

LLMs as verification oracles for Solidity

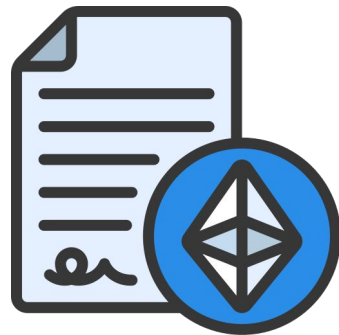
Massimo Bartoletti

Enrico Lipparini

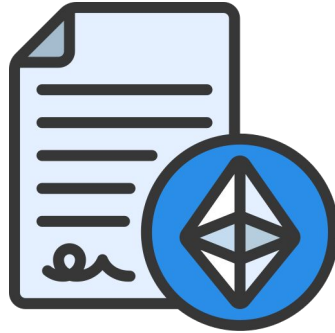
Livio Pompianu

University of Cagliari, Italy

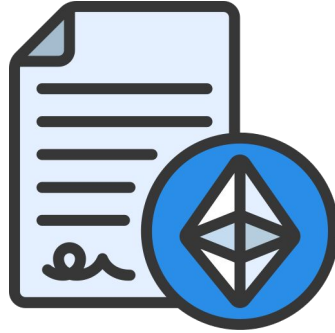




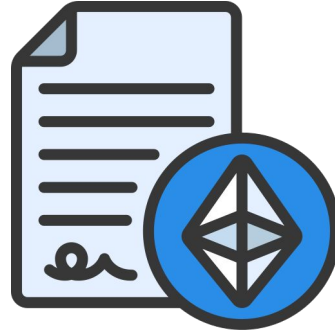
> \$100 billion



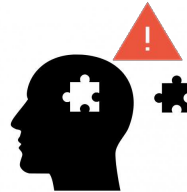
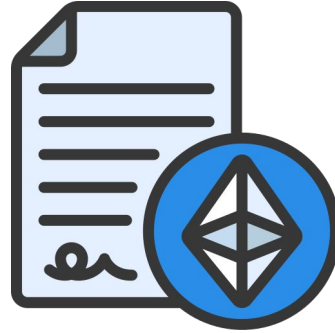
> \$100 billion



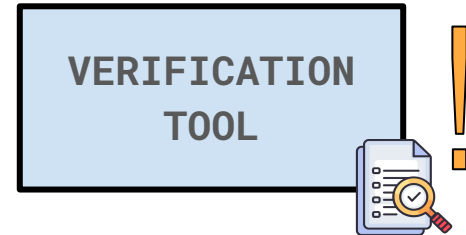
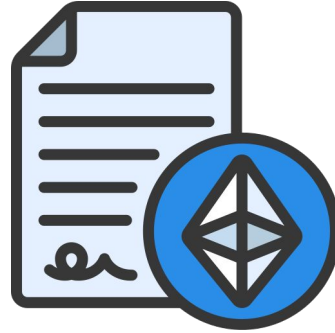
> \$100 billion

