Static Analysis-based Repair of Memory Errors in C Programs

Hakjoo Oh
Korea University

2/25/2025@IFIP WG 2.4 Meeting 70 (Singapore)

PL/SE Research @Korea Univ.

- Members: ~15 graduate students
- Research area: intersection of programming languages (PL) and software engineering (SE)
  - program analysis and testing
  - program synthesis and repair

- Publication: PL, SE, and Security
  - PL: POPL('22),PLDI('12,'14,'20,'24),OOPSLA('15,'17a,'17b,'18a,'18b,'19,'20,'23,'24a,'24b,25)
  - SE: ICSE('17,'18,'19,'20,'21'22a,'22b,'23a,'23b,'23c), FSE('18,'19,'20,'21,'22,'23), ASE('18,'24a,'24b)
  - Security: IEEE S&P('17,'20), USENIX Security('21,'23)

http://kupl.github.io

SEOUL

Line 1
Line 2
Line 3
Line 4
Line 5
Line 6
Line 7
Line 8
Line 9
A'REX(Airport Railroad)
Suin Bundang Line
ShinBundang Line
Gyeongui·Jungang Line
Gyeongchun Line
Ui-Sinseol LRT Line
Sillim LRT Line
Railway
Transfer Station
Train Station

Museum/Gallery
Performance Hall
Shopping
Casino
Sightseeing
Local Hot Spot
Multi-purpose Cultural Space
Hotel
Bus Terminal
Tourist Information Center

0 800 1600m

N
S

A'REX (Airport Railroad)
Operating Hours 05:15-23:50

Incheon Int'l Airport Terminal 2 · Incheon Int'l Airport Terminal 1 · Incheon Int'l Airport Cargo Terminal · Unseo · Yeongjong · Cheongna Int'l City · Geomam · Gyeyang · Gimpo Int'l Airport
Express Train
Seoul Station · Gongdeok · Hongik Univ. · Digital Media City · Magongnaru

Korea Univ.

Sightseeing

D3 108 Steps
D1 April 19th National Cemetery
C4 63 Square
D2 Gyeonghoegung Palace
C3 Gyeongui Line Forest Park
E2 Gyeongchun Line Forest Trail
D3 Gyeonghuigung Palace
B4 Gocheok Sky Dome
D2 Gwanghwamun
D4 Seoul National Cemetery
C3 National Assembly Building
D2 Gilsangsa Temple
D2 Naksan Park
D3 Namsangol Hanok Village
D3 Namsan Baekbeom Square
D3 Namsan Seoul Tower
D3 Namsan Botanical Garden
D3 Deoksugung Palace
C3 Dongnimmun Gate
F4 Lotte World
F4 Lotte World Tower
D2 Mareonnier Park
D3 Myeong-dong Cathedral
F4 Mongchontoseong Fortress
D3 Bosingak Belfry
E4 Bongsunsa Temple
E2 Dream Forest
D2 Bugak Skyway
D2 Bugak Palgakjeong
D2 Bukchon Hanok Village
B2 Sangam DMC
A3 Seoseoulhosu Park
D1 Seoul Hiking Tourism Center
D3 Seoullo 7017
E3 Seoul Forest
A3 Seoul Botanic Park
F3 Seoul Amsa-dong Historic Site
B3 Seoul World Cup Stadium
B3 Seoul Battleship Park
C5 Hoamsanseong Fortress, Seoul
E4 Seokchon Lake
B3 Seonyudo Park
E4 Seonjeongneung Royal Tombs
D3 Sungnyemun
E5 Yangjae Citizen's Forest
D3 City Hall, Seoul Square
A3 Yangcheon Hyanggyo
B3 Yanghwajin Foreign Missionary Cemetery
E3 Children's Grand Park
F4 Olympic Park
D3 Yongsan Family Park
B3 World Cup Park
C1 Eunpyeong Hanok Village
E2 Ureung Royal Tomb
D3 Seoul Central Mosque
C2 Jahamun Tunnel
E4 Jamsil Sports Complex
C3 Jeoldusan Martyrs Shrine
D3 Jogyesa Temple
D3 Jongmyo Shrine
C1 Jingwansa
D2 Changgyeonggung Palace
D2 Changdeokgung Palace
D2 Changgyeonggung Gate
D2 Cheonggwadae
D2 Pungnaptoseong Earthen Fortification
F1 Taereung · Gangneung Royal Tomb
E1 K-Star Road
A4 Hangdong Railroad Trail
D2 Pureun Arboretum
E5 Heonilleung Royal Tomb
C5 Hoapsa Temple
C3 Hyochang Park
D3 Heunginjimun Gate

Local Hot Spot

D4 Garosu-gil·Seosu-gil
D4 Gangnam Station Street
E3 Kondae Rodeo Street
D3 Gyeongnidan-gil
D2 Daehangno Street
B3 Mangridan Road
D3 Myeong-dong
B4 Mullae Art Village
C2 Buam-dong
D2 Samcheong-dong
C3 Sangsu Cafe Street
C4 Syarosu-gil
D4 Seorae Village
D2 Seochon Cafe and Restaurant Street
E3 Seongsudong Handmade Shoe Street
E3 Seongsudong Cafe Street
F4 Songridan-gil
C3 Sinchon Edae Street
E3 Apgujeong Rodeo Street
E4 Yangjaecheon Cafe Street
C3 Yeonnam-dong Café District
C3 Yeonhui Multi-Culture Street
D3 Usadan-gil
D3 Itaewon Street
D2 Ikseon-dong Hanok Village
D3 Insa-dong
C3 Hapjeong Cafe Street
D3 Haebangchon Village
C3 Hongdae Street

Museum/Gallery

C2 Hongjie Yuyeon
C3 SeMA Bunker
D2 Kansong Art Museum
A3 Gyeomjae Jeongseon Art Museum
D3 National Theater of Korea
C3 National Meteorological Museum of Korea
C2 National Memorial of the Korea Provisional Government
D2 National Children's Science Center
D3 National Museum of Korea
D4 National Hangeul Museum
A3 Nationanl Aviation Museum of Korea
D3 MMCA Deoksugung
D2 MMCA Seoul
E1 Dooly Museum
D5 Nam Seoul Annex Building of Seoul Museum of Art
D3 Seoul Museum of History
C2 Seoul Museum
F4 SOMA Museum of Art
F4 Songpa Book Museum
D4 Hangaram Arts Center Museum
D3 Yongsan Art & Craft Center
D2 Literary Museum of Yun Dong-Ju the Poet
C1 Eunpyeong History Hanok Museum
D3 War Memorial of Korea
E3 Cheonggyecheon Museum
C2 Korea Furniture Museum
B2 Korean Film Museum
F4 Seoul Baekje Museum
D2 Seoul City Wall Museum
A3 Heojun Museum
D2 Whanki Art Museum

Shopping

C3 IFC Mall
F4 Garak Market
E4 Gangnam Station Underground Shopping Center
E3 Galleria Department Store Luxury Hall
D4 Goto Mall
D3 Gwangjang Market
C2 Nagwon Instrument Arcade
D3 Namdaemun Market
C4 Noryangjin Fisheries Wholesale Market
B4 Daerim Central Market (China Town)
C3 The Hyundai Seoul
D3 Dongdaemun Shopping Complex
B3 Dongmyo Flea Market
F4 Lotte Duty-Free World-Tower Store
A3 Lotte Mall Gimpo Airport
E4 Lotte Department Store Jamsil Branch
B3 Mangwon Market
E2 Seoul Yangnyeong Market
B3 Seoul Folk Flea Market
D4 Central City
D3 The SHILLA Duty-Free Seoul Store
C3 I'PARK mall / Shilla I'PARK Duty
E3 Cheongdam-dong Luxury Fashion Street
B4 Time Square
C3 Tongin Market
D3 Pyeonghwa Market
B4 Hyundai Department Store D-Cube City
B3 Hyundai Department Store Mokdong Store
E4 Hyundai Department Store Trade Center Store
D3 Hyundai Department Store Apgujeong Main Store

Multi-purpose Cultural Space

C3 KT&G Sangsang Madang
C4 Nodeulseom Islet
D3 Dongdaemun Design Plaza (DDP)
B3 Oil Tank Culture Park
C3 Culture Station Seoul 284
F4 Seoul Book Bogo
E2 Seoul K-medi Center
D4 Sebitseom Islet
D3 Sewoon Plaza
E3 Under Stand Avenue
D4 Daily teum
E3 CommonGround
D4 COEX
D3 Korea House
E4 Horim Art center

Performance Hall

A3 LG Arts Center
D4 Kwanglim Arts Center
D4 National Gugak Center
C3 National Theater Company of Korea
D3 National Theater of Korea
D3 Jeongdong Theater
B4 D-Cube Link Art Center
D3 Blue Square
D3 Sejong Center for the Performing Arts
D4 Seoul Arts Center
F3 Universal Arts Center
E4 Korea Cultural House

Hotel

C3 1 Grand Mercure Ambassador Hotel and Residences Seoul Yongsan
F3 2 Grand Walkerhill Seoul
E4 3 Grand InterContinental Seoul Parnas
D3 4 Grand Hyatt Seoul
C3 5 Novotel Suites Ambassador Seoul Yongsan
D3 6 Novotel Ambassador Seoul Dongdaemun Hotels & Residences
C3 7 Novotel Ambassador Seoul Yongsan
D3 8 The Plaza Hotel Seoul
D3 9 Lotte Hotel Seoul
F4 10 Lotte Hotel World
A3 11 Mayfield Hotel
D3 12 Mondrian Seoul Itaewon
F3 13 Vista Walkerhill Seoul
C2 14 Swiss Grand Hotel Seoul
F4 15 Signiel Seoul
E3 16 Andaz Seoul Gangnam
D3 17 The Westin Chosun Seoul
E4 18 InterContinental Seoul COEX
E4 19 Josun Palace Seoul Gangnam
C3 20 Conrad Seoul
E4 21 Park Hyatt Seoul
C3 22 Fairmont Ambassador Seoul
D3 23 Four Seasons Hotel Seoul
D3 24 The Shilla Seoul
D3 25 JW Marriott Dongdaemun Square Seoul
D4 26 JW Marriott Hotel Seoul

visitseoul.net

SEOUL

- Museum/Gallery
- Performance Hall
- Shopping
- Casino
- Sightseeing
- Local Hot Spot
- Multi-purpose Cultural Space
- Hotel
- Bus Terminal
- Tourist Information Center

- Line 1
- Line 2
- Line 3
- Line 4
- Line 5
- Line 6
- Line 7
- Line 8
- Line 9
- A'REX(Airport Railroad)
- Suin Bundang Line
- ShinBundang Line
- Gyeongui·Jungang Line
- Gyeongchun Line
- Ui-Sinseol LRT Line
- Sillim LRT Line
- Railway
- Transfer Station
- Train Station

A'REX (Airport Railroad)
Operating Hours 05:15-23:50

Incheon Int'l Airport Terminal 2 · Incheon Int'l Airport Terminal 1 · Incheon Int'l Airport Cargo Terminal · Unseo · Yeongjong · Cheongna Int'l City · Geomam · Gyeyang · Gimpo Int'l Airport
Express Train
Seoul Station · Gongdeok · Hongik Univ. · Digital Media City · Magongnaru

Sightseeing

D3	108 Steps
D1	April 19th National Cemetery
C4	63 Square
B2	Gyeonghuigung Palace
C3	Gyeongui Line Forest Park
E2	Gyeongchun Line Forest Trail
C3	Gyeonghuigung Palace
B4	Gocheok Sky Dome
D2	Gwanghwamun
D4	Seoul National Cemetery
C3	National Assembly Building
D2	Gilsangsa Temple
D2	Naksan Park
D3	Namsangol Hanok Village
D3	Namsan Baekbeom Square
D3	Namsan Seoul Tower
D3	Namsan Botanical Garden
D3	Deoksugung Palace
C3	Dongnimmun Gate
F4	Lotte World
F4	Lotte World Tower
D2	Mareonnier Park
D3	Myeong-dong Cathedral
F4	Mongchontoseong Fortress
D3	Bosingak Belfry
E4	Bongsunsa Temple
E2	Dream Forest
D2	Bugak Skyway
D2	Bugak Palgakjeong
D2	Bukchon Hanok Village
B2	Sangam DMC
A3	Seoseoulhosu Park
D1	Seoul Hiking Tourism Center
D3	Seoullo 7017
E3	Seoul Forest
A3	Seoul Botanic Park
F3	Seoul Amsa-dong Historic Site
B3	Seoul World Cup Stadium
B3	Seoul Battleship Park
C5	Hoamsanseong Fortress, Seoul
F4	Seokchon Lake
B3	Seonyudo Park
E4	Seonjeongneung Royal Tombs
D3	Sungnyemun
E5	Yangjae Citizen's Forest
D3	City Hall, Seoul Square
A3	Yangcheon Hyanggyo
B3	Yanghwajin Foreign Missionary Cemetery
E3	Children's Grand Park
F4	Olympic Park
D3	Yongsan Family Park
B3	World Cup Park
C1	Eunpyeong Hanok Village
E2	Ureung Royal Tomb
D3	Seoul Central Mosque
C2	Jahamun Tunnel
E4	Jamsil Sports Complex
C3	Jeoldusan Martyrs Shrine
D3	Jogyesa Temple
D3	Jongmyo Shrine
C1	Jingwansa
D2	Changgyeonggung Palace
D2	Changdeokgung Palace
D2	Changuimun Gate
D2	Cheongwadae
D2	Pungnaptoseong Earthen Fortification
F1	Taereung · Gangneung Royal Tomb
B3	K-Star Road
A4	Hangdong Railroad Trail
D2	Pureun Arboretum
E5	Heonilleung Royal Tomb
C5	Hoapsa Temple
C3	Hyochang Park
D3	Heunginjimun Gate

