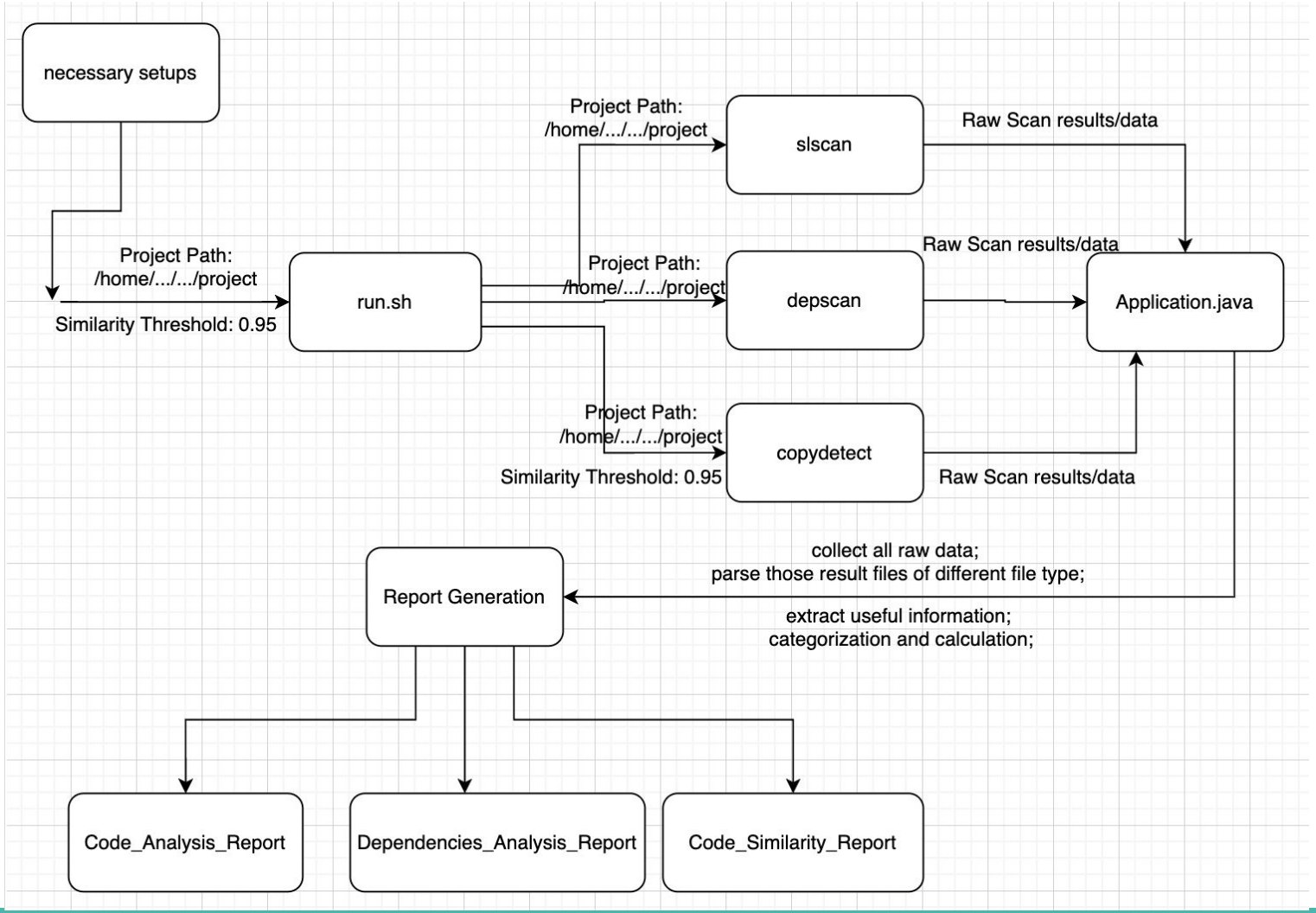
COMSE6156 Project Demo

Tianzhi Huang

Project Overview

- **Functionality**: Scan projects and generate vulnerability reports of
  - Code error, misuse of function, APIs (Static Code Analysis)
  - Dependencies vulnerabilities (Dependency Scan)
  - Code similarity and duplication (Code scan and comparison)
- **Value**: Building a integrated program that could detect several aspects of possible security vulnerabilities and combining results to produce a detailed report.
- **Running Platform**: linux
- **Support Project language**: Python, Java (need to build before being scanned)
- **Prerequisite (setup)**: Python 3.6, Java(JDK,JDK), Docker, 2 jar files (Json-simple, Jsoup)


necessary setups
Project Path:
/home/.../.../project
slscan
Raw Scan results/data
Project Path:
/home/.../.../project
Similarity Threshold: 0.95
run.sh
Project Path:
/home/.../.../project
depscan
Raw Scan results/data
Application.java
Project Path:
/home/.../.../project
Similarity Threshold: 0.95
copydetect
Raw Scan results/data
collect all raw data;
parse those result files of different file type;
Report Generation
extract useful information;
categorization and calculation;
Code_Analysis_Report
Dependencies_Analysis_Report
Code_Similarity_Report

