ETH zürich

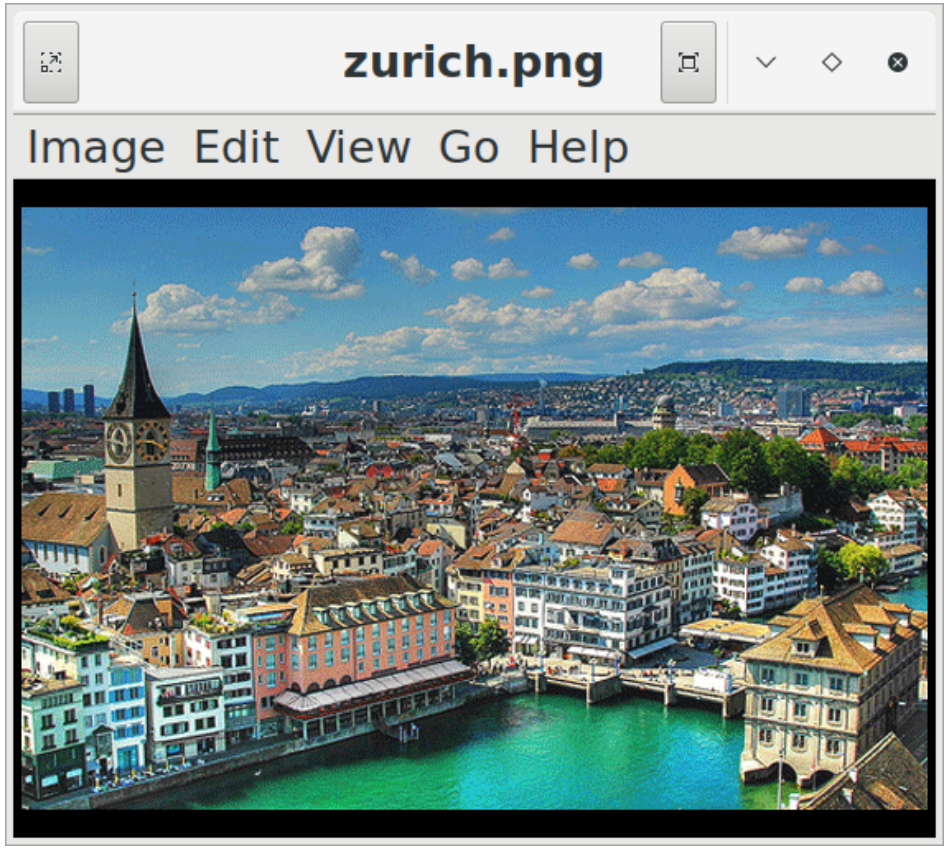
Force Open: Lightweight Black Box File Repair

Karl Wüst[1], Petar Tsankov[1], Saša Radomirović[2], Mohammad Torabi Dashti[1]

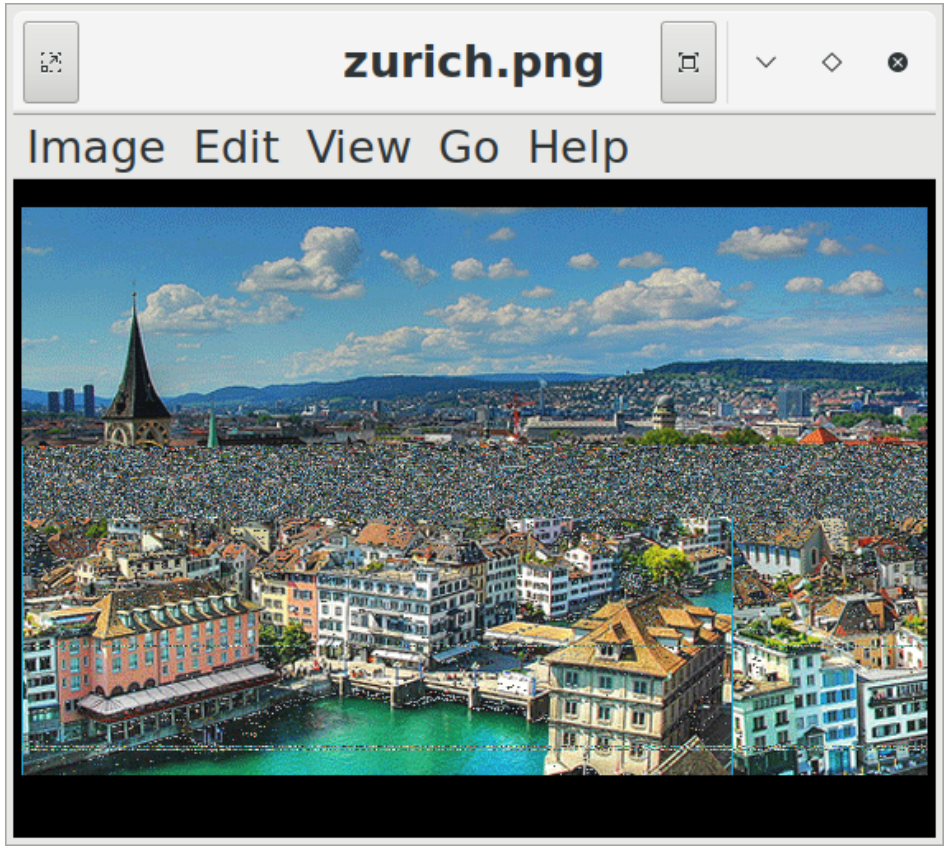
[1]ETH Zürich, [2]University of Dundee

DFRWS EU 2017

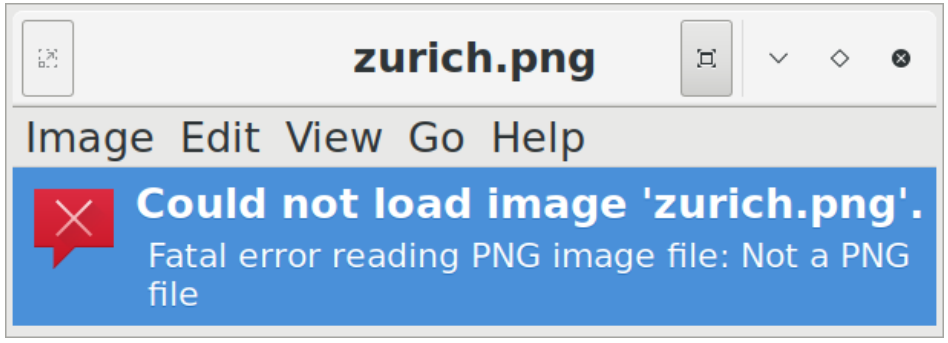
Motivation



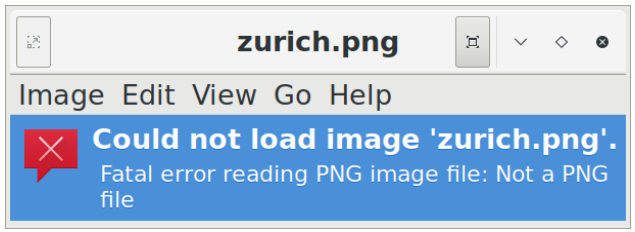
Motivation

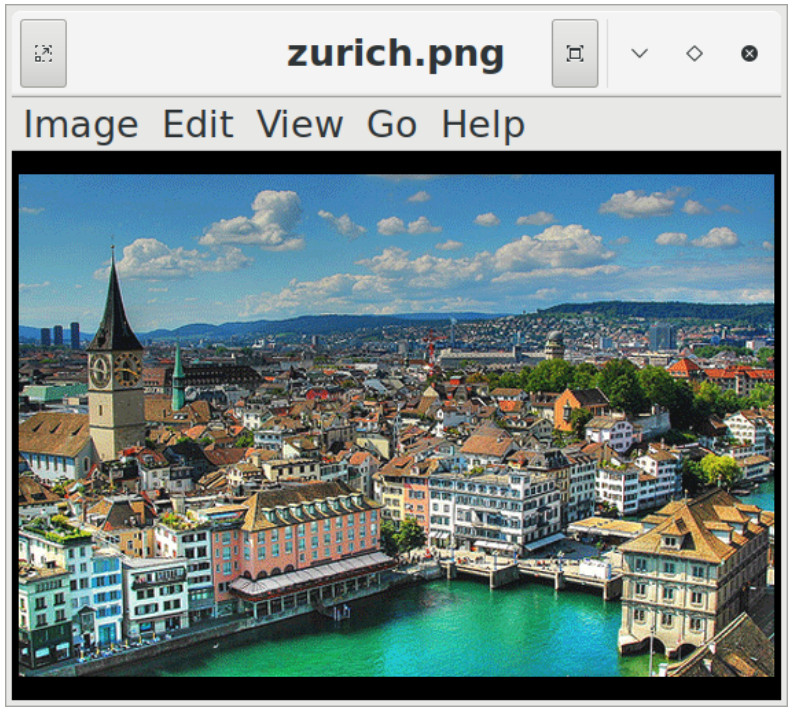


Motivation

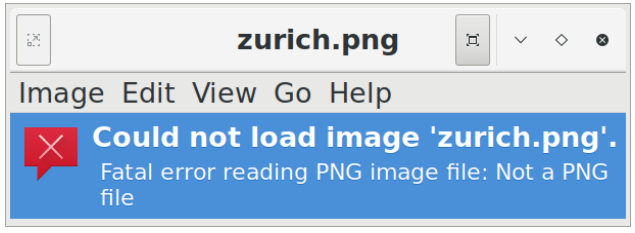

zurich.png
Image Edit View Go Help
Could not load image 'zurich.png'.
Fatal error reading PNG image file: Not a PNG file

