

Automated Patch Transplantation

Ridwan Salihin Shariffdeen, Shin Hwei Tan,
Mingyuan Gao, Abhik Roychoudhury

Proceedings of the 2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)

発表者：辻 裕真

背景

- ・ ソフトウェアの脆弱性が手動で修正されたとき、同様の脆弱性を持つ別のソフトウェアが悪用される可能性がある.
- ・ 手動で行われた修正からパッチを自動で抽出し、同様のテストケースで失敗する別のプログラムに適応するPatchWeaveを提案.

背景

自動プログラム修正

- ・ 自動プログラム修正（APR）はソフトウェアのバグや脆弱性を自動で修正する技術で近年研究が盛んに行われている.
- ・ APRはテストケースには合格するが正しくはない修正を行うことがある.

背景

自動プログラム修正

```
405 static void j2k_read_siz (opj_j2k_t *j2k) {
415     image->y1 = cio_read(cio, 4); /* Ysiz */
416     image->x0 = cio_read(cio, 4); /* X0siz */
417     image->y0 = cio_read(cio, 4); /* Y0siz */
418     cp->tdx = cio_read(cio, 4); /* XTsiz */
419     cp->tdy = cio_read(cio, 4); /* YTtiz */
420     cp->tx0 = cio_read(cio, 4); /* XT0siz */
421     cp->ty0 = cio_read(cio, 4); /* YT0siz */

423     if ((image->x0 < 0) || (image->x1 < 0) || (image->y0 < 0) || (image->y1 < 0)) {
424         opj_event_msg(j2k->cinfo, ...);
427         return;
428     } ...
517     cp->tw = int_ceildiv(image->x1 - cp->tx0, cp->tdx);
518     cp->th = int_ceildiv(image->y1 - cp->ty0, cp->tdy);
519     ...
    /* the overflow error */
560     cp->tcps = (opj_tcp_t*) opj_calloc(cp->tw * cp->th, sizeof(opj_tcp_t)); ...
622 }
```

Fig. 2. Overflow error in OpenJPEG 1.5.1

```
405 static void j2k_read_siz (...) { ....
423 - if ((image->x0 < 0) || (image->x1 < 0) ||
    (image->y0 < 0) || (image->y1 < 0)) {
423 + if ((image->x0 == 0) || (image->x1 < 0) ||
    (image->y0 < 0) || (image->y1 < 0)) {
    ....
622 }
```

(a) Patch generated by F1X

```
405 static void j2k_read_siz (...) { ....
518 - cp->th = int_ceildiv (image->y1 - cp->ty0, cp->tdy);
518 + if (!(1))
519 +     cp->th = int_ceildiv (image->y1 - cp->ty0, cp->tdy);
520     ....
623 }
```

