

다양한 바이너리 디핑 도구의 분석과 비교

2024.08.31

서재완, 이창선

Contents

PART **1** 발표자 소개

PART **2** 디핑 기술

PART **3** 기존 디핑 도구

PART **4** 개선된 디핑 도구

PART

1

발표자 소개



서재완

친구 사진 동영상

소개

직장

 현대자동차에서 근무
2024년 - 현재

 한국정보기술연구원 'Best of the Best' 8기 취약점 분석 트랙
에서 근무했음
2019년 7월 1일~2020년 3월

대학

 고려대학교 Korea University에서 정보보호학과 전공
2023년 졸업

- **BoB 8기** 취약점 분석
- 고려대학교 정보보호대학원
- 현대자동차



키다리 보안 아저씨

좋아요 1개 · 팔로워 2명

- BoB 멘토
- 키다리 보안 아저씨 멘토
- 취약점에 대한 분석 의지가 투철
- 혼자 만들고 지우고 사용하고 지우는 타입
- 배우는거, 연구, 발표, 가르치는 걸 좋아함
- 해당 주제는 키보아 2기 프로젝트로 진행
- 키보아 많.관.부!!!

by crattack

<https://www.facebook.com/secleg>

PART

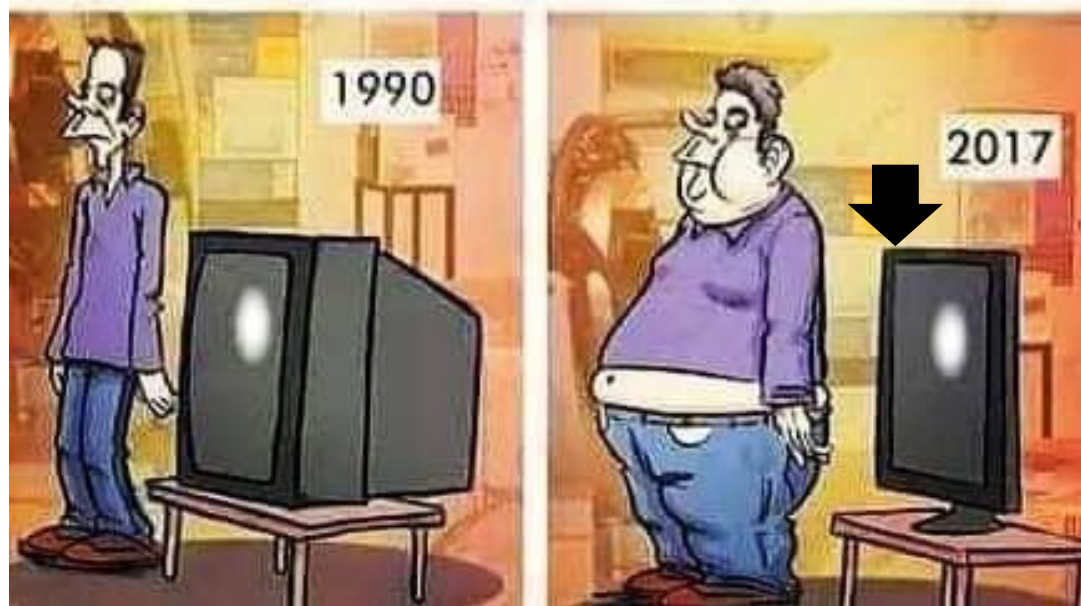
2

디핑 기술

• 디핑 기술이란?

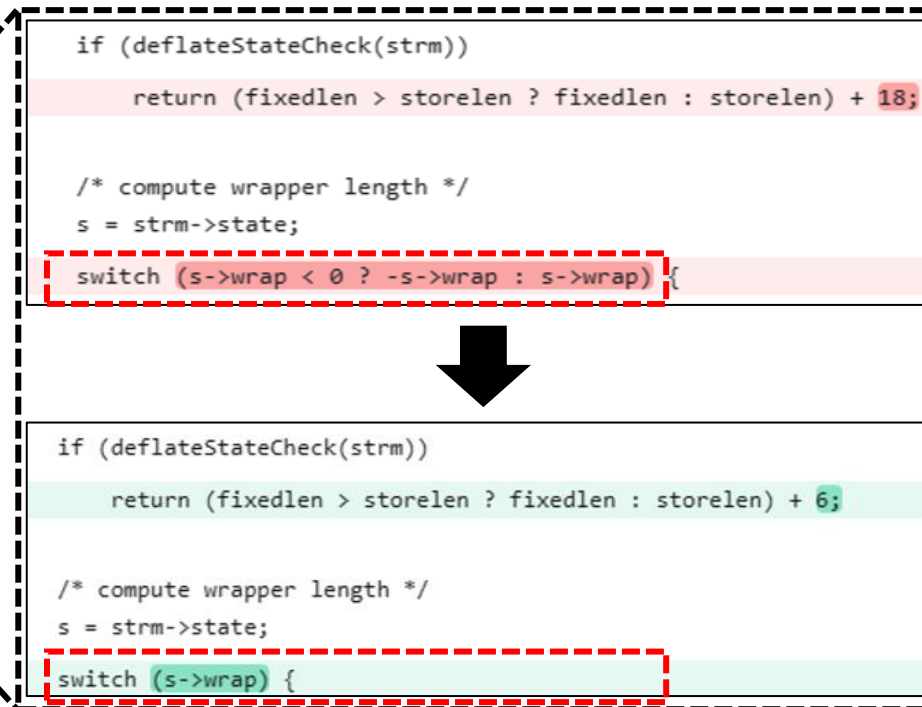
- 구 버전과 신 버전 파일 간의 코드 상 차이를 찾는 기술
- 바이너리 디핑 시 구 버전과 신 버전을 비교하는 방식으로는 File Hash 비교, 코드 비교와 같은 방식이 존재함

1990 - 2017



• 디핑 기술이란?

- 구 버전과 신 버전 파일 간의 코드 상 차이를 찾는 기술
- 바이너리 디핑 시 구 버전과 신 버전을 비교하는 방식으로는 File Hash 비교, 코드 비교와 같은 방식이 존재함





• File Hash 비교 방식

- 바이너리에 대한 Hash 값을 생성하여 구 버전과 신 버전의 차이를 비교
- Hash 값만을 기반으로 차이를 비교하기 때문에 실제로 어떤 부분의 코드가 어떻게 수정되었는지 알기 어려움

```
0016.htm; ; 70201212152236253411947113420912719267057
0020.htm; ; 58247184151241183135640228212621023214012
0021.htm; ; 16566542461561934218236126251169117141147
0022.htm; ; 23260751354925421016913318313522895414919
0030.htm; ; 53203759623218228151575111617114153113121
0040.htm; ; 20197901911305717214214390234792484816122
0045.htm; ; 46205187236102620015810530143779238190190
0050.htm; ; 10221014516819215415717169163232291962061
0060.htm; ; 18502724150602511555318310349117220142150
0070.htm; ; 21022215214161792616312254161292526232569
0075.htm; ; 90182182217221178572456714414999142151571
0080.htm; ; 11751751886517652163711025193441112471682
0082.htm; ; 17865209169228213114209104321448118652599
0090.htm; ; 96241121982514172296168219238204184224234
0110.htm; ; 61222028577150912552052082171831241331244
0115.htm; ; 77209113199981742407836911524965715014515
0116.htm; ; 46149691951832011291581961271922411302262
```

