Security Forum 2019

MICRO FOCUS

포티파이 for Future AppSec

양치기 소년에 비유해 본 포티파이

김상현 Security Expert

양치기 소년 이야기

이솝 우화의 이야기로 **양이란 동물들은 온순함의 상징처럼 여겨지지만 제 멋대로에 성깔까지 더러운 동물**이다. 만만한 초짜 양치기라면 작정하고 들이받는 일이 다반사다. 실제로 중세 유럽에서는 양치기가 양에게 들이받혀 사망하는 일이 있었고, 충차를 '들이받는 양(Battering RAM)'이라고 부를 만큼 양의 돌진력은 무시할 수 없다.

따라서 **우화에서와 달리 양치기는 매우 한가롭고 따분한 직업이 아니라 굉장히 고달프다**. **심심해서 그랬는지 혹은 고달파 스트레스를 받아서 그랬는지 모르지만 늑대가 왔다고 거짓말을 해 사람들을 여러 번 속여 먹었으며** 그 뒤 진짜 늑대가 오자 사람들에게 늑대가 왔다고 말했지만...

이번에도 소년이 거짓말을 하는 줄 알고 **아무도 오지 않아 양들 모두 (또는 양치기 소년까지) 늑대에게 잡아먹혔다는 내용**.

늑대는 양을 호시탐탐 잡아먹으려고 하며, 판본에 따라서는 마을 사람들이 다시 왔으나 **늑대를 막질 못해서 양이 몰살당하거나 심지어 양치기 소년이 늑대에게 잡아먹히는 경우가 있다**.

출처: 나무위키 – https://namu.wiki/w/%EC%96%91%EC%B9%98%EA%B8%B0%20%EC%86%8C%EB%85%84

Who’s who…

어플리케이션 (코드)

- 태생적으로 취약점을 내포하여 보호하여야 할 대상

코드 분석기 (포티파이)

- 늑대가 나타나거나 취약한 양이 있으면 마을 사람들에게 보고
- 필요한 경우 늑대를 바로 물리칠 수 있으면 마을 사람들에게 더욱 사랑 받음

개발자

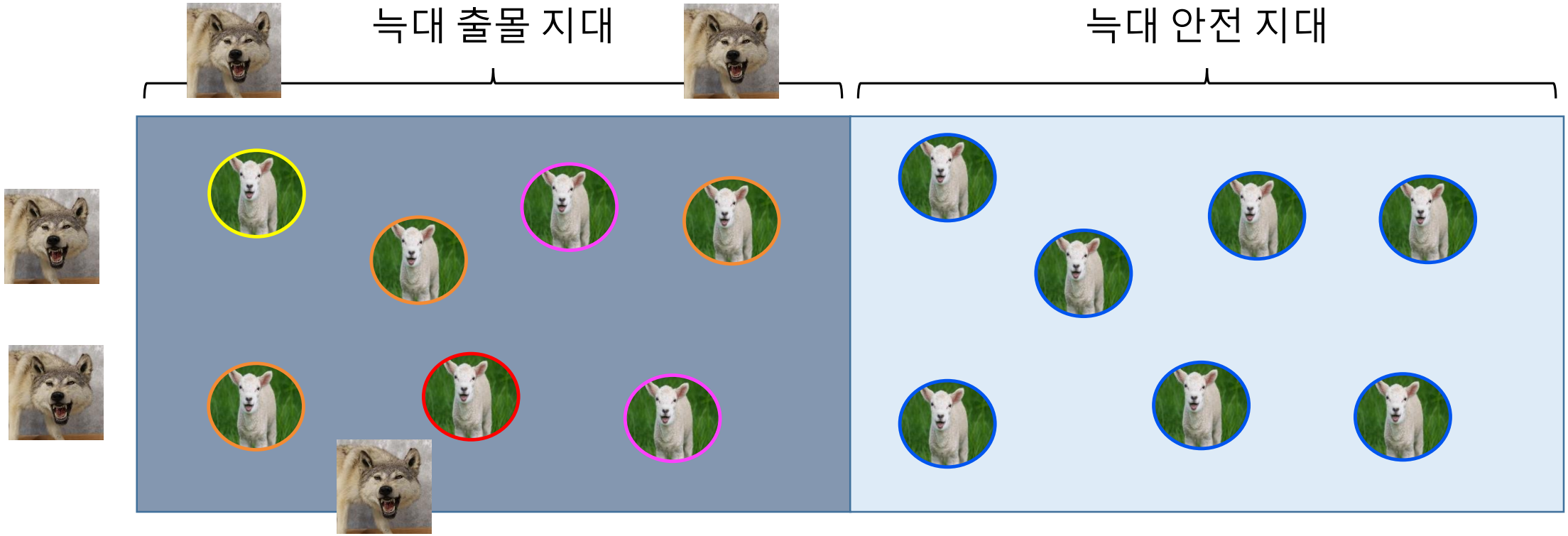
- 양의 실제 주인
- 건강 상태가 안 좋은 양도 많고, 가끔은 자기가 몇 마리의 양을 키우는지 모름

위협 (aka. 해커)

- 기회가 오기만을 기다리다 양을 잡아 먹으려고 함
- 크고 튼실한 양이면 더욱 좋고 Alpha Male을 잡으면 양 떼를 혼란에 몰아 넣을 수 있음

그럼 양치기 소년 이야기를 어플리케이션 보안 (AppSec)과 어떻게 연관지을 수 있을까요?

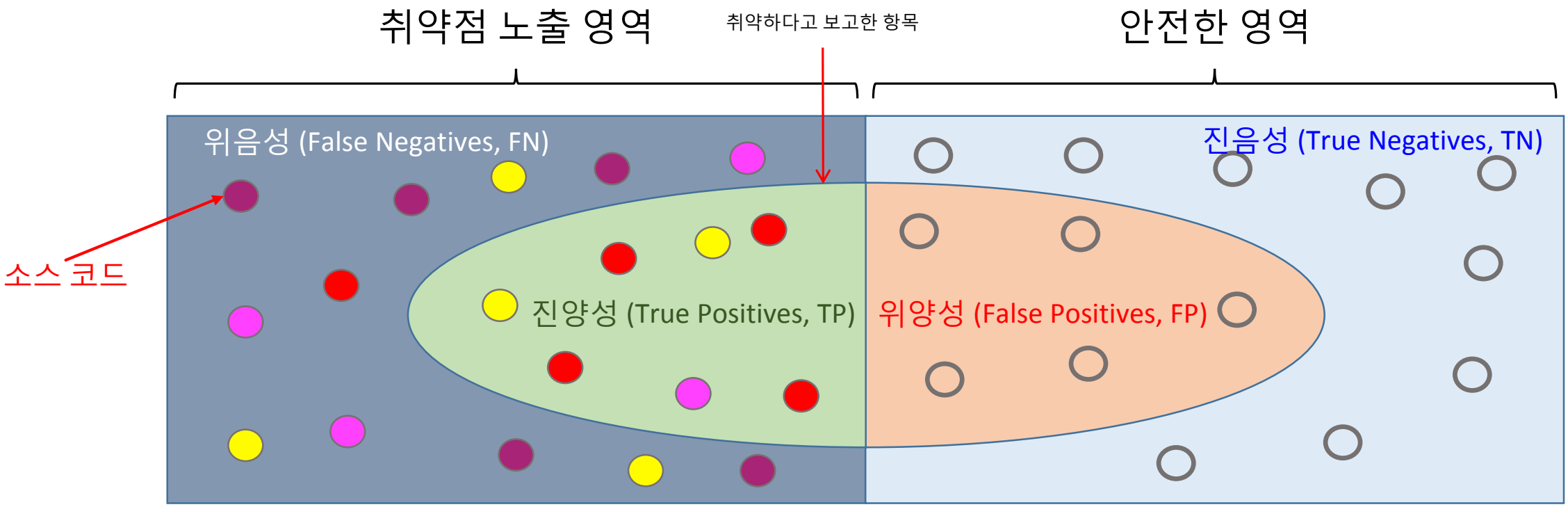
양치기 소년의 단순 일과



비실대는 양을
발견하거나,
늑대가 나타나면
소리치기!