(b) Patch generated by Prophet

Fig. 3. Patches generated using Automated Program Repair

どちらの修正もテストケースが失敗するステートメントが実行されないように変更しただけ

PatchWeave

ツールの概要

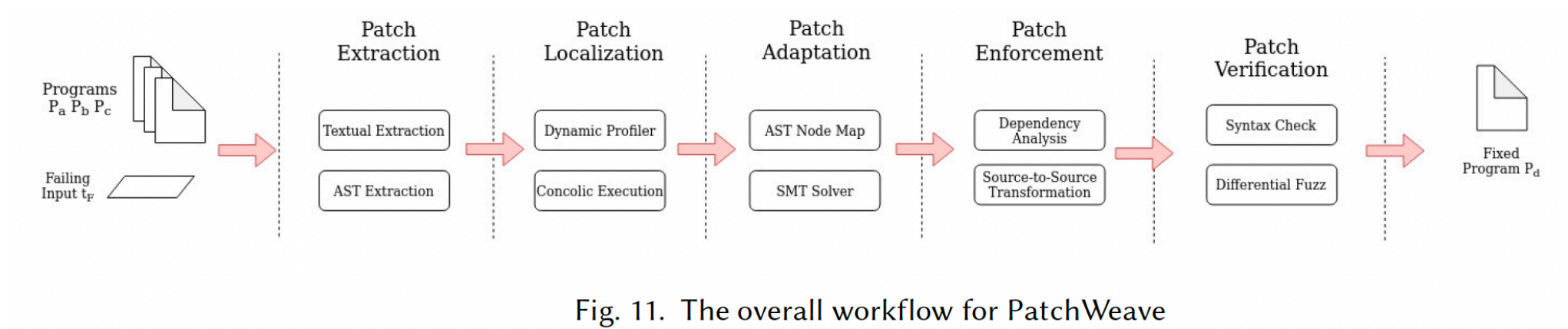


Fig. 11. The overall workflow for PatchWeave

入力

P_a : 脆弱性を持つプログラム

P_b : 脆弱性が修正されたプログラム

P_c : P_a と類似したプログラム

t_F : テストケース

PatchWeave

パッチの抽出

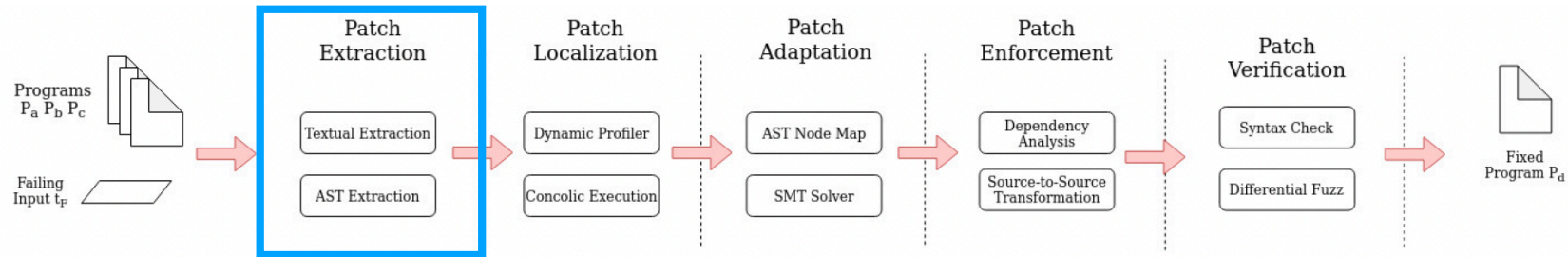


Fig. 11. The overall workflow for PatchWeave

- P_a , P_b のコード間のテキストの差分を取る.
- t_F に対して P_a , P_b を実行してプログラム内の動的実行を監視し、パッチではない変更を除外する.
- 変更箇所周辺のASTを作成する.
- GumTreeを使用してAST間の差分からパッチを抽出する.

PatchWeave

パッチの適用位置特定

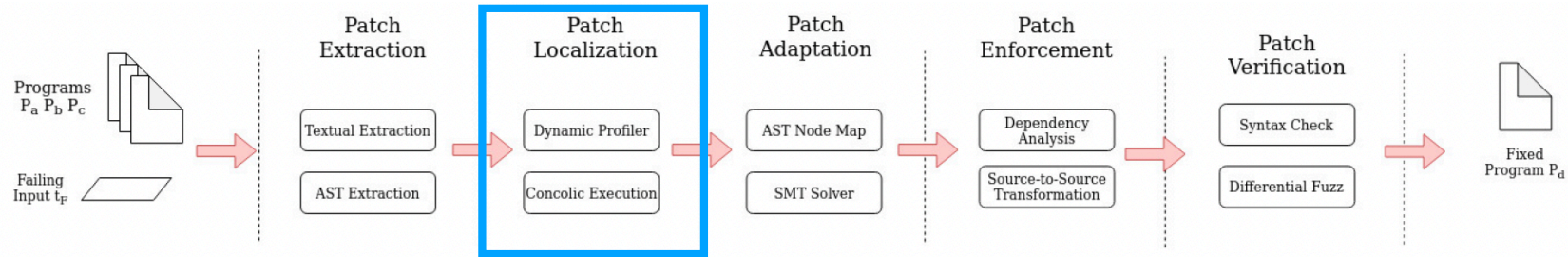


Fig. 11. The overall workflow for PatchWeave

- P_c で t_F を実行しその動的実行を P_a と比較して P_a での変更箇所と類似した箇所を特定する.

