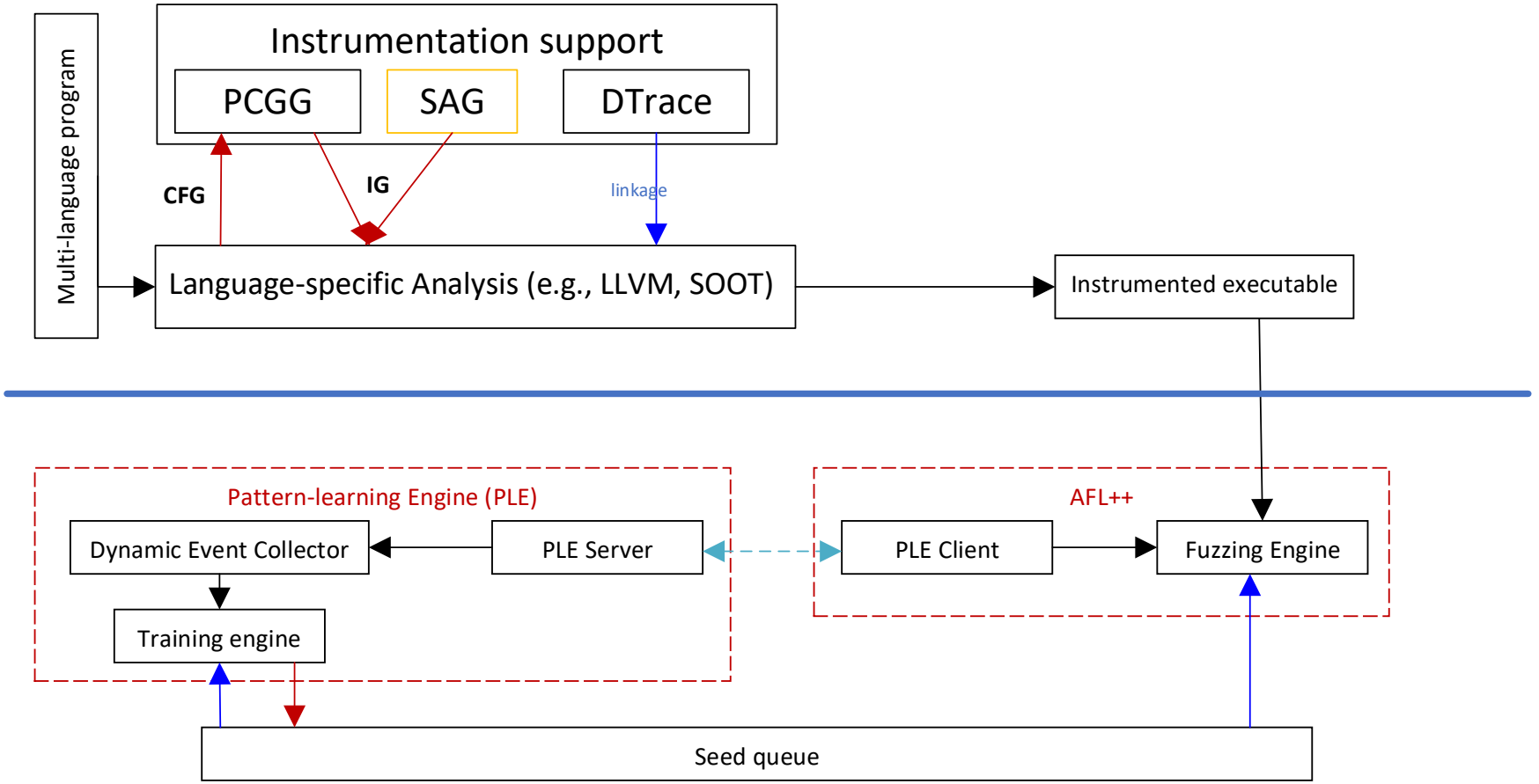

Multi-language program
Instrumentation support
PCGG
SAG
DTrace
CFG
IG
linkage
Language-specific Analysis (e.g., LLVM, SOOT)
Instrumented executable
Pattern-learning Engine (PLE)
Dynamic Event Collector
PLE Server
Training engine
AFL++
PLE Client
Fuzzing Engine
Seed queue

| Problems: | Constraints: |
| --- | --- |
| 1. path coverage guarantee<br>2. sensitive analysis guarantee | 1. language extensibility<br>2. performance (minimize the instrumentation) |