Hash 예제

Name	Current MD5
 test.exe	BCAB5CC93A1FB92A9C5ECBED5220D845
 test2.exe	A9C5924063A253F64FB86BC924BE6996

실제 테스트 결과

• 코드 비교 방식

- 바이너리의 pseudo code를 식별한 후, 구 버전과 신 버전의 차이점을 비교
- 대부분의 바이너리 디핑 도구들은 File Hash 비교 방식보다 코드 비교 방식이 더 정확하게 차이점을 식별하기 때문에 코드 비교 방식을 사용하고 있음

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7      int a;
8      int b;
9      cin >> a >> b;
10     int sum = a+b;
11     cout << sum << endl;
12     return 0;
13 }
14
```

→ **int mul = a*b;**
→ **cout << mul**



```
848 /* if can't get parameters, return larger bound plus a wrapper */
849 if (deflateStateCheck(strm))
850     return (fixedlen > storelen ? fixedlen : storelen) + 18;
851
852 /* compute wrapper length */
853 s = strm->state;
854 switch (s->wrap < 0 ? !s->wrap : s->wrap) {
855     case 0: /* raw deflate */
856         wraplen = 0;
857         break;
858     case 1: /* zlib wrapper */
859         wraplen = 6 + (s->strstart > 4 ? 0);
860         break;
861     #ifdef GZIP
862     case 2: /* gzip wrapper */
863         wraplen = 18;
864         if (s->gzhead != Z_NULL) { /* user-supplied gzip header */
865             Bytef *str;
866             if (s->gzhead->extra != Z_NULL)
867                 wraplen += 2 + s->gzhead->extra_len;
868             str = s->gzhead->name;
869             if (str != Z_NULL)
870                 do {
871                     wraplen++;
872                 } while (*str++);
873             str = s->gzhead->comment;
874             if (str != Z_NULL)
875                 do {
876                     wraplen++;
877                 } while (*str++);
878             if (s->gzhead->hcrc)
879                 wraplen += 2;
880         }
881         break;
882     #endif
883     default: /* for compiler happiness */
884         wraplen = 18;
885 }
886 /* if not default parameters, return one of the conservative bounds */
```

```
848 /* if can't get parameters, return larger bound plus a zlib wrapper */
849 if (deflateStateCheck(strm))
850     return (fixedlen > storelen ? fixedlen : storelen) + 6;
851
852 /* compute wrapper length */
853 s = strm->state;
854 switch (s->wrap) {
855     case 0: /* raw deflate */
856         wraplen = 0;
857         break;
858     case 1: /* zlib wrapper */
859         wraplen = 6 + (s->strstart > 4 ? 0);
860         break;
861     #ifdef GZIP
862     case 2: /* gzip wrapper */
863         wraplen = 18;
864         if (s->gzhead != Z_NULL) { /* user-supplied gzip header */
865             Bytef *str;
866             if (s->gzhead->extra != Z_NULL)
867                 wraplen += 2 + s->gzhead->extra_len;
868             str = s->gzhead->name;
869             if (str != Z_NULL)
870                 do {
871                     wraplen++;
872                 } while (*str++);
873             str = s->gzhead->comment;
874             if (str != Z_NULL)
875                 do {
876                     wraplen++;
877                 } while (*str++);
878             if (s->gzhead->hcrc)
879                 wraplen += 2;
880         }
881         break;
882     #endif
883     default: /* for compiler happiness */
884         wraplen = 6;
885 }
886 /* if not default parameters, return one of the conservative bounds */
```

PART

3

기존 디핑 도구

• Radare2 (<https://github.com/radareorg/radare2>)

- 다양한 플랫폼과 아키텍처를 포함하여 바이너리를 해석, 디버깅 및 분석하는 기능을 제공함
- 결과가 텍스트 형태로 제공되기 때문에 가독성이 부족함

The screenshot displays the Radare2 application interface with several panels open:

- Registers (dr):** Lists registers and their values, such as `rax = 0x00000000`, `rbx = 0x00000000`, etc.
- Breakpoints:** A list of breakpoints with addresses and comments.
- Disassembly (pd):** Shows assembly instructions with their addresses and comments, including cross-references (XREFs).
- Functions (af1):** Lists identified functions with their addresses and names.
- Entropy Fir:** A bar chart showing entropy values across different sections of the binary.
- Hexdump (xc \$r*16):** Displays raw hexadecimal data from memory.
- agfb:** A graph visualization showing control flow or data flow.

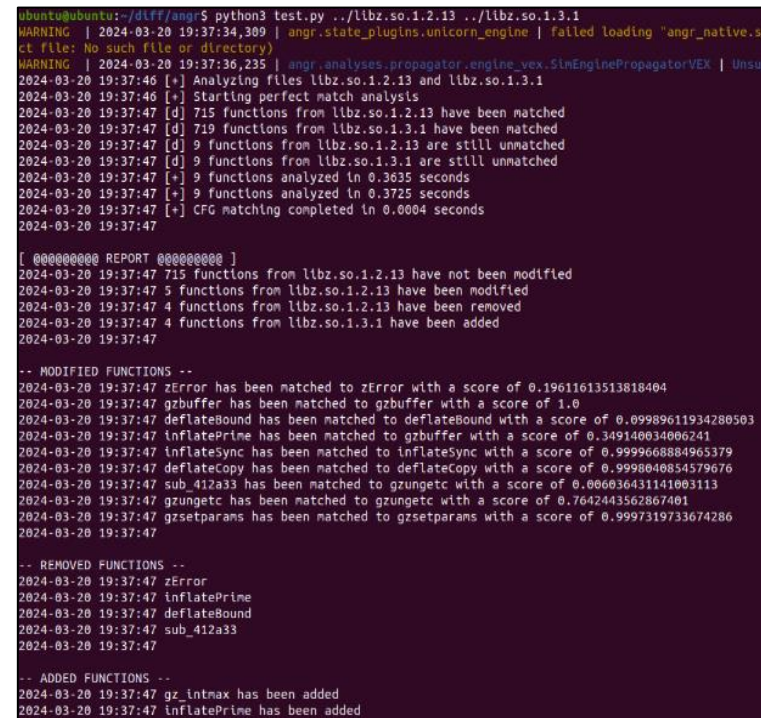
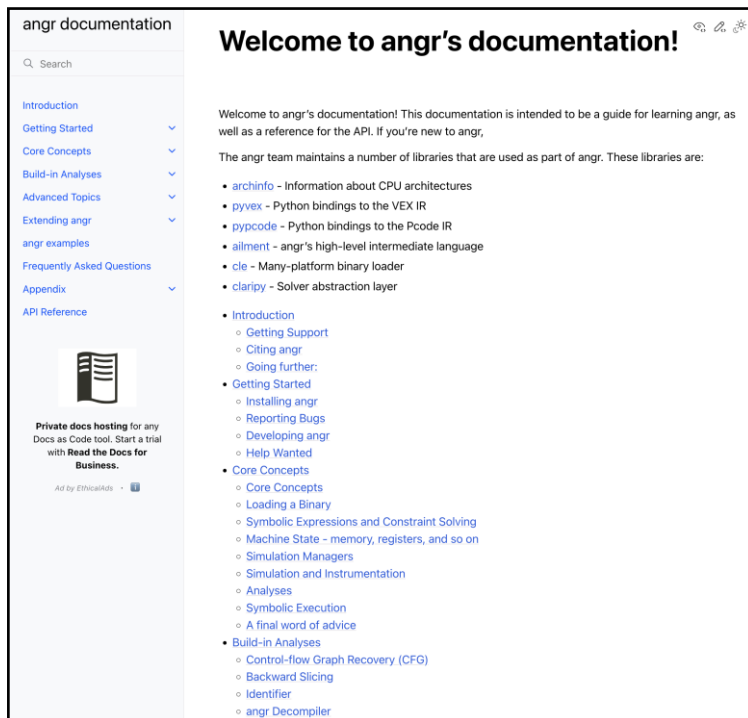
- Radare2 (<https://github.com/radareorg/radare2>)

- 다양한 플랫폼과 아키텍처를 포함하여 바이너리를 해석, 디버깅 및 분석하는 기능을 제공함
- 결과가 텍스트 형태로 제공되기 때문에 가독성이 부족함

