UNIVERSITÄT BERN

Security in Android Applications

Master Thesis

Pascal Gadient

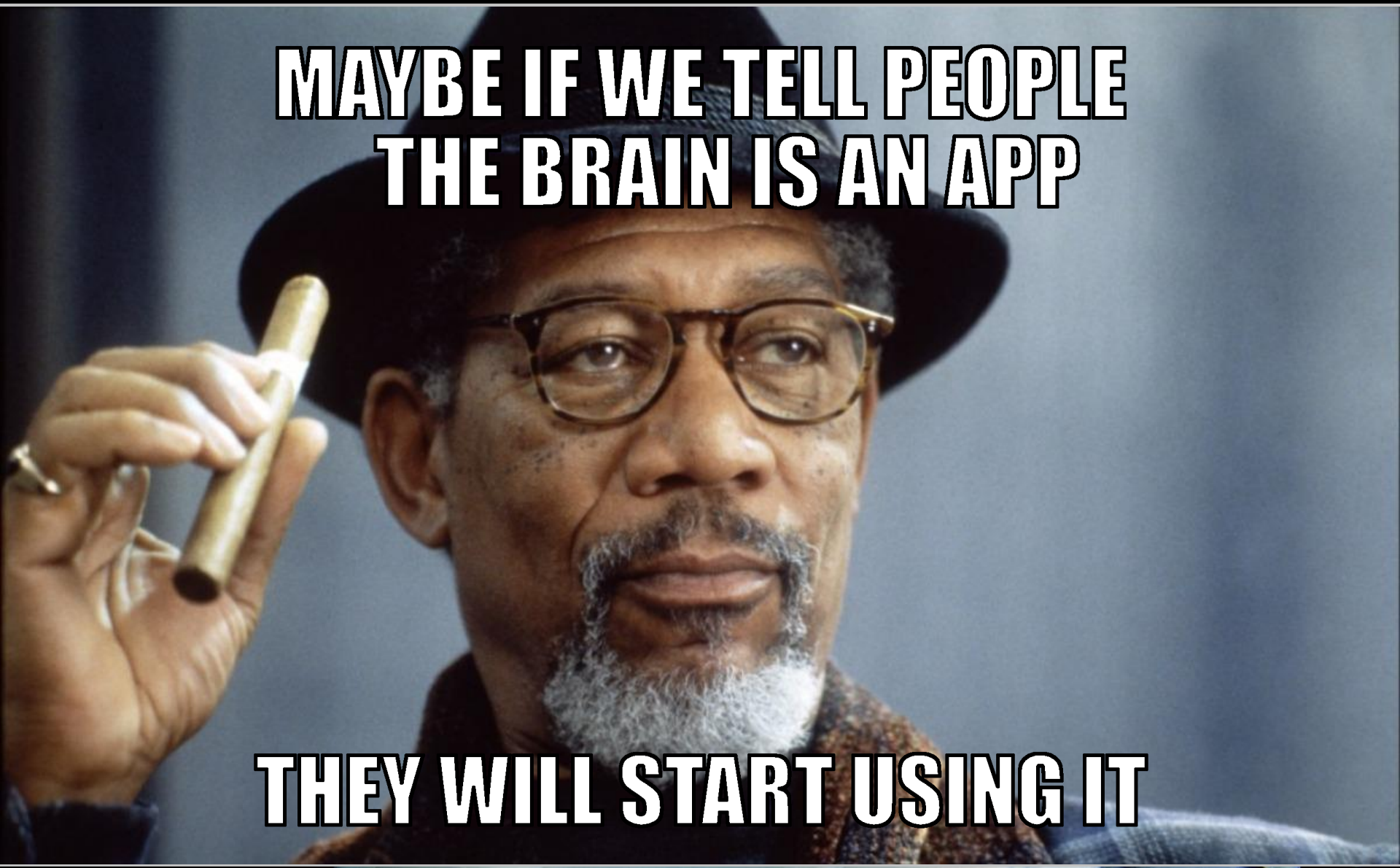
Software Composition Group
University of Bern
Switzerland

Agenda

- Introduction
- What are security code smells?
- How prevalent are they?
- Why identifying security smells is helpful?
- Conclusion