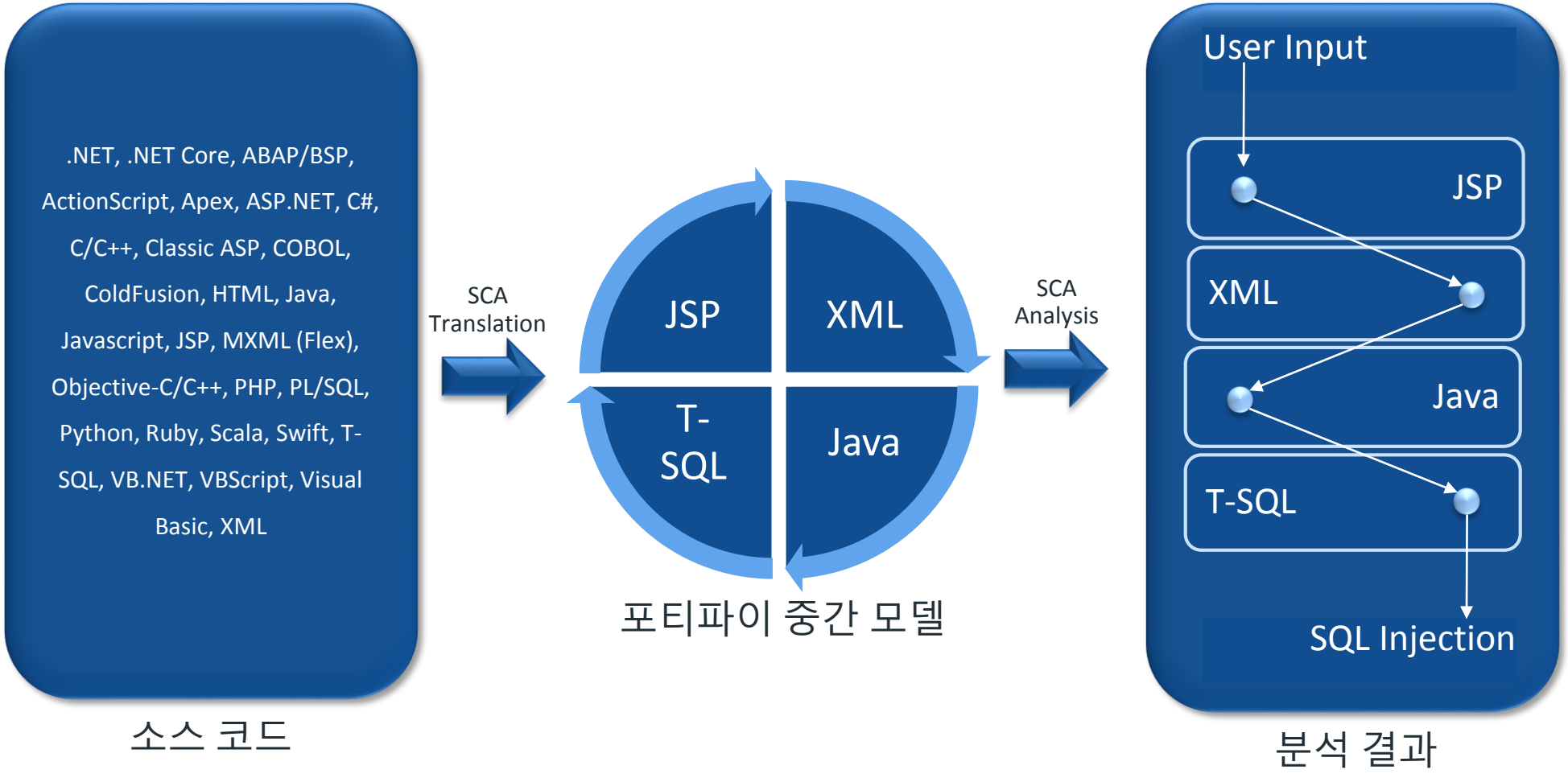
보더콜리랑
놀면서 유유자적

그렇다면 AppSec은?



- AppSec 툴 중 대표적인 **소스 코드 분석기가 수행하는 취약점 분석 결과를 도식화한 것**입니다.
- **모든 취약점을 검출할 수는 없습**니다.
  - **취약점 노출 영역 중 일부만 검출 (좌측 타원)**
- 때로는 안전한 영역에 있거나 **취약하지 않은 항목도 취약하다고 합니다**.
  - **우측 타원**
  - **우리가 아는 그 양치기 소년?**

코드 분석 툴인 포티파이의 Cross-language 분석

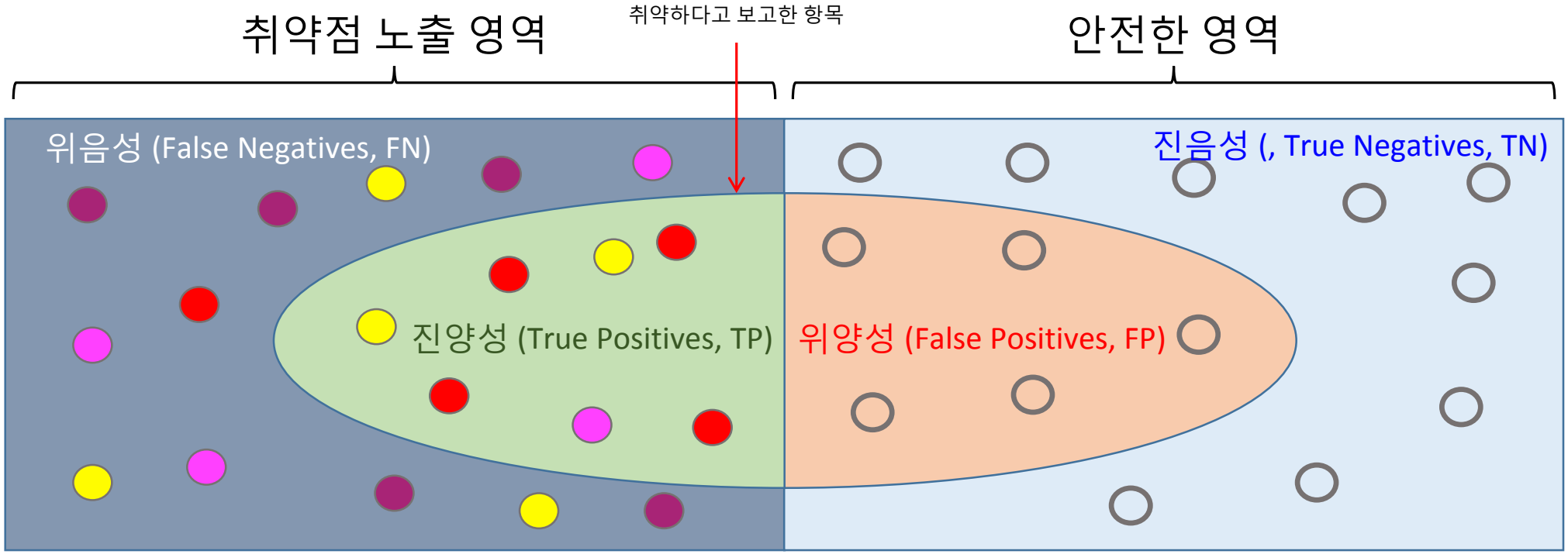


소스 코드

포티파이 중간 모델

분석 결과

코드 분석 지표 (Metrics)



진양성률, 정탐률
True Positive Rate
(Sensitivity, Recall)

How many of REAL vulnerabilities are ACTUALLY reported as vulnerable?

$$= \frac{\#\text{진양성}}{\#\text{위음성} + \#\text{진양성}}$$

정확도
Precision

How many of REPORTED vulnerabilities are REALLY vulnerable?

$$= \frac{\#\text{진양성}}{\#\text{진양성} + \#\text{위양성}}$$

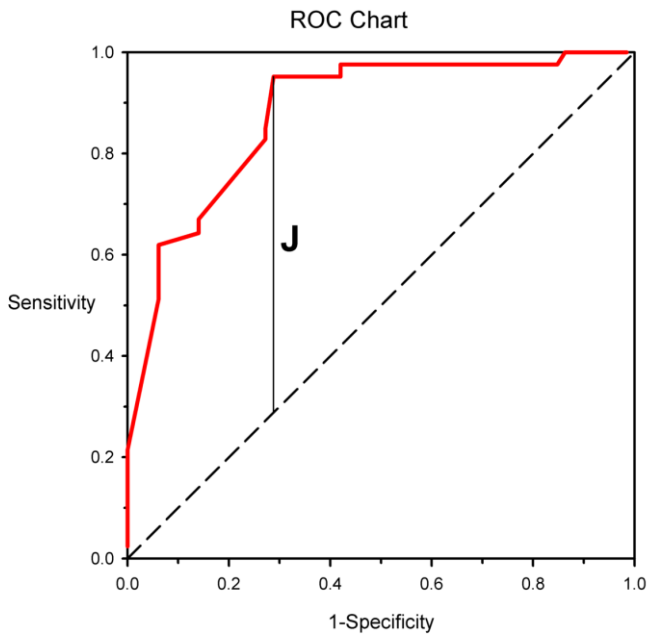
위양성률, 오탐률
False Positive Rate
(1-Specificity, Fall-out)

How many of non-vulnerabilities are INCORRECTLY reported as vulnerable?

