LinuxTesting.org

Memory safety verification with SMGCPA: results, challenges and development plans

Anton Vasilyev

ISP RAS

Ivannikov Institute for System Programming of the RAS

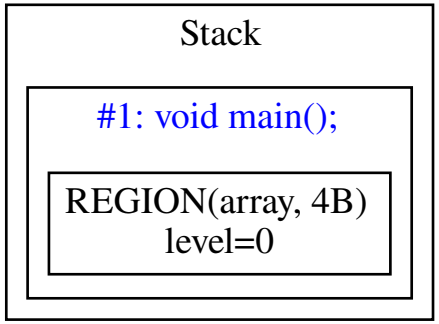
Memory Safety

- NULL pointer dereference
- Buffer overflow

- Memory management
  - Memory leaks
  - Use after free
  - Double free
  - Invalid free

Symbolic Memory Graph

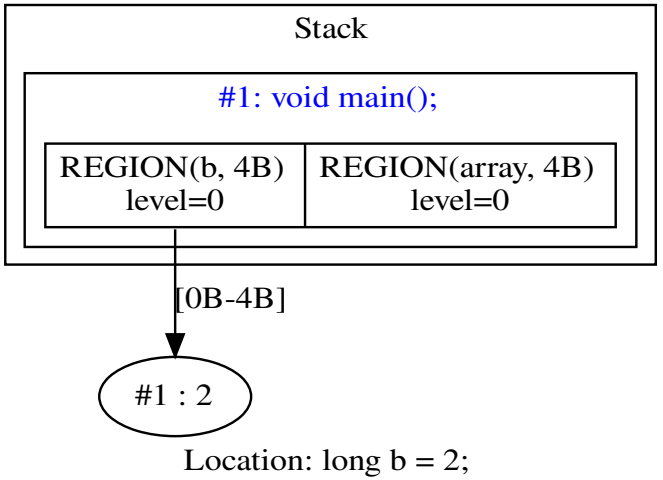
```
void main() {
    void *array;
    long b = 2;
    long c = 3;
    array = calloc(1, 16);
    memcpy(&array[4], &b, 4);
    memcpy(&array[5], &c, 4);
}
```



Location: void *array;

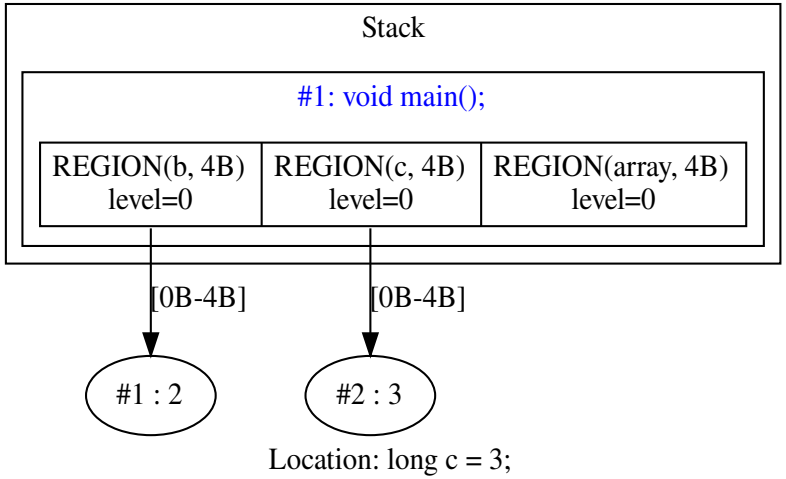
Symbolic Memory Graph

```
void main() {
    void *array;
    long b = 2;
    long c = 3;
    array = calloc(1, 16);
    memcpy(&array[4], &b, 4);
    memcpy(&array[5], &c, 4);
}
```



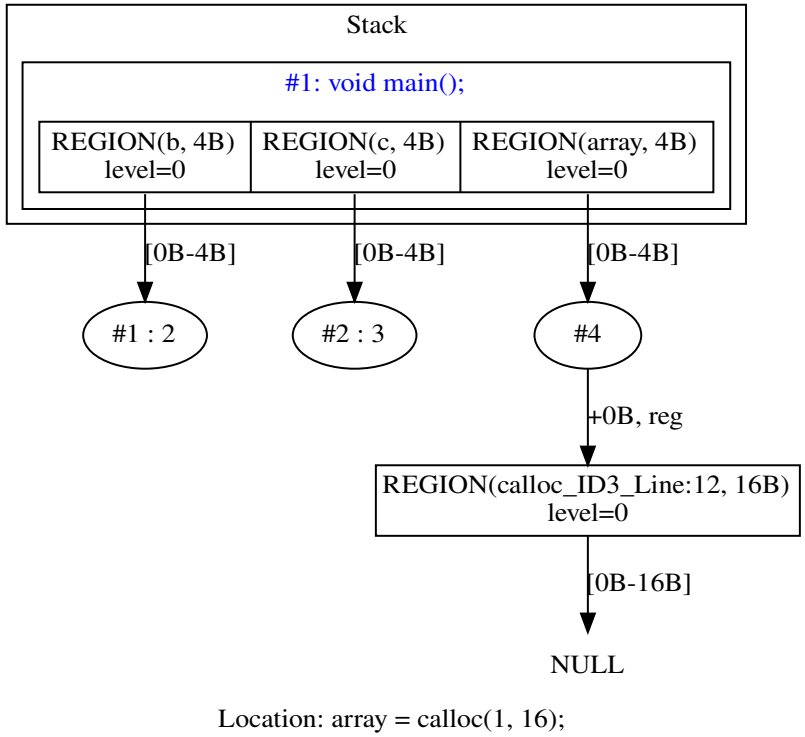
Symbolic Memory Graph

```
void main() {
    void *array;
    long b = 2;
    long c = 3;
    array = calloc(1, 16);
    memcpy(&array[4], &b, 4);
    memcpy(&array[5], &c, 4);
}
```