Software is Everywhere

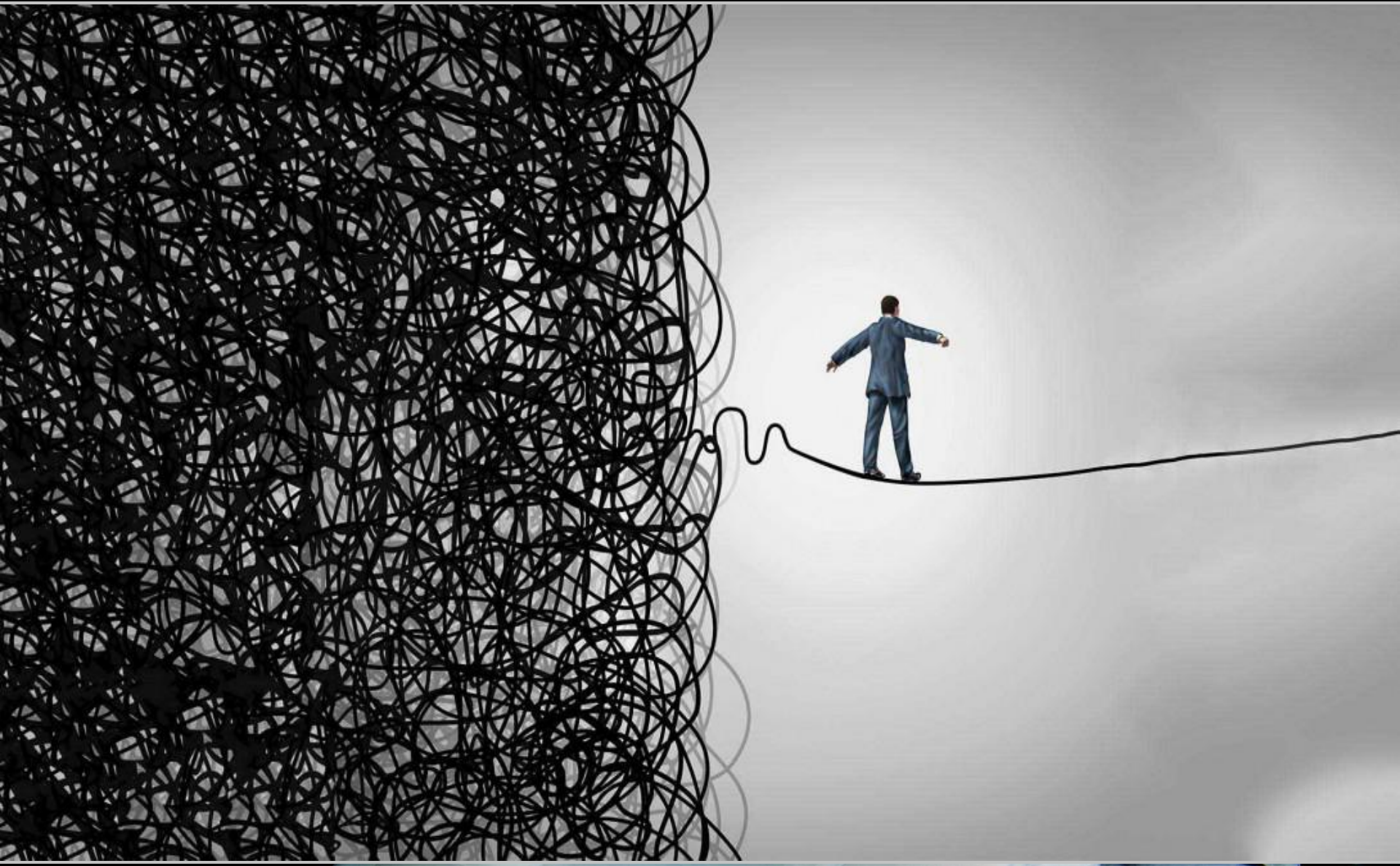
We Love Apps


MAYBE IF WE TELL PEOPLE
THE BRAIN IS AN APP
THEY WILL START USING IT

Mobile Device Addiction

Mobile Security is Vital

Software Insecurity Thrives



Security Code Smell

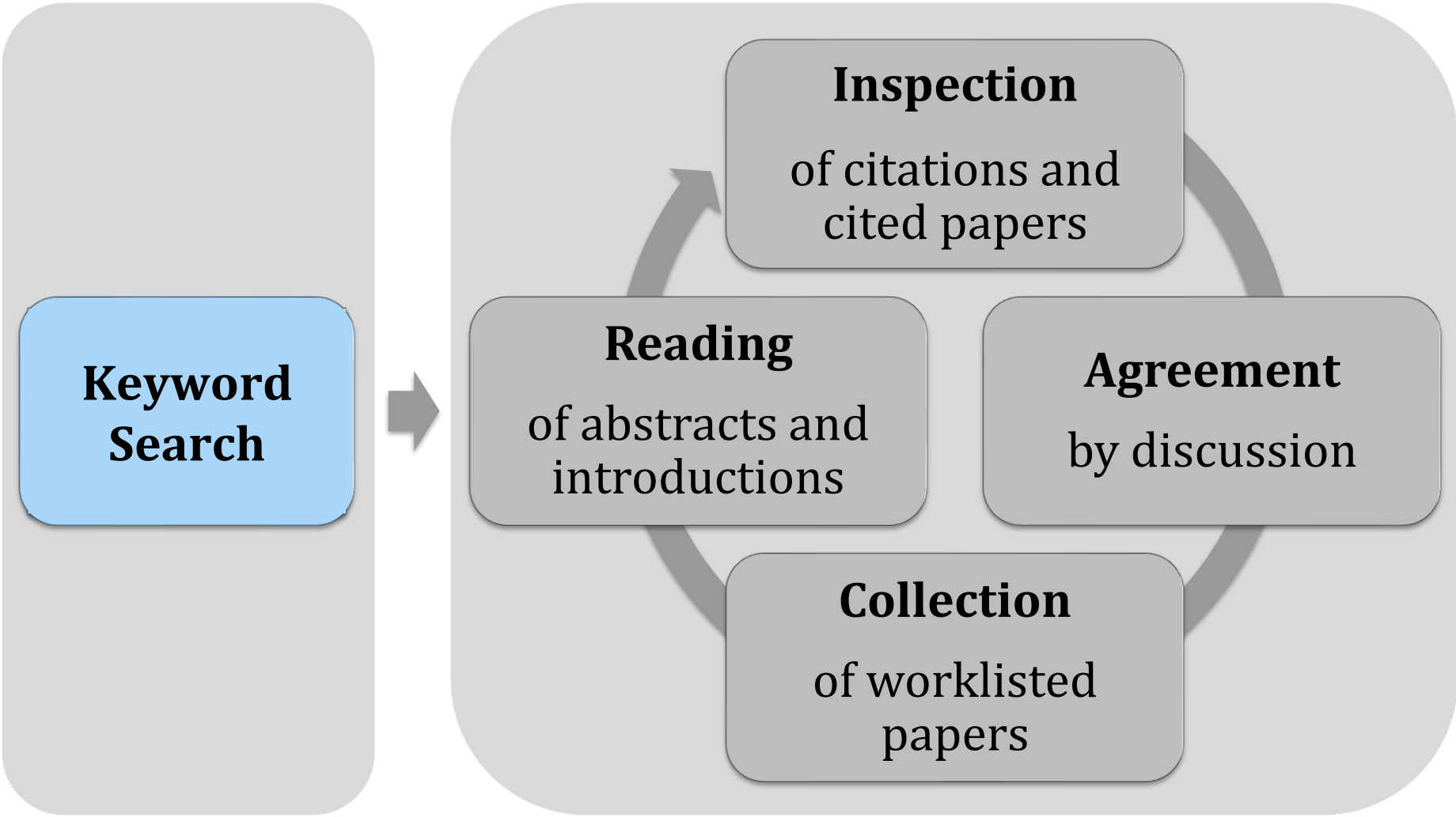
Definition:

Symptoms in the code that indicate
the prospect of security and privacy vulnerabilities

Research Goals

- RQ_1: What are the security code smells in Android apps?
- RQ_2: How prevalent are security smells in benign apps?
- RQ_3: To which extent identifying such smells facilitates detecting security vulnerabilities?

Literature Review



What are the security code smells in Android apps?

Insufficient Attack Protection

- Unreliable Information Sources
- Untrustworthy / Outdated Libraries
- Native Code
- Open to Piggybacking
- Unnecessary Permissions

Security Invalidation

- Weak Crypto Algorithm or Configuration
- Improper Certificate Use
- Unacknowledged Distribution

Broken Access Control

- Insecure Inter-Component Communication
- Unprotected System Sockets
- Custom Scheme Channel

Sensitive Data Exposure

- Insecure Storage
- Exposed Identifiers

Lax Input Validation

- Unverified JavaScript Code
- Dynamic Code Loading
- SQL Injection

How prevalent are security smells in apps?

Scope of the Study

- Random apps from AndroZoo
- Corpora size:

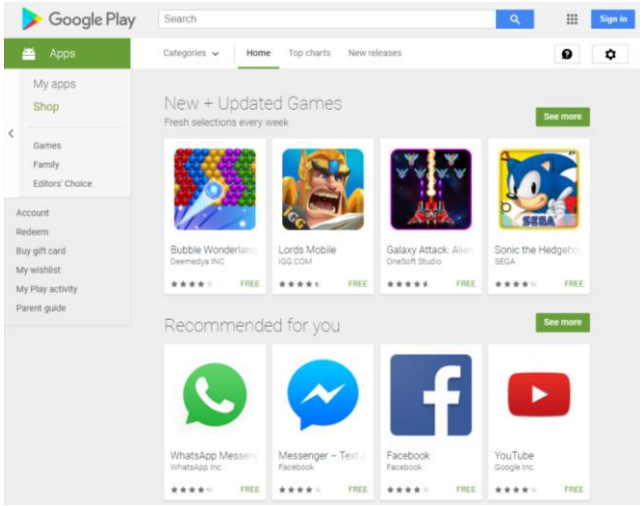
  46,000 apps

  440 GB

- Lightweight analysis

- 10 out of 28 smells analysed

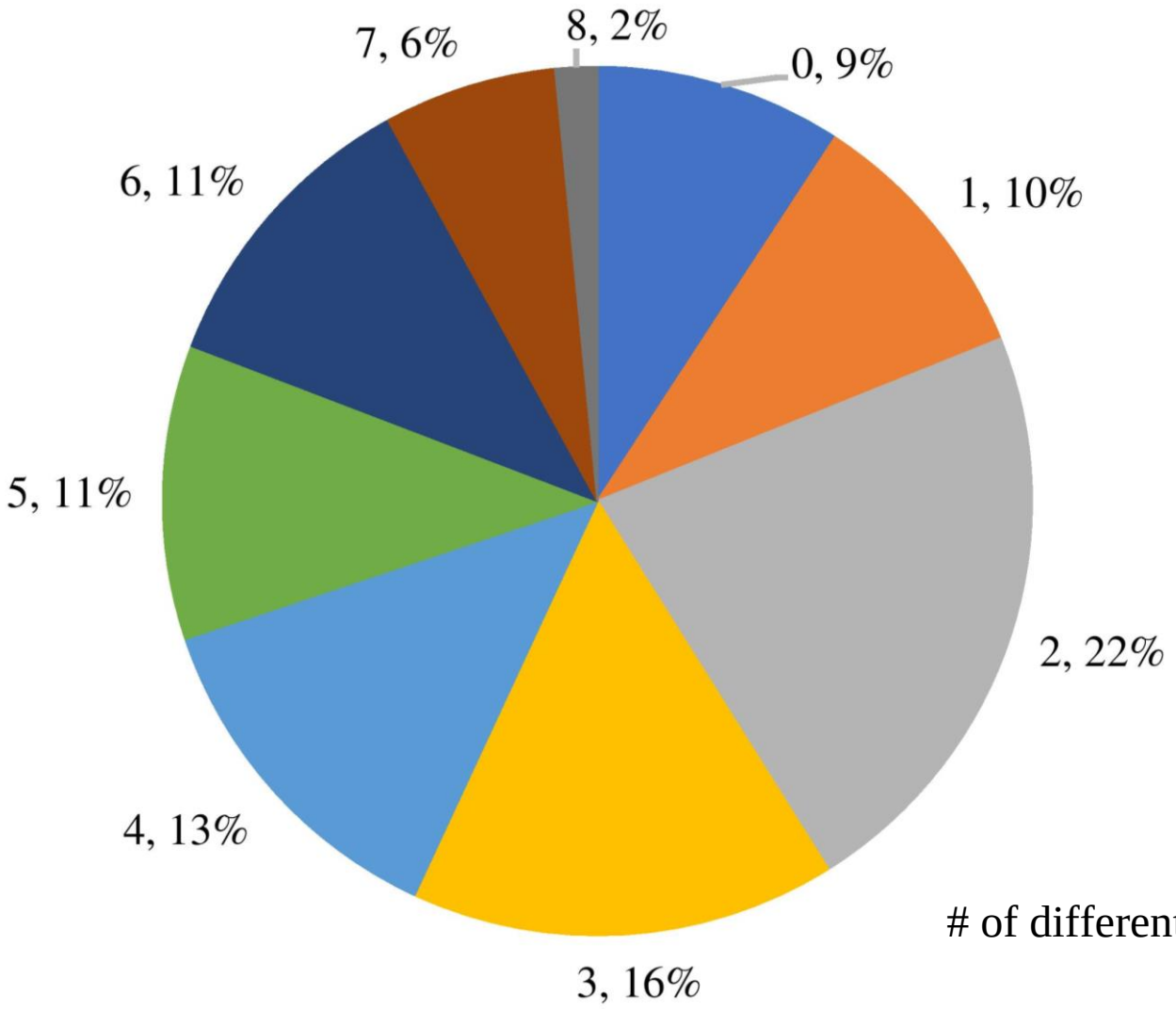
Subjects of the Study



Apktool

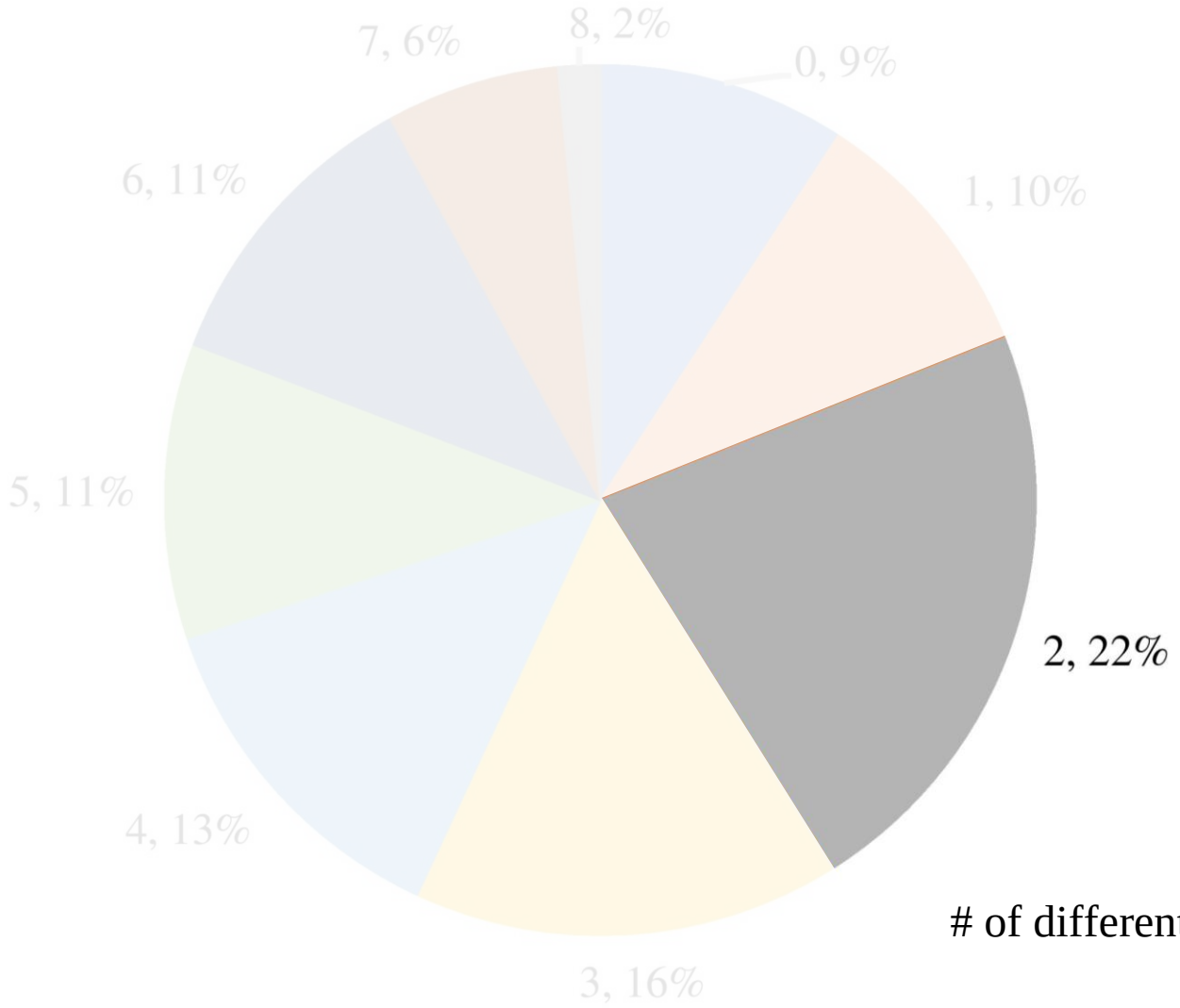