$$= \frac{\#\text{위양성}}{\#\text{위양성} + \#\text{진음성}}$$

Youden Index란

- Youden's J statistic (also called Youden's index) is a single statistic that captures the performance of a dichotomous diagnostic test.
- Youden 인덱스는 이분법적인 진단 (참/거짓)의 정확성을 판단하는데 사용되는 통계적인 수치



$$J = \text{sensitivity} + \text{specificity} - 1 = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}} + \frac{\text{true negatives}}{\text{true negatives} + \text{false positives}} - 1$$

https://en.wikipedia.org/wiki/Youden%27s_J_statistic

True Positive Rate (Sensitivity, Recall) = #TP / (#TP + #FN)

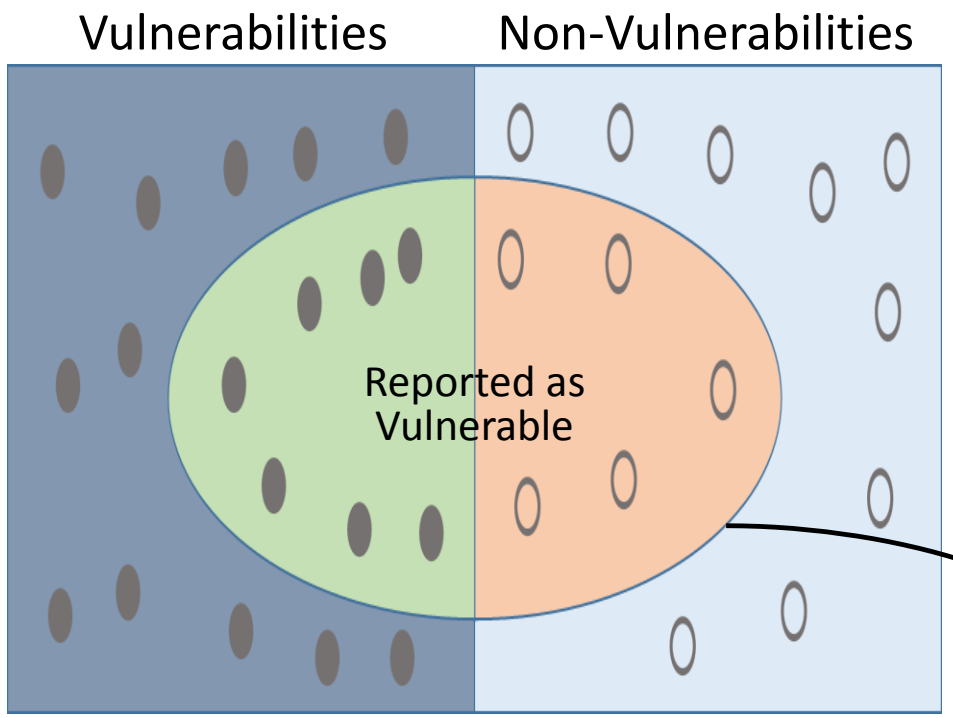
How many of REAL vulnerabilities are ACTUALLY reported as vulnerable?

False Positive Rate (1-Specificity, Fall-out) = #FP / (#FP + #TN)

How many of non-vulnerabilities are INCORRECTLY reported as vulnerable?

즉, 직관적으로 **정탐율 – 오탐율**로 이해될 수 있습니다.

좋은 소스 코드 분석기 -> 높은 Youden Index

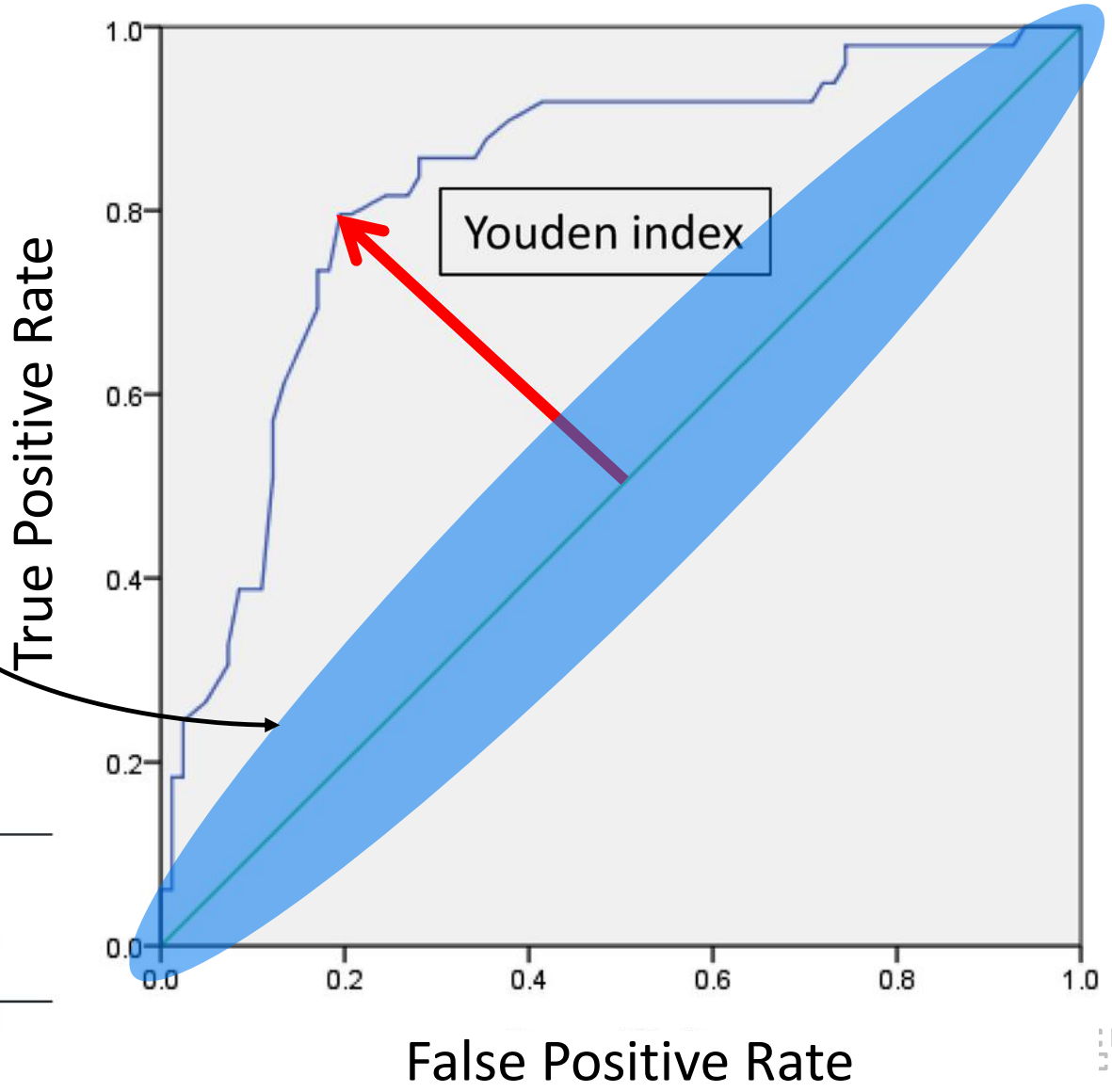


True Positive Rate (Sensitivity, Recall) = #TP / (#TP + #FN)

