

Stack Overflow Meets Replication: Security Research Amid Evolving Code Snippets

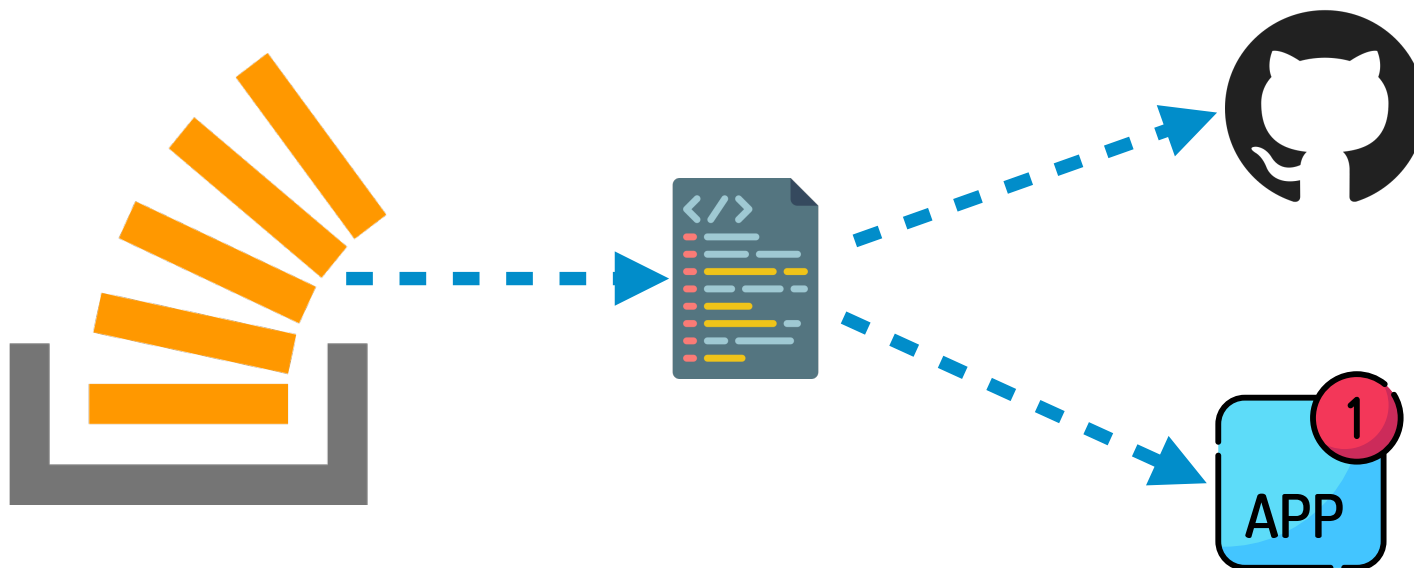
Alfusainey Jallow, Sven Bugiel

34th USENIX Security Symposium | August 2025



Stack Overflow

A widely-used resource from where developers **reuse functional code snippets** to accelerate development.





Data-driven Security Studies on Stack Overflow

Stack Overflow Considered Harmful?

The Impact of Copy&Paste on Android Application Security

Felix Fischer, Konstantin Böttinger, Huang Xiao, Christian Stransky*, Yasemin Acar*, Michael Backes*, Sascha Fahl*
Fraunhofer Institute for Applied and Integrated Security; *CISPA, Saarland University

Abstract—Online programming discussion on Stack Overflow serve as a rich source of information for developers. Available information includes...

A Study of C/C++ Code Weaknesses on Stack Overflow

Haoxiang Zhang¹, Shaowei Wang², Heng Li¹, Tse-Hsun Chen¹, and Ahmed E. Hassan, Fe...