PatchWeave

パッチの変換

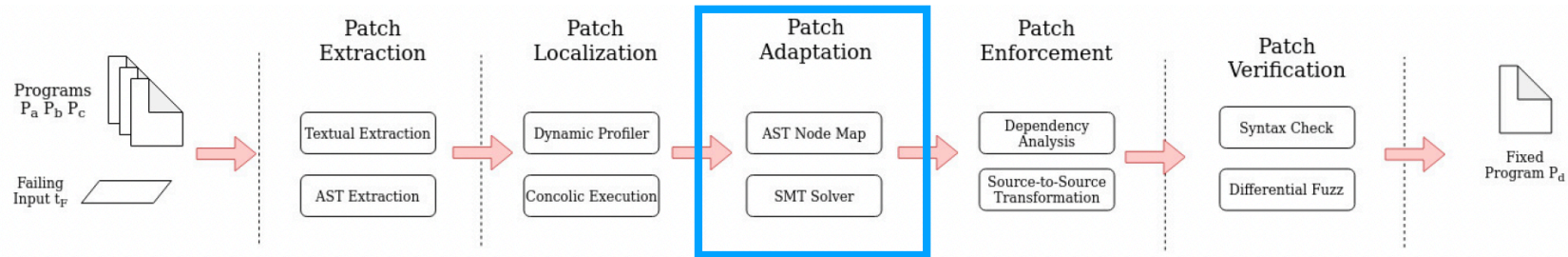


Fig. 11. The overall workflow for PatchWeave

- P_a, P_b, P_c のAST間でマッピングを行う.

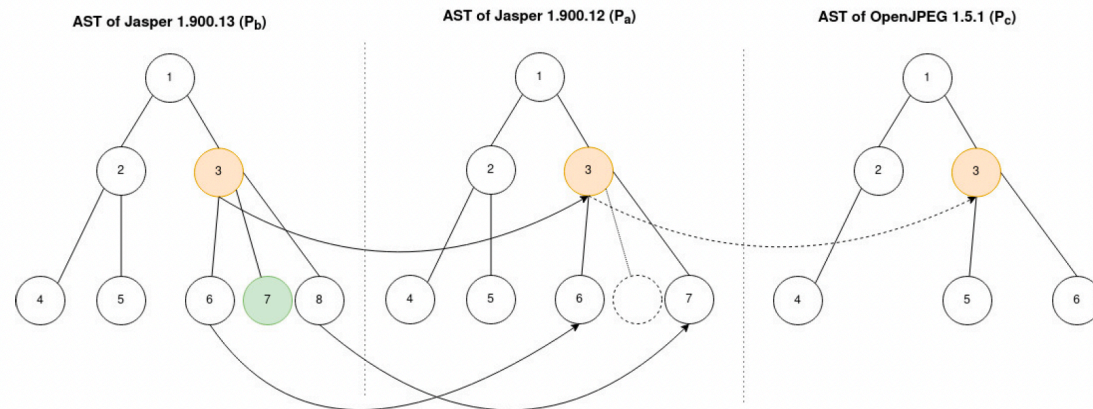


Fig. 13. AST Node Mapping in Patch Adaptation Phase

- P_a, P_b の各変数を P_c に適用できるようにマッピングする.

