StorFuzz: Using Data Diversity to Overcome Fuzzing Plateaus Supplementary Tables

Leon Weiß
Ruhr University Bochum
Bochum, Germany
leon.weiss@rub.de

Tobias Holl
Ruhr University Bochum
Bochum, Germany
tobias.holl@rub.de

Kevin Borgolte
Ruhr University Bochum
Bochum, Germany
kevin.borgolte@rub.de

This document contains supplementary tables reporting on all experiments for StorFuzz:

List of Tables

This is supplementary material to:

L. Weiß, T. Holl, and K. Borgolte. "StorFuzz: Using Data Diversity to Overcome Fuzzing Plateaus." In: *Proceedings of the 48th IEEE/ACM International Conference on Software Engineering (ICSE)*. Association for Computing Machinery (ACM)/Institute of Electrical and Electronics Engineers (IEEE), Apr. 2026. DOI: 10.1145/3744916.3773179

ICSE '26, Rio de Janeiro, Brazil

Table 1: Corpus size of the initial OSS-Fuzz corpus compared to the saturated corpus, and the corpora created by diversifying the saturated corpus with StorFuzz, DDFuzz, or restarting LibAFL for 24 hours. We also report the difference to the seed set size. The largest diversified corpus per benchmark is highlighted. All pairwise differences are significant.

Benchmark	OSS-Fuzz Corpus	Saturated Corpus		Diversified Corpora (Median of 10 Trials)					
				StorFuzz		LibAFL		DDFuzz	
bloaty	12 841	15 832	(1.23×)	40 776	(2.58×)	2 479	(0.16×)	3 953.5	(0.25×)
curl	15 803	24 682	(1.56×)	28 818.5	(1.17×)	2 469	(0.10×)	4 890	(0.20×)
freetype2	21 198	48 954	(2.31×)	108 563.5	(2.22×)	6 418.5	(0.13×)	58 406	(1.19×)
harfbuzz	29 745	48 105	(1.62×)	88 038	(1.83×)	7 723.5	(0.16×)	12 984.5	(0.27×)
jsoncpp	3 530	3 635	(1.03×)	2 720	(0.75×)	336	(0.09×)	567.5	(0.16×)
lcms	5 188	5 388	(1.04×)	7 482.5	(1.39×)	562	(0.10×)	598	(0.11×)
libjpeg-turbo	6 398	6 755	(1.06×)	14 152	(2.10×)	1 216	(0.18×)	1 832.5	(0.27×)
libpcap	13 554	15 014	(1.11×)	31 085	(2.07×)	2 246.5	(0.15×)	13 908.5	(0.93×)
libpng	8 296	8 536	(1.03×)	17 829.5	(2.09×)	771	(0.09×)	1 101.5	(0.13×)
libxml2	17 456	42 708	(2.45×)	24 776.5	(0.58×)	3 959	(0.09×)	7 929.5	(0.19×)
libxslt	32 636	70 440	(2.16×)	17 214	(0.24×)	3 533	(0.05×)	8 484.5	(0.12×)
mbedtls	4 482	4 901	(1.09×)	42 346	(8.64×)	770	(0.16×)	1 080	(0.22×)
openh264	20 109	25 240	(1.26×)	54 209.5	(2.15×)	2 372	(0.09×)	7 914	(0.31×)
openssl	11 125	15 357	(1.38×)	18 318	(1.19×)	1 057.5	(0.07×)	1 198	(0.08×)
openthread	1 368	4 949	(3.62×)	15 111.5	(3.05×)	698.5	(0.14×)	1 219	(0.25×)
proj4	16 360	26 905	(1.64×)	24 514	(0.91×)	5 088	(0.19×)	10 412.5	(0.39×)
re2	10 738	20 712	(1.93×)	9 588	(0.46×)	934	(0.05×)	10 366	(0.50×)
sqlite3	16 715	40 653	(2.43×)	28 750.5	(0.71×)	6 195	(0.15×)	19 723.5	(0.49×)
stb	6 507	7 642	(1.17×)	25 330	(3.31×)	936.5	(0.12×)	7 257	(0.95×)
systemd	8 534	10 000	(1.17×)	4 510	(0.45×)	1 050	(0.10×)	1 440	(0.14×)
vorbis	5 655	5 833	(1.03×)	12 203.5	(2.09×)	403	(0.07×)	557	(0.10×)
woff2	7 496	7 918	(1.06×)	31 144	(3.93×)	654.5	(0.08×)	7 947.5	(1 ×)
zlib	1 460	1 566	(1.07×)	4 118	(2.63×)	168	(0.11×)	1 193	(0.76×)