Local Hot Spot

D4	Garosu-gil·Seosu-gil
D4	Gangnam Station Street
E3	Kondae Rodeo Street
D3	Gyeongridan-gil
D2	Daehangno Street
B3	Mangridan Road
D3	Myeong-dong
B4	Mullae Art Village
C2	Buam-dong
D2	Samcheong-dong
C3	Sangsu Cafe Street
C4	Syarosu-gil
D4	Seorae Village
D2	Seochon Cafe and Restaurant Street
E3	Seongsudong Handmade Shoe Street
E3	Seongsudong Cafe Street
F4	Songridan-gil
C3	Sinchon Edae Street
E3	Apgujeong Rodeo Street
E4	Yangjaecheon Cafe Street
C3	Yeonnam-dong Café District
C3	Yeonhui Multi-Culture Street
D3	Usadan-gil
D3	Euljiro
D3	Itaewon Street
D2	Ikseon-dong Hanok Village
D3	Insa-dong
C3	Hapjeong Cafe Street
D3	Haebangchon Village
C3	Hongdae Street

Museum/Gallery

C2	Hongjc Yuyeon
C3	SeMA Bunker
D2	Kansong Art Museum
A3	Gyeomjae Jeongseon Art Museum
D3	National Theater of Korea
C3	National Meteorological Museum of Korea
C2	National Memorial of the Korea Provisional Government
D2	National Children's Science Center
D3	National Museum of Korea
D4	National Hangeul Museum
A3	Nationanl Aviation Museum of Korea
D3	MMCA Deoksugung
D2	MMCA Seoul
E1	Dooly Museum
D3	Leeum, Samsung Museum of Art
C2	Seodaemun Museum of Natural History
C3	Seodaemun Prison History Hall
E3	Seoul Children Museum
E2	Seoul Urban Life Museum
D3	Seoul Museum of Art Seosomun Main Branch
D5	Nam Seoul Annex Building of Seoul Museum of Art
D3	Seoul Museum of History
C2	Seoul Museum
F4	SOMA Museum of Art
F4	Songpa Book Museum
D4	Hangaram Arts Center Museum
D3	Yongsan Art & Craft Center
D2	Literary Museum of Yun Dong-Ju the Poet
C1	Eunpyeong History Hanok Museum
D3	War Memorial of Korea
E3	Cheonggyecheon Museum
C2	Korea Furniture Museum
B2	Korean Film Museum
F4	Seoul Baekje Museum
D2	Seoul City Wall Museum
A3	Heojun Museum
D2	Whanki Art Museum

Shopping

C3	IFC Mall
F4	Garak Market
E4	Gangnam Station Underground Shopping Center
E3	Galleria Department Store Luxury Hall
D4	Goto Mall
D3	Gwangjang Market
D2	Nagwon Instrument Arcade
D3	Namdaemun Market
C4	Noryangjin Fisheries Wholesale Market
B4	Daerim Central Market (China Town)
C3	The Hyundai Seoul
D3	Dongdaemun Shopping Complex
B3	Dongmyo Flea Market
F4	Lotte Duty-Free World-Tower Store
A3	Lotte Mall Gimpo Airport
F4	Lotte Department Store Jamsil Branch
B3	Mangwon Market
E2	Seoul Yangnyeong Market
B3	Seoul Folk Flea Market
D4	Central City
D3	The SHILLA Duty-Free Seoul Store
C3	I'PARK mall / Shilla I'PARK Duty
E3	Cheongdam-dong Luxury Fashion Street
B4	Time Square
C3	Tongin Market
D3	Pyeonghwa Market
B4	Hyundai Department Store D-Cube City
B3	Hyundai Department Store Mokdong Store
E4	Hyundai Department Store Trade Center Store
D3	Hyundai Department Store Apgujeong Main Store

Multi-purpose Cultural Space

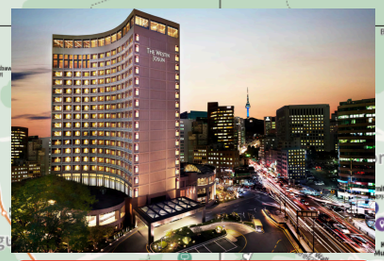
C3	KT&G Sangsang Madang
C4	Nodeulseom Islet
D3	Dongdaemun Design Plaza (DDP)
B3	Oil Tank Culture Park
C3	Culture Station Seoul 284
F4	Seoul Book Bogo
E2	Seoul K-medi Center
D4	Sebitseom Islet
D3	Sewoon Plaza
E3	Under Stand Avenue
D4	Daily teum
E3	CommonGround
E4	COEX
D3	Korea House
E4	Horim Art center

Hotel

C3	1 Grand Mercure Ambassador Hotel and Residences Seoul Yongsan
F3	2 Grand Walkerhill Seoul
E4	3 Grand InterContinental Seoul Parnas
D3	4 Grand Hyatt Seoul
C3	5 Novotel Suites Ambassador Seoul Yongsan
D3	6 Novotel Ambassador Seoul Dongdaemun Hotels & Residences
C3	7 Novotel Ambassador Seoul Yongsan
D3	8 The Plaza Hotel Seoul
D3	9 Lotte Hotel Seoul
F4	10 Lotte Hotel World
A3	11 Mayfield Hotel
D3	12 Mondrian Seoul Itaewon
F3	13 Vista Walkerhill Seoul
C2	14 Swiss Grand Hotel Seoul
F4	15 Signiel Seoul
E3	16 Andaz Seoul Gangnam
D3	17 The Westin Chosun Seoul
E4	18 InterContinental Seoul COEX
E4	19 Josun Palace Seoul Gangnam
C2	20 Conrad Seoul
E4	21 Park Hyatt Seoul
C3	22 Fairmont Ambassador Seoul
D3	23 Four Seasons Hotel Seoul
D3	24 The Shilla Seoul
D3	25 JW Marriott Dongdaemun Square Seoul
D4	26 JW Marriott Hotel Seoul

Performance Hall

A3	LG Arts Center
D4	Kwanglim Arts Center
D4	National Gugak Center
C3	National Theater Company of Korea
D3	National Theater of Korea
D3	Jeongdong Theater
B4	D-Cube Link Art Center
B3	Blue Square
D3	Sejong Center for the Performing Arts
D4	Seoul Arts Center
F3	Universal Arts Center
E4	Korea Cultural House



Korea Univ.

PLDI 2025

ASE 2025

visitseoul.net

My IFIP Talks

- Meeting 67, York Harbor (April 23-27, 2023)
- Meeting 69, Lugano (May 12-16, 2024)

Data-Driven Static Analysis

Hakjoo Oh

25 April 2023 @IFIP WG 2.4 Meeting 67, York Harbor

PL4XGL: A Programming Language Approach to Explainable Graph Learning

Hakjoo Oh
Korea University

(co-work with Minseok Jeon and Jihyeok Park)

IFIP WG 2.4 Meeting @Lugano, Switzerland

My IFIP Talks

- Meeting 67, York Harbor (April 23-27, 2023)
- Meeting 69, Lugano (May 12-16, 2024)

Data-Driven Static Analysis

Hakjoo Oh

25 April 2023 @IFIP WG 2.4 Meeting 67, York Harbor

PL4XGL: A Programming Language Approach to Explainable Graph Learning

Hakjoo Oh
Korea University

(co-work with Minseok Jeon and Jihyeok Park)

IFIP WG 2.4 Meeting @Lugano, Switzerland

Today: automated program repair (APR)

APR Research @Korea Univ.

- Automatic Diagnosis and Correction of Logical Errors for Functional Programming Assignments. OOPSLA 2018
- Context-Aware and Data-Driven Feedback Generation for Programming Assignments. ESEC/FSE 2021
- MemFix: Static Analysis-Based Repair of Memory Deallocation Errors for C. ESEC/FSE 2018
- SAVER: Scalable, Precise, and Safe Memory-Error Repair. ICSE 2020 (deployed in industry)
- NPEX: Repairing Java Null Pointer Exceptions without Tests. ICSE 2022
- PyTER: Effective Program Repair for Python Type Errors. ESEC/FSE 2022
- SmartFix: Fixing Vulnerable Smart Contracts by Accelerating Generate-and-Verify Repair using Statistical Models. ESEC/FSE 2023
- Reducing the Cost of LLM-based APR via Execution-Guided Static Analysis. In progress
- Accurate Detection of Overfitting Patches in Automated Program Repair through Semantic Anti-Patterns. In progress

APR Research @Korea Univ.

- Automatic Diagnosis and Correction of Logical Errors for Functional Programming Assignments. OOPSLA 2018
- Context-Aware and Data-Driven Feedback Generation for Programming Assignments. ESEC/FSE 2021
- MemFix: Static Analysis-Based Repair of Memory Deallocation Errors for C. ESEC/FSE 2018
- SAVER: Scalable, Precise, and Safe Memory-Error Repair. ICSE 2020 (deployed in industry)
- NPEX: Repairing Java Null Pointer Exceptions without Tests. ICSE 2022
- PyTER: Effective Program Repair for Python Type Errors. ESEC/FSE 2022
- SmartFix: Fixing Vulnerable Smart Contracts by Accelerating Generate-and-Verify Repair using Statistical Models. ESEC/FSE 2023
- Reducing the Cost of LLM-based APR via Execution-Guided Static Analysis. In progress
- Accurate Detection of Overfitting Patches in Automated Program Repair through Semantic Anti-Patterns. In progress

APR Research @Korea Univ.

- Automatic Diagnosis and Correction of Logical Errors for Functional Programming Assignments. OOPSLA 2018
- Context-Aware and Data-Driven Feedback Generation for Programming Assignments. ESEC/FSE 2021
- MemFix: Static Analysis-Based Repair of Memory Deallocation Errors for C. ESEC/FSE 2018
- SAVER: Scalable, Precise, and Safe Memory-Error Repair. ICSE 2020 (deployed in industry)
- NPEX: Repairing Java Null Pointer Exceptions without Tests. ICSE 2022
- PyTER: Effective Program Repair for Python Type Errors. ESEC/FSE 2022
- SmartFix: Fixing Vulnerable Smart Contracts by Accelerating Generate-and-Verify Repair using Statistical Models. ESEC/FSE 2023
- Reducing the Cost of LLM-based APR via Execution-Guided Static Analysis. In progress
- Accurate Detection of Overfitting Patches in Automated Program Repair through Semantic Anti-Patterns. In progress

APR Research @Korea Univ.

- Automatic Diagnosis and Correction of Logical Errors for Functional Programming Assignments. OOPSLA 2018
- Context-Aware and Data-Driven Feedback Generation for Programming Assignments. ESEC/FSE 2021
- MemFix: Static Analysis-Based Repair of Memory Deallocation Errors for C. ESEC/FSE 2018
- SAVER: Scalable, Precise, and Safe Memory-Error Repair. ICSE 2020 (deployed in industry)
- NPEX: Repairing Java Null Pointer Exceptions without Tests. ICSE 2022
- PyTER: Effective Program Repair for Python Type Errors. ESEC/FSE 2022
- SmartFix: Fixing Vulnerable Smart Contracts by Accelerating Generate-and-Verify Repair using Statistical Models. ESEC/FSE 2023
- Reducing the Cost of LLM-based APR via Execution-Guided Static Analysis. In progress
- Accurate Detection of Overfitting Patches in Automated Program Repair through Semantic Anti-Patterns. In progress

APR Research @Korea Univ.

- Automatic Diagnosis and Correction of Logical Errors for Functional Programming Assignments. OOPSLA 2018
- Context-Aware and Data-Driven Feedback Generation for Programming Assignments. ESEC/FSE 2021
- MemFix: Static Analysis-Based Repair of Memory Deallocation Errors for C. ESEC/FSE 2018
- SAVER: Scalable, Precise, and Safe Memory-Error Repair. ICSE 2020 (deployed in industry)
- NPEX: Repairing Java Null Pointer Exceptions without Tests. ICSE 2022
- PyTER: Effective Program Repair for Python Type Errors. ESEC/FSE 2022
- SmartFix: Fixing Vulnerable Smart Contracts by Accelerating Generate-and-Verify Repair using Statistical Models. ESEC/FSE 2023
- Reducing the Cost of LLM-based APR via Execution-Guided Static Analysis. In progress
- Accurate Detection of Overfitting Patches in Automated Program Repair through Semantic Anti-Patterns. In progress

APR Research @Korea Univ.