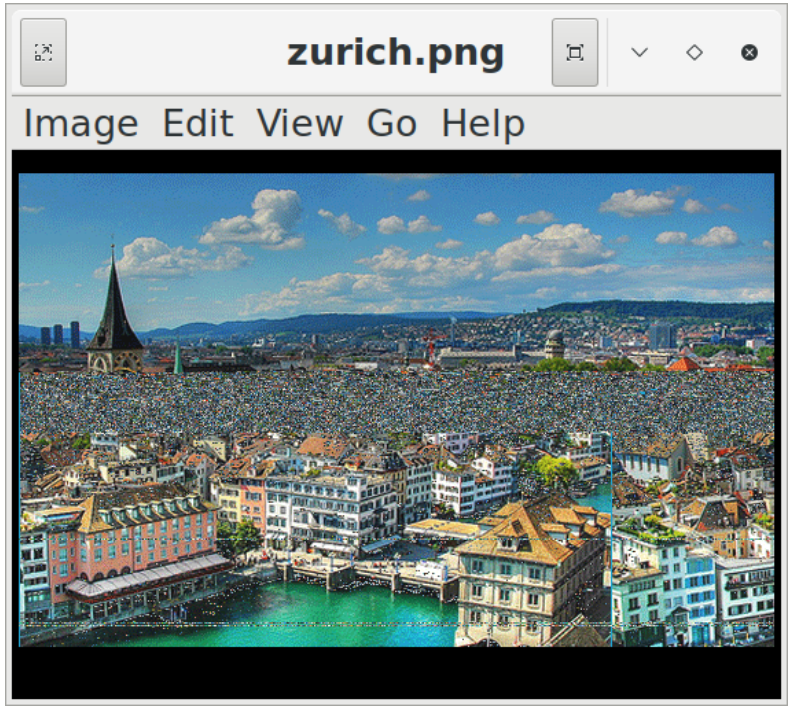
Motivation


zurich.png
Image Edit View Go Help
Could not load image 'zurich.png'.
Fatal error reading PNG image file: Not a PNG file


zurich.png
Image Edit View Go Help

Motivation


zurich.png
Image Edit View Go Help
Could not load image 'zurich.png'.
Fatal error reading PNG image file: Not a PNG file


zurich.png
Image Edit View Go Help

Goals

For an *invalid* input

- Get file viewer to produce an output
- The output should be similar to the original file

Our Approach

- Change the execution of the file viewer to open corrupted files

Our Approach

- Change the execution of the file viewer to open corrupted files
- Learn behaviour by opening (many) valid files

Our Approach

- Change the execution of the file viewer to open corrupted files
- Learn behaviour by opening (many) valid files

$\Rightarrow$ Two phases:

- Training phase
- Force open phase

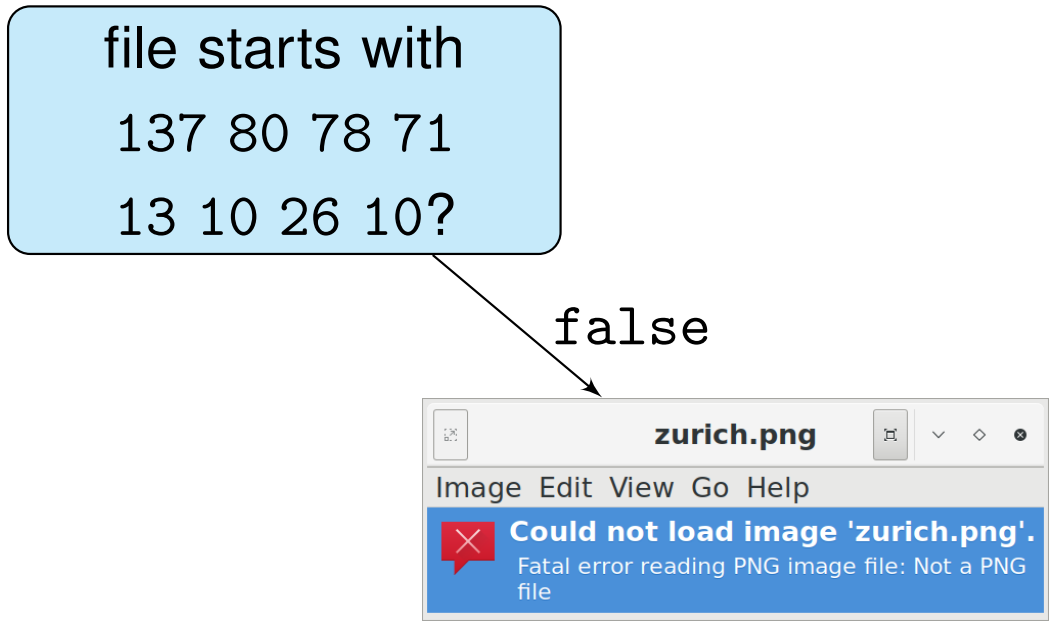
File Viewer Execution

file starts with

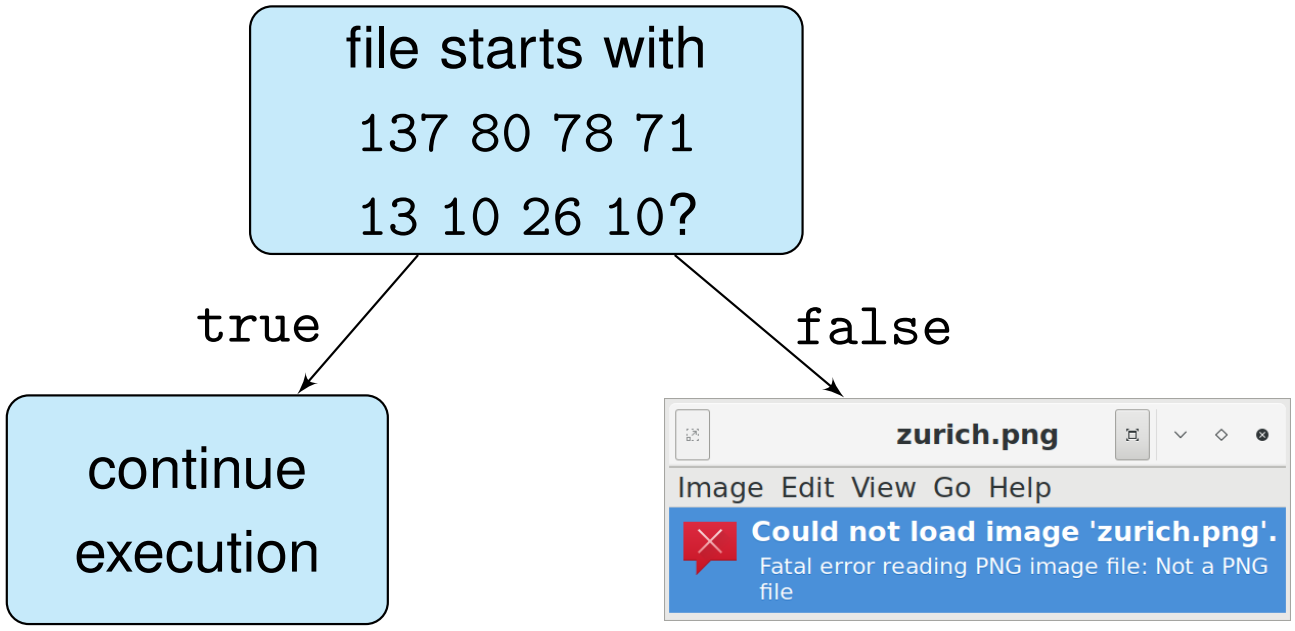
137 80 78 71

13 10 26 10?

