Institute of Software Engineering
Empirical Software Engineering (ESE)

Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Bachelorarbeit

Automatische Ableitung von Regeln für statische Sicherheitsanalysen aus öffentlichen Sicherheitslücken-Datenbanken

Maher Albaba

Studiengang:	Informatik
Prüfer/in:	Prof. Dr. Stefan Wagner
Betreuer/in:	Markus Haug, Dr. Ana Cristina Franco da Silva
Beginn am:	01.12.2021
Beendet am:	01.06.2022

Kurzfassung

Während des Zweiten Weltkriegs verlor Deutschland den Krieg, weil Alan Turing damals , die verschlüsselte Nachrichten von den deutschen Soldaten entschlüsseln lassen konnte. Diese Beispiel kann uns zeigen, dass die Informationssicherheit heute eines der wichtigsten Themen. Sicherheitslücken können zu Kriegen führen, Volkswirtschaften schwächen und das Leben von Menschen gefährden, daher suchen die Forscher immer nach neuen Maßnahmen, um diese Sicherheitslücken zu vermeiden. Aber es gibt immer noch viele Sicherheitslücken, die nicht behoben wurden, daher wird immer noch daran geforscht, wie man sie so schnell wie möglich beheben kann. Das Ziel in der vorliegenden Arbeit ist es zu beantworten,wie wir Experten dabei unterstützen können, neue Regeln abzuleiten, um diese Sicherheitslücken schneller zu schließen. Um die Forschungsfrage zu beantworten, wurde einen Ansatz erfunden,um die Ableitung von Regeln für die Schwachstellen aus CVE Datenbank schneller und leichter zu machen. Die Evolution wurde durch Interviews mit Experten durchgeführt, die den Ansatz bewerteten. Eine Ausweitung dieses Ansatzes wäre notwendig. Dazu wurden Anknüpfungspunkte vorgestellt.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Verzeichnis der Algorithmen

1 Einleitung

1.1 Problemdefinition

Kryptografische Bibliotheken, wie die Java Cryptography API (JCA) oder Bouncy Castle, bieten eine integrale Ressource zur Erfüllung von Sicherheitsanforderungen. Die falsche Verwendung von diesen APIs kann zu erhebliche Probleme führen.

Dazu gibt es SAST (Static Application Security Testing)Tools , die können den Missbrauch von den oben genannten APIs erkennen. Die SAST Tools bekommen Regelsätze als Eingabe. Diese Regelsätze können beschreiben, wie die APIs verwenden sollen.

Die Reglsätze werden von eine Experte abgeleitet. Und um ein Experte ein Regelsatz ableiten zu können,soll der Experte die veröffentliche Schwachstellen kennen, was viel Zeit und Geld kostet. Deshalb gibt es eine Datenbank der Name (CVE), in dieser Datenbank werden viele Sicherheitslücken gespeichert.

1.2 Forschungsfrage

Bis jetzt erfolgt die Ableitung manuell.deshalb sollen die Experte viel Zeit damit investieren.Das führt dazu,dass viele Schwachstellen noch nicht behoben werden konnten, und jeden Tag werden neue Sicherheitslücken entdeckt und in CVE Datenbank gespeichert.

In dieser Arbeit antworten wir die Frage,wie wir die Experte beim Ableiten von neuen Regeln unterstützen können,um die Zeit und die kosten ersparen zu können.

1.3 Gliederung

Die Arbeit ist in folgender Weise gegliedert:

Kapitel 2 – Grundlagen: Hier werden werden die Grundlagen dieser Arbeit beschrieben.

Kapitel 3 – Studiendesign: Hier wird die Studiendesign beschrieben,um zu zeigen, wie die Forschungsfrage beantwortet wird.

Kapitel 4 – Ansatz: Den Ansatz wird hier diskutiert.

Kapitel 5 – Evaluation: Hier werden die Evaluation und ihre Ergebnisse diskutiert.

Kapitel 6 – Zusammenfassung und Ausblick fasst die Ergebnisse der Arbeit zusammen und stellt Anknüpfungspunkte vor.

2 Grundlagen

In diesem Kapitel werden alle Grundlagen erklärt, die für die Arbeit relevant sind. SAST-Tools und kryptografische Bibliotheken und CVE-Datenbanken werden ausführlich erläutert.

2.1 the java cryptography api (jca)

Java Cryptography API (JCA) ist eine kryptografische Bibliothek, die Teil der Java-Plattform ist. Es enthält eine Reihe von APIs für digitale Signaturen, Zertifikate und Zertifikatsvalidierung, Verschlüsselung (symmetrischer/asymmetrischer Strom/Blockverschlüsselung) und Schlüsselgenerierung und -verwaltung sowie sichere Generierung von Zufallszahlen[ora].

2.2 Bouncy Castle

Die Bouncy Castle Crypto API ist eine Sammlung kryptografischer Open-Source-Programmierschnittstellen (APIs) für die Programmiersprachen Java und C. Die Bouncy Castle Crypto API enthält auch eine Schnittstelle zur Programmiersprache C, enthält aber nicht alle Algorithmen der Java-Bibliothek [wika].

2.3 SAST Tools

Static Application Security Testing (SAST) wird verwendet, um Software zu sichern, indem der Software-Quellcode untersucht wird, um Quellen von Sicherheitslücken zu identifizieren. SAST-Tools werden in den Entwicklungsprozess integriert, um Entwicklungsteams zu unterstützen. SAST-Tools konzentrieren sich wie andere Sicherheitstools darauf, das Risiko von Anwendungsausfallzeiten zu verringern oder sicherzustellen, dass in Anwendungen gespeicherte private Informationen nicht kompromittiert werden[wikc].

2.3.1 SonarQube

SonarQube ist eine Plattform zur statischen Analyse und technischen Qualitätsbewertung des Quellcodes. SonarQube analysiert den Quellcode für verschiedene Qualitätsbereiche und zeigt die Ergebnisse auf der Website an. SonarQube ist in Java programmiert, unterstützt aber die Programmiersprachen JavaScript, Groovy, Flex, PHP, PL/SQL, C, Cobol, .NET und Visual Basic.[Son].

2.3.2 Eclipse CogniCrypt

Eclipse CogniCrypt ist auch eine Werkzeug, die Falsche Verwendung von Kryptografischen Bibliotheken erkennen und beheben kann. Dazu gibt es zwei Komponenten.

Code Generation : die überprüft,ob den Code genug sicher ist.

Static Code analysis : die überprüft den code direkt in Eclipse kontinuierlich[Ecl].

2.4 CVE Datenbank

Die CVE-Datenbank ist eine Datenbank, in der Schwachstellen und andere Sicherheitslücken gespeichert werden. Ziel ist es, diese Probleme in Zukunft vermeiden zu können. Jede Sicherheitslücke hat eine ID, eine CWE-Nummer und eine Beschreibung. Die ID, der zur Identifizierung der Schwachstelle verwendet wird, enthält die folgende Codierung: CVE-Year-Index. Die Beschreibung beschreibt das Problem genau, es kann Quellcode oder Text sein. Derzeit enthält die Datenbank etwa 170.000 Schwachstellen.