PatchWeave

パッチの変換

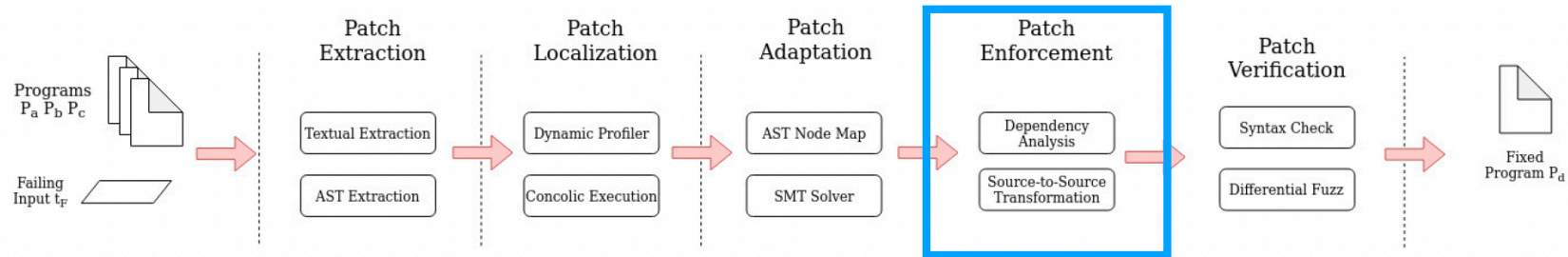


Fig. 11. The overall workflow for PatchWeave

- パッチを適用し依存関係に基づいて必要な関数, ライブラリなどを追加する.

```
char load(ImlibImage * im ..) {
    ....
    im->w = w = cinfo.image_width;
    im->h = h = cinfo.image_height;

+   if (!IMAGE_DIMENSIONS_OK(w, h)) {
+       im->w = im->h = 0;
+       jpeg_destroy_decompress(&cinfo);
+       fclose(f);
+       return 0;
+   }
    ....
}
```

(a) Developer's patch in imlib2 1.4.3

```
+ # define IMAGE_DIMENSIONS_OK(w, h) \
+   ( ((w) > 0) && ((h) > 0) && \
+     ((unsigned long long)(w) * \
+      (unsigned long long)(h) <= \
+      (1ULL << 29) - 1) )

char load(ImlibImage * im ..) {
    ....
    im->w = w = cinfo.image_width;
    im->h = h = cinfo.image_height;
+   if (IMAGE_DIMENSIONS_OK(w, h)) {
+       return 0;
+   }
    ....
}
```

(b) Adapted patch for imlib2 1.4.0

Fig. 7. Example for Class-II: CVE-2006-4806

評価

データセット

- CVEデータベースに公開されている脆弱性を再現するための手順を公開しているリポジトリからプロジェクトを選択.

Name	Description	LOC	Version Range	Count
JasPer	Image manipulations	26k	1.900.2 - 2.0.14	44
OpenJPEG	JPEG 2000 image manipulations	200k	1.4 - 2.1.1	8
LibWebP	WebP image manipulations	67k	0.1.2 - 1.0.2	14
LibTIFF	TIFF images processing	70k	4.0.0 - 4.0.9	10
LibMing	SWF processing	66k	0.4.3 - 0.4.8	6
Libsndfile	Audio manipulation	52k	1.0.25 - 1.0.28	4
Libzip	ZIP archive processor	13k	1.0.0 - 1.5.2	14
WavPack	Lossless Wave file compressor	33k	4.40.0 - 5.1.0	12