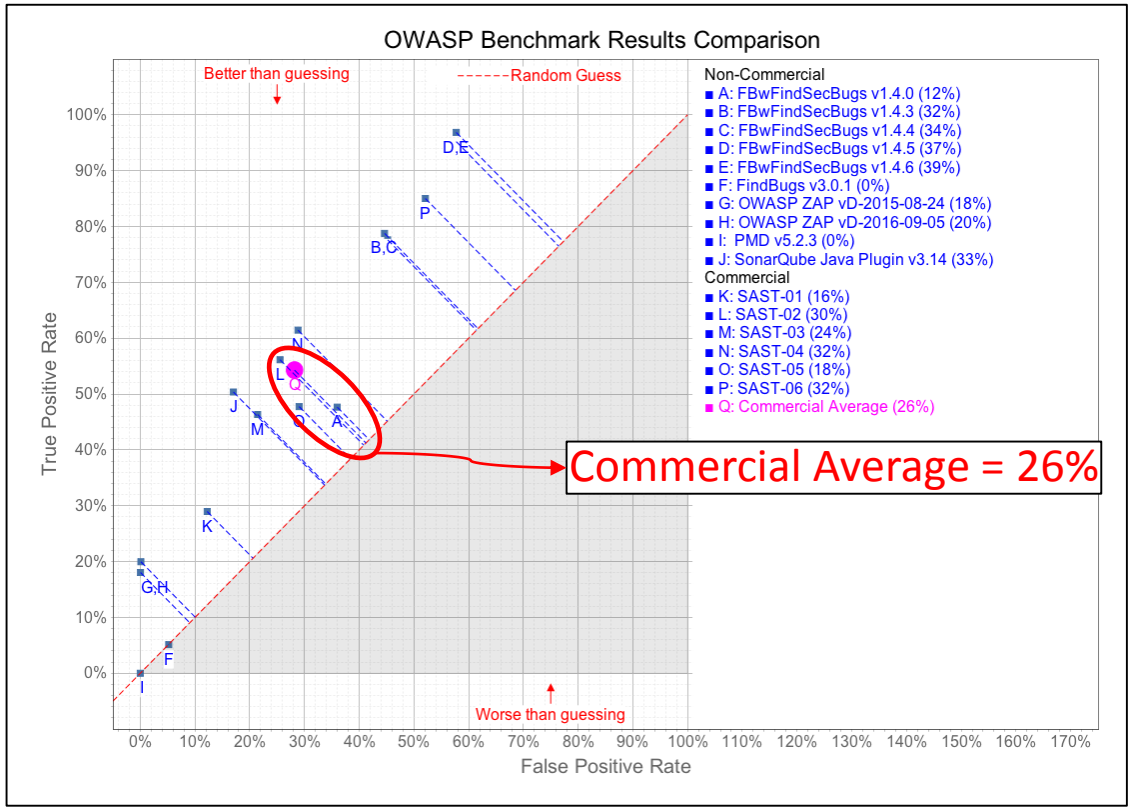
How many of REAL vulnerabilities are ACTUALLY reported as vulnerable?

False Positive Rate (1-Specificity, Fall-out) = #FP / (#FP + #TN)

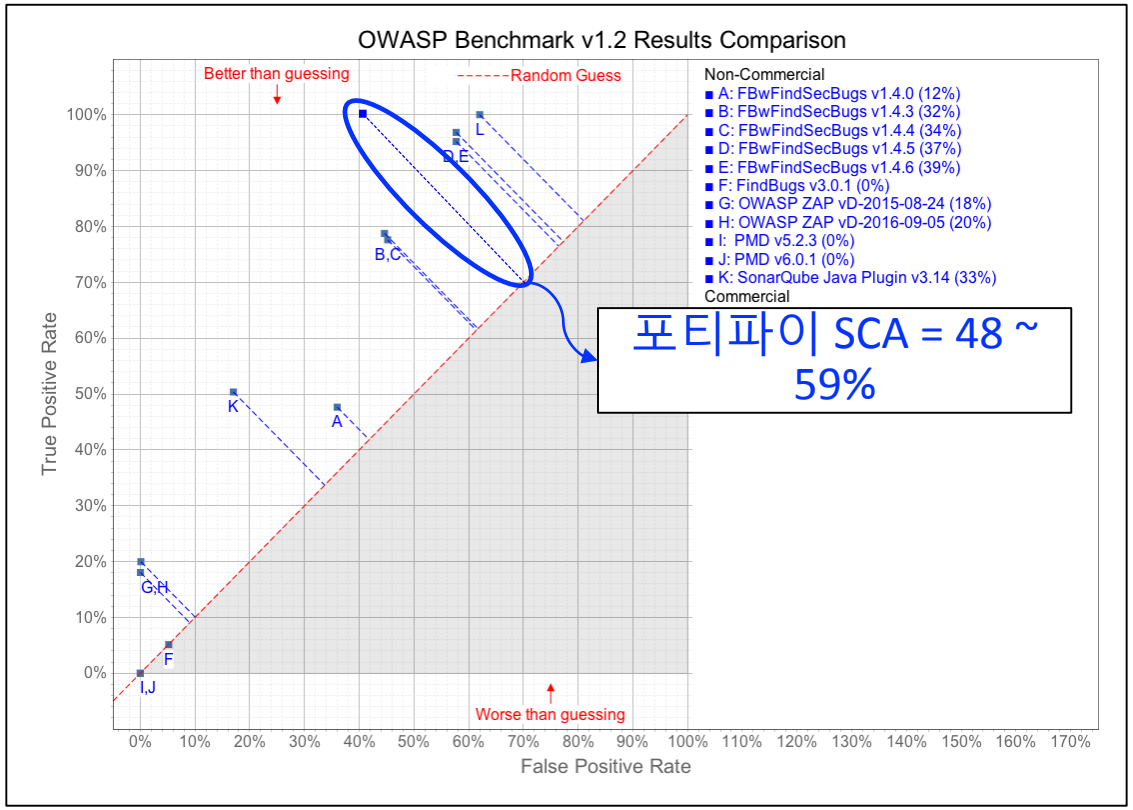
How many of non-vulnerabilities are INCORRECTLY reported as vulnerable?



FYI: 포티파이 SCA's Youden Index



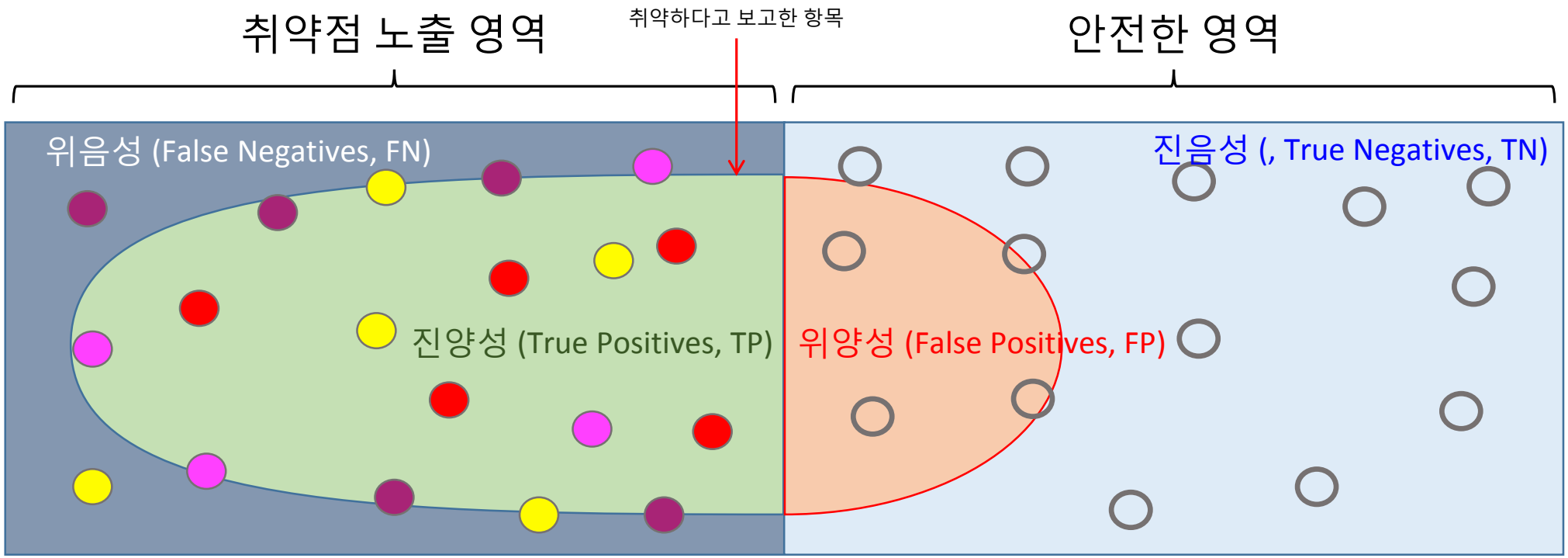
vs.