- Automatic Diagnosis and Correction of Logical Errors for Functional Programming Assignments. OOPSLA 2018
- Context-Aware and Data-Driven Feedback Generation for Programming Assignments. ESEC/FSE 2021
- MemFix: Static Analysis-Based Repair of Memory Deallocation Errors for C. ESEC/FSE 2018
- SAVER: Scalable, Precise, and Safe Memory-Error Repair. ICSE 2020 (deployed in industry)
- NPEX: Repairing Java Null Pointer Exceptions without Tests. ICSE 2022
- PyTER: Effective Program Repair for Python Type Errors. ESEC/FSE 2022
- SmartFix: Fixing Vulnerable Smart Contracts by Accelerating Generate-and-Verify Repair using Statistical Models. ESEC/FSE 2023
- Reducing the Cost of LLM-based APR via Execution-Guided Static Analysis. In progress
- Accurate Detection of Overfitting Patches in Automated Program Repair through Semantic Anti-Patterns. In progress

APR Research @Korea Univ.

- Automatic Diagnosis and Correction of Logical Errors for Functional Programming Assignments. OOPSLA 2018
- Context-Aware and Data-Driven Feedback Generation for Programming Assignments. ESEC/FSE 2021
- MemFix: Static Analysis-Based Repair of Memory Deallocation Errors for C. ESEC/FSE 2018
- SAVER: Scalable, Precise, and Safe Memory-Error Repair. ICSE 2020 (deployed in industry)
- NPEX: Repairing Java Null Pointer Exceptions without Tests. ICSE 2022
- PyTER: Effective Program Repair for Python Type Errors. ESEC/FSE 2022
- SmartFix: Fixing Vulnerable Smart Contracts by Accelerating Generate-and-Verify Repair using Statistical Models. ESEC/FSE 2023
- Reducing the Cost of LLM-based APR via Execution-Guided Static Analysis. In progress
- Accurate Detection of Overfitting Patches in Automated Program Repair through Semantic Anti-Patterns. In progress

Our approach: Static analysis-based program repair

APR Research @Korea Univ.

- Automatic Diagnosis and Correction of Logical Errors for Functional Programming Assignments. OOPSLA 2018
- Context-Aware and Data-Driven Feedback Generation for Programming Assignments. ESEC/FSE 2021
- MemFix: Static Analysis-Based Repair of Memory Deallocation Errors for C. ESEC/FSE 2018
- SAVER: Scalable, Precise, and Safe Memory-Error Repair. ICSE 2020 (deployed in industry)
- NPEX: Repairing Java Null Pointer Exceptions without Tests. ICSE 2022
- PyTER: Effective Program Repair for Python Type Errors. ESEC/FSE 2022
- SmartFix: Fixing Vulnerable Smart Contracts by Accelerating Generate-and-Verify Repair using Statistical Models. ESEC/FSE 2023
- Reducing the Cost of LLM-based APR via Execution-Guided Static Analysis. In progress
- Accurate Detection of Overfitting Patches in Automated Program Repair through Semantic Anti-Patterns. In progress

Our approach: Static analysis-based program repair

Memory Errors in C Programs

- Memory-leak (**ML**), use-after-free (**UAF**), and double-free (**DF**) are prevalent in real-world C programs

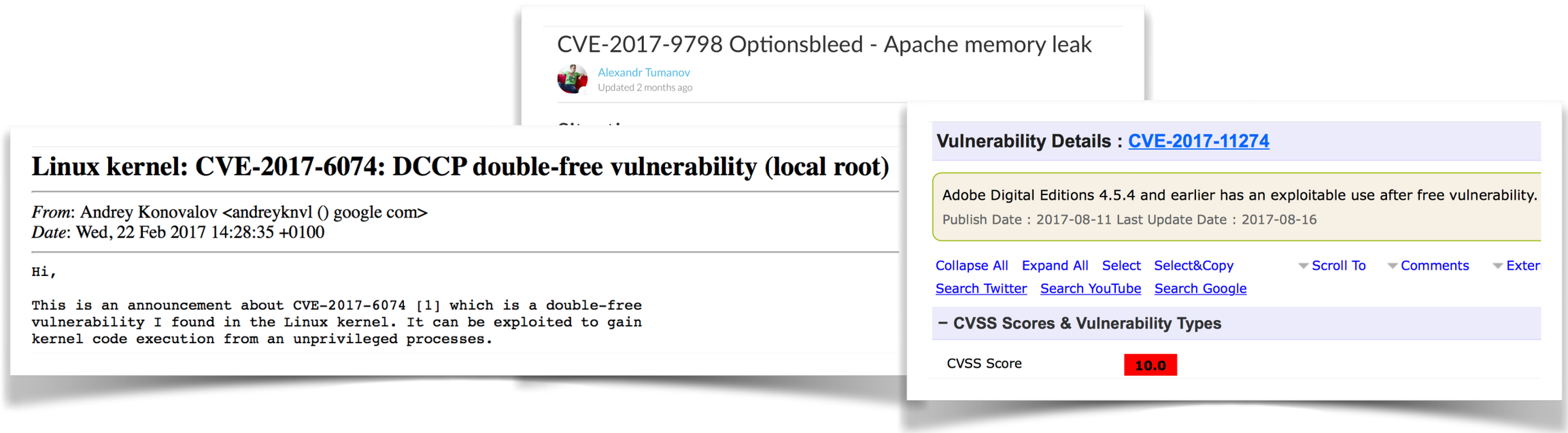
Repository	#commits	ML	DF	UAF	Total	*-overflow
linux	721,119	3,740	821	1,986	**6,363**	5,092
openssl	21,009	220	36	12	**264**	61
numpy	17,008	58	2	2	**59**	53
php	105,613	1,129	148	197	**1,449**	649
git	49,475	350	19	95	**442**	258

Memory Errors in C Programs

- Memory-leak (**ML**), use-after-free (**UAF**), and double-free (**DF**) are prevalent in real-world C programs

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

- Significant sources of security vulnerabilities



Goal

- Long-term goal: Fully automated detection and repair
- This talk: SAVER, a system to automatically fix memory errors

- To be practical, SAVER is designed to be
  - **Scalable**: Capable of handling large, industry programs
  - **Precise**: Effectively fixes diverse bugs with high fix rates
  - **Safe**: Generated patches are safe, not introducing new errors

(1) Double-Free in 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;
```

(1) Double-Free in Linux Kernel

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

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

  goto err;
}
```

Allocated

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

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

(1) Double-Free in 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;
```

Allocated

Freed

(1) Double-Free in 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;
```

Allocated

Freed

Double Free

(1) Double-Free in 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;
```

Allocated

Freed

Double Free

(1) Double-Free in 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;
```

Allocated

Freed

Double Free

Double Free

(1) Double-Free in 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;
```

(1) Double-Free in 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

Challenge 1: Difficult to ensure that bugs are fixed correctly

```
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;
```

(1) Double-Free in Linux Kernel

Second attempt (9 months later)

```
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;
```

(1) Double-Free in Linux Kernel

```
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;
```

memory leak

Challenge 2: Patches may introduce new errors

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.

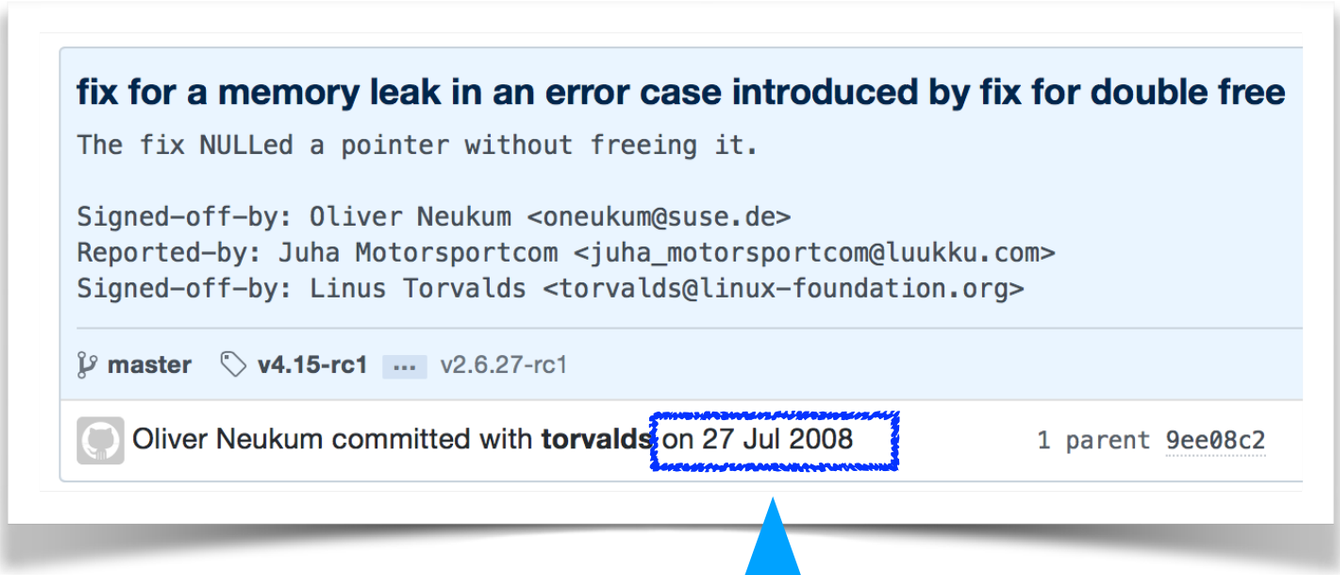
Signed-off-by: Oliver Neukum <oneukum@suse.de>
Signed-off-by: Greg Kroah-Hartman <gregkh@suse.de>

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

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

Second attempt (9 months later)

(1) Double-Free in Linux Kernel

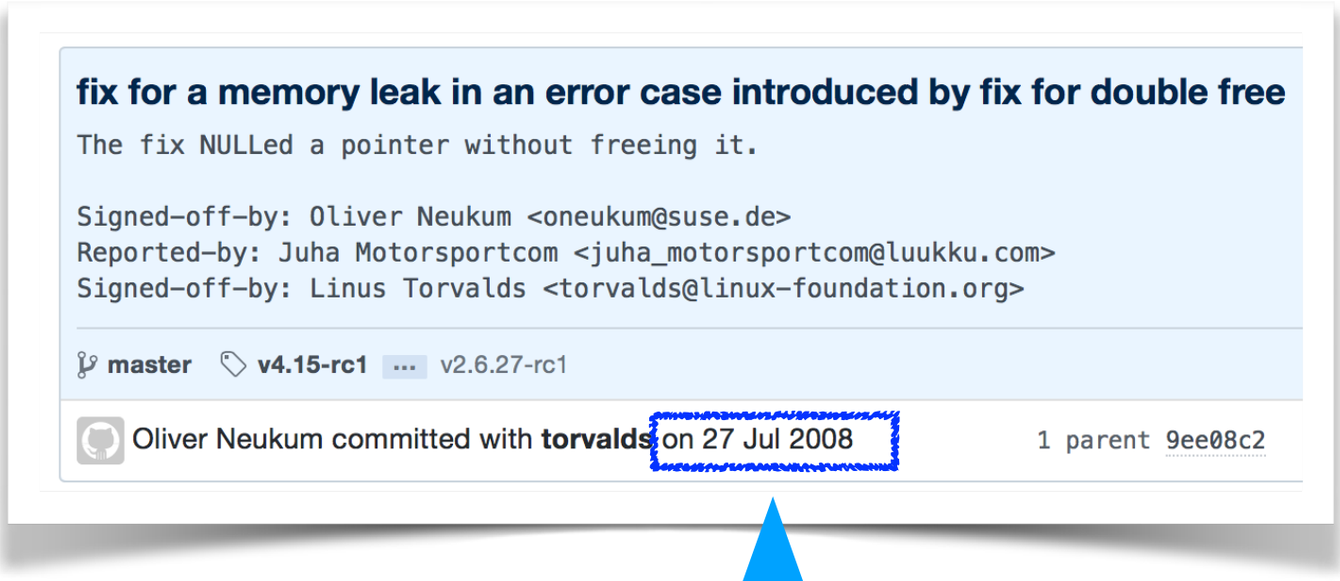


Third attempt
(10 months after bug detection)

```
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;
```

(1) Double-Free in Linux Kernel

Challenge 3: The resulting patches are often not of high quality



Third attempt
(10 months after bug detection)

```
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;
```

SAVER-Generated Patch

```
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;
```

✓Fast (few mins)

✓Safety guarantee

```
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;
```

(2) Memory Leak in Snort

```
1  int append_data (Node *node, int *ndata) {
2      if (!(Node *n = malloc(sizeof(Node)))
3          return -1; // failed to be appended
4      n->data = ndata;
5      n->next = node->next; node->next = n;
6      return 0; // successfully appended
7  }
8
9  Node *lx = ... // a linked list
10 Node *ly = ... // a linked list
11 for (Node *node = lx; node != NULL; node = node->next) {
12     int *dptr = malloc(sizeof(int));
13     if (!dptr) return;
14     *dptr = *(node->data);
15 (-) append_data(ly, dptr); // potential memory-leak
16
17 }
```

Infer

Memory Leak:
An object allocated at line 12
becomes unreachable after line 15

(2) Memory Leak in Snort

