2024 한양대학교

# SW/ICT/AI 종합학술대회

[분야 : 연구인턴/산학연구개발]

## 제목: 보안 취약점 탐지 및 관리의 효율성을 높이는 도구 개발

소지연(22학번)

2024. 11. 21

한양대학교 ERICA 소프트웨어융합대학 / 한양대학교 ERICA SW중심대학사업단

# Reasons for starting the project

## ❖ 헥사휴먼케어: 웨어러블 로봇 연구개발 및 생산하는 회사

1. 로봇 움직임 정의 및 제어를 위해 c/c++ 프로그래밍 언어 사용
2. 소스코드의 보안 취약점을 탐지하기 위해 Checkmarx 사용

***왜 Checkmarx를 사용할까?***

한국인터넷진흥원인 KISA가 상용 SAST 도구인 Checkmarx를 기준으로 소프트웨어의 보안 취약점을 검사 및 평가하기 때문이다

*** KISA(Korea Korea Internet & Security Agency)**
대한민국의 인터넷 및 정보보호를 총괄하는 기관으로, 안전하고 신뢰할 수 있는 디지털 환경의 조성 뿐만 아니라 정보보호 안전망 구축과 디지털 인프라 안정성 확보에 앞장서고 있다.

## ❖ 문제상황

Checkmarx로 개발한 소스코드를 검사하기 위해서는 물리적 이동 필요
= 즉각적인 보안 취약점 발견 불가 + 개발 진행에 지연 발생

***자체적으로 운영 가능한 보안 취약점 탐지 솔루션의 개발 필요성 강조***

HEXAR

# What is ‘Checkmarx’?

## ❖ Code, Supply Chain, Cloud 분야로 세분화

Code 분야에서 SAST(Static Application Security Testing) 솔루션 제공하여 소스코드 분석 및 잠재적인 보안 취약점 탐지

## ❖ 다양한 보안 표준 및 규정을 기반으로 검사

| | |
|---|---|
| OWASP Top 10 2013, 2017, 2021 | OWASP(Open Web Application Security Project)에서 취약점 발생 빈도와 심각도를 기준으로 발표하는 가장 흔하고 치명적인 웹 애플리케이션 보안 취약점 10가지 목록 |
| PCI DSS v3.2.1 | 신용카드 결제와 관련된 데이터 보안을 강화하기 위해 국제 카드 브랜드(MasterCard, American Express, VISA, 등)가 공동으로 개발한 보안 표준 |
| NIST SP 800-53 | 미국 국립표준기술연구소(NIST)에서 발행한 보안 및 개인정보 보호 통제 지침 |
| ASD STIG 4.10, 5.2 | 호주 ASD(Australian Signals Directorate)에서 제공하는 보안 가이드라인 |
| MOIS(KISA) Secure Coding 2021 | 대한민국 행정안전부(MOIS)와 KISA(한국인터넷진흥원)가 공동으로 발행한 안전한 소프트웨어 개발을 위한 보안 코딩 가이드라인 |

*이 외에도 FISMA 2014, SANS top 25, CWE top 25, OWASP Mobile Top 10 2016 등을 기준으로 검사 수행*

# CWE vs. CVE

## CWE

Common Weakness Enumeration의 약자로, 소프트웨어 보안 약점을 분류하고 설명하기 위해 MITRE가 주도하여 만든 공통 약점 목록

- ✓소프트웨어 개발자, 보안 전문가 등이 시스템 설계와 코드 속 구조적 취약점 식별 및 개선 시 사용
- ✓코딩 오류, 설계 결함, 구현 실수와 같은 취약점을 체계적으로 분류
- ✓취약점에 대해 공통된 언어를 사용하여 효과적인 협력을 도움

## CVE

Common Vulnerabilities and Exposures의 약자로, 공개된 소프트웨어 보안 취약점 및 노출을 표준 형식으로 식별하고 기술하는 공통 취약점 목록

- ✓ 각 CVE에 해당하는 이름, 설명, 영향도 및 해결방법 등 제공
- ✓기업이 사용하는 소프트웨어의 취약점을 쉽게 탐지하고 대응하도록 지원
- ✓취약점이 발견된 년도와 임의의 번호 부여한 CVE ID를 통해, 전 세계에서 유일하게 식별가능한 번호 제공

***SAST(정적 코드 분석) 도구는 CWE 분류를 기반으로 코드 점검***

# Choose open source to utilize

## ❖ Cppcheck

C 및 C++ 코드를 정적 분석하는 오프소스 취약점 분석 도구로,
잠재적인 버그와 코드 품질 문제 발견

1) **무료로 제공되며, 누구나 사용할 수 있는 오픈소스 도구**

   비용 부담 없이 다운로드 및 사용이 가능하고, 원하는 방식대로 수정하거나 추가하는 등의 커스터마이징이 가능하다.

