(a) Binary Code

Addr	Value	Disassembly 1
804a634	74	JZ 0x804a64c (A)
804a635	17	
...		...
804a646	7b	JNP 0x804a64c (B)
804a647	05	
...		
804a64c	4b	... (C)
804a64d	25	
...		

Ctrl Flow Pattern:
$addr_1$: $instr_1$
...
$addr_2$: $instr_2$
...
$addr_3$: $instr_3$

Source Pattern:
```
if (...)    //instr1
   if (...) //instr2
     s1;
s2;         //instr3
```

(a) Hint I: Control Flow Convergence

(b) Binary Code

Addr	Value	Disassembly 1
804a634	74	JZ 0x804a648 (1)
804a635	12	
...		...
804a646	eb	JMP 0x804a628 (2)
804a647	e0	
804a648	e5	... (3)
...	4b	

Ctrl Flow Pattern:
$addr_1$: $instr_1$
...
$addr_2$: $instr_2$
$addr_2+|instr_2|$: $instr_3$

Source Code:
```
while (...) { //instr1
  ...
    s1;       //instr2
}
s2;           //instr3
```

(b) Hint II: Control Flow Crossing

Jump Table
```
40040d: 00 00     ADD  al, (rax)
40040f: 00 08     ADD  cl, (rax)
400411: 10 60 00  ADC  ah,0x0(rax)
         ...
```

String
```
400370: 00 5f 5f  ADD  bl,0x5f(rdi)
400373: 67 6d     INSL (dx), es:(edi)
400375: 6f        OUTSL ds:(rsi), (dx)
400376: 6e        OUTSB ds:(rsi),(dx)
```

Instructions
```
4005cb: 48 89 c2     MOV rax, rdx
4005ce: 48 03 55 f8  ADD -0x8(rbp), rdx
4005d2: 8b 45 f4      MOV -0xc(rbp), eax
4005d5: 89 02        MOV eax,(rdx)
```
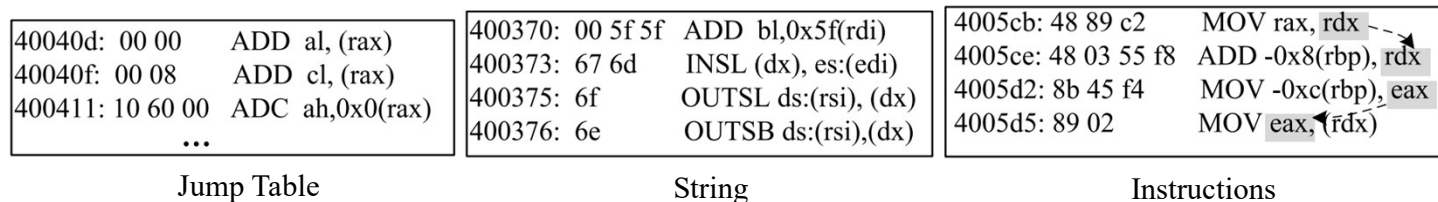
(c) Hint III: Register Definition-use Relation