```
1  int append_data (Node *node, int *ndata) {
2      if (!(Node *n = malloc(sizeof(Node)))
3          return -1; // failed to be appended
4      n->data = ndata;
5      n->next = node->next; node->next = n;
6      return 0; // successfully appended
7  }
8
9  Node *lx = ... // a linked list
10 Node *ly = ... // a linked list
11 for (Node *node = lx; node != NULL; node = node->next) {
12     int *dptr = malloc(sizeof(int));
13     if (!dptr) return;
14     *dptr = *(node->data);
15 (-) append_data(ly, dptr); // potential memory-leak
16
17 }
```

Normal execution

Infer

```
Memory Leak:
An object allocated at line 12
becomes unreachable after line 15
```

(2) Memory Leak in Snort

```
1  int append_data (Node *node, int *ndata) {
2      if (!(Node *n = malloc(sizeof(Node))))
3          return -1; // failed to be appended
4      n->data = ndata;
5      n->next = node->next; node->next = n;
6      return 0; // successfully appended
7  }
8
9  Node *lx = ... // a linked list
10 Node *ly = ... // a linked list
11 for (Node *node = lx; node != NULL; node = node->next) {
12     int *dptr = malloc(sizeof(int));
13     if (!dptr) return;
14     *dptr = *(node->data);
15 (-) append_data(ly, dptr); // potential memory-leak
16
17 }
```

Buggy execution

Infer

Memory Leak:
An object allocated at line 12
becomes unreachable after line 15

(2) Memory Leak in Snort

```
1  int append_data (Node *node, int *ndata) {
2      if (!(Node *n = malloc(sizeof(Node)))
3          return -1; // failed to be appended
4      n->data = ndata;
5      n->next = node->next; node->next = n;
6      return 0; // successfully appended
7  }
8
9  Node *lx = ... // a linked list
10 Node *ly = ... // a linked list
11 for (Node *node = lx; node != NULL; node = node->next) {
12     int *dptr = malloc(sizeof(int));
13     if (!dptr) return;
14     *dptr = *(node->data);
15 (-) append_data(ly, dptr); // potential memory-leak
16 (+) if ((append_data(ly, dptr)) == -1) free(dptr);
17 }
```

Infer

Memory Leak:
An object allocated at line 12
becomes unreachable after line 15

cf) SAVER vs. Generative AI

- LLMs do not guarantee safety, e.g., GPT4-generated patch:

```
1  int append_data (Node *node, int *ndata) {
2      if (!(Node *n = malloc(sizeof(Node)))
3          return -1; // failed to be appended
4      n->data = ndata;
5      n->next = node->next; node->next = n;
6      return 0; // successfully appended
7  }
8
9  Node *lx = ... // a linked list
10 Node *ly = ... // a linked list
11 for (Node *node = lx; node != NULL; node = node->next) {
12     int *dptr = malloc(sizeof(int));
13     if (!dptr) return;
14     *dptr = *(node->data);
15 (-) append_data(ly, dptr); // potential memory-leak
16 (+) append_data(ly, dptr); free(dptr);
17 }
```

UAF or DF introduced

cf) SAVER vs. Generative AI

- LLMs do not guarantee safety, e.g., GPT4-generated patch:

```
1  int append_data (Node *node, int *ndata) {
2      if (!(Node *n = malloc(sizeof(Node)))
3          return -1; // failed to be appended
4      n->data = ndata;
5      n->next = node->next; node->next = n;
6      return 0; // successfully appended
7  }
8
9  Node *lx = ... // a linked list
10 Node *ly = ... // a linked list
11 for (Node *node = lx; node != NULL; node = node->next) {
12     int *dptr = malloc(sizeof(int));
13     if (!dptr) return;
14     *dptr = *(node->data);
15 (-) append_data(ly, dptr); // potential memory-leak
16 (+) append_data(ly, dptr); free(dptr);
17 }
```

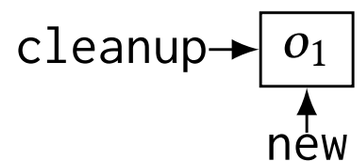
UAF or DF introduced

(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  }
```

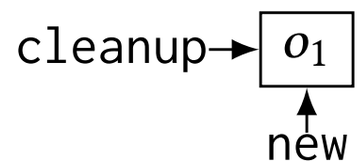
(3) Use-After-Free in Binutils

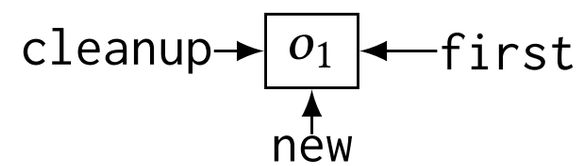
```
1  struct node *cleanup; // list of objects to be deallocated
2  struct node *first = NULL;
3  for (...) {
4      struct node *new = xmalloc(sizeof(*new));
5      make_cleanup(new); // add new to the cleanup list
6      new->name = ...;
7      ...
8      if (...) {
9          first = new;
10
11         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 }
```



(3) Use-After-Free in Binutils

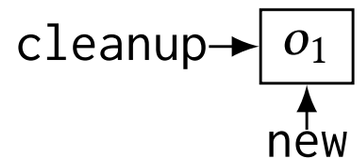
```
1   struct node *cleanup; // list of objects to be deallocated
2   struct node *first = NULL;
3   for (...) {
4       struct node *new = xmalloc(sizeof(*new));
5       make_cleanup(new); // add new to the cleanup list
6       new->name = ...;
7       ...                          true
8       if (...) {
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  }
```

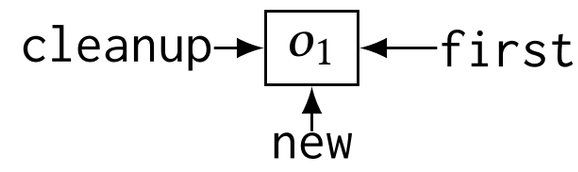


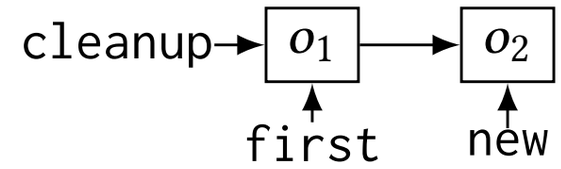


(3) Use-After-Free in Binutils

```
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 }
```

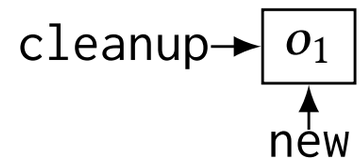


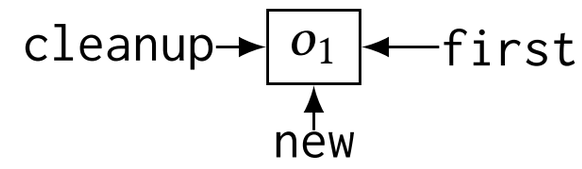


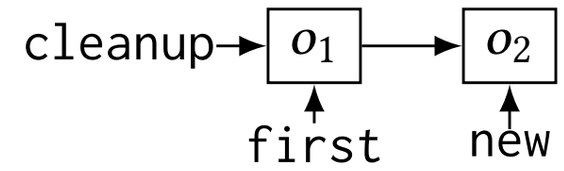


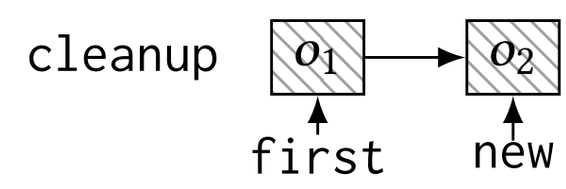
(3) Use-After-Free in Binutils

```
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      ...                                    false
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 }
```



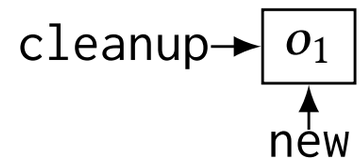


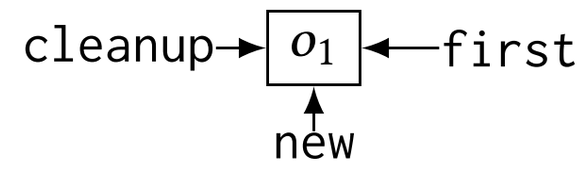


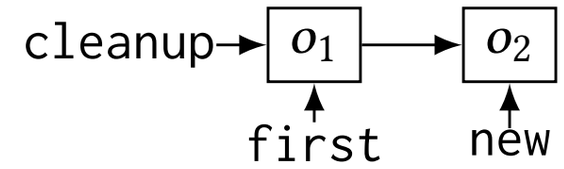


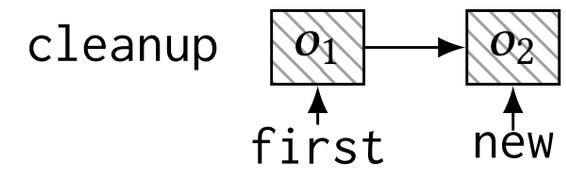
(3) Use-After-Free in Binutils

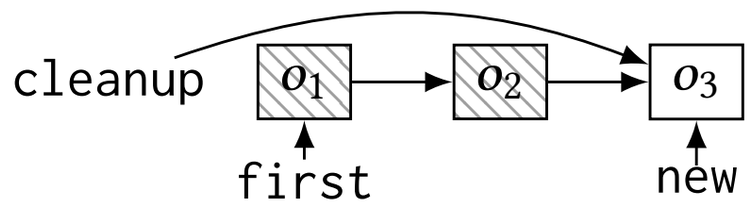
```
1  struct node *cleanup; // list of objects to be deallocated
2  struct node *first = NULL;
3  for (...) {
4      struct node *new = xmalloc(sizeof(*new));
5      make_cleanup(new); // add new to the cleanup list
6      new->name = ...;
7      ...
8      if (...) {
9          first = new;
10
11         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 }
```





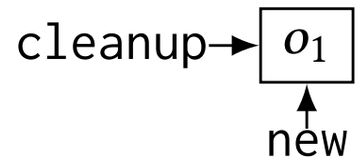


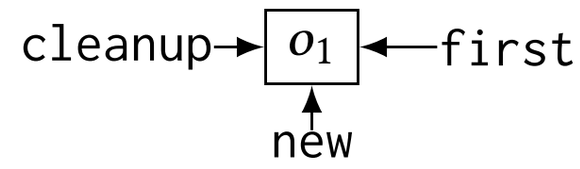


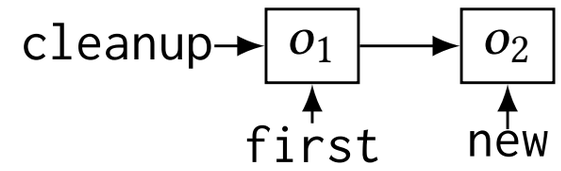


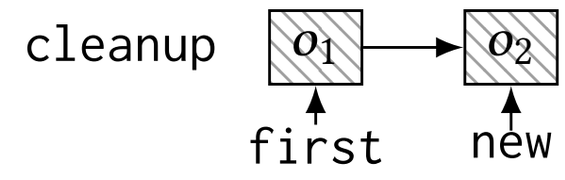
(3) Use-After-Free in Binutils

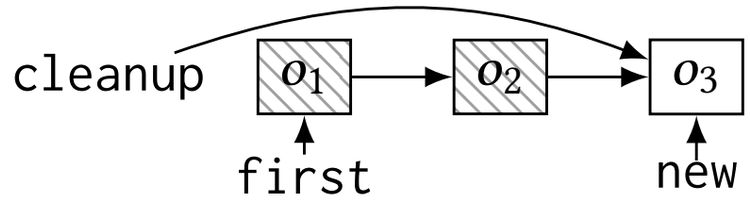
```
1  struct node *cleanup; // list of objects to be deallocated
2  struct node *first = NULL;
3  for (...) {
4      struct node *new = xmalloc(sizeof(*new));
5      make_cleanup(new); // add new to the cleanup list
6      new->name = ...;
7      ...                                    false
8      if (...) {
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 }
```