Measuring the Effects of Stack Overflow Code Snippet Evolution on Open-Source Software Security

Alfusainey Jallow, Michael Schilling, Michael Backes, Sven Bugiel
CISPA Helmholtz Center for Information Security

You Get Where You're Looking For The Impact of Information Sources on Code Security

Yasemin Acar, Michael Backes, Sascha Fahl, Doowon Kim[†], Michelle L. Mazurek[†], Christian Stransky[†]
[†]University of Maryland, College Park

Dicos: Discovering Insecure Code Snippets from Stack Overflow Posts by Leveraging User Discussions

Hyunji Hong

Seunghoon Woo

Heejo Lee*

Snakes in Paradise?: Insecure Python-related Coding Practices in Stack Overflow

Akond Rahman, Effat Farhana, and Nasif Imtiaz
North Carolina State University, Raleigh, North Carolina
Email: aarahman@ncsu.edu, efarhan@ncsu.edu, simtiaz@ncsu.edu

The most popular question and answers posted on Stack Overflow to contain Python-related insecure coding practices. A systematic analysis on how frequently insecure code in SO answers can help the SO community identify and remove insecure Python code blocks.

▲ This [Recipe](#) provides a nice function to do what you are asking. I've modified it to use the MD5 hash, instead of the SHA1, as your original question asks

6

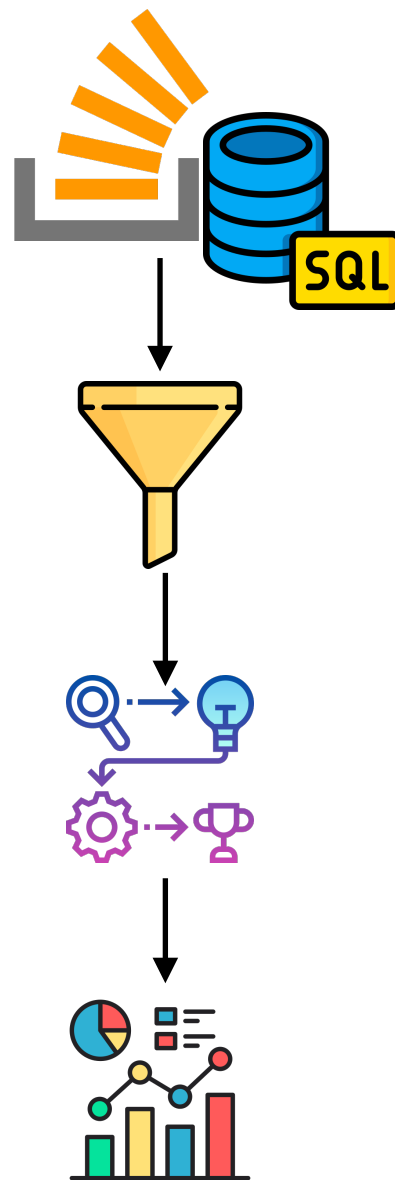
▼

```
def GetHashCodeOfDirs(directory, verbose=0):  
    import hashlib, os
```



Typical Workflow in Stack Overflow Research

Researchers typically analyze **static Stack Overflow snapshots**, applying code filtering and experimental methods to draw conclusions.



Archived Snapshot
(via archive.org)

Code Filtering
(e.g., by language)

Experiment Pipeline
(e.g., ML, NLP)