```
ubuntu@ubuntu-virtual-machine:~$ radiff2 -AC phpListAdminAuthentication_vul.php phpListAdminAuthentication_patch.php
INFO: Analyze all flags starting with sym. and entry0 (aa)
INFO: Analyze imports (af@@@i)
WARN: Basic block at 0x00001af0 is too large
INFO: Analyze symbols (af@@@s)
INFO: Recovering variables
INFO: Analyze all functions arguments/locals (afva@@@F)
INFO: Analyze all flags starting with sym. and entry0 (aa)
INFO: Analyze imports (af@@@i)
INFO: Analyze symbols (af@@@s)
INFO: Recovering variables
INFO: Analyze all functions arguments/locals (afva@@@F)
fcn.00000000 105202 0x0 |      NEW  (0.000000)
fcn.00000000  2172 0x0 |      NEW  (0.000000)
```

- angr (<https://angr.io/>)

- 바이너리의 Symbolic execution을 지원하여 버그 발견, 리버스 엔지니어링 등 프로그램을 분석하는 기능을 제공함
- 결과가 텍스트 형태로 제공되기 때문에 가독성이 부족함



• elf_diff (https://github.com/noseglasses/elf_diff)

- ELF 바이너리 간 차이점을 찾거나, 패치 또는 업데이트가 적용된 바이너리의 변경사항을 분석하는 기능을 제공함
- x64, x86 플랫폼에 대한 분석만 지원하기 때문에 다양한 플랫폼을 분석할 수 없음

Multi Page

elf_diff binary comparison

Old binary: ./elf_diff_linux/testing/libelf_diff_test_debug_05.4
New binary: ./elf_diff_linux/testing/libelf_diff_test_release_new.4

- Statistics
- Symbols
 - Persisting
 - Disappeared
 - New
 - Similar

Persisting Symbols Overview

Symbol	Type	Size/Bytes	Old	New	Delta/Bytes
persisting@1000	D	T	18	18	0
persisting@1001	D	T	18	18	0
xyz	D	D	6	6	0

Columns

Symbol: The symbol name (possibly mangled)

Type: The symbol type (see the documentation of binutils tool nm for more information)

Old Size: The old symbol size either in RAM or program memory

New Size: The new symbol size either in RAM or program memory

Delta: The change in symbol size

Persisting symbol persisting@1000 : old size: 18 bytes, new size: 18 bytes, delta: 0 bytes

Old	New
1. 0x00000000	1. 0x00000000
2. 0x00000000	2. 0x00000000
3. 0x00000000	3. 0x00000000
4. 0x00000000	4. 0x00000000
5. 0x00000000	5. 0x00000000
6. 0x00000000	6. 0x00000000
7. 0x00000000	7. 0x00000000

Generated: 2021-05-11 19:14:00 by elf_diff @022f90b24021e0b0a70762c0e42a00d06 (https://github.com/DanGelsinger/elf_diff)
© 2021 by noseeglasses (noseeglasses@gmail.com)
Powered by exttable tables from [kryogenix.org](https://github.com)

Single Page

ELF Binary Comparison

Overview

- Binaries
 - Resource Consumption
 - Program Storage
 - Static RAM
 - Legend
- Symbols
 - Persisting Symbols
 - Symbols Disappeared
 - New Symbols
 - Similar Symbols

Binaries

old ./elf_diff_linux/testing/libelf_diff_test_debug_05.4
new ./elf_diff_linux/testing/libelf_diff_test_release_new.4

Statistics

Program Storage

	Oldbytes	Newbytes	Delta/Bytes
overall	329	335	6
text	321	323	2
data	0	0	0

Static RAM

	Oldbytes	Newbytes	Delta/Bytes
overall	0	0	0
data	0	0	0
bss	0	0	0

Legend

text	instructions
data	initialized global or static variables
bss	uninitialized global or static variables

Symbols

7 symbols found in ./elf_diff_linux/testing/libelf_diff_test_debug_05.4
7 symbols found in ./elf_diff_linux/testing/libelf_diff_test_release_new.4

Symbols

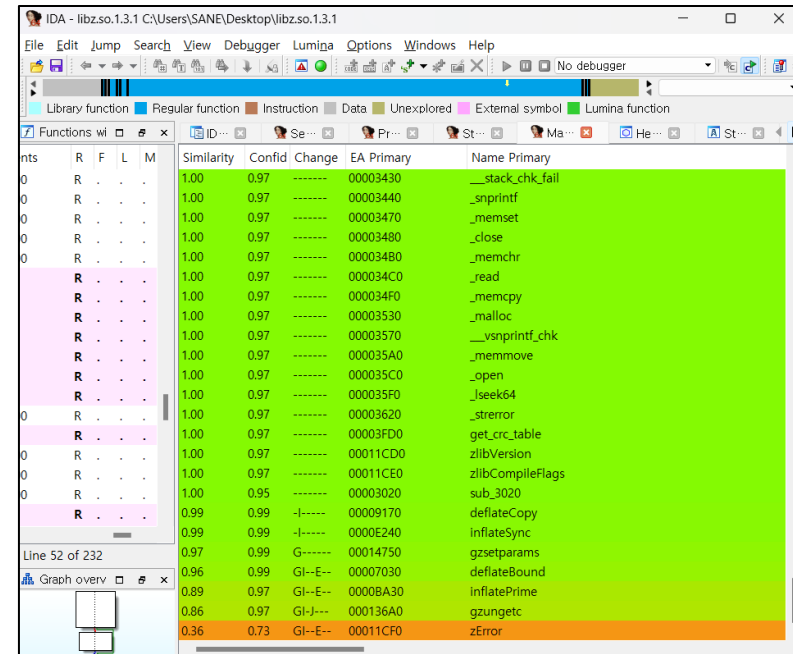
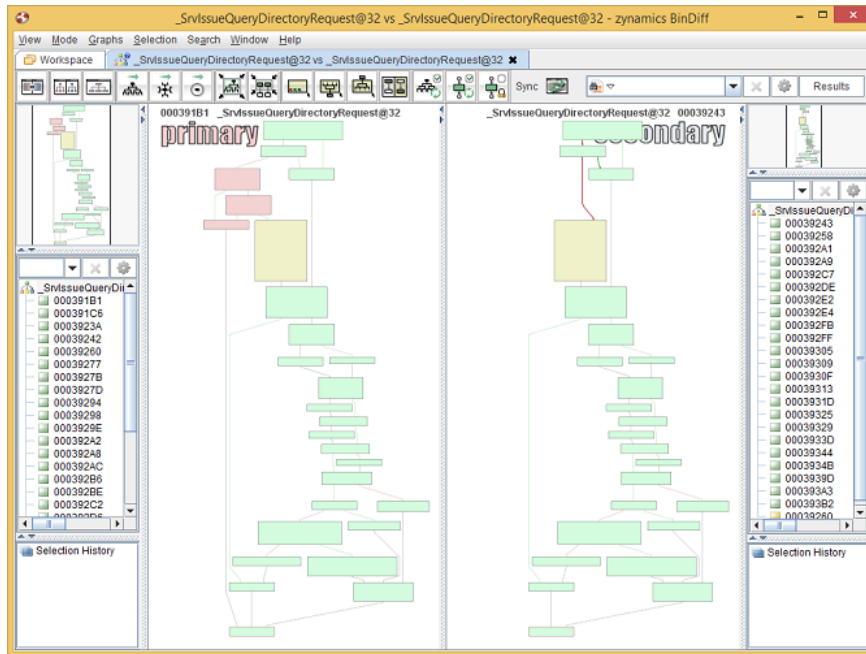
Persisting Symbols

- **elf_diff** (https://github.com/noseglasses/elf_diff)

- ELF 바이너리 간 차이점을 찾거나, 패치 또는 업데이트가 적용된 바이너리의 변경사항을 분석하는 기능을 제공함
- x64, x86 플랫폼에 대한 분석만 지원하기 때문에 다양한 플랫폼을 분석할 수 없음

Symbol Classes		Symbol Selection	
Class	Entities	Old Binary	New Binary
Old	150	Total	150 151
New	151	Selected	150 151
Persisting	144	Dropped	0 0
Disappeared	6	Selection Regex	.*
Appeared	7	Exclusion Regex	
Similar	26		
Migrated	0		

- **BinDiff** (<https://github.com/google/bindiff>)
 - 다른 버전의 바이너리나 악성코드 샘플 간의 변경사항을 식별하는 기능을 제공함
 - IDA Pro가 유료이기 때문에 일반적인 사용자가 쉽게 사용하기 어려움



- **differ (<https://github.com/patacca/differ>)**

- Weisfeiler-Lehman 커널 그래프를 이용하여 유사한 함수를 비교하는 기능을 제공함
- 결과가 텍스트 형태로 제공되기 때문에 가독성이 부족함