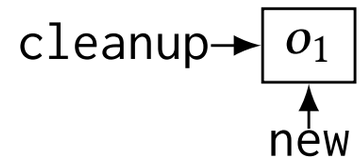


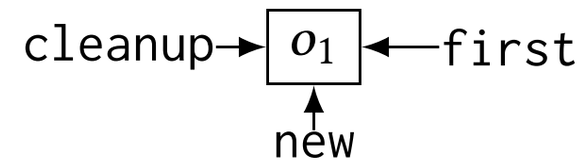


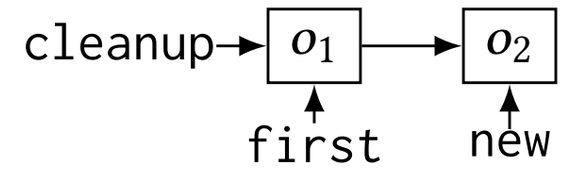


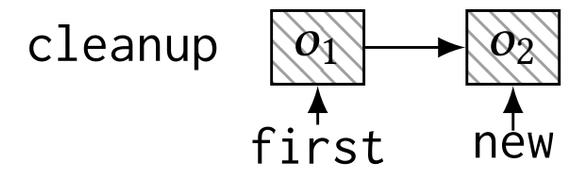
(3) Use-After-Free in Binutils

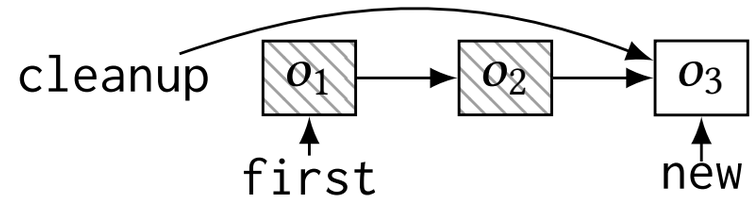
```
1  struct node *cleanup; // list of objects to be deallocated
2  struct node *first = NULL;
3  for (...) {
4      struct node *new = xmalloc(sizeof(*new));
5      make_cleanup(new); // add new to the cleanup list
6      new->name = ...;
7      ...                                  false
8      if (...) {
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;                        use-after-free
17   do_cleanups(); // deallocate all objects in cleanup
18 }
```











SAVER-Generated Patch

```
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  (+)     tmp = first->name;
11          continue;
12      }
13    /* potential use-after-free: `first->name` */
14  (-)   if (first == NULL || new->name != first->name)
15  (+)   if (first == NULL || new->name != tmp)
16          continue;
17    do_cleanups(); // deallocate all objects in cleanup
18  }
```

How SAVER Works

```
1  p = malloc(1); //o1
2  if (C)
3    q = p;
4  else
5    q = malloc(1); //o2
6  *p = 1;
7  free(q);
```

Memory leak: o_1 is not freed when the false branch is taken

How SAVER Works

```
1  p = malloc(1); //o1
2  if (C)
3    q = p;
4  else
5    q = malloc(1); //o2
6  *p = 1;
7  free(q);
```

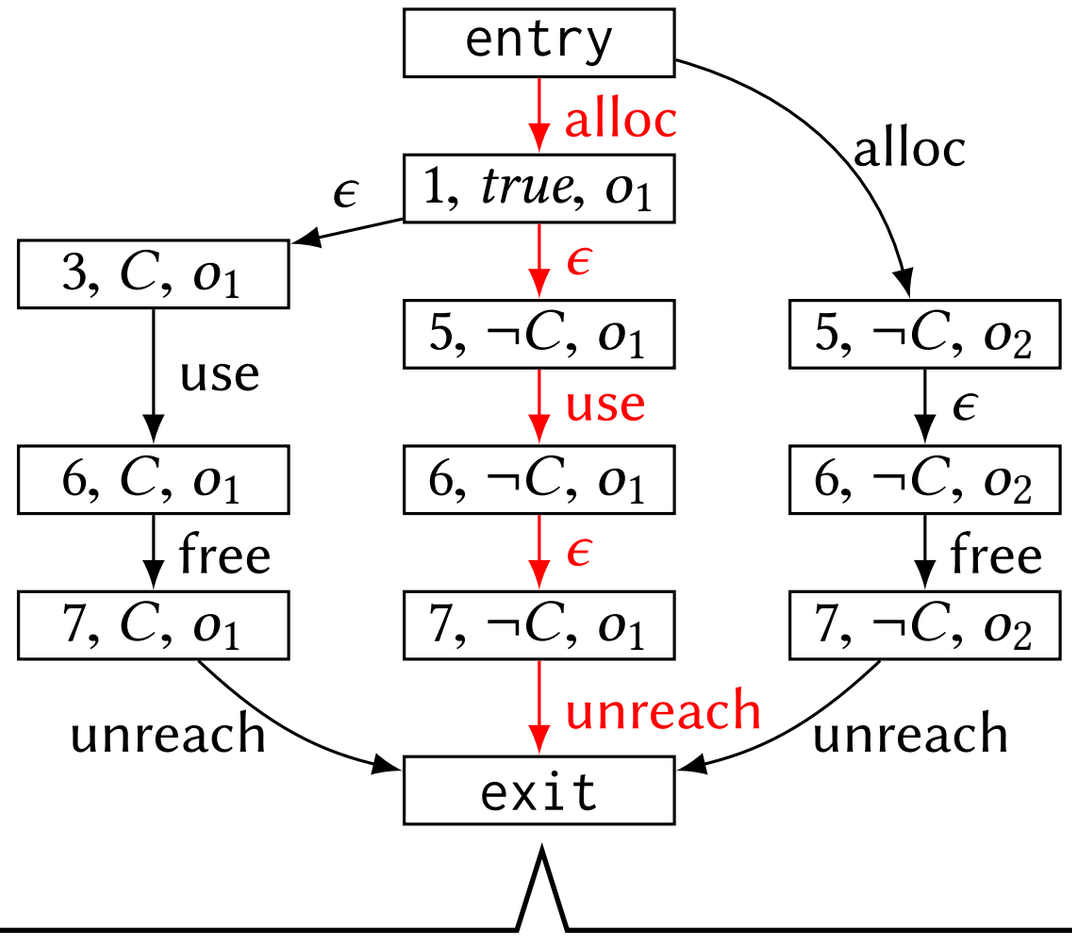
if (!C) free(p);

Memory leak: o_1 is not freed when the false branch is taken

How SAVER Works

1. Run a static analysis to generate *object flow graph*

```
1  p = malloc(1); //o1
2  if (C)
3    q = p;
4  else
5    q = malloc(1); //o2
6  *p = 1;
7  free(q);
```

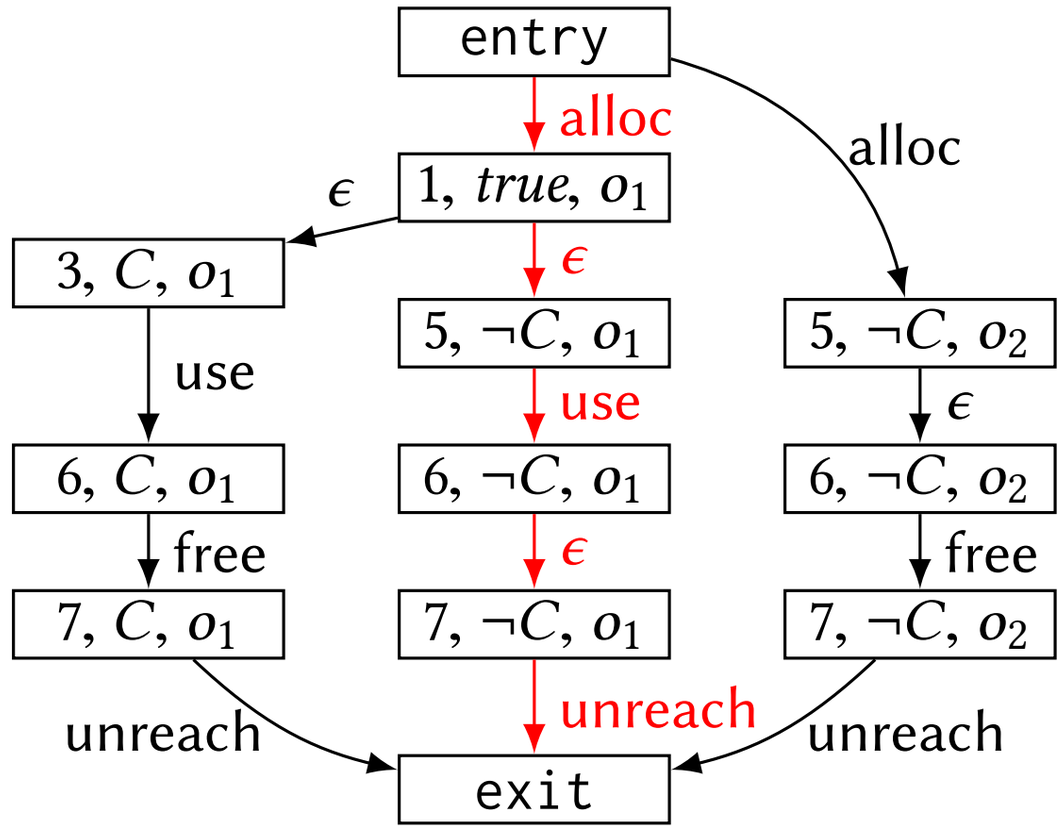


- Vertex: (program point, path condition, available heap object)
- Edge: control flow labeled w/ events that could occur for objects

How SAVER Works

2. Relabel object flow graph to eliminate buggy paths

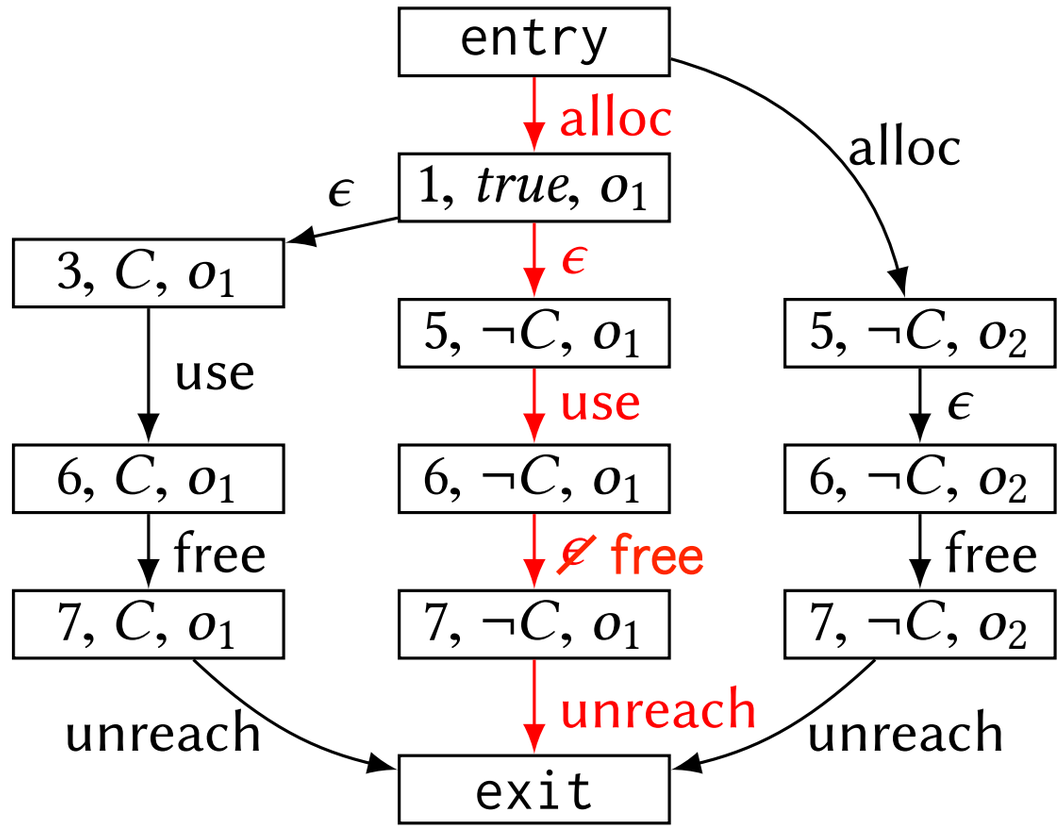
```
1  p = malloc(1); //o1
2  if (C)
3    q = p;
4  else
5    q = malloc(1); //o2
6  *p = 1;
7  free(q);
```