**P1**

1. get the block-level CFG of a function, encode each CFG node as a block ID
2. the CFG is language-independent, satisfying the constraint 1.
3. apply the instrumentation algorithm on the CFG to compute PCG guidance, satisfying the constraint 2.

**P2**

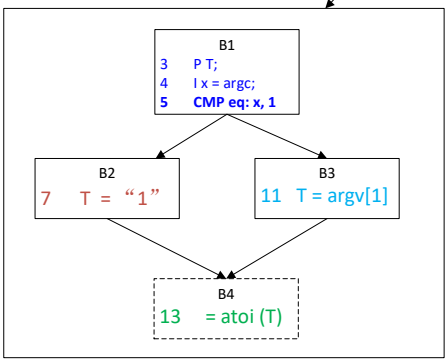
1. define a language-independent IR to retain the necessary information for SAG, satisfying the constraint 1.
2. extend the block-level CFG to instruction (or statement)-level CFG: each node represents a statement in IR
3. Def-Use computation on the use of each CMP and SWITCH node of the CFG: get the definition statement for each branch variables as SAG
4. merge SAG and PCGG, satisfying the constraint 2.

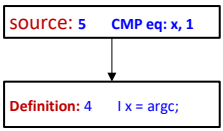
```
1 int main (int argc, char* argv[])
2 {
3     char *T;
4     int x = argc;
5     if (x == 1)
6     {
7         T = “1” ;
8     }
9     else
10    {
11        T = argv[1];
12    }
13    return atoi (T);
14}
```

IR

```
1 I main (I argc, P argv[])
2 {
3     P T;
4     I x = argc;
5     CMP eq: x, 1
6     {
7         T = “1”
8     }
9     else
10    {
11        T = argv[1]
12    }
13    = atoi (T)
14}
```