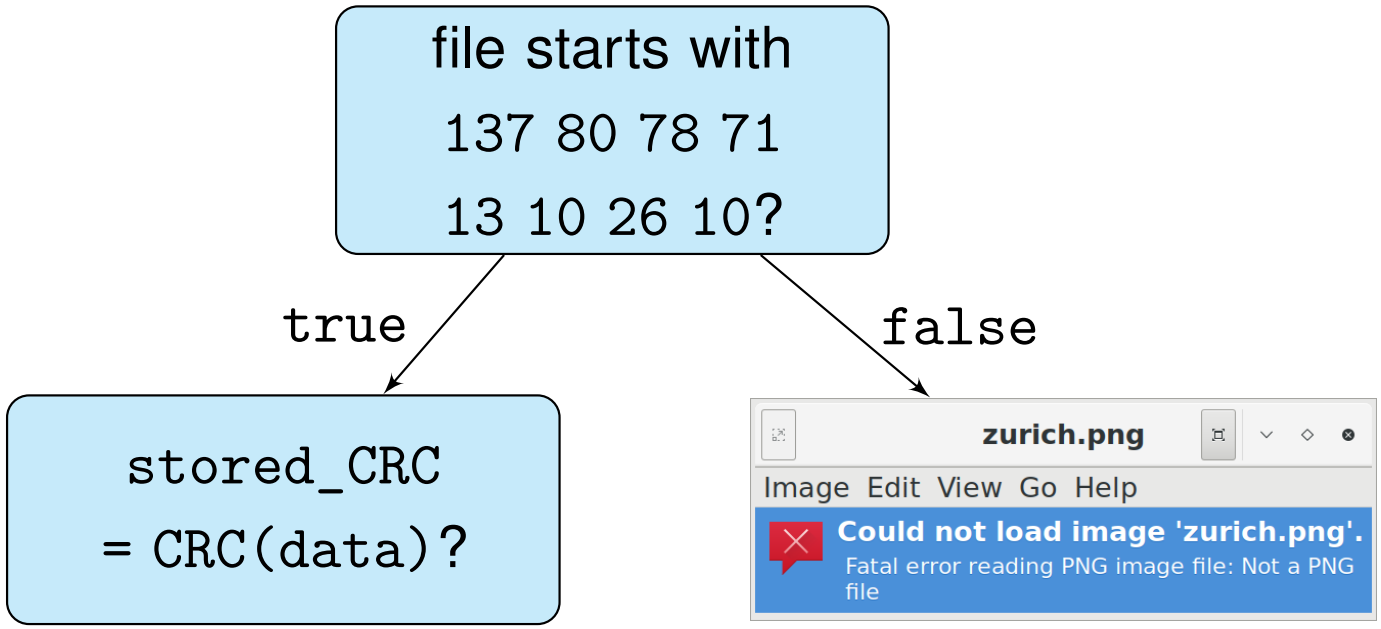
File Viewer Execution



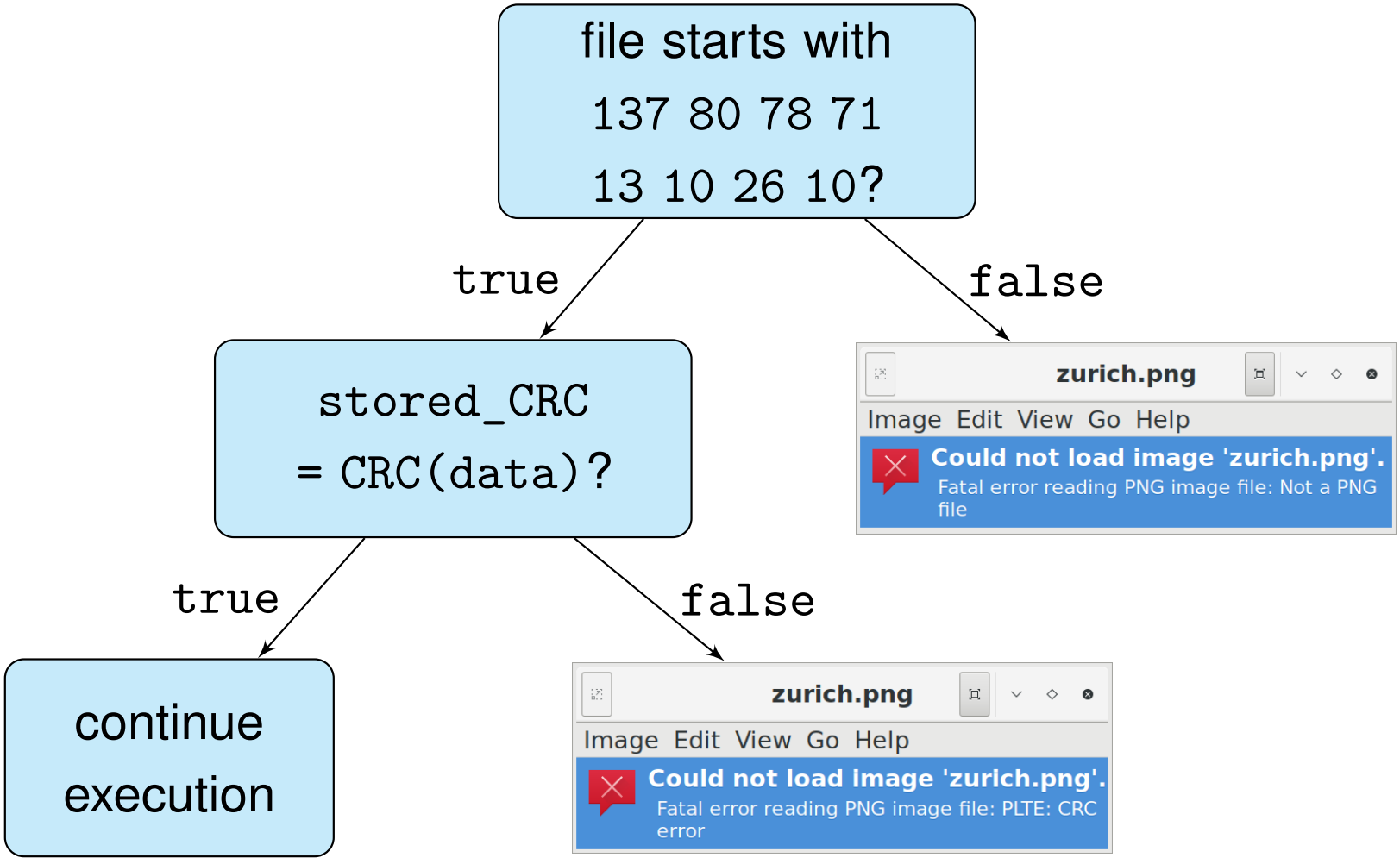
File Viewer Execution



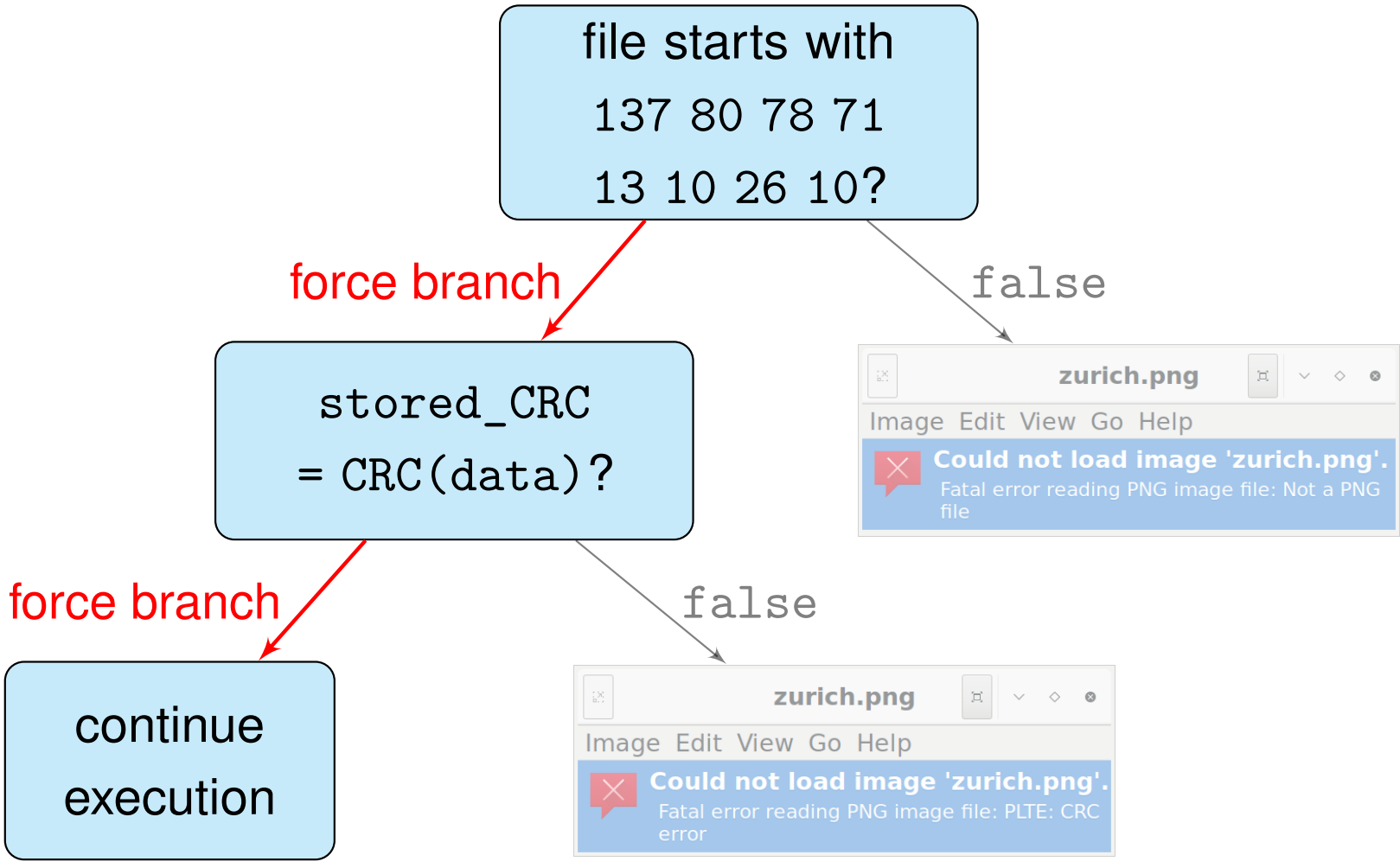
File Viewer Execution



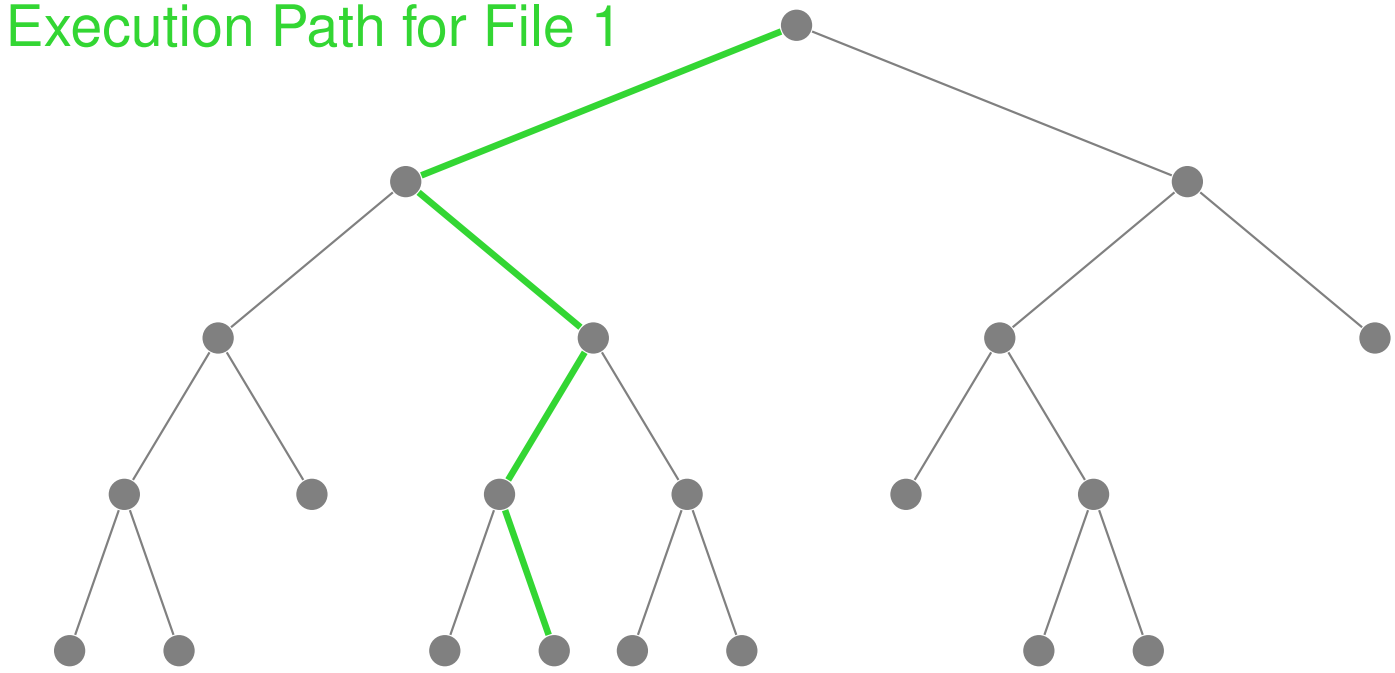
