Shane Harvey 108272239

CSE 409 Interim Report

Pin for Stack Smashing

We would like to implement stack-smashing detection. Pin allows for a very nice way to examine instructions and registers (what we want) and allows insertion of arbitrary C/C++ code (again what we want). Our basic strategy is to save the return instruction pointer on all call instructions and then check that this value is the same when we encounter a return (ret) instruction. From our experience so far we believe this is possible by using the function INS\_IsCall() and then passing the IARG\_RETURN\_IP as a parameter to a function that appends this value to a list. Then when we encounter a ret, using INS\_IsRet(), we can read the return address off the stack and check if it is the same as the last saved return address. We are currently testing if using IARG\_RETURN\_IP in this context is incorrect, we might want to change it to the current instruction pointer because that is what the call instruction saves on the stack.

We were able to make a Pin tool that could display the return\_ip’s at each call instruction(956 for a simple hello world c program) and the value at the stack by examining the value at REG\_ESP just before the ret. Our problem is that many of the IAR\_RETURN\_IP’s are zero and it looks like the values aren’t matching up the way we think they should. This is probably because of the different ways you can ret in x86, eg ret, retn, retf, ret with an immediate value.