ISP RAS

Anxiety: a dynamic symbolic execution framework

Alexander Gerasimov
Senior Researcher
agerasimov@ispras.ru

Dynamic symbolic execution

- **Test coverage generation & debugging**
  - ✓ 1975 - SELECT (LISP)
  - ✓ 1976 – EFFIGY (PL/I)
  - ✓ 2004 – CUTE, DART (C)
- **Defects detection**
  - ✓ 2006 – EXE (C)
  - ✓ 2008 – BitBlaze (Executable, System-wide)
  - ✓ 2008 – SAGE (Executable)
  - ✓ 2008 – KLEE (LLVM)
  - ✓ 2010 – Avalanche (Executable)
  - ✓ 2011 – S^2E (Executable, System-wide)
  - ✓ 2012 – Mayhem (Executable)

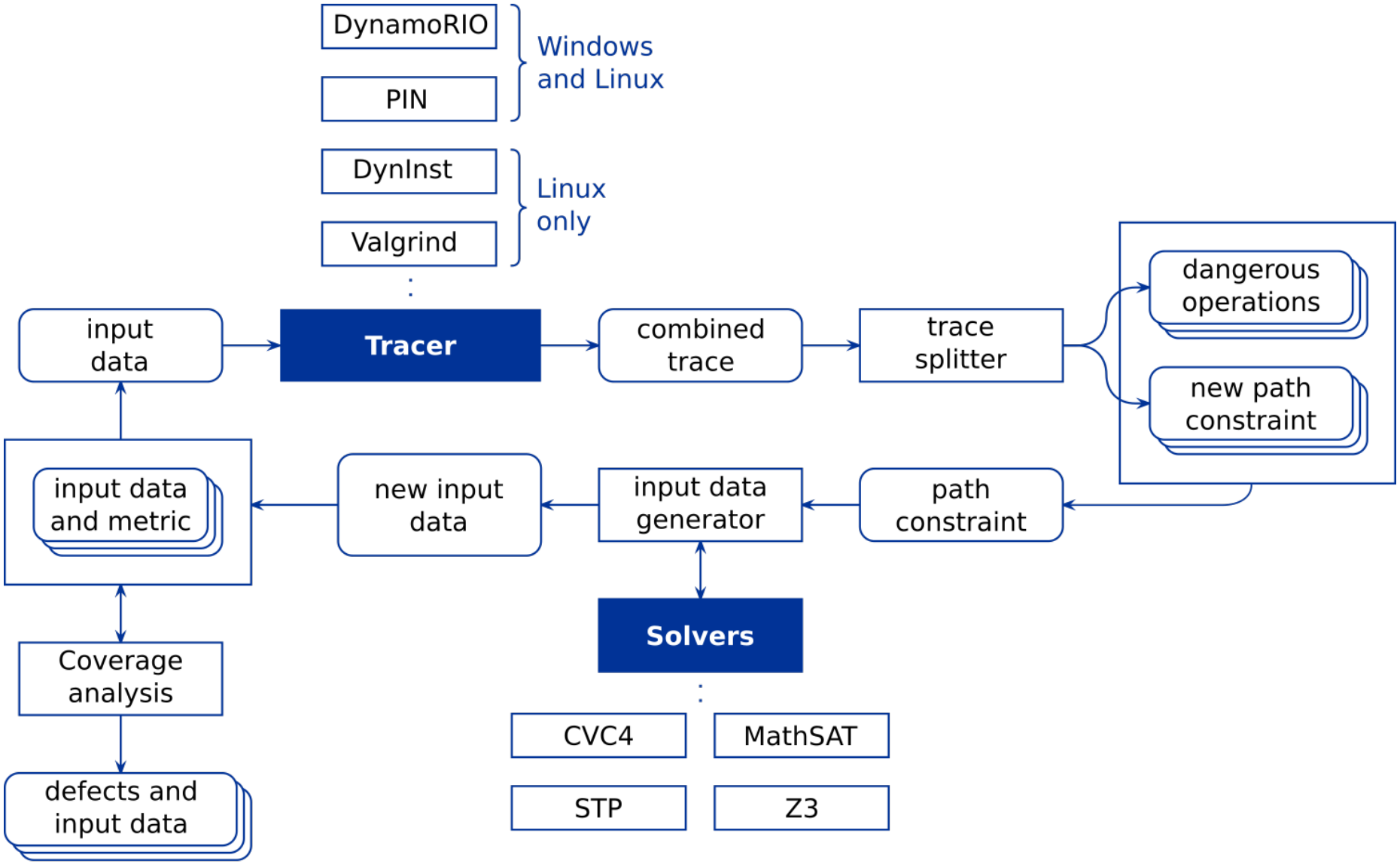
Limitations and requirements

- **Avalanche tool**
  - ✓ Based on Valgrind DBI Framework (Linux)
  - ✓ Heavily rely on CVC format

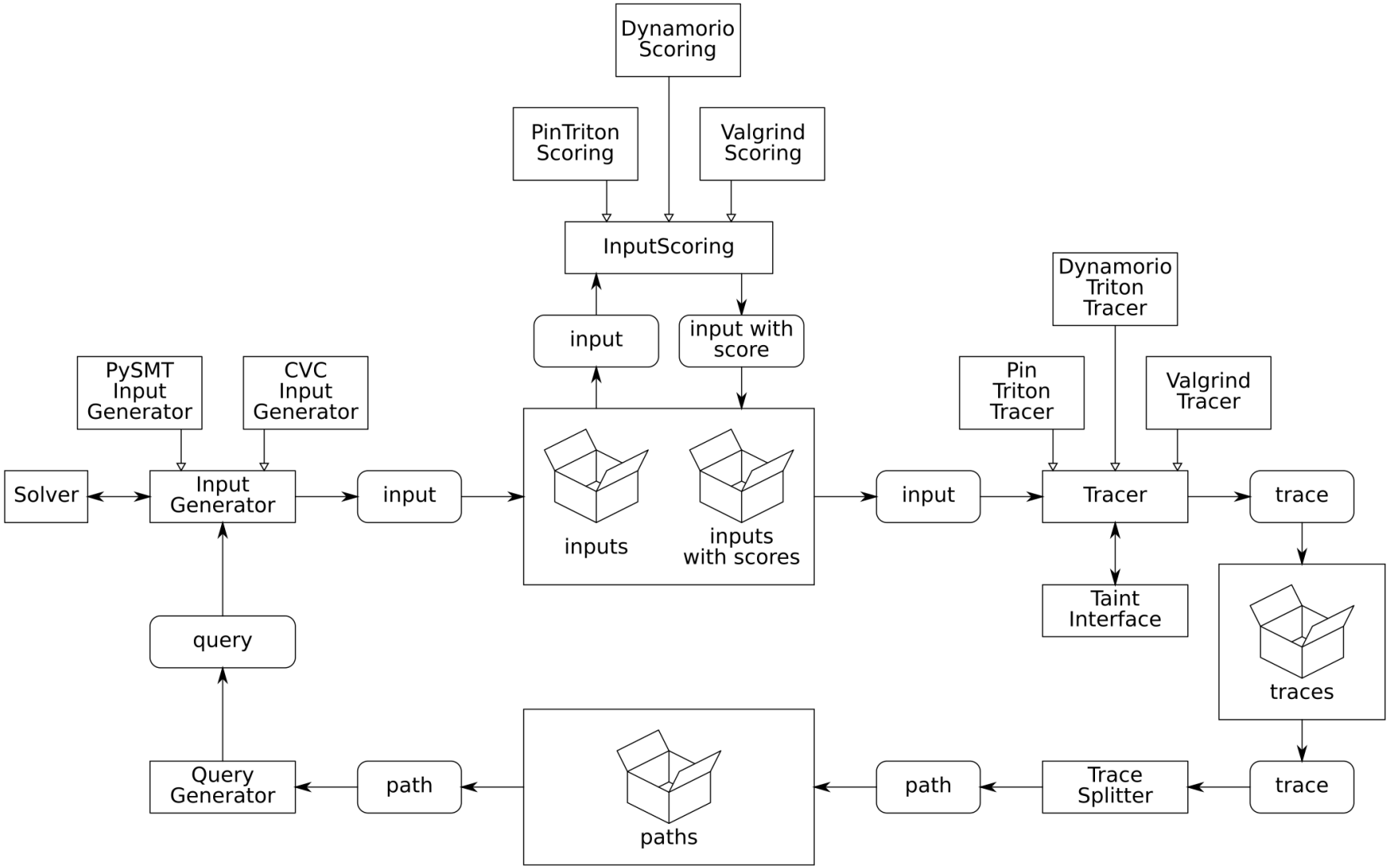
Develop a more flexible framework for dynamic symbolic execution