File Viewer Execution



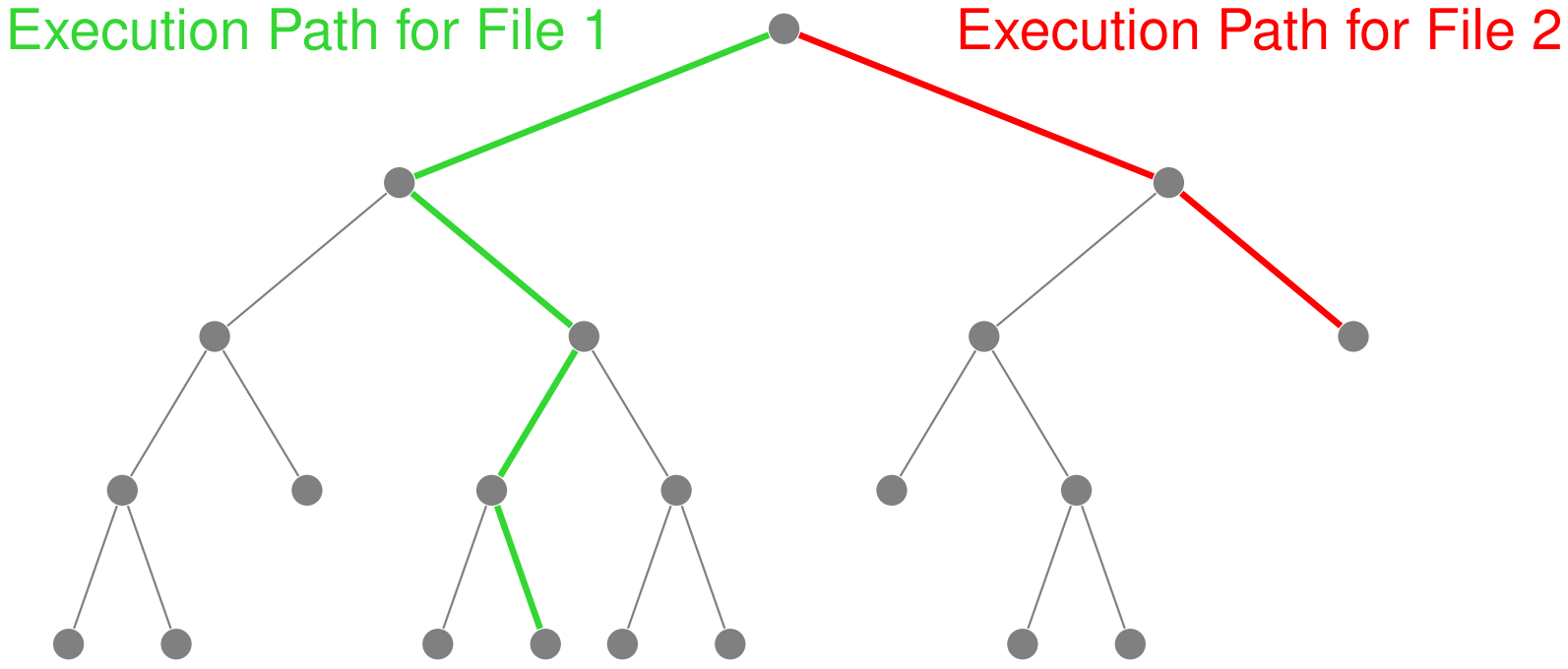
File Viewer Execution



File Viewer Execution



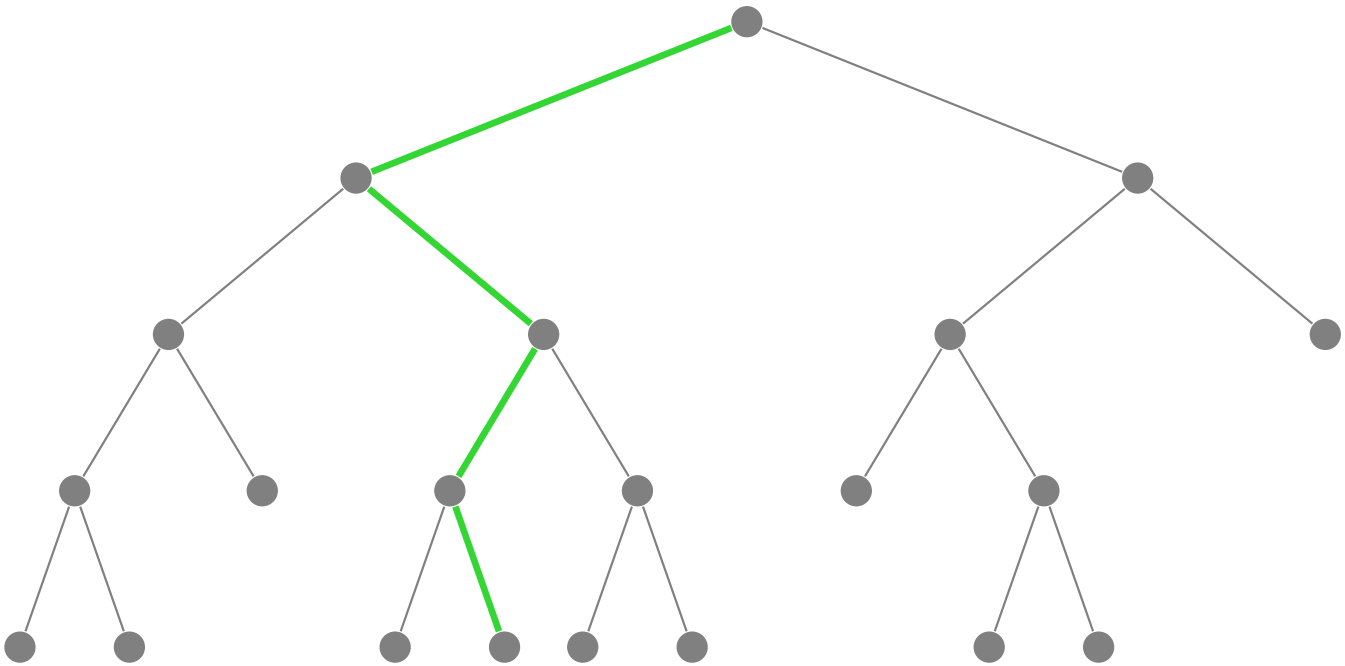
File Viewer Execution



File 1 is valid, i.e. satisfies the file format specification

