CISPA
HELMHOLTZ CENTER FOR
INFORMATION SECURITY

Instruction Pattern Search

Metodi Mitkov

28.03.2022

Installation and Dependencies

- Python 3.7 and above
- Clone GitHub repository – https://github.com/metodi022/instrsearch
- Create a python virtual environment
- Install angr (framework for analysing binaries)

Usage

- Arguments

```
-p PATH          path to binary file
-s SEARCH        search pattern
```

- Optional Arguments

```
-b BASE          base address of binary in hex
-a ARCH          architecture of binary
-o OUTPUT        CSV output location
-v               verbose mode; prints output in console
-d               debug mode; prints additional information
```

Simple Example 1

PowerShell 7 (x64)

```
PS D:\Programming\instrsearch> python.exe .\instrsearch.py -p .\example\binaries\toy1 -s "\ADDR: cmp \ANY, \ANY" -v
WARNING | 2022-03-27 17:04:34,378 | cle.loader | The main binary is a position-independent executable. It is being loade
d with a base address of 0x400000.
0x401050 deregister_tm_clones
0x40105e: cmp rax, rdi

0x401140 __libc_csu_init
0x401191: cmp rbp, rbx

PS D:\Programming\instrsearch>
```

angr will print warnings and errors during the analysis process

Simple Example 2

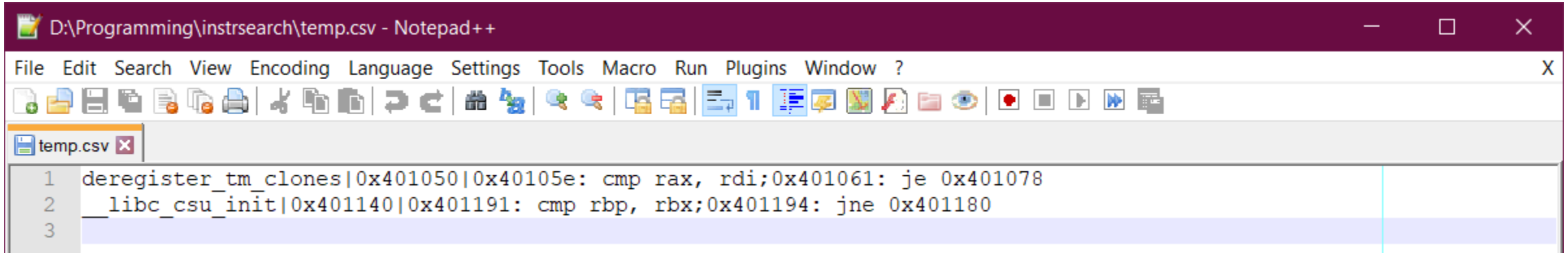
PowerShell 7 (x64)

```
PS D:\Programming\instrsearch> python.exe .\instrsearch.py -p .\example\binaries\toy1 -s "\ADDR: cmp \ANY, \ANY; \ADDR:
\ANYINS" -d -o "./temp.csv"
[*] Preparing cache
[*] Parsing with:
        [+] (0x[a-fA-F0-9]+): cmp ([^\s\r\n,]+), ([^\s\r\n,]+)
        [+] (0x[a-fA-F0-9]+): ([^\n\r]+)
[*] Loading angr project 2022-03-27 17:10:58.615250
WARNING | 2022-03-27 17:10:58,646 | cle.loader | The main binary is a position-independent executable. It is being loade
d with a base address of 0x400000.
[*] Loaded example\binaries\toy1, AMD64 Iend_LE
[*] Entry object <ELF Object toy1, maps [0x400000:0x40402f]>; entry address 0x401020
[*] CFGFast analysis initiated 2022-03-27 17:10:58.646531
[*] Search initiated 2022-03-27 17:10:58.646531
[*] Closing files 2022-03-27 17:10:58.662150
PS D:\Programming\instrsearch>
```

CSV Output Format

- **|** separator for CSV values
- **;** separator for instructions

- Function Name **|** Function Address **|** instruction **;** instruction



Search Pattern

- Any valid python RegEx expression `[a-zA-Z0-9]{2,6}(1234)+`
- Each instruction begins with `\ADDR: cmp rax, rcx`
- Multiple instructions can be chained `\ADDR: cmp rax, rcx; \ADDR: jg 0x1234`

- Shortcuts

\GP	matches any general purpose register		
\IMM	matches any immediate value		
\ADDR	matches any address	**\DEREF**	matches any dereference
\AVX	matches any AVX register	**\ANY**	matches any mnemonic or operand
\ANYINS	matches any instruction		

Extending Search Pattern

- Shortcuts can be easily extended in the code

```
pattern_dict: Dict[str, str] = {
    "\\ANYINS": "([^\\n\\r]+)",
    "\\ANY": "([^\\s\\r\\n,]+)",
    "\\ADDR": "(0x[a-fA-F0-9]+)",
    "\\IMM": "([0-9]+)",
    "\\GP": "(([re]?[abcd][xhl])|(r[01234589]{1,2}[dwb]?)|([re]?(si|di|bp|sp)l?))",
    "\\DEREF": "(((word|dword|qword) ptr )?\\[[^\\]+\\])",
    "\\AVX": "([xyzXYZ]?(MM|mm)[0-9][0-5]?)"
}
```

Example Searches

- `\ADDR: cmp \GP, \ANY; \ADDR: \ANYINS`

- `\ADDR: add eax, eax; (\ADDR: \ANYINS){2,4}; \ADDR: sub eax, ecx`

Performance – Ubuntu Focal Kernel (~17MB)

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

- Initial run of a binary
  - First analysis ~15 minutes
  - Search ~2 minutes

- Subsequent runs of the **same binary**
  - Reload cached analysis ~1 minute
  - Search ~2minutes

1) Improved runtime by **caching** analysed functions on first run.
2) Quick integrity check with **md5**.

Limitations

- angr **disassembly** not perfect – disassembly errors resulting in blocks or functions not disassembled

- Basic block granularity

- No **dynamic analysis**

```
\ADDR: in \ANY, 0x3d
```

✅ in rax, 0x3d

❌ mov rax, 0x3d

❌ in rcx, rax