Study Results & Insights



The results of such studies are **valid
at the time they were conducted**



Code Snippets Evolve Over Time

2 refine the code to avoid resource leakage

Source Link

edit approved Jul 4, 2020 at 18:20



143 ● 1 ● 1 ● 9

Inline

Side-by-side

Side-by-side Markdown

```
char *result = (char *) malloc(size);
if(!result) {
    fputs("Memory error.\n", stderr);
    return NULL;
}

if(fread(result, 1, size, file) != size) {
    fputs("Read error.\n", stderr);
    return NULL;
}

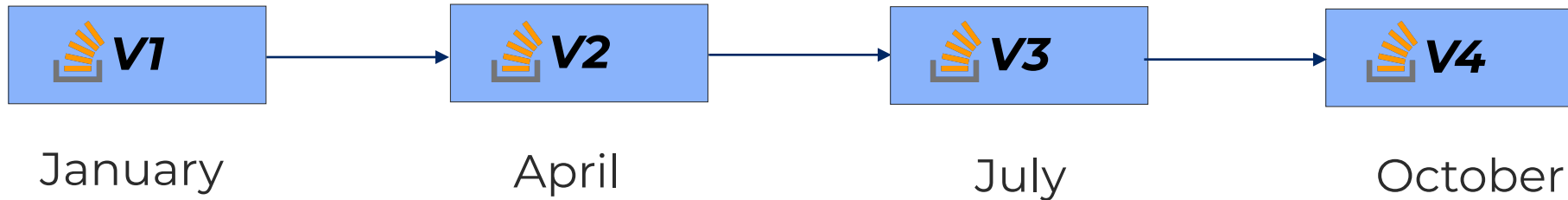
fclose(file);
```

```
char *result = (char *) malloc(size);
if(!result) {
    fputs("Memory error.\n", stderr);
    fclose(file);
    return NULL;
}

if(fread(result, 1, size, file) != size) {
    fputs("Read error.\n", stderr);
    fclose(file);
    return NULL;
}
```



Stack Overflow Dataset Release Model



Quarterly release model (every release is a *snapshot*)



Released on archive.org (“only latest version is preserved”)



Researchers study a specific snapshot → Cross-sectional studies



Cross-sectional studies analyze only a snapshot in time

→ Not ideal for analyzing evolving phenomena (c.f., stock prices or climate)



Longitudinal studies can reveal trends and recurring patterns across versions



Meta Research Questions

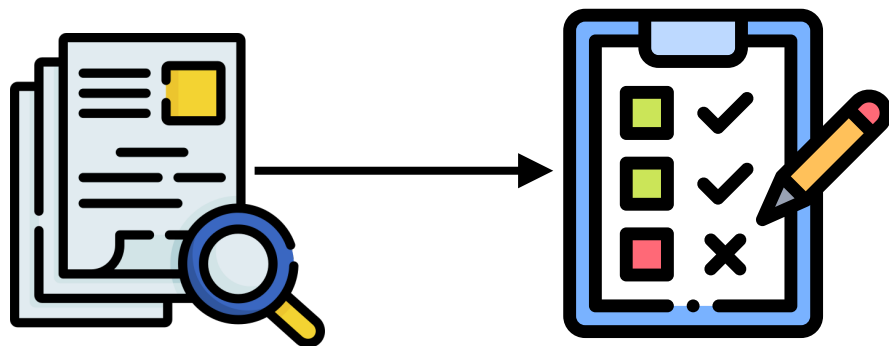
MQ1: Which aspects of Stack Overflow affect the results of prior research?

MQ2: How much do Stack Overflow code snippets and surrounding context evolve?

MQ3: How would the results of prior research differ if replicated on a newer dataset version?



Study Methodology



Systematic LR*

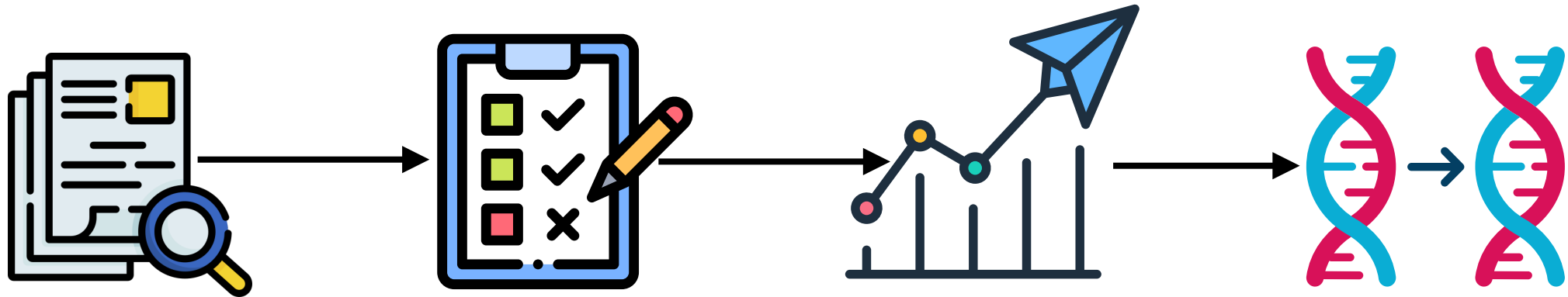
Comparison Criteria (MQ1)