2.5 CWE Datenbank

Common Weakness Enumeration (CWE™) ist eine von der Community entwickelte Liste gängiger Arten von Software- und Hardwareschwächen, die Auswirkungen auf die Sicherheit haben. „Schwächen“ sind Bugs oder andere Fehler in der Software- oder Hardwareimplementierung, im Code, Design oder in der Architektur, die, wenn sie nicht behoben werden, dazu führen können, dass Systeme, Netzwerke oder Hardware anfällig für Angriffe sind. Die CWE-Liste und die zugehörige Klassifizierungstaxonomie dienen als Sprache, die verwendet werden kann, um diese Schwächen in Bezug auf CWEs zu identifizieren und zu beschreiben[MITb].

Jedes Jahr wird eine Liste der 25 gefährlichsten Software-Schwachstellen der letzten zwei Jahre veröffentlicht, die leicht zu finden und auszunutzen sind.In einigen Fällen kann ein Angreifer das System vollständig übernehmen.in Tabelle 2.1 sind Top 5 der gefährlichsten Software-Schwächen in 2021[MITa].

Rank	ID	NAME
1	CWE-787	Out-of-bounds Write
2	CWE-79	Cross-site Scripting
3	CWE-125	Out-of-bounds Read
4	CWE-20	Improper Input Validation
5	CWE-78	OS Command Injection

Tabelle 2.1: Top 5 Most Dangerous Software Weaknesses (CWE Top 5) in 2021

3 Studiendesign

Wir beschreiben nun die Schritte, die unternommen wurden, um das Ziel in dieser Arbeit zu erreichen. Wir haben das Problem und die Forschungsfrage in der Einleitung gesehen. Dann wurde eine Methode erfunden und implementiert. Schließlich gab es eine Evaluation zur Bewertung von Arbeit und Methoden. Abbildung 3.1 verdeutlicht das Studiendesign

3.1 Problem Definition

Experten verbringen zu viel Zeit damit, Schwachstellen zu finden. Es kostet Zeit und Geld, und jeden Tag tauchen neue Sicherheitslücken auf, die behoben werden müssen, aber die Experten können sie nicht so schnell beheben. Deshalb sollte einen Ansatz erfunden werden,um das Problem zu lösen.

3.2 Ansatz

Zunächst werden 2 Ansätze vorgeschlagen, die aber im Rahmen dieser Arbeit nicht umgesetzt werden können. Im Kapitel 4 ist das Problem ausführlich beschrieben. Um dieses Problem zu lösen, wurde eine Methode erfunden, die Experten dabei hilft, neue Regeln abzuleiten. Der Ansatz wird dann in Form eine Werkzeug implementiert.

3.3 Implementierung

Um die Werkzeug zu implementieren, wurden Daten aus der CVE-Datenbank abgerufen. Dann sollte ich wiesen, welche Informationen ich aus der CVE-Datenbank benötige und was ich

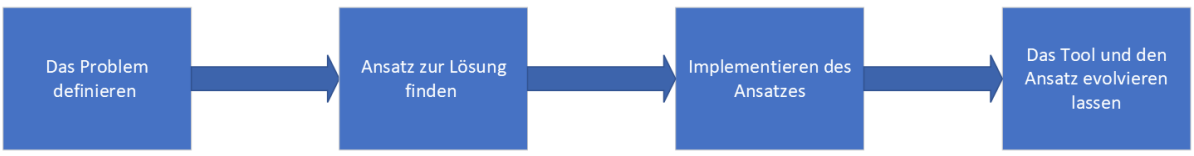

Abbildung 3.1: Studiendesign

damit machen kann, um mein Ziel zu erreichen. Ich habe dann ein Backend Implementiert, um diese Daten so zu ändern, dass sie in der Werkzeug hilfreich sind. Andererseits wurde eine Schnittstelle implementiert, um die Nutzung der Werkzeug zu vereinfachen. Außerdem wurde eine Datenbank erstellt, um die neuen Regeln zu speichern.

3.4 Evaluation

Abschließend sollte die Werkzeug evaluiert werden, wobei der Zweck der Evaluation ist aufzuzeigen, ob das Tool dem Experten bei der Ableitung helfen kann und was der Experte verbessern möchte. Die Evolution wurde durch Experteninterviews durchgeführt und die Ergebnisse werden zur Verfügung gestellt.

4 Ansatz

Hier untersuchen wir den Ansatz um die Forschungsfrage zu beantworten.Wir schlagen zuerst zwei Ansätze vor, aber diese Ansätze können nicht umgesetzt werden, dann diskutieren wir den Ansatz, der in dieser Arbeit verwendet wurde.

4.1 Rahmenbedingung

Für jede Schwachstelle gibt es eine Beschreibung, wir können eine Werkzeug implementieren, die die Beschreibung untersucht und eine oder mehrere Lösungen vorschlägt, zum Beispiel: Es wurde ein Verschlüsselungsschema verwendet, das nicht sicher ist, dann schlägt die Werkzeug ein bessere Verschlüsselungsschema vor.Allerdings kann Eine falsche Ableitung jedoch zu schwerwiegenden Problemen führen,oder das Regel ist nicht sicher genug, um die Schwachstelle zu vermeiden. Daher sollte der halbautomatische Ableitung unter Kontrolle von Experten durchgeführt werden.d.h: die Regeln sollten von Experten ausprobiert werden, bevor die Regeln veröffentlicht werden.Das bedeutet,dass die vollautomatische Ableitung nicht machbar ist.

Ebenso kann der einen halbautomatische Ableitung im Rahmen einer solchen Arbeit nicht umgesetzt werden. Das erste Problem, das bei einigen Schwachstellen auftaucht, ist, dass die Beschreibung kein Text, sondern Quellcode ist, was es schwieriger macht.Besonders wenn kein Clean-code berücksichtigt wurde. Zweitens soll künstliche Intelligenz eingebaut werden, was bedeutet, dass Werkzeug Zeit braucht, um zu lernen, wie neue Regeln abgeleitet werden können. Überdies kann es sein,dass neue Schwachstellen auftreten, die zuvor keine ähnliche Schwachstelle hatten, sodass die Werkzeug diese nicht kennt. Beispielsweise bei der 5G Netzwerke sind immer noch vielleicht bis jetzt keine Schwachstellen aufgetreten.

Außerdem haben einige Schwachstellen in der CVE-Datenbank keine cwe-Nummer, sodass wir nicht sagen können, in welche Kategorie die Schwachstelle fällt. Am Ende konnten wir feststellen, dass ein solche Werkzeug im Rahmen dieser Arbeit nicht erfunden werden kann, weshalb wir nach einem alternativen Ansatz suchen.

Um Zeit und Geld zu sparen, wurde als Grundlage für zukünftige Forschungen in diesem Bereich ein Tool implementiert, das Experten dabei hilft, neue Regeln abzuleiten.

4.2 DOF Tool

4.2.1 Vorteile

Der Experte kann so wahrscheinlich mehr Regeln in derselben Zeit ableiten, was mehr Sicherheitslücken vermeidet. Zudem besitzen die Experten sehr spezielle Fähigkeiten, daher sind sie kostenintensiv. Eine Zeitersparnis bedeutet, dass diese wertvolle Zeit besser genutzt wird.

