Tutorial on PIN tool

S R Swamy

Indian Institute of Technology Madras

srswamy@cse.iitm.ac.in

January 22, 2018

What is PIN?

- PIN: A tool for instrumentation of programs
- Supports Linux, OS X, Windows operating systems.
- Inserts arbitrary code in arbitrary places in the executables.
- PIN is similar to JIT compiler.

PIN Granularity

Three main levels of granularity:

- Routine
- Instruction
- Image

Along with the above levels, Pin also has one obivious level - trace granularity.
A trace is a straight-line instruction sequence with exactly one entry. It usually ends with an unconditional branch.
Examples: Calls, returns and unconditional jumps.

PIN tool applications

- Memory leaks: Memory allocated, but not freed.
- Double freeing: Memory deallocated more than once.
- Freeing unallocated memory: Deallocating memory that hasn't been allocated.
- Profiling

KNOB: Commandline switching

Knobs automate the parsing and management of command line switches.
Exmaple: KNOB<string> KnobOutputFile(KNOB_MODE_WRITEONCE, "pintool", "o", "inscount.out", "specify output file name");

Controlling and Initializing

This group of functions is used to initialize Pin, start the application, and a call backs for events like application exit.

- PIN_Init (INT32 argc, CHAR **argv)
- PIN_InitSymbols ()
- PIN_AddFiniFunction (FINI_CALLBACK fun, VOID *val)
- PIN_StartProgram (PIN_CONFIGURATION_INFO options=PIN_CreateDefaultConfigurationInfo())
- PIN_AddThreadFiniFunction (THREAD_FINI_CALLBACK fun, VOID *val)
- PIN_AddThreadStartFunction (THREAD_START_CALLBACK fun, VOID *val)

Instrumentation Functions

- INS_AddInstrumentFunction (INS_INSTRUMENT_CALLBACK fun, VOID *val)
- INS_InsertPredicatedCall (INS ins, IPOINT ipoint, AFUNPTR funptr,...)
- INS_InsertCall (INS ins, IPOINT action, AFUNPTR funptr,...)
- INS_InsertIfPredicatedCall (INS ins, IPOINT action, AFUNPTR funptr,...)
- INS_InsertThenPredicatedCall (INS ins, IPOINT action, AFUNPTR funptr,...)

Instrumentation arguments

All argument lists must end with IARG_END.

- **IARG_INST_PTR** : The address of the instrumented instruction.
- **IARG_REG_VALUE** : Value of the register.
- **IARG_MEMORYREAD_EA** : Effective address of memory read.
- **IARG_MEMORYWRITE_EA** : Effective address of memory write.
- **IARG_MEMORYREAD_SIZE** and **IARG_MEMORYWRITE_SIZE**
- **IARG_MEMORYOP_EA**: Effective address of a memory operand

IPOINT

Determines where the analysis call is inserted relative to the instrumented object.

- IPOINT_BEFORE
- IPOINT_AFTER
- IPOINT_ANYWHERE
- IPOINT_TAKEN_BRANCH