Web parser

Prevalence of Smells



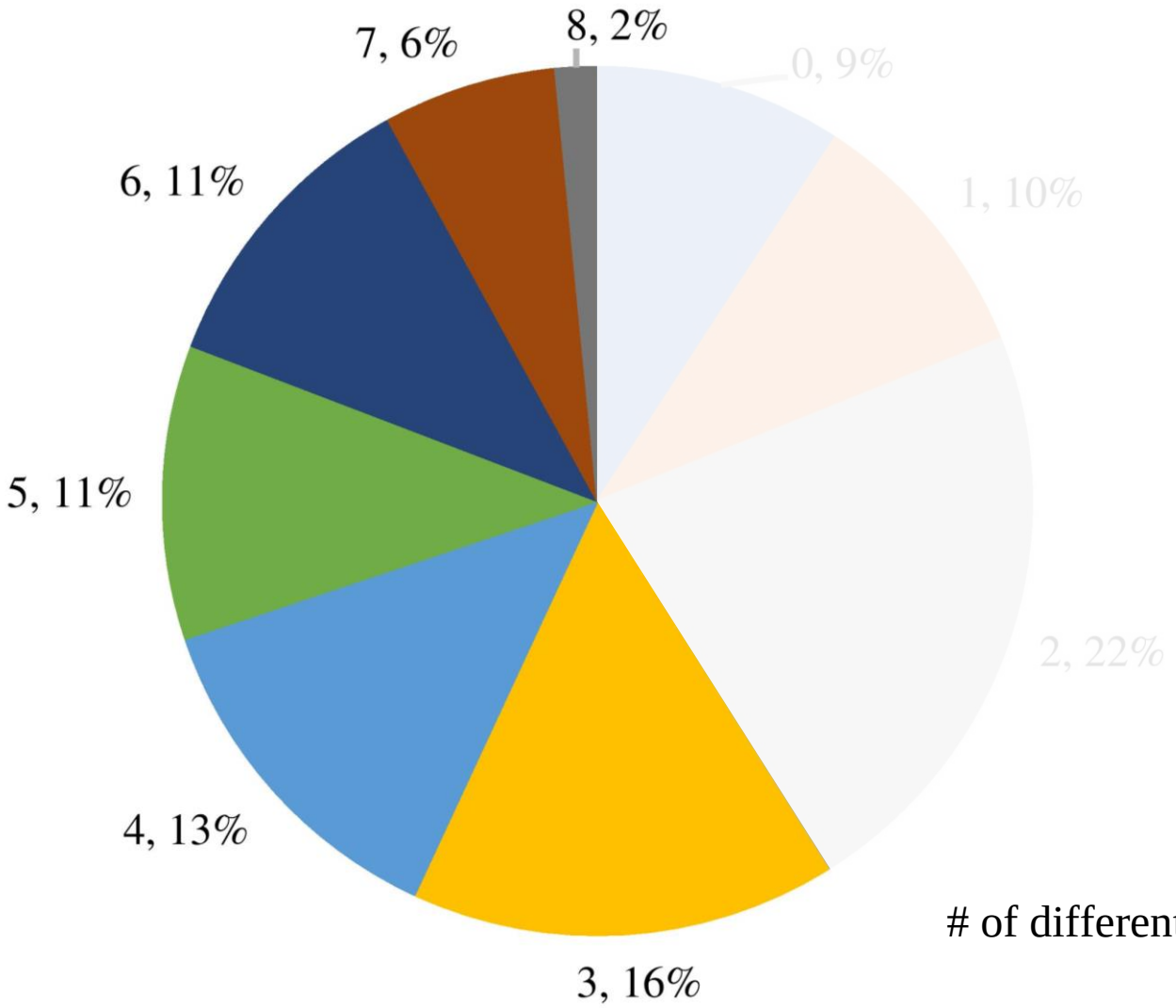
# of different smells apps suffer

Prevalence of Smells



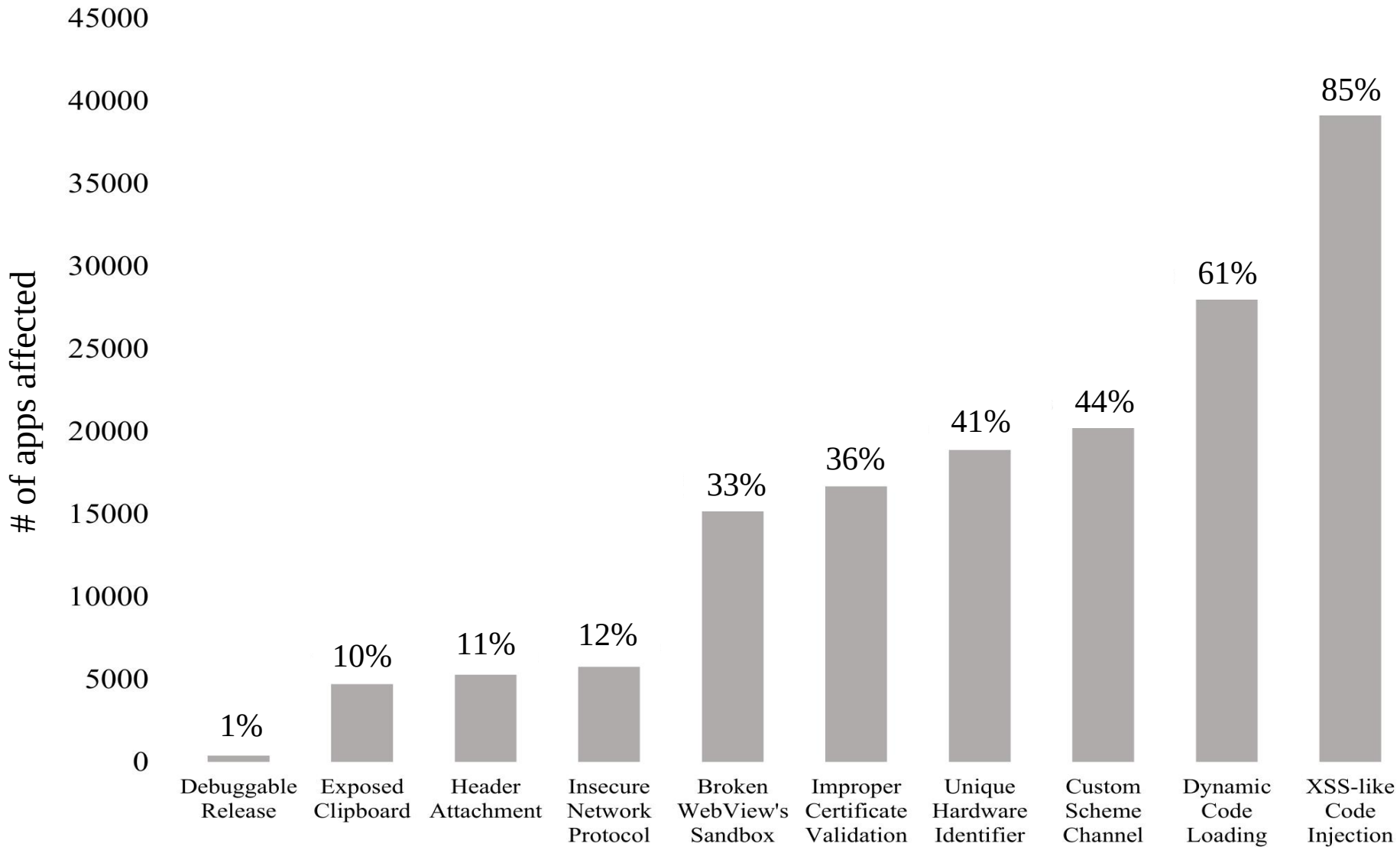
# of different smells apps suffer

Prevalence of Smells