		Dataset Snapshot	D1. Prog. Languages	D2. Code Scanners	D3. Code Evolution	D4. Surrounding Context	D5. Sample Size Filter	R1. Artifact Availability	R2. Language Detection	R3. Code Reuse Detect.	R4. Human-Centered
Zhang et al.	[84]	12/2018	C	✓	✓	✗	✓	✗	⊙, M	✗	✗
Hong et al.	[37]	12/2020	C	⊙, N	✓	✓	✓	○	✓	⊙, S	✗
Fischer et al.	[27]	03/2018	J	⊙, M	✗	✗	✓	⊙	N/A	✗	✗
Fischer et al.	[25]	03/2016	J	⊙, M	✗	✗	✓	⊙	N/A	⊙, P	✗
Rahman et al.	[62]	12/2018	P	⊙, SM	✗	✗	✓	⊙	✓	✗	✗
Campos et al.	[22]	12/2018	JS	✓	✗	✗	✓	⊙	✓	✓	✗
Verdi et al.	[78]	09/2018	C	✗	✗	✗	✓	⊙	✓	✓	✗
Selvaraj et al.	[68]	01/2022	C	✓	✓	✗	✓	⊙	⊙, M	✗	✗
Acar et al.	[6]	10/2015	J	✗	✗	✗	✗	✗	N/A	✗	✓
Chen et al.	[19]	?/2018	J	✗	✗	✗	✓	✗	N/A	✗	✗
Meng et al.	[57]	08/2017	J	✗	✗	✓	✓	⊙	✓	✗	✗
Ragkhitwets	[61]	01/2016	J	✗	✓	✗	✓	⊙	✓	⊙, SI, CC	✓
Bai et al.	[11]	N/A	J	✗	✗	✗	✓	✗	N/A	⊙, M	✓
Bagherzadeh et al.	[10]	N/A	J, S	✗	✗	✓	✓	⊙	✓	✗	✗
Chen et al.	[18]	?/2018	J	⊙, M	✗	✗	✓	✗	N/A	✗	✗
Zhang et al.	[85]	10/2016	J	⊙, P	✗	✗	✓	✗	✓	✓, CC	✗

9 * Kitchenham et al. Guidelines for performing systematic literature reviews in software engineering, ESEM 2007