Ein Experte kann auch neue Regeln basierend auf den Ideen der anderen Experten ableiten.

Man kann nachverfolgen, aus welchen CVEs , wann und von wem eine Regel abgeleitet wurde. Das ist unter Umständen auch eine wichtige Hilfe für einen Experten.

4.2.2 Funktionen

Die Werkzeug kan folgende Aufgaben erfüllen:

1. Nach einer bestimmte Schwachstelle suchen: bis jetzt gibt es keine Schnittstelle,um nach einem bestimmten Schwachstelle zu suchen.Und das ist so wichtig,denn zurzeit sind etwa.170.000 Schwachstellen gespeichert .
2. Nach einem Anbieter suchen:hier werden alle Produkte aufgelistet.
3. Nach einem Produkt suchen:das hilft,wenn man die ID der Schwachstelle nicht kennt
4. Die ähnliche Problem sehen.
5. Es wird immer nach dem Prozentsatz sortiert und angezeigt, dass dieses Problem mit ähnlichen Problemen übereinstimmt.
6. Experten können neue Regeln für spezifische Probleme ableiten und in einer Datenbank speichern: Das hilft auch anderen Experten.
7. Diese Regel wird mit dem Namen des Erstellers und dem Datum der Erstellung eingehalten.

4.2.3 Komponenten und Technologien

Wir werden nun einige wichtige Komponenten des Ansatzes untersuchen.

Algorithmus 4.1 API Integration

```
import requests
# request delivery of data
response = requests.get('http://cve.circl.lu/api/browse')
res = response.status_code
print(res)
```

Code	Message
200	Ok
301	Moved Permanently
302	Moved Termporarily
404	Not Found
500	Internal Server Error
503	Service Unavailable

Tabelle 4.1: response Status code

CVE Search API

" API (Application Programming Interface) ist ein Satz von Befehlen, Funktionen, Protokollen und Objekten,die Programmierer verwenden können, um eine Software zu erstellen oder mit einem externen System zu interagieren[tal]".

Cve Search Api ist ein API , das uns ermöglicht , die Einträge vom CVE Datenbank abzurufen. Dieses API liefert immer die aktualisierte Daten. Der Algorithmus 4.1 zeigt, wie die API in dem System integriert ist. Tabelle 4.1 zeigt, was die Statuscodes sind. Wenn der Code den Status 200 zurückgibt, bedeutet dies, dass die angeforderten Daten bei Bedarf übertragen werden können[Ryt]. Tabelle 4.2 zeigt die JSON-formatierten Rückgaben, an denen wir in dieser Arbeit interessiert sind[cir].

Url	Return
/api/cve/<cveid>	alle verfügbaren Informationen für die angegebene CVE ID
api/cwe	eine Liste aller CWEs
/api/last/<limit>	die letzten n Schwachstellen
/api/dbInfo	Datenbankstatistiken
/api/browse/vendor	Alle Produkte eines Anbieters auflisten
/api/search/vendor/product	Alle Schwachstellen des angegebenen Produkts

Tabelle 4.2: CVE APIs

Beispielrückgaben für bestimmte Einträge finden Sie in im Algorithmus 4.2. Die Rückgabe umfasst die Schwachstelle, die Beschreibung der Schwachstelle, das Eintrittsdatum und die Lösung(falls existiert).

Django

Django ist ein kostenloses, in Python geschriebenes Open-Source-Framework für Webanwendungen. Es bietet viele vorgefertigte Lösungen, um die Entwicklng von Webanwendungen zu vereinfachen. Der Rahmen ist da, damit wir das Rad nicht neu erfinden müssen[dja]. In unserer Werkzeug wird Django verwendet,um das Backend zu Implementieren. Dann wird eine menge von APIs produziert,um mit dem Frontend verbindet werden zu können.

React.js

React ist eine JavaScript-Softwarebibliothek, zum Rendern von Benutzeroberflächenkomponenten dient. Single-Page-Webanwendungen, kann aber auch serverseitig mittels Node.js (vor-)gerendert werden[wikb]. In unserer Werkzeug wird React.js benutzt,um das Frontend zu Implementieren. Die Oberflächen werden damit implementiert,und mit den APIs,die von Backend geliefert sind,verbunden.

MySQL

"MySQL ist ein quelloffenes SQL-Datenbank-Managementsystem und die Grundlage für dynamische Webseiten. MySQL verwaltet, zeigt, speichert und ändert Daten in Tabellen – die klassische Aufgabe eines Datenbank-Managementsystems (DBS). Es funktioniert dabei als Client-Server-System: Die jeweilige Datenbank ist der Server. Die Software auf der Client-Seite schickt Befehle an die Datenbank. Die Datenbank übersetzt die Befehle in ausführbaren Code, führt die Befehle aus und sendet die Informationen darüber an den Client[che]".

4.2.4 Implementierung

Hier wird die eigentliche Implementierung beschrieben,die Beziehung zwischen Komponenten ist in Abbildung 4.1 zu sehen

Algorithmus 4.2 Rückgabe Beispiel

```
import requests
# request delivery of data
response = requests.get('https://cve.circl.lu/api/cve/CVE-2021-3328')
res = response.json()
print(res)
Output :
{'Modified': '2021-04-14T20:20:00', 'Published': '2021-04-08T18:15:00', 'access':
    {'authentication': 'NONE', 'complexity': 'LOW', 'vector': 'NETWORK'}, 'assigner':
    'cve@mitre.org', 'capec': [{'id': '537', 'name': 'Infiltration of Hardware
    Development Environment', 'prerequisites': 'The victim must use email or removable
    media from systems running the IDE (or systems adjacent to the IDE systems). The
    victim must have a system running exploitable applications and/or a vulnerable
    configuration to allow for initial infiltration. The attacker must have working
    knowledge of some if not all of the components involved in the IDE system as well as
    the infrastructure.', 'related_weakness': ['125'], 'solutions': '', 'summary': "An
    attacker, leveraging the ability to manipulate components of primary support systems
    and tools within the development and production environments, inserts malicious
    software within the hardware and/or firmware development environment. The
    infiltration purpose is to alter developed hardware components in a system destined
    for deployment at the victim's organization, for the purpose of disruption or further
    compromise."}, {'id': '540', 'name': 'Overread Buffers', 'prerequisites': 'For this
    type of attack to be successful, a few prerequisites must be met. First, the targeted
    software must be written in a language that enables fine grained buffer control.
    (e.g., c, c++) Second, the targeted software must actually perform buffer operations
    and inadequately perform bounds-checking on those buffer operations. Finally, the
    adversary must have the capability to influence the input that guides these buffer
    operations.', 'related_weakness': ['125'], 'solutions': '', 'summary': 'An adversary
    attacks a target by providing input that causes an application to read beyond the
    boundary of a defined buffer. This typically occurs when a value influencing where to
    start or stop reading is set to reflect positions outside of the valid memory
    location of the buffer. This type of attack may result in exposure of sensitive
    information, a system crash, or arbitrary code execution.'}], 'cvss': 5.0,
    'cvss-time': '2021-04-14T20:20:00', 'cvss-vector': 'AV:N/AC:L/Au:N/C:N/I:N/A:P',
    'cwe': 'CWE-125', 'id': 'CVE-2021-3328', 'impact': {'availability': 'PARTIAL',
    'confidentiality': 'NONE', 'integrity': 'NONE'}, 'last-modified':
    '2021-04-14T20:20:00', 'references':
    ['https://jankopecky.net/index.php/2021/04/08/cve-2021-3328-abyss-web-server-remote-dos/'],
    'summary': 'An issue was discovered in Aprelium Abyss Web Server X1 2.12.1 and 2.14.
    A crafted HTTP request can lead to an out-of-bounds read that crashes the
    application.', 'vulnerable_configuration': [{'id':
    'cpe:2.3:a:aprelium:abyss_web_server_x1:2.12.1:*:*:*:*:*:*:*', 'title':
    'cpe:2.3:a:aprelium:abyss_web_server_x1:2.12.1:*:*:*:*:*:*:*'}, {'id':
    'cpe:2.3:a:aprelium:abyss_web_server_x1:2.14:*:*:*:*:*:*:*', 'title':
    'cpe:2.3:a:aprelium:abyss_web_server_x1:2.14:*:*:*:*:*:*:*'}],
    'vulnerable_configuration_cpe_2_2': [], 'vulnerable_product':
    ['cpe:2.3:a:aprelium:abyss_web_server_x1:2.12.1:*:*:*:*:*:*:*',
    'cpe:2.3:a:aprelium:abyss_web_server_x1:2.14:*:*:*:*:*:*:*']}
Process finished with exit code 0
```