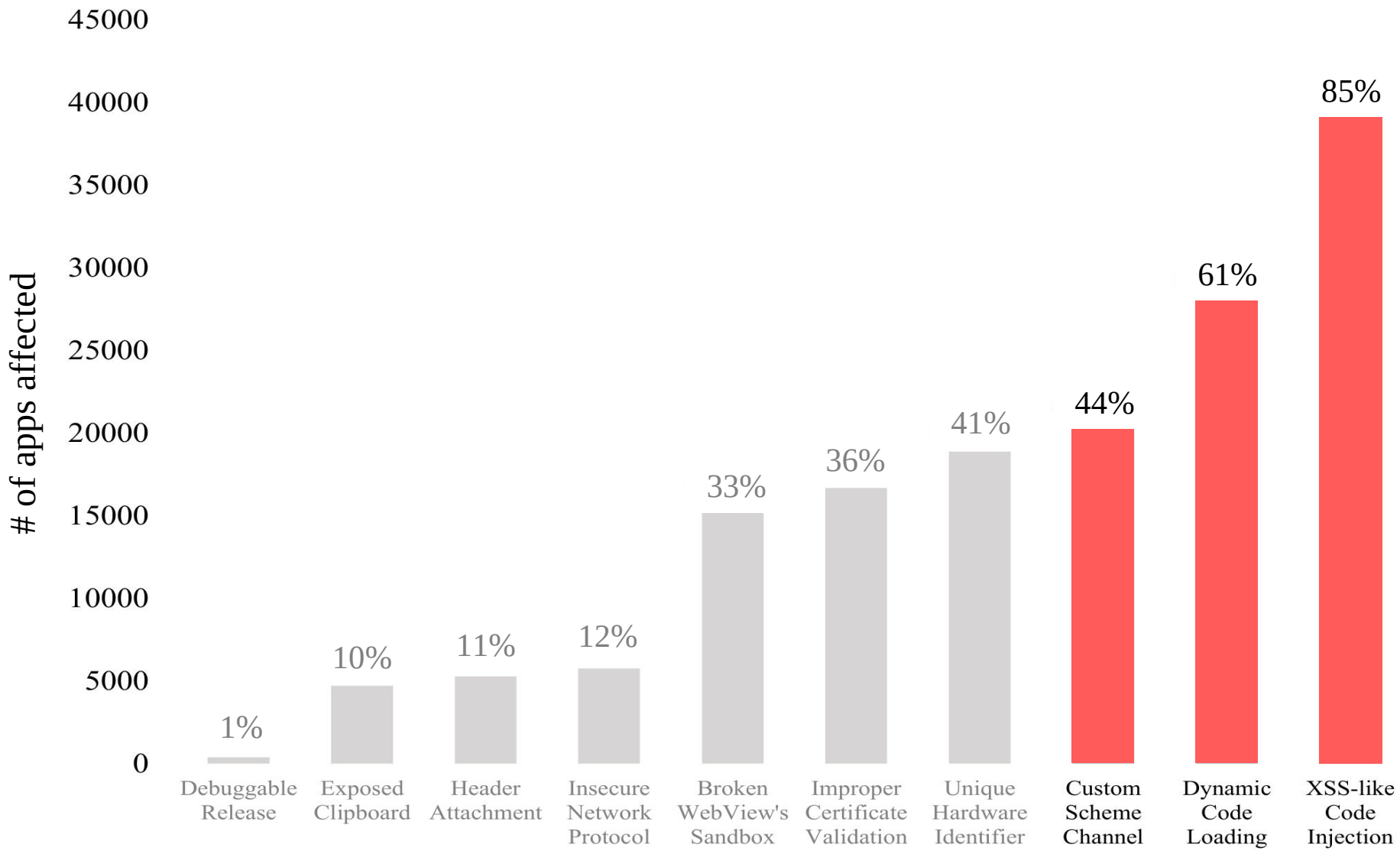


# of different smells apps suffer

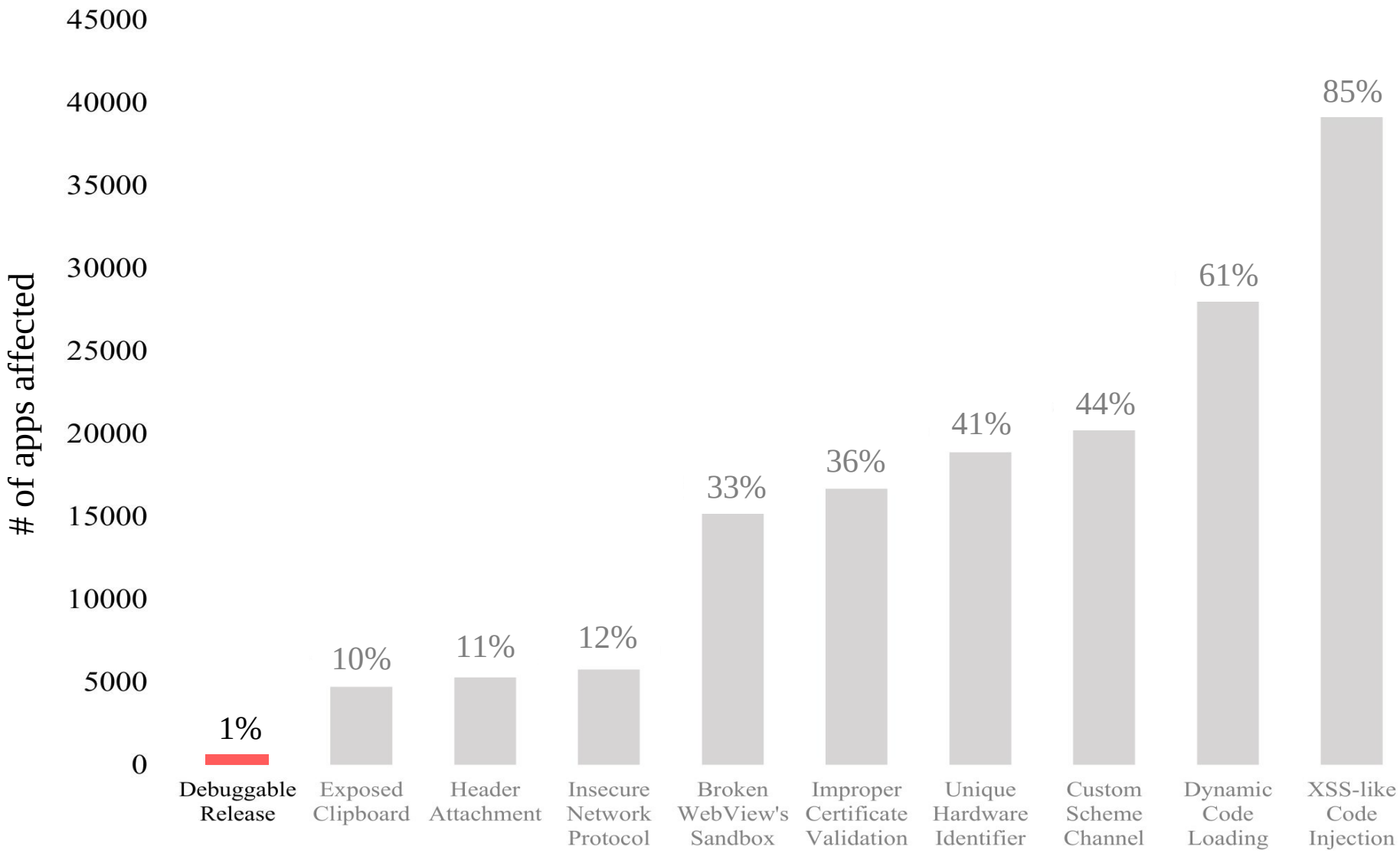
Distribution of Each Smell



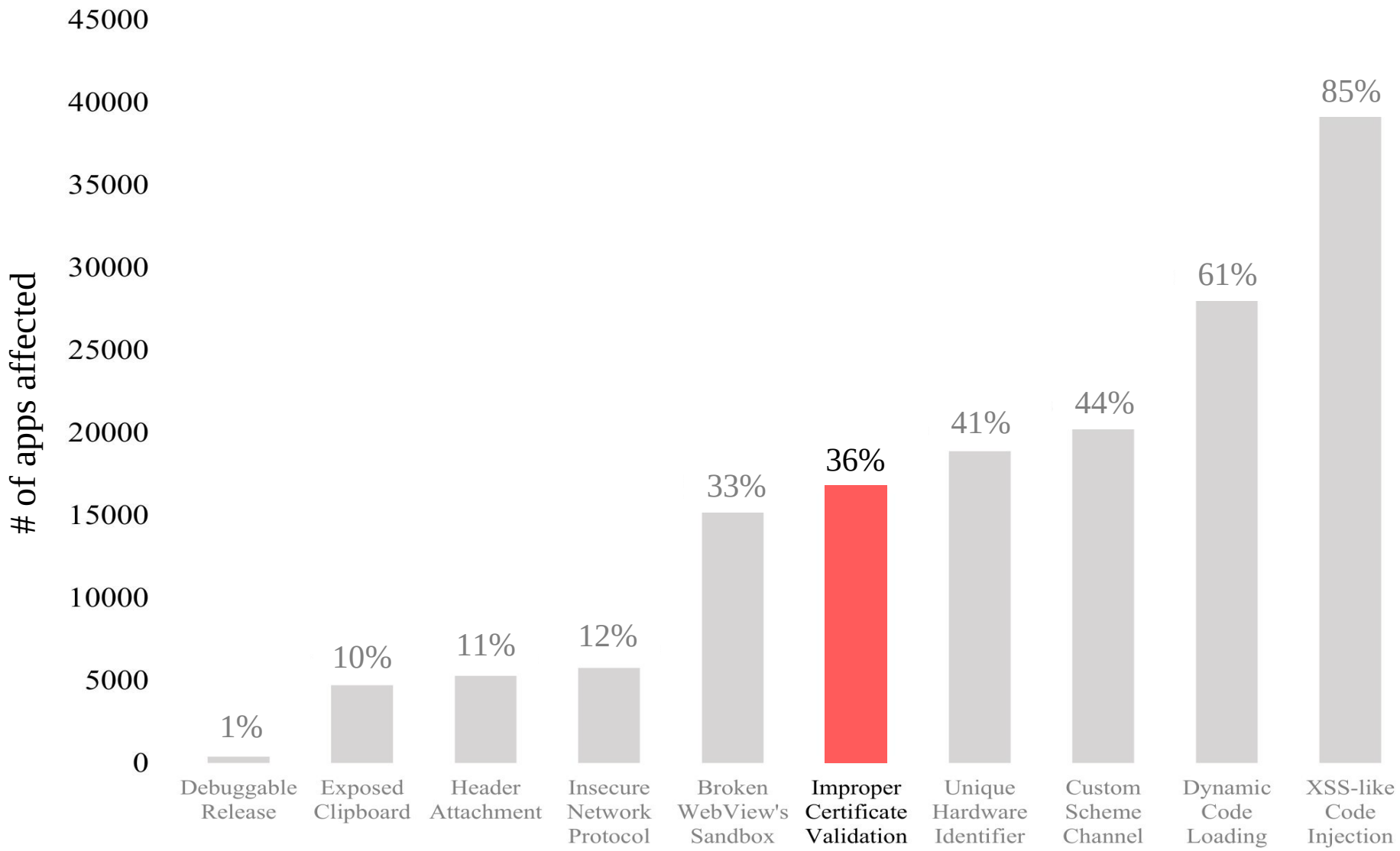
Distribution of Each Smell