Study Methodology



Systematic LR*

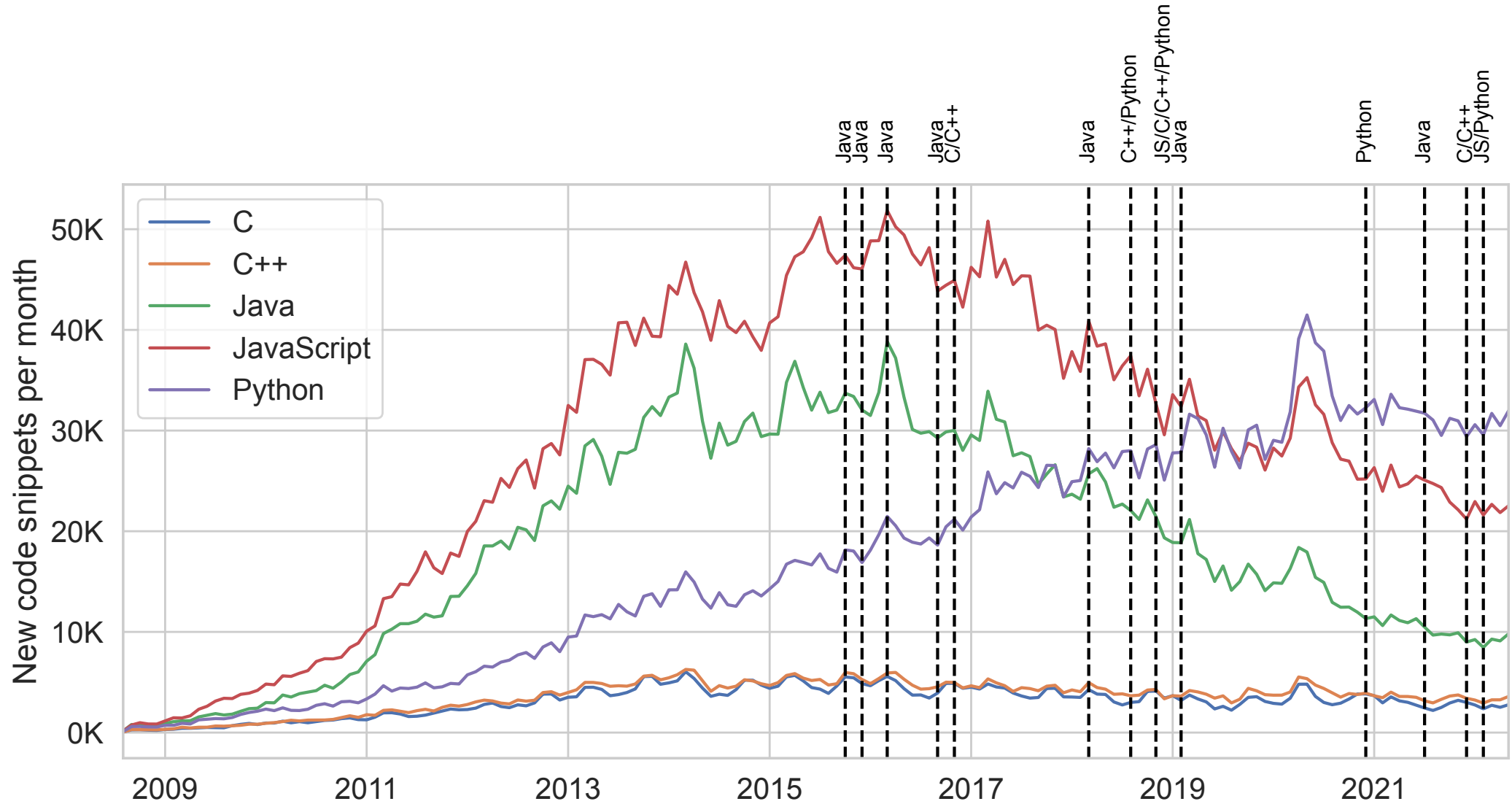
Comparison
Criteria (MQ1)

Evolution (MQ2)

Replication
(MQ3)