How SAVER Works

2. Relabel object flow graph to eliminate buggy paths

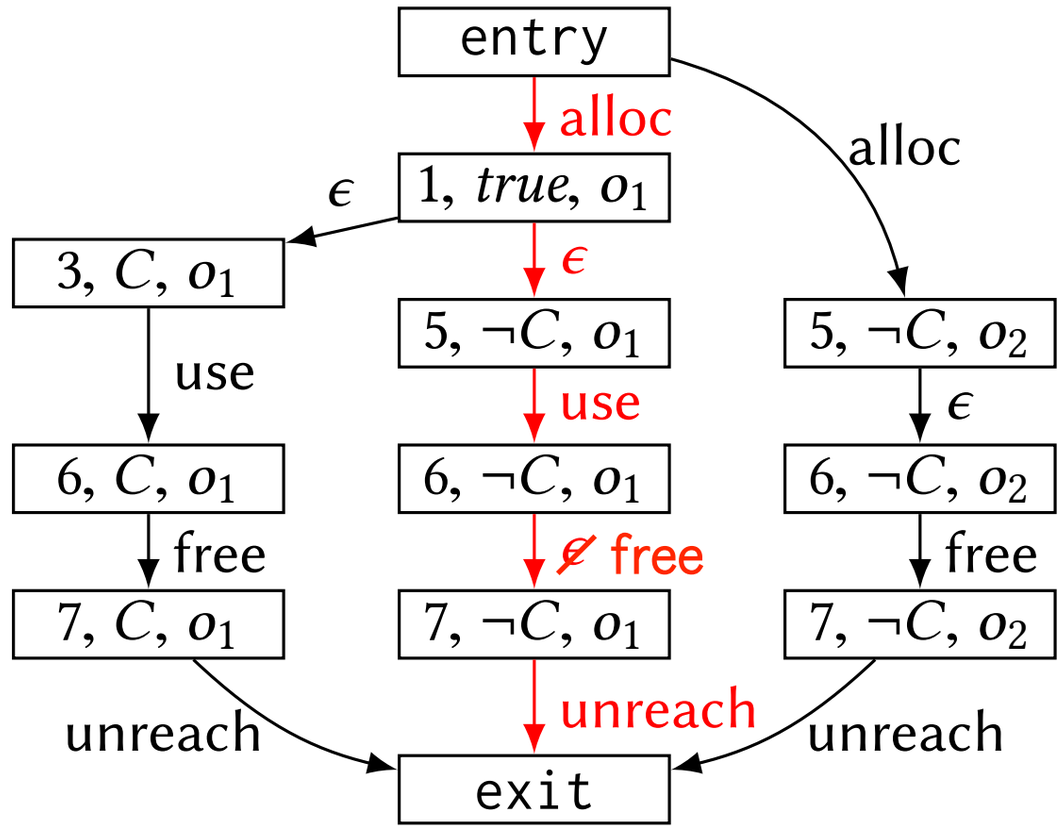
```
1  p = malloc(1); //o1
2  if (C)
3    q = p;
4  else
5    q = malloc(1); //o2
6  *p = 1;
7  free(q);
```



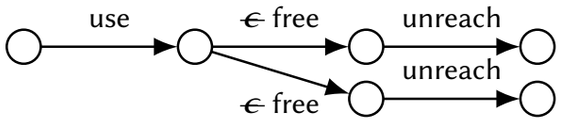
How SAVER Works

2. Relabel object flow graph to eliminate buggy paths

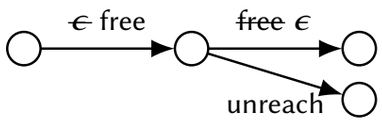
```
1  p = malloc(1); //o1
2  if (C)
3    q = p;
4  else
5    q = malloc(1); //o2
6  *p = 1;
7  free(q);
```



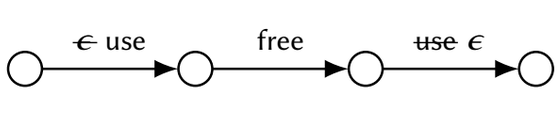
- SAVER supports four types of re-labeling strategies:



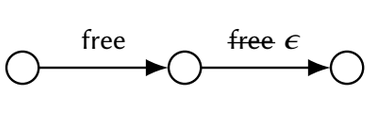
(a) Inserting free



(b) Relocating free



(c) Relocating use (dereference)



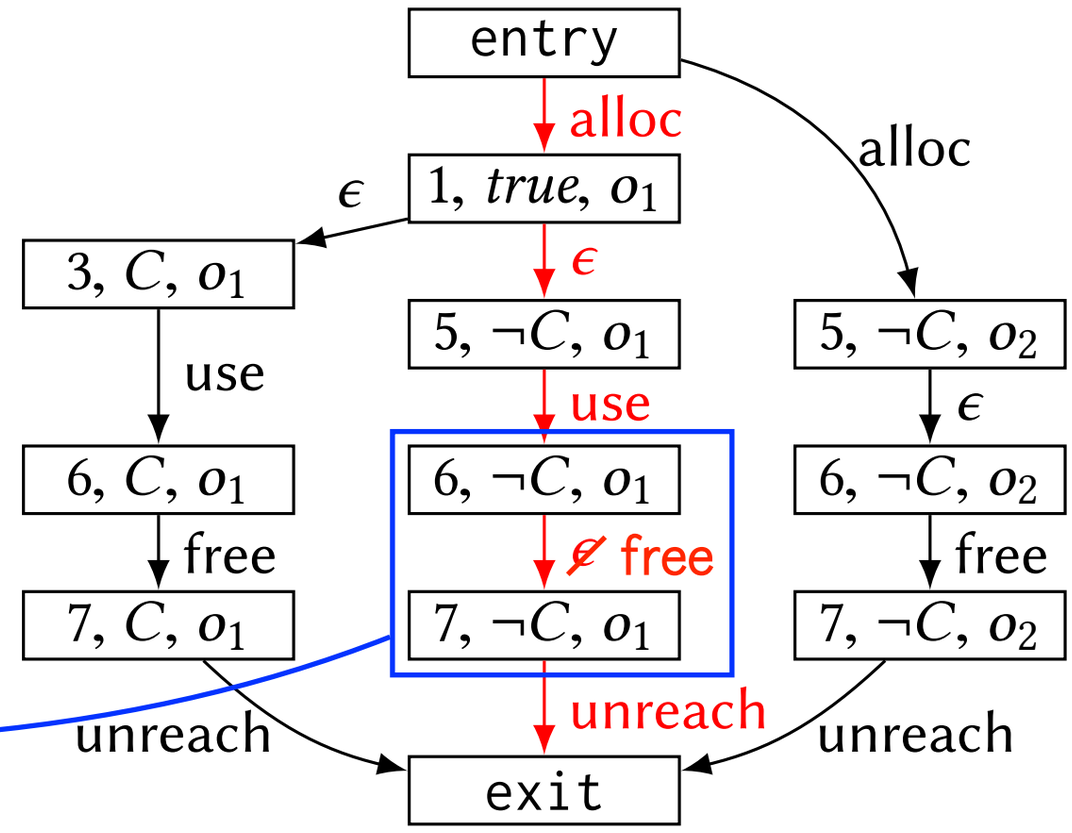
(d) Deleting free

How SAVER Works

3. Generate a patch from the re-labeled edge

```
1  p = malloc(1); //o1
2  if (C)
3    q = p;
4  else
5    q = malloc(1); //o2
6  *p = 1;
7  free(q);
```

if (!C) free(p);



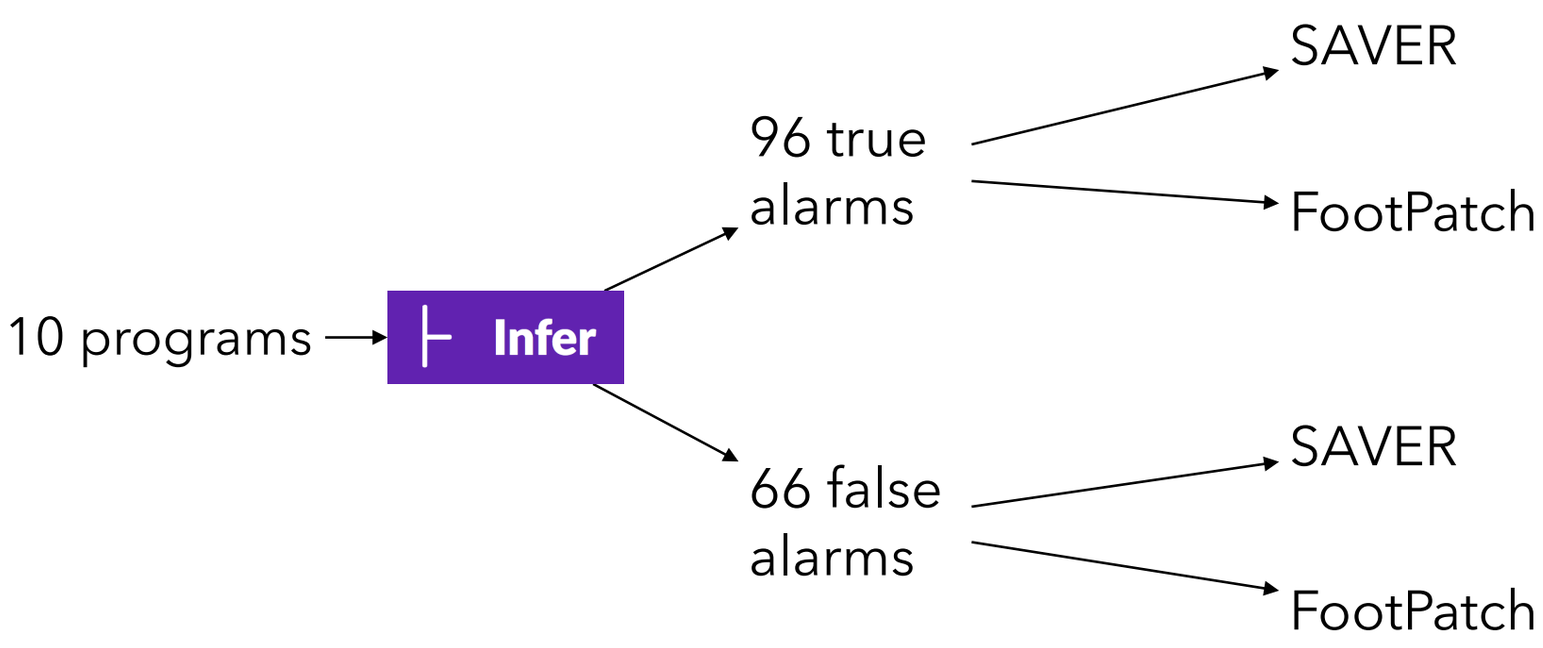
For Deployment

- Improved scalability (in the paper)
  - Selective path sensitivity
  - Program slicing
- Improved usability (not in the paper)
  - Build failures
  - Robust translation from IR-level patches to source-level

Effectiveness

Existing memory error repair tool [ICSE 2018]

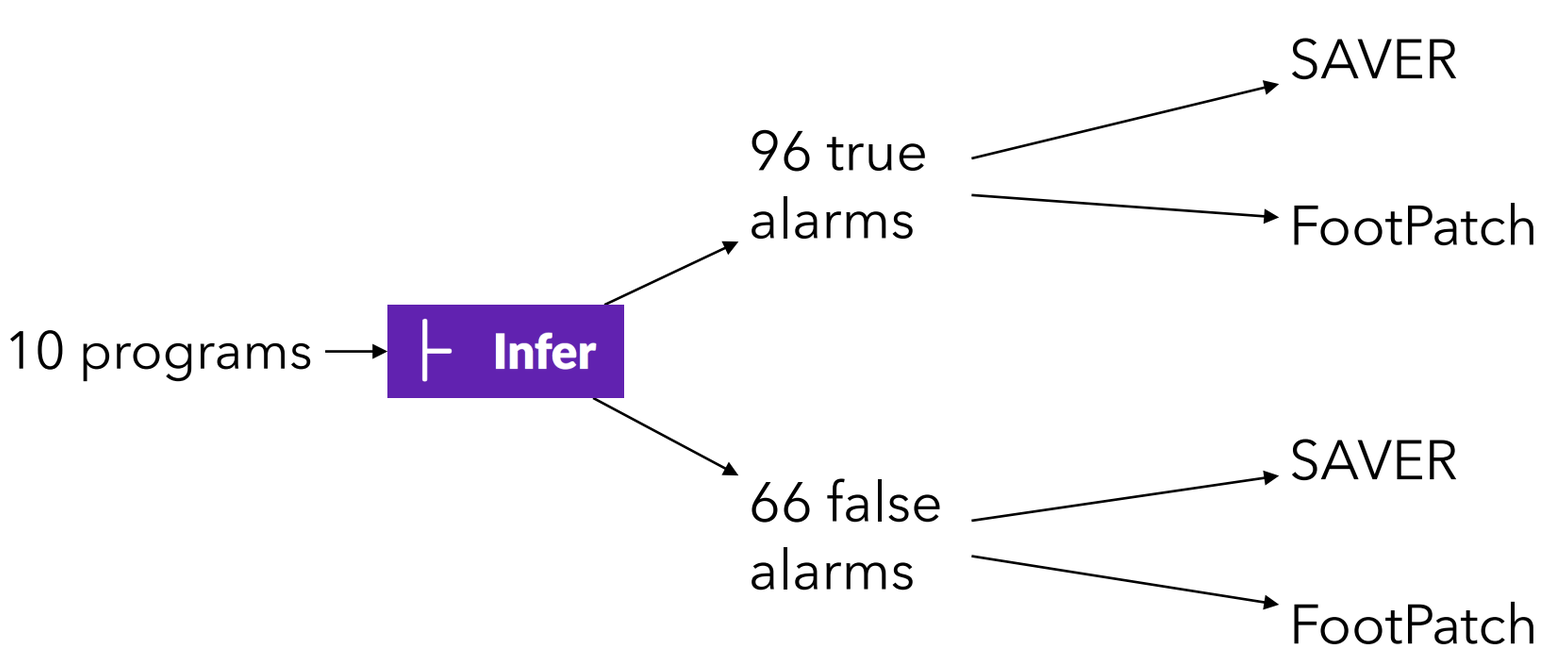
		Infer				SAVER						FootPatch [60]						
Program	**kLoC**	**#T**	**#F**	**Pre(s)**	**Fix(s)**	G_T	$✓_T$	$△_T$	$✗_T$	G_F	$✗_F$	**Fix(s)**	G_T	$✓_T$	$△_T$	$✗_T$	G_F	$✗_F$
rappel (ad8efd7)	2.2	1	0	2.2	0.0	1	1	0	0	0	0	8.9	1	1	0	0	0	0
flex (d3de49f)	22.3	3	4	26.3	2.5	0	0	0	0	0	0	51.0	0	0	0	0	1	1
WavPack (22977b2)	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