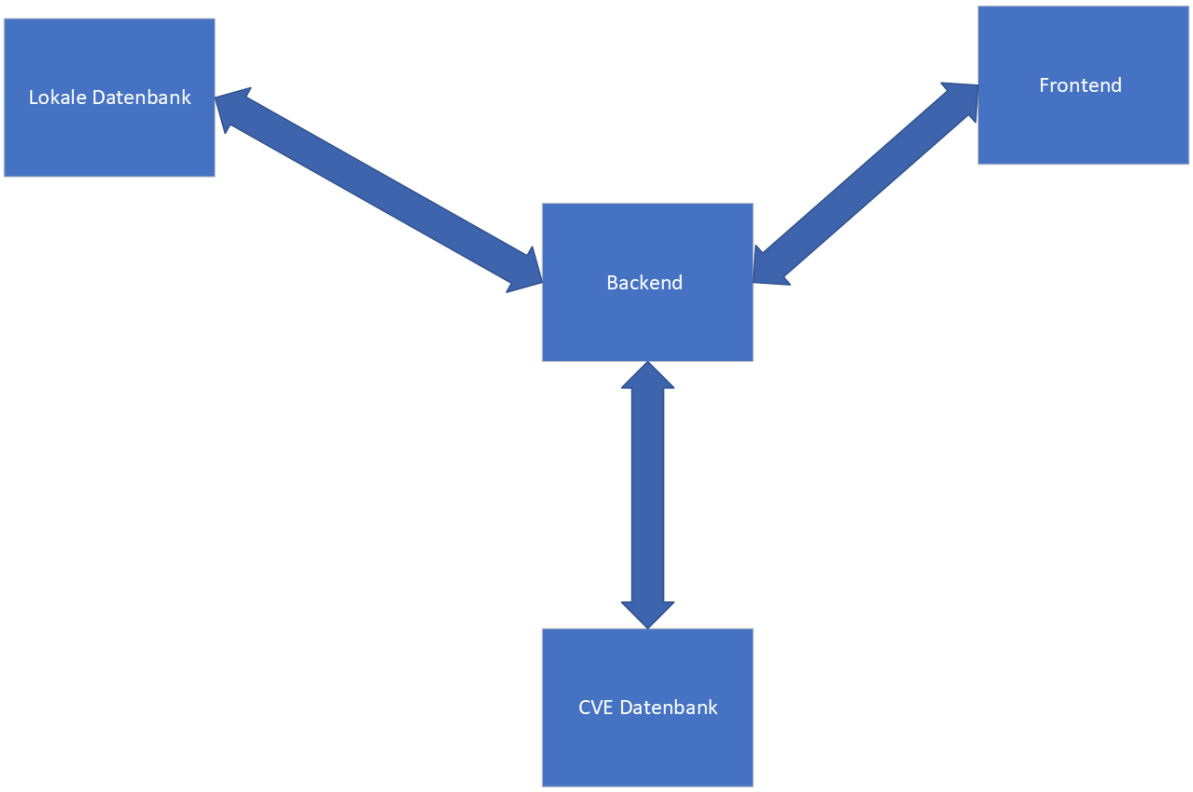

Abbildung 4.1: Beziehung zwischen Komponenten

Backend

Das Backend ist mit Django implementiert. Django ist eine Python-Bibliothek, und wie wir in Kapitel 6 sehen werden, kann dieser Ansatz potenziell mit künstliche intelligenz erweitert werden, also wird Django verwendet, weil die Programmiersprache Python großartig für künstliche intelligenz ist. Das REST-Framework wird verwendet. Das REST-Framework ermöglicht es uns auch, reguläre funktionsbasierte Ansichten zu verwenden. Außerdem können die zu bearbeitenden Anfragen konfiguriert werden[Dja].

@api-view:
Es ermöglicht uns, die Logik zu beschreiben, die unseren API Endpunkt ausmacht. Für jedes API wird eine Funktion implementiert. In dieser Funktion wird beschrieben,was dieses API machen soll[Ros]. Es gibt zwei Arten von Anfragen :

- "GET : Die GET-Methode ist in der Lage, Informationen jeglicher Art mithilfe der Ergebnis-URI zu identifizieren. Wenn die URI auf einem Prozess basiert, welcher Daten zurückgibt, besteht dieser aus den erzeugten Daten. Die vollständige URI besteht aus der URL, einem Fragezeichen als Trennzeichen und den Daten.

ID	API	Get oder Post
1	get-CVE-and-description	Get
2	get-vendor	Get
3	get-product	Get
4	post-solutions	Post

Tabelle 4.3: Backend APIs

ID	CVE-ID	Author	Solution	Date

Tabelle 4.4: tool-solutionsmodel tabelle

- Post :Die POST-Methode wird vorwiegend eingesetzt, um dem Server mitzuteilen, dass eine Anforderung des Clients weitere Daten enthält.[sel]"

Tabelle 4.3 zeigt,die im Back-End verwendeten APIs.

Frontend

Die Werkzeug enthält folgende Seiten:

1. Startseite.
2. Search by ID
3. Search by Vendor
4. Search by Product
5. Add Solution

Die Seiten werden wir danach nähere Kennenlernen.

Datenbank

Die erstellte Datenbank enthält eine Tabelle namens tool-solutionsmodel(Tabelle 4.4). Alle erstellten Lösungen werden in dieser Tabelle gespeichert. Beim Zugriff auf die Seite(search by ID) können Inhalte aufgerufen und angezeigt werden. Die Tabelle sieht so aus Tabelle 4.4.