```
ubuntu@ubuntu-virtual-machine:~/diff/differ$ python3 differ.py ../libz.so.1.2.13 ../libz.so.1.3.1
WARNING | 2024-03-17 23:54:02,066 | angr.analyses.propagator.engine_vex.SimEnginePropagatorVEX |
64x2.
[+] Analyzing files libz.so.1.2.13 and libz.so.1.3.1
[+] Starting perfect match analysis
[d] 715 functions from libz.so.1.2.13 have been matched
[d] 719 functions from libz.so.1.3.1 have been matched
[d] 9 functions from libz.so.1.2.13 are still unmatched
[d] 9 functions from libz.so.1.3.1 are still unmatched
[+] 9 functions analyzed in 0.2791 seconds
[+] 9 functions analyzed in 0.2894 seconds
[+] CFG matching completed in 0.0004 seconds

-- REPORT --

715 functions from libz.so.1.2.13 have not been modified
5 functions from libz.so.1.2.13 have been modified
4 functions from libz.so.1.2.13 have been removed
4 functions from libz.so.1.3.1 have been added

-- MODIFIED FUNCTIONS --
zError has been matched to inflateSync with a score of 0.0
gzbuffer has been matched to gzbuffer with a score of 1.0
deflateBound has been matched to deflateBound with a score of 0.05648720128299485
inflatePrime has been matched to zError with a score of 0.23225520382279044
inflateSync has been matched to inflateSync with a score of 0.9998998405104889
deflateCopy has been matched to deflateCopy with a score of 0.9998722449088332
sub_412a33 has been matched to gzungetc with a score of 0.004555253231743772
gzungetc has been matched to gzungetc with a score of 0.9427744796058642
gzsetparams has been matched to gzsetparams with a score of 0.9996355958365204

-- REMOVED FUNCTIONS --
zError
inflatePrime
deflateBound
sub_412a33

-- ADDED FUNCTIONS --
gz_intmax has been added
inflatePrime has been added
deflateBound has been added
zError has been added
```

• VulSeeker (<https://github.com/buptsseGJ/VulSeeker>)

- 취약한 함수가 주어지는 경우 바이너리가 동일한 취약점을 보유하고 있는지 식별하는 기능을 제공함
- IDA Pro의 LSFG(데이터 흐름 그래프 및 제어 흐름 그래프) 생성 기능이 필요하기 때문에 실제로 사용하기 어려움 (IDA Pro가 유료이기 때문)

	A	B	C	D	E	F	G	H
1	ssl3_get_new_session_ticket	0.779860506	CVE-2015-1791/tomato-K26USI	0.754279	0.858834	0.842039	0.939141	0.861537
2	SSL_use_PrivateKey_file	0.813533497	CVE-2015-1791/tomato-K26USI	0.83667	0.915928	0.860441	0.830639	0.857082
3	dtls1_process_heartbeat	0.757162424	CVE-2015-1791/tomato-K26USI	0.701984	0.91458	0.900188	0.734073	0.848396
4	sub_1B538	0.750669112	CVE-2015-1791/tomato-K26USI	0.73313	0.861505	0.830864	0.757742	0.742112
5	ssl_load_ciphers	0.777979152	CVE-2015-1791/tomato-K26USI	0.765611	0.830626	0.810815	0.847601	0.777333
6	dtls1_retransmit_message	0.759088302	CVE-2015-1791/tomato-K26USI	0.783538	0.856652	0.777182	0.723469	0.876985
7	SSL_COMP_add_compression_me	0.758093453	CVE-2015-1791/tomato-K26USI	0.687755	0.840968	0.865802	0.838305	0.734407
8	ssl3_send_certificate_request	0.739226993	CVE-2015-1791/tomato-K26USI	0.700885	0.867661	0.883274	0.782318	0.860255
9	dtls1_output_cert_chain	0.798552317	CVE-2015-1791/tomato-K26USI	0.871274	0.872065	0.817703	0.694537	0.83981
10	dtls1_send_certificate_request	0.782831686	CVE-2015-1791/tomato-K26USI	0.769057	0.877017	0.861597	0.770878	0.836904
11	SSL_CTX_use_PrivateKey_file	0.808007215	CVE-2015-1791/tomato-K26USI	0.792566	0.908248	0.891886	0.757432	0.839758
12	dtls1_get_message	0.78131873	CVE-2015-1791/tomato-K26USI	0.798184	0.870821	0.875729	0.891357	0.874836
13	ssl_verify_cert_chain	0.781932954	CVE-2015-1791/tomato-K26USI	0.757434	0.840313	0.84384	0.77997	0.831527
14	tls1_process_heartbeat	0.764426048	CVE-2015-1791/tomato-K26USI	0.726876	0.903016	0.849088	0.708788	0.790018
15	tls1_final_finish_mac	0.738763034	CVE-2015-1791/tomato-K26USI	0.8425	0.831694	0.818513	0.765804	0.794352
16	SSL_new	0.761570325	CVE-2015-1791/tomato-K26USI	0.881754	0.790288	0.712022	0.690948	0.839968
17	SSL_add_file_cert_subjects_to_stac	0.659087902	CVE-2015-1791/tomato-K26USI	0.525855	0.795265	0.82171	0.815435	0.629997
18	dtls1_heartbeat	0.763528253	CVE-2015-1791/tomato-K26USI	0.728572	0.838857	0.800785	0.721583	0.766781
19	dtls1_send_client_verify	0.743147902	CVE-2015-1791/tomato-K26USI	0.68741	0.794986	0.826722	0.885647	0.720047
20	sub_3DC48	0.752343301	CVE-2015-1791/tomato-K26USI	0.78057	0.81061	0.76497	0.846113	0.793118

- DeepBinDiff (<https://github.com/yueduan/DeepBinDiff>)

- Tensorflow를 통해 바이너리 코드를 비교하여 차이점을 식별하는 기능을 제공함
- x86 바이너리에 대한 차이점 식별 기능만 제공함
- Dependency를 맞춰서 설치해야 하기 때문에 쉽게 설치하여 사용하기 어려움