- **Requirements:**
  - ✓ User-mode programs analysis
  - ✓ Support Windows and Linux programs
  - ✓ Support different SMT-solvers
  - ✓ Support modular structure for solving different tasks

Anxiety framework



Anxiety framework



Analysis results

Program	Reason	Step	Condition
vde_l3	SEGV	2	Incorrect argument which starts with ‘-’
roarfilt	SEGV	49	Argument ‘-R’
umax_pp	SEGV	444	Argument ‘-[xywhlt] [^]+’
pnmhistmap	SEGV	20	File with content ‘50 36 37 50 30 50 39 32 49 00‘
jasper	ABRT	4	File with content ‘4d 49 46 0a a3 0a‘ and arguments are ‘-f <file> -t mif –T [type]‘
fdtdump	SEGV	31	File with incorrect content and length more than 8 bytes
faad	SEGV	6	File with content ‘41 44 49 46 00 00 01 00 00 00 00 60’
nettle-hash	ABRT	2	Incorrect argument starts with ‘-’ and ‘--’
instat	FPE	179	Argument ‘-i 0’

Goals

- **Anxiety: a framework for dynamic symbolic execution**
  - ✓ Supports user-mode programs dynamic symbolic execution
  - ✓ Supports analysis for Windows and Linux
  - ✓ Supports different SMT-solvers
  - ✓ Modular structure of tool allows adoption to different tasks

Limitations of approach

Under-taintedness due to indirect dependencies

```
char convtab = {‘a’, ‘b’, … , ‘a’, ‘b’, …};

char tolower(const char ch){

   // Symbolic index, untainted buffer
   return convtab[ch – ‘a’]; // *(convtab + ch – ‘a’)

}
```

Limitations

Under-taintedness due to indirect dependencies

```
Node* create(const Token &token) {
   If ( token == ‘select’ ) {          // External data
      return new SelectNode(); // Internal data
   } else if ( …
      …
   return NULL;                        // Internal data
}
```

Limitations

Under-taintedness due to indirect dependencies

```
switch(i) { // Symbolic i
    case 0: doZero(); break;
    case 1: doOne();
    case 2: doTwo(); break;
    default: doDefault();
}
```

```
mov ax, i
cmp ax, 2
jg DEFAULT
cmp ax, 0
jl DEFAULT
jmp table[ax]
data table
  ZERO
  ONE
  TWO
end data
ZERO: call doZero
jmp END
ONE: call doOne
TWO: call doTwo
jmp END
DEFAULT: call
doDefault
END:
```

