



中国科学院
CHINESE ACADEMY OF SCIENCES



中国科学院 信息工程研究所
INSTITUTE OF INFORMATION ENGINEERING, CAS

AURC: Detecting Errors in Program Code and Documentation

Peiwei Hu, Ruigang Liang, Ying Cao, Kai Chen, and Runze Zhang

SKLOIS, Institute of Information Engineering, CAS, China

School of Cyber Security, University of Chinese Academy of Sciences, China

Beijing Academy of Artificial Intelligence, China

1. Background

- ★ Open-source libraries greatly affect the security of downstream software.



Log4Shell

CVE-2021-44228

Affect 1800+ products

2. Previous Studies

How to detect the bugs in libraries:

Based on the
usage extracted
from documents

Based on majority
voting

Based on the
behavior in the
similar context

2. Previous Studies

How to detect the bugs in libraries:

Based on the
usage extracted
from documents

Based on majority
voting

Based on the
behavior in the
similar context

- ★ Documents are not trustworthy.
- ★ The majority voting is not trustworthy.
- ★ The usage in the similar context is not trustworthy.

3.1 Our Solution

How experienced programmers find the usage facing the unfamiliar API

Read documents

Scan the source code of API

Figure out how the existing code uses the API



★ API Usage Reference (AUR)