Table 3. Experiment subjects and their details

評価

RQ1：実際のプロジェクトでPatchWeaveはどの程度利用できるか。

ID	Bug ID	Exploit ID	Donor	Target	Target Location	Error	Patch Commit	Patch Class	Time (min)	Diff. Fuzz.	Patch Size	Function Hops	Patch Similarity
1	Bug-169	CVE-2016-8691	JasPer-1.900.3	OpenJPEG-1.5.1	int.h@87	DZ	d8c2604	III	5.0	37/0	3	1	C2
2	CVE-2016-8691	CVE-2016-8691	OpenJPEG-1.5.2	JasPer-1.900.2	jpc_dec.c@1194	DZ	e55d5e2	III	8.0	22/0	3	4	C2
3	N/A	CVE-2016-9387	JasPer-1.900.13	OpenJPEG-1.5.1	j2k.c@560	IO	d91198a	IV	7.5	5/0	27	1	C2
4	CVE-2016-9387	CVE-2016-9387	OpenJPEG-1.5.2	JasPer-1.900.12	jpc_dec.c@1234	IO	6e0162a	III	8.5	42/0	3	1	C2
5	Bug-155	CVE-2017-6850	JasPer-2.0.12	OpenJPEG-1.5.1	cio.c@146	NPD	7692d6d	III	✗	-/-	-	-	-
6	CVE-2016-6850	CVE-2017-6850	OpenJPEG-1.5.2	JasPer-1.900.30	jas_malloc.c@111	NPD	7720188	III	15.0	9/0	4	1	C2
7	N/A	CVE-2016-8692	JasPer-1.900.3	OpenJPEG-1.3	int.h@87	DZ	3c55b39	III	3.5	52/0	3	1	C2
8	CVE-2016-8692	CVE-2016-8692	OpenJPEG-1.4	JasPer-1.900.2	jpc_dec.c@1196	DZ	f4d394d	III	5.5	20/0	3	4	C2
9	N/A	CVE-2016-9387	JasPer-1.900.14	OpenJPEG-2.1.0	j2k.c@2099	UIO	ba2b9d00	III	6.0	42/0	4	1	C2
10	N/A	CVE-2016-9387	OpenJPEG-2.1.1	JasPer-1.900.13	jpc_dec.c@1244	UIO	58fc8645	III	18.0	46/0	13	2	C2
11	Bug-312	CVE-2016-9262	OpenJPEG-1.5.1	LibWebP@0.3.0	jpegdec.c@251	SIO	6280b5ad	III	2.5	90/0	4	1	C2
12	N/A	CVE-2016-9830	JasPer-1.900.4	LibWebP@0.2.0	cwebp.c@120	ShO	6109f6a	II	5.0	99/0	2	1	C1
13	N/A	CVE-2016-9830	LibWebP@0.3.0	JasPer-1.900.3	mif_cod.c@394	ShO	7a650c6a	I	11.0	98/0	1	13	C1
14	CVE-2016-9390	CVE-2016-9390	OpenJPEG-1.5.2	JasPer-1.900.13	jpc_mct.c@151	HBO	69cd4f9	III	✗	-/-	-	-	-
15	Bug-297	CVE-2016-9390	JasPer-1.900.14	OpenJPEG-2.1.0	libopenjpeg/mct.c@84	HBO	dee11ec	IV	✗	-/-	-	-	-
16	CVE-2016-9393	CVE-2016-9393	JasPer-1.900.17	OpenJPEG-1.5.2	j2k.c@447	UIO	f7038068	III	12.0	42/0	3	1	N.A.
17	CVE-2016-8884	CVE-2016-8884	LibTiff-3.8.0	Jasper-1.900.8	bmp_dec.c@394	MWE	50373d7d	III	26.0	18/0	8	23	C2
18	N/A	CVE-2017-6828	Libsndfile-1.0.26	WavPack-5.1.0	common.c@992	ShO	3e91aaf7	III	✗	-/-	-	-	-
19	CVE-2016-9387	CVE-2016-9387	Jasper-1.900.13	Jasper-1.900.2	jpc_dec.c@1206	IO	d91198a	II	8.0	34/0	29	1	C1
20	CVE-2016-9265	CVE-2016-9265	LibMing-0.4.8	LibMing-0.4.6	listmp3.c@187	DZ	b0704f80	I	5.5	98/0	2	1	C1
21	CVE-2016-9266	CVE-2016-9266	LibMing-0.4.8	LibMing-0.4.6	listmp3.c@94	NS	2e5a98a0	I	5.5	29/0	17	1	C1
22	Bugzilla-2634	CVE-2017-14039	LibTiff-4.0.8	LibTiff-4.0.0	tiff2ps.c@2443	HBO	5ed9fea5	I	26.0	75/0	5	18	C1
23	CVE-2017-8365	CVE-2017-8365	Libsndfile-master	Libsndfile-1.0.26	src/pcm.c@670	GBO	fd0484ab	I	16.0	73/0	9	3	C1
24	CVE-2017-14107	CVE-2017-14107	Libzip-1.3.0	Libzip-1.1.2	zip_dirent.c@108	MAF	9b46957e	I	42.0	90/0	4	13	C1