Table 2: Corpus sizes of the corpora created by diversifying the saturated corpus with StorFuzz, DDFuzz, or restarting LibAFL for at most 24 hours. We report the median size reached by each fuzzer in the number of executions achieved in the slowest trial across all fuzzers, thereby accounting for different execution speeds. The largest diversified corpus per benchmark is highlighted.

Benchmark	Diversified Corpora (Median of 10 Trials)		
	StorFuzz	LibAFL	DDFuzz
bloaty	24 657	2 473.5	3 949
curl	24 094.5	2 465	4 882.5
freetype2	72 462.5	6 399.5	58 324
harfbuzz	67 377	7 700.5	12 958
jsoncpp	2 480.5	336	567.5
lcms	6 907.5	562	598
libjpeg-turbo	11 943.5	1 075.5	1 832.5
libpcap	25 542.5	2 239	13 827.5
libpng	14 459.5	771	1 101.5
libxml2	21 432	3 946.5	7 915.5
libxslt	11 938	3 511	8 477.5
mbedtls	23 805	770	1 078
openh264	51 829.5	2 371	7 886
openssl	14 707.5	1 056.5	1 198
openthread	13 059	683	1 219
proj4	17 379	5 087	9 860.5
re2	8 153.5	933	10 359.5
sqlite3	16 812	6 076	19 472.5
stb	20 756.5	936	7 224
systemd	4 498.5	1 049	1 440
vorbis	11 022.5	403	557
woff2	22 299.5	654	7 947.5
zlib	3 729.5	168	1 192.5

Table 3: *Median* number of branches covered at different points during the fuzzing campaigns starting from the saturated corpus (OSS-Fuzz and five 120-hours LibAFL runs). Two fuzzers in the configuration means that it switched between the two fuzzers at the indicated times. We report the median coverage over 10 trials and the change relative to the previously reported point. Green (▢) marks the best performing fuzzing configuration at the specific point in the run, the best configuration per target appears in **bold**. We omit values for improved readability, when there is no change.

Benchmark	Fuzzer Configuration	Seed Coverage (after 5 × 120h)	Code Coverage after 144 h	168 h	192 h	216 h
bloaty	LibAFL	7 537	7 539 (+2)			7 539.5 (+0.5)
	LibAFL/LibAFL		7 538.5 (+1.5)	7 539 (+0.5)	7 539.5 (+0.5)	
	StorFuzz/LibAFL		7 556.5 (+19.5)		7 638 (+81.5)	7 639 (+1)
	DDFuzz/LibAFL		7 538 (+1)			7 539 (+1)
curl	LibAFL	11 554	11 556.5 (+2.5)	11 557 (+0.5)		
	LibAFL/LibAFL		11 557 (+3)	11 558 (+1)	11 558.5 (+0.5)	
	StorFuzz/LibAFL		11 569.5 (+15.5)	11 571 (+1.5)	11 574.5 (+3.5)	11 575 (+0.5)
	DDFuzz/LibAFL		11 557 (+3)	11 558 (+1)		11 558.5 (+0.5)
freetype2	LibAFL	18 043	18 045 (+2)		18 045.5 (+0.5)	
	LibAFL/LibAFL		18 044.5 (+1.5)	18 045 (+0.5)	18 046 (+1)	18 046.5 (+0.5)
	StorFuzz/LibAFL		18 054.5 (+11.5)	18 055 (+0.5)	18 060 (+5)	18 060.5 (+0.5)
	DDFuzz/LibAFL		18 045 (+2)	18 045.5 (+0.5)	18 046.5 (+1)	18 047 (+0.5)
harfbuzz	LibAFL	21 846	21 847 (+1)		21 850.5 (+3.5)	21 859 (+8.5)
	LibAFL/LibAFL		21 847.5 (+1.5)	21 852 (+4.5)	21 867 (+15)	21 877 (+10)
	StorFuzz/LibAFL		21 870.5 (+24.5)	21 873.5 (+3)	21 878 (+4.5)	21 889 (+11)
	DDFuzz/LibAFL		21 854.5 (+8.5)	21 871 (+16.5)	21 873.5 (+2.5)	21 888 (+14.5)
jsoncpp	**LibAFL**	525				
	LibAFL/LibAFL					
	StorFuzz/LibAFL					
	DDFuzz/LibAFL					
lcms	**LibAFL**	2 476				
	LibAFL/LibAFL					
	StorFuzz/LibAFL					
	DDFuzz/LibAFL					
libjpeg-turbo	**LibAFL**	3 417				
	LibAFL/LibAFL					
	StorFuzz/LibAFL					
	DDFuzz/LibAFL					
libpcap	**LibAFL**	4 465				
	LibAFL/LibAFL					
	StorFuzz/LibAFL					
	DDFuzz/LibAFL					
libpng	**LibAFL**	2 116				
	LibAFL/LibAFL					
	StorFuzz/LibAFL					
	DDFuzz/LibAFL					