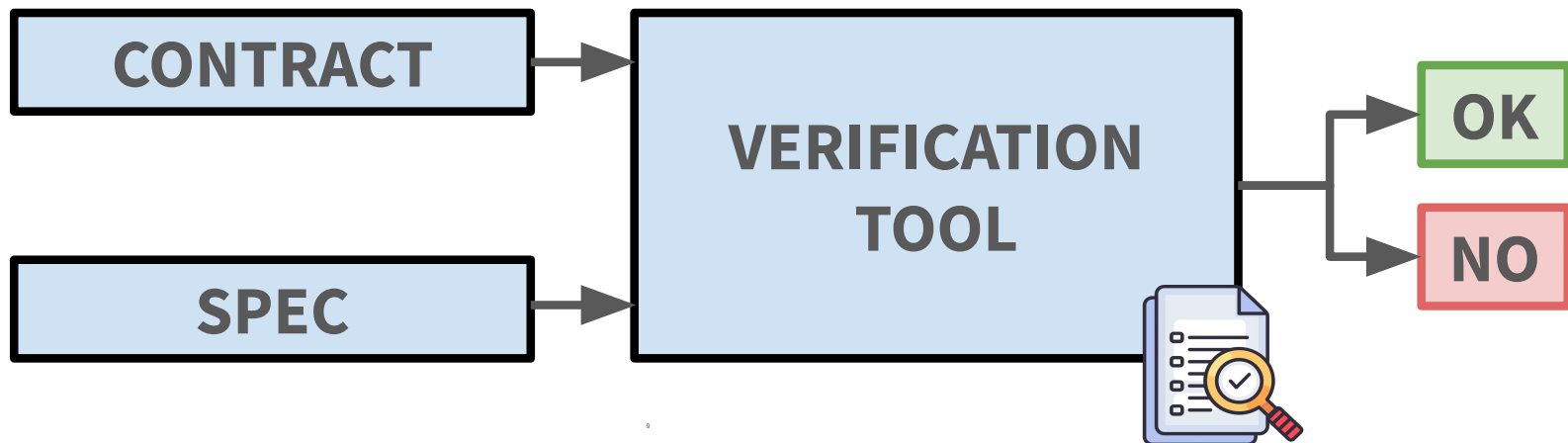


> \$100 billion

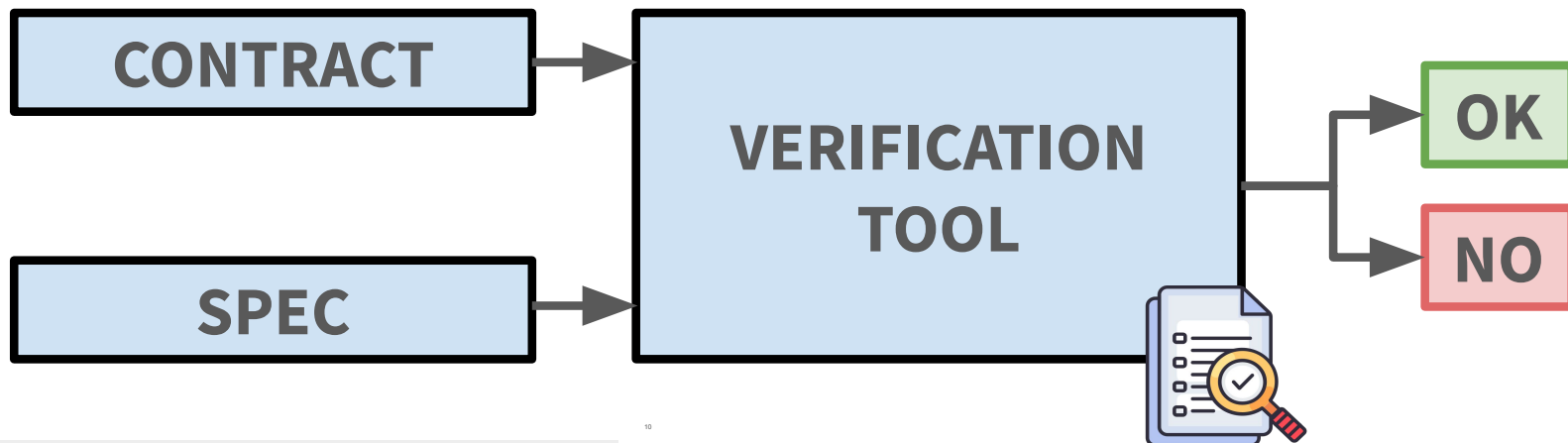


> \$100 billion



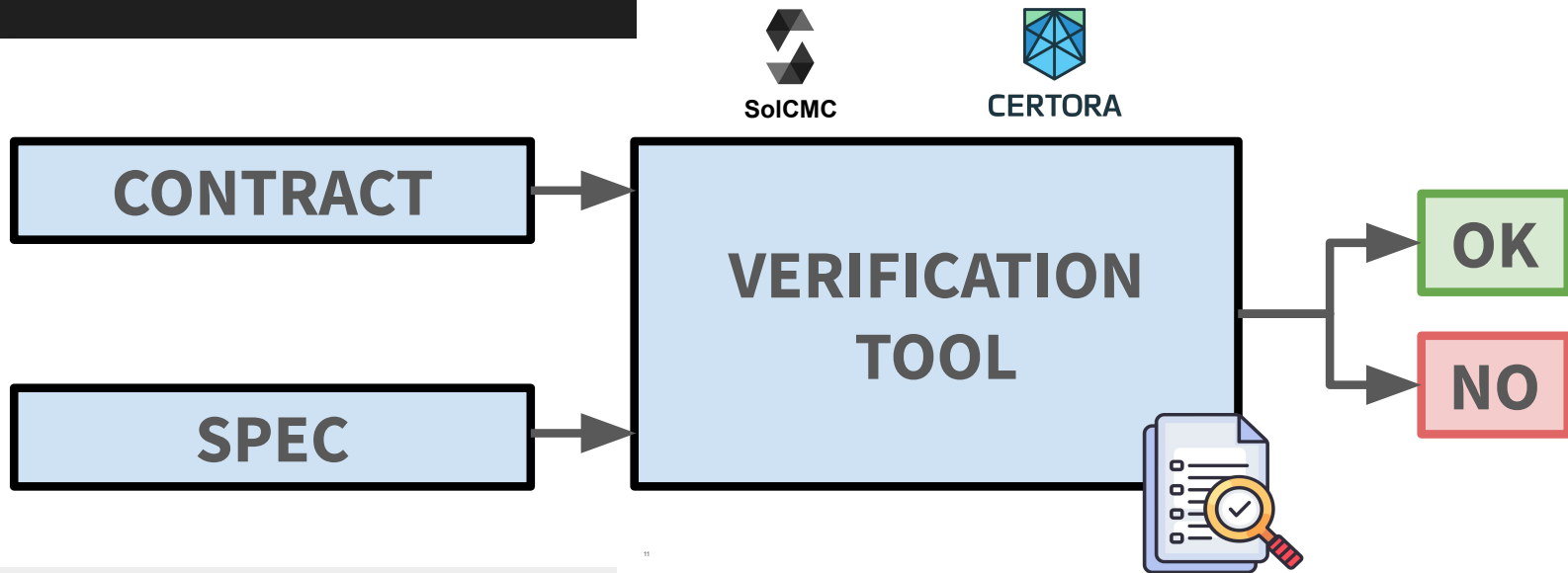



```
contract Bank {  
    mapping (address user => uint credit) credits;  
  
    function deposit() public payable {  
        credits[msg.sender] += msg.value;  
    }  
}
```



*"after a non-reverting **deposit()**,
the credits of **msg.sender**
are increased by **msg.value**"*

```
contract Bank {  
    mapping (address user => uint credit) credits;  
  
    function deposit() public payable {  
        credits[msg.sender] += msg.value;  
    }  
}
```