Division By Zero Error {DZ - Divide by Zero}, Integer Overflow {IO - Integer Overflow, UIO - Unsigned Integer Overflow, SIO - Signed Integer Overflow},
Memory Error {NPD - Null Pointer Dereference, MAF - Memory Allocation Failure, MWE - Memory Write Error}, Shift Overflow {ShO - Shift Overflow, NS - Negative Shift},
Buffer Overflow {HBO - Heap Buffer Overflow, GBO - Global Buffer Overflow}

Table 4. Summary of PatchWeave experiment results

4つ以外の全ての脆弱性に対するパッチを正しく移植できた。

評価

RQ2：移植されたコードの品質はどうか.

開発者が作成したパッチと比較

ID	Bug ID	Exploit ID	Donor	Target	Target Location	Error	Patch Commit	Patch Class	Time (min)	Diff. Fuzz.	Patch Size	Function Hops	Patch Similarity
1	Bug-169	CVE-2016-8691	JasPer-1.900.3	OpenJPEG-1.5.1	int.h@87	DZ	d8c2604	III	5.0	37/0	3	1	C2
2	CVE-2016-8691	CVE-2016-8691	OpenJPEG-1.5.2	JasPer-1.900.2	jpc_dec.c@1194	DZ	e55d5e2	III	8.0	22/0	3	4	C2
3	N/A	CVE-2016-9387	JasPer-1.900.13	OpenJPEG-1.5.1	j2k.c@560	IO	d91198a	IV	7.5	5/0	27	1	C2
4	CVE-2016-9387	CVE-2016-9387	OpenJPEG-1.5.2	JasPer-1.900.12	jpc_dec.c@1234	IO	6e0162a	III	8.5	42/0	3	1	C2
5	Bug-155	CVE-2017-6850	JasPer-2.0.12	OpenJPEG-1.5.1	cio.c@146	NPD	7692d6d	III	✗	-/-	-	-	-
6	CVE-2016-6850	CVE-2017-6850	OpenJPEG-1.5.2	JasPer-1.900.30	jas_malloc.c@111	NPD	7720188	III	15.0	9/0	4	1	C2
7	N/A	CVE-2016-8692	JasPer-1.900.3	OpenJPEG-1.3	int.h@87	DZ	3c55b39	III	3.5	52/0	3	1	C2
8	CVE-2016-8692	CVE-2016-8692	OpenJPEG-1.4	JasPer-1.900.2	jpc_dec.c@1196	DZ	f4d394d	III	5.5	20/0	3	4	C2
9	N/A	CVE-2016-9387	JasPer-1.900.14	OpenJPEG-2.1.0	j2k.c@2099	UIO	ba2b9d00	III	6.0	42/0	4	1	C2
10	N/A	CVE-2016-9387	OpenJPEG-2.1.1	JasPer-1.900.13	jpc_dec.c@1244	UIO	58fc8645	III	18.0	46/0	13	2	C2
11	Bug-312	CVE-2016-9262	OpenJPEG-1.5.1	LibWebP@0.3.0	jpegdec.c@251	SIO	6280b5ad	III	2.5	90/0	4	1	C2
12	N/A	CVE-2016-9830	JasPer-1.900.4	LibWebP@0.2.0	cwebp.c@120	ShO	6109f6a	II	5.0	99/0	2	1	C1
13	N/A	CVE-2016-9830	LibWebP@0.3.0	JasPer-1.900.3	mif_cod.c@394	ShO	7a650c6a	I	11.0	98/0	1	13	C1