https://www.owasp.org/index.php/Benchmark

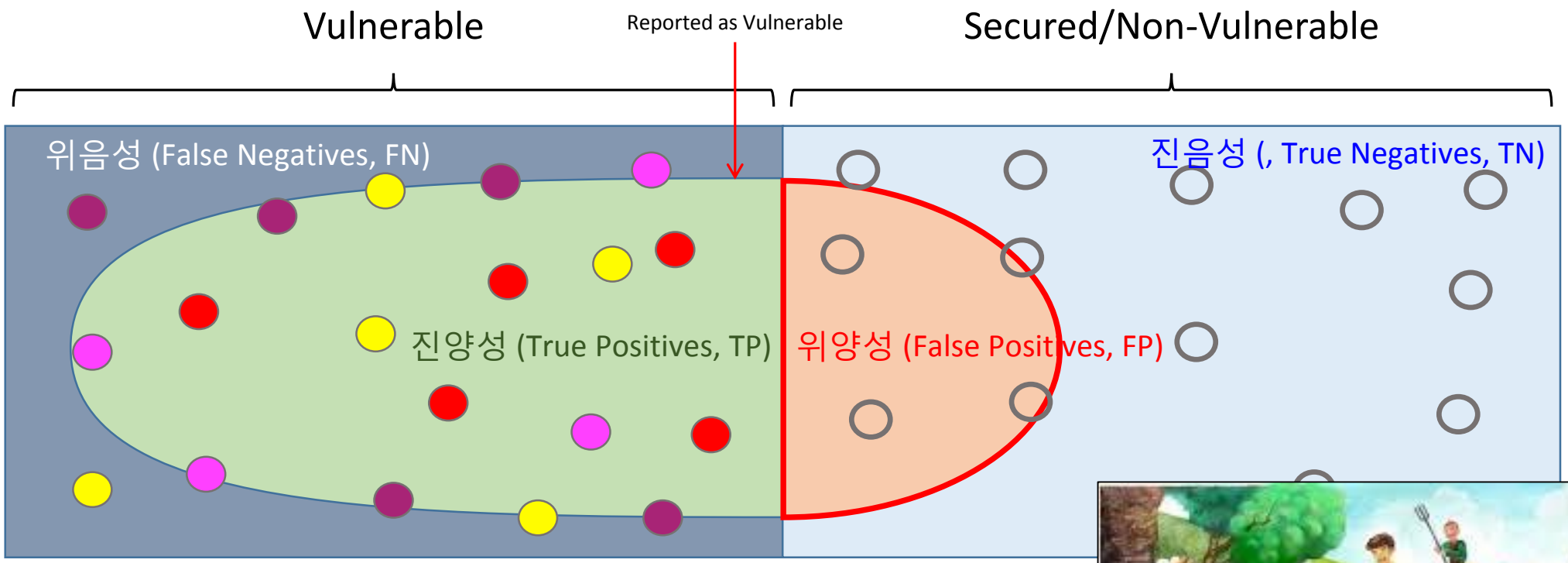
Micro Focus Software Security Research White Paper

정탐 vs. 오탐 다이어그램 지향점



- 소스 코드 분석기가 나아갈 방향은 명확합니다.
- **가능한 많은 취약점을 놓치지 말고 검출**하여야 하며, **정상 처리에 대해서는 과하게 검출하지 말아야 합니다**.
- 따라서 **전체적인 검출 영역이 왼쪽으로 움직이게** 됩니다.

포티파이 – 반복되는 과탐 문제



- 하지만 **조금의 과탐에도 개발자들은 민감하게 반응**합니다.
- **귀한 시간과 노력을 헛되게 하니까요.**

포티파이 - 또 다른 양치기 소년?

- 과탐이 반복되면 결국에는 신뢰를 잃게 될 것을 자명합니다.
- **포티파이가 양치기 소년**이 아닐까요?

Atlassian's DevSecOps / SAST Section – 양치기 소년의 교훈

Alongside detecting violations in coding best practices, static code analyzers detect security vulnerabilities in code that you own and in (possibly insecure) libraries that you import. This is called **SAST (static analysis security testing)** and modern tools integrate well with the continuous delivery pipeline. Make sure you choose a SAST scanner that's compatible with the programming language of your choice.

A word of caution: SAST can often report false positives and hence plan for a layer of persistence that helps pipelines "remember".
False positives can annoy the team to the point where they stop responding to broken pipeline notifications, and that's dangerous. Once teams have identified an error as a false positive with proper justification, don't let the pipeline flag it again and again. This can lead to teams disabling SAST or letting pipelines ignore SAST errors altogether.

오탐은 개발팀을 성가시게 함으로써 DevOps 파이프라인에 발생하는 문제에 반응하지 못하도록 하며, 이는 매우 위험한 일입니다. 개발팀이 문제를 **일단 오탐으로 인지하게 되면 Pipeline이 그것을 반복해서 무시**하도록 하며, 이것은 **정적 코드 분석을 아예 중지하거나 보안 결함을 전적으로 무시하는 결과**에 이를 수 있습니다.

Machine Learning Assisted Auditing

몇가지 관련 수치들

전체 취약점 검출: **19,352,393**

고객이나 분석가에 의해 Suppress됨: **6,085,610**

잔여 취약점: **13,266,783**

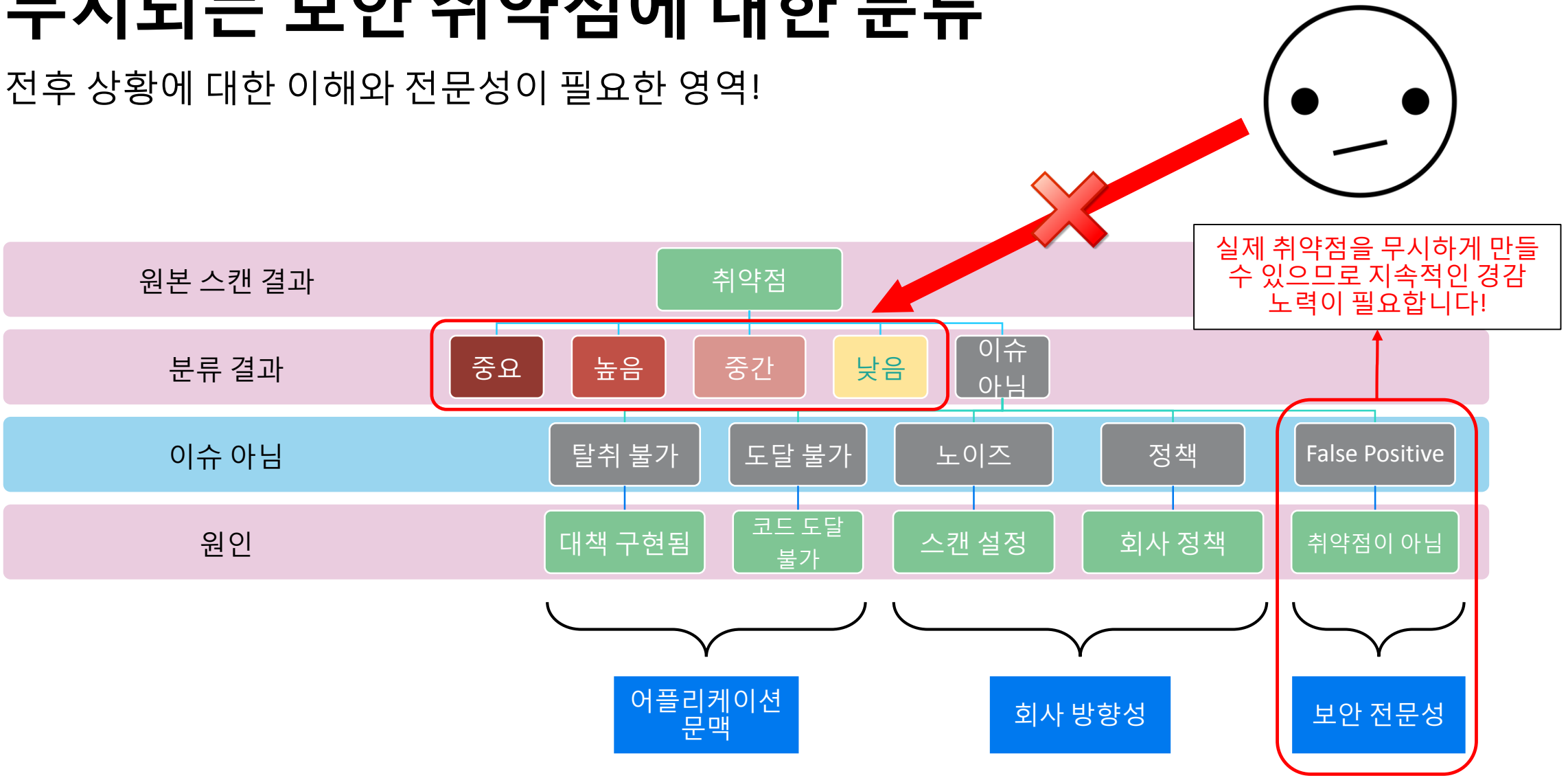
출처: 포티파이 On-Demand (FoD) 연구팀, 2018년

31% overall suppression rate

- Suppress 된 취약점 중 95%는 어플리케이션 맥락에 대한 이해없이 이루어짐
- 5%만이 기술적으로 False Positive로 판명
  - **하지만 이 5%가 실제 취약점을 무시하는 결과로 이어지지 않도록 지속적으로 개선 중입니다.**

무시되는 보안 취약점에 대한 분류

전후 상황에 대한 이해와 전문성이 필요한 영역!