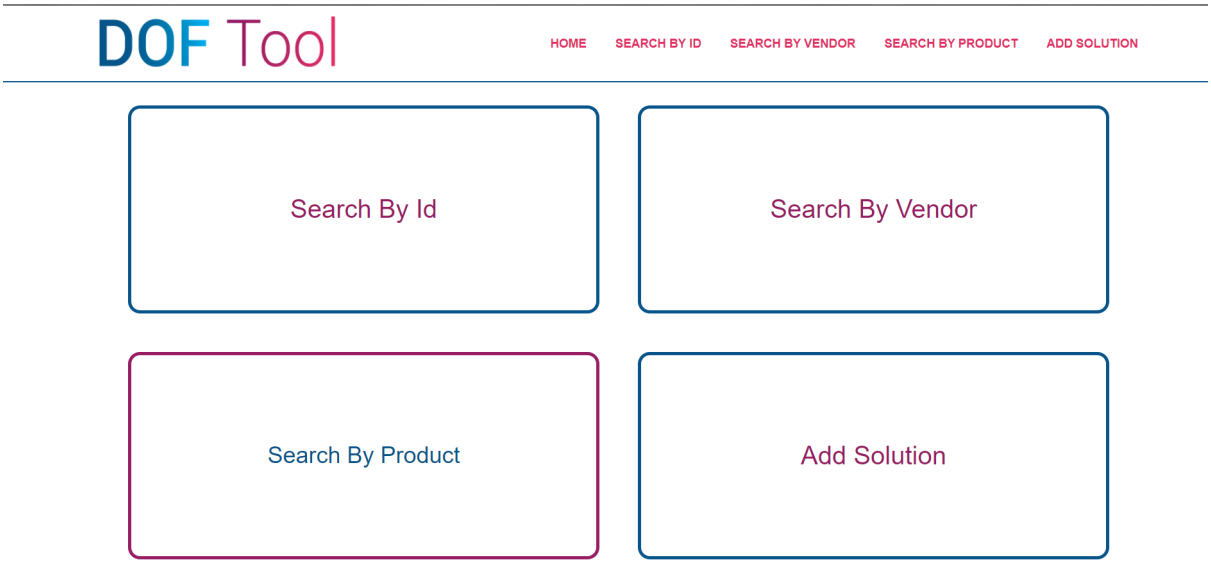

Abbildung 4.2: Startseite

4.2.5 Seiten

Startseite

In der Startseite (Abbildung 4.2)steht folgendes zur Auswahl :

1. Search by ID
2. Search by Vendor
3. Search by Product
4. Add Solution

Search by ID

Auf dieser Seite(Abbildung 4.4,Abbildung 4.5 und Abbildung 4.5) kann der Benutzer eine CVE-ID angeben und Folgendes passiert.

1. ES wird die Schwachstellenbeschreibung für die angegebene ID zurückgegeben.
2. Jede Schwachstelle hat eine cwe-id, dann werden alle Probleme mit derselben cwe-id zurückgegeben.
3. Jede Schwachstelle hat eine Beschreibung.wird die Schwachstellenbeschreibung mit der Beschreibung für die angegebene ID , ohne Stoppwörter, vergliechen. Stoppwörter sind häufig vorkommende und irrelevante Wörter im Text, wie (und,das,etc...)

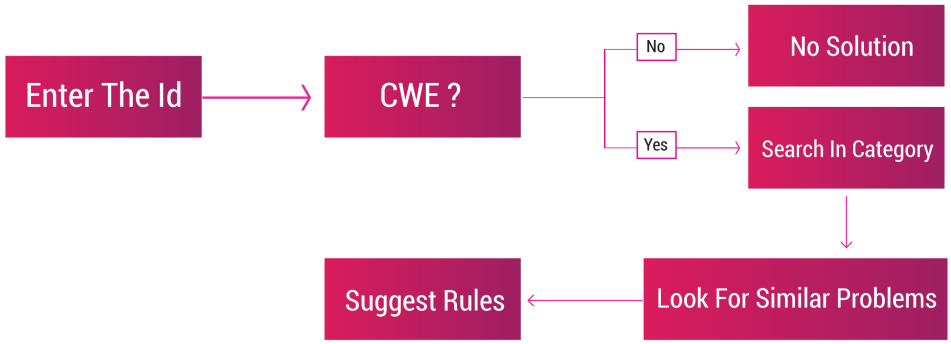

Abbildung 4.3: Prozess

4. Es berechnet den Übereinstimmungsprozentsatz basierend auf String-Matching.
5. Jedes Problem wird in der Datenbank gesucht, ob es bereits eine Lösung gibt, und wenn ja, wird die Lösung zusammen mit dem Namen des Erstellers und der Erstellungszeit zurückgegeben. Gibt „no solution“ zurück, wenn es keine gibt.
6. Ähnliche Schwachstellen werden nach Übereinstimmungsprozentsatz in absteigender Reihenfolge sortiert

Die Abbildung 4.3 erklärt das Prozess.

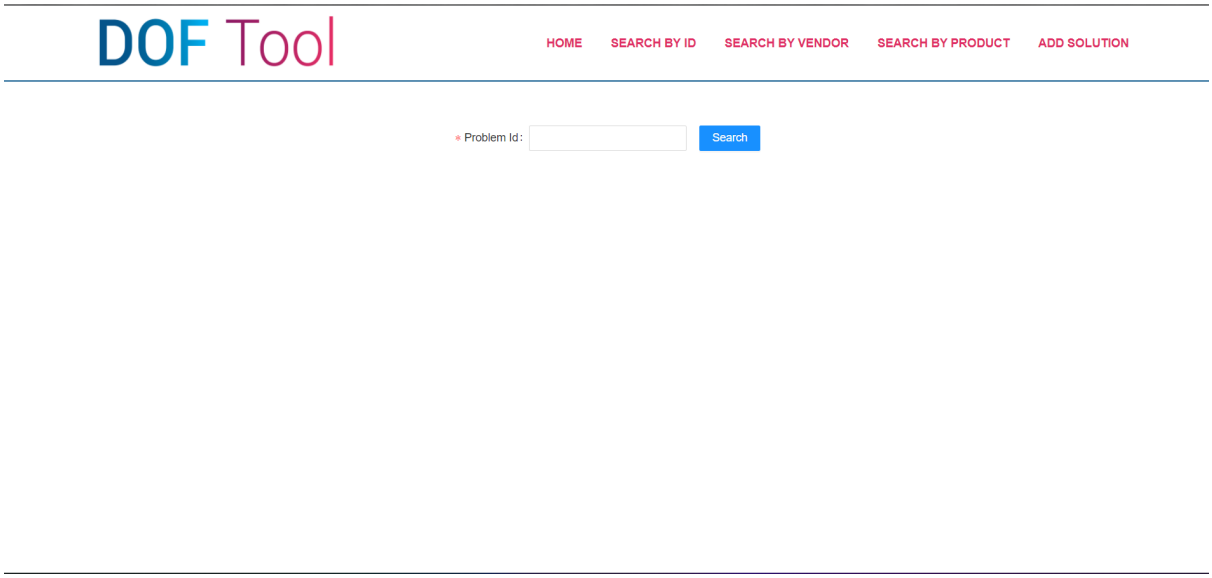

Abbildung 4.4: Search by ID(1)