14	CVE-2016-9390	CVE-2016-9390	OpenJPEG-1.5.2	JasPer-1.900.13	jpc_mct.c@151	HBO	69cd4f9	III	✗	-/-	-	-	-
15	Bug-297	CVE-2016-9390	JasPer-1.900.14	OpenJPEG-2.1.0	libopenjpeg/mct.c@84	HBO	dee11ec	IV	✗	-/-	-	-	-
16	CVE-2016-9393	CVE-2016-9393	JasPer-1.900.17	OpenJPEG-1.5.2	j2k.c@447	UIO	f7038068	III	12.0	42/0	3	1	N.A.
17	CVE-2016-8884	CVE-2016-8884	LibTiff-3.8.0	Jasper-1.900.8	bmp_dec.c@394	MWE	50373d7d	III	26.0	18/0	8	23	C2
18	N/A	CVE-2017-6828	Libsndfile-1.0.26	WavPack-5.1.0	common.c@992	ShO	3e91aaf7	III	✗	-/-	-	-	-
19	CVE-2016-9387	CVE-2016-9387	Jasper-1.900.13	Jasper-1.900.2	jpc_dec.c@1206	IO	d91198a	II	8.0	34/0	29	1	C1
20	CVE-2016-9265	CVE-2016-9265	LibMing-0.4.8	LibMing-0.4.6	listmp3.c@187	DZ	b0704f80	I	5.5	98/0	2	1	C1
21	CVE-2016-9266	CVE-2016-9266	LibMing-0.4.8	LibMing-0.4.6	listmp3.c@94	NS	2e5a98a0	I	5.5	29/0	17	1	C1
22	Bugzilla-2634	CVE-2017-14039	LibTiff-4.0.8	LibTiff-4.0.0	tiff2ps.c@2443	HBO	5ed9fea5	I	26.0	75/0	5	18	C1
23	CVE-2017-8365	CVE-2017-8365	Libsndfile-master	Libsndfile-1.0.26	src/pcm.c@670	GBO	fd0484ab	I	16.0	73/0	9	3	C1
24	CVE-2017-14107	CVE-2017-14107	Libzip-1.3.0	Libzip-1.1.2	zip_dirent.c@108	MAF	9b46957e	I	42.0	90/0	4	13	C1

Division By Zero Error {DZ - Divide by Zero}, Integer Overflow {IO - Integer Overflow, UIO - Unsigned Integer Overflow, SIO - Signed Integer Overflow},
Memory Error {NPD - Null Pointer Dereference, MAF - Memory Allocation Failure, MWE - Memory Write Error}, Shift Overflow {ShO - Shift Overflow, NS - Negative Shift},
Buffer Overflow {HBO - Heap Buffer Overflow, GBO - Global Buffer Overflow}

Table 4. Summary of PatchWeave experiment results

C1：構文的に同等 C2：意味的に同等

ほぼ全ての脆弱性に対して構文的に同等か意味的に同等なパッチを適用できた.

まとめ

- ・脆弱性に対する手動の修正を抽出し，同様の脆弱性を持つ他のプログラムにパッチを適用するPatchWeaveを提案した.
- ・ほとんどの脆弱性に対するパッチを移植できた.
- ・移植したパッチは開発者が手動で作成したものと比較し，構文的に同等か意味的に同等であったことが分かった.

所感

- ・ テストケースの実行時のプログラムの動作からパッチに関する変更だけを抽出できるのがすごいと思った.
- ・ 同一の脆弱性を持つ類似したプログラムに対しての修正なのでツールの使い所が限られているのでは.