Table 3: *Continued from previous page*

Benchmark	Fuzzer Configuration	Seed Coverage (after 5 × 120h)	Code Coverage after 144 h	168 h	192 h	216 h
libxml2	LibAFL	16 311		16 312 (+1)		16 313 (+1)
	LibAFL/LibAFL		16 311.5 (+0.5)	16 314 (+2.5)	16 315 (+1)	16 315.5 (+0.5)
	StorFuzz/LibAFL		16 314 (+3)	16 314.5 (+0.5)	16 319 (+4.5)	16 321 (+2)
	DDFuzz/LibAFL		16 313 (+2)	16 314.5 (+1.5)	16 315 (+0.5)	
libxslt	LibAFL	12 135	12 140 (+5)	12 141 (+1)	12 142.5 (+1.5)	12 144.5 (+2)
	LibAFL/LibAFL		12 143.5 (+8.5)	12 144.5 (+1)		12 145 (+0.5)
	StorFuzz/LibAFL		12 158.5 (+23.5)	12 160 (+1.5)	12 162 (+2)	12 163 (+1)
	DDFuzz/LibAFL		12 139 (+4)	12 146 (+7)	12 148 (+2)	12 148.5 (+0.5)
mbedtls	LibAFL	4 418				
	LibAFL/LibAFL					
	StorFuzz/LibAFL		4 419 (+1)			
	DDFuzz/LibAFL					
openh264	LibAFL	9 839				
	LibAFL/LibAFL					
	StorFuzz/LibAFL		9 841 (+2)			
	DDFuzz/LibAFL					
openssl	LibAFL	5 974	5 977 (+3)			
	LibAFL/LibAFL		5 977 (+3)			
	StorFuzz/LibAFL		5 977 (+3)	5 977.5 (+0.5)	5 979 (+1.5)	
	DDFuzz/LibAFL		5 977 (+3)			5 978 (+1)
openthread	LibAFL	4 581	4 635.5 (+54.5)	4 708 (+72.5)	4 710.5 (+2.5)	
	LibAFL/LibAFL		4 637.5 (+56.5)	4 712 (+74.5)	4 720 (+8)	4 721.5 (+1.5)
	StorFuzz/LibAFL		4 651 (+70)	4 658.5 (+7.5)	4 671.5 (+13)	4 684 (+12.5)
	DDFuzz/LibAFL		4 583.5 (+2.5)	4 684 (+100.5)	4 690.5 (+6.5)	4 720 (+29.5)
proj4	LibAFL	11 110	11 112 (+2)	11 112.5 (+0.5)	11 115 (+2.5)	11 116.5 (+1.5)
	LibAFL/LibAFL		11 112 (+2)	11 113 (+1)		11 114 (+1)
	StorFuzz/LibAFL		11 119 (+9)	11 120 (+1)	11 125.5 (+5.5)	11 127.5 (+2)
	DDFuzz/LibAFL		11 111 (+1)	11 112 (+1)	11 113 (+1)	11 115 (+2)
re2	LibAFL	2 937		2 938 (+1)		
	LibAFL/LibAFL		2 938 (+1)			
	StorFuzz/LibAFL		2 938 (+1)			
	DDFuzz/LibAFL		2 938 (+1)			2 939 (+1)
sqlite3	LibAFL	21 545	21 583 (+38)	21 587 (+4)	21 592 (+5)	21 598.5 (+6.5)
	LibAFL/LibAFL		21 578.5 (+33.5)	21 601.5 (+23)	21 614 (+12.5)	21 622.5 (+8.5)
	StorFuzz/LibAFL		21 592 (+47)	21 609 (+17)	21 623.5 (+14.5)	21 630.5 (+7)
	DDFuzz/LibAFL		21 582 (+37)	21 601.5 (+19.5)	21 613.5 (+12)	21 626 (+12.5)