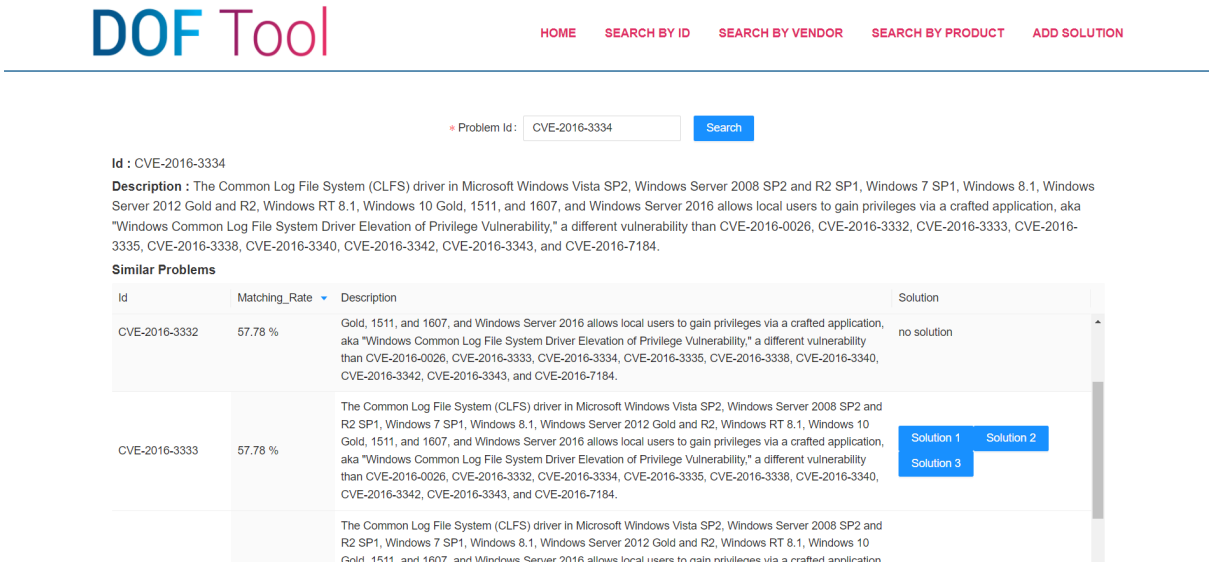

Abbildung 4.5: Search by ID(2)

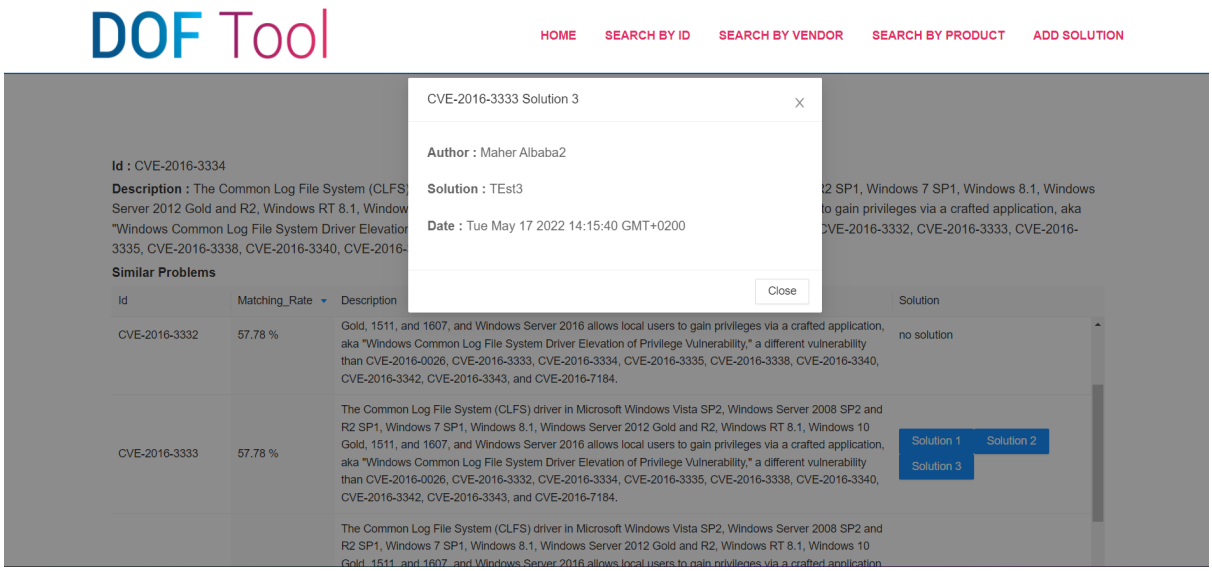

Abbildung 4.6: Search by ID(3)

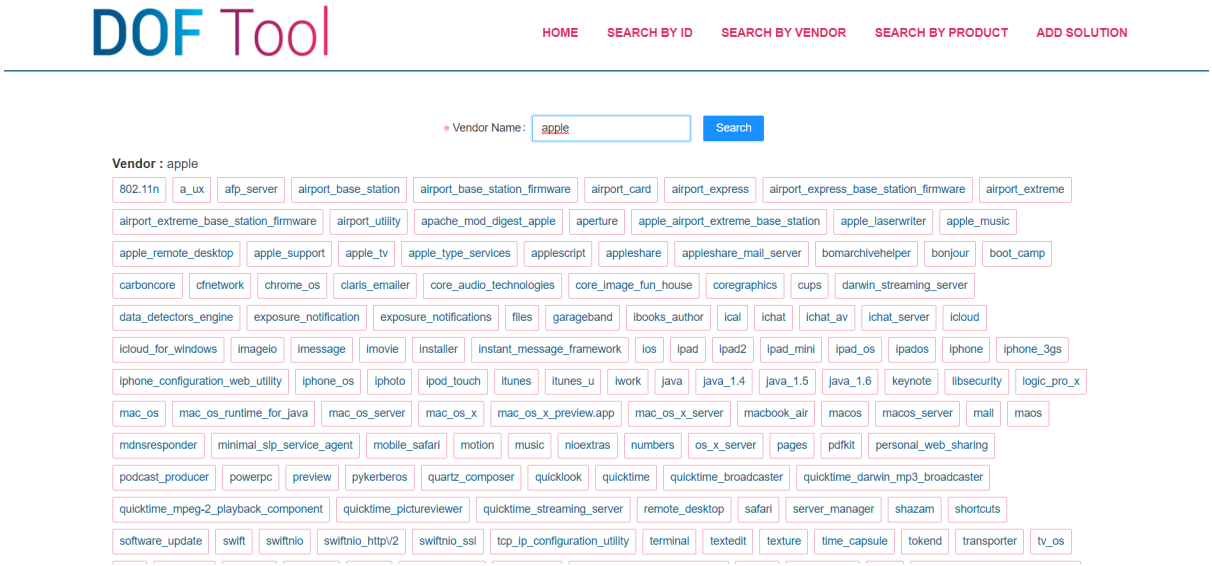

Abbildung 4.7: Search by vendor

Search by Vendor

Auf dieser Seite(Abbildung 4.7) werden alle Produkte vom angegebenen Anbieter zurückgegeben.