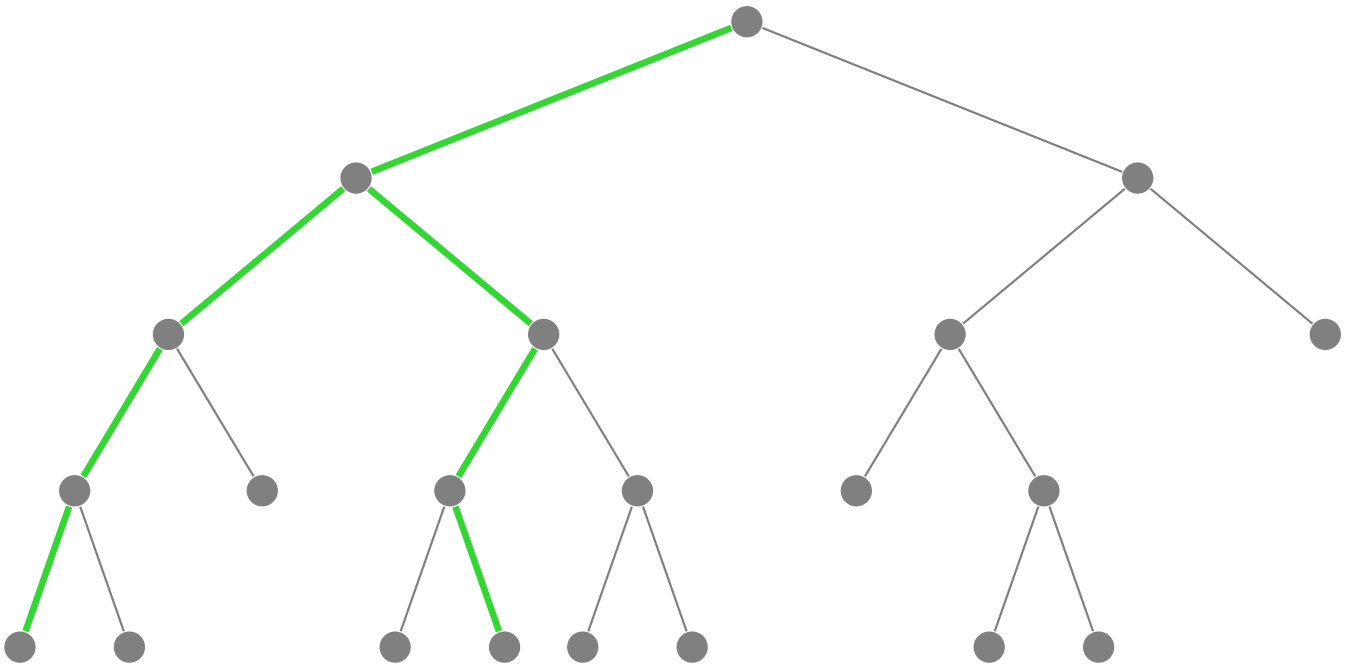
File 2 is invalid, i.e. does not satisfy the format specification

Force Open



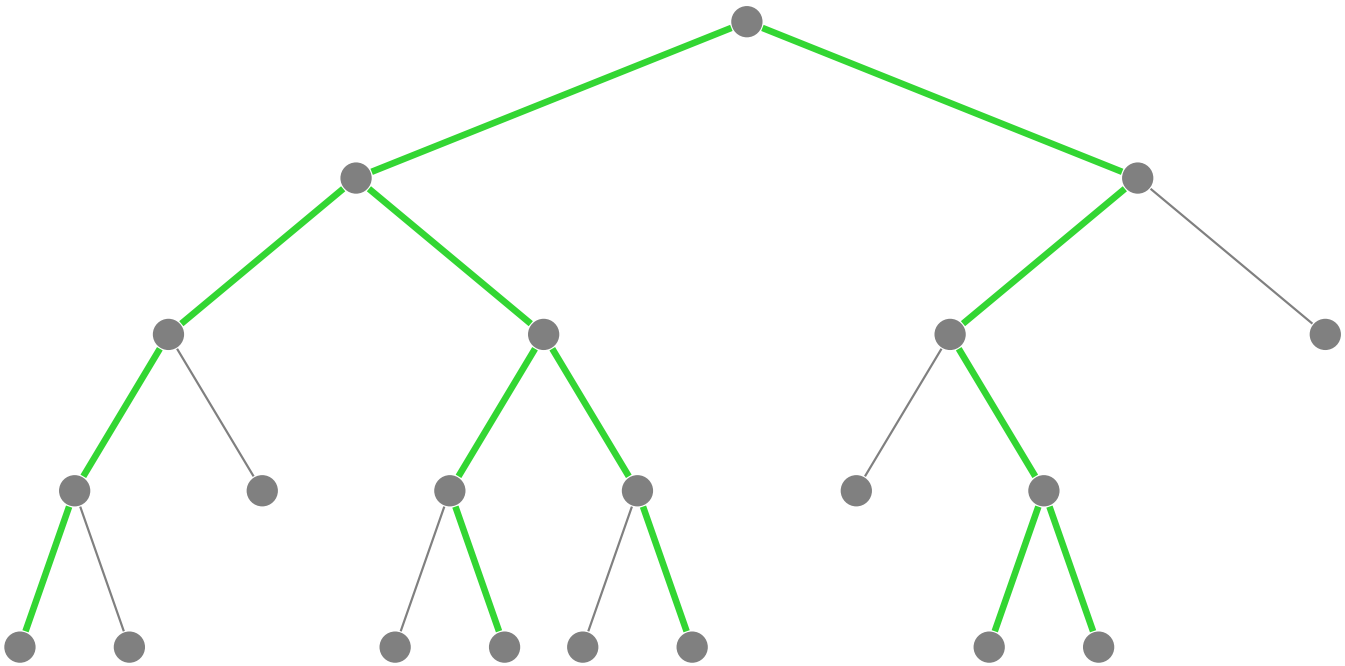
Record branches for executions with valid files

