고려대학교
프로그램 자동 수정 연구 소개

오학주

고려대학교 정보대학 컴퓨터학과

11 Feb 2022 @ERC Workshop, Jeju

소프트웨어 분석 연구실@Korea Univ.

- Members: 10 PhD and 6 MS students
- Research areas: programming languages (PL), software engineering (SE), software security
  - program analysis and testing
  - program synthesis and repair

- Publication: top-venues in PL, SE, and Security:
  - PL: POPL('22),PLDI('20),OOPSLA('15,'17a,'17b,'18a,'18b,'19,'20)
  - SE: ICSE('17,'18,'19,'20,'21'22a,'22b), FSE('18,'19,'20,'21), ASE('18), ISSTA('20)
  - Security: IEEE S&P('17,'20), USENIX Security('21)

http://prl.korea.ac.kr

소프트웨어 분석 연구실@Korea Univ.

students

g languages (PL), software engineering

ng

pair

- Publication: top-venues in PL, SE, and Security:
  - PL: POPL('22),PLDI('20),OOPSLA('15,'17a,'17b,'18a,'18b,'19,'20)
  - SE: ICSE('17,'18,'19,'20,'21'22a,'22b), FSE('18,'19,'20,'21), ASE('18), ISSTA('20)
  - Security: IEEE S&P('17,'20), USENIX Security('21)

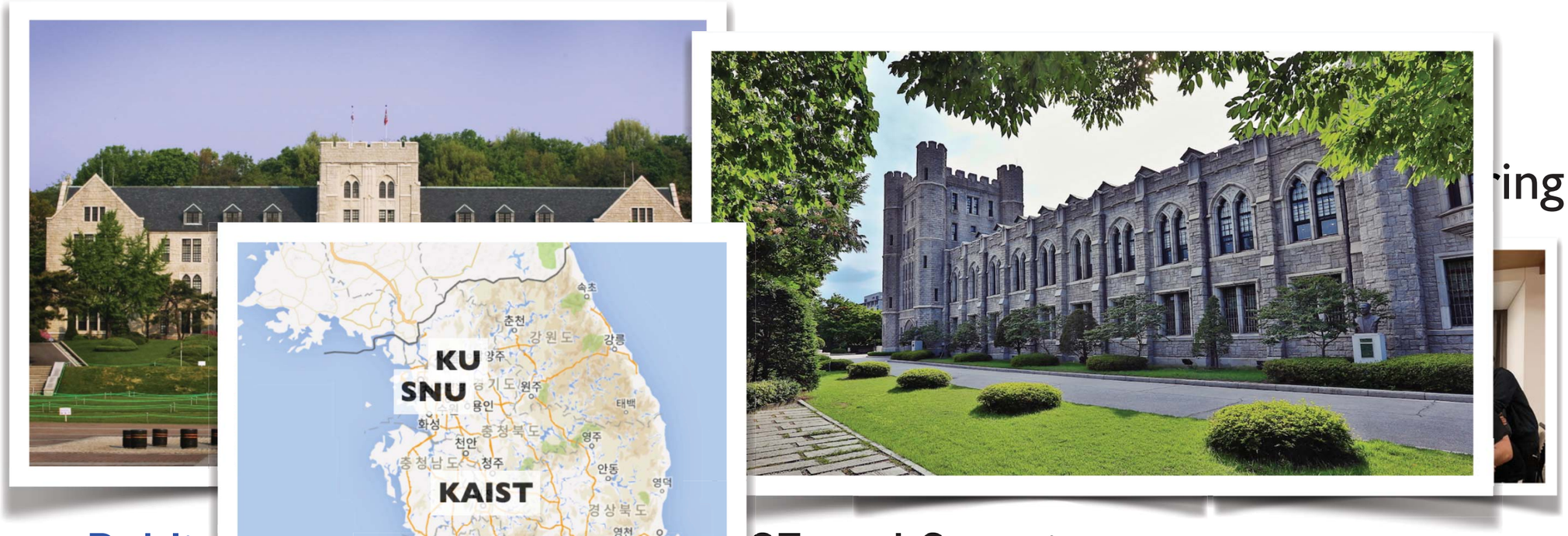
http://prl.korea.ac.kr

소프트웨어 분석 연구실@Korea Univ.

- Publication: top-venues in PL, SE, and Security:
  - PL: POPL('22),PLDI('20),OOPSLA('15,'17a,'17b,'18a,'18b,'19,'20)
  - SE: ICSE('17,'18,'19,'20,'21'22a,'22b), FSE('18,'19,'20,'21), ASE('18), ISSTA('20)
  - Security: IEEE S&P('17,'20), USENIX Security('21)

http://prl.korea.ac.kr

소프트웨어 분석 연구실@Korea Univ.



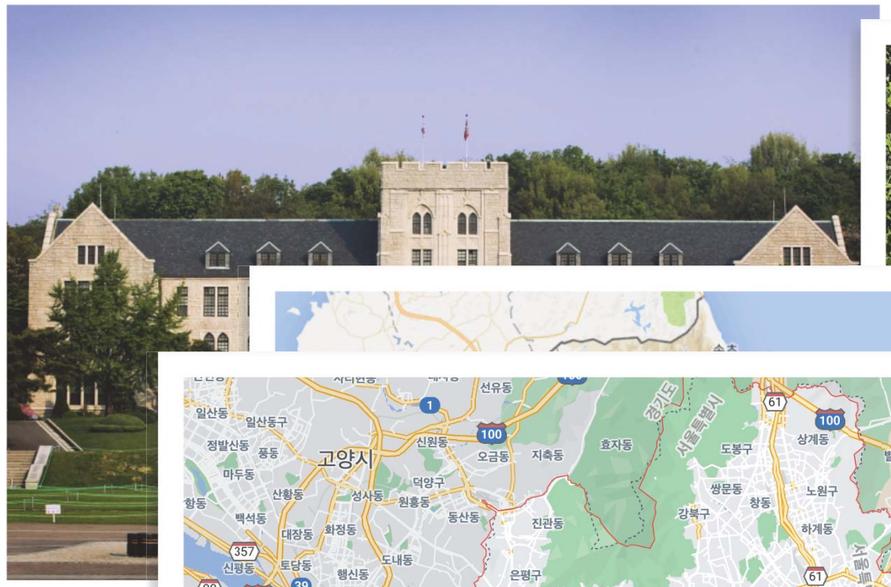
ring

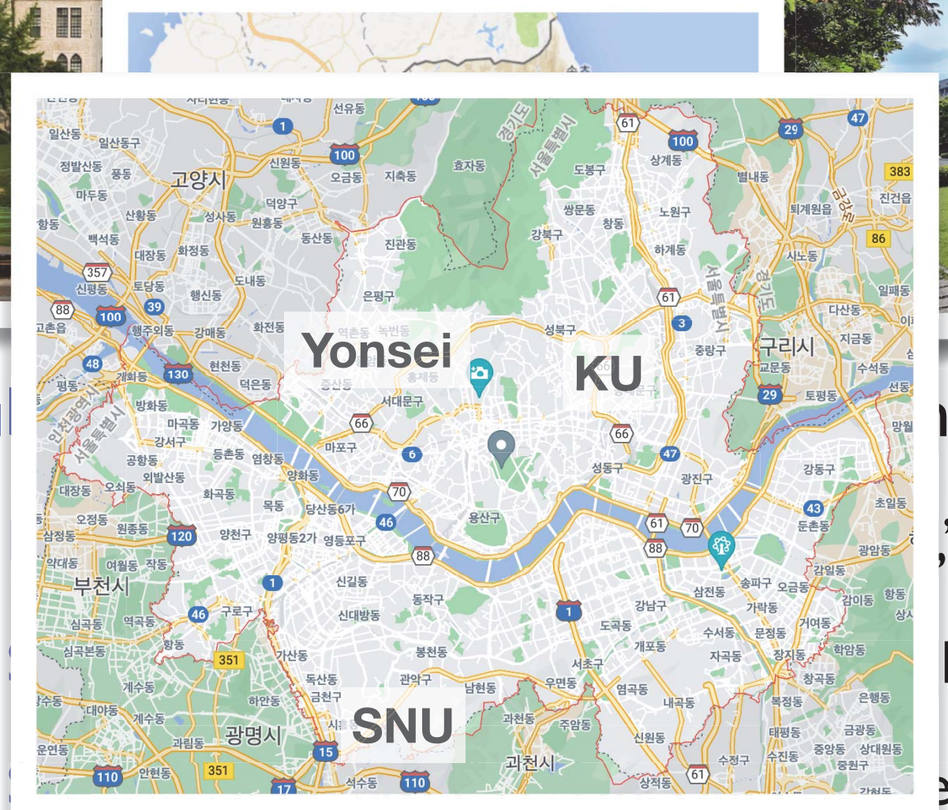
- Public
  - PL: I LA('15,'17a,'17b,'18a,'18b,'19,'20)
  - SE: I 2b), FSE('18,'19,'20,'21), ASE('18), ISSTA('20)
  - Security: IEEE S&P('17,'20), USENIX Security('21)

SE, and Security:

http://prl.korea.ac.kr

소프트웨어 분석 연구실@Korea Univ.

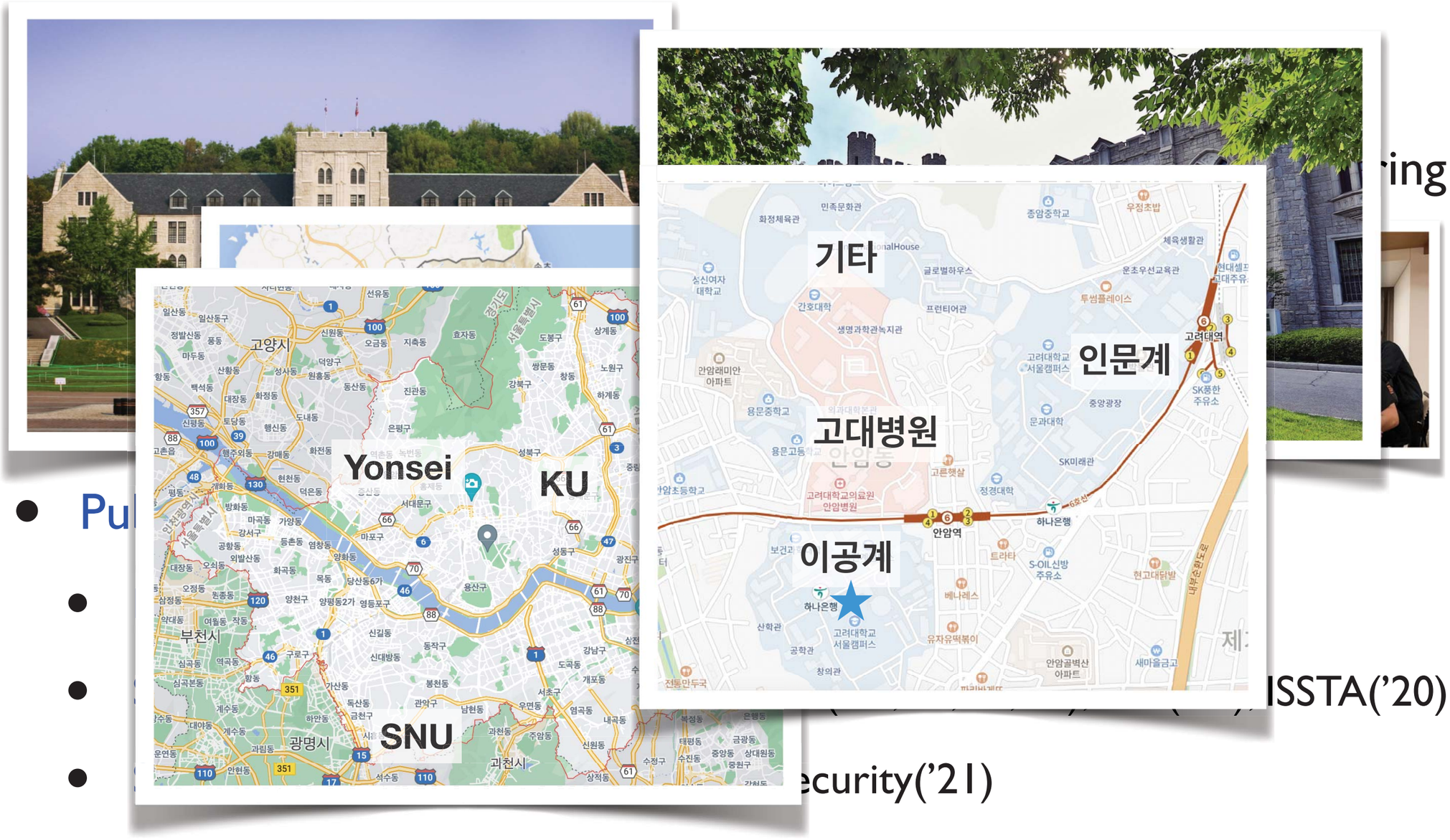




- Pu ring
  - nd Security:
  - ,'17a,'17b,'18a,'18b,'19,'20)
  - E('18,'19,'20,'21), ASE('18), ISSTA('20)
  - ecurity('21)

http://prl.korea.ac.kr

소프트웨어 분석 연구실@Korea Univ.



- Pu
  -
  -
  -

ring

ISSTA('20)

ecurity('21)

http://prl.korea.ac.kr

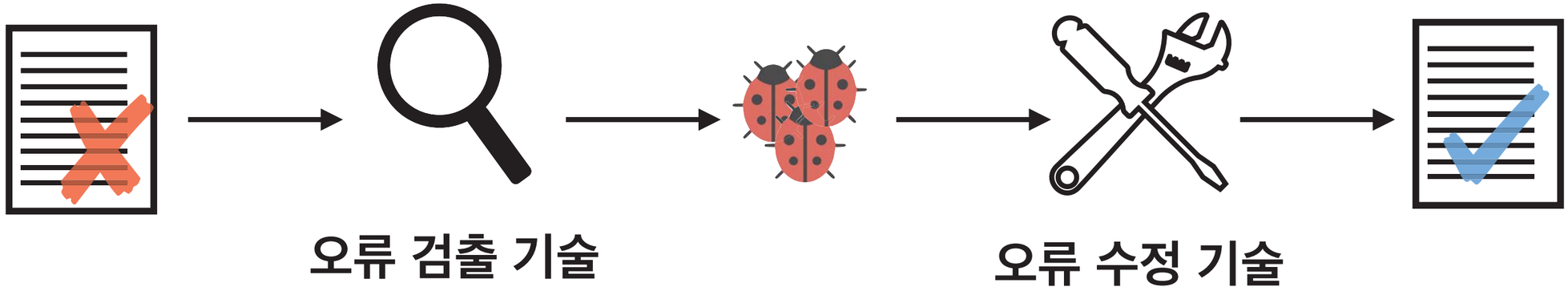
소프트웨어 분석 연구실@Korea Univ.