Table 3: *Continued from previous page*

Benchmark	Fuzzer Configuration	Seed Coverage (after 5 × 120h)	Code Coverage after 144 h	168 h	192 h	216 h
stb	**LibAFL**	2 322				
	LibAFL/LibAFL					
	StorFuzz/LibAFL					
	DDFuzz/LibAFL					
systemd	**LibAFL**	271				
	LibAFL/LibAFL					
	StorFuzz/LibAFL					
	DDFuzz/LibAFL					
vorbis	**LibAFL**	1 381				
	LibAFL/LibAFL					
	StorFuzz/LibAFL					
	DDFuzz/LibAFL					
woff2	**LibAFL**	1 258				
	LibAFL/LibAFL					
	StorFuzz/LibAFL					
	DDFuzz/LibAFL					
zlib	**LibAFL**	473				
	LibAFL/LibAFL					
	StorFuzz/LibAFL					
	DDFuzz/LibAFL					

Table 4: *Median* number of branches covered at different points during the fuzzing campaigns starting from the corpus saturated with both LibAFL and libFuzzer (five runs of 120 hours each). Two fuzzers in the configuration means that it switched between the two fuzzers at the indicated times. We report the median coverage over 10 trials and the change relative to the previously reported point. Green (□) marks the best performing fuzzing configuration at the specific point in the run, the best configuration per target appears in **bold**. We omit values for improved readability, when there is no change.

Benchmark	Fuzzer Configuration	Seed Coverage (after 2 × 5 × 120h)	Code Coverage after 144 h	168 h	192 h	216 h
bloaty	StorFuzz/LibAFL	7 550	7 565.5 (+15.5)	7 566.5 (+1)	7 596.5 (+30)	
	WingFuzz/libFuzzer		7 628.5 (+78.5)	7 629 (+0.5)	7 651 (+22)	7 651.5 (+0.5)
curl	StorFuzz/LibAFL	11 623	11 633.5 (+10.5)	11 634.5 (+1)		11 635 (+0.5)
	WingFuzz/libFuzzer		11 636 (+13)	11 637.5 (+1.5)		11 639 (+1.5)
freetype2	StorFuzz/LibAFL	18 107	18 126 (+19)	18 129 (+3)	18 129.5 (+0.5)	
	WingFuzz/libFuzzer		18 134 (+27)	18 135 (+1)	18 136 (+1)	
harfbuzz	**StorFuzz/LibAFL**	21 879	21 908 (+29)	21 914.5 (+6.5)	21 922 (+7.5)	21 927.5 (+5.5)
	WingFuzz/libFuzzer		21 881 (+2)	21 904 (+23)	21 910.5 (+6.5)	21 913.5 (+3)
jsoncpp	**StorFuzz/LibAFL**	525				
	WingFuzz/libFuzzer					
lcms	**StorFuzz/LibAFL**	2 476		2 480 (+4)		
	WingFuzz/libFuzzer			2 476.5 (+0.5)	2 477 (+0.5)	
libjpeg-turbo	**StorFuzz/LibAFL**	3 417				
	WingFuzz/libFuzzer					
libpcap	StorFuzz/LibAFL	4 469				
	WingFuzz/libFuzzer			4 469.5 (+0.5)	4 470 (+0.5)	
libpng	**StorFuzz/LibAFL**	2 116				
	WingFuzz/libFuzzer					
libxml2	StorFuzz/LibAFL	16 394		16 395 (+1)	16 396 (+1)	16 397.5 (+1.5)
	WingFuzz/libFuzzer		16 396.5 (+2.5)		16 400 (+3.5)	16 401 (+1)
libxslt	**StorFuzz/LibAFL**	12 217	12 231 (+14)	12 236.5 (+5.5)	12 237 (+0.5)	12 239 (+2)
	WingFuzz/libFuzzer		12 222 (+5)	12 226.5 (+4.5)	12 231 (+4.5)	12 234.5 (+3.5)
mbedtls	**StorFuzz/LibAFL**	4 420				
	WingFuzz/libFuzzer					
openh264	**StorFuzz/LibAFL**	9 841				
	WingFuzz/libFuzzer					
openssl	StorFuzz/LibAFL	5 982	5 985 (+3)			
	WingFuzz/libFuzzer		5 985.5 (+3.5)	5 986 (+0.5)	5 986.5 (+0.5)	5 987.5 (+1)
openthread	**StorFuzz/LibAFL**	4 778	4 791 (+13)	4 795 (+4)	4 799 (+4)	4 800.5 (+1.5)
	WingFuzz/libFuzzer		4 780 (+2)	4 781 (+1)		4 781.5 (+0.5)
proj4	StorFuzz/LibAFL	11 351	11 356 (+5)	11 362 (+6)	11 369.5 (+7.5)	11 370 (+0.5)
	WingFuzz/libFuzzer		11 371.5 (+20.5)	11 377 (+5.5)	11 391.5 (+14.5)	11 397 (+5.5)
re2	StorFuzz/LibAFL	2 963				
	WingFuzz/libFuzzer		2 965 (+2)			2 966 (+1)
sqlite3	**StorFuzz/LibAFL**	21 585	21 682.5 (+97.5)	21 694.5 (+12)	21 707 (+12.5)	21 715 (+8)
	WingFuzz/libFuzzer		21 665.5 (+80.5)	21 678.5 (+13)	21 695 (+16.5)	21 703 (+8)