"after a non-reverting **deposit()**,
the credits of *msg.sender*
are increased by *msg.value*"

issues

The screenshot shows the RStudio IDE with a script editor on the left and a console on the right. The script editor contains the following R code:

```

# 1. Laden der Pakete
library(ggplot2)
library(dplyr)
library(readr)

# 2. Arbeitsverzeichnis festlegen
setwd("C:/Users/.../RStudio")

# 3. Daten laden
data_path = "C:/Users/.../RStudio/data/"
data_file = "data.csv"
data = read_csv(paste0(data_path, data_file))

# 4. Daten explorieren
summary(data)
str(data)

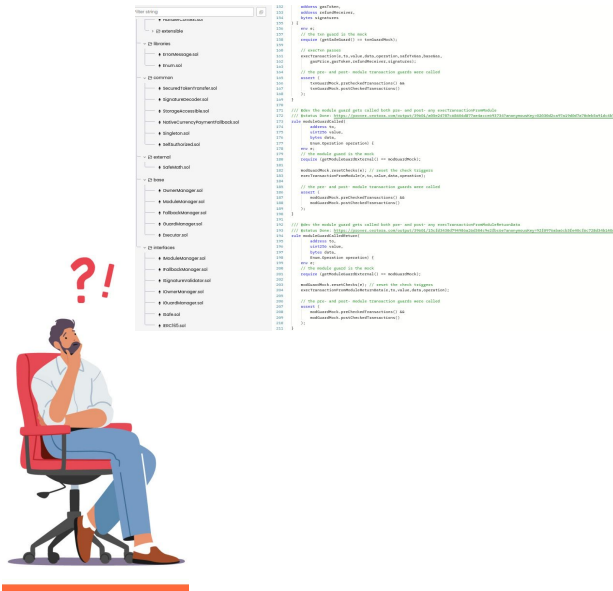
# 5. Daten visualisieren
ggplot(data, aes(x = "Year", y = "Sales")) +
  geom_line() +
  theme_minimal()

```

The console shows the output of the R code, including the working directory and the structure of the data file.