- Members: 10 PhD and 6 MS students
- Research areas: programming languages (PL), software engineering (SE), software security
  - program analysis and testing
  - program synthesis and repair

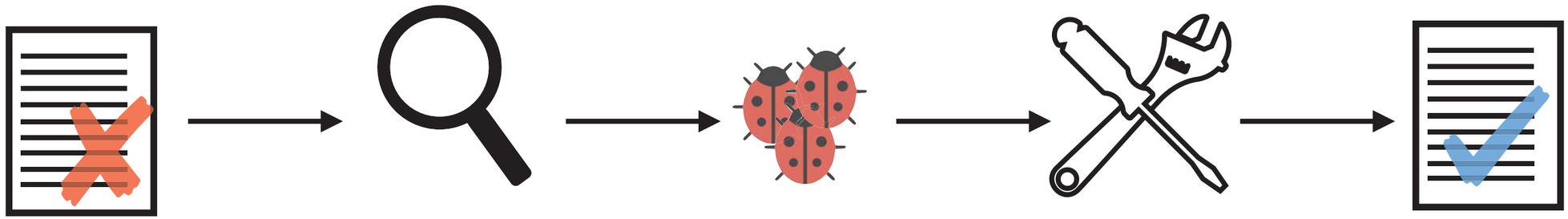
- Publication: top-venues in PL, SE, and Security:
  - PL: POPL('22),PLDI('20),OOPSLA('15,'17a,'17b,'18a,'18b,'19,'20)
  - SE: ICSE('17,'18,'19,'20,'21'22a,'22b), FSE('18,'19,'20,'21), ASE('18), ISSTA('20)
  - Security: IEEE S&P('17,'20), USENIX Security('21)

http://prl.korea.ac.kr

연구 방향: 오류 자동 검출 & 수정

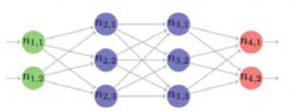


연구 방향: 오류 자동 검출 & 수정

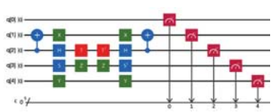


오류 검출 기술

오류 수정 기술

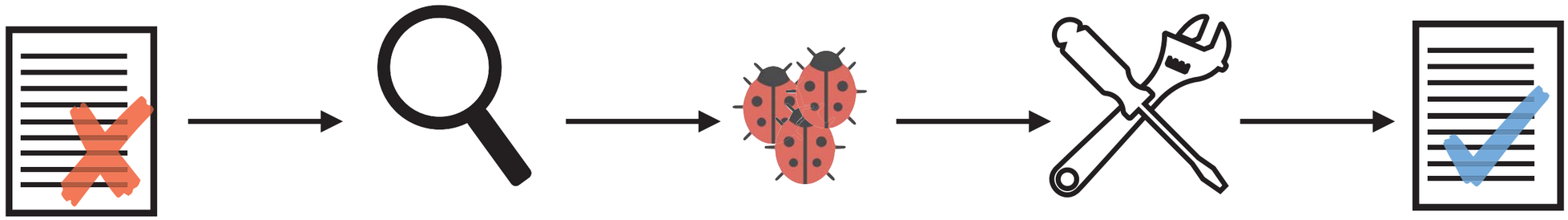


ethereum



⋮

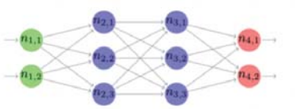
연구 방향: 오류 자동 검출 & 수정



오류 검출 기술

오류 수정 기술

정적 분석

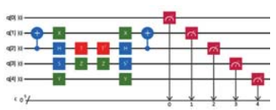


검증(SMT)

콘콜릭

ethereum

기호 실행

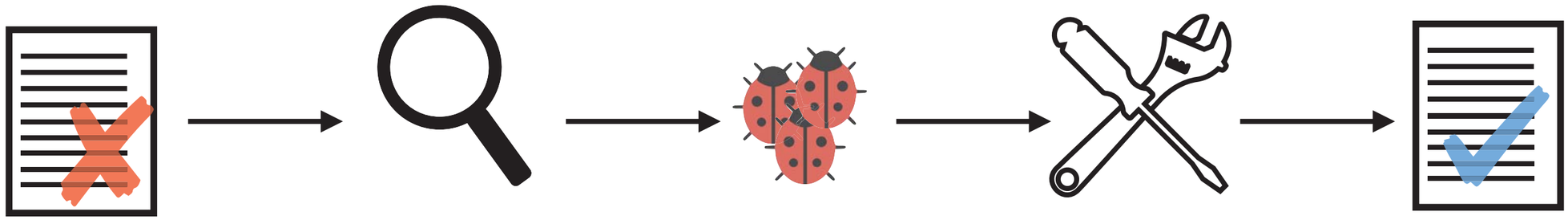


퍼징

⋮

⋮

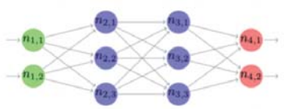
연구 방향: 오류 자동 검출 & 수정



오류 검출 기술

오류 수정 기술

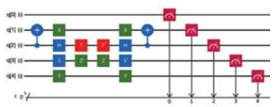
정적 분석



검증(SMT)

콘콜릭

기호 실행



퍼징

⋮

⋮

구문 오류

의미 오류

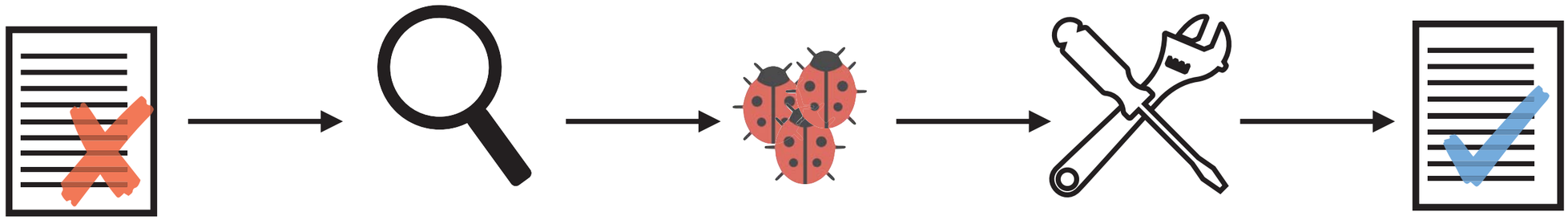
기능 오류

보안 오류

정형 명세

⋮

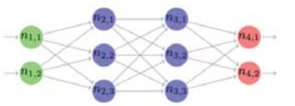
연구 방향: 오류 자동 검출 & 수정



오류 검출 기술

오류 수정 기술

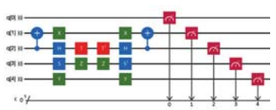
정적 분석



검증(SMT)

콘콜릭

기호 실행



퍼징

⋮

구문 오류

의미 오류

기능 오류

보안 오류

정형 명세

⋮

정적 분석

기호 실행

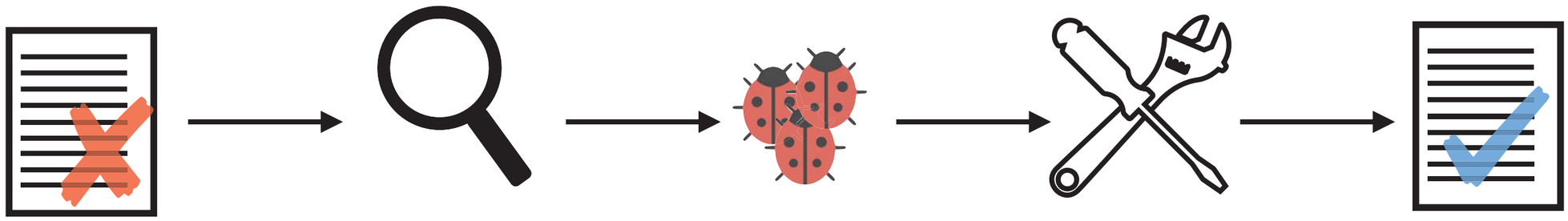
코드 합성

코드 마이닝

기계 학습

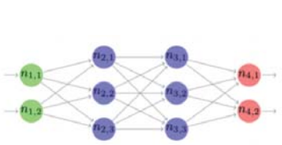
⋮

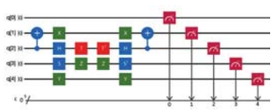
연구 방향: 오류 자동 검출 & 수정



오류 검출 기술

오류 수정 기술





⋮

×

정적 분석

검증(SMT)

콘콜릭

기호 실행

퍼징

⋮

×

구문 오류

의미 오류

기능 오류

보안 오류

정형 명세

⋮

×

정적 분석

기호 실행

코드 합성

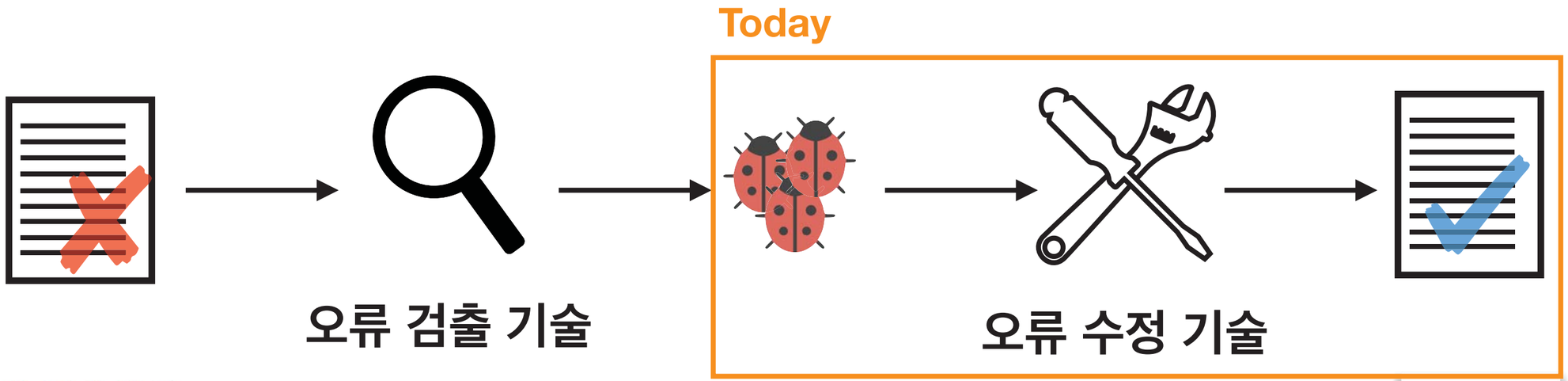
코드 마이닝

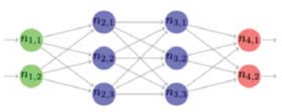
기계 학습

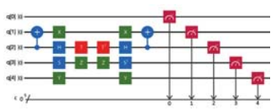
⋮

=

연구 방향: 오류 자동 검출 & 수정







⋮

×

정적 분석

검증(SMT)