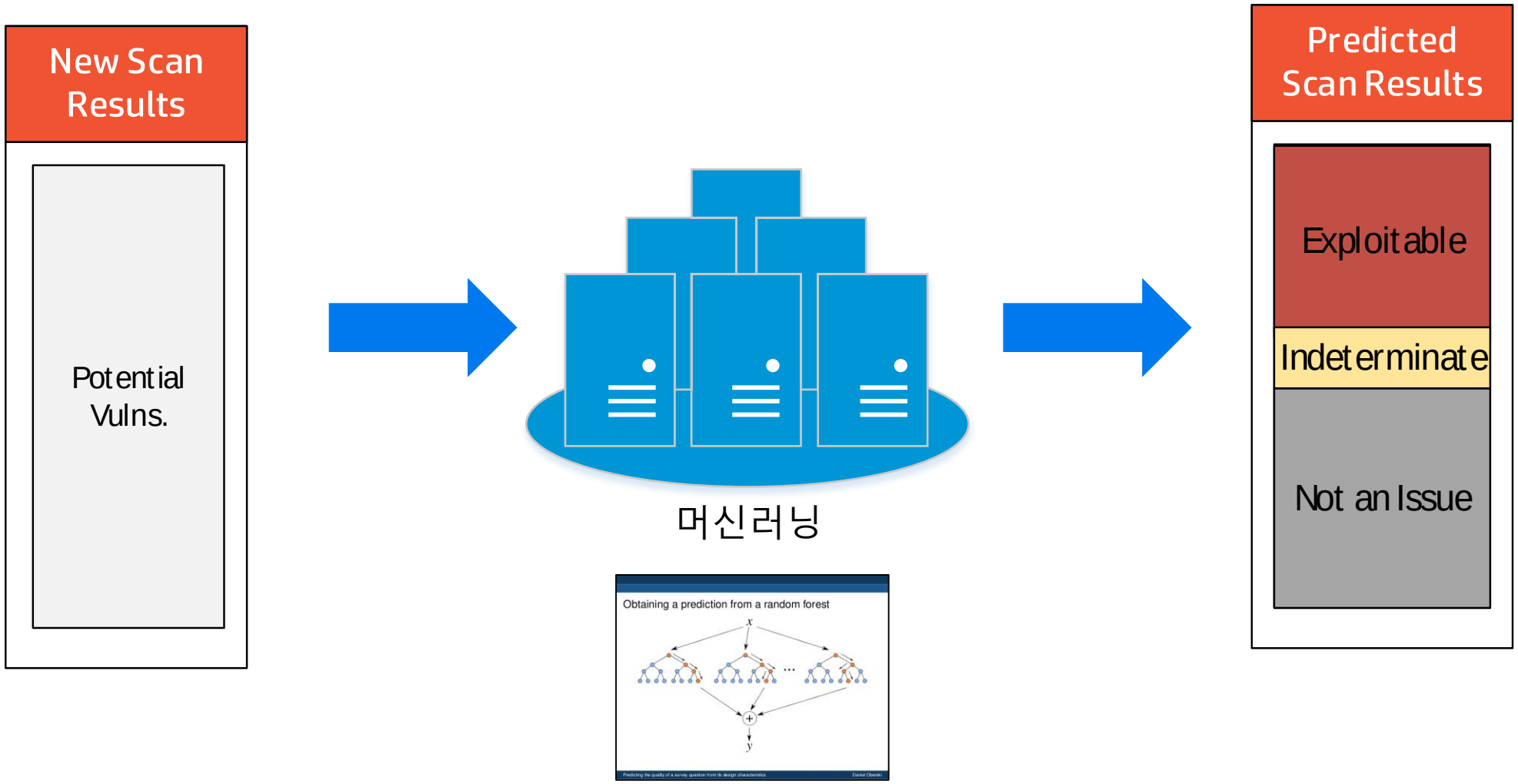


포티파이 머신러닝 – Audit Assistant 기술

머신러닝을 통한 취약점 분석/예측

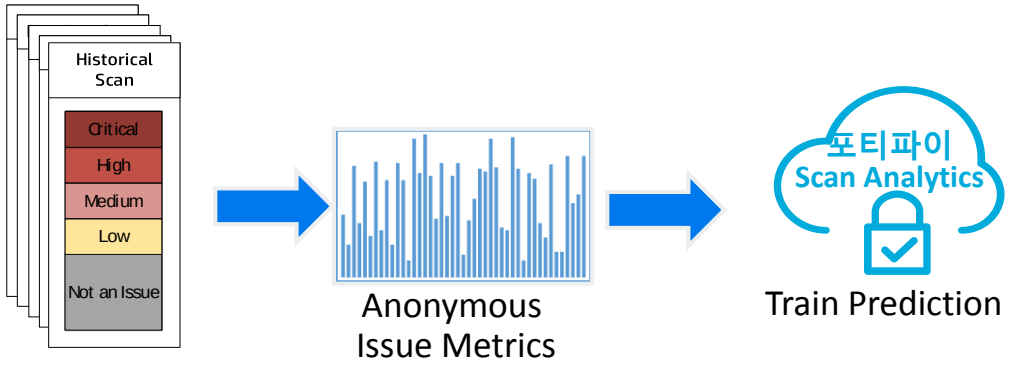
머신러닝을 통한 취약점 분석 – 포티파이 Audit Assistant

Machine learning assisted identification of relevant scan results

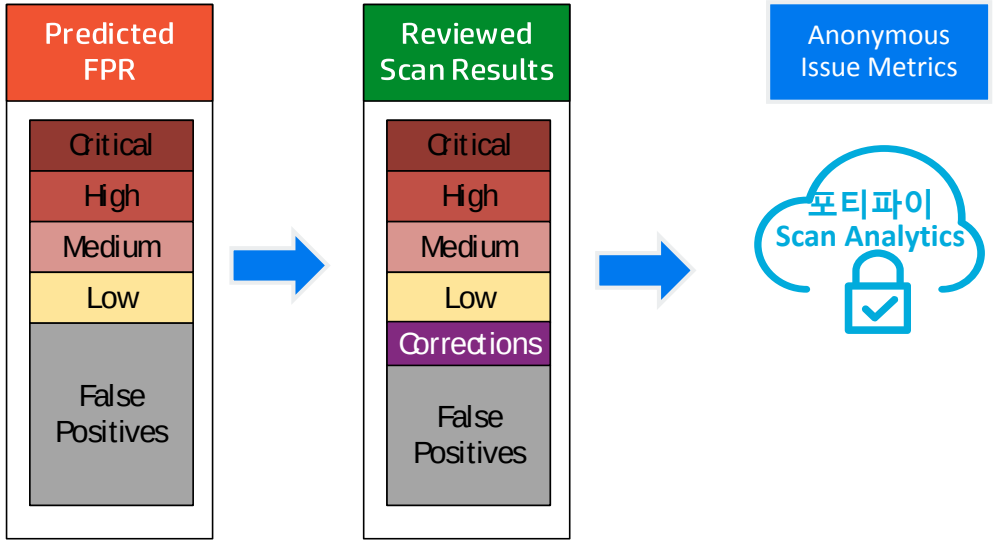


조직별 특성에 따른 머신러닝 학습

Training



Corrections

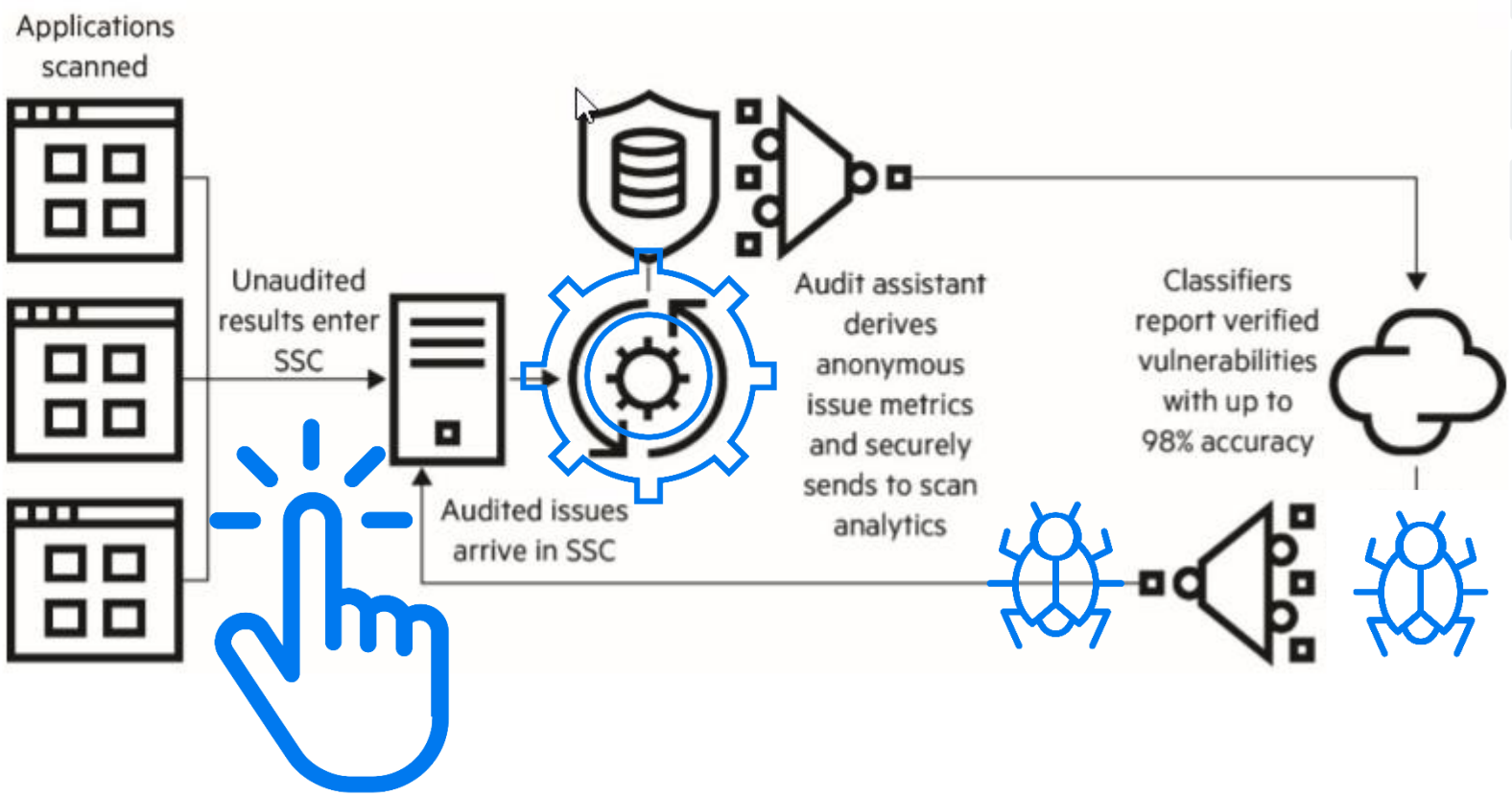


- 필요할 경우 **예측된 결과를 수정하고 다시 학습 데이터로 Feedback**할 수 있으며, 이를 통해 각 **개발 조직의 특성을 반영**할 수 있습니다.

溫故而知新 – 과거를 밝혀 미래를 더 잘 알다

- 보안 취약점의 **중요도 판별 기준**
  - **영향도 (Impact):** How scary is this issue?
  - **개연성 (Likelihood)**: How likely is it that someone can exploit this?
- **중요도와 익명화(Anonymized)된 소스 메타 정보를 바탕으로 각 이슈의 Fingerprint 정보를 생성**합니다.
- **“Random Forests”와 같은 머신러닝 알고리즘은 과거의 기술적/문맥적 분석 정보를 활용하여 보안 이슈의 참/거짓에 대한 예측치를 제공**합니다.
- **과거의 분석 결과를 학습함**으로써 **향후 분석의 정확도를 높일 수 있습**니다 - Past Audits Powering Future Audits!

포티파이 머신러닝 Automation



- 자동 학습
- 자동 예측
- 오토 태깅

포티파이 머신러닝 도입효과

Available Now

- 머신러닝 내장
- 취약점이 가장 빈번 데이터 및 제어 흐름 지원
- **분석 시간 35% 단축**
- **API를 통한 자동화**
- **내장 자동화 since 18.1**

Leading financial services group in Asia
Single Project Case Study:

- **감사가 필요한 이슈 37% 감소**
- Over 3000 of 8000 issues predicted
- 2 full days of audit time saved on 1 release

World's largest oil & gas company

"Scan Analytics helps our team greatly reduce time spent in analyzing false positives."