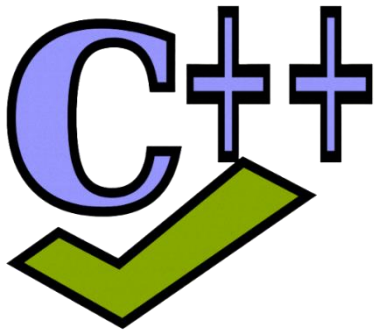
2) **Windows, macOS, Linux 등 여러 운영 체제에서 사용 가능**

3) **코드를 실행하지 않고도 소스 코드 검토하는 방식**

   컴파일러가 탐지하지 못하는 Null 포인터 참조, 배열 경계 초과, 메모리 문제 등을 분석하는데 특화되어 있다.

4) **다양한 설정 옵션**

   특정 검사 규칙을 활성화하거나 비활성화할 수 있어 사용자 맞춤형 분석이 가능하다.



# Use of code analysis tools

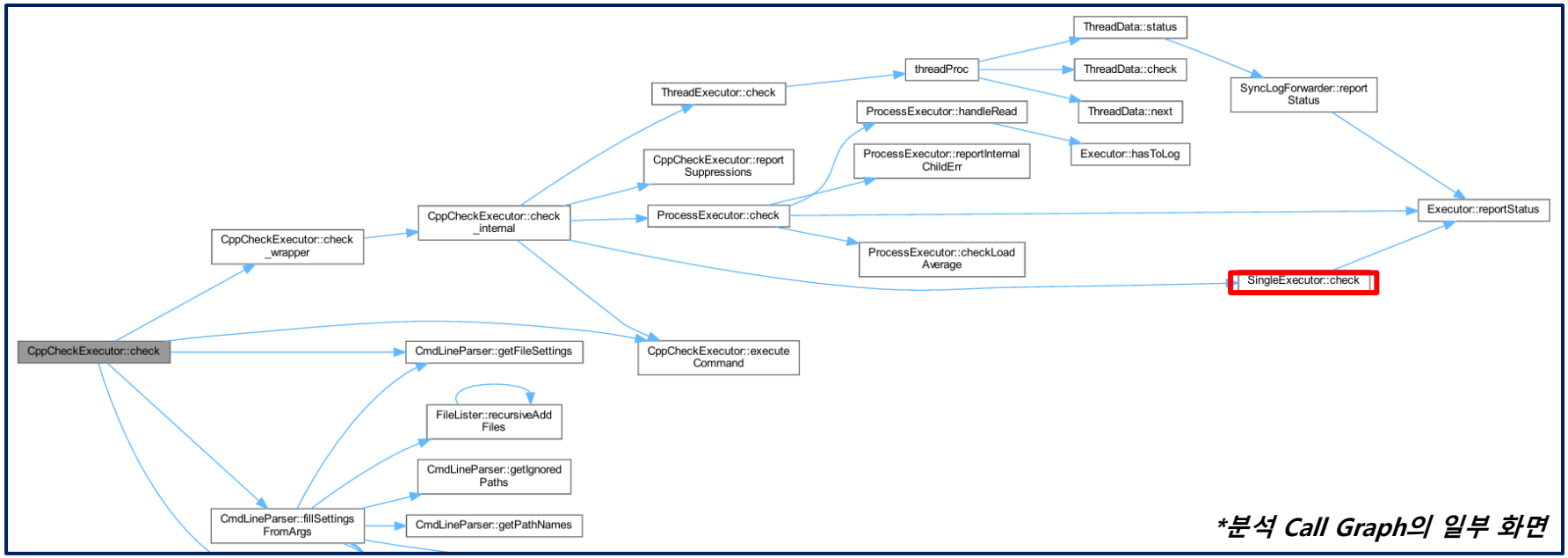
❖ **Graphviz**

그래프(노드와 엣지) 데이터를 시각화하기 위한 오픈소스 도구

❖ **Doxygen**

C, C++, Python, Java 등 다양한 프로그래밍 언어로 작성된 코드를 문서화하기 위한 오픈소스 도구

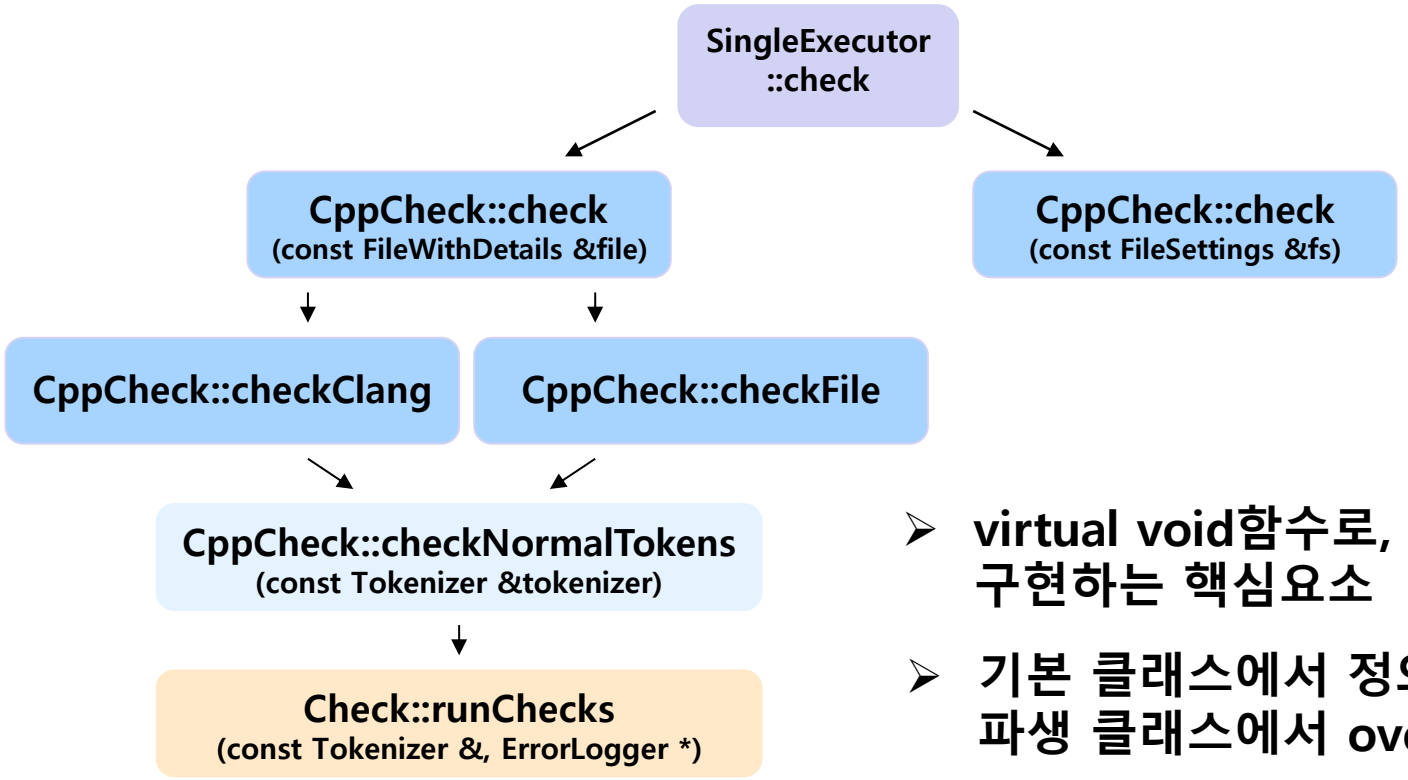
***클래스 다이어그램, 호출 그래프 등을 통해 코드 구조 파악 용이***




**분석 Call Graph의 일부 화면*

# Vulnerability detection flow

## ❖ SingleExecutor::check




```
/** run checks, the token list is not simplified */
virtual void runChecks(const Tokenizer &, ErrorLogger *) = 0;
```

- **virtual void함수로, 다형성을 구현하는 핵심요소**
- **기본 클래스에서 정의된 함수를 파생 클래스에서 override 가능**

→ lib/check*.cpp의 각각 내부에 구현된 runChecks() 실행

*Ex) CheckBufferOverrun::runChecks, CheckNullPointer::runChecks, CheckSizeof::runChecks 등등*

# Future plans

## ❖ 검사 수행되는 CWE 정리 및 추가 적용 CWE 선정

1. Checkmarx에서 검사되는 CWE들과 Cppcheck에서 검사되는 CWE 비교
2. 오픈소스로 공개된 Cppcheck에 추가할 CWE 선정 및 적용

## ❖ Cppcheck의 분석 결과 커스터마이징

1. Cppcheck의 소스코드 분석 결과 형식 확인
2. Checkmarx와 관계된 결과를 얻을 수 있도록 옵션 추가 혹은 형식 변환
3. 보안적인 측면에서 해당 도구를 활용하여 Checkmarx를 기반으로 개발하는 기업들에 도움