Force Open



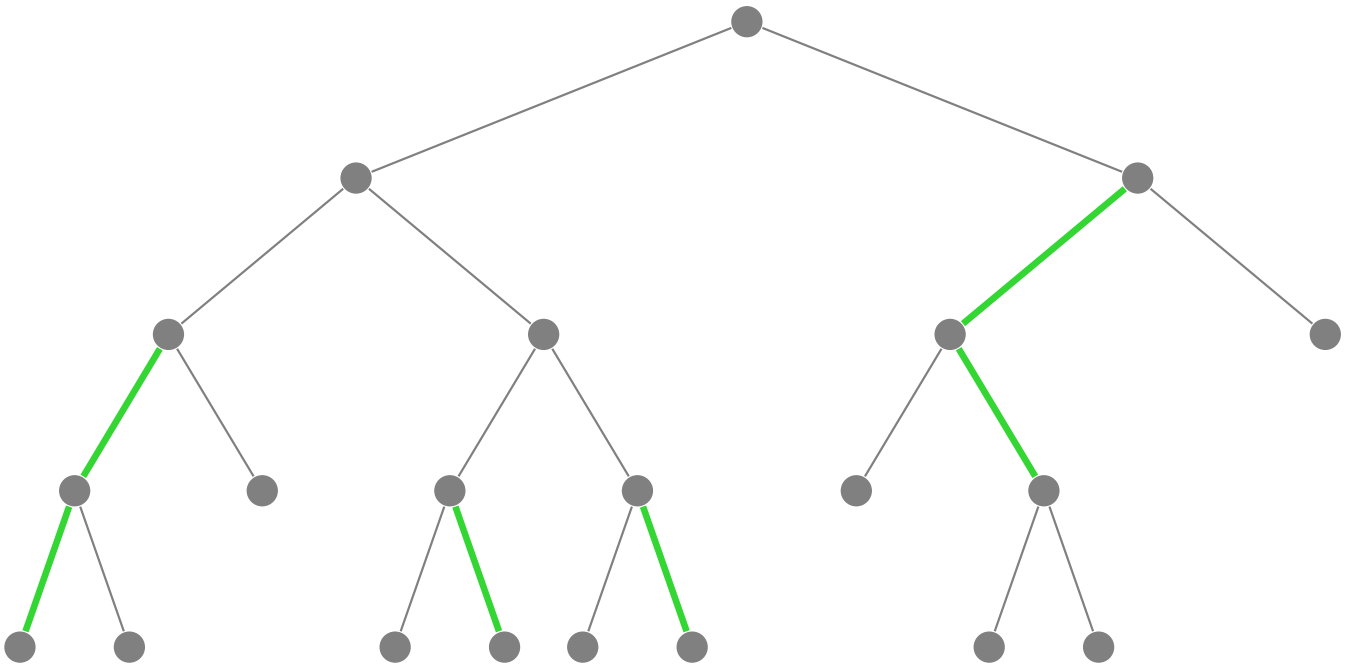
Record branches for executions with valid files

Force Open



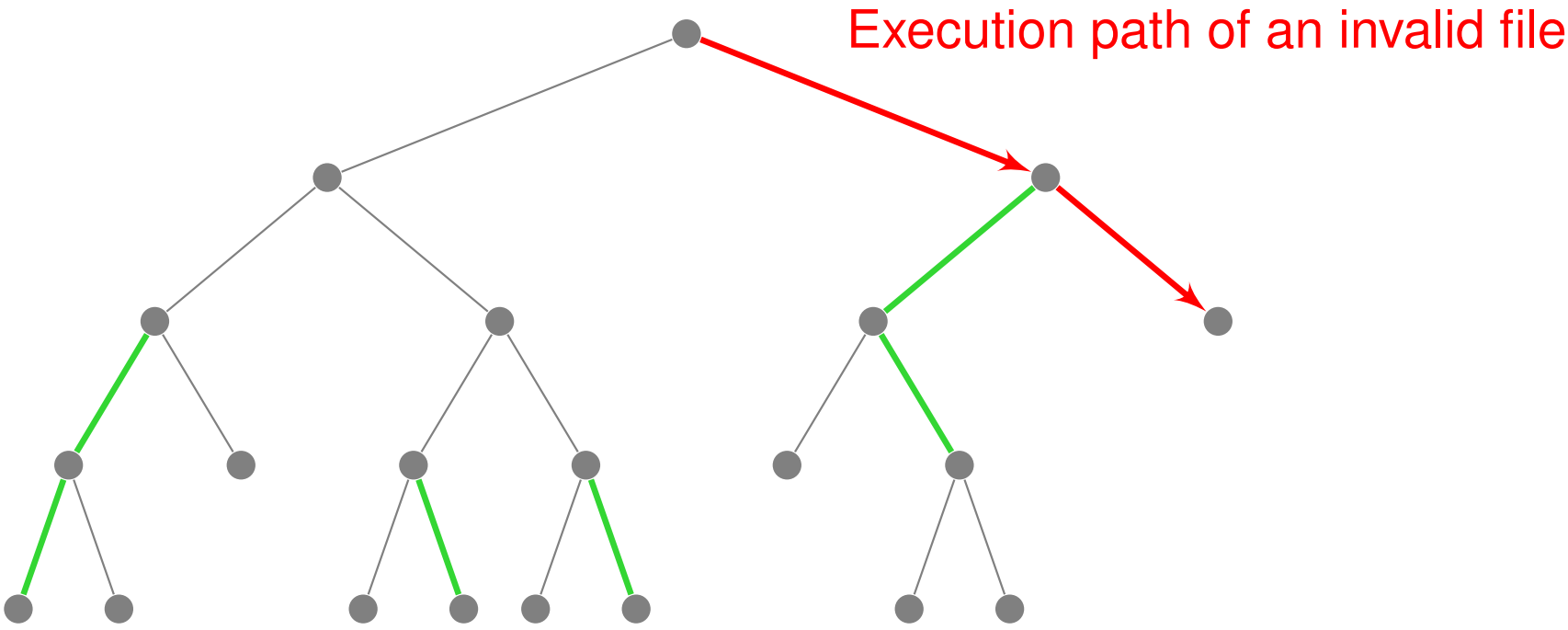
Record branches for executions with valid files