Symbolic Memory Graph

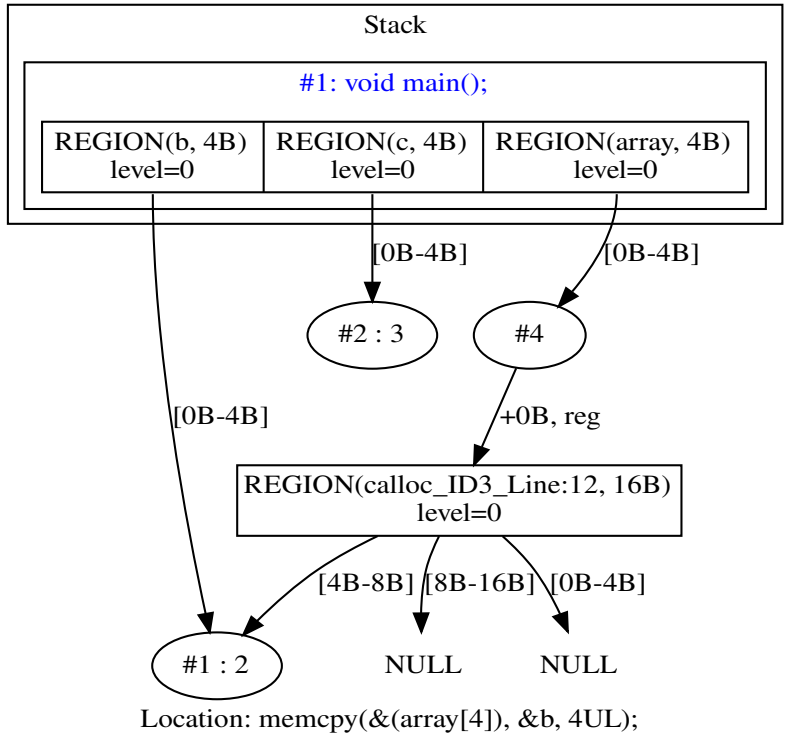
```
void main() {
    void *array;
    long b = 2;
    long c = 3;
    array = calloc(1, 16);
    memcpy(&array[4], &b, 4);
    memcpy(&array[5], &c, 4);
}
```



Location: array = calloc(1, 16);

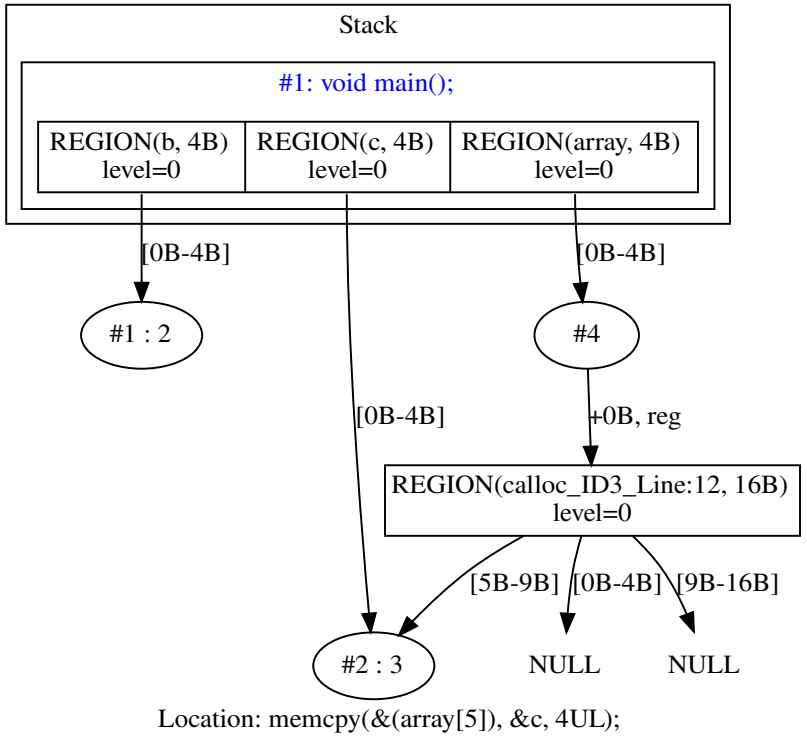
Symbolic Memory Graph

```
void main() {
    void *array;
    long b = 2;
    long c = 3;
    array = calloc(1, 16);
    memcpy(&array[4], &b, 4);
    memcpy(&array[5], &c, 4);
}
```



