Binary Context-Sensitive Recognizer (BCSR)

Hong Pham
December 4, 2007

Motivation

- Virus Signatures
  - Database of hexadecimals
  - Definitive registers
    - Alter the registers and it is another signature

Register Manipulation

- Different registers give different signatures

```
add   $4   %eax        add   $4   %eax
mov   $5   %ebx        mov   $5   %ecx
sub   %eax %ebx        sub   %eax %ecx
      (a)                    (b)
```

Figure 1

BCSR

- Program generator to recognize context-sensitive binary signatures
- General representation of signatures, not dependent on registers
- The signatures are specified by the user in the source specification

Source Specification

{definitions}
%%
{rules}
%%
{user subroutines}

- A signature
  - Binary signature
  - actions
- Variable construct
  - [name, size, values]
  - Global / Local
- Ambiguous source rules

Example

```
89 c3:    mov   %eax, %ebx
FF c3:    inc   %ebx
75 f2:    jne   58942345
```

Example

```
89 c3:      mov    %eax, %ebx
FF c3:      inc    %ebx
75 f2:      jne    58942345
```

```
            mov    %eax, $1
            inc    $1
            jne    $2
```

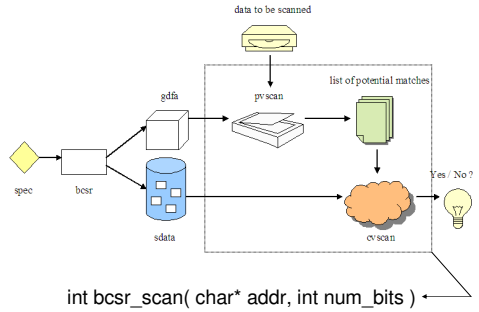
Example

```
89 c3:      mov    %eax, %ebx
FF c3:      inc    %ebx
75 f2:      jne    58942345
```

```
            mov    %eax, $1
            inc    $1
            jne    $2
```

```
%%
            1000 1001 1100 0 [a, 3, *]
            1111 1111 1100 0 [a]
            0111 0101 [b, 8, *] {}
```

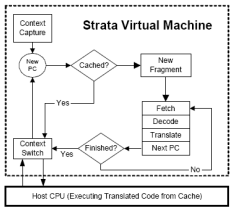
BCSR Process



int bcsr_scan(char* addr, int num_bits)

Strata

- Software dynamic translator
- Fragment creation
  - Conditional or indirect control transfer
  - trampoline



Experiments

- Protocol
  - Scanning Strata fragments
  - Spec Int Benchmarks
  - Red Hat Linux
  - X86_64
- Statistics
  - Overhead

Results

- ???

Issue 1

- Specs are too general !!!

Issue 1

- Specs are too general !!!
- Signature

```
pop    %eax
push   %ecx
add    %eax, %ebx
add    %ecx, %eax
push   %ecx
```

Issue 1

- Specs are too general !!!
- Signature

```
pop    %eax
push   %ecx
add    %eax, %ebx
add    %ecx, %eax
push   %ecx
```

```
pop    $1
push   $2
add    $1, %ebx
add    $2, $1
push   $2
```

Issue 1

- Specs are too general !!!
- Signature

```
pop    %eax
push   %ecx
add    %eax, %ebx
add    %ecx, %eax
push   %ecx
```

```
pop    $1
push   $2
add    $1, %ebx
add    $2, $1
push   $2

pop    %eax
push   %eax
add    %eax, %ebx
add    %eax, %eax
push   %eax
```

Issue 1

- Specs are too general !!!
- Signature

```
pop    %eax
push   %ecx
add    %eax, %ebx
add    %ecx, %eax
push   %ecx
```

- False positives

```
pop    $1
push   $2
add    $1, %ebx
add    $2, $1
push   $2

pop    %eax
push   %eax
add    %eax, %ebx
add    %eax, %eax
push   %eax
```

Issue 2

- Multiple fragments

Issue 2

- Multiple fragments
- Signatures contains the following:
  - Conditional or indirect control transfers

Issue 2

- Multiple fragments
- Signatures contains the following:
  - Conditional or indirect control transfers
- False negatives

Future Work

- Address Issue 1 and 2
- Extend the language
  - Star, functionality, …
- Symbolic code
  - Write in assembly rather than binary

Questions??