Formal Verification

issues



Formal Verification issues



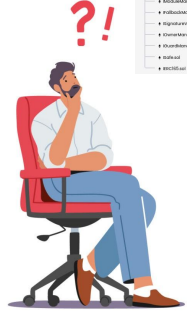
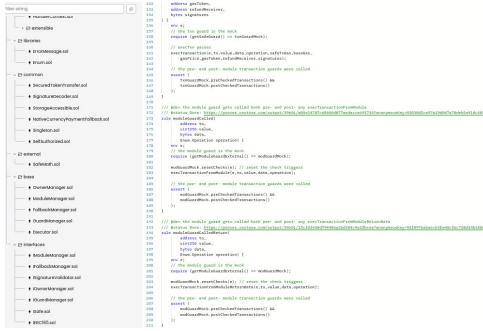
```
101 // ...
102 // ...
103 // ...
104 // ...
105 // ...
106 // ...
107 // ...
108 // ...
109 // ...
110 // ...
111 // ...
112 // ...
113 // ...
114 // ...
115 // ...
116 // ...
117 // ...
118 // ...
119 // ...
120 // ...
121 // ...
122 // ...
123 // ...
124 // ...
125 // ...
126 // ...
127 // ...
128 // ...
129 // ...
130 // ...
131 // ...
132 // ...
133 // ...
134 // ...
135 // ...
136 // ...
137 // ...
138 // ...
139 // ...
140 // ...
141 // ...
142 // ...
143 // ...
144 // ...
145 // ...
146 // ...
147 // ...
148 // ...
149 // ...
150 // ...
151 // ...
152 // ...
153 // ...
154 // ...
155 // ...
156 // ...
157 // ...
158 // ...
159 // ...
160 // ...
161 // ...
162 // ...
163 // ...
164 // ...
165 // ...
166 // ...
167 // ...
168 // ...
169 // ...
170 // ...
171 // ...
172 // ...
173 // ...
174 // ...
175 // ...
176 // ...
177 // ...
178 // ...
179 // ...
180 // ...
181 // ...
182 // ...
183 // ...
184 // ...
185 // ...
186 // ...
187 // ...
188 // ...
189 // ...
190 // ...
191 // ...
192 // ...
193 // ...
194 // ...
195 // ...
196 // ...
197 // ...
198 // ...
199 // ...
200 // ...
201 // ...
202 // ...
203 // ...
204 // ...
205 // ...
206 // ...
207 // ...
208 // ...
209 // ...
210 // ...
211 // ...
212 // ...
213 // ...
214 // ...
215 // ...
216 // ...
217 // ...
218 // ...
219 // ...
220 // ...
221 // ...
222 // ...
223 // ...
224 // ...
225 // ...
226 // ...
227 // ...
228 // ...
229 // ...
230 // ...
231 // ...
232 // ...
233 // ...
234 // ...
235 // ...
236 // ...
237 // ...
238 // ...
239 // ...
240 // ...
241 // ...
242 // ...
243 // ...
244 // ...
245 // ...
246 // ...
247 // ...
248 // ...
249 // ...
250 // ...
251 // ...
252 // ...
253 // ...
254 // ...
255 // ...
256 // ...
257 // ...
258 // ...
259 // ...
260 // ...
261 // ...
262 // ...
263 // ...
264 // ...
265 // ...
266 // ...
267 // ...
268 // ...
269 // ...
270 // ...
271 // ...
272 // ...
273 // ...
274 // ...
275 // ...
276 // ...
277 // ...
278 // ...
279 // ...
280 // ...
281 // ...
282 // ...
283 // ...
284 // ...
285 // ...
286 // ...
287 // ...
288 // ...
289 // ...
290 // ...
291 // ...
292 // ...
293 // ...
294 // ...
295 // ...
296 // ...
297 // ...
298 // ...
299 // ...
300 // ...
301 // ...
302 // ...
303 // ...
304 // ...
305 // ...
306 // ...
307 // ...
308 // ...
309 // ...
310 // ...
311 // ...
312 // ...
313 // ...
314 // ...
315 // ...
316 // ...
317 // ...
318 // ...
319 // ...
320 // ...
321 // ...
322 // ...
323 // ...
324 // ...
325 // ...
326 // ...
327 // ...
328 // ...
329 // ...
330 // ...
331 // ...
332 // ...
333 // ...
334 // ...
335 // ...
336 // ...
337 // ...
338 // ...
339 // ...
340 // ...
341 // ...
342 // ...
343 // ...
344 // ...
345 // ...
346 // ...
347 // ...
348 // ...
349 // ...
350 // ...
351 // ...
352 // ...
353 // ...
354 // ...
355 // ...
356 // ...
357 // ...
358 // ...
359 // ...
360 // ...
361 // ...
362 // ...
363 // ...
364 // ...
365 // ...
366 // ...
367 // ...
368 // ...
369 // ...
370 // ...
371 // ...
372 // ...
373 // ...
374 // ...
375 // ...
376 // ...
377 // ...
378 // ...
379 // ...
380 // ...
381 // ...
382 // ...
383 // ...
384 // ...
385 // ...
386 // ...
387 // ...
388 // ...
389 // ...
390 // ...
391 // ...
392 // ...
393 // ...
394 // ...
395 // ...
396 // ...
397 // ...
398 // ...
399 // ...
400 // ...
401 // ...
402 // ...
403 // ...
404 // ...
405 // ...
406 // ...
407 // ...
408 // ...
409 // ...
410 // ...
411 // ...
412 // ...
413 // ...
414 // ...
415 // ...
416 // ...
417 // ...
418 // ...
419 // ...
420 // ...
421 // ...
422 // ...
423 // ...
424 // ...
425 // ...
426 // ...
427 // ...
428 // ...
429 // ...
430 // ...
431 // ...
432 // ...
433 // ...
434 // ...
435 // ...
436 // ...
437 // ...
438 // ...
439 // ...
440 // ...
441 // ...
442 // ...
443 // ...
444 // ...
445 // ...
446 // ...
447 // ...
448 // ...
449 // ...
450 // ...
451 // ...
452 // ...
453 // ...
454 // ...
455 // ...
456 // ...
457 // ...
458 // ...
459 // ...
460 // ...
461 // ...
462 // ...
463 // ...
464 // ...
465 // ...
466 // ...
467 // ...
468 // ...
469 // ...
470 // ...
471 // ...
472 // ...
473 // ...
474 // ...
475 // ...
476 // ...
477 // ...
478 // ...
479 // ...
480 // ...
481 // ...
482 // ...
483 // ...
484 // ...
485 // ...
486 // ...
487 // ...
488 // ...
489 // ...
490 // ...
491 // ...
492 // ...
493 // ...
494 // ...
495 // ...
496 // ...
497 // ...
498 // ...
499 // ...
500 // ...
501 // ...
502 // ...
503 // ...
504 // ...
505 // ...
506 // ...
507 // ...
508 // ...
509 // ...
510 // ...
511 // ...
512 // ...
513 // ...
514 // ...
515 // ...
516 // ...
517 // ...
518 // ...
519 // ...
520 // ...
521 // ...
522 // ...
523 // ...
524 // ...
525 // ...
526 // ...
527 // ...
528 // ...
529 // ...
530 // ...
531 // ...
532 // ...
533 // ...
534 // ...
535 // ...
536 // ...
537 // ...
538 // ...
539 // ...
540 // ...
541 // ...
542 // ...
543 // ...
544 // ...
545 // ...
546 // ...
547 // ...
548 // ...
549 // ...
550 // ...
551 // ...
552 // ...
553 // ...
554 // ...
555 // ...
556 // ...
557 // ...
558 // ...
559 // ...
560 // ...
561 // ...
562 // ...
563 // ...
564 // ...
565 // ...
566 // ...
567 // ...
568 // ...
569 // ...
570 // ...
571 // ...
572 // ...
573 // ...
574 // ...
575 // ...
576 // ...
577 // ...
578 // ...
579 // ...
580 // ...
581 // ...
582 // ...
583 // ...
584 // ...
585 // ...
586 // ...
587 // ...
588 // ...
589 // ...
590 // ...
591 // ...
592 // ...
593 // ...
594 // ...
595 // ...
596 // ...
597 // ...
598 // ...
599 // ...
600 // ...
601 // ...
602 // ...
603 // ...
604 // ...
605 // ...
606 // ...
607 // ...
608 // ...
609 // ...
610 // ...
611 // ...
612 // ...
613 // ...
614 // ...
615 // ...
616 // ...
617 // ...
618 // ...
619 // ...
620 // ...
621 // ...
622 // ...
623 // ...
624 // ...
625 // ...
626 // ...
627 // ...
628 // ...
629 // ...
630 // ...
631 // ...
632 // ...
633 // ...
634 // ...
635 // ...
636 // ...
637 // ...
638 // ...
639 // ...
640 // ...
641 // ...
642 // ...
643 // ...
644 // ...
645 // ...
646 // ...
647 // ...
648 // ...
649 // ...
650 // ...
651 // ...
652 // ...
653 // ...
654 // ...
655 // ...
656 // ...
657 // ...
658 // ...
659 // ...
660 // ...
661 // ...
662 // ...
663 // ...
664 // ...
665 // ...
666 // ...
667 // ...
668 // ...
669 // ...
670 // ...
671 // ...
672 // ...
673 // ...
674 // ...
675 // ...
676 // ...
677 // ...
678 // ...
679 // ...
680 // ...
681 // ...
682 // ...
683 // ...
684 // ...
685 // ...
686 // ...
687 // ...
688 // ...
689 // ...
690 // ...
691 // ...
692 // ...
693 // ...
694 // ...
695 // ...
696 // ...
697 // ...
698 // ...
699 // ...
700 // ...
701 // ...
702 // ...
703 // ...
704 // ...
705 // ...
706 // ...
707 // ...
708 // ...
709 // ...
710 // ...
711 // ...
712 // ...
713 // ...
714 // ...
715 // ...
716 // ...
717 // ...
718 // ...
719 // ...
720 // ...
721 // ...
722 // ...
723 // ...
724 // ...
725 // ...
726 // ...
727 // ...
728 // ...
729 // ...
730 // ...
731 // ...
732 // ...
733 // ...
734 // ...
735 // ...
736 // ...
737 // ...
738 // ...
739 // ...
740 // ...
741 // ...
742 // ...
743 // ...
744 // ...
745 // ...
746 // ...
747 // ...
748 // ...
749 // ...
750 // ...
751 // ...
752 // ...
753 // ...
754 // ...
755 // ...
756 // ...
757 // ...
758 // ...
759 // ...
760 // ...
761 // ...
762 // ...
763 // ...
764 // ...
765 // ...
766 // ...
767 // ...
768 // ...
769 // ...
770 // ...
771 // ...
772 // ...
773 // ...
774 // ...
775 // ...
776 // ...
777 // ...
778 // ...
779 // ...
780 // ...
781 // ...
782 // ...
783 // ...
784 // ...
785 // ...
786 // ...
787 // ...
788 // ...
789 // ...
790 // ...
791 // ...
792 // ...
793 // ...
794 // ...
795 // ...
796 // ...
797 // ...
798 // ...
799 // ...
800 // ...
801 // ...
802 // ...
803 // ...
804 // ...
805 // ...
806 // ...
807 // ...
808 // ...
809 // ...
810 // ...
811 // ...
812 // ...
813 // ...
814 // ...
815 // ...
816 // ...
817 // ...
818 // ...
819 // ...
820 // ...
821 // ...
822 // ...
823 // ...
824 // ...
825 // ...
826 // ...
827 // ...
828 // ...
829 // ...
830 // ...
831 // ...
832 // ...
833 // ...
834 // ...
835 // ...
836 // ...
837 // ...
838 // ...
839 // ...
840 // ...
841 // ...
842 // ...
843 // ...
844 // ...
845 // ...
846 // ...
847 // ...
848 // ...
849 // ...
850 // ...
851 // ...
852 // ...
853 // ...
854 // ...
855 // ...
856 // ...
857 // ...
858 // ...
859 // ...
860 // ...
861 // ...
862 // ...
863 // ...
864 // ...
865 // ...
866 // ...
867 // ...
868 // ...
869 // ...
870 // ...
871 // ...
872 // ...
873 // ...
874 // ...
875 // ...
876 // ...
877 // ...
878 // ...
879 // ...
880 // ...
881 // ...
882 // ...
883 // ...
884 // ...
885 // ...
886 // ...
887 // ...
888 // ...
889 // ...
890 // ...
891 // ...
892 // ...
893 // ...
894 // ...
895 // ...
896 // ...
897 // ...
898 // ...
899 // ...
900 // ...
901 // ...
902 // ...
903 // ...
904 // ...
905 // ...
906 // ...
907 // ...
908 // ...
909 // ...
910 // ...
911 // ...
912 // ...
913 // ...
914 // ...
915 // ...
916 // ...
917 // ...
918 // ...
919 // ...
920 // ...
921 // ...
922 // ...
923 // ...
924 // ...
925 // ...
926 // ...
927 // ...
928 // ...
929 // ...
930 // ...
931 // ...
932 // ...
933 // ...
934 // ...
935 // ...
936 // ...
937 // ...
938 // ...
939 // ...
940 // ...
941 // ...
942 // ...
943 // ...
944 // ...
945 // ...
946 // ...
947 // ...
948 // ...
949 // ...
950 // ...
951 // ...
952 // ...
953 // ...
954 // ...
955 // ...
956 // ...
957 // ...
958 // ...
959 // ...
960 // ...
961 // ...
962 // ...
963 // ...
964 // ...
965 // ...
966 // ...
967 // ...
968 // ...
969 // ...
970 // ...
971 // ...
972 // ...
973 // ...
974 // ...
975 // ...
976 // ...
977 // ...
978 // ...
979 // ...
980 // ...
981 // ...
982 // ...
983 // ...
984 // ...
985 // ...
986 // ...
987 // ...
988 // ...
989 // ...
990 // ...
991 // ...
992 // ...
993 // ...
994 // ...
995 // ...
996 // ...
997 // ...
998 // ...
999 // ...
1000 // ...
```