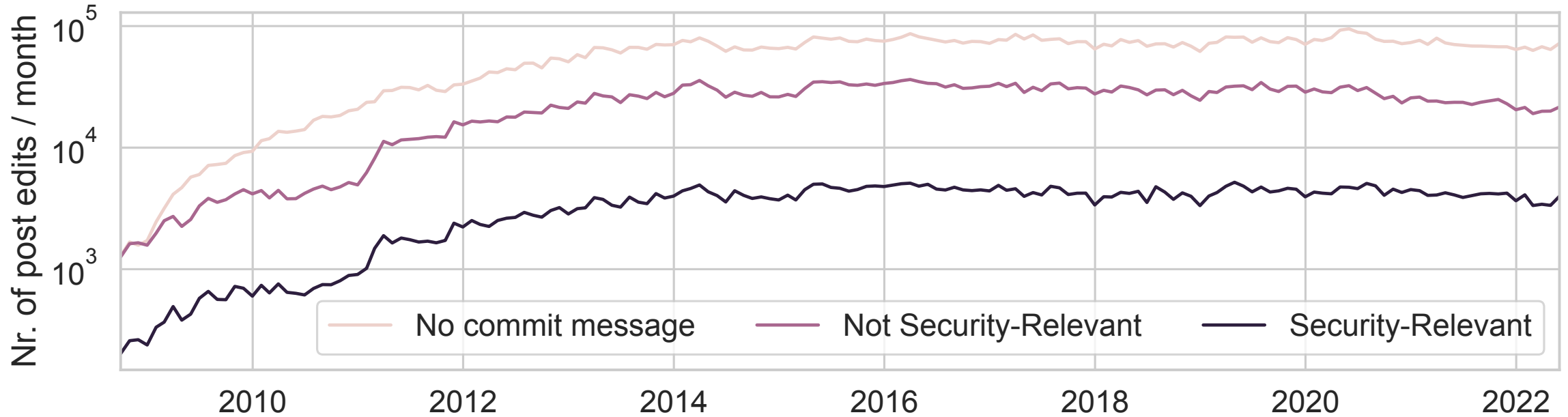


MQ2: Evolution on Stack Overflow





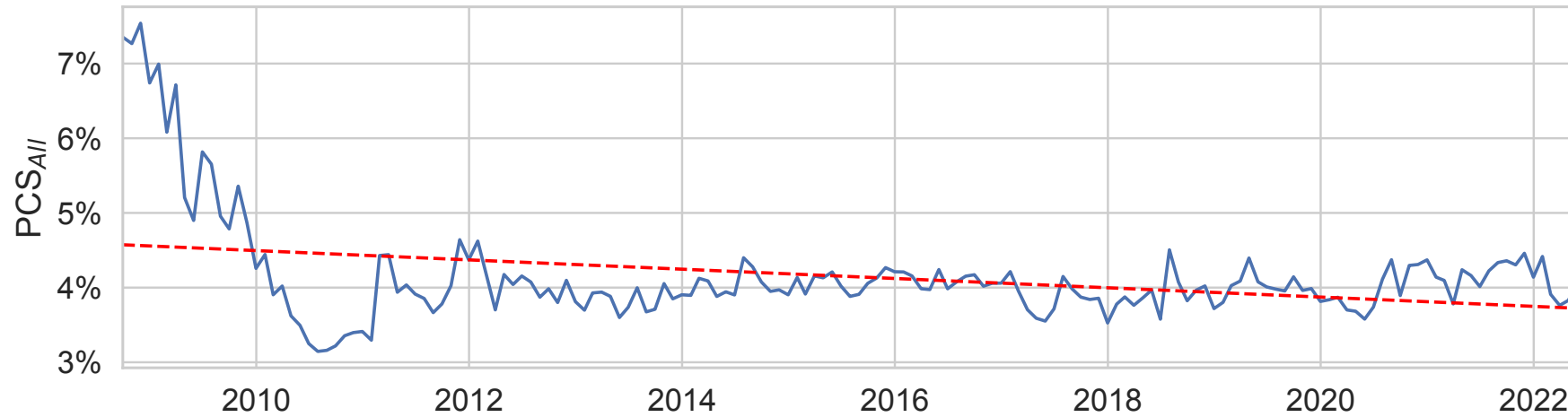
MQ2: Evolution on Stack Overflow



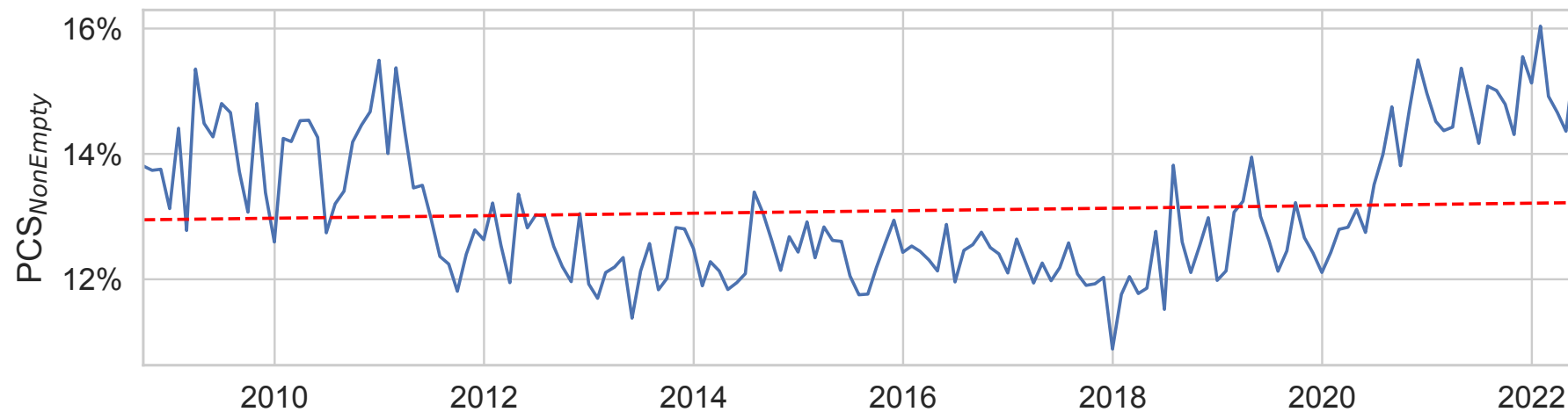
Number of monthly (30-day interval) post edits categorized by their **security relevance**.



MQ2: Evolution on Stack Overflow



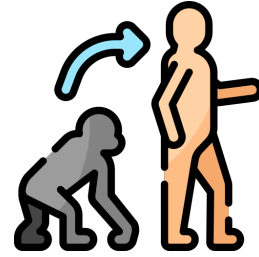
Including empty commits



Excluding empty commits



MQ2: Evolution on Stack Overflow



Programming languages **trend differently** on Stack Overflow.

Many comments **raised security-relevant** issues.

MQ3 Case Study 1: C/C++ Code Weaknesses

Zhang et al.* studied whether revisions to C/C++ snippets increase or decrease the snippets' security

- **RQ1:** What are the types of code weaknesses that are detected in C/C++ code snippets on Stack Overflow?
- **RQ2:** How do code with weaknesses evolve through revisions?