Running example

```
th2888@cs6111vm:~/cs6156$ ./run.sh /home/th2888/cs6156/SANGRIA/service-operation/ 0.95
Project Path Being Scanned: /home/th2888/cs6156/SANGRIA/service-operation/
Code Similarity threshold: 0.95
Start Scanning Processes. This may take some time to finish...
```

```
Generating Analysis Reports...
Reports have been generated in /home/th2888/cs6156/SANGRIA/service-operation/ as Code_Analysis_Report_*, Dependenci
es Analysis Report *, and Code Similarity Analysis Report *
```

```
th2888@cs6111vm:~/cs6156/SANGRIA/service-operation$ ls
'Application$DependenceResult.class'   Dependencies_Analysis_Report_1650963391638.txt
'Application$ScanResult.class'         json-simple-1.1.1.jar
 Application.class                     jsoup-1.14.3.jar
 Application.java                      logs
 Code_Analysis_Report_1650963391572.txt  mvnw.cmd
 Code_Similarity_Analysis_Report_1650963392155.txt  pom.xml
th2888@cs6111vm:~/cs6156/SANGRIA/service-operation$
```

Java Project Code Static Analysis Result

Scanned Project Path: /home/th2888/cs6156/SANGRIA/service-operation/

Code Threat and Vulnerabilities Found Summary:

1).Source File Result:
total 0
high 0
critical 0
low 0
medium 0

2).Infrastructure Security:
total 0
high 0
critical 0
low 0
medium 0

3).Class File Result:
total 4
high 2
critical 2
low 0
medium 0

4).Secrets Audit:
total 0
high 0
critical 0
low 0
medium 0

3.Issue Details:

This API MD5 (MDX) is not a recommended cryptographic hash function
At MD5Utils.java:[lines 6-58]
In class com.sangria.operation.utils.MD5Utils
In method com.sangria.operation.utils.MD5Utils.getMD5(String)
At MD5Utils.java:[line 13]
Value MD5.

File Location: file:///home/th2888/cs6156/SANGRIA/service-operation/src/main/java/com/sangria/operation/utils/MD5Utils.java
At Line 13, code content: MessageDigest md = MessageDigest.getInstance("MD5");

Issue Level: error
Issue Severity: CRITICAL

Project Dependencies Analysis Report

Scanned Project Path: /home/th2888/cs6156/SANGRIA/service-operation/

Dependency Analysis Found Vulnerabilities Summary:

MEDIUM 4
LOW 9
CRITICAL 3

2.
ID: CVE-2022-22950
Package Name: org.springframework:spring-core
CVSS Score (A higher score indicates higher severity): 2.0
Severity: LOW
Related Urls:
1.https://nvd.nist.gov/vuln/detail/CVE-2022-22950
2.https://tanzu.vmware.com/security/cve-2022-22950
Threat Description:
Allocation of Resources Without Limits or Throttling in Spring Framework
In Spring Framework versions 5.3.0 - 5.3.16, 5.2.0 - 5.2.19, and older unsupported versions, it is possible for a user to provide a specially crafted SpEL expression that may cause a denial of service condition.

```
Code Similarity Analysis Report

Scanned Project Path: /home/th2888/cs6156/SANGRIA/service-operation/

Code Similarity Found Summary:

Number of files tested: 154
Number of reference files: 154
Test files above display threshold: 26 (16.88%)
```

2).
Test file: /home/th2888/cs6156/SANGRIA/service-operation/src/main/java/com/sangria/operation/dto/InventoryClearDTO.java (100.00%)
Reference file: /home/th2888/cs6156/SANGRIA/service-operation/src/main/java/com/sangria/operation/dto/PlayerDeleteDTO.java (100.00%)
Token overlap: 76

Similar Code Snippet:

```
package com.sangria.operation.dto;

import lombok.Data;

@Data
public class InventoryClearDTO {

        String token;
        String inventoryId;

}


package com.sangria.operation.dto;

import lombok.Data;

@Data
public class PlayerDeleteDTO {

    String token;
    String playerId;

}
```

Possible work to do

- Evaluation program to evaluate data in order to verify RQ
- Combining three reports into one
- Better program structure organization (separate files in terms of different functionalities)
- ...