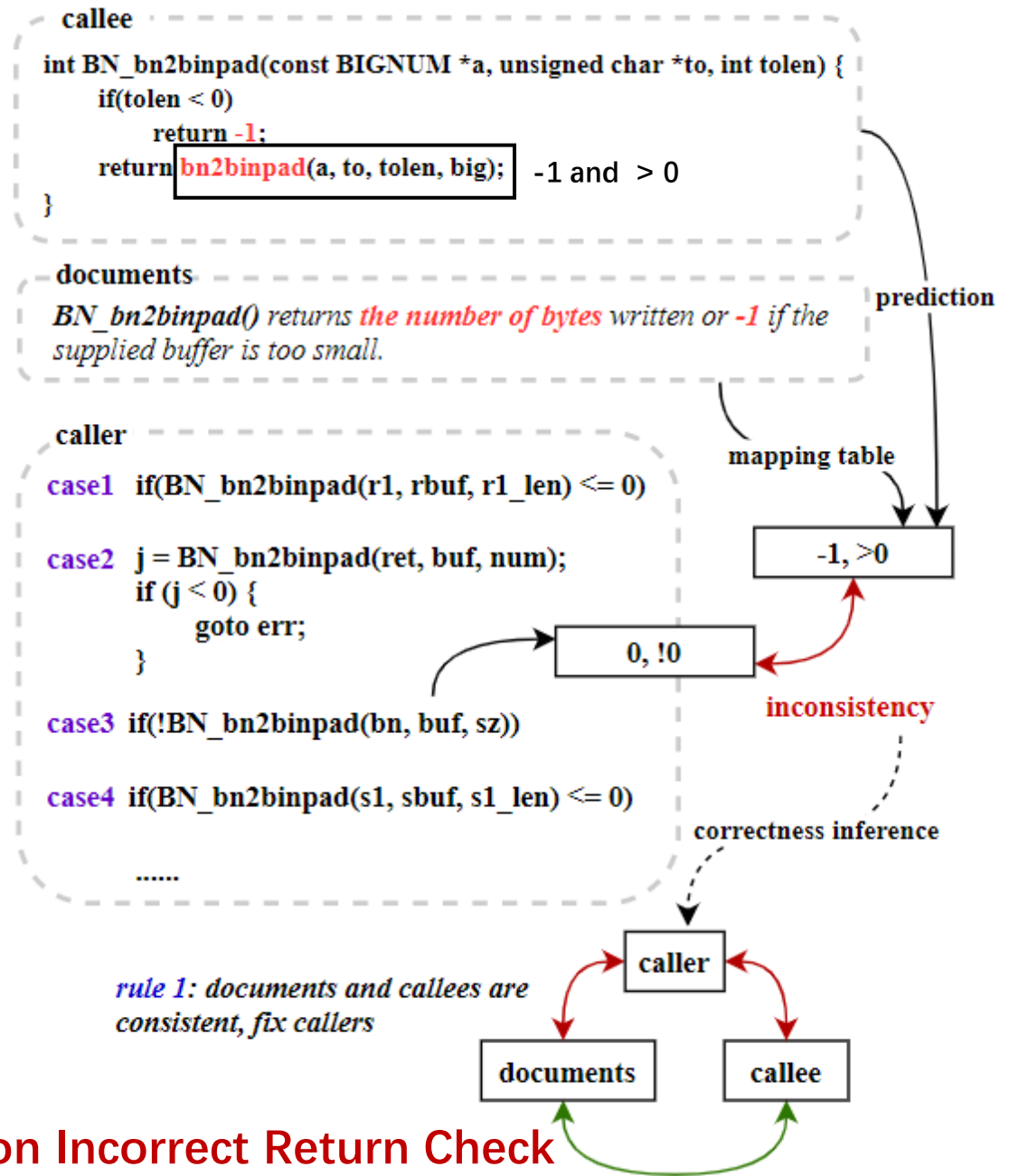
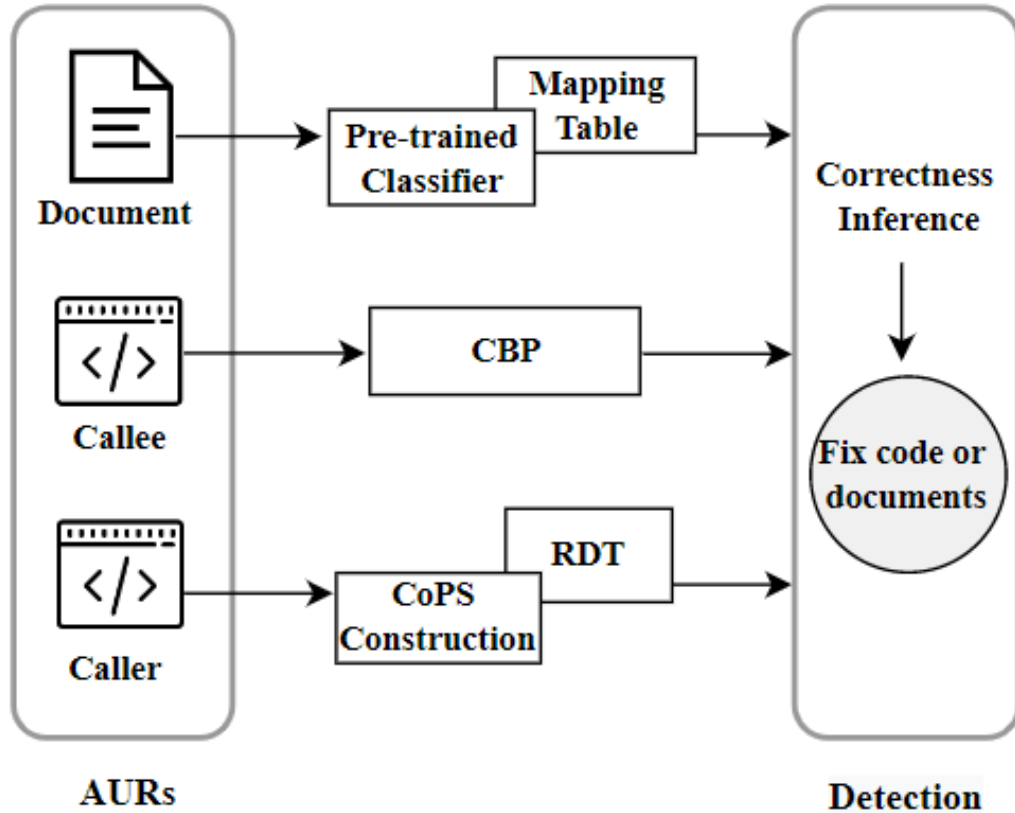
Document

Callee

Caller

3.1 Our Solution

★ API Usage Reference (AUR)

