SECAI

AI-supported Security Testing

Supervised by
Michael Schlichtig, Markus Schmidt
Secure Software Engineering
Motivation
Static Application Security Testing (SAST) Tools

63.7% applications ≥ 1 misuse cryptography

Code issues are widespread

Support developers in secure software development
Preparing slides on Friday …

… and then …

CrowdStrike IT outage affected 8.5 million Windows devices, Microsoft says

By Joe Tidy, Cyber correspondent, BBC News
Problem
Usability issues

Pain Points Using Program Analyzers in %

Wrong checks are on by default
Bad warning messages
Too many false positives
Too slow
No suggested fixes
Difficult to fit into workflow
Bad visualization of warnings
No suppression of warnings
No ranking of warnings
Can't selectively turn off analysis
Complex user interface
Can't handle all language features
No support for custom rules
Misses too many issues
Not cross platform

How warning reports often look like
GitHub Action example of CogniCrypt

https://github.com/CROSSINGTUD/CryptoAnalysis

Your code for file encryption compiles?

But is it secure?

```java
1 public class FileEncrypt {
2     private static Key generateKey(String password) throws Exception {
3         DESKeySpec dks = new DESKeySpec(password.getBytes("utf-8"));
4         SecretKeyFactory keyFactory = SecretKeyFactory.getInstance("DES");
5         return keyFactory.generateSecret(dks);
6     }
7 
8 public static String encryptFile(String password, String srcFile, String destFile) {
9     IvParameterSpec iv = new IvParameterSpec("123456".getBytes("utf-8"));
10    Cipher cipher = Cipher.getInstance("DES/CBC/PKCS5Padding");
11    cipher.init(Cipher.ENCRYPT_MODE, generateKey(password), iv);
12    InputStream is = new FileInputStream(srcFile);
13    CipherInputStream cis = new CipherInputStream(is, cipher);
14    // write cis into destFile and return;
15 }
16 }
```

Simplified real-world example.

How can we help developers?

Developing secure code is difficult

```java
public class FileEncrypt {
    private static Key generateKey(String password) throws Exception {
        DESKeySpec dks = new DESKeySpec(password.getBytes("utf-8"));
        SecretKeyFactory keyFactory = SecretKeyFactory.getInstance("DES");
        return keyFactory.generateSecret(dks);
    }

    public static String encryptFile(String password, String srcFile, String destFile) {
        IvParameterSpec iv = new IvParameterSpec("123456".getBytes("utf-8"));
        Cipher cipher = Cipher.getInstance("DES/CBC/PKCS5Padding");
        cipher.init(Cipher.ENCRYPT_MODE, generateKey(password), iv);
        InputStream is = new FileInputStream(srcFile);
        CipherInputStream cis = new CipherInputStream(is, cipher);
        // write cis into destFile and return;
    }
}
```

Goal: Provide developers with a useful interface for using SAST tools such as CogniCrypt.

Simplified real-world example.

What else can we do?

Leveraging AI

- Ranking Warning Messages
  - Filter False Positives
  - Severity Score
  - Difficulty, …

What else can we do?

Current student theses

- Secure code generation by combining LLMs and SAST

- Use LLMs to explain warning messages
Summary

Goals of our Project Group

- Provide developers with a useful interface for using SAST tools such as CogniCrypt.

- Build the SecAI plugin with valuable features!
SecAI
Join our Project Group

Interested in any of these topics?

- Security & Cryptography
- Machine Learning & AI
- User Interface Design
- Software Engineering

Michael Schlichtig
Research Associate
Secure Software Engineering
michael.schlichtig@uni-paderborn.de
Heinz Nixdorf Institut

Markus Schmidt
Employee
Secure Software Engineering
markus.schmidt@uni-paderborn.de
Heinz Nixdorf Institut

go.upb.de/pg-sse-SecAI