콘콜릭

기호 실행

퍼징

⋮

×

구문 오류

의미 오류

기능 오류

보안 오류

정형 명세

⋮

×

정적 분석

기호 실행

코드 합성

코드 마이닝

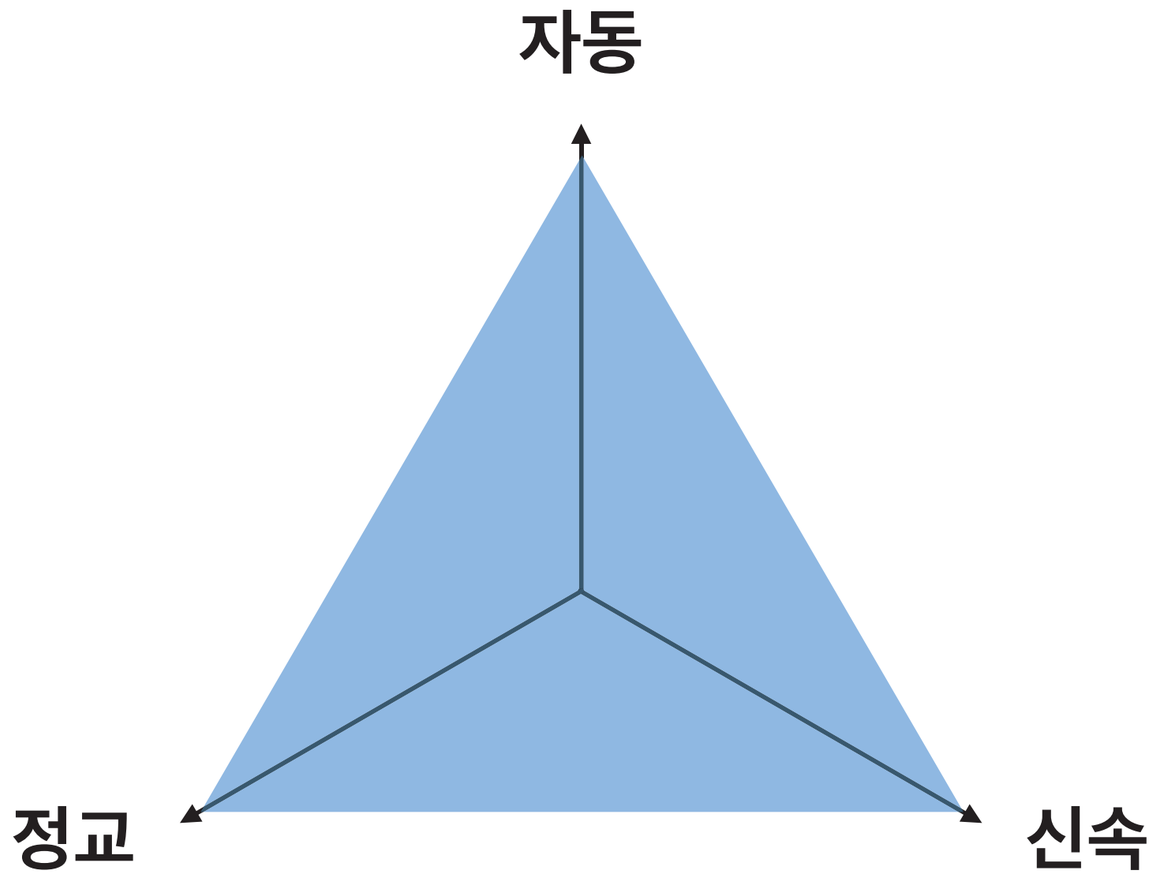
기계 학습

⋮

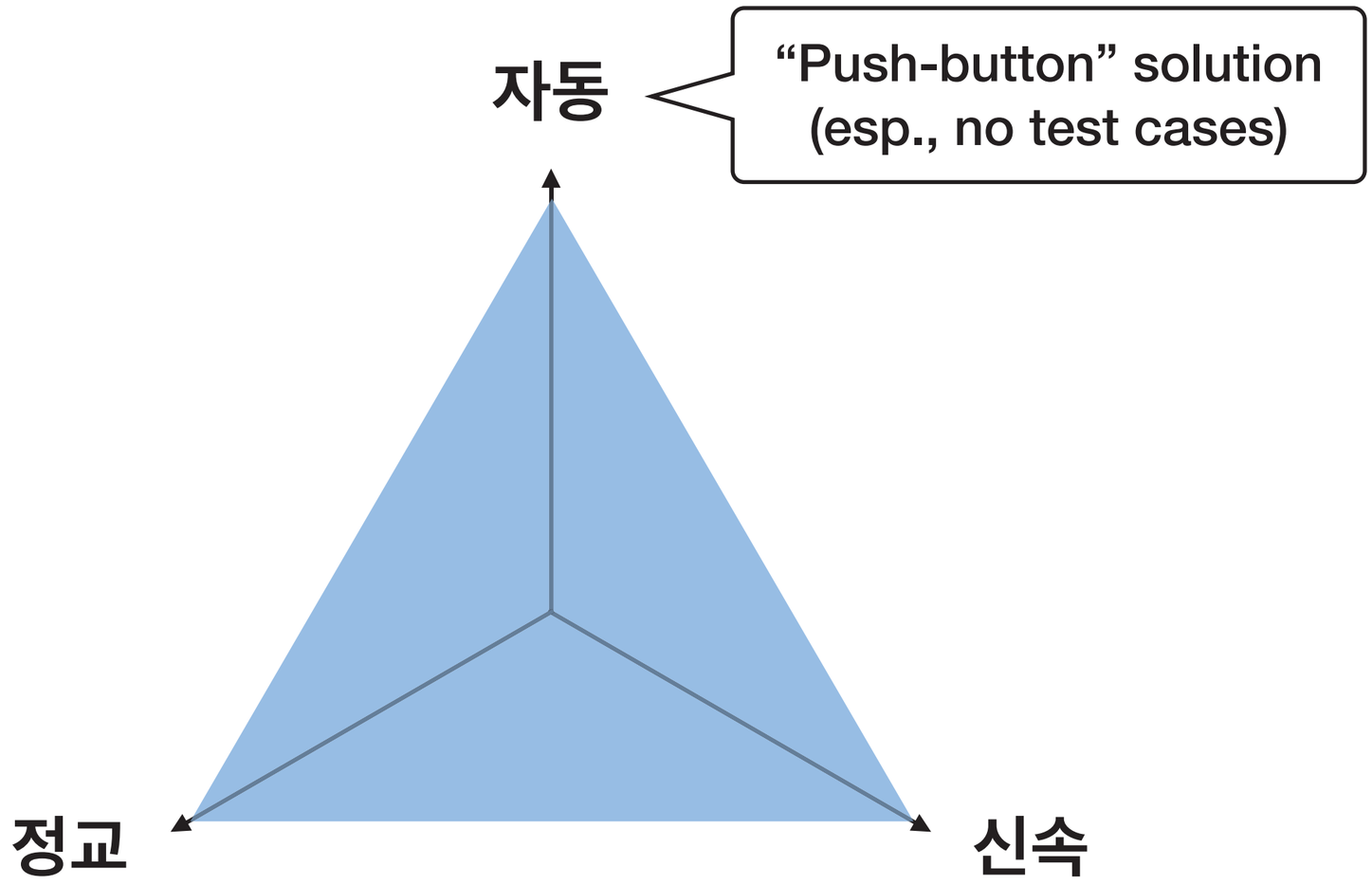
=

⋮

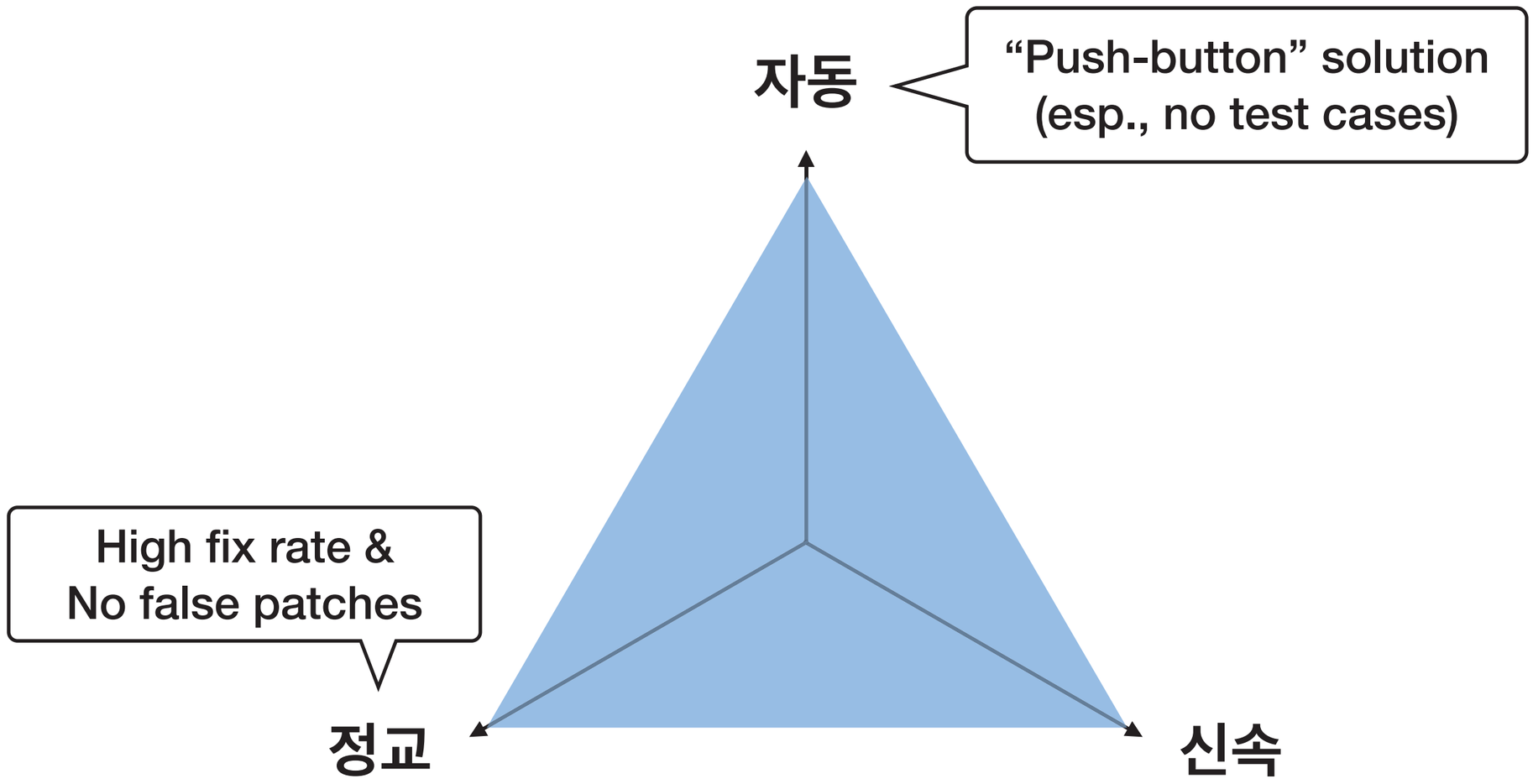
목표: “실용적” 오류 자동 수정 기술



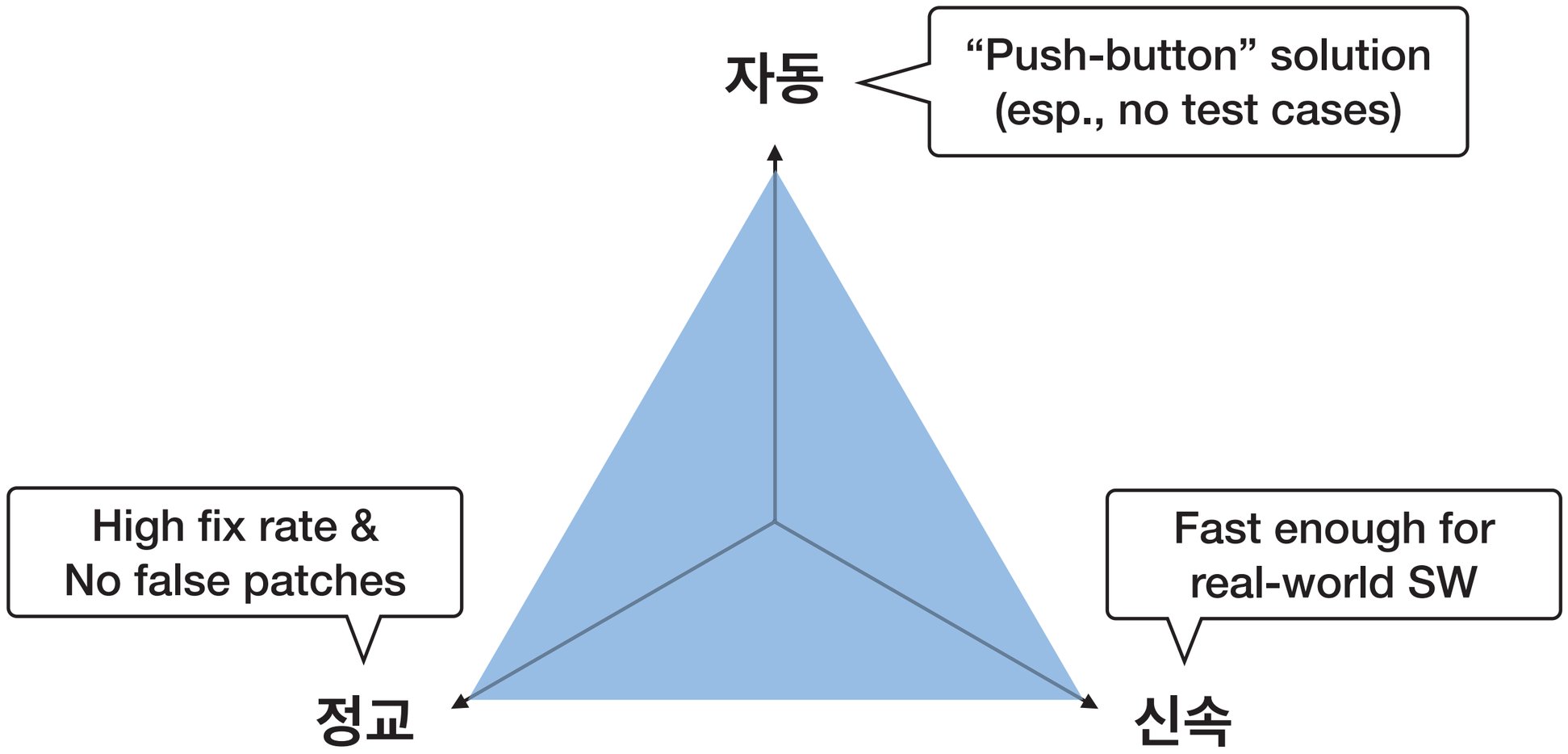
목표: “실용적” 오류 자동 수정 기술



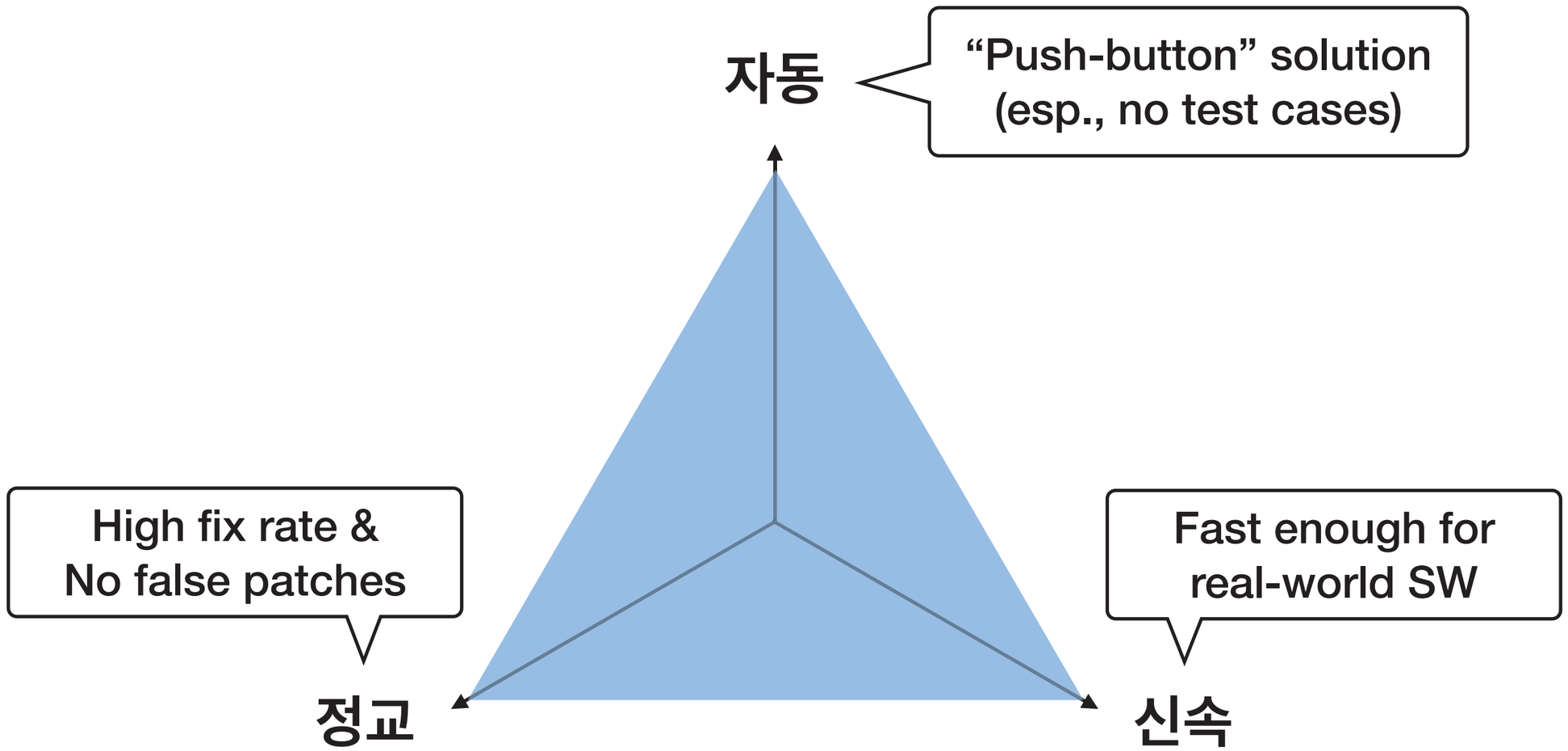
목표: “실용적” 오류 자동 수정 기술



목표: “실용적” 오류 자동 수정 기술



목표: “실용적” 오류 자동 수정 기술



=> 주요 프로그래밍 언어 / 오류에 특화

연구 진행 상황

- C/C++ 메모리 오류 자동 수정 [FSE'18, ICSE'20]
- Java 널 포인터 오류 자동 수정 [ICSE'22]
- Python 타입 오류 자동 수정 (in progress)
- Solidity 보안 오류 자동 수정 (in progress)
- …
- OCaml 프로그래밍 과제 자동 수정 [OOPSLA'18, OOSLA'19, FSE'21]