Table 4: *Continued from previous page*

Benchmark	Fuzzer Configuration	Seed Coverage (after 2 × 5 × 120h)	Code Coverage after 144 h	168 h	192 h	216 h
stb	**StorFuzz/LibAFL**	2 322				
	WingFuzz/libFuzzer					
systemd	**StorFuzz/LibAFL**	271				
	WingFuzz/libFuzzer					
vorbis	StorFuzz/LibAFL	1 381				
	WingFuzz/libFuzzer		1 384 (+3)			
woff2	**StorFuzz/LibAFL**	1 263				
	WingFuzz/libFuzzer					
zlib	**StorFuzz/LibAFL**	473				
	WingFuzz/libFuzzer					

Table 5: Bugs discovered by StorFuzz. In some cases the bugs were publicly reported by others while we were in the process of responsible disclosure. We have anonymized the references where necessary.

Identifier	Description	Public Reference	CVE
ImageMagick-1	(Rediscovered) UAF in `ReadTIFFImage`	Commit 30f7a3d	(n.a.)
ImageMagick-2 / LibRaw-4	Heap OOB Read due to a Bug in LibRaw `open_datastream`	PR #679	(requested)
ImageMagick-3	Heap OOB Reads	Commit 81ac8a0 & Commit bac413a	CVE-2025-43965
ImageMagick-4 / libheif-3	SEGV due to NULL-Pointer Dereference in libheif `ImageItem_Grid::get_decoder`	Issue #1473	
ImageMagick-5 / libheif-2	SEGV due to NULL-Pointer Dereference in libheif `ImageItem::get_coded_image_colorspace`	Issue #1455	CVE-2025-43967
ImageMagick-6 / LibRaw-1	Heap OOB Read in LibRaw `parse_tiff_ifd`	Commit 66fe663	CVE-2025-43961
ImageMagick-7 / LibRaw-2	Heap OOB Read in LibRaw `phase_one_correct`	Commit 66fe663	CVE-2025-43962 & CVE-2025-43964
ImageMagick-8 / LibRaw-3	SEGV due to Heap OOB Read in LibRaw `parse_one_correct`	Commit be26e76	CVE-2025-43963
UPX-1	SEGV due to OOB Read in `elf_lookup("JNI_OnLoad")`	Issue #871	†
UPX-2	SEGV due to OOB Read in `invert_pt_dynamic`	Issue #872 (d4bc364)	†
UPX-3	SEGV due to OOB Read in `invert_pt_dynamic` due to integer overflow	Issue #872 (4aa92d8)	†
UPX-4	OOB Read in `get_dynsym_name`	Issue #871	†
UPX-5	OOB Read in `PackMachBase::canUnpack() 1682`	Issue #874	†
UPX-6	OOB Read in `PackMachBase::canUnpack() 1862 / 1866`	Issue #875	†
libheif-1	SEGV due to NULL-Pointer Dereference in `ImageItem_iden::get_luma_bits_per_pixel`	Issue #1455	CVE-2025-43967
libheif-1b	SEGV due to NULL-Pointer Dereference in `ImageItem_iden::get_coded_image_colorspace`	Commit b385553	CVE-2025-43966
VLC-1	Heap OOB Write in mp4.c `FragCreateTrunIndex` (MP4 demuxer)	Issue #28959	(n.a.)
VLC-2	Heap OOB Write in spudec/parse.c `ParseRLE` (SPU decoder)	Issue #28960	(n.a.)
VLC-3	Heap OOB Write in svcdsub.c `SVCDSubRenderImage` (SVCD decoder)	Issue #28961	(n.a.)
VLC-4	Limited Heap OOB Write in substx3g.c `Decode` (tx3g decoder)	Issue #28965	(n.a.)
VLC-5	Double Free in libmp4.c `MP4_ReadBox_sgpd` and `MP4_FreeBox_sgpd` (MP4 demuxer)	Issue #28967	(n.a.)
VLC-6	Possible Stack-based Buffer Overflow in `aout_ChannelReorder` (audio output)	Issue #28968	(n.a.)
VLC-7	Heap OOB Read in wav.c `ChunkParseFmt` (WAV demuxer)	Issue #28969	(n.a.)
VLC-8	Limited Heap OOB Read in meta.c `iTUNTripletCallback` (MP4 demuxer)	Issue #28970	(n.a.)
VLC-9	Heap OOB Read in `aout_ChannelReorder` (audio output)	Issue #28971	(n.a.)
VLC-10	(Rediscovered) Limited Heap OOB Read in ty.c `find_es_header` (TY demuxer)		(pending)[$$]
VLC-11	Arbitrary Read via `es_format_Copy` due to Use of Uninitialized Memory (via MP4 demuxer)	Issue #28972	(n.a.)
VLC-12	Division by Zero in libmp4.c `MP4_ReadBox_iloc` (MP4 demuxer)	Issue #28973	(n.a.)
VLC-13	Division by Zero in avi.c `AVI_Rescale` (AVI demuxer)	Issue #28974	(n.a.)