Force Open



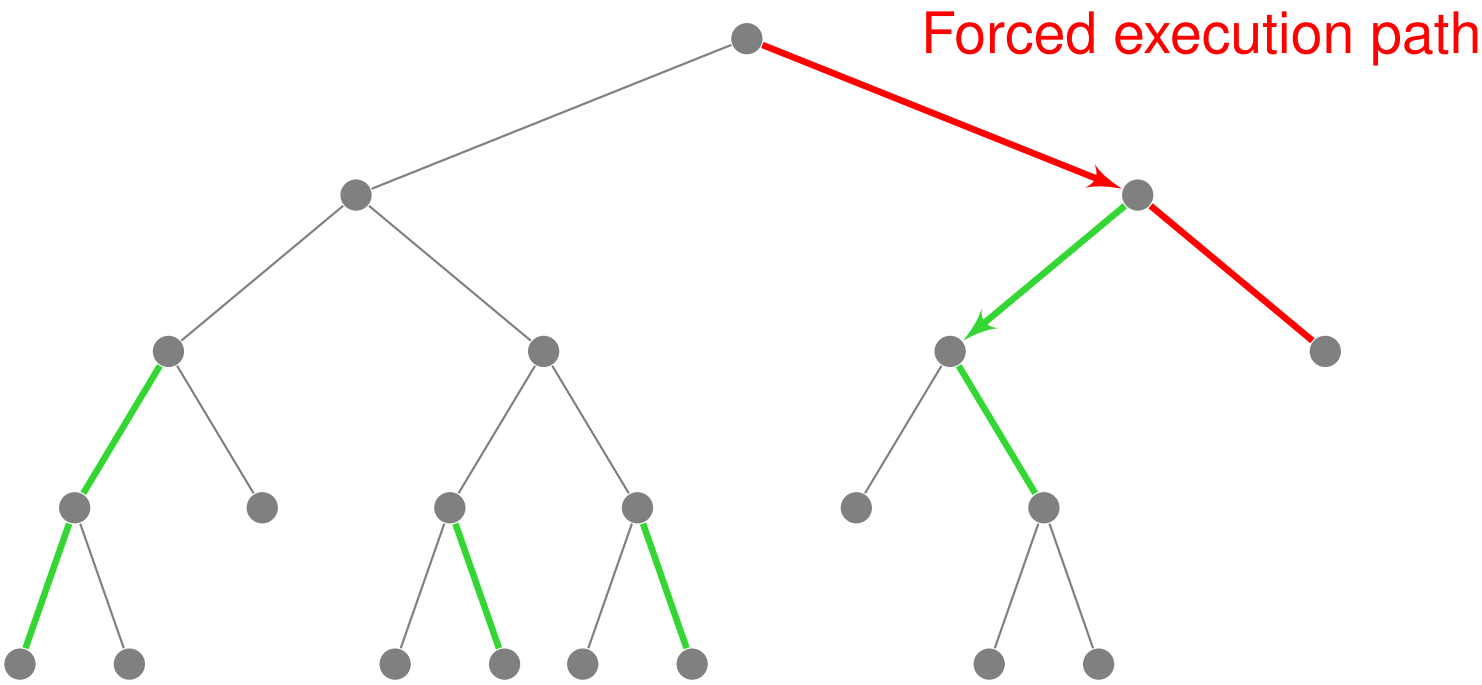
Keep branches that are always taken

Force Open



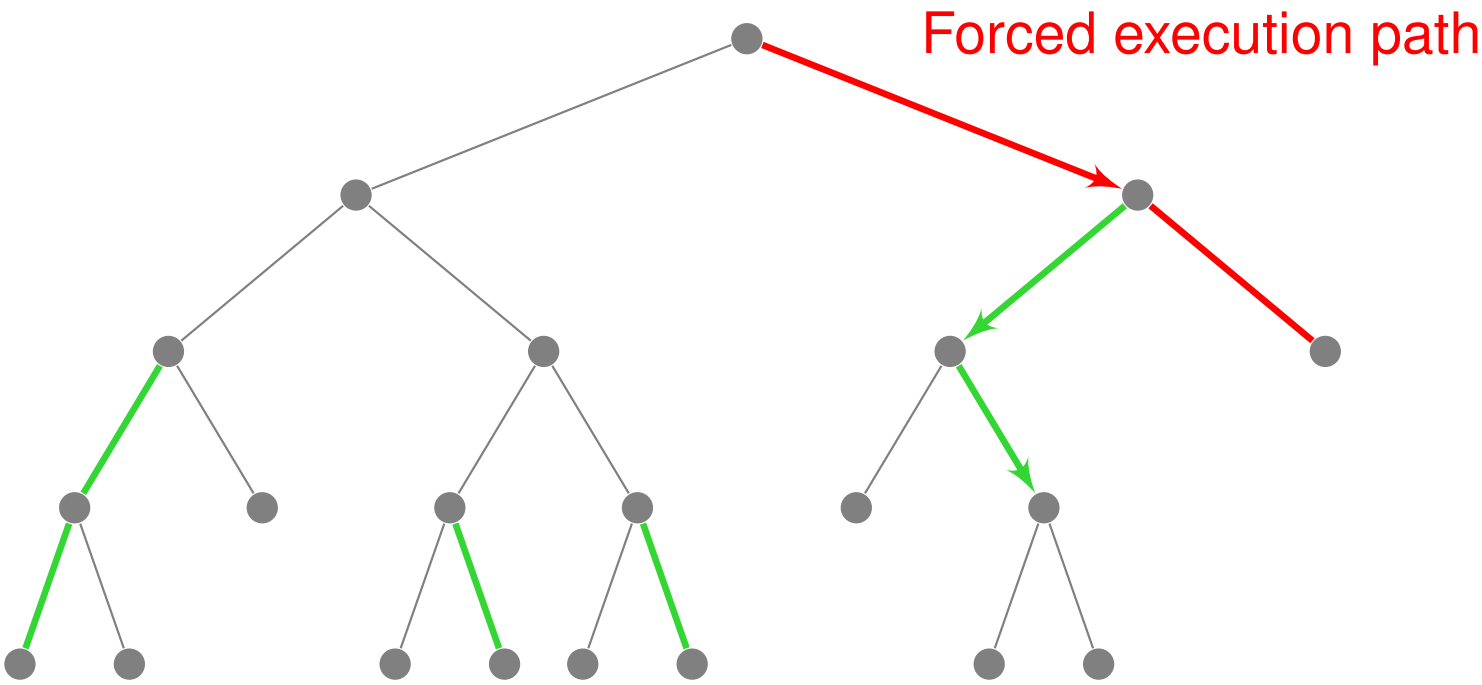
Force branches when opening invalid file

Force Open



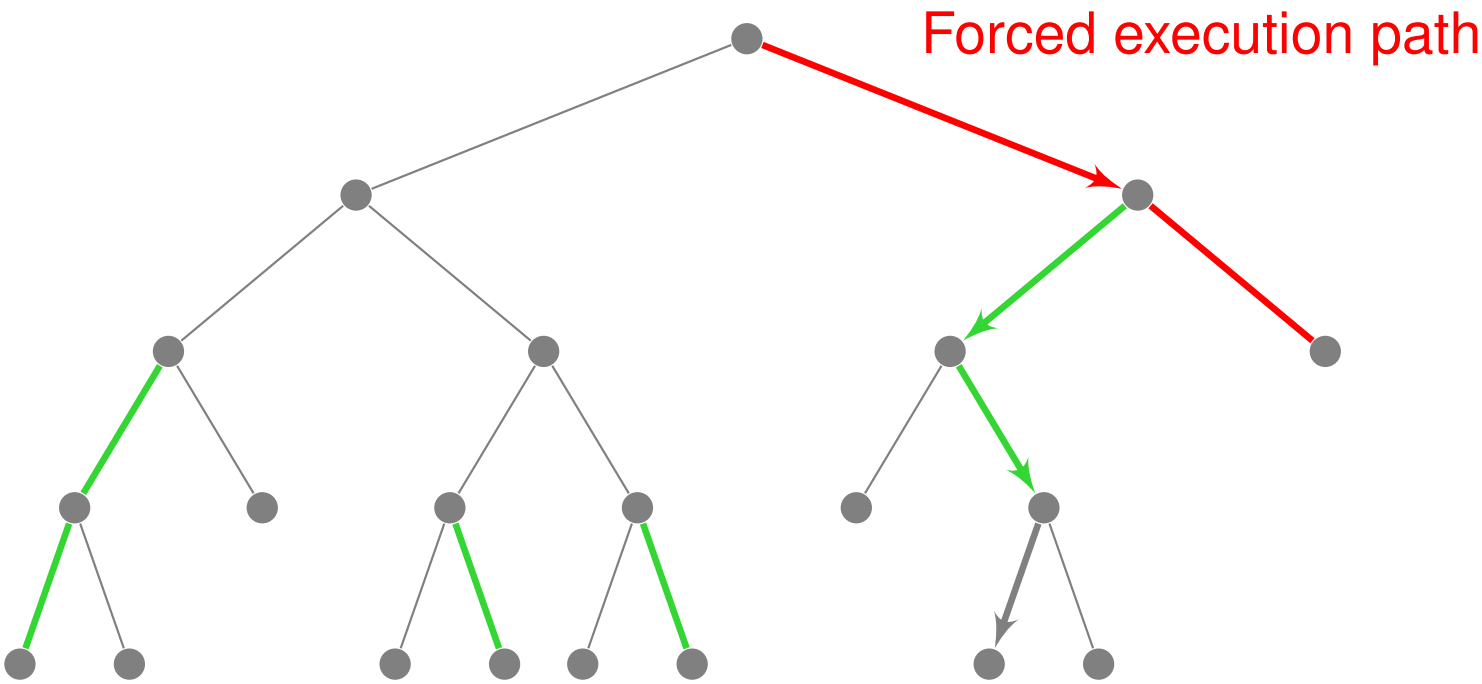
Force branches when opening invalid file

Force Open



Force branches when opening invalid file

Force Open



Force branches when opening invalid file

Black Box

- Uses binary instrumentation & execution hijacking
- Requires only a file viewer binary and valid files
- Generic: no format specific heuristics required

Implementation & Evaluation

- Tools in Python & C++

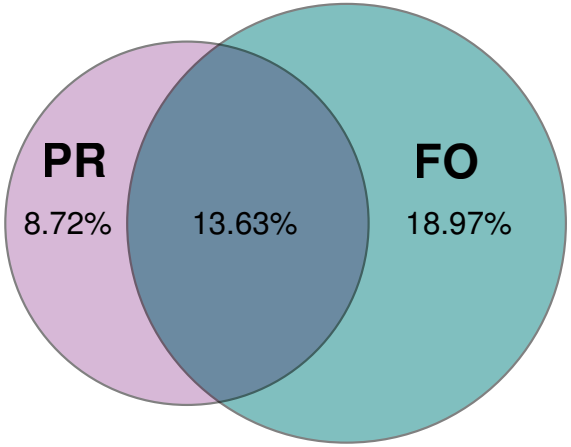
Implementation & Evaluation

- Tools in Python & C++
- Intel Pin framework for instrumentation

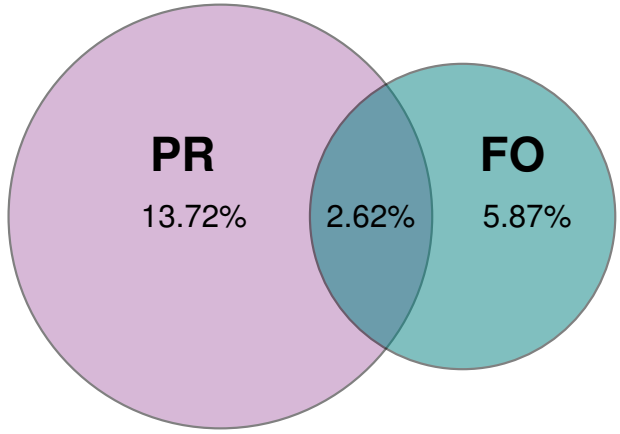
Implementation & Evaluation

- Tools in Python & C++
- Intel Pin framework for instrumentation
- Random file corruptions