메모리 관리 오류

- 메모리 관리를 수동으로 하는 언어(e.g., C/C++)에서 발생:
  - Memory-leak (CWE-401): 메모리를 너무 늦게 해제
  - Use-after-free (CWE-416): 메모리를 너무 빨리 해제
  - Double-free (CWE-415): 메모리를 여러번 해제

Memory-Leak

```
p = malloc(1);
...
return;
```

Use-After-Free

```
p = malloc(1);
...
free(p);
...
use(p);
```

Double-Free

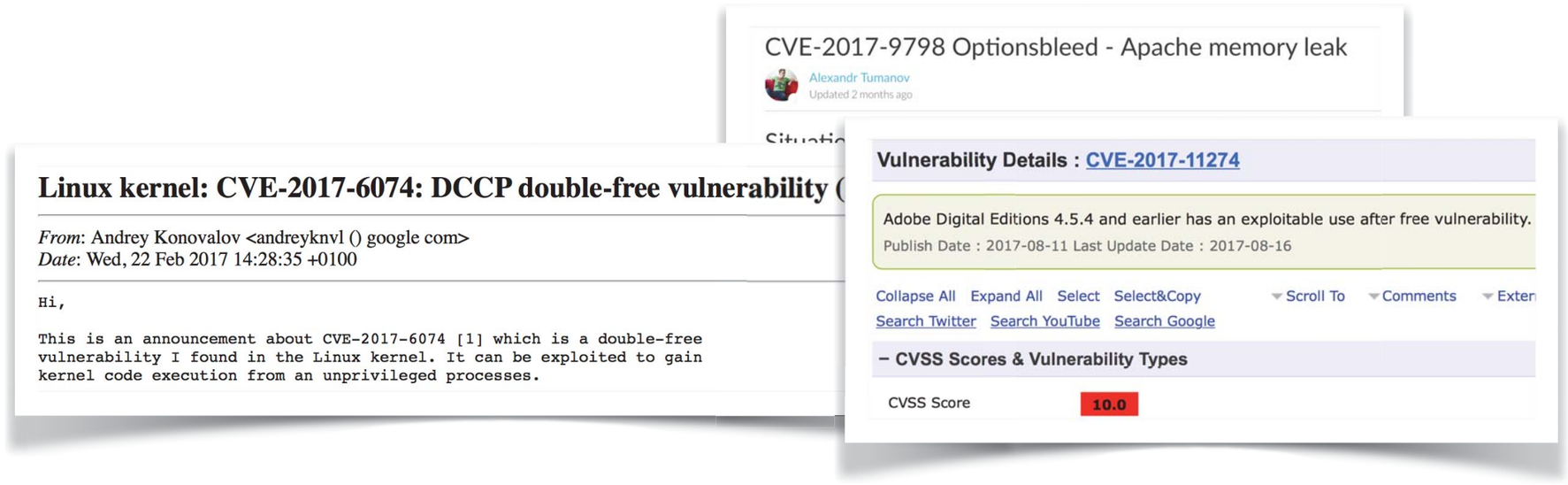
```
p = malloc(1);
...
free(p);
...
free(p);
```

메모리 관리 오류

- C/C++ 프로그램에서 가장 빈번하게 발생

Repository	#commits	ML	DF	UAF	Total	*-overflow
linux	721,119	3,740	821	1,986	**6,363**	5,092
openssl	21,009	220	36	12	**264**	61
numpy	17,008	58	2	2	**59**	53
php	105,613	1,129	148	197	**1,449**	649
git	49,475	350	19	95	**442**	258

- 소프트웨어 결함의 주요 원인이지만 정확한 수정이 까다로움



사례 1: Linux Kernel

```
in = malloc(1);
out = malloc(1);
... // use in, out
free(out);
free(in);

in = malloc(2);
if (in == NULL) {

  goto err;
}

out = malloc(2);
if (out == NULL) {
  free(in);

  goto err;
}
... // use in, out
err:
  free(in);
  free(out);
  return;
```

사례 1: Linux Kernel

```
in = malloc(1);
out = malloc(1);
... // use in, out
free(out);
free(in);

in = malloc(2);
if (in == NULL) {

  goto err;
}
```

메모리 할당

```
out = malloc(2);
if (out == NULL) {
  free(in);

  goto err;
}
... // use in, out
err:
  free(in);
  free(out);
  return;
```

사례 1: Linux Kernel

```
in = malloc(1);
out = malloc(1);
... // use in, out
free(out);
free(in);

in = malloc(2);
if (in == NULL) {

  goto err;
}

out = malloc(2);
if (out == NULL) {
  free(in);

  goto err;
}
... // use in, out
err:
  free(in);
  free(out);
  return;
```

메모리 할당

메모리 해제

사례 1: Linux Kernel

```
in = malloc(1);
out = malloc(1);
... // use in, out
free(out);
free(in);

in = malloc(2);
if (in == NULL) {

  goto err;
}

out = malloc(2);
if (out == NULL) {
  free(in);

  goto err;
}
... // use in, out
err:
  free(in);
  free(out);
  return;
```

메모리 할당

메모리 해제

메모리 중복 해제
(double-free)

사례 1: Linux Kernel

```
in = malloc(1);
out = malloc(1);
... // use in, out
free(out);
free(in);

in = malloc(2);
if (in == NULL) {

  goto err;
}

out = malloc(2);
if (out == NULL) {
  free(in);

  goto err;
}
... // use in, out
err:
  free(in);
  free(out);
  return;
```

메모리 할당

메모리 해제

메모리 중복 해제
(double-free)

사례 1: Linux Kernel

USB: fix double frees in error code paths of ipaq driver

the error code paths can be enter with buffers to freed buffers.
Serial core would do a kfree() on memory already freed.

Signed-off-by: Oliver Neukum <oneukum@suse.de>
Signed-off-by: Greg Kroah-Hartman <gregkh@suse.de>

master v4.15-rc1 ... v2.6.24-rc1

Oliver Neukum committed with **gregkh** on 18 Sep 2007 1 par

```
in = malloc(1);
out = malloc(1);
... // use in, out
free(out);
free(in);

in = malloc(2);
if (in == NULL) {
  out = NULL;
  goto err;
}

out = malloc(2);
if (out == NULL) {
  free(in);
  in = NULL;
  goto err;
}
... // use in, out
err:
  free(in);
  free(out);
  return;
```

사례 1: Linux Kernel

수동 디버깅의 문제 1:
오류가 제거되었는지 확신하기 어려움

```
in = malloc(1);
out = malloc(1);
... // use in, out
free(out);
free(in);

in = malloc(2);
if (in == NULL) {
  out = NULL;
  goto err;
}

out = malloc(2);
if (out == NULL) {
  free(in);
  in = NULL;
  goto err;
}
... // use in, out
err:
  free(in);
  free(out);
  return;
```

사례 1: Linux Kernel

9개월 후에 다시 오류 수정을 시도

```
in = malloc(1);
out = malloc(1);
... // use in, out
free(out);
free(in);

in = malloc(2);
if (in == NULL) {
  out = NULL;
  goto err;
}
free(out);
out = malloc(2);
if (out == NULL) {
  free(in);
  in = NULL;
  goto err;
}
... // use in, out
err:
  free(in);
  free(out);
  return;
```

사례 1: Linux Kernel

memory leak

```
in = malloc(1);
out = malloc(1);
... // use in, out
free(out);
free(in);

in = malloc(2);
if (in == NULL) {
  out = NULL;
  goto err;
}
free(out);
out = malloc(2);
if (out == NULL) {
  free(in);
  in = NULL;
  goto err;
}
... // use in, out
err:
  free(in);
  free(out);
  return;
```

수동 디버깅의 문제 2:
오류 수정 과정에서 새로운 오류가 발생

USB: fix double kfree in ipaq in error case

in the error case the ipaq driver leaves a dangling pointer to already freed memory that will be freed again.

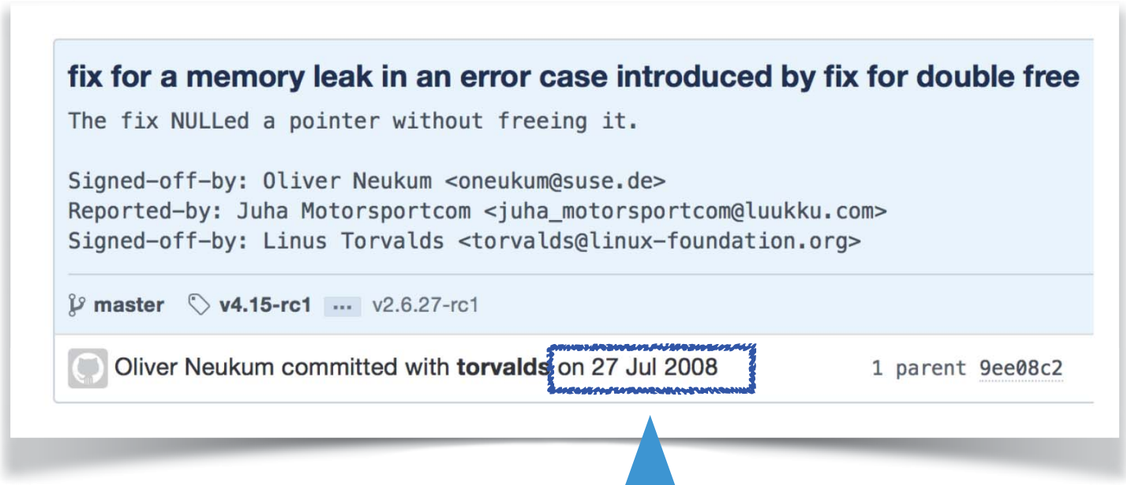
Signed-off-by: Oliver Neukum <oneukum@suse.de>
Signed-off-by: Greg Kroah-Hartman <gregkh@suse.de>

master v4.15-rc1 ... v2.6.27-rc1

Oliver Neukum committed with gregkh on 30 Jun 2008 1 parent 35

9개월 후에 다시 오류 수정을 시도

사례 1: Linux Kernel

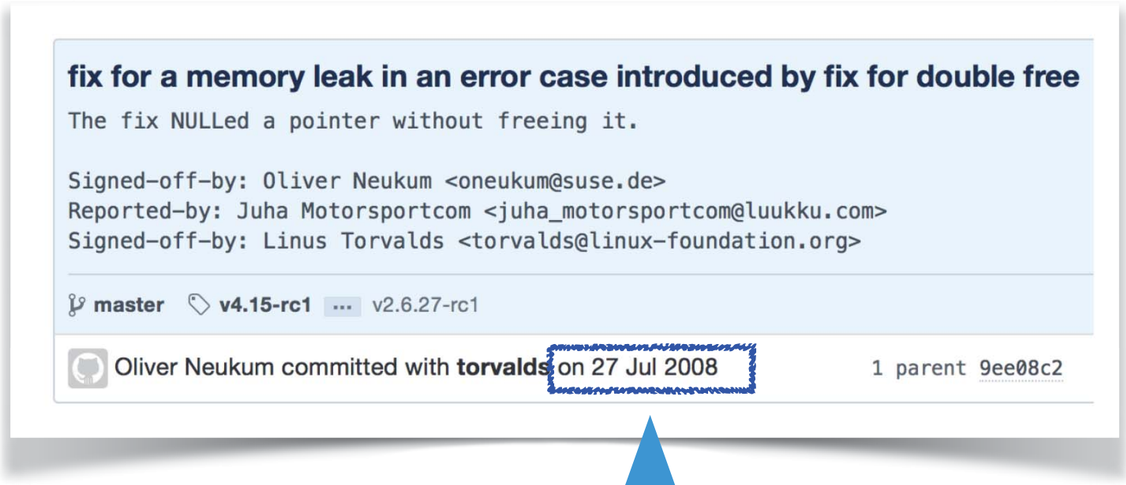


오류 발견에서 수정까지 총 10개월 소요

```
in = malloc(1);
out = malloc(1);
... // use in, out
free(out);
free(in);
out = NULL;
in = malloc(2);
if (in == NULL) {
  out = NULL;
  goto err;
}
free(out);
out = malloc(2);
if (out == NULL) {
  free(in);
  in = NULL;
  goto err;
}
... // use in, out
err:
  free(in);
  free(out);
  return;
```

사례 1: Linux Kernel

수동 디버깅의 문제 3:
오류는 제거했지만 코드 품질이 떨어짐



오류 발견에서 수정까지 총 10개월 소요

```
in = malloc(1);
out = malloc(1);
... // use in, out
free(out);
free(in);
out = NULL;
in = malloc(2);
if (in == NULL) {
  out = NULL;
  goto err;
}
free(out);
out = malloc(2);
if (out == NULL) {
  free(in);
  in = NULL;
  goto err;
}
... // use in, out
err:
  free(in);
  free(out);
  return;
```

ICSE 2020

SAVER: 메모리 오류 자동 수정기

```
in = malloc(1);
out = malloc(1);
... // use in, out
free(out);
free(in);

in = malloc(2);
if (in == NULL) {

  goto err;
}

out = malloc(2);
if (out == NULL) {
  free(in);

  goto err;
}
... // use in, out
err:
  free(in); // double-free
  free(out);// double-free
  return;
```

SAVER

✓개발생산성↑
✓SW품질↑
✓안전성 보장

```
in = malloc(1);
out = malloc(1);
... // use in, out
free(out);
free(in);

in = malloc(2);
if (in == NULL) {

  goto err;
}
free(out);
out = malloc(2);
if (out == NULL) {
  free(in);

  goto err;
}
... // use in, out
err:
  free(in);
  free(out);
  return;
```

사례 2: Memory Leak in Swoole

```
1  int swTableColumn_add(swTable *table, ...) {
2    col = sw_malloc(sizeof(swTableColumn));
3    if (type == SW_TABLE_INT)
4        col->size = 1;
5    col->index = table->size;
6    return swHashMap_add(table->columns, ..., col);
7  }
8
9  int swHashMap_add(swHashMap *hmap, ..., void *data) {
10   node = sw_malloc(sizeof(swHashMap_node));
11   if (node == NULL)
12       return SW_ERR;
13   node->data = data;
14   swHashMap_node_add(hmap, ... node);
15   return SW_OK;
16 }
```