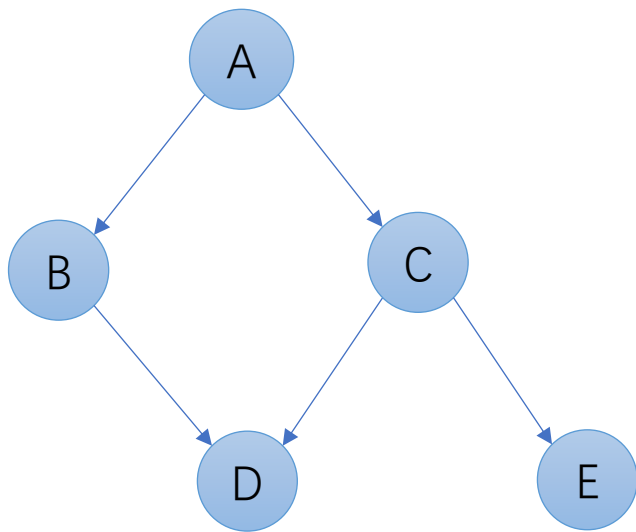


Focusing on Incorrect Return Check

3.2 Analysis of Callee

- ★ Challenge :1. Long call chain
2. Return statement location

➔ heavy analysis



Topological sorting
D E B C A

```
1 int cms_main() {  
2     /* ... 946 lines of code ... */  
3     ret = SMIME_write_CMS(out, cms);  
4     if (ret <= 0) {  
5         ret = 6;  
6         goto end;  
7     }  
8     ret = 0;  
9 end:  
10     /* ret equals to 0 or 6 */  
11     return ret;  
12 }
```

Context-sensitive
Backtrace Prediction

3.2 Analysis of Documents

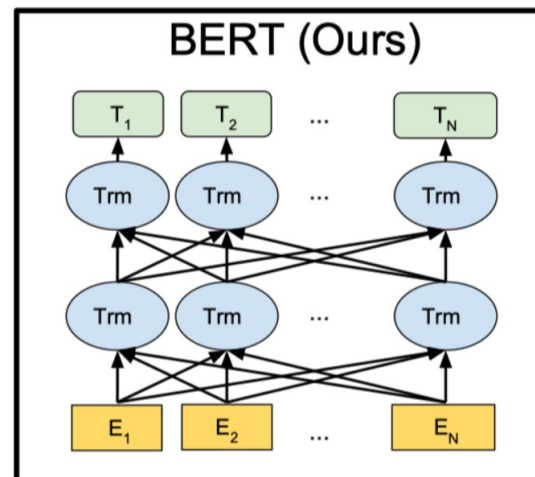
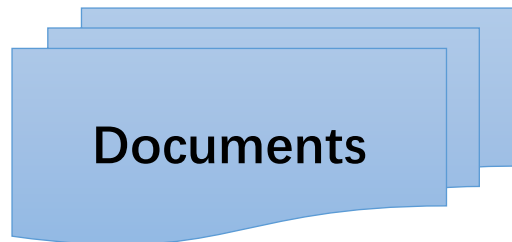
- ★ Challenge :
1. Contain sentences that are irrelevant to the return values.
 2. Documents may describe the return values in natural language.



extract return values from documents

Filter Out Irrelevant Sentences

Mapping



Word	Range
nonzero	$(-, 0) \cup (0, +)$
zero	0
length,size	$(0, +)$
negative	$(-, 0)$

3.3 AUR should be modified

★ Challenge : Which AUR should be modified facing inconsistency?

Table 3: Rules of Correctness Inference

Consistency Check			Modified Subject
Caller	Callee	Document	
X	✓	✓	Caller
✓	✓	X	Document
X	✓	/	Caller
	Others		Manual Check

4 Experiments – Effectiveness

Table 4: Effectiveness of AURC.