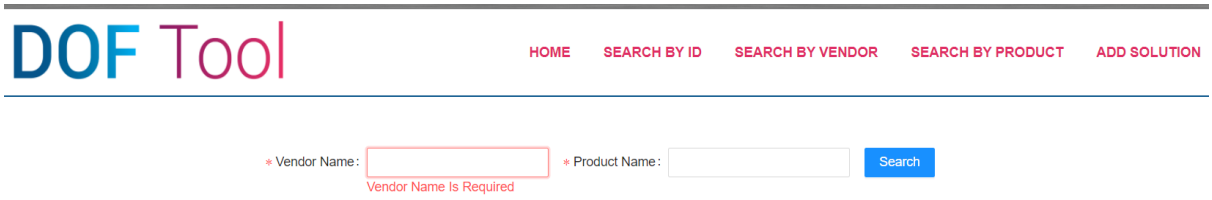

Abbildung 4.8: Search by product

Search by product

Hier können Sie einen Anbieter und ein bestimmtes Produkt angeben, und die Schwachstellen für dieses Produkt werden aufgelistet. Dies hilft Benutzern, die die ID nicht kennen(Abbildung 4.8).

Note: Zum Zeitpunkt der Erstellung des Berichts funktioniert die API nicht mehr, dies liegt am API-Anbieter.

Add solution

Der Benutzer kann eine Lösung für eine bestimmte Schwachstelle eingeben, die in der Datenbank gespeichert wird. Folgende Angaben sind notwendig:

1. CVE-ID
2. Name des Erstellers
3. Die Lösung

Datum und Uhrzeit werden automatisch gespeichert(Abbildung 4.9).

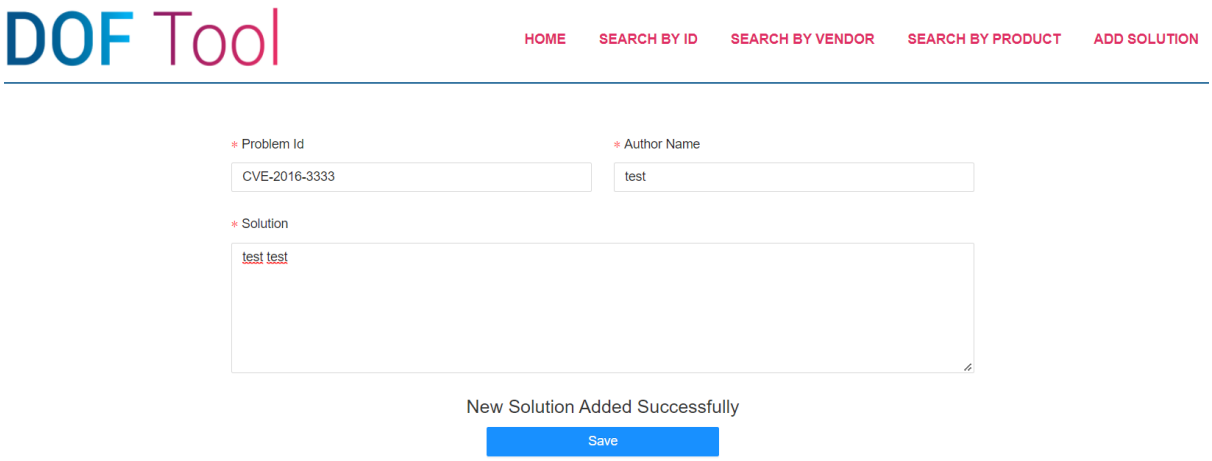

Abbildung 4.9: add solution

5 Evaluation

Zur Evaluation der Werkzeug habe ich Experteninterviews durchgeführt Erstens wird ein Termin mit einem Experten vereinbart. zweitens wurde ein Protokoll für die Durchführung des Interviews verfasst.Dann wurde eine Zeitschätzung vorgenommen . Es ist wichtig zu wissen, welche Qualifikationen die Experten haben sollen. Bei unserer Arbeit müssen die Experten im Bereich der Informatik tätig sein und näheren Kontakt mit den Schwachstellen der CVE-Datenbank haben, idealerweise sollten die Experten die Regeln bereits ableiten können[And05].

Da die Experten über besondere Qualifikationen verfügen sollten, standen nur wenige Experten zur Verfügung und aufgrund des Zeitdrucks gab es nur einen Teilnehmer. In dieser Arbeit wurde ein Interview-Preamble erstellt und an dem Experte gesendet, wo er die Beschreibung unserer Arbeit lesen konnte, sie enthält auch Informationen zum Datenschutz. Der Experte bleibt anonym und während des Interviews wurde die Stimmte aufgezeichnet, um den Bericht zu erstellen, wonach die Daten vernichtet werden.

Im Interview wurde zuerst erklärt, was in diesem Interview besprochen wird, dann wurden Fragen gestellt und dann wurde alles zusammengefasst,und dazu einen Bericht geschrieben. Das Interview-Preamble finden Sie im Abbildung 5.1.

Die Arbeit konzentriert auf die CVE-Datenbank und Informationssicherheit , deshalb sollte der Experte in diesem Bereich arbeiten. die vierte Frage ist relevant , damit der Experte die Ableitung der Regel mit und ohne Tool besser beurteilen kann .bei den andere Fragen ist es wichtig zu wissen, welche Verbesserungen aus Expertensicht möglich sind. Die Fragen die im Interview gestellt wurden, sind:

- Arbeiten Sie in der IT-Branche?
- arbeiten Sie im Bereich Informationssicherheit?
- Kennen Sie schon die CVE-Datenbank?
- Wenn ja, haben Sie bereits eine Regel abgeleitet?
- Was sind die Schwächen dieses Tools?
- Welche Verbesserungen können Sie vorschlagen?

Interview Preamble : DOF Tool

University of Stuttgart

Name	Email
Maher Albaba	e-mail address
Participant	e-mail address

PURPOSE

DOF tool is implemented to help the experts in deriving rules for the vulnerabilities stored in CVE database. The goal of the tool is to search for vulnerabilities, these vulnerabilities are stored in the CVE Database. The goal of the tool:

1. search for vulnerabilities by id
2. look for similar vulnerabilities
3. Add a solution to a specific problem

Evaluate whether the tool helps the expert to derive a new rule, and the tool should make it easier to find a vulnerability, either by its ID or by the product name

CONFIDENTIALITY AND ANONYMITY

Everything that is said in the interviews is treated as strictly confidential and will not be disclosed to anyone outside our research team without the participant's permission, not even to his/her manager. Results are aggregated and anonymized before publication. In no way can individual statements be traced back to a company or an employee. After the interview, participants also have the possibility to exclude or redact statements before the analysis. The complete interview transcripts will NOT be published.

INTERVIEW PROCESS AND ANALYSIS

Interviewees will be notified about the general topic of the study so that they are familiar with it and can gather information beforehand should they decide to do so. Interview time is estimated between 15 and 30 minutes. We kindly ask your permission to record the interview for transcription. The written transcript will be sent back to the participant for his/her final approval. This is the last chance to remove or redact statements from the interview. After that, the audio will be destroyed.

Abbildung 5.1: Interview-Preamble

5.1 Ergebnis

Der Experte arbeitet in der IT-Branche und im Bereich Informationssicherheit, kannte aber die CVE-Datenbank noch nicht.Das bedeutet,dass der Experte teil unserer Zielgruppe ist.