Effectiveness

Existing memory error repair tool [ICSE 2018]

		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



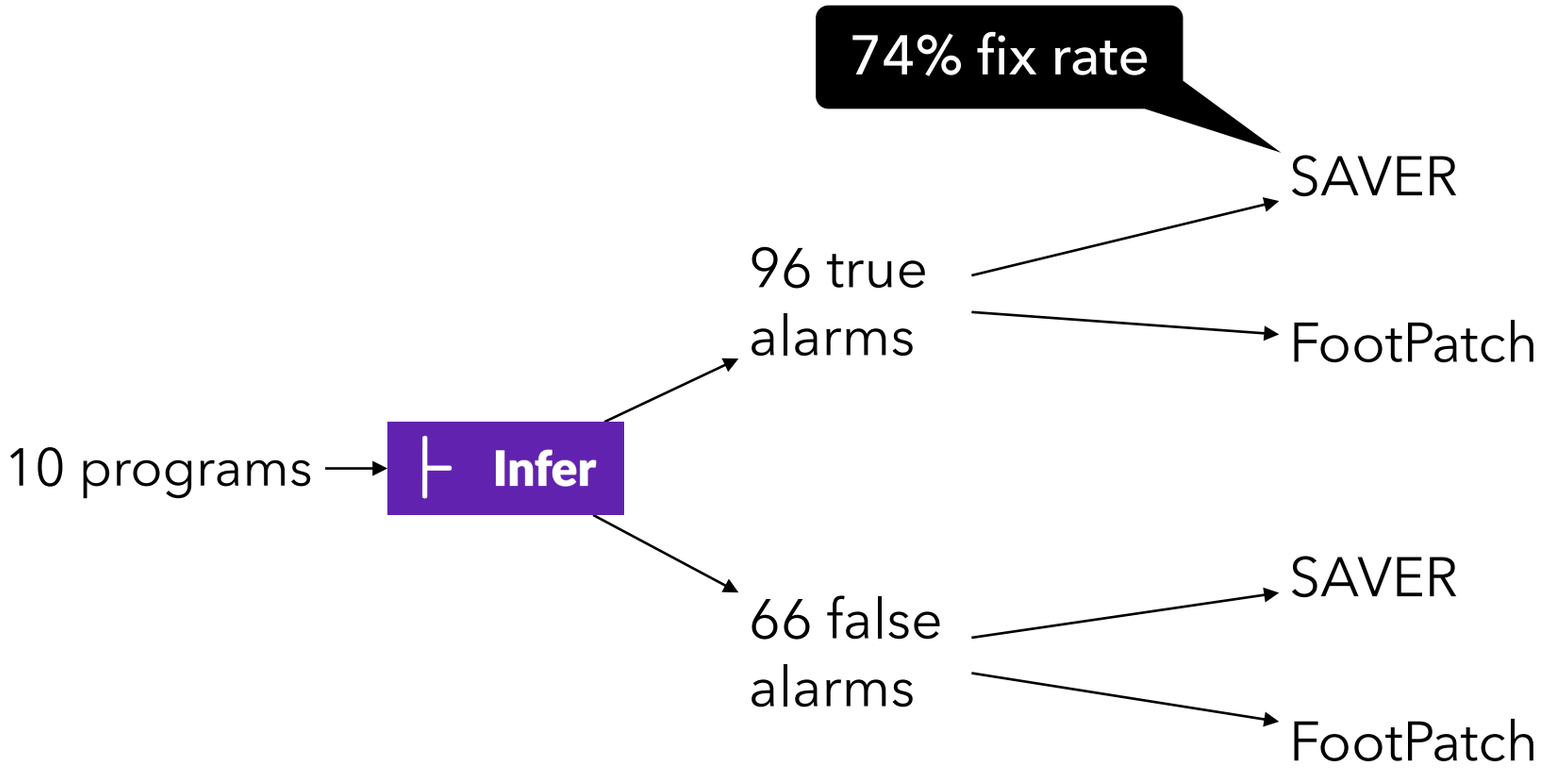
Generated 72 patches

(Fixed: 71, Safe: 1, Unsafe: 0)

Effectiveness

Existing memory error repair tool [ICSE 2018]

		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



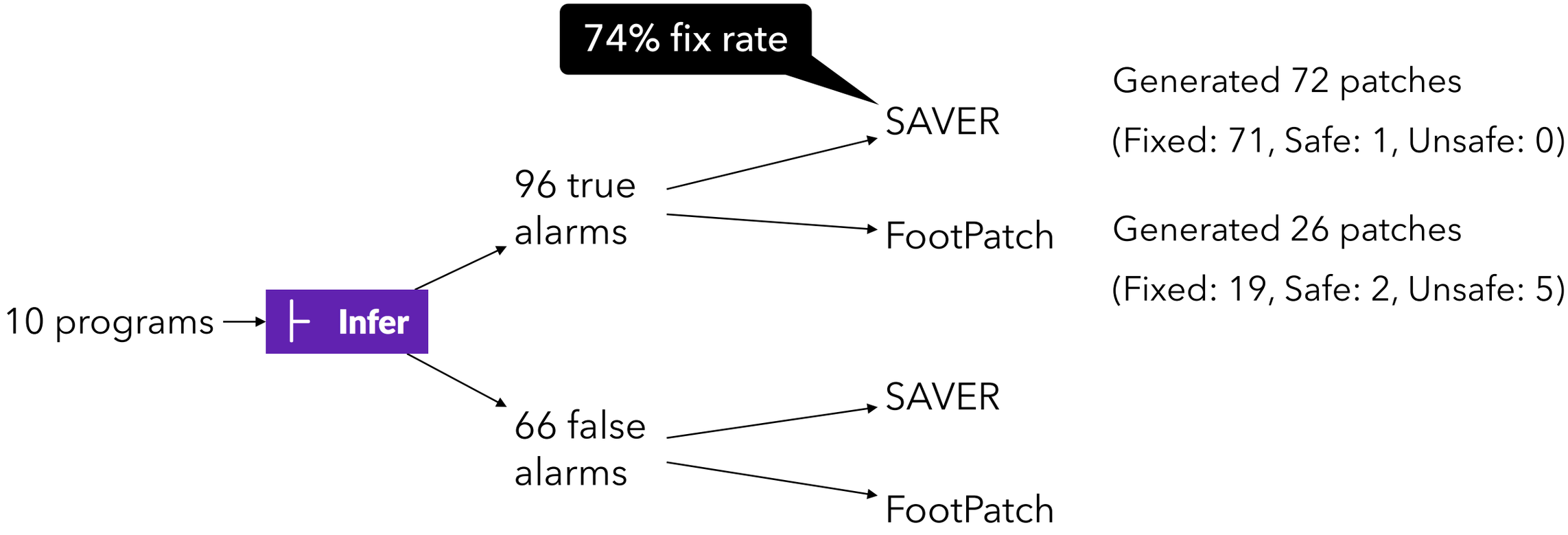
Generated 72 patches

(Fixed: 71, Safe: 1, Unsafe: 0)

Effectiveness

Existing memory error repair tool [ICSE 2018]

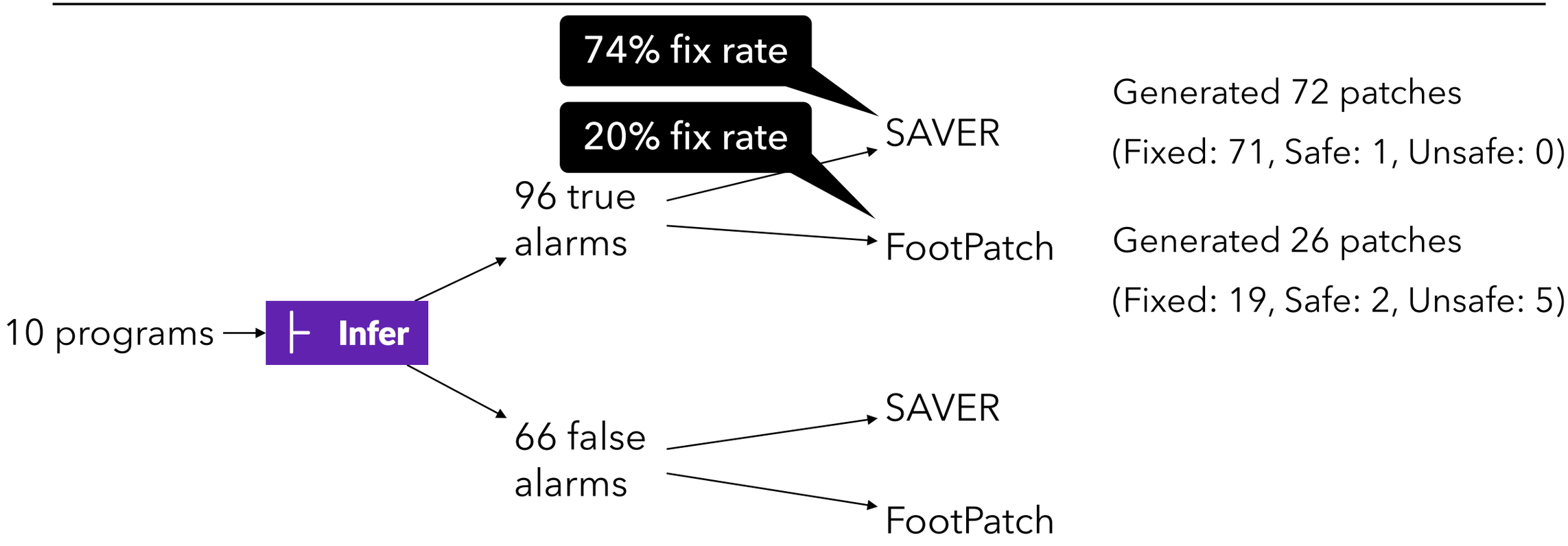
		Infer				SAVER						FootPatch [60]						
Program	**kLoC**	**#T**	**#F**	**Pre(s)**	**Fix(s)**	G_T	✓$_T$	△$_T$	✗$_T$	G_F	✗$_F$	**Fix(s)**	G_T	✓$_T$	△$_T$	✗$_T$	G_F	✗$_F$
rappel (ad8efd7)	2.2	1	0	2.2	0.0	1	1	0	0	0	0	8.9	1	1	0	0	0	0
flex (d3de49f)	22.3	3	4	26.3	2.5	0	0	0	0	0	0	51.0	0	0	0	0	1	1
WavPack (22977b2)	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



Effectiveness

Existing memory error repair tool [ICSE 2018]

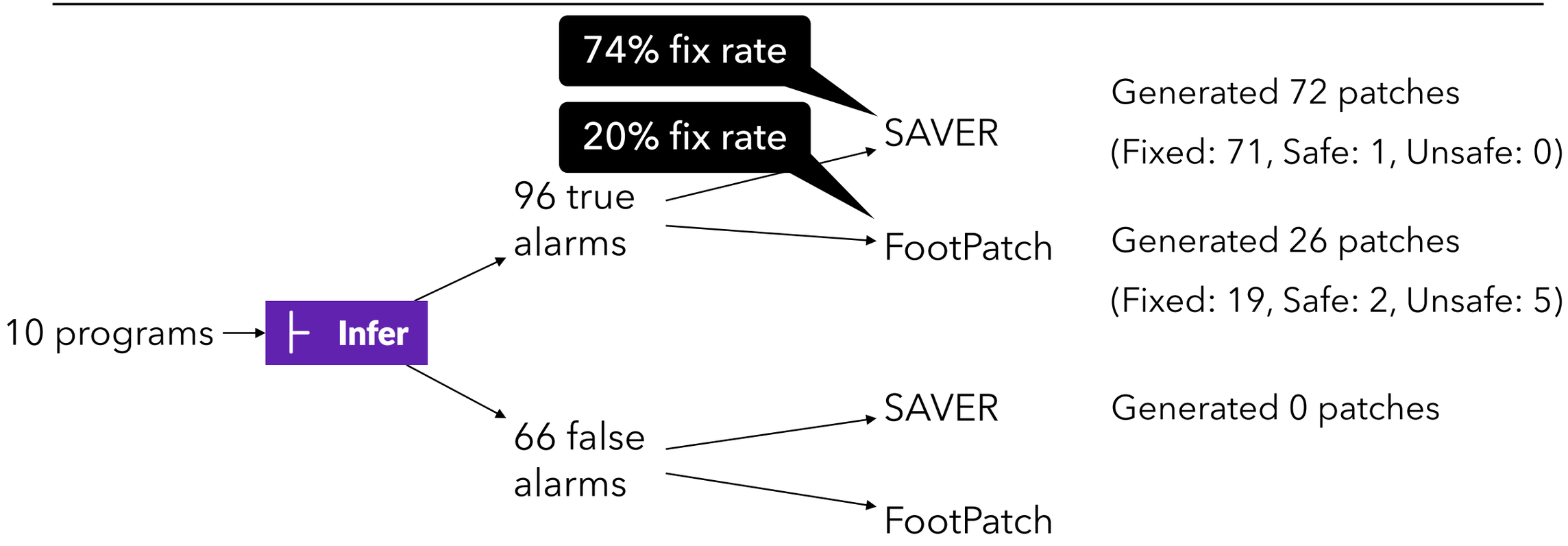
		Infer				SAVER						FootPatch [60]						
Program	**kLoC**	**#T**	**#F**	**Pre(s)**	**Fix(s)**	G_T	✓$_T$	△$_T$	✗$_T$	G_F	✗$_F$	**Fix(s)**	G_T	✓$_T$	△$_T$	✗$_T$	G_F	✗$_F$
rappel (ad8efd7)	2.2	1	0	2.2	0.0	1	1	0	0	0	0	8.9	1	1	0	0	0	0
flex (d3de49f)	22.3	3	4	26.3	2.5	0	0	0	0	0	0	51.0	0	0	0	0	1	1
WavPack (22977b2)	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