World's largest software company
Automatically auditing 20% of findings

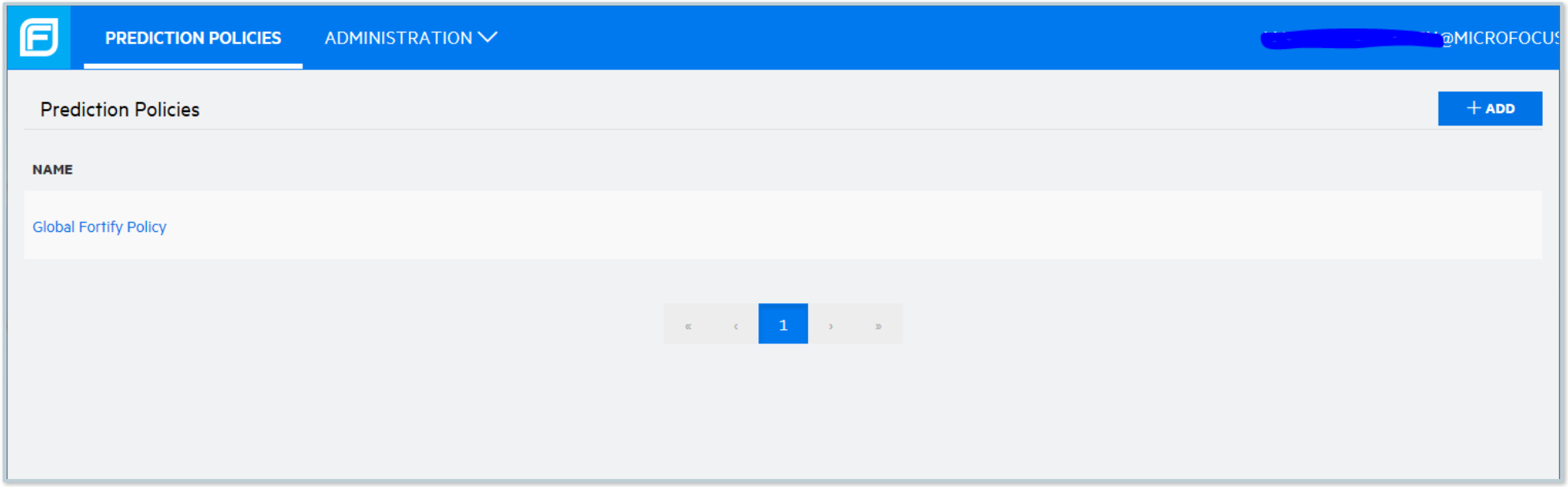
MICRO FOCUS

Fortify

머신러닝 Update – 2019년 5월

편의성 강화

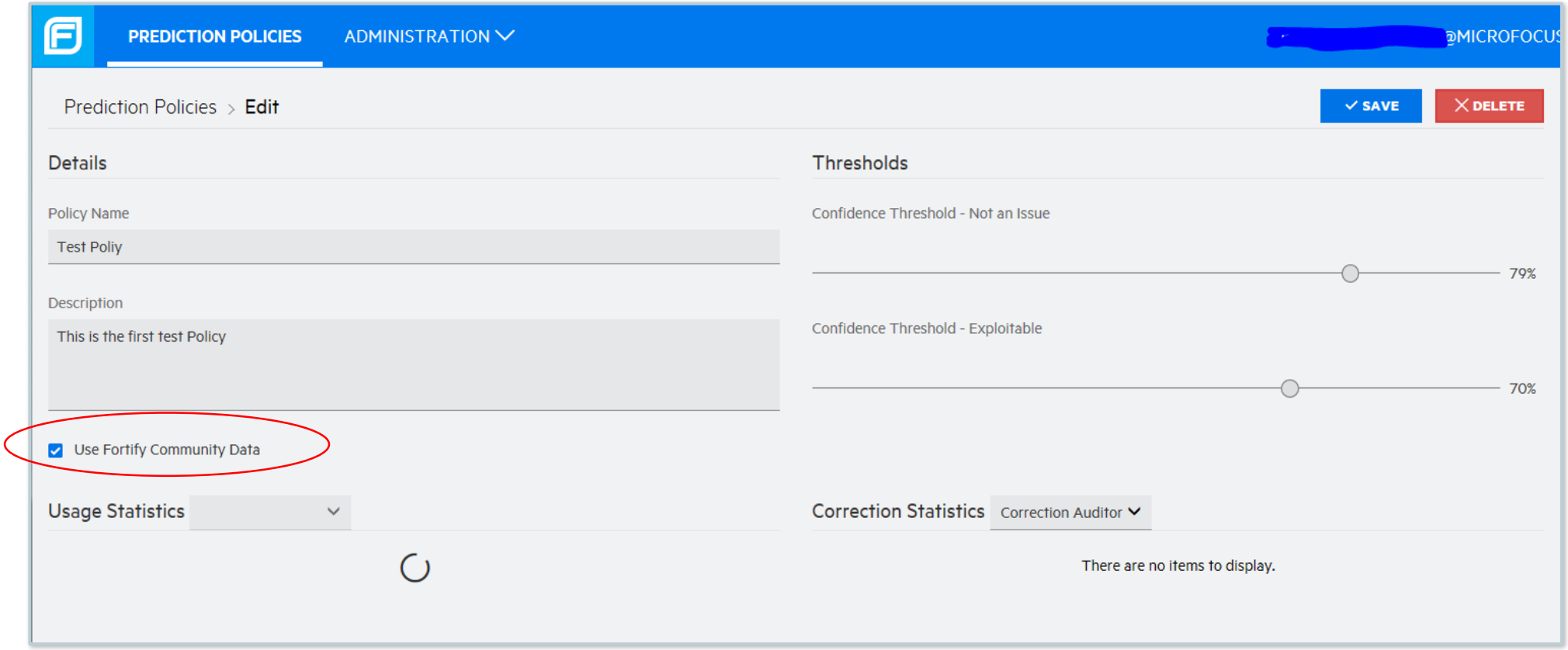
전역 머신러닝 정책 자동 생성