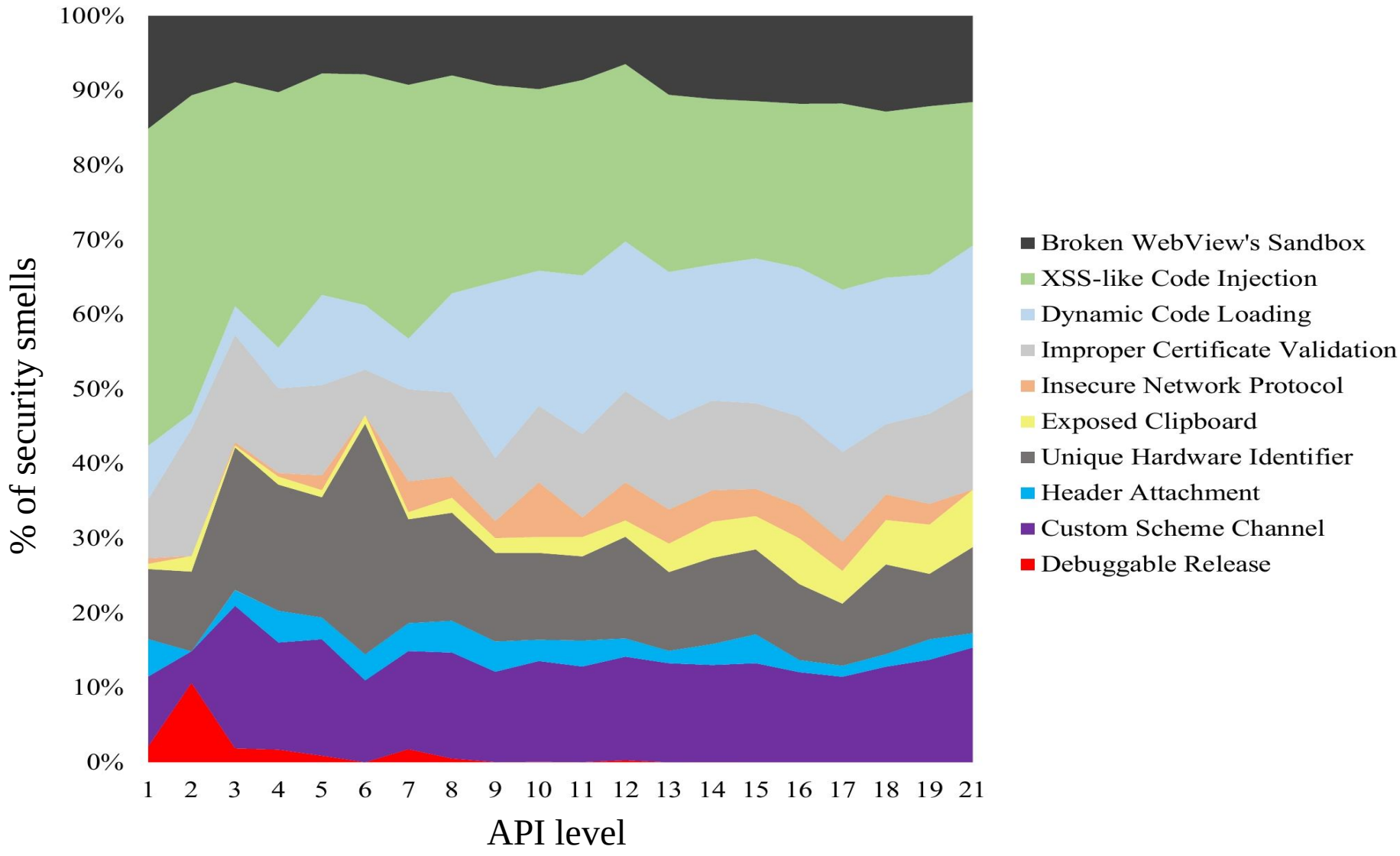
Distribution of Each Smell



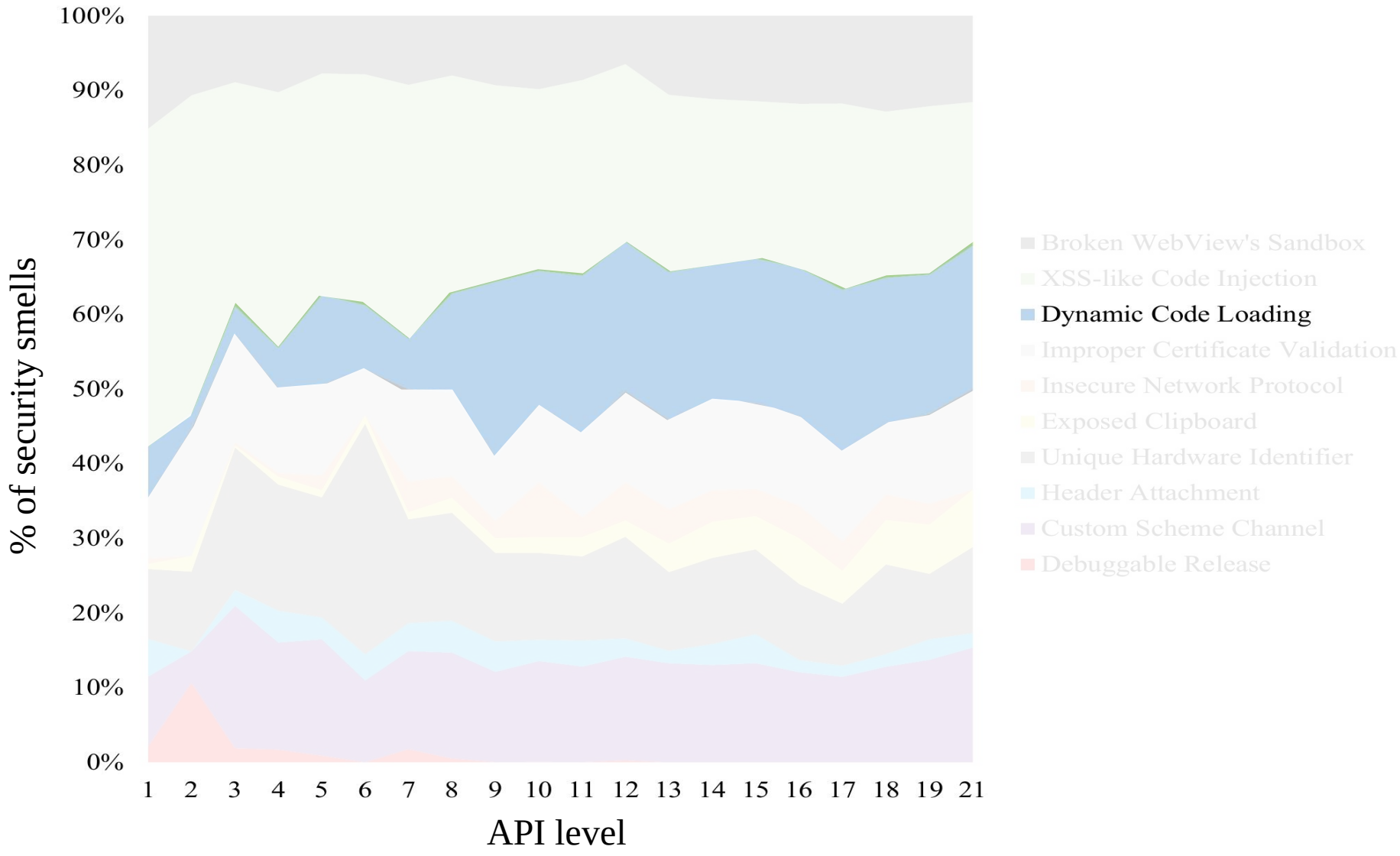
Distribution of Each Smell



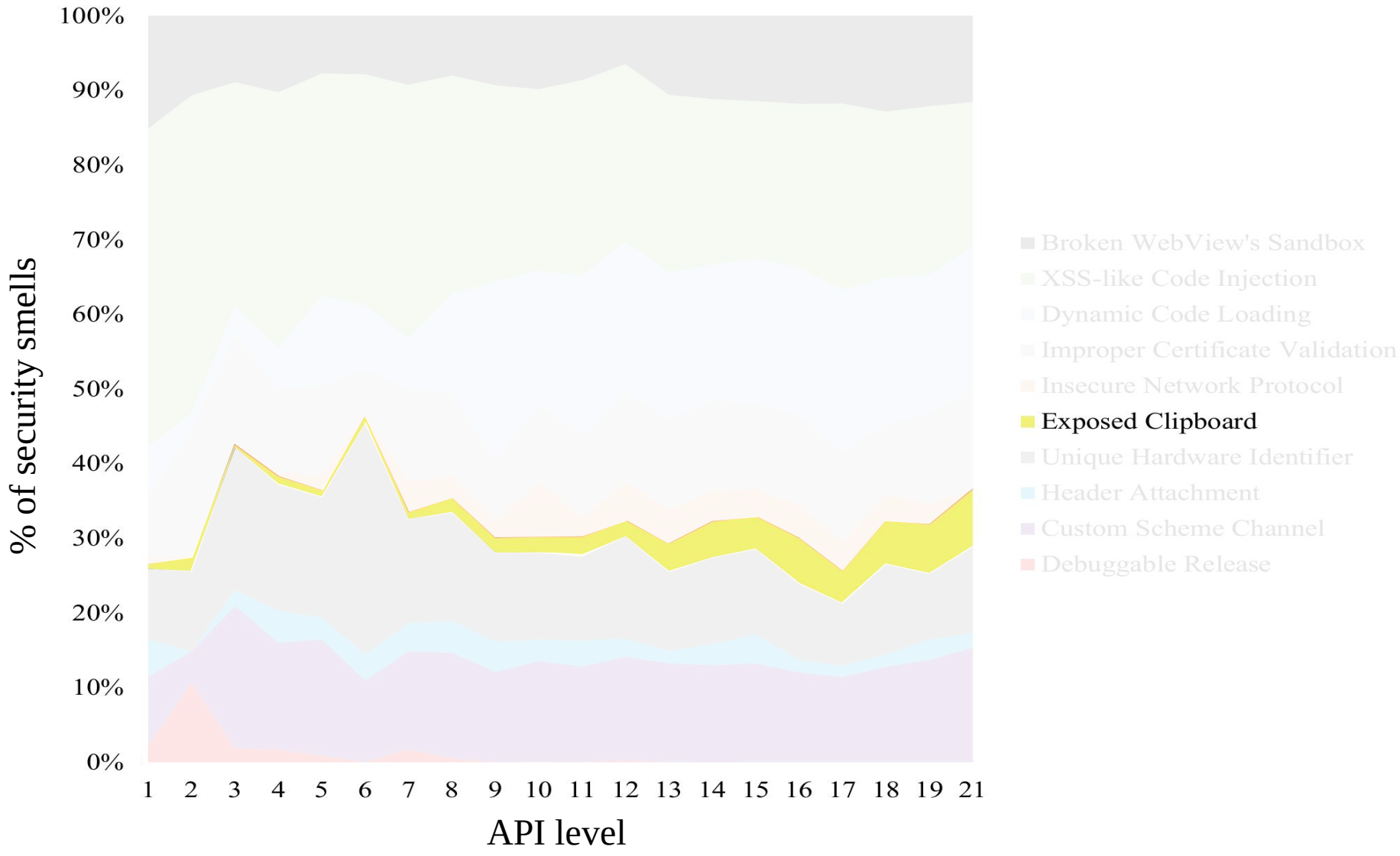
Distribution of Each Smell in API Levels