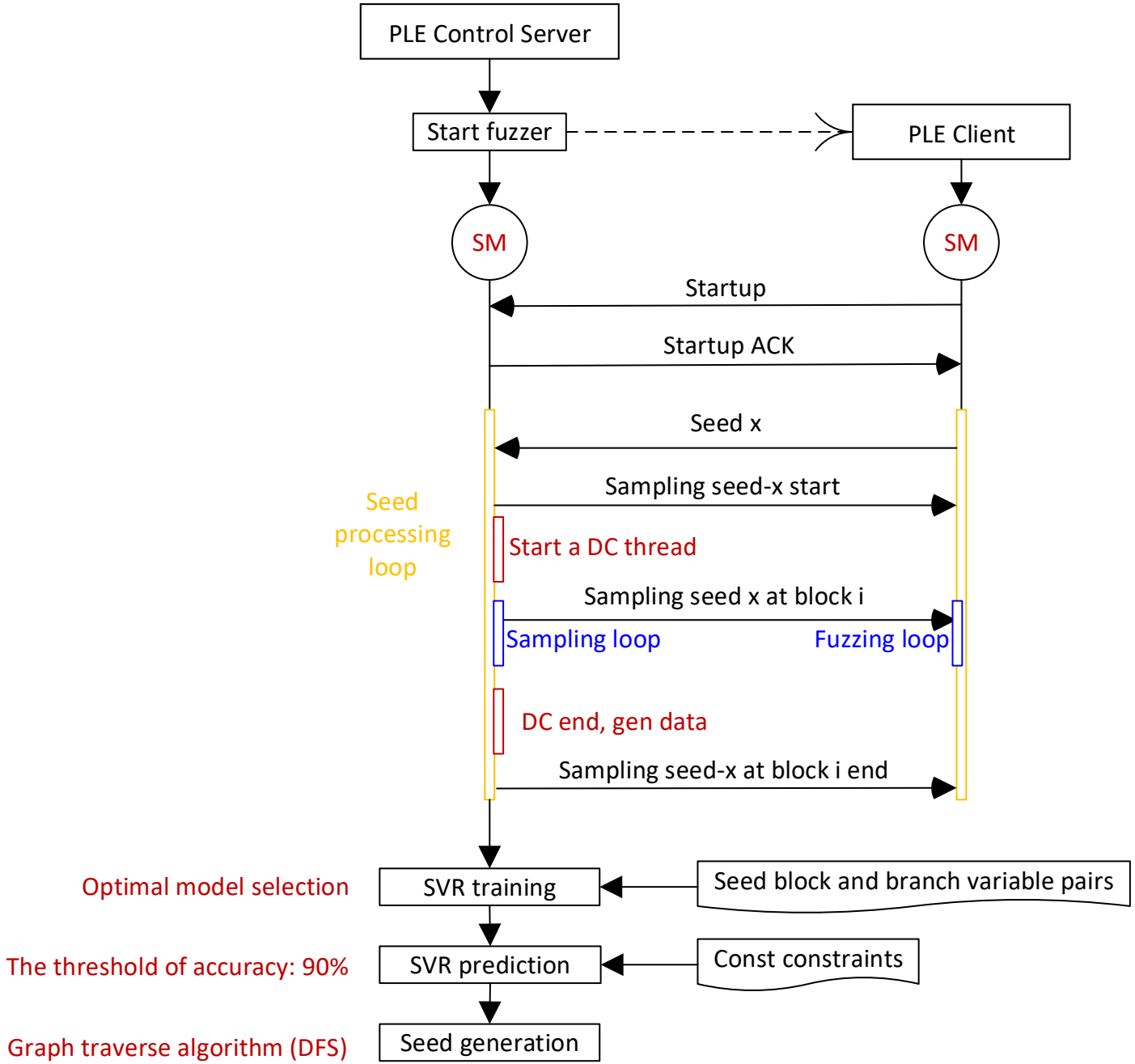
CFG

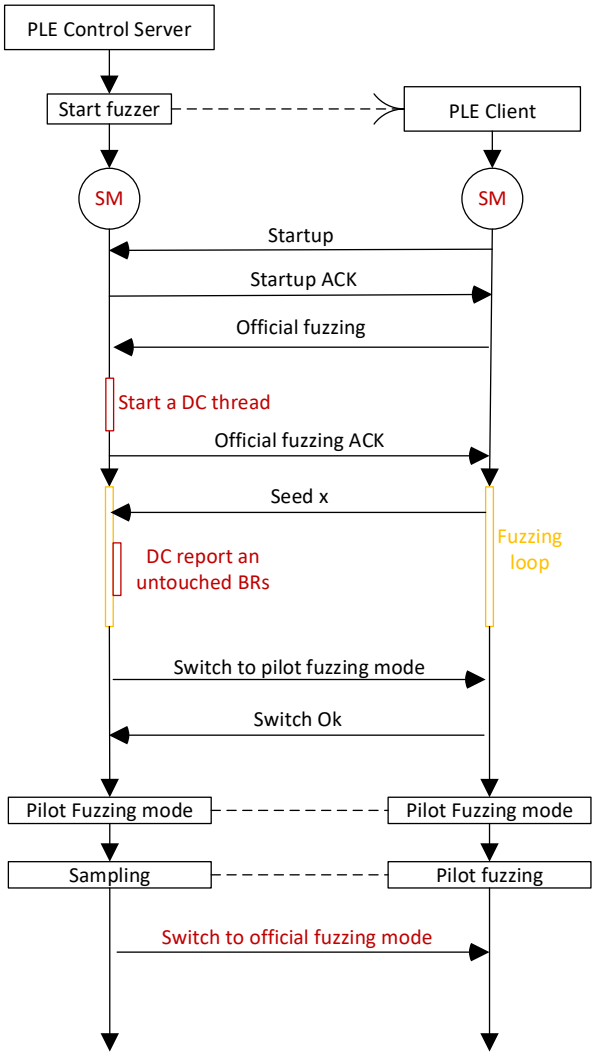


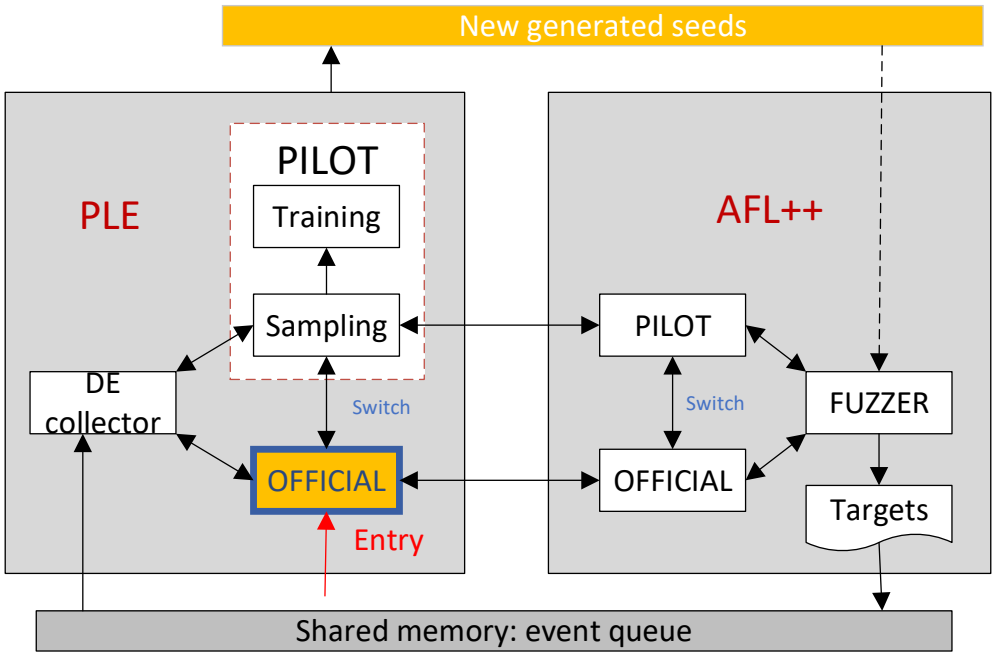




PCGG: B1, B2, B3
SAG: B1

=> PCGG & SAG:
B1, B2, B3


PLE Control Server
Start fuzzer
PLE Client
SM
SM
Startup
Startup ACK
Seed x
Sampling seed-x start
Seed processing loop
Start a DC thread
Sampling seed x at block i
Sampling loop
Fuzzing loop
DC end, gen data
Sampling seed-x at block i end
Optimal model selection
SVR training
Seed block and branch variable pairs
The threshold of accuracy: 90%
SVR prediction
Const constraints
Graph traverse algorithm (DFS)
Seed generation


PLE Control Server
Start fuzzer
PLE Client
SM
SM
Startup
Startup ACK
Official fuzzing
Start a DC thread
Official fuzzing ACK
Seed x
Fuzzing loop
DC report an untouched BRs
Switch to pilot fuzzing mode
Switch Ok
Pilot Fuzzing mode
Pilot Fuzzing mode
Sampling
Pilot fuzzing
Switch to official fuzzing mode


New generated seeds
PLE
PILOT
Training
Sampling
DE collector
Switch
OFFICIAL
Entry
AFL++
PILOT
Switch
OFFICIAL
FUZZER
Targets
Shared memory: event queue