Table 5: *Continued from previous page*

Identifier	Description	Public Reference	CVE
VLC-14	Assertion Failure in `AVI_IndexLoad` (AVI demuxer)	Issue #28975	(n.a.)
VLC-15	Assertion Failure in `vlc_meta_SetWithPriority` (multiple code paths)	Issue #28976	(n.a.)
VLC-16	Assertion Failure in `date_Increment` (multiple code paths)	Issue #28977	(n.a.)
VLC-17	Assertion Failure in `date_Increment` (multiple code paths)	Issue #28978	(n.a.)
VLC-18	Assertion Failure in `aout_ChannelReorder` (e.g. via WAV demuxing)	Issue #28979	(n.a.)
VLC-19	Assertion Failure in `picture_Setup`	Issue #28980	(n.a.)
VLC-20	Assertion Failure in SPU subtitle rendering (`Render`)	Issue #28981	(n.a.)
VLC-21	Assertion Failure in `webvtt_region_Reduce` (WebVTT processing)	Issue #28982	(n.a.)
VLC-22	Assertion Failure in `TrackUpdateStarttimes` (MP4 demuxing)	Issue #28983	(n.a.)
VLC-23	Assertion Failure in `MP4_GetAudioFrameInfo` (MP4 demuxing)	Issue #28984	(n.a.)
VLC-24	Assertion Failure in `SetupAudioES` (MP4 demuxing / ES setup)	Issue #28985	(n.a.)
VLC-25	NULL-Pointer Dereference in `text_segment_ruby_New` (via WebVTT)	Issue #28986	(n.a.)
VLC-26	NULL-Pointer Dereference in `webvtt_FillStyleFromCssDeclaration` (via `CSSGrammar.y`)	Issue #28987	(n.a.)
VLC-27	NULL-Pointer Dereference in CSS parsing (`CSSGrammar.y`)	Issue #28988	(n.a.)
assimp-1	SEGV: Heap OOB Write in `ParseLV4MeshBonesVertices`	Issue #6024[$]	CVE-2025-3159
assimp-2	Heap OOB Write in `CSMImporter::InternReadFile`	PR #6138[$]	CVE-2025-2592
assimp-3	Stack OOB Write in `GetNextLine<char>`	Issue #6016[$]	CVE-2025-2151
assimp-4	SIGSEGV: Heap OOB Read in `MDCImporter::ValidateSurfaceHeader`	Issue #6167	CVE-2025-5165
assimp-5	SIGSEGV: Heap OOB Read in `MDCImporter::InternReadFile`	Issue #6168	CVE-2025-5166
assimp-6	SIGSEGV: Constant-Pointer Dereference in `NDOImporter::InternReadFile`	PR #6055[$]	
assimp-7	SIGSEGV: Constant-Pointer Dereference in `CSMImporter::InternReadFile`	Issue #6012[$]	CVE-2025-2751
assimp-8	Heap OOB Read in `LWOImporter::GetS0`	Issue #6169	CVE-2025-5167
assimp-9	Heap OOB Read in `MDLImporter::ImportUVCoordinate_3DGS_MDL345`	Issue #6170	CVE-2025-5168
assimp-10	Heap OOB Read in `MDLImporter::InternReadFile_3DGS_MDL345`	Issue #6171	CVE-2025-5169
assimp-11	Heap OOB Read in `MDLImporter::InternReadFile_Quake1`	Issue #6172	CVE-2025-5200