Codebase	Inconsistency Detection				Inconsistency Type				Running Time (s)			
	Report	Code/True	Doc/True	Acc	Rule 1	Rule 2	Rule 3	Rule 4	Classification	CBP	CoPS	Detection
OpenSSL	534	424/403	110/83	0.910	178	67	135	106	22.80	222	71	0.55
libzip	2	2/2	0/0	1	0	0	2	0	1.47	4	1	0.003
libwebsockets	8	0/0	8/8	1	0	8	0	0	1.75	34	8	0.047
GnuTLS	35	22/22	13/8	0.857	0	8	20	2	3.29	63	21	0.19
curl	2	2/2	0/0	1	0	0	2	0	12.65	46	11	0.064
mpg123	7	5/5	2/2	1	0	1	5	1	1.71	13	2	0.029
httpd	20	0/0	20/16	0.800	0	12	0	4	8.13	172	32	0.077
libgit2	129	46/37	83/73	0.852	5	46	31	28	14.02	57	16	0.155
libxml2	106	60/51	46/29	0.754	41	29	5	5	26.36	69	13	0.12
net-snmp	14	9/7	5/5	0.857	5	4	2	1	4.37	58	18	0.087
Average	–	–	–	–	–	–	–	–	9.65	74	19	0.135
All	857	570/529	287/224	0.879	229	175	202	147	–	–	–	–

False Positive: 12.1%

False Negative: 9.1%

529 code bugs 224 document defects

4 Experiments – Components

Table 7: Performance of classifier.

	Group1		Group2		Group3	
	<i>Acc</i>	<i>Recall</i>	<i>Acc</i>	<i>Recall</i>	<i>Acc</i>	<i>Recall</i>
Group1	99.5%	99.2%	89.5%	82.3%	95.2%	91.8%
Group2	90.8%	98.8%	99.9%	99.8%	94.8%	94.8%
Group3	97.4%	96.1%	92.5%	86.0%	99.9%	100%

The average accuracy and recall are **95.5%** and **94.3%**, respectively.

Performance of Classifier

$$Cov = \frac{LC(\text{return statement}) - LC(\text{CBP finishes})}{LC(\text{return statement})}$$

= 12%

88% of code does not need to be analyzed

$$PathRate = \frac{NumberOfPaths(\text{replacement})}{NumberOfPaths(\text{no replacement})}$$

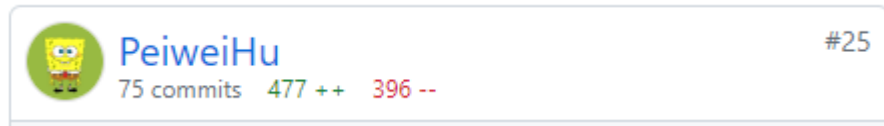
= 0.06%

CBP can save the analysis of 99.94% paths

Performance of CBP

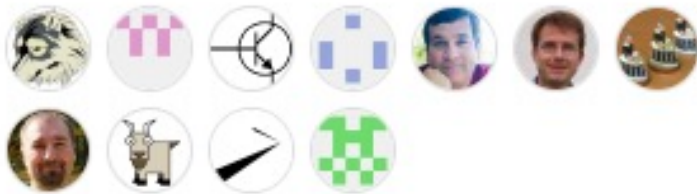
5 Practical Effectiveness

- ★ **236** code patches and **103** document patches have been merged by maintainers.
- ★ We rank **25/810** in OpenSSL contributors with the help of findings of AURC
- ★ We get positive feedback from maintainers.



PeiweiHu #25
75 commits 477 ++ 396 --

Contributors 810



+ 799 contributors



✓ paulidale approved these changes on Jan 6, 2022

paulidale left a comment

Nice work, approved with the two style nits adjusted.

tmshort commented on Jun 2, 2022

Pushed to master and openssl-3.0. Thank you!

6 Summary

- ★ We propose a new method to detect defects in **both code and documents**. (cross-check three AURs)
- ★ We propose techniques to extract information from AURs including documents, callees, callers.
- ★ We test our prototype on real-world codebases. 236 code patches and 103 document patches have been merged by maintainers.



中国科学院
CHINESE ACADEMY OF SCIENCES



中国科学院 信息工程研究所
INSTITUTE OF INFORMATION ENGINEERING, CAS

AURC: Detecting Errors in Program Code and Documentation

Thanks for your attention!