Infer

```
Memory Leak:
An object allocated at line 2
becomes unreachable after line 7
```

사례 2: Memory Leak in Swoole

```
1   int swTableColumn_add(swTable *table, ...) {
2     col = sw_malloc(sizeof(swTableColumn));
3     if (type == SW_TABLE_INT)
4         col->size = 1;
5     col->index = table->size;
6     return swHashMap_add(table->columns, ..., col);
7   }
8
9   int swHashMap_add(swHashMap *hmap, ..., void *data) {
10    node = sw_malloc(sizeof(swHashMap_node));
11    if (node == NULL)
12        return SW_ERR;
13    node->data = data;
14    swHashMap_node_add(hmap, ... node);
15    return SW_OK;
16  }
```

정상 실행 경로

Infer

```
Memory Leak:
An object allocated at line 2
becomes unreachable after line 7
```

사례 2: Memory Leak in Swoole

```
1  int swTableColumn_add(swTable *table, ...) {
2    col = sw_malloc(sizeof(swTableColumn));
3    if (type == SW_TABLE_INT)
4        col->size = 1;
5    col->index = table->size;
6    return swHashMap_add(table->columns, ..., col);
7  }
8
9  int swHashMap_add(swHashMap *hmap, ..., void *data) {
10   node = sw_malloc(sizeof(swHashMap_node));
11   if (node == NULL)
12       return SW_ERR;
13   node->data = data;
14   swHashMap_node_add(hmap, ... node);
15   return SW_OK;
16 }
```

Infer

```
Memory Leak:
An object allocated at line 2
becomes unreachable after line 7
```

사례 2: Memory Leak in Swoole

```
1   int swTableColumn_add(swTable *table, ...) {
2     col = sw_malloc(sizeof(swTableColumn));
3     if (type == SW_TABLE_INT)
4         col->size = 1;
5     col->index = table->size;
6     return swHashMap_add(table->columns, ..., col);
7   }
8
9   int swHashMap_add(swHashMap *hmap, ..., void *data) {
10    node = sw_malloc(sizeof(swHashMap_node));
11    if (node == NULL)
12        return SW_ERR;
13    node->data = data;
14    swHashMap_node_add(hmap, ... node);
15    return SW_OK;
16  }
```

Memory leak

Infer

```
Memory Leak:
An object allocated at line 2
becomes unreachable after line 7
```

사례 2: Memory Leak in Swoole

```
1  int swTableColumn_add(swTable *table, ...) {
2    (  if ((int ret = swHashMap_add(table->columns, ..., col)) == SW_ERR)
3    1      free(p);
4           return ret;
5    col->index = table->size;
6    return swHashMap_add(table->columns, ..., col);
7  }
8
9  int swHashMap_add(swHashMap *hmap, ..., void *data) {
10   node = sw_malloc(sizeof(swHashMap_node));
11   if (node == NULL)
12       return SW_ERR;
13   node->data = data;
14   swHashMap_node_add(hmap, ... node);
15   return SW_OK;
16 }
```

SAVER

Infer

```
Memory Leak:
An object allocated at line 2
becomes unreachable after line 7
```

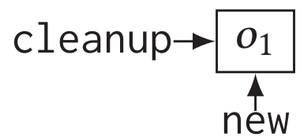
사례 3: Use-After-Free in Binutils

```
1  struct node *cleanup; // list of objects to be deallocated
2  struct node *first = NULL;
3  for (...) {
4      struct node *new = xmalloc(sizeof(*new));
5      make_cleanup(new); // add new to the cleanup list
6      new->name = ...;
7      ...
8      if (...) {
9          first = new;
10
11         continue;
12     }
13   /* potential use-after-free: `first->name` */
14 (-)  if (first == NULL || new->name != first->name)
15
16         continue;
17   do_cleanups(); // deallocate all objects in cleanup
18 }
```

use-after-free

사례 3: Use-After-Free in Binutils

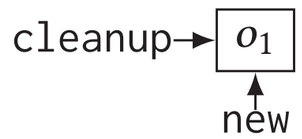
```
1   struct node *cleanup; // list of objects to be deallocated
2   struct node *first = NULL;
3   for (...) {
4       struct node *new = xmalloc(sizeof(*new));
5       make_cleanup(new); // add new to the cleanup list
6       new->name = ...;
7       ...
8       if (...) {
9           first = new;
10
11          continue;
12      }
13    /* potential use-after-free: `first->name` */
14  (-)   if (first == NULL || new->name != first->name)
15
16          continue;
17    do_cleanups(); // deallocate all objects in cleanup
18  }
```

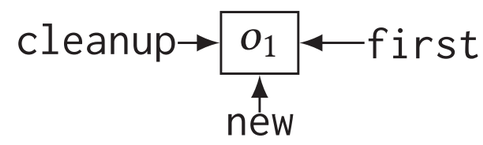


use-after-free

사례 3: Use-After-Free in Binutils

```
1   struct node *cleanup; // list of objects to be deallocated
2   struct node *first = NULL;
3   for (...) {
4       struct node *new = xmalloc(sizeof(*new));
5       make_cleanup(new); // add new to the cleanup list
6       new->name = ...;
7       ...
8       if (...) {
9           first = new;
10
11          continue;
12      }
13    /* potential use-after-free: `first->name` */
14  (-)   if (first == NULL || new->name != first->name)
15
16          continue;
17    do_cleanups(); // deallocate all objects in cleanup
18  }
```



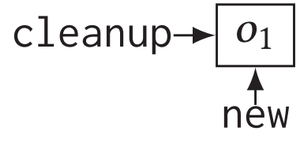


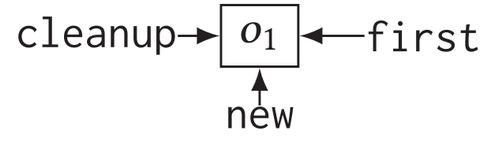
use-after-free

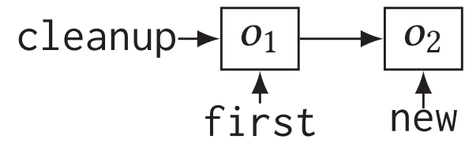
사례 3: Use-After-Free in Binutils

```
1  struct node *cleanup; // list of objects to be deallocated
2  struct node *first = NULL;
3  for (...) {
4      struct node *new = xmalloc(sizeof(*new));
5      make_cleanup(new); // add new to the cleanup list
6      new->name = ...;
7      ...
8      if (...) {
9          first = new;
10
11         continue;
12     }
13   /* potential use-after-free: `first->name` */
14 (-)  if (first == NULL || new->name != first->name)
15
16         continue;
17   do_cleanups(); // deallocate all objects in cleanup
18 }
```

use-after-free



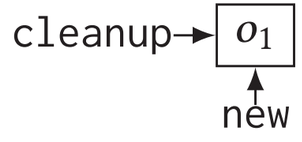


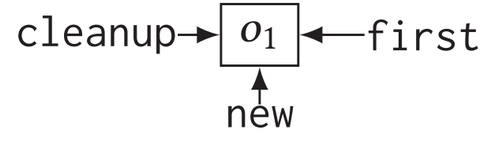


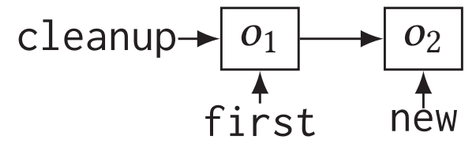
사례 3: Use-After-Free in Binutils

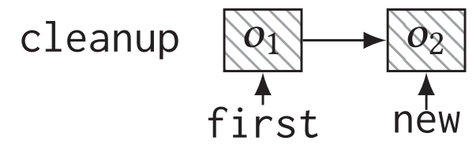
```
1  struct node *cleanup; // list of objects to be deallocated
2  struct node *first = NULL;
3  for (...) {
4      struct node *new = xmalloc(sizeof(*new));
5      make_cleanup(new); // add new to the cleanup list
6      new->name = ...;
7      ...
8      if (...) {
9          first = new;
10
11         continue;
12     }
13   /* potential use-after-free: `first->name` */
14 (-)  if (first == NULL || new->name != first->name)
15
16         continue;
17   do_cleanups(); // deallocate all objects in cleanup
18 }
```

use-after-free





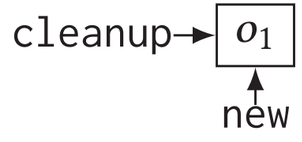


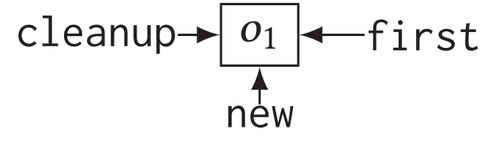


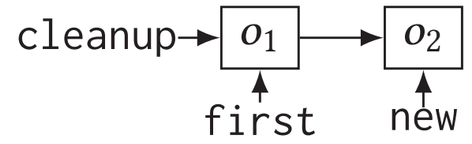
사례 3: Use-After-Free in Binutils

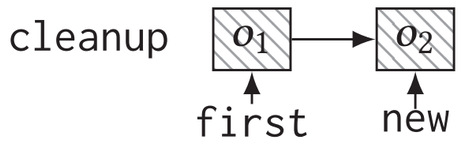
```
1   struct node *cleanup; // list of objects to be deallocated
2   struct node *first = NULL;
3   for (...) {
4       struct node *new = xmalloc(sizeof(*new));
5       make_cleanup(new); // add new to the cleanup list
6       new->name = ...;
7       ...
8       if (...) {
9           first = new;
10
11          continue;
12      }
13    /* potential use-after-free: `first->name` */
14  (-)   if (first == NULL || new->name != first->name)
15
16          continue;
17    do_cleanups(); // deallocate all objects in cleanup
18  }
```

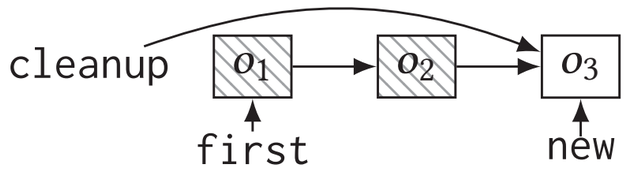
use-after-free











SAVER가 생성한 패치

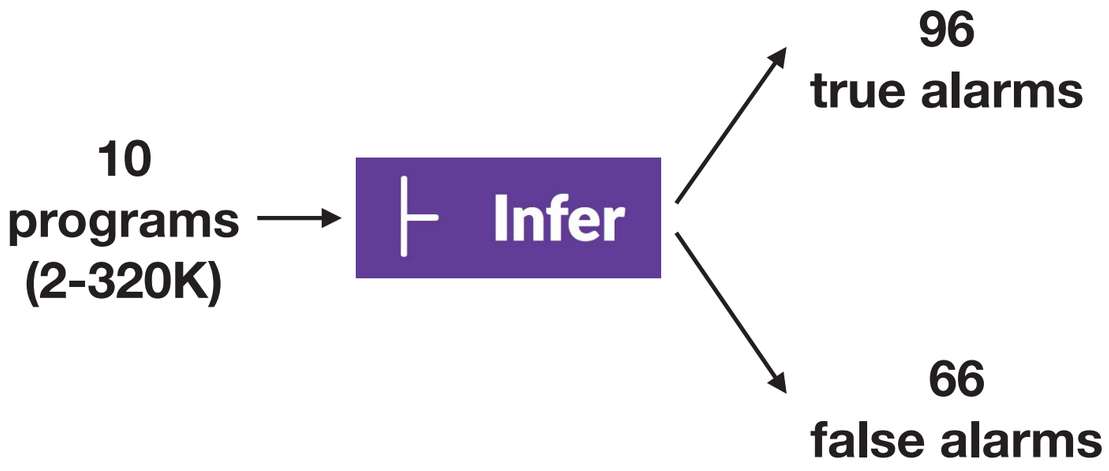
```
1   struct node *cleanup; // list of objects to be deallocated
2   struct node *first = NULL;
3   for (...) {
4       struct node *new = xmalloc(sizeof(*new));
5       make_cleanup(new); // add new to the cleanup list
6       new->name = ...;
7       ...
8       if (...) {
9           first = new;
(+)         tmp = first->name;
10          continue;
11      }
12    /* potential use-after-free: `first->name` */
(-)   if (first == NULL || new->name != first->name)
(+)   if (first == NULL || new->name != tmp)
13          continue;
14    do_cleanups(); // deallocate all objec
15  }
```

임시 변수를 도입해서 해제 되기 전에 필요한 값을 기억

(포인터 접근없이) 필요한 값을 읽음

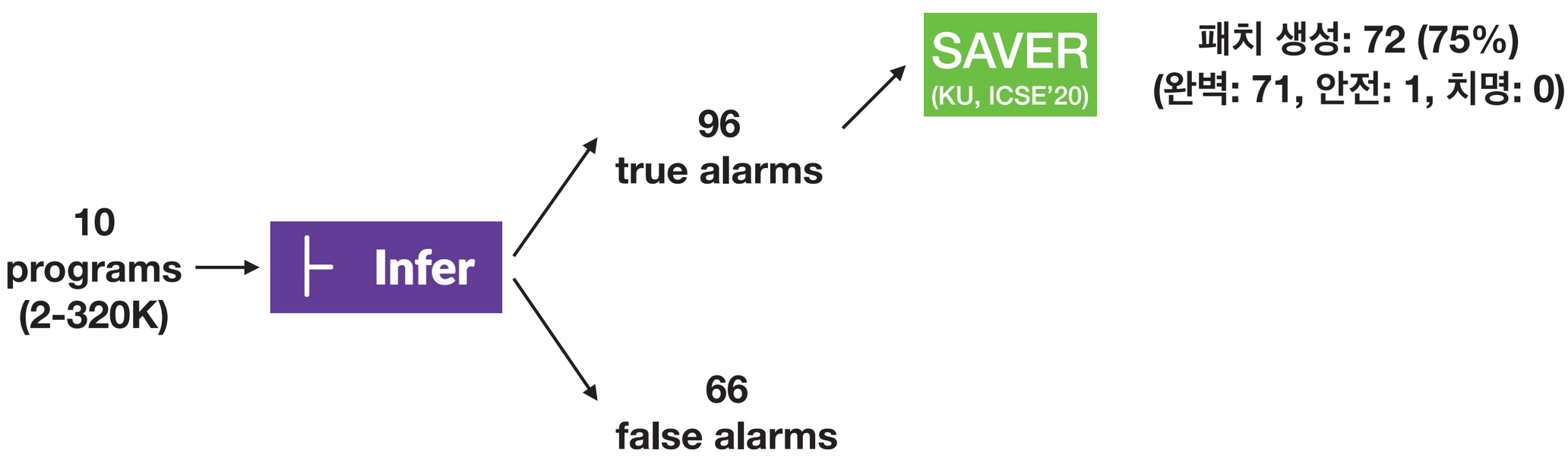
SAVER 성능

		Infer				SAVER						FootPatch [60]						
Program	**kLoC**	**#T**	**#F**	**Pre(s)**	**Fix(s)**	G_T	$✓_T$	$△_T$	$✗_T$	G_F	$✗_F$	**Fix(s)**	G_T	$✓_T$	$△_T$	$✗_T$	G_F	$✗_F$
rappel (ad8efd7)	2.2	1	0	2.2	0.0	1	1	0	0	0	0	8.9	1	1	0	0	0	0
flex (d3de49f)	22.3	3	4	26.3	2.5	0	0	0	0	0	0	51.0	0	0	0	0	1	1
WavPack (22977b2)	31.2	1	2	44.6	22.1	0	0	0	0	0	0	67.9	0	0	0	0	2	2
Swoole (a4256e4)	43.0	15	3	88.5	10.1	11	11	0	0	0	0	392.5	9	7	0	2	1	1
lxc (72cc48f)	49.9	3	5	230.6	5.8	3	3	0	0	0	0	179.6	0	0	0	0	1	1
p11-kit (ead7ara)	62.9	33	9	646.2	288.8	24	24	0	0	0	0	566.4	8	7	1	0	2	2
x264 (d4099dd)	73.2	10	0	144.3	9.9	10	10	0	0	0	0	426.9	2	2	0	0	0	0
recutils-1.8	92.0	10	11	144.1	44.4	8	8	0	0	0	0	662.2	3	2	1	0	0	0
inetutils-1.9.4	116.9	4	5	44.8	2.5	4	4	0	0	0	0	182.1	0	0	0	0	0	0
snort-2.9.13	320.8	16	27	2372.0	216.0	11	10	1	0	0	0	4636.4	3	0	0	3	19	18
Total	814.4	96	66	3743.6	602.1	72	71	1	0	0	0	7173.9	26	19	2	5	26	25



