

Pattern	Description
ASSIGN	An assignment statement, <i>e.g.</i> $r_0 = r_0 + r_1$;
ASSIGN_CONSTANT	An assignment statement including a constant, <i>e.g.</i> $r_0 = r_0 + 0x01$;
CONTROL	A control statement where the target of the jump is unknown, <i>e.g.</i> $\text{if } (zf == 1) \text{ JMP } [r_0 + r_1 + 0x10]$;
CONTROL_CONSTANT	A control statement where the target of the jump is known. <i>e.g.</i> $\text{if } (zf == 1) \text{ JMP } 0x400567$;
CALL	A call statement where the target of the call is unknown, <i>e.g.</i> $\text{CALL } r_5$;
CALL_CONSTANT	A call statement where the target of the call is known, <i>e.g.</i> $\text{CALL } 0x603248$;
FLAG	A statement including a flag, <i>e.g.</i> $cf = 1$;
FLAG_STACK	A statement including flag register with stack, <i>e.g.</i> $eflags = [sp = sp - 0x1]$;
HALT	A halt statement, <i>e.g.</i> HALT ;
JUMP	A jump statement where the target of the jump is unknown, <i>e.g.</i> $\text{JMP } [r_0 + r_5 + 0x10]$;
JUMP_CONSTANT	A jump statement where the target of the jump is known, <i>e.g.</i> $\text{JMP } 0x680376$;
JUMP_STACK	A return jump, <i>e.g.</i> $\text{JMP } [sp = sp - 0x8]$;
LIBCALL	A library call, <i>e.g.</i> $\text{compare}(r_0, r_5)$;
LIBCALL_CONSTANT	A library call including a constant, <i>e.g.</i> $\text{compare}(r_0, 0x10)$;
LOCK	A lock statement, <i>e.g.</i> lock ;
STACK	A stack statement, <i>e.g.</i> $r_0 = [sp = sp - 0x1]$;
STACK_CONSTANT	A stack statement including a constant, <i>e.g.</i> $[sp = sp + 0x1] = 0x432516$;
TEST	A test statement, <i>e.g.</i> r_0 and r_5 ;
TEST_CONSTANT	A test statement including a constant, <i>e.g.</i> r_0 and $0x10$;
UNKNOWN	Any unknown assembly instruction that cannot be translated.
NOTDEFINED	The default pattern, <i>e.g.</i> all the new statements when created are assigned this default value.