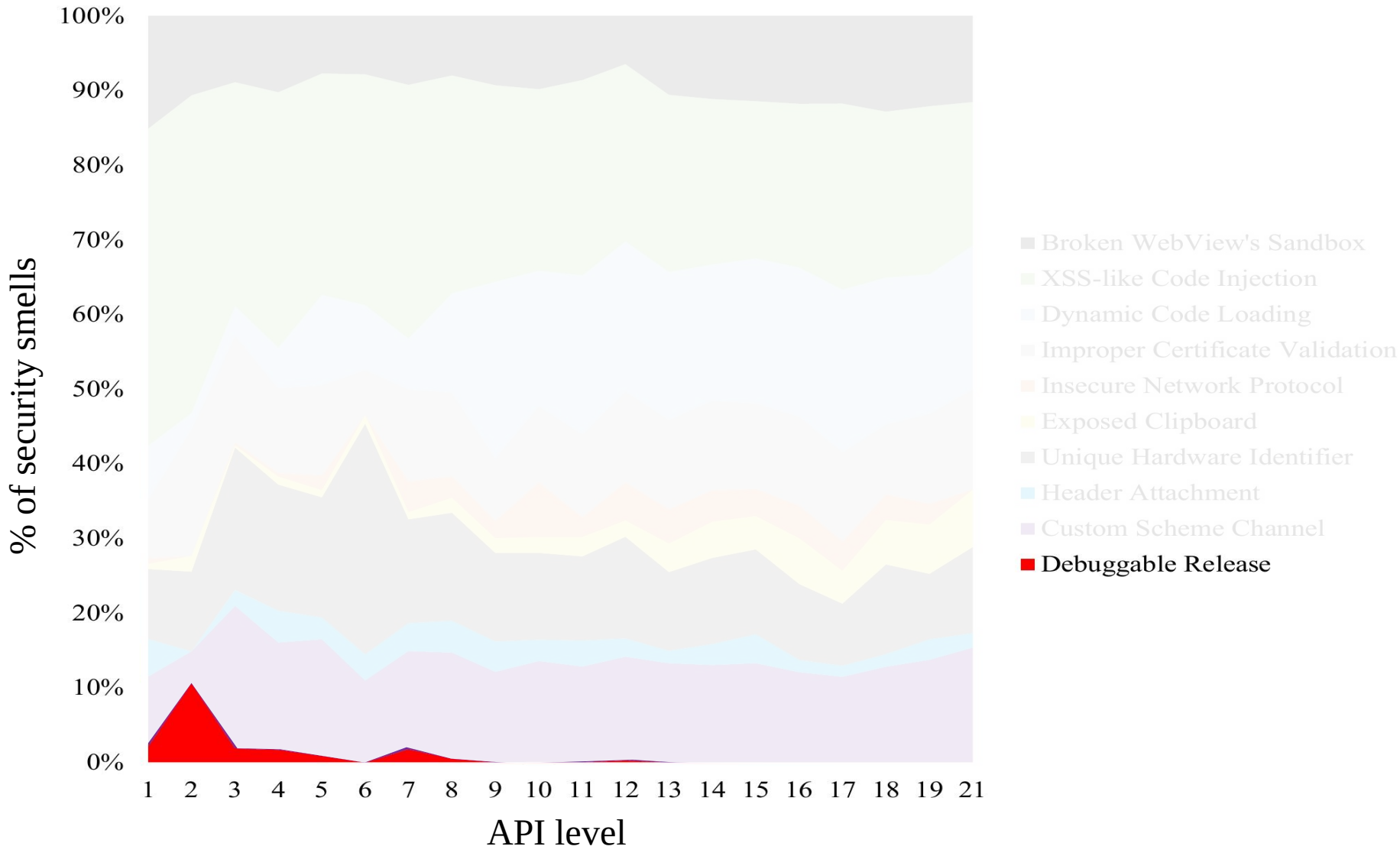
Distribution of Each Smell in API Levels



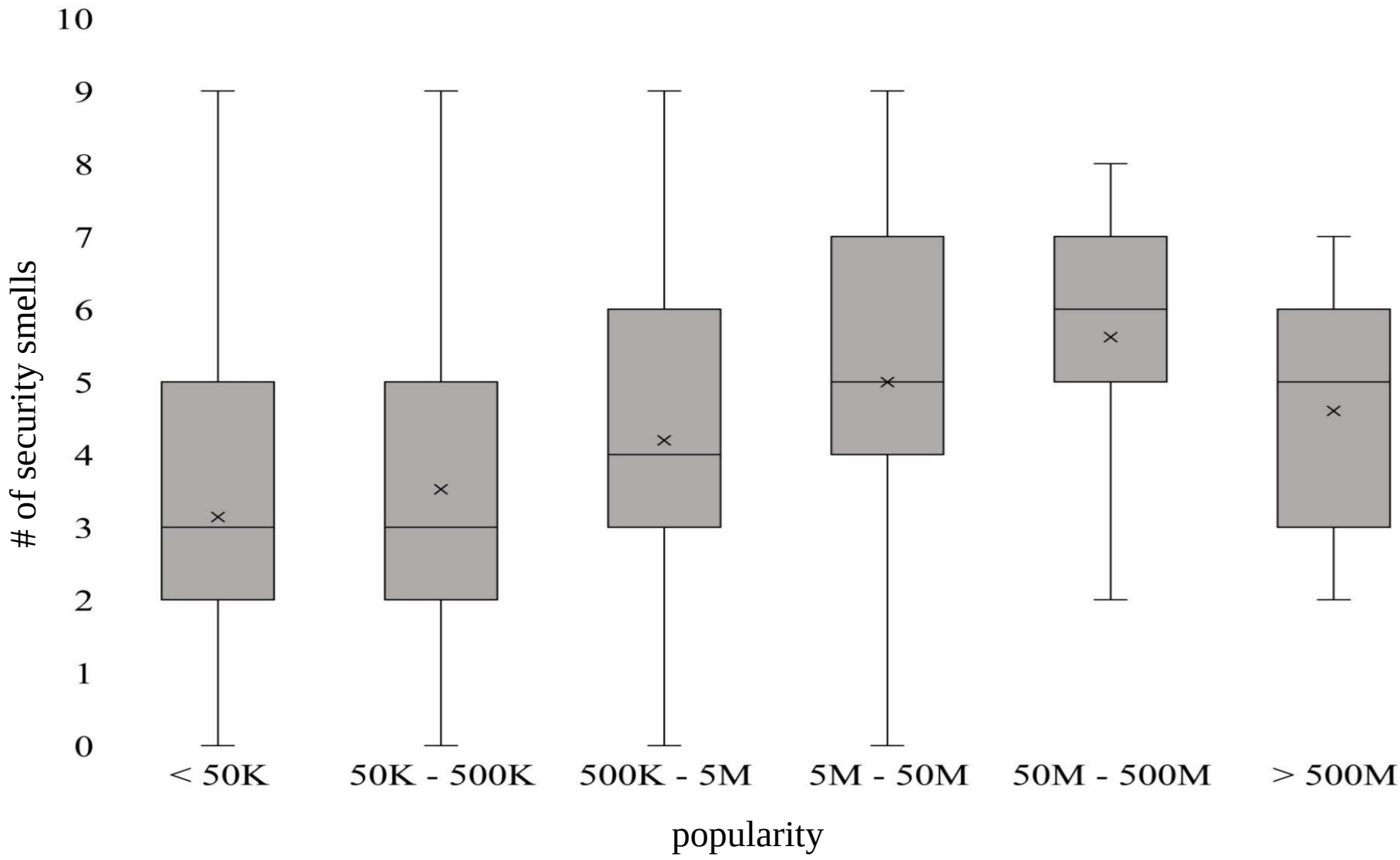
Distribution of Each Smell in API Levels