Semantics abstraction



Formal Verification issues



Semantics abstraction





SolCMC



CERTORA

**VERIFICATION
TOOL**



RQ1



VERIFICATION
TOOL



RQ1



VERIFICATION
TOOL



RQ2



SoICMC



CERTORA

VS

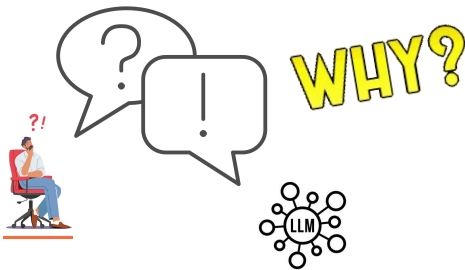


RQ1



VERIFICATION
TOOL

RQ3



RQ2



SoICMC



CERTORA

VS

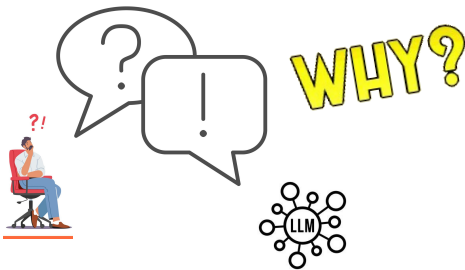


RQ1



VERIFICATION
TOOL

RQ3



RQ2



SoICMC



CERTORA

VS

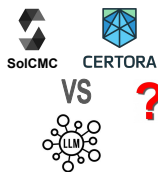


RQ4





RQ1



RQ2



RQ3



RQ4

Controlled dataset

 [bitbart](#) / [contracts-verification-benchmark](#)

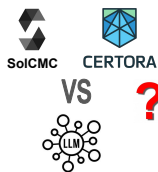
- 5 use cases ($\rightarrow$ LP)
- 63 mutations
- 159 properties in natural language
- 2034 verification tasks
- specs for SolCMC / Certora

