that the page size is 4 KiB, that the i and j index variables are register-allocated rather than memory allocated, and that you have a traditional TLB with 8 entries. For each approach, compute the number of TLB misses, assuming that for each approach the TLB starts off as empty.

```
double a[1024][1024];
// approach 1 - ij loop ordering
for( i = 0; i < 1024; i++ ){
        for( j = 0; j < 1024; j++ ){
                a[i][j] = 0.0;
        }
}
// approach 2 - ji loop ordering
for( j = 0; j < 1024; j++ ){
        for( i = 0; i < 1024; i++ ){
                a[i][j] = 0.0;
        }
}
```

Approach 1 is 2048 misses. It has row major ordered sequence and each row makes up 8KiB. The size of a double is 8B. The TLB entries map 4KiB pages. If my access is I,J row oriented access 2048 TLB misses.

Approach 2 is 1,048,576 misses. The distance between column entires is 8KiB so the the stride is significantly longer between each access. This would result in every access being a miss. If my access is J,I column oriented access

4. Explain the purpose of the attack described in section 8.4.2 of jumping into the middle of an x86 instruction. Would checking a table of valid branch targets as part of simulating an indirect branch prevent this?

The attacker might be trying to access protected information or skip permissions checks for load and stores. As long as we can protect the table from being modified the author suggests using a table with only valid entry points into the code.