```
ubuntu@ubuntu-virtual-machine:~/DeepBinDiff/src$ pip3 install lapjv
Collecting lapjv
  Downloading lapjv-1.3.27.tar.gz (11 kB)
  Installing build dependencies ... error
  ERROR: Command errored out with exit status 1:
   command: /usr/bin/python3 /usr/lib/python3/dist-packages/pip install --ignore-installed --no-user --prefix /tmp/pip-build-env-lxalp8ru/overlay --no-warn-script-location --no-binary :none: --only-binary :none: -i https://pypi.org/simple -- 'setuptools>=42' wheel 'numpy>=2.0'
   cwd: None
  Complete output (0 lines):
  Collecting setuptools>=42
    Downloading setuptools-71.1.0-py3-none-any.whl (2.3 MB)
  Collecting wheel
    Downloading wheel-0.43.0-py3-none-any.whl (65 kB)
  ERROR: Could not find a version that satisfies the requirement numpy>=2.0 (from versions: 1.3.0, 1.4.1, 1.5.0, 1.5.1, 1.6.0, 1.6.1, 1.6.2, 1.7.0, 1.7.1, 1.7.2, 1.8.0, 1.8.1, 1.8.2, 1.9.0, 1.9.1, 1.9.2, 1.9.3, 1.10.0.post2, 1.10.1, 1.10.2, 1.10.4, 1.11.0, 1.11.1, 1.11.2, 1.11.3, 1.12.0, 1.12.1, 1.13.0, 1.13.1, 1.13.3, 1.14.0, 1.14.1, 1.14.2, 1.14.3, 1.14.4, 1.14.5, 1.14.6, 1.15.0, 1.15.1, 1.15.2, 1.15.3, 1.15.4, 1.16.0, 1.16.1, 1.16.2, 1.16.3, 1.16.4, 1.16.5, 1.16.6, 1.17.0, 1.17.1, 1.17.2, 1.17.3, 1.17.4, 1.17.5, 1.18.0, 1.18.1, 1.18.2, 1.18.3, 1.18.4, 1.18.5, 1.19.0, 1.19.1, 1.19.2, 1.19.3, 1.19.4, 1.19.5, 1.20.0, 1.20.1, 1.20.2, 1.20.3, 1.21.0, 1.21.1, 1.21.2, 1.21.3, 1.21.4, 1.21.5, 1.21.6, 1.22.0, 1.22.1, 1.22.2, 1.22.3, 1.22.4, 1.23.0rc1, 1.23.0rc2, 1.23.0rc3, 1.23.0, 1.23.1, 1.23.2, 1.23.3, 1.23.4, 1.23.5, 1.24.0rc1, 1.24.0rc2, 1.24.0, 1.24.1, 1.24.2, 1.24.3, 1.24.4)
  ERROR: No matching distribution found for numpy>=2.0
  -----
  ERROR: Command errored out with exit status 1: /usr/bin/python3 /usr/lib/python3/dist-packages/pip install --ignore-installed --no-user --prefix /tmp/pip-build-env-lxalp8ru/overlay --no-warn-script-location --no-binary :none: --only-binary :none: -i https://pypi.org/simple -- 'setuptools>=42' wheel 'numpy>=2.0' Check the logs for full command output.
```

• 기존 디핑 도구 동작 여부

- 15개의 오픈소스 바이너리 디핑 도구들을 실제로 테스트해본 결과 대부분의 도구들이 동작하지 않는다는 문제점이 존재함 (5개 도구만 정상 동작)

번호	도구	동작여부
1	TRACY	X
2	Multi-MH	X
3	discovRE	X
4	Esh	X
5	Genius	X
6	VulSeeker	X
7	DeepBinDiff	X
8	Radare2	O
9	angr	O
10	elf_diff	O
11	BinDiff	O
12	Ghidriff	X
13	Diaphora	X
14	Just Another Differ	X
15	differ	O

• 기존 디핑 도구 성능 테스트

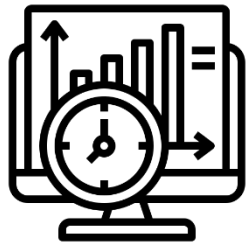
- 명령어 제공 여부
 - 명령어를 통해 사용자가 원하는 기능을 실행할 수 있는지
- 멀티 플랫폼 지원 여부
 - 다양한 플랫폼(x64, x86, arm 등)을 분석 가능한지
- 결과 보고서 생성 여부
 - 디핑 결과가 보고서 파일로 생성되는지
- 함수 주소 표시 여부
 - 분석된 함수의 주소가 표시되는지
- 분석 소요 시간 표시 여부
 - 분석에 소요된 시간이 표시되는지

• 기존 디핑 도구 성능 테스트 결과

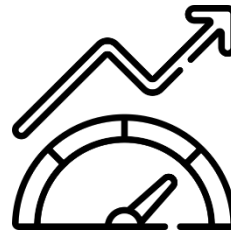
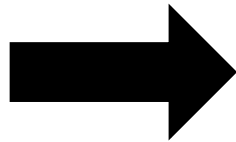
이름	명령어 제공	멀티 플랫폼 지원	결과 보고서	함수 주소 표시	분석 소요 시간
Radare2	○	○	X	○	X
angr	○	○	X	X	X
elf_diff	○	X	○	X	X
BinDiff	X	○	X	○	○
differ	○	○	X	X	X

• 기존 디핑 도구 분석 결과

- 바이너리 디핑 도구들에 대한 성능 테스트를 수행한 결과 Radare2와 BinDiff가 가장 많은 기능들을 제공함
- 하지만 해당 도구들도 성능적인 부분에서 부족한 점이 존재함
- 따라서 이러한 부분들을 보완한 도구를 개발할 필요성이 존재함



기존 디핑 도구



Radare2
BinDiff

but



결과 보고서 제공 X,
유료, etc.

PART

4

개선된 디핑 도구

• CRABA

- 기존 디핑 도구들의 문제점을 개선하여 다양한 기능을 제공하는 도구를 개발함



+



CRA

BAce

CRABA