Certora Reports

- Safe (smart account wallet)
- InfiniFi (yield-bearing deposits)
- 75 properties (written in CVL + NL)



RQ1



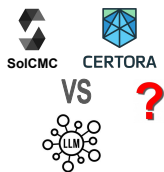
RQ2

specs
expressible
in Certora {

Dataset	Tool	F1 score
DS _{CVL}	Certora	77%
DS _{CVL}	GPT-5	92%
DS _{CVL}	GPT-4	68%

specs
expressible
in SolCMC {

DS _{SolCMC}	SolCMC	79%
DS _{SolCMC}	GPT-5	90%
DS _{SolCMC}	GPT-4	66%



RQ1

RQ2

specs
expressible
in Certora {

Dataset	Tool	F1 score
DS _{CVL}	Certora	77%
DS _{CVL}	GPT-5	92%
DS _{CVL}	GPT-4	68%

specs *not*
expressible in Certora

DS _{¬CVL}	GPT-5	91%
---------------------	-------	-----

specs
expressible
in SolCMC {

DS _{SolCMC}	SolCMC	79%
DS _{SolCMC}	GPT-5	90%
DS _{SolCMC}	GPT-4	66%

specs *not*
expressible in SolCMC

DS _{¬SolCMC}	GPT-5	93%
------------------------	-------	-----



RQ3

PoC generated by GPT-5

- Deploy with payees [Owner O, Alice A], shares [1, 2] (totalShares = 3).
- Send 3 wei to the contract.
- Call release(O): pays 1 wei to O.
- Call release(A): pays 2 wei to A. Now totalReleased = 3, balance = 0, released[O]=1, released[A]=2, totalReceived = 3.
- Send 2 wei to the contract (balance = 2, totalReleased = 3, totalReceived = 5).
- Call release(A): payment = $\text{floor}(5 \cdot 2/3) - 2 = 1$; pays 1 wei; totalReleased = 4; balance = 1; released[A] = 3; totalReceived remains 5.
- Call release(O): payment = $\text{floor}(5 \cdot 1/3) - 1 = 0$; else-branch executes and transfers 1 wei to owner; totalReleased stays 4; balance becomes 0; totalReceived becomes 4.
- Now for A: $\text{RHS} = \text{floor}(4 \cdot 2/3) = 2$, but released[A] = 3 and $\text{releasable}(A) = \text{floor}(4 \cdot 2/3) - 3 = -1$ (underflow $\rightarrow$ revert).

Hence the property " $\text{released}[A] + \text{releasable}(A) == (\text{totalReceived} * \text{shares}[A]) / \text{totalShares}$ " is violated (releasable(A) cannot be computed).

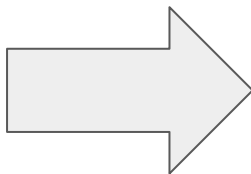


RQ3

PoC generated by GPT-5

- Deploy with payees [Owner O, Alice A], shares [1, 2] (totalShares = 3).
- Send 3 wei to the contract.
- Call release(O): pays 1 wei to O.
- Call release(A): pays 2 wei to A. Now totalReleased = 3, balance = 0, released[O]=1, released[A]=2, totalReceived = 3.
- Send 2 wei to the contract (balance = 2, totalReleased = 3, totalReceived = 5).
- Call release(A): payment = $\text{floor}(5 \cdot 2/3) - 2 = 1$; pays 1 wei; totalReleased = 4; balance = 1; released[A] = 3; totalReceived remains 5.
- Call release(O): payment = $\text{floor}(5 \cdot 1/3) - 1 = 0$; else-branch executes and transfers 1 wei to owner; totalReleased stays 4; balance becomes 0; totalReceived becomes 4.
- Now for A: $\text{RHS} = \text{floor}(4 \cdot 2/3) = 2$, but released[A] = 3 and $\text{releasable}(A) = \text{floor}(4 \cdot 2/3) - 3 = -1$ (underflow $\rightarrow$ revert). Hence the property " $\text{released}[A] + \text{releasable}(A) == (\text{totalReceived} * \text{shares}[A]) / \text{totalShares}$ " is violated (releasable(A) cannot be computed).

Manual validation in Forge



```
contract PaymentSplitterTest is Test {
    PaymentSplitter public ps;
    uint256 private totalReceived;

    function setUp() public {
        // ...
    }

    function test_fair_split_eq() public {
        address o = address(0x1);
        address b = address(0x2);
        address c = address(0x3);

        address[] memory payees = new address[](2);
        payees[0] = address(o);
        payees[1] = address(b);

        uint256[] memory shares = new uint256[](2);
        shares[0] = 1;
        shares[1] = 1;

        vm.prank(o);
        ps = new PaymentSplitter(payees, shares);

        // C sends 100 wei to the contract
        vm.deal(address(c), 2);
        vm.prank(c);
        (bool success,) = address(ps).call(value:2(""));
        assert(success);

        ps.release(payable(b));
        ps.release(payable(c));

        // rhs = (totalReceived * shares[b]) / totalShares
        uint totalReceived = address(ps).balance + ps.getTotalReleased();
        uint rhs = (totalReceived * ps.getShares(b)) / ps.getTotalShares();
        assertEq(rhs, 0);

        // lhs = released[b] + releasable(b)
        vm.expectRevert();
        uint releasable_b = ps.releasable(b);

        // violation: lhs (reverted) != rhs
    }
}
```