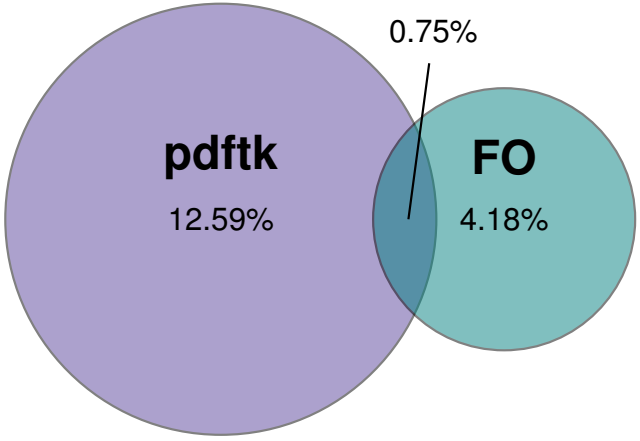
Results



3095 PNG Files



1261 JPG Files



1723 PDF Files

FO: Force Open

PR: PixRecovery

PNG 1-Byte Corruptions

Chunk	Field	# of files	# of repaired files
File Header		74	59
IHDR	chunk type	32	9
IHDR	chunk length	30	26
IHDR	chunk data (compression/filter)	25	18
IHDR	chunk data (other)	106	1
IHDR	CRC	44	33
IDAT	chunk type	55	20
IDAT	chunk length	1	0
PLTE	chunk type	3	0
PLTE	chunk length	3	0
PLTE	chunk data	70	33
PLTE	CRC	5	2
ancillary chunks	chunk type	72	50
ancillary chunks	chunk length	77	49

Limitations

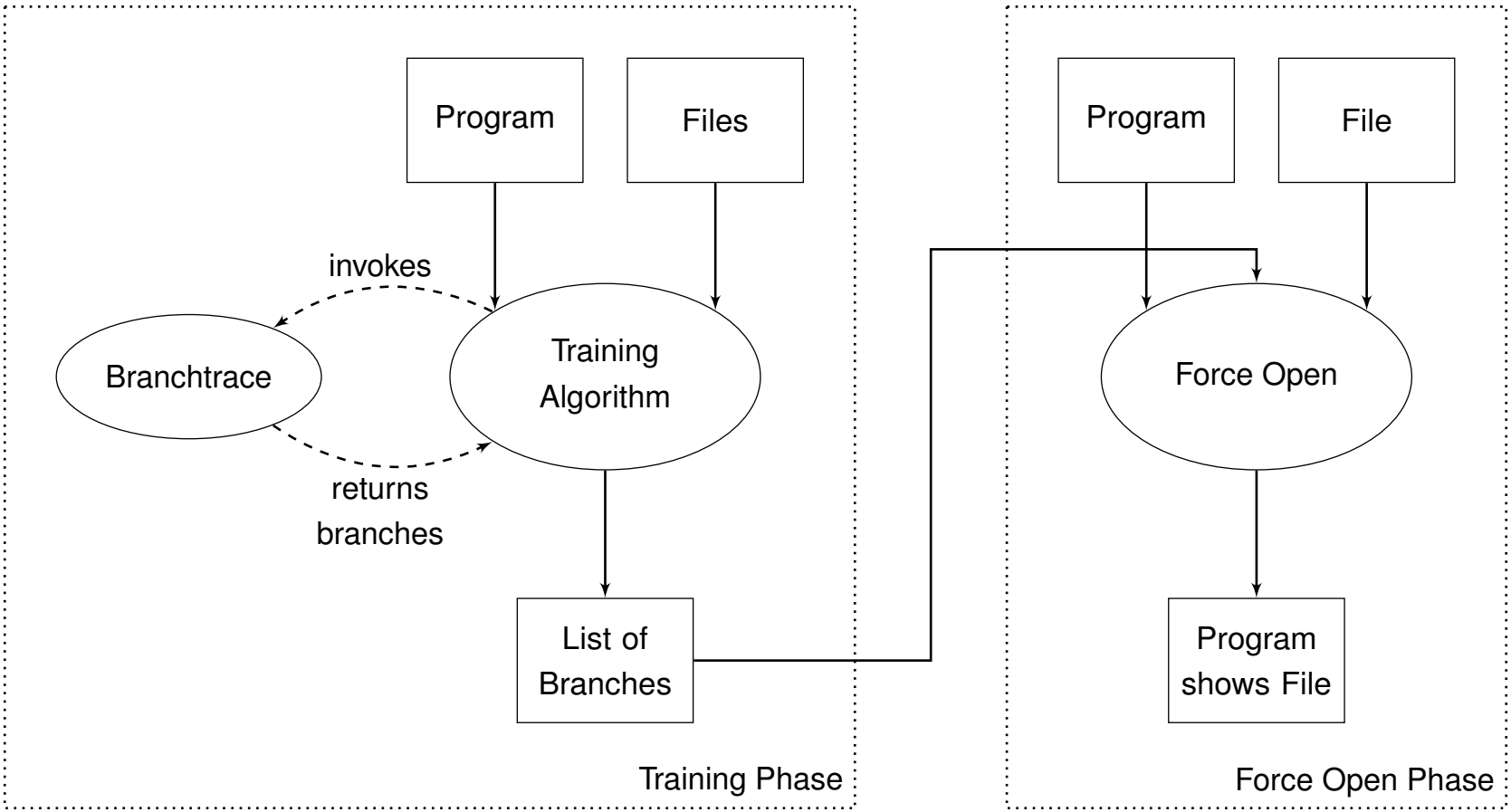
- Not suitable for highly data dependent corruptions
- Requires if-then-else statements
- Unwanted side effects (requires sandboxing)

Conclusion

- Lightweight black box approach to file repair
- Modify execution of file viewer instead of the file itself
- Applied to PNG, JPG, and PDF, but applicable to any file format
- Complementary to existing tools
- Tools available at `https://github.com/fldpi/forceopen`

Questions?

Workflow


Program
Files
invokes
Branchtrace
Training Algorithm
returns branches
List of Branches
Training Phase
Program
File
Force Open
Program shows File
Force Open Phase

Results

	Corrupt bytes	Number of files	Repaired files: Force Open (FO)		Ref. Tool (RT)		FO and RT		only FO		only RT	
PNG	1	597	306	(51.26%)	207	(34.67%)	144	(24.12%)	162	(27.14%)	63	(10.55%)
	2	624	305	(48.88%)	174	(27.88%)	119	(19.07%)	186	(29.81%)	55	(8.81%)
	4	612	215	(35.13%)	155	(25.33%)	88	(14.38%)	127	(20.75%)	67	(10.95%)
	8	632	129	(20.41%)	98	(15.51%)	46	(7.28%)	83	(13.13%)	52	(8.23%)
	16	630	54	(8.57%)	58	(9.21%)	25	(3.97%)	29	(4.60%)	33	(5.24%)
	1 – 16	3095	1009	(32.60%)	692	(22.36%)	422	(13.63%)	587	(18.97%)	270	(8.72%)
JPG	1	248	37	(14.92%)	58	(23.39%)	9	(3.63%)	28	(11.29%)	49	(19.76%)
	2	256	30	(11.72%)	45	(17.58%)	14	(5.47%)	16	(6.25%)	31	(12.11%)
	4	249	20	(8.03%)	34	(13.65%)	6	(2.41%)	14	(5.62%)	28	(11.24%)
	8	253	14	(5.53%)	25	(9.88%)	3	(1.19%)	11	(4.35%)	22	(8.70%)
	16	255	6	(2.35%)	44	(17.25%)	1	(0.39%)	5	(1.96%)	43	(16.86%)
	1 – 16	1261	107	(8.49%)	206	(16.34%)	33	(2.62%)	74	(5.87%)	173	(13.72%)
PDF	1	312	30	(9.62%)	33	(10.58%)	3	(0.96%)	27	(8.65%)	30	(9.62%)
	2	320	22	(6.88%)	41	(12.81%)	3	(0.94%)	19	(5.94%)	38	(11.88%)
	4	349	16	(4.58%)	49	(14.04%)	0	(0.00%)	16	(4.58%)	49	(14.04%)
	8	358	9	(2.51%)	48	(13.41%)	2	(0.56%)	7	(1.96%)	46	(12.85%)
	16	384	8	(2.08%)	59	(15.36%)	5	(1.30%)	3	(0.78%)	54	(14.06%)
	1 – 16	1723	85	(4.93%)	230	(13.35%)	13	(0.75%)	72	(4.18%)	217	(12.59%)