```
def main():
    parser = argparse.ArgumentParser(description="Compare binaries and generate CFGs for differing functions.")
    parser.add_argument("file1_or_folder1", help="The path to the first binary file or directory")
    parser.add_argument("file2_or_folder2", help="The path to the second binary file or directory")
    parser.add_argument("output_dir", help="The directory to save the comparison results")
    parser.add_argument("-cb", "--compare", action="store_true", help="Perform comparison on individual files")
    parser.add_argument("-fb", "--folder-binary", action="store_true", help="Perform comparison on all binary files in folders")
    parser.add_argument("-ft", "--folder-text", action="store_true", help="Perform comparison on all text files in folders")

    args = parser.parse_args()

    if args.compare:
        compare_binaries(args.file1_or_folder1, args.file2_or_folder2, args.output_dir)
    elif args.folder_binary:
        compare_folder_binary(args.file1_or_folder1, args.file2_or_folder2, args.output_dir)
    elif args.folder_text:
        compare_folder_text(args.file1_or_folder1, args.file2_or_folder2, args.output_dir)
    else:
        print("Please specify either --compare (-c) for single file comparison, --folder-binary (-fb) for binary folder comparison, or --folder-text (-ft) for text folder comparison")

if __name__ == "__main__":
    main()
```

• CRABA 개발 과정

- 기존 디핑 도구들의 문제점을 개선하여 다양한 기능을 제공하는 도구를 개발함

Comparison Report: ntkrnlmp.exe.x64.10.0.22621.1413 vs ntkrnlmp.exe.x64.10.0.22621.1344				
Execution Start Time: 2024-06-03 17:36:23				
Execution End Time: 2024-06-03 17:58:04				
Total Functions Analyzed: 22903				
Different Functions: 11606				
Identical Functions: 78				
Total Execution Time: 00:21:41.26				
Function Name	Function Address	Size (Bytes)	Radiff2 Result	Size Difference (Bytes)
sym_ntoskrnl_exe_PsGetCurrentServerSilo	0x14029f480	2043524	UNMATCH	256
sym_ntoskrnl_exe_Ordinal_4	0x1402f2540	1763119	UNMATCH	256
fcn_1402f26cc	0x1402f26cc	1762732	UNMATCH	256
fcn_14038f4f4	0x14038f4f4	1325826	UNMATCH	256
fcn_1402149ec	0x1402149ec	2328980	UNMATCH	256
fcn_1403eb720	0x1403eb720	1068294	NEW	1068294
sym_ntoskrnl_exe_Ordinal_7	0x1402d3ff0	1853622	UNMATCH	256
fcn_1404341d0	0x1404341d0	146	NEW	146
sym_ntoskrnl_exe_EtwWriteEx	0x14029c640	2055342	UNMATCH	256
fcn_1402b7be0	0x1402b7be0	1956464	UNMATCH	256
fcn_14028d6c4	0x14028d6c4	2098787	UNMATCH	256
fcn_1405c188c	0x1405c188c	38	NEW	38
fcn_1405c18e0	0x1405c18e0	27	NEW	27
fcn_1405c16e4	0x1405c16e4	41	NEW	41
sym_ntoskrnl_exe_CcAddDirtyPagesToExternalCache	0x140399980	1280925	UNMATCH	256
sym_ntoskrnl_exe_CcAsyncCopyRead	0x140288c00	2110030	UNMATCH	256
sym_ntoskrnl_exe_CcCopyReadEx	0x14023a890	2369223	UNMATCH	256
sym_ntoskrnl_exe_CcDeductDirtyPagesFromExternalCache	0x140399640	1281631	UNMATCH	256
fcn_1405256a4	0x1405256a4	202	NEW	202
fcn_1402334c8	0x1402334c8	2391739	UNMATCH	256

• CRABA 개선점

- 명령어를 통해 사용자가 원하는 기능을 선택할 수 있는 옵션을 제공함

```
usage: craba.py [-h] [-cb] [-fb] [-ft] file1_or_folder1 file2_or_folder2 output_dir

Compare binaries and generate CFGs for differing functions.

positional arguments:
  file1_or_folder1      The path to the first binary file or directory
  file2_or_folder2      The path to the second binary file or directory
  output_dir            The directory to save the comparison results