Die Werkzeug und die Arbeit wurden mit 8 von 10 bewertet(10 als perfekt und 0 als sehr schlecht).

Die Werkzeug kann die Experten unterstützen,denn es darstellt eine Suchmaschine(Wie google) für die sicherheitslücken.

Eines der Probleme ist, dass Benutzer die Regeln nicht bearbeiten können.weil die Experten vielleicht eine Regel verbessern möchten oder haben festgestellt, dass die Regel doch nicht genug sicher sind.

Darüber hinaus sollte es eine Möglichkeit geben, die Regeln zu evaluieren, insbesondere wenn es mehrere Lösungen für die Schwachstelle gibt,so konnte der Benutzer sich besser orientieren.

Das String-Matching muss nicht 100% korrekt sein, um die Übereinstimmung zu überprüfen.Denn es ähnliche Wörter geben könnte , aber in unterschiedlichen Kontexten.

5.2 Einschränkungen

Da die Evaluation nur von Experten durchgeführt werden sollte, entstand das Problem, dass es nicht viele Experten gab, die die Evaluation durchführen konnten, also wurde die Evaluation nur mit einem Experte durchgeführt. Daher war es nicht möglich, möglichst viele Meinungen zu sammeln. Wiederholte Evaluation mit mehreren Teilnehmern ist wünschenswert.

Außerdem muss man sich über einen langen Zeitraum die Werkzeug probieren, um den Ansatz gut beurteilen zu können.Ein Experte sollte versuchen, eine neue Regel abzuleiten, und ein anderer Experte sollte eine Regel für ein ähnliches Problem ableiten, um zu beurteilen, ob der Ansatz Zeit gespart hat. Dieser Zufall dauert so lange und sollte von mehreren Experten durchgeführt werden.

Hinzu kam das Problem, dass an der CVE-Datenbank kaum wissenschaftlich gearbeitet wurde.

6 Zusammenfassung und Ausblick

6.1 Zusammenfassung

Die Hauptfrage dieser Arbeit ist, ob wir automatisch Regeln für die Schwachstellen aus der CVE-Datenbank ableiten können. Wir haben etwas über SAST-Tools und kryptographische Bibliotheken gelernt. Dann haben wir uns die CVE-Datenbank angesehen.Im Kapitel Studiendesign haben wir erläutert, wie wir die Forschungsfrage beantworten wollten.Dann diskutierten wir den Ansatz , und in diesem Abschnitt zeigten wir, wie der Ansatz erfunden und in die Praxis umgesetzt wurde.anschließend wurde gezeigt, wie die Evaluation ablief, und diskutierten die Ergebnisse.

6.2 Ausblick

Aus meiner Sicht ist eine Erweiterung dieser Arbeit sehr notwendig. Allerdings sollte die Erweiterung in größerem Umfang erfolgen, da künstliche Intelligenz integriert werden soll.

Die Werkzeug soll von viele Experten probiert werden.so kann Die Werkzeug erweitert werden,um selber zu lernen,wie neue Regeln abgleitet werden können.Die neue Regeln dürfen nicht freigegeben werden.Zuerst soll eine Probe durch die Experte erfolgen.

jedoch dauert die Ableitung von einer neuen Regel eine Weile , was bedeutet, dass die Erweiterung erst erfolgen kann , nachdem viele neue Regeln abgeleitet wurden.

Die Erweiterung erfolgt durch die Einbindung künstlicher Intelligenz. Ein Experte kann eine neue Regel für eine Schwachstelle ableiten und im System speichern. Die Werkzeug versucht dann, ähnliche Regeln für ähnliche Probleme abzuleiten. Wenn die Regeln nicht ausreichen, werden weitere Regeln erstellt. Die Regeln sollten immer von den Experten getestet und dazu Berichte erstellt werden.

Eine Erweiterung wäre auch,die vorgeschlagenen Änderungen,die in Kapitel 5 besprochen wurden,umzusetzen.Und auch die Probleme,die im selben Kapitel diskutiert wurden,beheben.

Literaturverzeichnis

[And05] S. H. B. Anda. *Experiences from conducting semi-structured interviews in empirical software engineering research.* 11th IEEE International Software Metrics Symposium (METRICS'05), 2005. ISBN: 0-7695-2371-4. DOI: 10.1109/METRICS.2005.24 (zitiert auf S. 35).

[che] checkdomain.de. *MySQL.* URL: https://www.checkdomain.de/hosting/lexikon/mysql/ (zitiert auf S. 24).

[cir] circl. *API documentation.* URL: https://cve.circl.lu/api/#auth (zitiert auf S. 23).

[Dja] Django -REST-framework.org. *Django REST framework.* URL: https://www.django-rest-framework.org/api-guide/views/ (zitiert auf S. 26).

[dja] djangogirls. *Django.* URL: https://tutorial.djangogirls.org/de/django// (zitiert auf S. 24).

[Ecl] Eclipse Foundation. *Eclipse CogniCrypt.* URL: https://www.eclipse.org/cognicrypt/documentation/ (zitiert auf S. 16).

[MITa] MITRE. *2021 CWE Top 25 Most Dangerous Software Weaknesses.* URL: https://cwe.mitre.org/top25/archive/2021/2021_cwe_top25.html (zitiert auf S. 16).

[MITb] MITRE. *About CWE.* URL: https://cwe.mitre.org/about/index.html (zitiert auf S. 16).

[ora] oracle. *Java Cryptography Architecture (JCA) Reference Guide.* URL: https://docs.oracle.com/javase/8/docs/technotes/guides/security/crypto/CryptoSpec.html#Introduction (zitiert auf S. 15).

[Ros] Roshan Chokshi. *What is an APIView?* URL: https://dev.to/chokshiroshan/what-is-an-apiview-2g8o (zitiert auf S. 26).

[Ryt] Ryte wiki. *HTTP Status Code.* URL: https://de.ryte.com/wiki/HTTP_Status_Code/ (zitiert auf S. 23).

[sel] selfphp.de. *GET und POST.* URL: https://www.selfphp.de/praxisbuch/praxisbuchseite.php?site=183&group=32 (zitiert auf S. 27).

[Son] SonarQube. *SonarQube Documentation.* URL: https://docs.sonarqube.org/latest/ (zitiert auf S. 16).

[tal] talend. *API.* URL: https://www.talend.com/de/resources/was-ist-eine-api/#:~:text=Eine%5C%20API%5C%20(Application%5C%20Programming%5C%20Interface,einem%5C%20externen%5C%20System%5C%20zu%5C%20interagieren./ (zitiert auf S. 23).

[wika] wikipedia. *Bouncy Castle*. URL: https://de.wikipedia.org/wiki/Bouncy_Castle (zitiert auf S. 15).

[wikb] wikipedia. *React*. URL: https://de.wikipedia.org/wiki/React/ (zitiert auf S. 24).

[wikc] wikipedia. *Static application security testing*. URL: https://en.wikipedia.org/wiki/Static_application_security_testing (zitiert auf S. 15).

Alle URLs wurden zuletzt am 31. 05. 2022 geprüft.

Erklärung

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Exemplaren überein.

Ort, Datum, Unterschrift