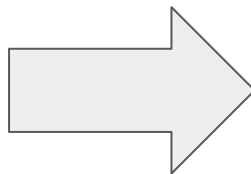



RQ3

PoC generated by GPT-5

- Deploy with payees [Owner O, Alice A], shares [1, 2] (totalShares = 3).
- Send 3 wei to the contract.
- Call release(O): pays 1 wei to O.
- Call release(A): pays 2 wei to A. Now totalReleased = 3, balance = 0, released[O]=1, released[A]=2, totalReceived = 3.
- Send 2 wei to the contract (balance = 2, totalReleased = 3, totalReceived = 5).
- Call release(A): payment = $\text{floor}(5 \cdot 2/3) - 2 = 1$; pays 1 wei; totalReleased = 4; balance = 1; released[A] = 3; totalReceived remains 5.
- Call release(O): payment = $\text{floor}(5 \cdot 1/3) - 1 = 0$; else-branch executes and transfers 1 wei to owner; totalReleased stays 4; balance becomes 0; totalReceived becomes 4.
- Now for A: $\text{RHS} = \text{floor}(4 \cdot 2/3) = 2$, but released[A] = 3 and releasable(A) = $\text{floor}(4 \cdot 2/3) - 3 = -1$ (underflow → revert). Hence the property "released[A] + releasable(A) == (totalReceived * shares[A]) / totalShares" is violated (releasable(A) cannot be computed).

Manual validation in Forge



Correctness	92%
Completeness	89%
Coherence	95%

```
contract PaymentSplitterTest is Test {
    PaymentSplitter public ps;
    uint256 private totalReceived;

    function setUp() public {
        // ...

    }

    function test_fair_split_eq() public {
        address o = address(0x1);
        address b = address(0x2);
        address c = address(0x3);

        address[] memory payees = new address[](2);
        payees[0] = address(o);
        payees[1] = address(b);

        uint256[] memory shares = new uint256[](2);
        shares[0] = 1;
        shares[1] = 1;

        vm.prank(o);
        ps = new PaymentSplitter(payees, shares);

        // C sends 100 wei to the contract
        vm.deal(address(c), 2);
        vm.prank(c);
        (bool success,) = address(ps).call(value:2);
        assert(success);

        ps.release(payable(b));
        ps.release(payable(c));

        // rhs = (totalReceived * shares[b]) / totalShares
        uint totalReceived = address(ps).balance + ps.getTotalReleased();
        uint rhs = (totalReceived * ps.getShares(b)) / ps.getTotalShares();
        assertEq(rhs, 0);

        // lhs = released[b] + releasable(b)
        vm.expectRevert();
        uint releasable_b = ps.releasable(b);

        // violation: lhs (reverted) != rhs
    }
}
```

```
contract Bank {
    mapping (address user => uint credit) credits;

    function deposit() public payable {
        credits[msg.sender] += msg.value;
    }
}
```

*"after a non-reverting **deposit()**,
the credits of **msg.sender**
are increased by **msg.value**"*



- Deploy with payees [Owner O, Alice A], shares [1, 2] (totalShares = 3).
- Send 3 wei to the contract.
- Call release(O): pays 1 wei to O.
- Call release(A): pays 2 wei to A. Now totalReleased = 3, balance = 0, released[O]=1, released[A]=2, totalReceived = 3.
- Send 2 wei to the contract (balance = 2, totalReleased = 3, totalReceived = 5).
- Call release(A): payment = $\text{floor}(5 \cdot 2/3) - 2 = 1$; pays 1 wei; totalReleased = 4; balance = 1; released[A] = 3; totalReceived remains 5.
- Call release(O): payment = $\text{floor}(5 \cdot 1/3) - 1 = 0$; else-branch executes and transfers 1 wei to owner; totalReleased stays 4; balance becomes 0; totalReceived becomes 4.
- Now for A: $\text{RHS} = \text{floor}(4 \cdot 2/3) = 2$, but released[A] = 3 and $\text{releasable}(A) = \text{floor}(4 \cdot 2/3) - 3 = -1$ (underflow → revert). Hence the property " $\text{released}[A] + \text{releasable}(A) == (\text{totalReceived} * \text{shares}[A]) / \text{totalShares}$ " is violated (releasable(A) cannot be computed).



```
abstract contracts[] cs;

contract BankTest is Test {
  Bank immutable bank;

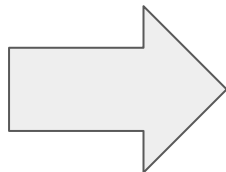
  constructor() {
    abstract address bank_deployer;
    vm.prank(bank_deployer);
    bank = new Bank();
  }

  function test_withdraw_assets_credit_others_violation()
    abstract transaction[] txs;
    abstract address user;
    uint256 user_creditsBefore = bank.credits(user);

    abstract uint256 amount;
    abstract address sender;
    vm.prank(sender);
    bank.withdraw(amount); // should not revert

    uint256 user_creditsAfter = bank.credits(user);

    assertNotEq(sender, user, "user equal to sender");
    assertNotEq(user_creditsBefore, user_creditsAfter, "cr
  }
}
```



```
contract Attacker {
  Bank public bank;
  User public user;

  constructor(Bank _bank) {
    bank = _bank;
  }

  function setUser(User _user) external {
    user = _user;
  }

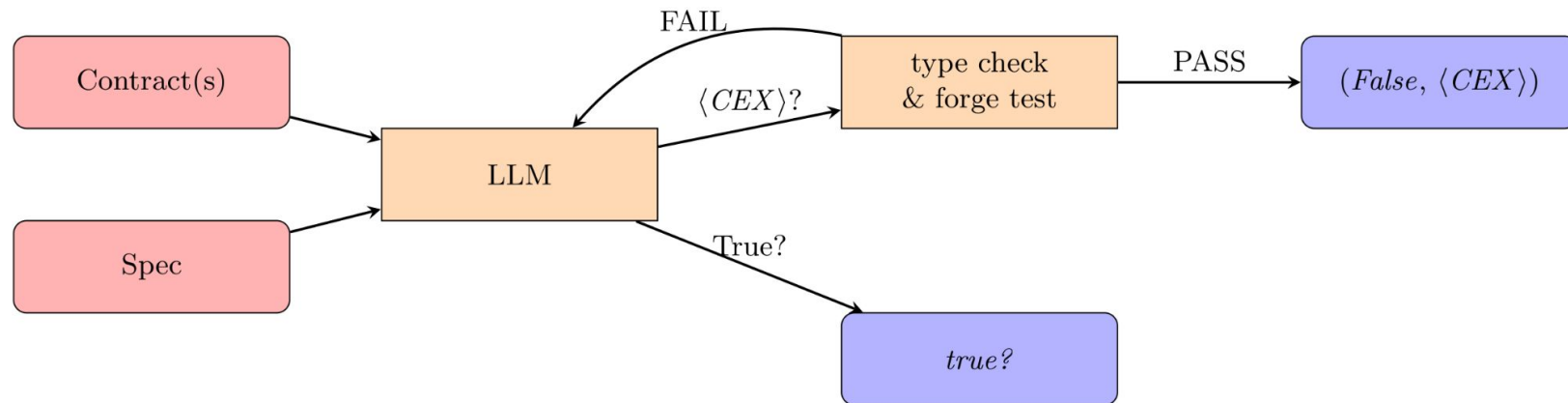
  // When receiving ETH from Bank.withdraw, forward it to User,
  // which deposits into Bank, crediting the User.
  receive() external payable {
    user.doDeposit{value: msg.value}(bank);
  }
}

contract BankTest is Test {
  constructor() {
    address bank_deployer = address(0xBEEF);
    [...] // deployment of bank
  }

  function test_withdraw_assets_credit_others_violation() public {
    User u = new User();
    Attacker a = new Attacker(bank);
    a.setUser(u);
    vm.deal(address(a), 1 ether);
    vm.prank(address(a));
    bank.deposit{value: 1 ether}();

    address user = address(u);
    [...] // def. of user_creditsBefore
    uint256 amount = 1 ether;
    address sender = address(a);
    [...] // call to bank.withdraw and asserts
  }
}
```