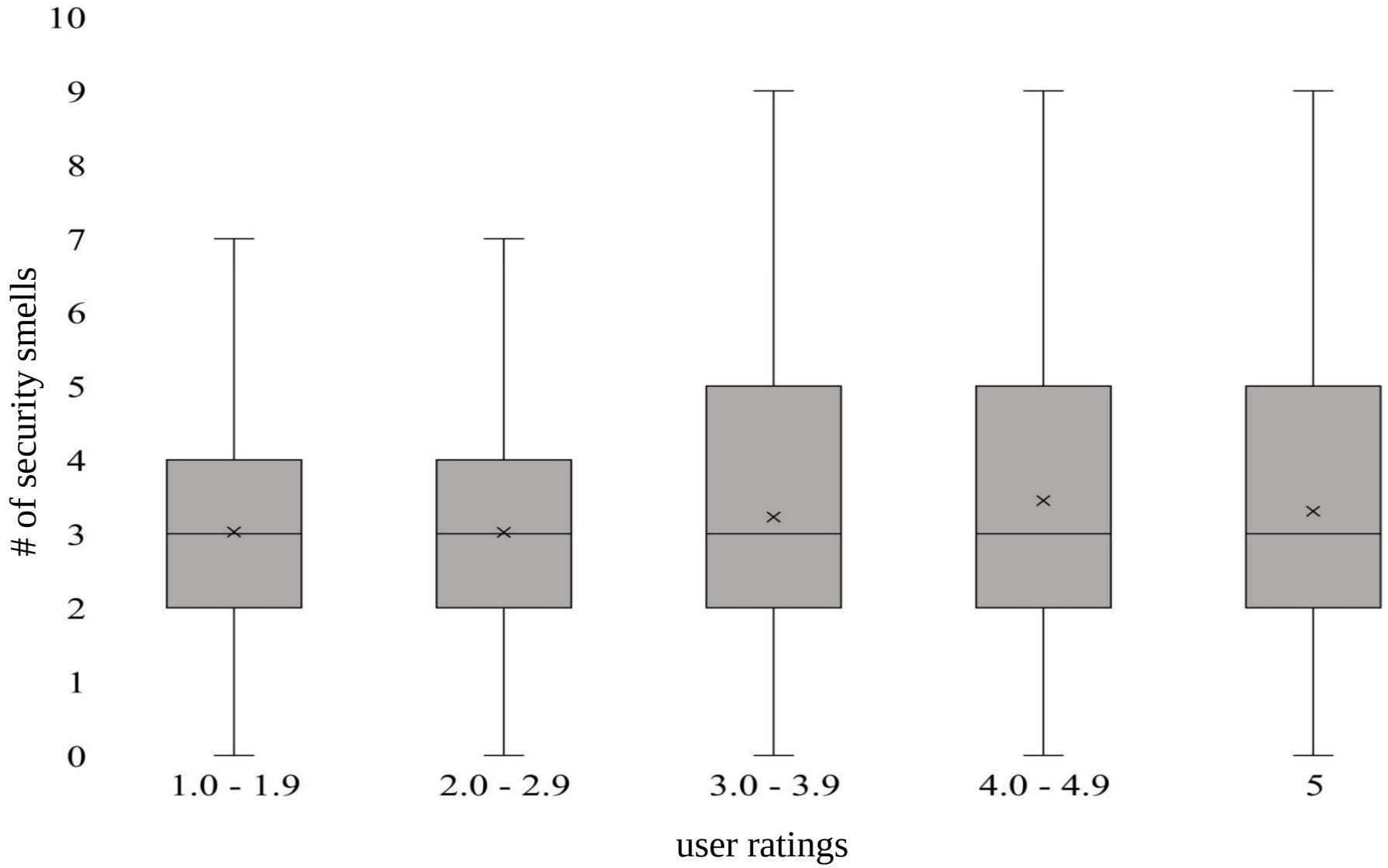
Distribution of Each Smell in API Levels



The Impact of Number of Downloads

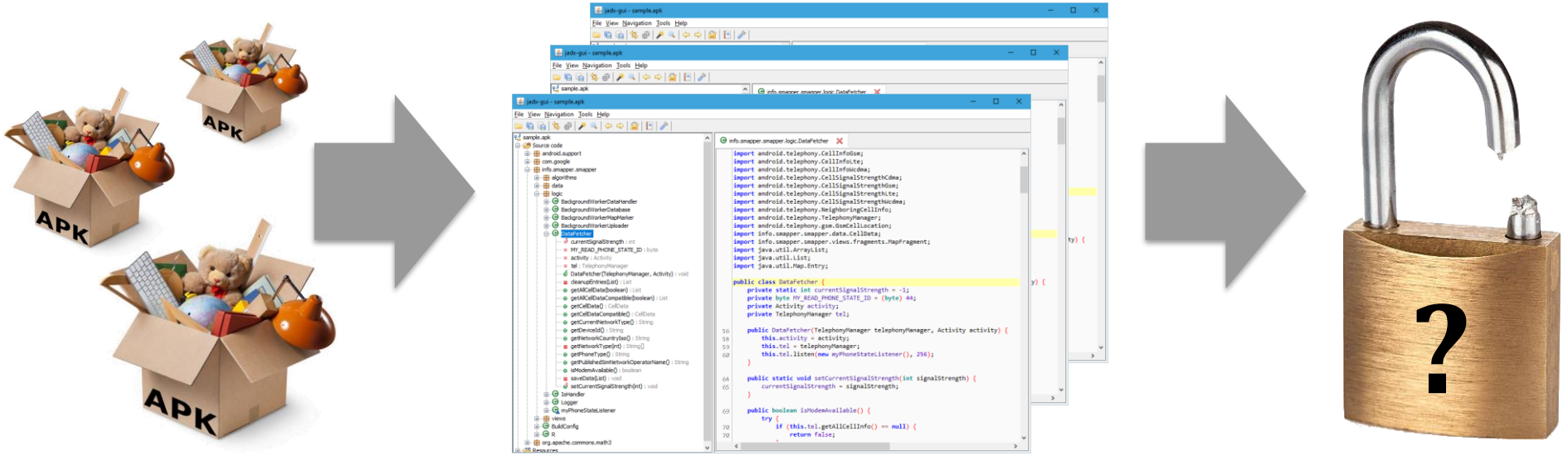


The Impact of User Ratings

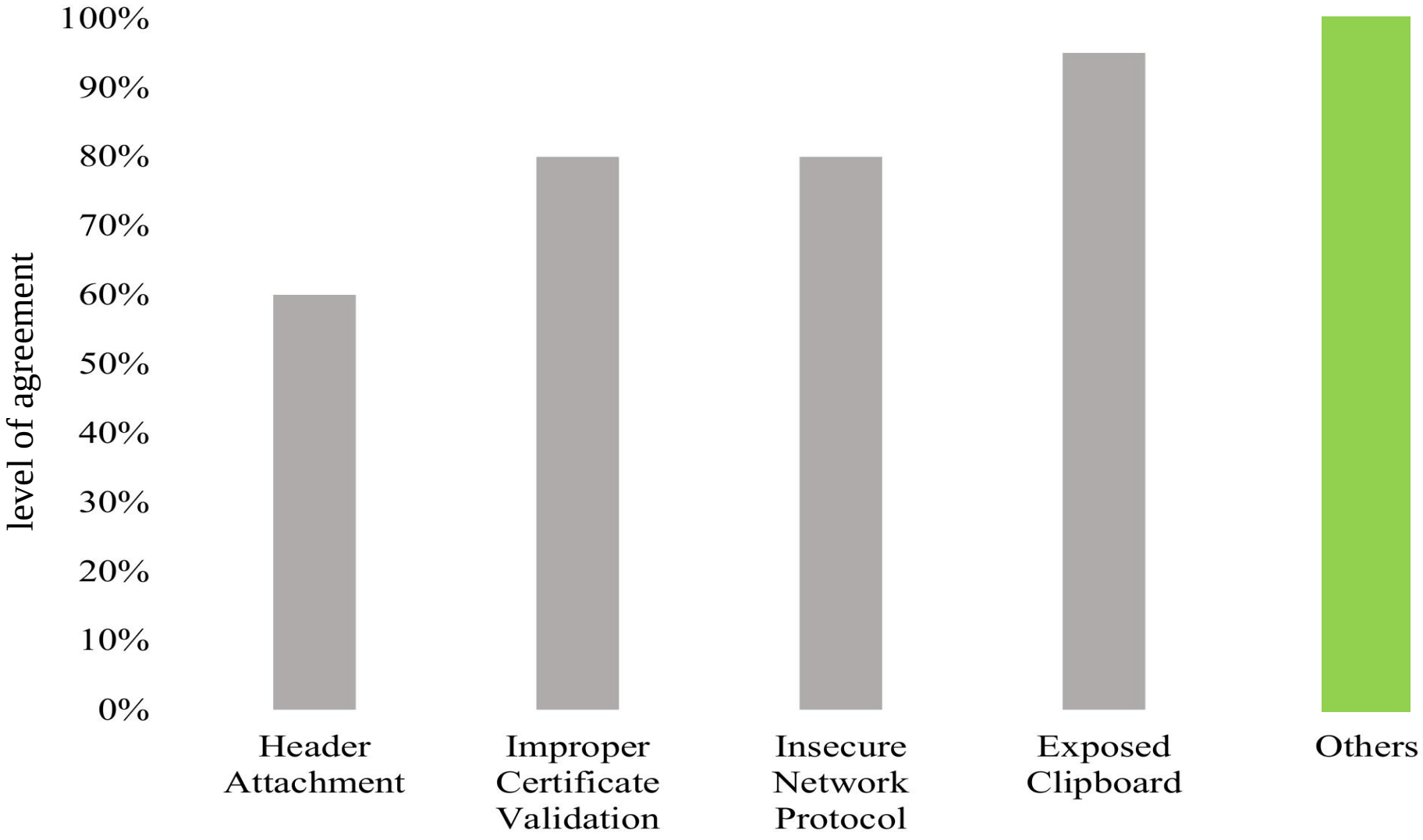


To which extent identifying security smells facilitates detecting vulnerabilities?

Study Design



Result


100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
level of agreement
Header Attachment
Improper Certificate Validation
Insecure Network Protocol
Exposed Clipboard
Others

Where Vulnerability Reports Failed?

- **Header Attachment**

  Data sensitivity of headers

- **Improper Certificate Validation**

  Customised TrustManagers with pinning support

- **Insecure Network Protocol**

  Local web resources in middleware

- **Exposed Clipboard**

  Data sensitivity of content

Summary

Our Contribution

- Increase in security awareness
- Evaluation of security smell distribution
- Lightweight analysis assessment
- In future: In-depth exploration

Thank you
for your attention!