May the 18th - Fuzzing project report

Goal (12th May - 18th May)🏆

- **Modification** of the `imagecountbranches.cpp` script to do the different between unique or not branches
- **Create** a MakeFile for the compilation of our Pin script
- **Create** a GitHub to start to organize the project
- **Automate** the processus of collecting applications, inputs and all we have need to run the experimentation
- **Check** if you binary use the **address sanitizer approach** to detect some vulnerabilities such as **buffer over-read/write** (abort the program when you accessing not allowed part) **WARNING** : PinTool can be not compatible with some applications which use address sanitizer
- **Run** my `imagecountbranches` script on one of the application of fuzzer-test-suite locally to see some results
- **Start** to think about how to implement a call graph in your script to have another metric about the bug depth (how many function are call?) (CALL instruction)

What I have done ? 👨‍💻

`imagecountbranches.cpp` modification

I have done some modifications ont this Pin script.
Firstly, we have to do a difference between unique branche and the others.
Secondly, the "execution" number of a branche (to detect a loop for example)
Finally, we have 2 numbers :

- the number of unique conditional branches
- the number of conditional branches (total)

The script:

Script on GitHub

Makefile

The Makefile:

```
CC=g++
CXXFLAGS= -Wall -Werror -Wno-unknown-pragmas -D__PIN__=1 -DPIN_CRT=1 -fno-stack-protector -fn
LDFLAGS= -shared -Wl,--hash-style=sysv ../pin/intel64/runtime/pincrt/crtbegin
OBJDIR=build

"${OBJDIR}/imagecountbranches.so": "${OBJDIR}/imagecountbranches.o"
	$(CC) -shared -Wl,--hash-style=sysv ../pin/intel64/runtime/pincrt/crtbeginS.o -Wl,-Bs

"${OBJDIR}/imagecountbranches.o": imagecountbranches.cpp
	mkdir ${OBJDIR}
	$(CC) ${CXXFLAGS} -c $< -o $@

.PHONY: clean
clean:
	rm -rf ${OBJDIR}
```

Makefile GitHub

GitHub

Creation

GitHub repository

I have 2 branches to separate the finish work and the experimental work:

- `main` branch (finish): main
- `dev` branch (experimental): dev

Automate collection

The goal is to collect all we have need for the experimentation step (Google Cloud VM).
I choose to use a bash script to do this.
The bash script is composed of different parts:

- Clone **Fuzzer-test-suite** and **Pin**
- Run some others bash script such as `build.sh` (build and compile the binary)
- Run **Makefile** to compile our **Pin script**
- Run pin with the **Pin script** on the **binary** with its **inputs** (**error**)

The script:

```
#!/bin/bash

shopt -s nullglob

if [ -z "${1}" ]
then
	echo  "[+] Error: Please enter an application name"
	exit 1
fi

# config to have all we have want
wget https://software.intel.com/sites/landingpage/pintool/downloads/pin-3.18-98332-gaebd7b1e6
tar -xzf pin-3.18-98332-gaebd7b1e6-gcc-linux.tar.gz && mv pin-3.18-98332-gaebd7b1e6-gcc-linux
git clone https://github.com/google/fuzzer-test-suite.git

# compile, build and run
DIR=$(find . -type d -iname \*"${1}"\*)

if [ -z "${DIR}" ]
then
	echo  "[+] Error: Unknown application"
	exit 1
fi

NAME=$(echo "${DIR}" | cut -d "/" -f3)

cd scripts && make
cd .. && mkdir run_eval && cd run_eval && mkdir "${NAME}" && cd  "${NAME}"
../../fuzzer-test-suite/"${NAME}"/build.sh

for  input  in  $(find ../../fuzzer-test-suite/"${NAME}" -type f ! -name "*.*" | cut -d "/" -
do
	../../pin/pin -t ../../scripts/build/imagecountbranches.so -o ../../results/"${NAME}"
done
```

Script on GitHub

Check address sanitizer compiler option

The bash script of `libxml2-v2.9.2` use the `fsanitize=address` option to compile the binary. The binary is also in 64 bits. My tutor said that we could have some problems with Pin because it's not handle perfectly the address sanitizer approach.

For now, only with `libxml2-v2.9.2` and with its 4 inputs, I have only one problem with the `leak-bdbb2857b7a086f003db1c418e1d124181341fb1` input.

I will ask how to handle it.

Test and results on `libxml2-v2.9.2` locally

Resources

Fuzzer-test-suite : benchmarks results of binaries with its inputs

Intel: PinTool : Dynamic Binary Instrumentation

Run

After running `eval-automatic.sh` script on `libxml2-v2.9.2` , we have some results. You can see them here: GitHub results

Call graph

Resources

3 papers:
SimSight: A Virtual Machine Based Dynamic Call-Graph Generator

Trin-Trin: Who's Calling? A Pin-Based Dynamic Call Graph Extraction Framework | SpringerLink

(PDF) A Tool Suite for Simulation Based Analysis of Memory Access Behavior

The beginning

I read the 3 papers. I learned a lot about them. I have learned methods to do a call graph (dynamic). Different steps we can occur: Parser, Tracer, Analyzer. I had also some difficulties to understand some parts.

I started to write a Pin script to count the depth of a bug but with the functions (call/ret) and not branches (first method). We can insert instrumentation when we have a CALL insctruction but also a RET instruction (to avoid the "wide" problem). When I have a CALL instruction, incremented 1. When I have a RET instruction, decremented 1.

The part of the script:

```
// This function is called before every branch is executed

VOID  CallCount(ADDRINT  addr, BOOL  IsCall)
{
	if(CheckBounds(addr)) {
		if (IsCall) {
			callcount++;

		} else {
			callcount--;
		}
	}
}

VOID  Instruction(INS  ins, VOID  *v)
{
	if (INS_IsCall(ins)) {
		INS_InsertCall(ins, IPOINT_BEFORE, (AFUNPTR)CallCount, IARG_INST_PTR, IARG_BO

	} else  if (INS_IsRet(ins)) {
		INS_InsertCall(ins, IPOINT_BEFORE, (AFUNPTR)CallCount, IARG_INST_PTR, IARG_BO

	}
}
```

Where I stopped ? 🛑

I have to continue optimized the call graph script (avoid call instruction looping.
I have to fix the problem on the bash script (for loop for inputs)