angr

Unknown macro: 'html'

List

angr

Description

- Angr Python .
- Angr , Symbolic(Concolic) .
- Angr .
  - Angr VEX IR, SimuVex IR .
    - PyVEX(VEX IR) IR .
    - IR SimuVex .

Github

- https://github.com/angr/pyvex
- https://github.com/angr/simuvex

API

API	Description
Angr	• **API .** ○ ○ ○ ○ ○ ○ (VSA)
claripy	• Solver Engine ○ . ○ z3 .

cle	• Binary Loader
	  ○
	  ○

pyvex	• Binary Translator ○ VEX (IR)
archinfo	• Arch Information Repository • .

Github

- https://github.com/angr/angr
- https://github.com/angr/claripy
- https://github.com/angr/cle
- https://github.com/angr/pyvex
- https://github.com/angr/archinfo

Install

Command

```
lazenca0x0@ubuntu:~$ sudo apt-get install python-dev libffi-dev build-essential virtualenvwrapper

...  ...

(angr) lazenca0x0@ubuntu:~$ deactivate
lazenca0x0@ubuntu:~/Documents/angr$ workon angr
(angr) lazenca0x0@ubuntu:~/Documents/angr$
```

Execute and Exit

```
lazenca0x0@ubuntu:~/Documents/angr$ workon angr
(angr) lazenca0x0@ubuntu:~$ deactivate
lazenca0x0@ubuntu:~/Documents/angr$
```

- https://docs.angr.io/INSTALL.html

Example

Defcamp CTF Quals 2015 - r100

File

- r100

Source code

- .
  - fgets() .
    - 255.
  - sub_4006FD() , return .

- : "Incorrect password!"
- : "Nice!"

main()

```
signed __int64 __fastcall main(__int64 a1, char **a2, char **a3)
{
  signed __int64 result; // rax@3
  __int64 v4; // rcx@6
  char s; // [rsp+0h] [rbp-110h]@1
  __int64 v6; // [rsp+108h] [rbp-8h]@1

  v6 = *MK_FP(__FS__, 40LL);
  printf("Enter the password: ", a2, a3);
  if ( fgets(&s, 255, stdin) )
  {
    if ( (unsigned int)sub_4006FD(&s, 255LL) )
    {
      puts("Incorrect password!");
      result = 1LL;
    }
    else
    {
      puts("Nice!");
      result = 0LL;
    }
  }
  else
  {
    result = 0LL;
  }
  v4 = *MK_FP(__FS__, 40LL) ^ v6;
  return result;
}
```

Find address of function

- **angr .**
  - angr Symbolic execution Password .
  - 0x400864 jmp .
    - Password : 0x400855
    - Password : 0x400844

disassemble main

```
gdb-peda$ x/36i 0x4007E8
   0x4007e8:        push   rbp
   0x4007e9:        mov    rbp,rsp
   0x4007ec:        sub    rsp,0x110
   0x4007f3:        mov    rax,QWORD PTR fs:0x28
   0x4007fc:        mov    QWORD PTR [rbp-0x8],rax
   0x400800:        xor    eax,eax
   0x400802:        mov    edi,0x400937
   0x400807:        mov    eax,0x0
   0x40080c:        call   0x4005c0 <printf@plt>
   0x400811:        mov    rdx,QWORD PTR [rip+0x200850]        # 0x601068 <stdin>
   0x400818:        lea    rax,[rbp-0x110]
   0x40081f:        mov    esi,0xff
   0x400824:        mov    rdi,rax
   0x400827:        call   0x4005e0 <fgets@plt>
   0x40082c:        test   rax,rax
   0x40082f:        je     0x400866
   0x400831:        lea    rax,[rbp-0x110]
   0x400838:        mov    rdi,rax
   0x40083b:        call   0x4006fd
   0x400840:        test   eax,eax
   0x400842:        jne    0x400855
   0x400844:        mov    edi,0x40094c
   0x400849:        call   0x4005a0 <puts@plt>
   0x40084e:        mov    eax,0x0
   0x400853:        jmp    0x40086b
   0x400855:        mov    edi,0x400952
   0x40085a:        call   0x4005a0 <puts@plt>
   0x40085f:        mov    eax,0x1
   0x400864:        jmp    0x40086b
   0x400866:        mov    eax,0x0
   0x40086b:        mov    rcx,QWORD PTR [rbp-0x8]
   0x40086f:        xor    rcx,QWORD PTR fs:0x28
   0x400878:        je     0x40087f
   0x40087a:        call   0x4005b0 <__stack_chk_fail@plt>
   0x40087f:        leave
   0x400880:        ret
gdb-peda$
```

Source code

- .
  - "r100" angr.Project() API Project .
  - path_group() API Path group .
  - explore() API .
    - : find=0x400844
    - : avoid=0x400855
  - explore() API 0x400844 .

poc.py

```
import os
import angr

project = angr.Project("r100", auto_load_libs=False)
path_group = project.factory.path_group()
path_group.use_technique(angr.exploration_techniques.DFS())
avoid_addr = [0x400855]
find_addr = 0x400844

path_group.explore(find=find_addr, avoid=avoid_addr)
print path_group.found[0]
print path_group.found[0].state.posix.dumps(0)
```

- .

poc2.py

```
import os
import angr

project = angr.Project("defcamp_quals_2015_r100", auto_load_libs=False)
path_group = project.factory.path_group()
path_group.explore(find=lambda path: 'Nice!' in path.state.posix.dumps(1))
print path_group.found[0].state.posix.dumps(0)
```

- project.factory.path_group() : http://angr.io/api-doc/angr.html#angr.path_group.PathGroup
- path_group.explore() : http://angr.io/api-doc/angr.html#angr.path_group.PathGroup.explore

Result

- password .

Command

```
(angr) lazenca0x0@ubuntu:~/Documents/angr$ python symbolicECE.py
WARNING | 2017-09-06 23:27:40,112 | claripy | Claripy is setting the recursion limit to 15000. If Python
segfaults, I am sorry.
Code_Talkers

(angr) lazenca0x0@ubuntu:~/Documents/angr$ ./r100
Enter the password: Code_Talkers
Nice!
(angr) lazenca0x0@ubuntu:~/Documents/angr$
```

angr with it

- **Tool angr .**

List

Tool		github
angrop	gadget chain .	https://github.com/salls/angrop
Patcherex	.	https://github.com/shellphish/patcherex
rex	Exploit .	https://github.com/shellphish/rex
Driller	angr AFL .	https://github.com/shellphish/driller

Relation site

- https://angr.io
- http://angr.io/api-doc/
- https://github.com/angr/angr
- https://github.com/angr/angr-doc
- https://github.com/axt/angr-utils

Unknown macro: 'html'