- **RQ3:** What are the characteristics of the users who contributed code with weaknesses?



$$Answer_w = 11,235, Code_w = 11,748, Version_w = 14,934$$



MQ3 Case Study 1: Replication Results

Original Results

As the number of revisions to $Code_w$ increased from 1 to 3+, the proportion of improved $Code_w$ rose from **30.1% to 41.8%**.

Authors found CWE-758 to be the **sixth** most prevalent CWE type in C/C++ code snippets with **482** instances.

Authors found **12,998** CWE instances in the latest versions of **7,481** answers.

Replication Results

As the number of revisions to $Code_w$ increased from 1 to 3+, the proportion of improved $Code_w$ only rose from **3.1% to 7.4%**.

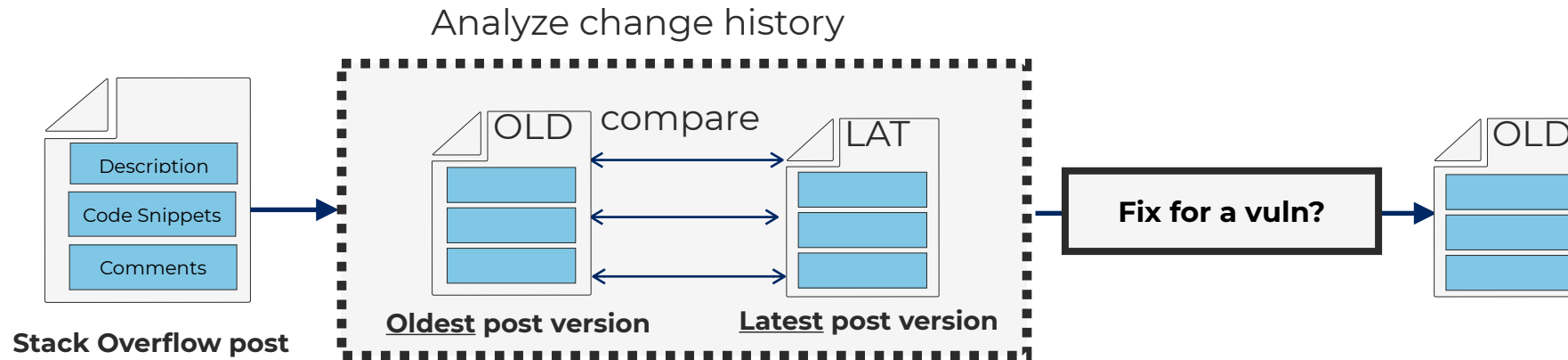
We found CWE-758 to be the **second** most prevalent CWE type in C/C++ code snippets with **10,911** instances. New **CWE-476** is now **sixth** most common.