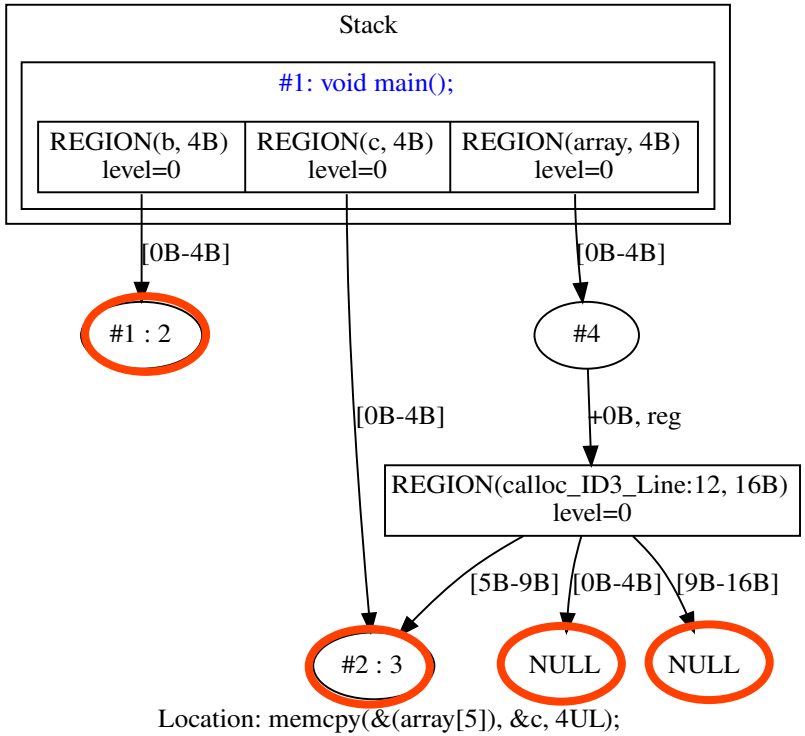
Symbolic Memory Graph

```
void main() {
    void *array;
    long b = 2;
    long c = 3;
    array = calloc(1, 16);
    memcpy(&array[4], &b, 4);
    memcpy(&array[5], &c, 4);
}
```



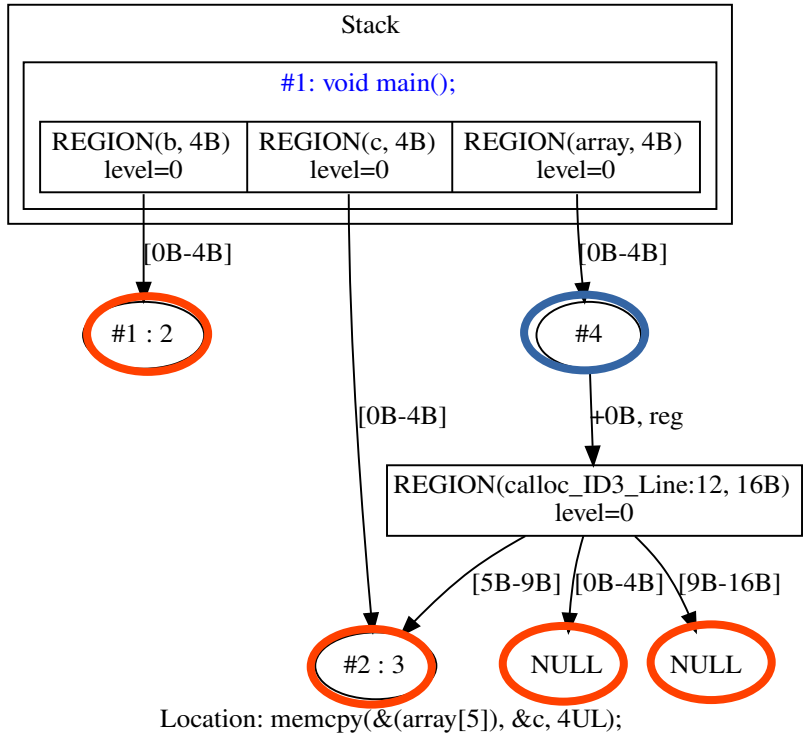
Symbolic Values

- Values



Symbolic Values

- Values
- Pointers



SMGCPA improvements

- Immutable state
- Invariant of graph memory: different values doesn`t intersects
- Efficient nonmodifying filtering

SMGCPA improvements

- Before
  - Number of computed successors: **126413**
- After
  - Number of computed successors: **447917**

Abstractions

- List abstractions
- Predicates
- Memory-on-Demand

Challenges

- How to cover more code
- How to represent results
- Merge states
- Improve stop operator

Future work

- CEGAR based on joining symbolic values with keeping history
- Add different levels of bug assurance
- Abstractions for strings and arrays
- Experiment with block abstractions

LinuxTesting.org

Thank you!

Anton Vasilyev

ISP RAS

Ivannikov Institute for System Programming of the RAS