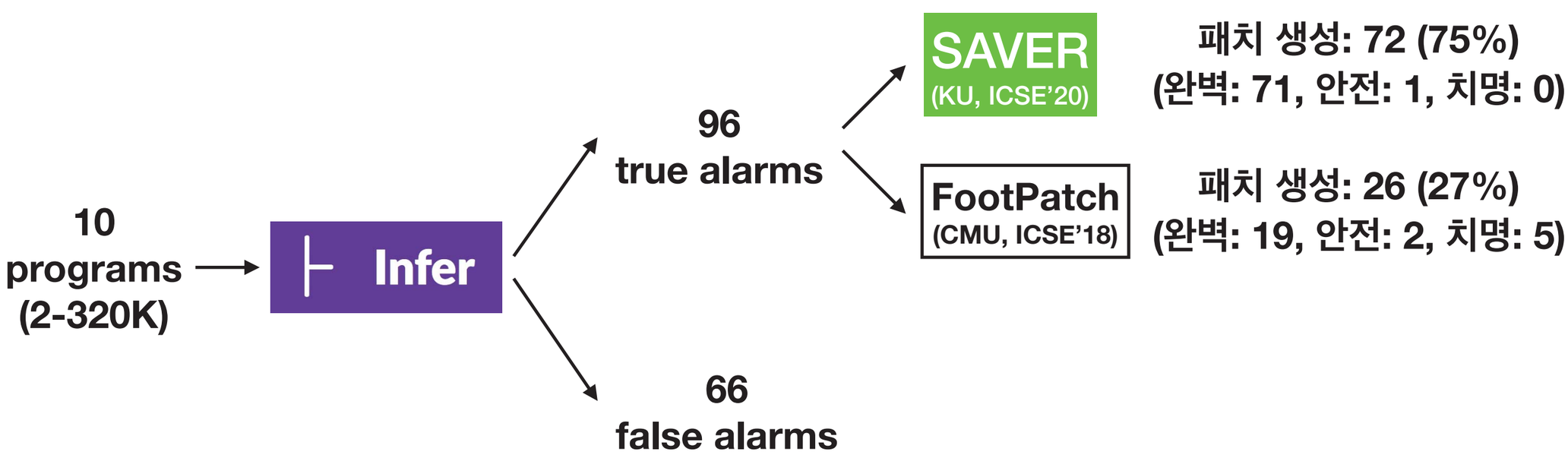
SAVER 성능

		Infer				SAVER						FootPatch [60]						
Program	**kLoC**	**#T**	**#F**	**Pre(s)**	**Fix(s)**	G_T	$✓_T$	$△_T$	$✗_T$	G_F	$✗_F$	**Fix(s)**	G_T	$✓_T$	$△_T$	$✗_T$	G_F	$✗_F$
rappel (ad8efd7)	2.2	1	0	2.2	0.0	1	1	0	0	0	0	8.9	1	1	0	0	0	0
flex (d3de49f)	22.3	3	4	26.3	2.5	0	0	0	0	0	0	51.0	0	0	0	0	1	1
WavPack (22977b2)	31.2	1	2	44.6	22.1	0	0	0	0	0	0	67.9	0	0	0	0	2	2
Swoole (a4256e4)	43.0	15	3	88.5	10.1	11	11	0	0	0	0	392.5	9	7	0	2	1	1
lxc (72cc48f)	49.9	3	5	230.6	5.8	3	3	0	0	0	0	179.6	0	0	0	0	1	1
p11-kit (ead7ara)	62.9	33	9	646.2	288.8	24	24	0	0	0	0	566.4	8	7	1	0	2	2
x264 (d4099dd)	73.2	10	0	144.3	9.9	10	10	0	0	0	0	426.9	2	2	0	0	0	0
recutils-1.8	92.0	10	11	144.1	44.4	8	8	0	0	0	0	662.2	3	2	1	0	0	0
inetutils-1.9.4	116.9	4	5	44.8	2.5	4	4	0	0	0	0	182.1	0	0	0	0	0	0
snort-2.9.13	320.8	16	27	2372.0	216.0	11	10	1	0	0	0	4636.4	3	0	0	3	19	18
Total	814.4	96	66	3743.6	602.1	72	71	1	0	0	0	7173.9	26	19	2	5	26	25



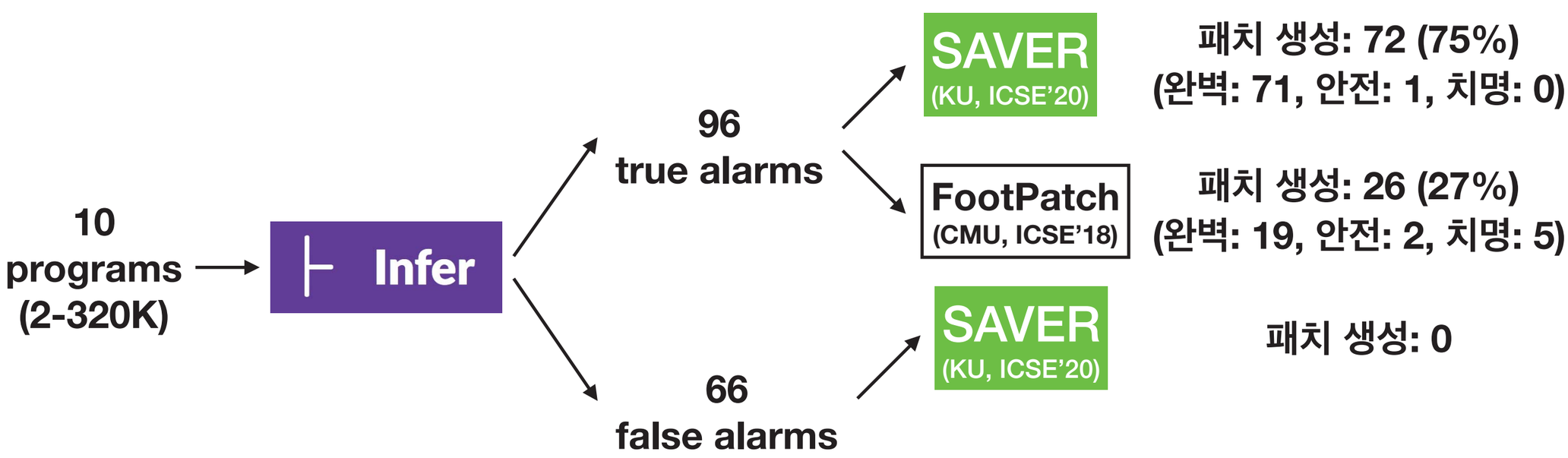
SAVER 성능

		Infer				SAVER						FootPatch [60]						
Program	**kLoC**	**#T**	**#F**	**Pre(s)**	**Fix(s)**	G_T	$✓_T$	$△_T$	$✗_T$	G_F	$✗_F$	**Fix(s)**	G_T	$✓_T$	$△_T$	$✗_T$	G_F	$✗_F$
rappel (ad8efd7)	2.2	1	0	2.2	0.0	1	1	0	0	0	0	8.9	1	1	0	0	0	0
flex (d3de49f)	22.3	3	4	26.3	2.5	0	0	0	0	0	0	51.0	0	0	0	0	1	1
WavPack (22977b2)	31.2	1	2	44.6	22.1	0	0	0	0	0	0	67.9	0	0	0	0	2	2
Swoole (a4256e4)	43.0	15	3	88.5	10.1	11	11	0	0	0	0	392.5	9	7	0	2	1	1
lxc (72cc48f)	49.9	3	5	230.6	5.8	3	3	0	0	0	0	179.6	0	0	0	0	1	1
p11-kit (ead7ara)	62.9	33	9	646.2	288.8	24	24	0	0	0	0	566.4	8	7	1	0	2	2
x264 (d4099dd)	73.2	10	0	144.3	9.9	10	10	0	0	0	0	426.9	2	2	0	0	0	0
recutils-1.8	92.0	10	11	144.1	44.4	8	8	0	0	0	0	662.2	3	2	1	0	0	0
inetutils-1.9.4	116.9	4	5	44.8	2.5	4	4	0	0	0	0	182.1	0	0	0	0	0	0
snort-2.9.13	320.8	16	27	2372.0	216.0	11	10	1	0	0	0	4636.4	3	0	0	3	19	18
Total	814.4	96	66	3743.6	602.1	72	71	1	0	0	0	7173.9	26	19	2	5	26	25



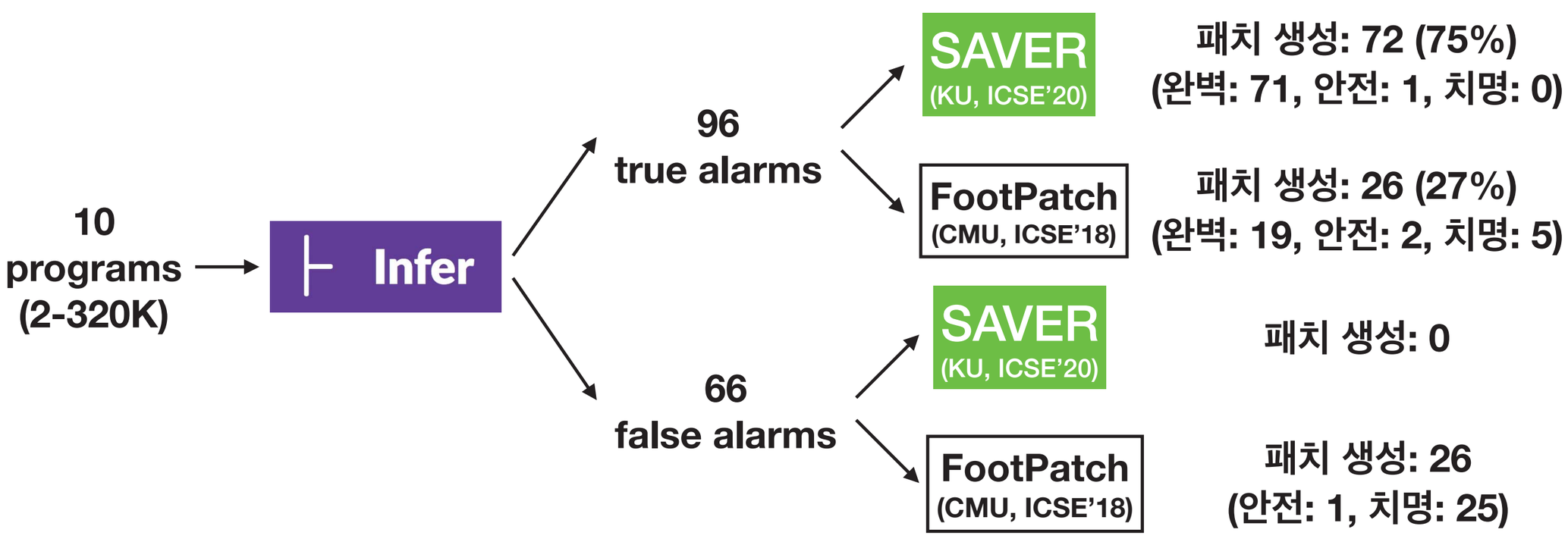
SAVER 성능

		Infer				SAVER						FootPatch [60]						
Program	**kLoC**	**#T**	**#F**	**Pre(s)**	**Fix(s)**	G_T	$✓_T$	$△_T$	$✗_T$	G_F	$✗_F$	**Fix(s)**	G_T	$✓_T$	$△_T$	$✗_T$	G_F	$✗_F$
rappel (ad8efd7)	2.2	1	0	2.2	0.0	1	1	0	0	0	0	8.9	1	1	0	0	0	0
flex (d3de49f)	22.3	3	4	26.3	2.5	0	0	0	0	0	0	51.0	0	0	0	0	1	1
WavPack (22977b2)	31.2	1	2	44.6	22.1	0	0	0	0	0	0	67.9	0	0	0	0	2	2
Swoole (a4256e4)	43.0	15	3	88.5	10.1	11	11	0	0	0	0	392.5	9	7	0	2	1	1
lxc (72cc48f)	49.9	3	5	230.6	5.8	3	3	0	0	0	0	179.6	0	0	0	0	1	1
p11-kit (ead7ara)	62.9	33	9	646.2	288.8	24	24	0	0	0	0	566.4	8	7	1	0	2	2
x264 (d4099dd)	73.2	10	0	144.3	9.9	10	10	0	0	0	0	426.9	2	2	0	0	0	0
recutils-1.8	92.0	10	11	144.1	44.4	8	8	0	0	0	0	662.2	3	2	1	0	0	0
inetutils-1.9.4	116.9	4	5	44.8	2.5	4	4	0	0	0	0	182.1	0	0	0	0	0	0
snort-2.9.13	320.8	16	27	2372.0	216.0	11	10	1	0	0	0	4636.4	3	0	0	3	19	18
Total	814.4	96	66	3743.6	602.1	72	71	1	0	0	0	7173.9	26	19	2	5	26	25