We found **7,679** CWE instances in latest versions of **5,721** answers.



MQ3 Case Study 2: DICOS

DICOS* examines snippet revisions for changes in **security sensitive APIs**, **security-related keywords**, and **control flows**



- **RQ1:** Are older posts more likely to provide insecure code snippets?
- **RQ2:** Are accepted answer posts more secure than non-accepted answer posts?
- **RQ3:** What types of insecure code snippets were discovered?



MQ3 Case Study 2: Replication Results

Original Results

Authors discovered 12,458 (1.9%) insecure posts, with 788 (6.3%) insecure posts with all three features.

Authors reported a 91% precision, 89% accuracy and 89% recall.

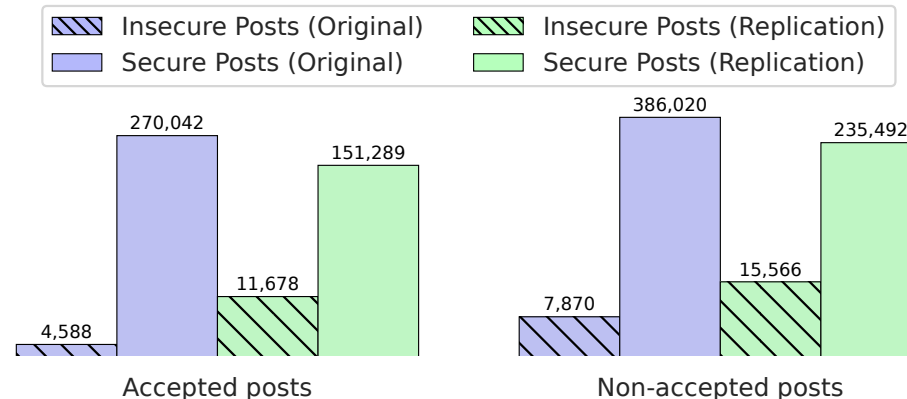
Authors found no difference between ratio of insecure posts for accepted (1.67%) and non-accepted (1.99%) answers.

Replication Results

We discovered 30,359 (2.9%) insecure posts, 4,887 (16.1%) has all three features, an increase of 155%.

We discovered 11% precision, 32% accuracy and 87% recall.

We found higher ratio of insecure posts between accepted (7.72%) and non-accepted (6.61%).





Recommendations

1. **Adopt Longitudinal Analyses with Stack Overflow Data**

- Cross-sectional snapshots limit understanding of whether security issues are persistent or transient
- Longitudinal analysis helps distinguish short-lived trends from long-term patterns
- Use Stack Overflow's versioned data to track changes and trends over time

2. **Promote Open Science Practices**

- Release code and data artifacts to enable reproducibility and replication
- Report exact software versions; use Docker or similar containers for consistency
- We commend USENIX'25/'26 for requiring artifact availability



Conclusion

- Cross-sectional studies are **valuable** but limited in capturing security trends; longitudinal analysis offers deeper insights
- Stack Overflow still remains relevant in the GenAI era, especially for complex or niche coding problems
- **Future work**
 - Inspire future replication studies for other evolving code sources (e.g., Chromium, GitHub)
 - Apply established longitudinal research methods from other disciplines to study evolving code datasets

Thank you!