Under-tainted jump

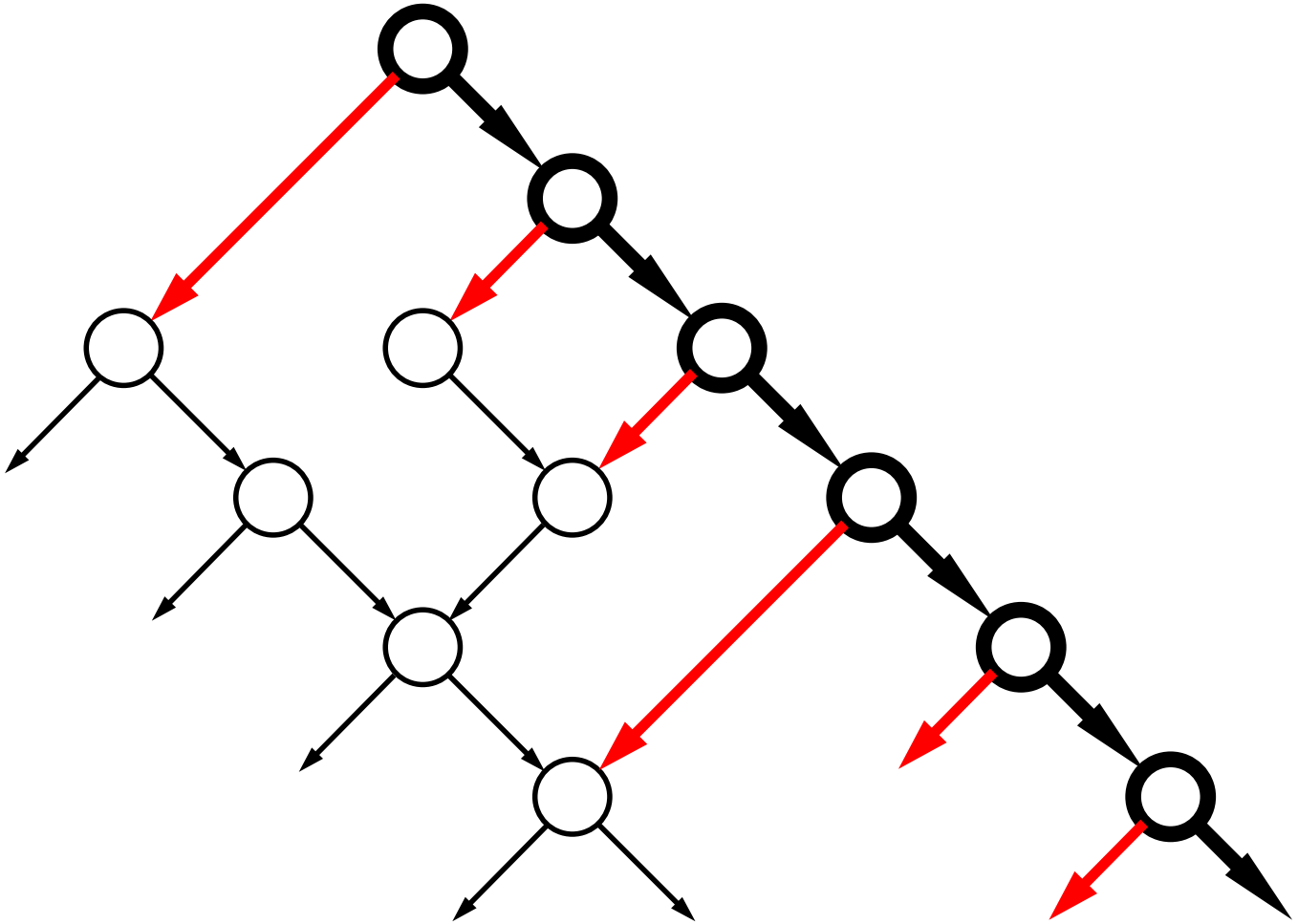
Limitations

Limited set of security predicates

- Divide by Zero

Limitations

Analysis path number explosion



Future research

- Under-/over-taintedness of program dataflow
- More security predicates (BoF, Memory Leak, ...)
- Analysis methods combination

Research & Development team

- Mikhail Ermakov
- Sergey Vartanov
- Leonid Kruglov
- Alexander Novikov
- Daniil Kutz
- Seryozha Asryan
- Alexander Gerasimov

Q & A session

Thank you!