results # iter.	Total	(TRUE, <i>true?</i>)	(FALSE, <i>true?</i>)	(FALSE, <i>False</i>)
Total	109	57	4	48
0	90	46	4	40
1	15	10	0	5
2	4	1	0	3
3	0	0	0	0

In the wild (RQ4)



In the wild (RQ4)



CERTORA

In the wild (RQ4)



Spotting inconsistencies
between
NL property - formal spec



CERTORA

In the wild (RQ4)



Spotting inconsistencies
between
NL property - formal spec



CERTORA

Completing
trace violation

In the wild (RQ4)



Spotting inconsistencies
between
NL property - formal spec

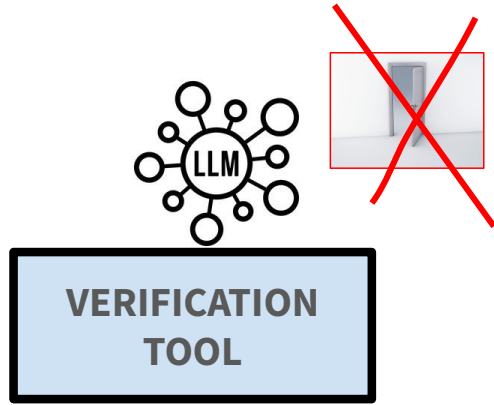


CERTORA

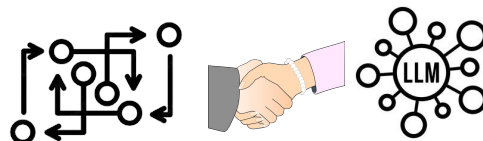
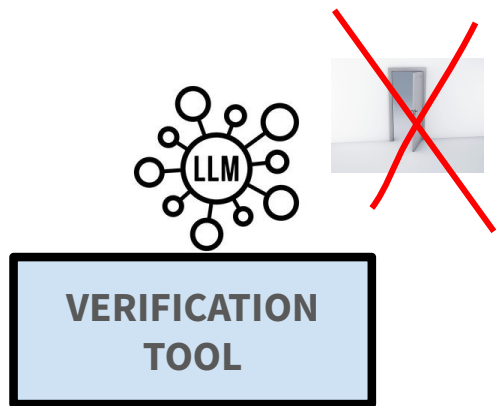
Completing
trace violation

Flattened
learning curve

Takeaway message



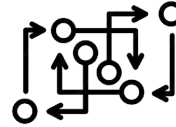
Takeaway message



Takeaway message



VERIFICATION
TOOL



```
contract PaymentSplitterTest is Test {
    PaymentSplitter public ps;
    function setUp() public {
        // ...
    }
    function test_fair_split_eq() public {
        address a = address(0x1);
        address b = address(0x2);
        address c = address(0x3);
        address() memory payees = new address[](2);
        payees[0] = address(a);
        payees[1] = address(b);
        uint256() memory shares = new uint256[](2);
        shares[0] = 1;
        shares[1] = 1;
        vm.prank(0);
        ps = new PaymentSplitter(payees, shares);
        // I send 100 wei to the contract
        vm.deal(address(c), 2);
        vm.prank(c);
        (bool success,) = address(ps).call(value(2) "");
        assert(success);
        ps.release(payable(b));
        ps.release(payable(c));
        // rhs = (totalReleased * shares[b]) / totalShares
        uint totalReleased = address(ps).balance + ps.getTotalReleased();
        uint rhs = (totalReleased * ps.getShares(b)) / ps.getTotalShares();
        assertEq(rhs, 0);
        // lhs = released(b) + released(c)
        vm.expectRevert();
        uint releasable = ps.releaseable(b);
        // violation: lhs (released) > rhs
    }
}
```



hallucinations