assimp-12	Heap OOB Reads in `LWOImporter::CountVertsAndFacesLWO2`	Issue #6173	CVE-2025-5201
assimp-13	Heap OOB Read in `MDLImporter::ParseSkinLump_3DGS_MDL7` / `SkipSkinLump`	Issue #6176	CVE-2025-5204
assimp-14	Heap OOB Read in `HL1MDLLoader::validate_header`	Issue #6174	CVE-2025-5202
assimp-15	Heap OOB Read in `SkipSpaces<char>`	Issue #6175	CVE-2025-5203
assimp-16	Multiple Out of Memory issues		(n.a.)
assimp-17	Multiple Issues where Requested Allocation Size Exceeds Maximum		(n.a.)
PHP-1	NULL-Pointer Dereference when using `register_tick_function` in destructor	Issue #18033	(n.a.)[*]

[†] UPX maintainers do not consider memory access violations security-relevant to UPX. They consider UPX to run in the same security context as its input.

[*] This bug does not constitute a security issue according to the project's security policy.

[$] The bug was publicly disclosed by others during the period of coordinated disclosure.

[$$] The bug had been reported to VideoLAN privately by others, but was unfixed at the time of our disclosure.

Table 6: Edges covered by different fuzzers in 24 hours starting from a saturated seed set. We can see that DDFuzz performs better than datAFLow and WingFuzz outperforms SGFuzz. The presented numbers are median result of 5 runs. The best performing libFuzzer/AFL-based fuzzer(s) for each benchmark are highlighted. Note that these coverage values cannot be compared to the results in the main text directly as they were obtained using a different machine.

	AFL-based		libFuzzer-based	
Benchmark	**datAFLow**	**DDFuzz**	**SGFuzz**	**WingFuzz**
bloaty	6 944	6 946	6 950	7 032
curl	11 513	11 514	11 526	11 552
freetype2	18 044	18 050	18 054	18 080
harfbuzz	11 355	11 366	11 354	11 371
jsoncpp	525	525	525	525
lcms	2 426	2 426	2 426	2 426
libjpeg-turbo	3 089	3 177	3 089	3 089
libpcap	4 465	4 465	4 466	4 466
libpng	2 116	2 116	2 116	2 116
libxml2	16 201	16 212	16 214	16 248
libxslt	11 492	11 500	11 527	11 505
mbedtls	4 332	4 332	4 332	4 333
openh264	9 683	9 683	9 684	9 683
openssl	5 961	5 959	5 961	5 966
openthread	4 561	4 569	4 525	4 585
proj4	10 529	10 529	10 526	10 589
re2	2 882	2 885	2 884	2 892
sqlite3	21 527	21 529	21 532	21 571
stb	2 322	2 322	2 322	2 322
systemd	(failed to build)	243	243	243
vorbis	1 388	1 389	1 383	1 383
woff2	1 255	1 255	1 255	1 255
zlib	473	473	473	473