Effectiveness

Existing memory error repair tool [ICSE 2018]

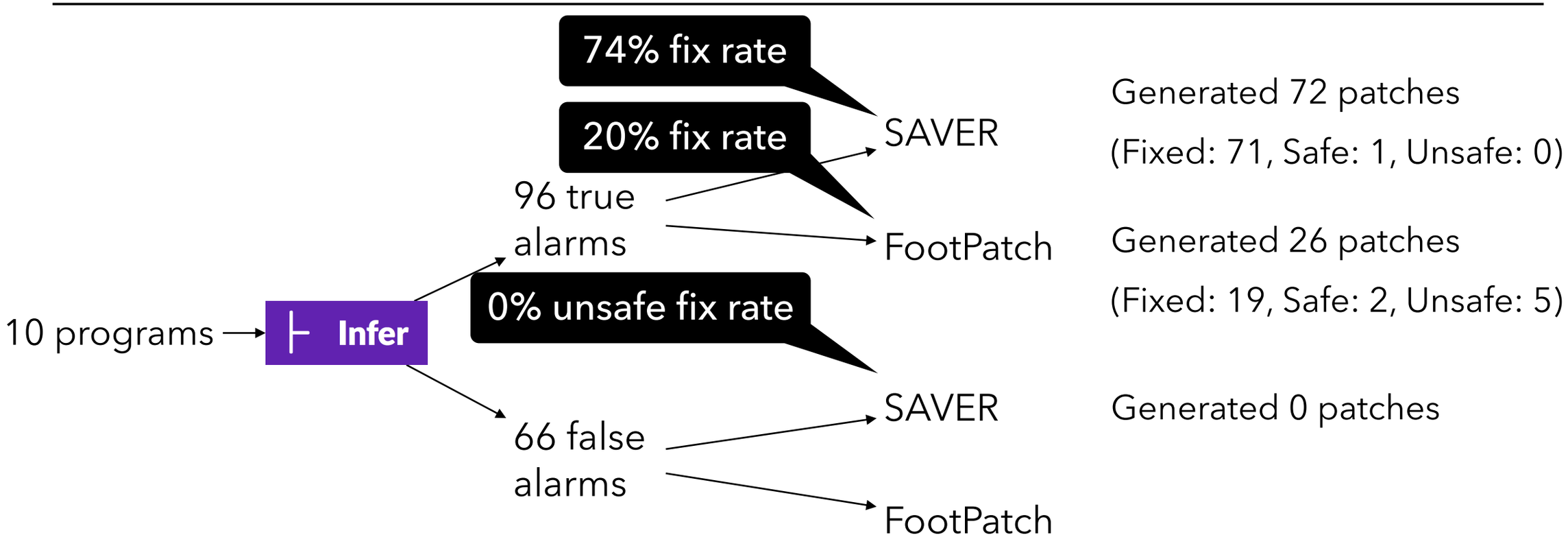
		Infer				SAVER						FootPatch [60]						
Program	**kLoC**	**#T**	**#F**	**Pre(s)**	**Fix(s)**	G_T	✓$_T$	△$_T$	✗$_T$	G_F	✗$_F$	**Fix(s)**	G_T	✓$_T$	△$_T$	✗$_T$	G_F	✗$_F$
rappel (ad8efd7)	2.2	1	0	2.2	0.0	1	1	0	0	0	0	8.9	1	1	0	0	0	0
flex (d3de49f)	22.3	3	4	26.3	2.5	0	0	0	0	0	0	51.0	0	0	0	0	1	1
WavPack (22977b2)	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



Effectiveness

Existing memory error repair tool [ICSE 2018]

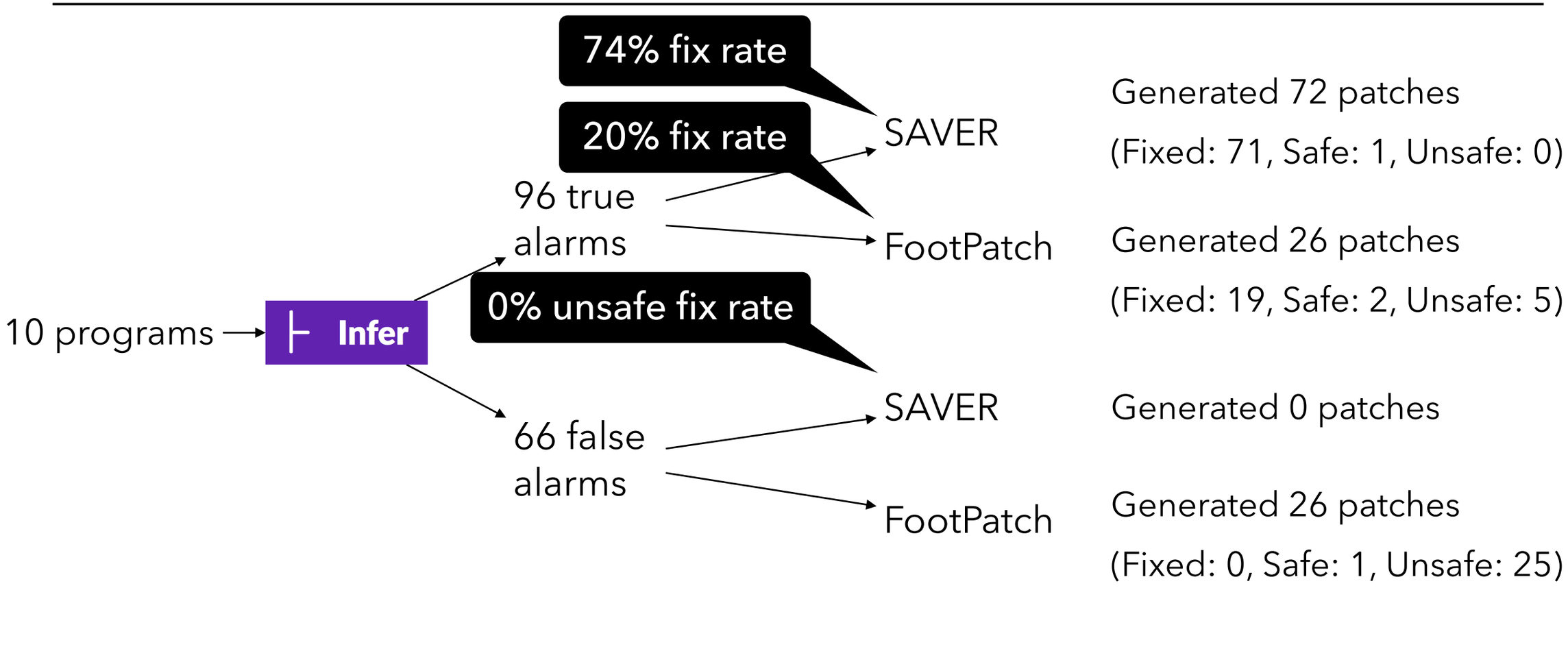
		Infer				SAVER						FootPatch [60]						
Program	**kLoC**	**#T**	**#F**	**Pre(s)**	**Fix(s)**	G_T	✓$_T$	△$_T$	✗$_T$	G_F	✗$_F$	**Fix(s)**	G_T	✓$_T$	△$_T$	✗$_T$	G_F	✗$_F$
rappel (ad8efd7)	2.2	1	0	2.2	0.0	1	1	0	0	0	0	8.9	1	1	0	0	0	0
flex (d3de49f)	22.3	3	4	26.3	2.5	0	0	0	0	0	0	51.0	0	0	0	0	1	1
WavPack (22977b2)	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



Effectiveness

Existing memory error repair tool [ICSE 2018]

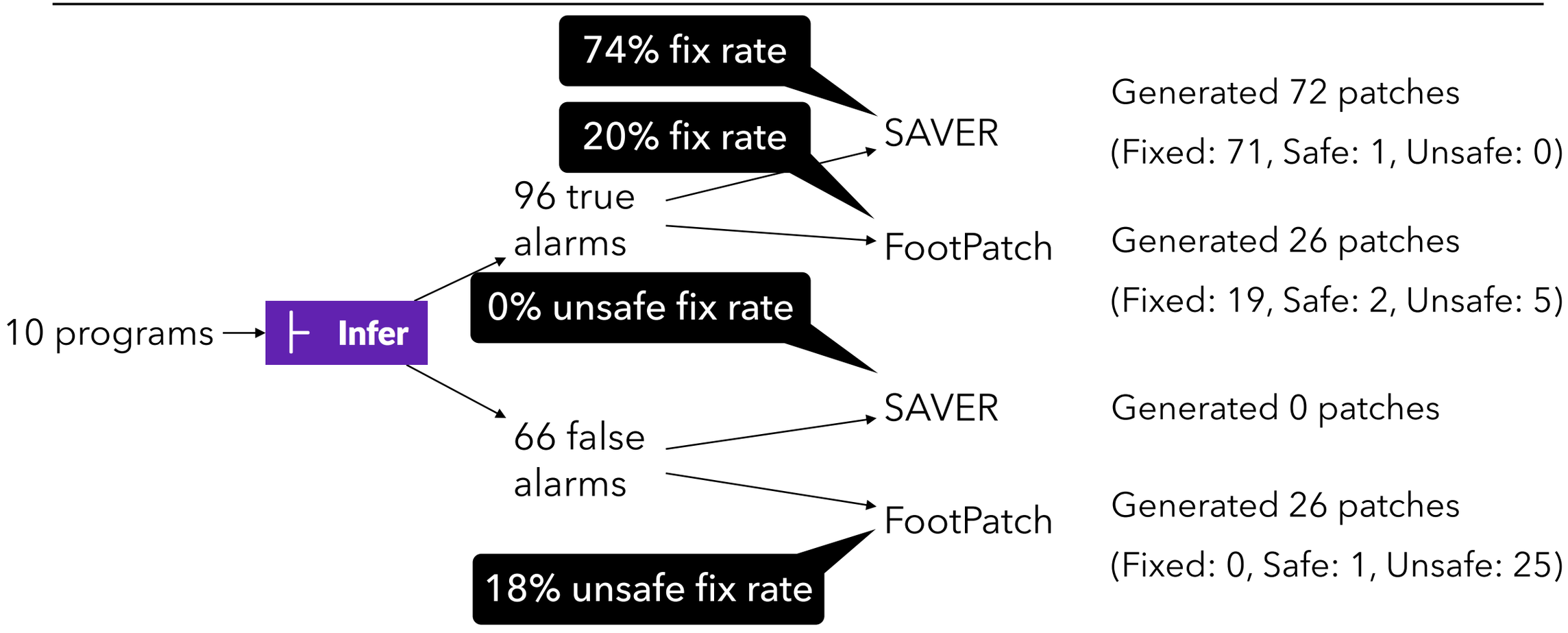
		Infer				SAVER						FootPatch [60]						
Program	**kLoC**	**#T**	**#F**	**Pre(s)**	**Fix(s)**	G_T	✓$_T$	△$_T$	✗$_T$	G_F	✗$_F$	**Fix(s)**	G_T	✓$_T$	△$_T$	✗$_T$	G_F	✗$_F$
rappel (ad8efd7)	2.2	1	0	2.2	0.0	1	1	0	0	0	0	8.9	1	1	0	0	0	0
flex (d3de49f)	22.3	3	4	26.3	2.5	0	0	0	0	0	0	51.0	0	0	0	0	1	1
WavPack (22977b2)	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



Effectiveness

Existing memory error repair tool [ICSE 2018]

		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



Summary

- **Static analysis-based program repair** for C memory errors
  - Scalable, precise, and safe
  - Successfully deployed in industry
  - https://github.com/kupl/kaprese

Thank you!