SAVER 성능

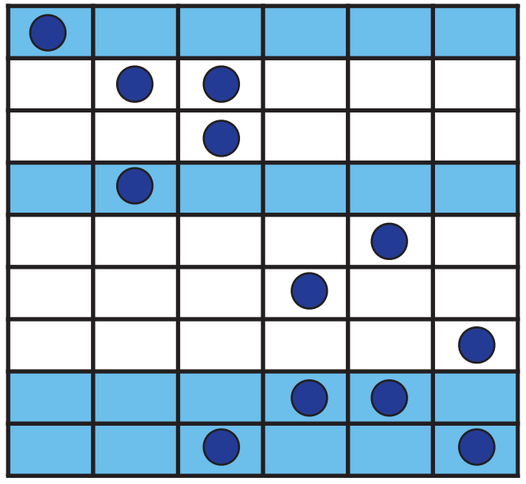
		Infer				SAVER						FootPatch [60]						
Program	**kLoC**	**#T**	**#F**	**Pre(s)**	**Fix(s)**	G_T	$✓_T$	$△_T$	$✗_T$	G_F	$✗_F$	**Fix(s)**	G_T	$✓_T$	$△_T$	$✗_T$	G_F	$✗_F$
rappel (ad8efd7)	2.2	1	0	2.2	0.0	1	1	0	0	0	0	8.9	1	1	0	0	0	0
flex (d3de49f)	22.3	3	4	26.3	2.5	0	0	0	0	0	0	51.0	0	0	0	0	1	1
WavPack (22977b2)	31.2	1	2	44.6	22.1	0	0	0	0	0	0	67.9	0	0	0	0	2	2
Swoole (a4256e4)	43.0	15	3	88.5	10.1	11	11	0	0	0	0	392.5	9	7	0	2	1	1
lxc (72cc48f)	49.9	3	5	230.6	5.8	3	3	0	0	0	0	179.6	0	0	0	0	1	1
p11-kit (ead7ara)	62.9	33	9	646.2	288.8	24	24	0	0	0	0	566.4	8	7	1	0	2	2
x264 (d4099dd)	73.2	10	0	144.3	9.9	10	10	0	0	0	0	426.9	2	2	0	0	0	0
recutils-1.8	92.0	10	11	144.1	44.4	8	8	0	0	0	0	662.2	3	2	1	0	0	0
inetutils-1.9.4	116.9	4	5	44.8	2.5	4	4	0	0	0	0	182.1	0	0	0	0	0	0
snort-2.9.13	320.8	16	27	2372.0	216.0	11	10	1	0	0	0	4636.4	3	0	0	3	19	18
Total	814.4	96	66	3743.6	602.1	72	71	1	0	0	0	7173.9	26	19	2	5	26	25



SAVER 작동 원리

```
in = malloc(1);
out = malloc(1);
... // use in, out
free(out);
free(in);

in = malloc(2);
if (in == NULL) {
  goto err;
}

out = malloc(2);
if (out == NULL) {
  free(in);

  goto err;
}
... // use in, out
err:
  free(in); // double-free
  free(out);// double-free
  return;
```

정적 분석



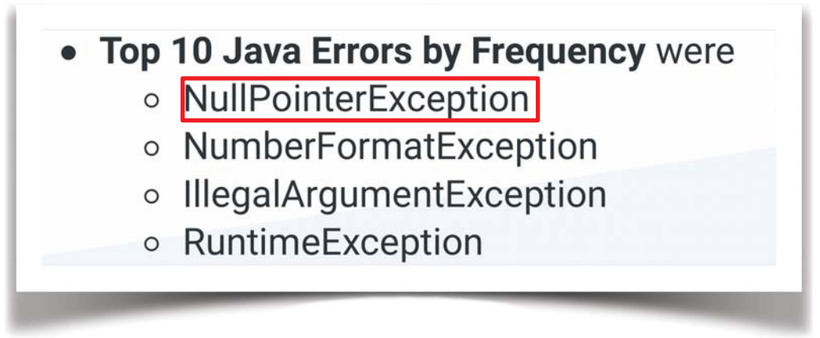
Exact Cover Problem
(NP-complete)

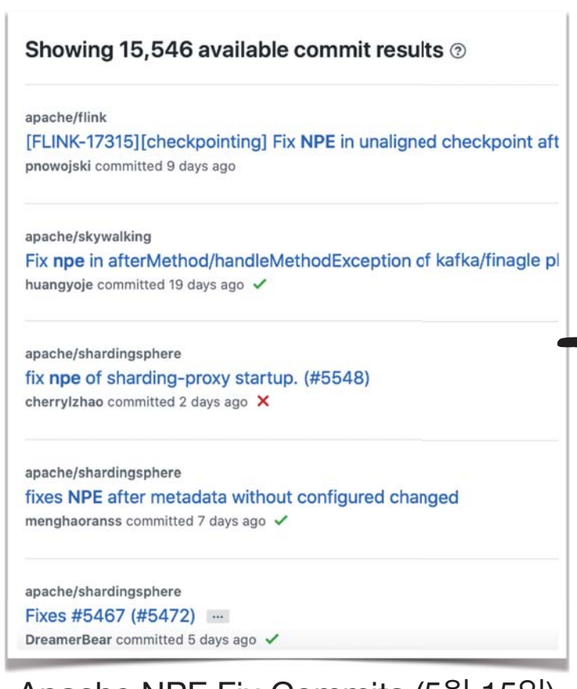
SMT

```
in = malloc(1);
out = malloc(1);
... // use in, out
free(out);
free(in);

in = malloc(2);
if (in == NULL) {
  goto err;
}
free(out);
out = malloc(2);
if (out == NULL) {
  free(in);

  goto err;
}
... // use in, out
err:
  free(in);
  free(out);
  return;
```

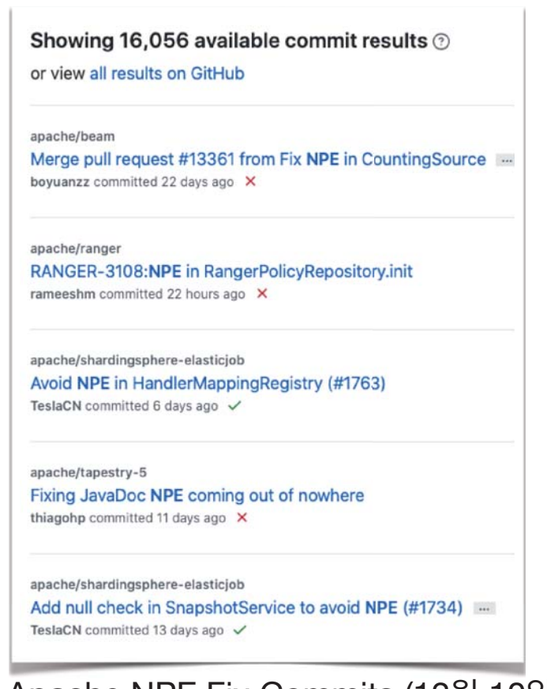
Null Pointer Exceptions (NPEs)

- Java에서 가장 흔히 발생하는 오류
  - 안드로이드 exception 가운데 약 40%
  - Mozila, Apache 메모리 오류 중 약 37%





Apache NPE Fix Commits (5월 15일)



Apache NPE Fix Commits (12월 10일)

Apache 재단에서 1년동안 약 1000건의 NPE 수정

챌린지: 올바른 NPE 패치?

```
public StrBuilder appendFixedWidthPad(Object obj, int width, char padChar)
{
        String str = (obj == null ? getNullText() : obj.toString());
        int strLen = str.length();
        ...
        this.size += width;
        return this;
}
```

Null Pointer Exception

챌린지: 올바른 NPE 패치?

```
String str = (obj == null ? getNullText() : obj.toString());
If (str != null) {
        int strLen = str.length();
        ...
}
this.size += width;
return this;
```

```
        ...
```

Null Pointer Exception

```
        this.size += width;
        return this;
    }
```

챌린지: 올바른 NPE 패치?

```
String str = (obj == null ? getNullText() : obj.toString());
If (str != null) {
}
this.si
return
```

```
String str = (obj == null ? getNullText() : obj.toString());
if (str == null)
        return null;
int strLen = str.length();
...
this.size += width;
return this;
```

```
this.size += width;
return this;
}
```

```
char padChar)
```

챌린지: 올바른 NPE 패치?

```
String str = (obj == null ? getNullText() : obj.toString());
If (str != null) {
}
this.si
return
```

```
String str = (obj == null ? getNullText() : obj.toString());
if (str
int str
...
this.si
return
this.siz
return
```

```
String str = (obj == null ? getNullText() : obj.toString());
if (str == null)
        return new StrBuilder();
int strLen = str.length();
...
this.size += width;
return this;
```

```
char padChar)
}
```

챌린지: 올바른 NPE 패치?

```
String str = (obj == null ? getNullText() : obj.toString());
If (str != null) {
}
this.si
return
```

```
String str = (obj == null ? getNullText() : obj.toString());
if (str
int str
...
this.si
return
```

```
String str = (obj == null ? getNullText() : obj.toString());
if (str == null)
int str
...
this.si
return
```

```
String str = (obj == null ? getNullText() : obj.toString());
if (str == null)
        str = new String();
int strLen = str.length();
...
this.size += width;
return this;
```

```
char padChar)
}
```

챌린지: 올바른 NPE 패치?

```
String str = (obj == null ? getNullText() : obj.toString());
If (str != null) {
}
this.si
return
}
```

```
String str = (obj == null ? getNullText() : obj.toString());
if (str
int str
...
return
return
```

```
String str = (obj == null ? getNullText() : obj.toString());
if (str == null)
int str
...
this.si
return
```

```
String str = (obj == null ? getNullText() : obj.toString());
if (str == null)
        str = new String();
int strLen = str.length();
...
this.size += width;
return this;
```

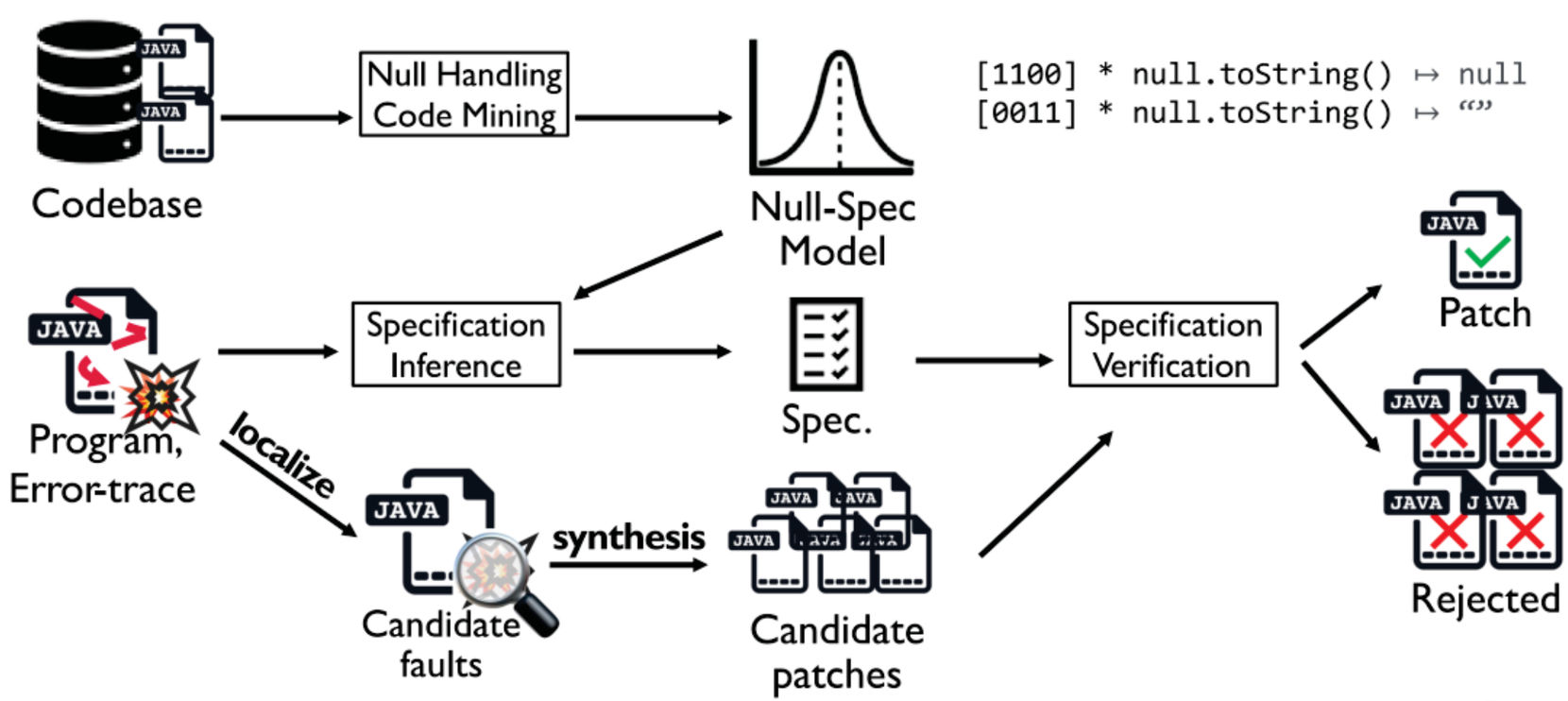
char padChar)

기존 기술: 오류 수정 명세로 테스트 케이스가 주어진다고 가정
=> NPEX: Repairing NPEs without Tests

ICSE 2022

NPEX: Java NPE 자동 수정기

- **정적 분석** 및 **코드 마이닝**을 통해 오류 수정 명세 자동 추론



OCaml 프로그래밍 과제에 응용

- Arithmetic expressions:

```
type aexp =
  | Const of int
  | Var of string
  | Power of (string * int)
  | Sum of aexp list
  | Times of aexp list
```

- Symbolic differentiation:

```
let rec diff (e, x) =
match e with
| Const n -> Const 0
| Var y -> if (x <> y) then Const 0 else Const 1
| Power (y, n) -> if (x <> y) then Const 0 else Times [Const n; Power (y, n-1)]
| Sum lst ->
  (match lst with
   | [hd] -> diff (hd, x)
   | hd::tl -> Sum [diff (hd, x); diff (Sum tl, x)])
| Times lst ->
  (match lst with
   | [hd] -> diff (hd, x)
   | hd::tl -> Sum [Times ((diff (hd, x))::tl); Times[hd; diff (Times tl, x)]])
```