options:
  -h, --help            show this help message and exit
  -cb, --compare        Perform comparison on individual files
  -fb, --folder-binary  Perform comparison on all binary files in folders
  -ft, --folder-text    Perform comparison on all text files in folders
```

• CRABA 개선점

- 바이너리 분석에 소요된 시간을 표시함

Comparison Report: ntkrnlmp.exe.x64.10.0.22621.1413 vs ntkrnlmp.exe.x64.10.0.22621.1344				
Execution Start Time: 2024-06-03 17:36:23				
Execution End Time: 2024-06-03 17:58:04				
Total Functions Analyzed: 22903				
Different Functions: 11606				
Identical Functions: 78				
Total Execution Time: 00:21:41.26				
Function Name	Function Address	Size (Bytes)	Radiff2 Result	Size Difference (Bytes)
sym_ntoskrnl.exe.PsGetCurrentServerSilo	0x14029f480	2043524	UNMATCH	256
sym_ntoskrnl.exe.Ordinal_4	0x1402f2540	1763119	UNMATCH	256
fcn_1402f26cc	0x1402f26cc	1762732	UNMATCH	256
fcn_14038f4f4	0x14038f4f4	1325826	UNMATCH	256
fcn_1402149ec	0x1402149ec	2328980	UNMATCH	256
fcn_1403eb720	0x1403eb720	1068294	NEW	1068294
sym_ntoskrnl.exe.Ordinal_7	0x1402d3ff0	1853622	UNMATCH	256
fcn_1404341d0	0x1404341d0	146	NEW	146
sym_ntoskrnl.exe.EtwWriteEx	0x14029c640	2055342	UNMATCH	256
fcn_1402b7be0	0x1402b7be0	1956464	UNMATCH	256
fcn_14028d6c4	0x14028d6c4	2098787	UNMATCH	256
fcn_1405c188c	0x1405c188c	38	NEW	38
fcn_1405c18e0	0x1405c18e0	27	NEW	27
fcn_1405c16e4	0x1405c16e4	41	NEW	41
sym_ntoskrnl.exe.CcAddDirtyPagesToExternalCache	0x140399980	1280925	UNMATCH	256
sym_ntoskrnl.exe.CcAsyncCopyRead	0x140288c00	2110030	UNMATCH	256
sym_ntoskrnl.exe.CcCopyReadEx	0x14023a890	2369223	UNMATCH	256
sym_ntoskrnl.exe.CcDeductDirtyPagesFromExternalCache	0x140399640	1281631	UNMATCH	256
fcn_1405256a4	0x1405256a4	202	NEW	202
fcn_1402334c8	0x1402334c8	2391739	UNMATCH	256

• CRABA 개선점

- 결과 보고서를 통해 바이너리 간 차이점을 쉽게 파악할 수 있음

Comparison Report: ntkrnlmp.exe.x64.10.0.22621.1413 vs ntkrnlmp.exe.x64.10.0.22621.1344				
Execution Start Time: 2024-06-03 17:36:23				
Execution End Time: 2024-06-03 17:58:04				
Total Functions Analyzed: 22903				
Different Functions: 11606				
Identical Functions: 78				
Total Execution Time: 00:21:41.26				
Function Name	Function Address	Size (Bytes)	Radiff2 Result	Size Difference (Bytes)
sym_ntoskrnl_exe_PsGetCurrentServerSilo	0x14029f480	2043524	UNMATCH	256
sym_ntoskrnl_exe_Ordinal_4	0x1402f2540	1763119	UNMATCH	256
fcn_1402f26cc	0x1402f26cc	1762732	UNMATCH	256
fcn_14038f4f4	0x14038f4f4	1325826	UNMATCH	256
fcn_1402149ec	0x1402149ec	2328980	UNMATCH	256
fcn_1403eb720	0x1403eb720	1068294	NEW	1068294
sym_ntoskrnl_exe_Ordinal_7	0x1402d3ff0	1853622	UNMATCH	256
fcn_1404341d0	0x1404341d0	146	NEW	146
sym_ntoskrnl_exe_EtwWriteEx	0x14029c640	2055342	UNMATCH	256
fcn_1402b7be0	0x1402b7be0	1956464	UNMATCH	256
fcn_14028d6c4	0x14028d6c4	2098787	UNMATCH	256
fcn_1405c188c	0x1405c188c	38	NEW	38
fcn_1405c18e0	0x1405c18e0	27	NEW	27
fcn_1405c16e4	0x1405c16e4	41	NEW	41
sym_ntoskrnl_exe_CcAddDirtyPagesToExternalCache	0x140399980	1280925	UNMATCH	256
sym_ntoskrnl_exe_CcAsyncCopyRead	0x140288c00	2110030	UNMATCH	256
sym_ntoskrnl_exe_CcCopyReadEx	0x14023a890	2369223	UNMATCH	256
sym_ntoskrnl_exe_CcDeductDirtyPagesFromExternalCache	0x140399640	1281631	UNMATCH	256
fcn_1405256a4	0x1405256a4	202	NEW	202
fcn_1402334c8	0x1402334c8	2391739	UNMATCH	256

• CRABA 개선점

- 다양한 플랫폼에 대한 디핑 기능을 지원함

Comparison Report: ntkrnlmp.exe.x64.10.0.22621.1413 vs ntkrnlmp.exe.x64.10.0.22621.1344

Execution Start Time: 2024-06-03 17:36:23

Execution End Time: 2024-06-03 17:58:04

Total Functions Analyzed: 22903

Different Functions: 11606

Identical Functions: 78

Total Execution Time: 00:21:41.26

exe 파일

Comparison Report: true vs false

Execution Start Time: 2024-06-11 16:06:28

Execution End Time: 2024-06-11 16:06:30

Total Functions Analyzed: 151

Different Functions: 67

Identical Functions: 0

Total Execution Time: 00:00:02.33

elf 파일

• CRABA 개선점

- 함수 주소를 표시하여 분석을 편리하게 함

Function Name	Function Address
sym_ntoskrnl.exe_PsGetCurrentServerSilo	0x14029f480
sym_ntoskrnl.exe_Ordinal_4	0x1402f2540
fcv_1402f26cc	0x1402f26cc
fcv_14038f4f4	0x14038f4f4
fcv_1402149ec	0x1402149ec
fcv_1403eb720	0x1403eb720
sym_ntoskrnl.exe_Ordinal_7	0x1402d3ff0
fcv_1404341d0	0x1404341d0
sym_ntoskrnl.exe_EtwWriteEx	0x14029c640
fcv_1402b7be0	0x1402b7be0
fcv_14028d6c4	0x14028d6c4
fcv_1405c188c	0x1405c188c
fcv_1405c18e0	0x1405c18e0
fcv_1405c16e4	0x1405c16e4
sym_ntoskrnl.exe_CcAddDirtyPagesToExternalCache	0x140399980
sym_ntoskrnl.exe_CcAsyncCopyRead	0x140288c00
sym_ntoskrnl.exe_CcCopyReadEx	0x14023a890
sym_ntoskrnl.exe_CcDeductDirtyPagesFromExternalCache	0x140399640
fcv_1405256a4	0x1405256a4
fcv_1402334c8	0x1402334c8

• CRABA 개선점

- Pseudo 코드를 통해 바이너리 간 차이점을 가시적으로 보여줌

Legends			
Colors	Links		
Added	[[first change		