OCaml 프로그래밍 과제에 응용

OCaml 프로그래밍 과제에 응용

```
let rec diff : aexp * string -> aexp
= fun (exp, x) ->
  match exp with
    | Const(i) -> Const(0)
    | Var(s) -> if s=x then Const(1) else Var(s)
    | Power(s, i) ->
      if s=x then if i>2 then Times[Const(i); Power(s, i-1)] else Times[Const(i); Var(s)]
           else Power(s, i)
    | Times(al) ->
      let rec timeiter lst =
        match lst with
          |[] -> []
          |hd::tl ->
           (match hd with
             |Const(i1) -> Const(i1)
             |_ -> diff(hd, x)
           )::(timeiter tl)
      in
      Times(timeiter al)
  | Sum(al) ->
    let rec sumeval lst =
      match lst with
        |[] -> []
        |hd::tl -> if hd=Const(0) then sumeval tl else hd::sumeval tl
    in
    let rec sumiter lst =
      match lst with
        |[] -> []
        |hd::tl -> diff(hd, x)::(sumiter tl)
    in
    Sum(sumeval(sumiter al))
```

OCaml 프로그래밍 과제에 응용

```
let rec diff : aexp * string -> aexp
= fun (exp, x) ->
  match exp with
    | Const(i) -> Const(0)
    | Var(s) -> if s=x then Const(1) else Var(s)
    | Power(s, i) ->
      if s=x then if i>2 then Times[Const(i); Power(s, i-1)] else Times[Const(i); Var(s)]
          else Power(s, i)
    | Times(al) ->
      let rec timeiter lst =
        match lst with
          |[] -> []
          |hd::tl ->
           (match hd with
             |Const(i1) -> Const(i1)
             |_ -> diff(hd, x)
           )::(timeiter tl)
      in
      Times(timeiter al)
    | Sum(al) ->
      let rec sumeval lst =
        match lst with
          |[] -> []
          |hd::tl -> if hd=Const(0) then sumeval tl else hd::sumeval tl
      in
      let rec sumiter lst =
        match lst with
          |[] -> []
          |hd::tl -> diff(hd, x)::(sumiter tl)
      in
      Sum(sumeval(sumiter al))
```

```
let rec num_x = fun lst s->
  if lst = [] then 0 else
    match List.hd lst with
      |Times a' -> num_x a' s + num_x (List.tl lst) s
      |Power(a',b') -> if a' = s && b'<> 0 then b' + num_x (List.tl lst) s else 0 +num_x
           (List.tl lst) s
      |Var a' -> if a'=s then 1 + num_x (List.tl lst) s else 0 +num_x (List.tl lst) s
      |Const a' -> if a'=0 then 0 else 0 +num_x (List.tl lst) s;;

let rec dif_times_lst = fun lst x ->
  match List.hd lst with
    |Const x' -> [Const x']@dif_times_lst (List.tl lst) x
    |Var x' -> if x' = x then (List.tl lst) else [Var x']@dif_times_lst (List.tl lst) x
    |_->[];;

let rec dif_Times = fun lst x ->
  match num_x lst x with
    |0-> [Const 0]
    |1-> dif_times_lst lst x
    |_->[Const (num_x lst x)]@dif_times_lst lst x;;

let rec dif_sums_lst = fun lst x ->
  match lst with
    |[]->[]
    |_-> match List.hd lst with
         |Const x' -> [Const 0]@dif_sums_lst (List.tl lst) x
         |Var x' -> if x'= x then [Const 1]@ (List.tl lst) else [Const 0]@dif_sums_lst (List.tl
              lst) x
         |Sum a' -> if num_x a' x =0 then [Const 0] else dif_sums_lst a' x @ dif_sums_lst
              (List.tl lst) x
         |Power(a',b')-> if a' = x then match b' with
                                  |1->[Const 1]@dif_sums_lst (List.tl lst) x
                                  |_->[Times[Const b';Power(a',b'-1)]]@dif_sums_lst
                                       (List.tl lst) x
                    else [Const 0];;
```

```
let rec dif_all_lst = fun lst x ->
  match lst with
    |[]->[]
    |_->match List.hd lst with
      |Const x' -> [Const 0]@dif_all_lst (List.tl lst) x
      |Var x' -> if x'= x then [Const 1]@ (List.tl lst) else [Const 0]@dif_all_lst (List.tl
           lst) x
      |Sum a' -> if num_x a' x =0 then [Const 0] else dif_sums_lst a' x @ dif_all_lst (List.tl
           lst) x
      |Power(a',b')-> if a' = x then match b' with
                                  |1->[Const 1]@dif_all_lst (List.tl lst) x
                                  |_->[Times[Const b';Power(a',b'-1)]]@dif_all_lst (List.tl
                                       lst) x
                     else [Const 0]
      |Times a' -> [Times(dif_Times a' x)]

let rec diff : aexp * string -> aexp
= fun (exp, x) ->
  match exp with
    |Const x' -> Const 0
    |Var x' -> if x'= x then Const 1 else Const 0
    |Sum a' -> Sum (dif_all_lst a' x)
    |Power(a',b')-> if a' = x then match b' with
                                  |1->Const 1
                                  |_->Times[Const b';Power(a',b'-1)]
                   else Const 0
    |Times a' -> Times (dif_all_lst a' x);;
```

OCaml 프로그래밍 과제에 응용

```
let rec diff : aexp * string -> aexp
= fun (exp, x) ->
 match exp with
   | Const(i) -> Const(0)
   | Var(s) -> if s=x then Const(1) else Var(s)
   | Power(s, i) ->
     if s=x then if i>2 then Times[Const(i); Power(s, i-1)] else Times[Const(i); Var(s)]
          else Power(s, i)
   | Times(al) ->
     let rec timeiter lst =
       match lst with
         |[] -> []
         |hd::tl ->
          (match hd with
            |Const(i1) -> Const(i1)
            |_ -> diff(hd, x)
          )::(timeiter tl)
     in
     Times(timeiter al)
 | Sum(al) ->
     let rec sumeval lst =
       match lst with
         |[] -> []
         |hd::tl -> if hd=Const(0) then sumeval tl else hd::sumeval tl
     in
     let rec sumiter lst =
       match lst with
         |[] -> []
         |hd::tl -> diff(hd, x)::(sumiter tl)
     in
     Sum(sumeval(sumiter al))
```

FixML:

((Sum lst)::tl)

```
let rec num_x = fun lst s->
 if lst = [] then 0 else
   match List.hd lst with
     |Times a' -> num_x a' s + num_x (List.tl lst) s
     |Power(a',b') -> if a' = s && b'<> 0 then b' + num_x (List.tl lst) s else 0 +num_x
          (List.tl lst) s
     |Var a' -> if a'=s then 1 + num_x (List.tl lst) s else 0 +num_x (List.tl lst) s
     |Const a' -> if a'=0 then 0 else 0 +num_x (List.tl lst) s;;

let rec dif_times_lst = fun lst x ->
 match List.hd lst with
   |Const x' -> [Const x']@dif_times_lst (List.tl lst) x
   |Var x' -> if x' = x then (List.tl lst) else [Var x']@dif_times_lst (List.tl lst) x
   |_->[];;

let rec dif_Times = fun lst x ->
 match num_x lst x with
   |0-> [Const 0]
   |1-> dif_times_lst lst x
   |_->[Const (num_x lst x)]@dif_times_lst lst x;;

let rec dif_sums_lst = fun lst x ->
 match lst with
   |[]->[]
   |_-> match List.hd lst with
        |Const x' -> [Const 0]@dif_sums_lst (List.tl lst) x
        |Var x' -> if x'= x then [Const 1]@ (List.tl lst) else [Const 0]@dif_sums_lst (List.tl
             lst) x
        |Sum a' -> if num_x a' x =0 then [Const 0] else dif_sums_lst a' x @ dif_sums_lst
             (List.tl lst) x
        |Power(a',b')-> if a' = x then match b' with
                                |1->[Const 1]@dif_sums_lst (List.tl lst) x
                                |_->[Times[Const b';Power(a',b'-1)]]@dif_sums_lst
                                     (List.tl lst) x
                  else [Const 0];;
```

```
let rec dif_all_lst = fun lst x ->
 match lst with
   |[]->[]
   |_->match List.hd lst with
     |Const x' -> [Const 0]@dif_all_lst (List.tl lst) x
     |Var x' -> if x'= x then [Const 1]@ (List.tl lst) else [Const 0]@dif_all_lst (List.tl
          lst) x
     |Sum a' -> if num_x a' x =0 then [Const 0] else dif_sums_lst a' x @ dif_all_lst (List.tl
          lst) x
     |Power(a',b')-> if a' = x then match b' with
                                |1->[Const 1]@dif_all_lst (List.tl lst) x
                                |_->[Times[Const b';Power(a',b'-1)]]@dif_all_lst (List.tl
                                     lst) x
                  else [Const 0]
     |Times a' -> [Times(dif_Times a' x)]

let rec diff : aexp * string -> aexp
= fun (exp, x) ->
 match exp with
   |Const x' -> Const 0
   |Var x' -> if x'= x then Const 1 else Const 0
   |Sum a' -> Sum (dif_all_lst a' x)
   |Power(a',b')-> if a' = x then match b' with
                                |1->Const 1
                                |_->Times[Const b';Power(a',b'-1)]
                  else Const 0
   |Times a' -> Times (dif_all_lst a' x);;
```

```
type aexp =
   |CONST of int
   | VAR of string
   | POWER of string * int
   | TIMES of aexp list
   | SUM of aexp list

type env = (string * int * int) list

let diff : aexp * string -> aexp
= fun (aexp, x) ->

   let rec deployEnv : env -> int -> aexp list
   = fun env flag ->
   match env with
   | hd::tl ->
   (
    match hd with
    |(x, c, p) ->
       if (flag = 0 && c = 0) then deployEnv tl flag
       else if (x = "const" && flag = 1 && c = 1) then deployEnv tl flag
       else if (p = 0) then (CONST c)::(deployEnv tl flag)
       else if (c = 1 && p = 1) then (VAR x)::(deployEnv tl flag)
       else if (p = 1) then TIMES[CONST c; VAR x]::(deployEnv tl flag)
       else if (c = 1) then POWER(x, p)::(deployEnv tl flag)
       else TIMES [CONST c; POWER(x, p)]::(deployEnv tl flag)
   )
   | [] -> []
   in

   let rec updateEnv : (string * int * int) -> env -> int -> env
   = fun elem env flag ->
   match env with
   | (hd::tl) ->
     (
     match hd with
     | (x, c, p) ->
       (
       match elem with
       |(x2, c2, p2) ->
         if (flag = 0) then
           if (x = x2 && p = p2) then (x, (c + c2), p)::tl
           else hd::(updateEnv elem tl flag)
         else
           if (x = x2) then (x, (c*c2), (p + p2))::tl
           else hd::(updateEnv elem tl flag)
       )
     )
   | [] -> elem::[]
   in

   let rec doDiff : aexp * string -> aexp
   = fun (aexp, x) ->
   match aexp with
   | CONST _ -> CONST 0
   | VAR v ->
     if (x = v) then CONST 1
     else CONST 0
   | POWER (v, p) ->
     if (p = 0) then CONST 0
     else if (x = v) then TIMES ((CONST p)::POWER (v, p-1)::[])
     else CONST 0
   | TIMES lst ->
     (
       match lst with
       (
         match (hd, diff_hd, tl, diff_tl) with
         | (CONST p, CONST s, [CONST r], CONST q) -> CONST (p*q + r*s)
         | (CONST p, _, _, CONST q) ->
           if (diff_hd = CONST 0 || tl = [CONST 0]) then CONST (p*q)
           else SUM [CONST(p*q); TIMES(diff_hd::tl)]
         | (_, CONST s, [CONST r], _) ->
           if (hd = CONST 0 || diff_tl = CONST 0) then CONST (r*s)
           else  SUM [TIMES [hd; diff_tl]; CONST(r*s)]
         | _ ->
           if (hd = CONST 0 || diff_tl = CONST 0) then TIMES(diff_hd::tl)
           else if (tl = [CONST 0] || diff_hd = CONST 0) then TIMES [hd; diff_tl]
           else SUM [TIMES [hd; diff_tl]; TIMES (diff_hd::tl)]
       )
     | [] -> CONST 0
     )
   | SUM lst -> SUM(List.map (fun aexp -> doDiff(aexp, x)) lst)
   in

   let rec simplify : aexp -> env -> int -> aexp list
   = fun aexp env flag ->
   match aexp with
   | SUM lst ->
     (
       match lst with
       | (CONST c)::tl -> simplify (SUM tl) (updateEnv ("const", c, 0) env 0) 0
       | (VAR x)::tl -> simplify (SUM tl) (updateEnv (x, 1, 1) env 0) 0
       | (POWER (x, p))::tl -> simplify (SUM tl) (updateEnv (x, 1, p) env 0) 0
       | (SUM lst)::tl -> simplify (SUM (List.append lst tl)) env 0
       | (TIMES lst)::tl ->
         (
           let l = simplify (TIMES lst) [] 1 in
           match l with
           | h::t ->
             if (t = []) then List.append l (simplify (SUM tl) env 0)
             else List.append (TIMES l::[]) (simplify (SUM tl) env 0)
           | [] -> []
         )
       | [] -> deployEnv env 0
     )
   | TIMES lst ->
     (
       match lst with
       | (CONST c)::tl -> simplify (TIMES tl) (updateEnv ("const", c, 0) env 1) 1
       | (VAR x)::tl -> simplify (TIMES tl) (updateEnv (x, 1, 1) env 1) 1
       | (POWER (x, p))::tl -> simplify (TIMES tl) (updateEnv (x, 1, p) env 1) 1
       | (SUM lst)::tl ->
         (
           let l = simplify (SUM lst) [] 0 in
           match l with
           | h::t ->
             if (t = []) then List.append l (simplify (TIMES tl) env 1)
              else List.append (SUM l::[]) (simplify (TIMES tl) env 1)
           | [] -> []  (* Feedback : Replace [] by ((Sum lst) :: tl) *)
         )
       | (TIMES lst)::tl -> simplify (TIMES (List.append lst tl)) env 1
       | [] -> deployEnv env 1
     )
   in

   let result = doDiff (aexp, x) in
   match result with
   | SUM _ -> SUM (simplify result [] 0)
   | TIMES _ -> TIMES (simplify result [] 1)
   | _ -> result
```

마무리

- C/C++ 메모리 오류 자동 수정 [FSE'18, ICSE'20]
- Java 널 포인터 오류 자동 수정 [ICSE'22]
- Python 타입 오류 자동 수정 (in progress)
- Solidity 보안 오류 자동 수정 (in progress)
- …
- OCaml 프로그래밍 과제 자동 수정 [OOPSLA'18, FSE'21]

감사합니다