Changed	(n)ext change		
Related	(C)op		
		<code>./php-cgi_L8.3.3.exe/main</code>	<code>./php-cgi_L8.3.3.exe/main</code>
		<pre> 1 int main (int rcx, int rdx) { 2 loc_01400050: 3 // CODE XREF from entry0 @ 0x14000203(x) 4 qword [var_18h] = rcx 5 push (rcp) 6 push (rsi) 7 push (rdi) 8 push (r12) 9 push (r13) 10 push (r14) 11 push (r15) 12 rbp = rbp - 0x140 13 eax = 0x150 // '\x15' 14 fcn.14000300 () 15 rbp = rbp + 0x140 16 rax = qword [0x140010008] // [0x140010008]-0x2b992ddfa232 17 rax = rax 18 qword [var_140h] = rax 19 eax = dword [0x140010108] // [0x140010108]-1 // reloc.WS2_32.dll_accept 20 r1 = r1p + 0x200 // [0x14000830] 21 r1d = 0 // reloc.WS2_32.dll_accept // reloc.WS2_32.dll_accept 22 dword [var_58h] = ecx 23 dword [arg1] = ecx 24 r13 = rdx 25 ecx = 1 // reloc.WS2_32.dll_accept 26 qword [var_70h] = rdx 27 dword [var_50h] = ecx 28 r1d = 0 // reloc.WS2_32.dll_accept 29 dword [var_50h] = eax 30 r1d = r1d 31 rax = qword [0x140011308] // [0x140011308]-0 32 33 edi = r1d 34 edx = rcx + 7 // reloc.WS2_32.dll_accept 35 dword [arg_70h] = ecx 36 r1d = r1d 37 rcx = r1p + 0x200 // [0x14000830] 38 39 dword [var_78h] = r1d 40 esi = r1d 41 dword [arg_70h] = r1d 42 dword [var_54h] = r1d 43 qword [var_50h] = rax 44 qword [var_50h] = rax 45 qword [arg_70h] = r1d 46 dword [var_50h] = 0x1f4 // 500 47 dword [var_54h] = r1d 48 qword [arg_70h] = r1d 49 dword [var_70h] = r1d 50 dword [var_50h] = r1d 51 dword [var_70h] = r1d 52 dword [var_50h] = r1d 53 54 dword [0x140010b90] = 0x1010000 // [0x140010b90]-0 55 dword [0x140010b94] = 0x10001 // [0x140010b94]-0 56 qword [0x140010b88] = r14 // [0x140010b88]-0 57 qword [sym.....] = r14 // [0x140010b88]-0 58 rcx = r1p + 0x200 // [0x140010d0] // (ptr 0x14000b380) "cgi-fcgi" 59 qword [sym.....] = r14 // [0x140010d0] // (ptr 0x14000b380) "cgi-fcgi" 60 var = dword [0x140011304] - r1d // [0x140011304]-0 // reloc.WS2_32.dll_accept // reloc.WS2_32.dll_accept 61 if (var) goto loc_01400050 // unlikely 62 goto loc_01400050 63 loc_01400050: 64 // CODE XREF from main @ 0x1400050(x) 65 ebx = dword [0x140011308] // [0x140011308]-0 66 dword [arg_70h] = ebx 67 goto loc_01400050 68 } </pre>	<pre> 1 int main (int rcx, int rdx) { 2 loc_01400050: 3 // CODE XREF from entry0 @ 0x14000203(x) 4 qword [var_18h] = rcx 5 push (rcp) 6 push (rsi) 7 push (rdi) 8 push (r12) 9 push (r13) 10 push (r14) 11 push (r15) 12 rbp = rbp - 0x140 13 eax = 0x150 // '\x15' 14 fcn.14000300 () 15 rbp = rbp + 0x140 // reloc.WS2_32.dll_accept 16 rax = qword [0x140010008] // [0x140010008]-0x2b992ddfa232 17 rax = rax 18 qword [var_140h] = rax 19 eax = dword [0x140010108] // [0x140010108]-1 // reloc.WS2_32.dll_accept 20 r1 = r1p + 0x200 // [0x14000830] 21 r1d = 0 // reloc.WS2_32.dll_accept // reloc.WS2_32.dll_accept 22 dword [var_58h] = ecx 23 dword [arg1] = ecx 24 r13 = rdx 25 ecx = 1 // reloc.WS2_32.dll_accept 26 qword [var_70h] = rdx 27 dword [var_50h] = ecx 28 r1d = 0 // reloc.WS2_32.dll_accept 29 dword [var_50h] = eax 30 r1d = r1d 31 rax = qword [0x140011308] // [0x140011308]-0 32 33 esi = r1d 34 ebx = rcx + 7 // reloc.WS2_32.dll_accept 35 dword [var_50h] = ecx 36 dword [arg_70h] = ecx 37 r1d = r1d 38 rcx = r1p + 0x200 // [0x14000830] 39 dword [var_70h] = r1d 40 dword [var_50h] = r1d 41 dword [arg_70h] = r1d 42 dword [var_54h] = r1d 43 qword [rcp] = rax 44 qword [arg_70h] = r1d 45 qword [var_50h] = r1d 46 dword [var_50h] = 0x1f4 // 500 47 dword [var_54h] = r1d 48 qword [arg_70h] = r1d 49 dword [var_70h] = r1d 50 dword [var_50h] = r1d 51 dword [var_70h] = r1d 52 dword [var_50h] = r1d 53 54 dword [0x140010b90] = 0x1010000 // [0x140010b90]-0 55 dword [0x140010b94] = 0x10001 // [0x140010b94]-0 56 qword [0x140010b88] = r14 // [0x140010b88]-0 57 qword [sym.....] = r14 // [0x140010b88]-0 58 rcx = r1p + 0x200 // [0x140010d0] // (ptr 0x14000b380) "cgi-fcgi" 59 qword [sym.....] = r14 // [0x140010d0] // (ptr 0x14000b380) "cgi-fcgi" 60 var = dword [0x140011304] - r1d // [0x140011304]-0 // reloc.WS2_32.dll_accept // reloc.WS2_32.dll_accept 61 if (var) goto loc_01400050 // unlikely 62 goto loc_01400050 63 loc_01400050: 64 // CODE XREF from main @ 0x1400050(x) 65 ebx = dword [0x140011308] // [0x140011308]-0 66 dword [arg_70h] = ebx 67 goto loc_01400050 68 } </pre>

- 그래프를 통해 바이너리 간 차이점을 가시적으로 보여줌



• 기존 디핑 도구와 CRABA의 성능 비교

- 기존 도구와 대비하여 모든 부분에서 성능이 개선됨

이름	명령어 제공	멀티 플랫폼 지원	결과 보고서	함수 주소 표시	분석 소요 시간
Radare2	○	○	X	○	X
angr	○	○	X	X	X
elf_diff	○	X	○	X	X
BinDiff	X	○	X	○	○
differ	○	○	X	X	X
CRABA	○	○	○	○	○

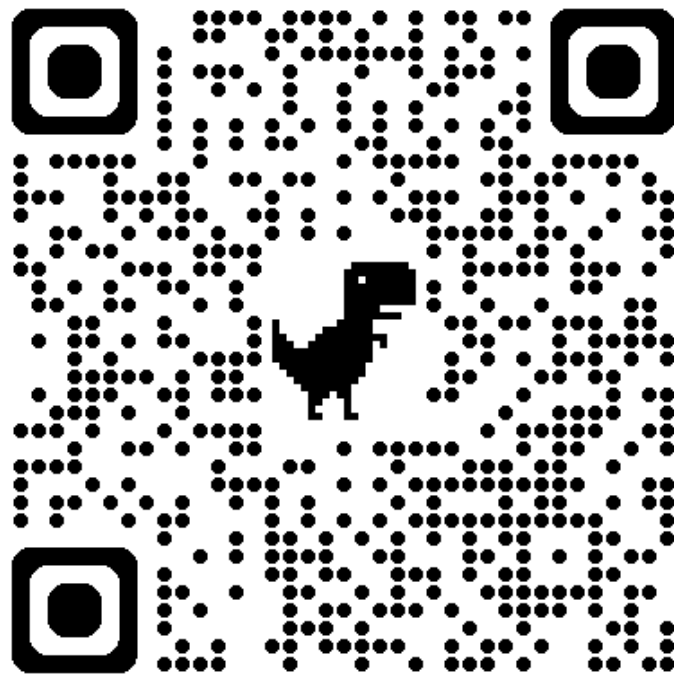
• 결론

- 무료로 퀄리티 높은 디핑 작업을 수행 가능함
- 기존 디핑 도구들에서 부족했던 부분(분석 시간 단축, 디핑 결과 시각화, 안정성 등)들을 개선함
- 따라서 기존 도구들 대비하여 더 좋은 결과를 얻을 수 있을 것으로 기대됨



CRABA

Thank You



다양한 바이너리 디핑 도구의 분석과 비교

2024.08.31

서재완, 이창선