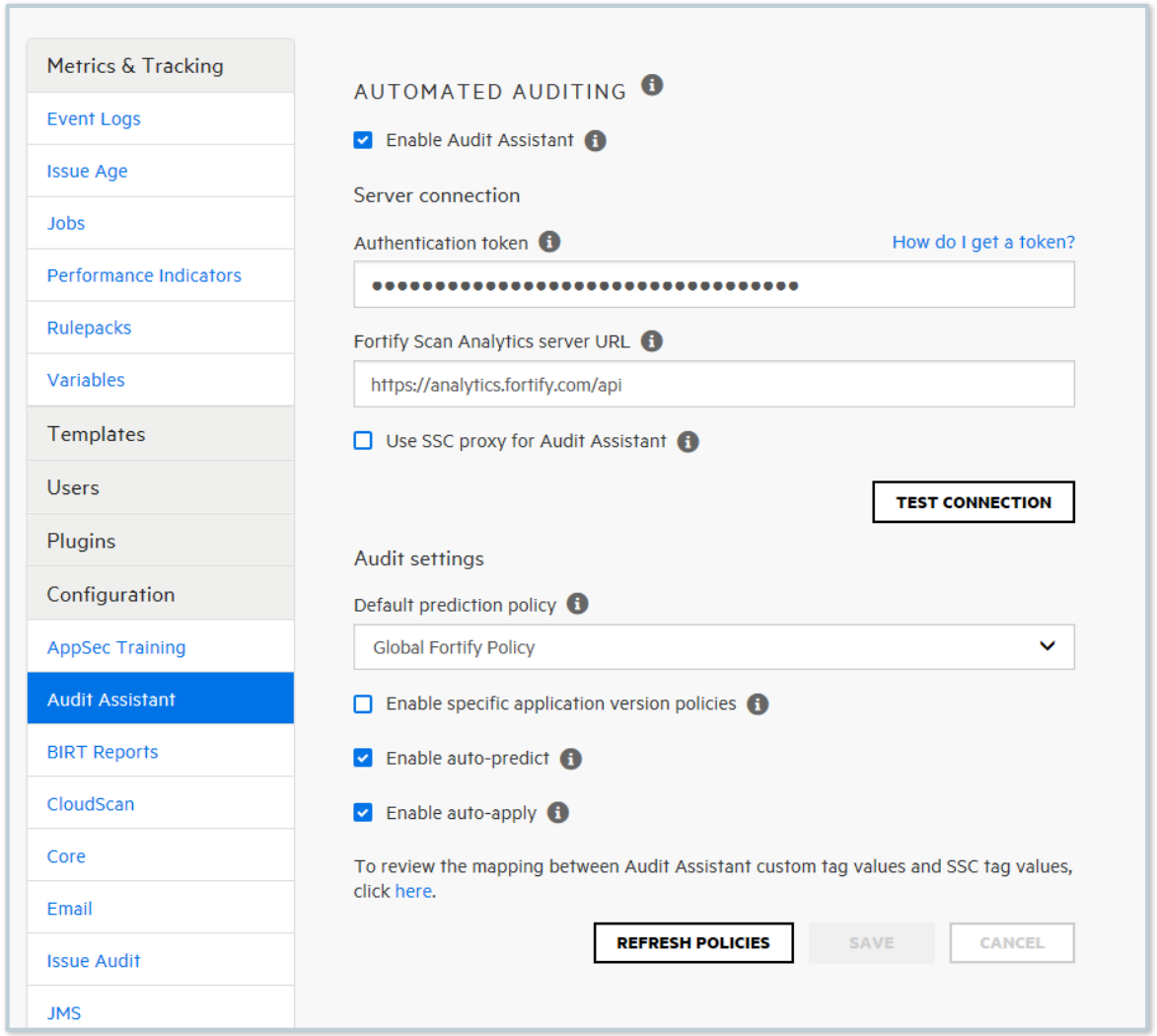
Custom 정책 지원

“ + ADD ” 버튼을 통한 Custom Policy 생성 가능

- “Use Fortify Community Data”를 선택하면 포티파이 구축한 머신러닝 데이터 활용 가능

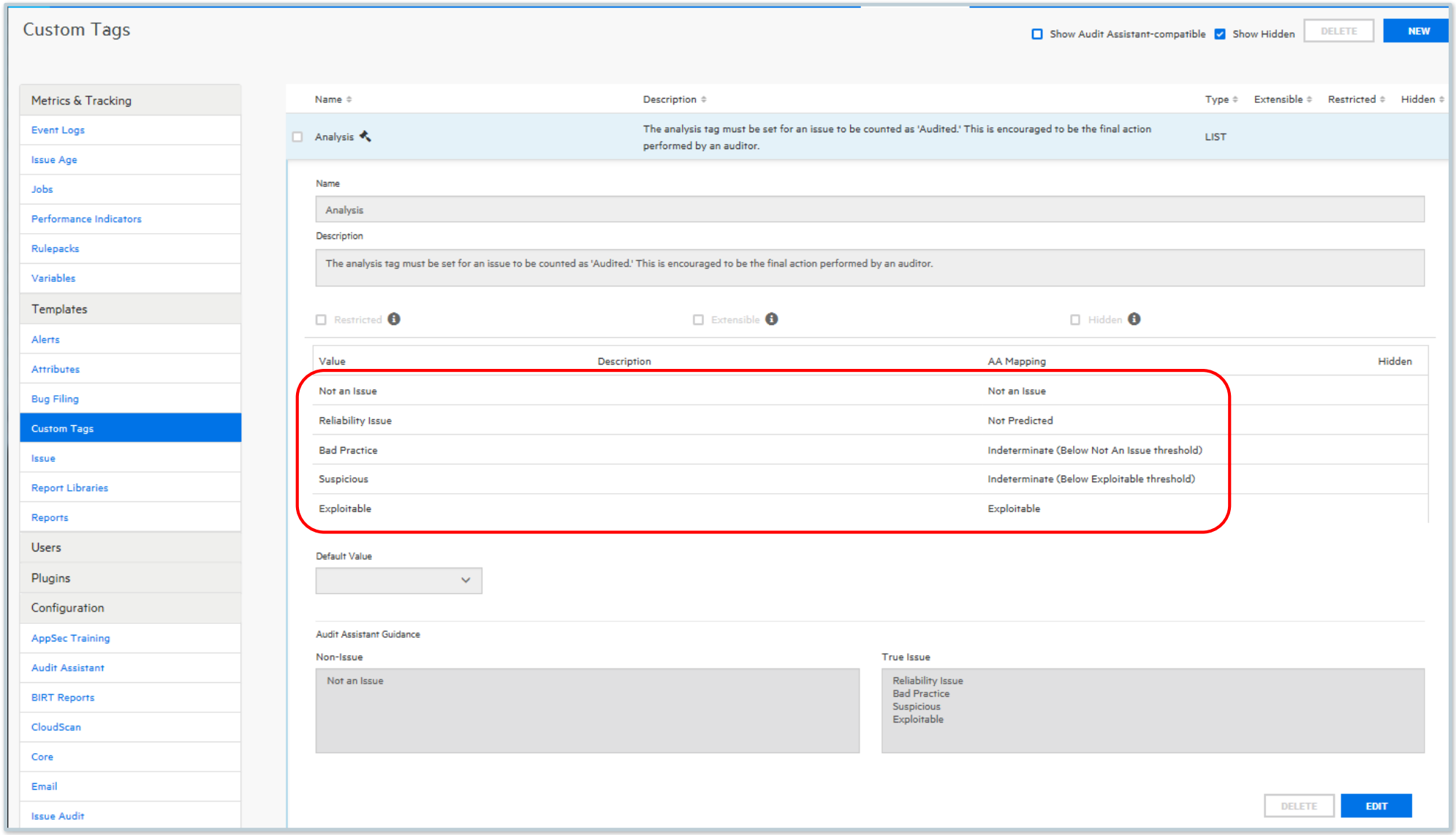


SSC를 통한 통합 설정

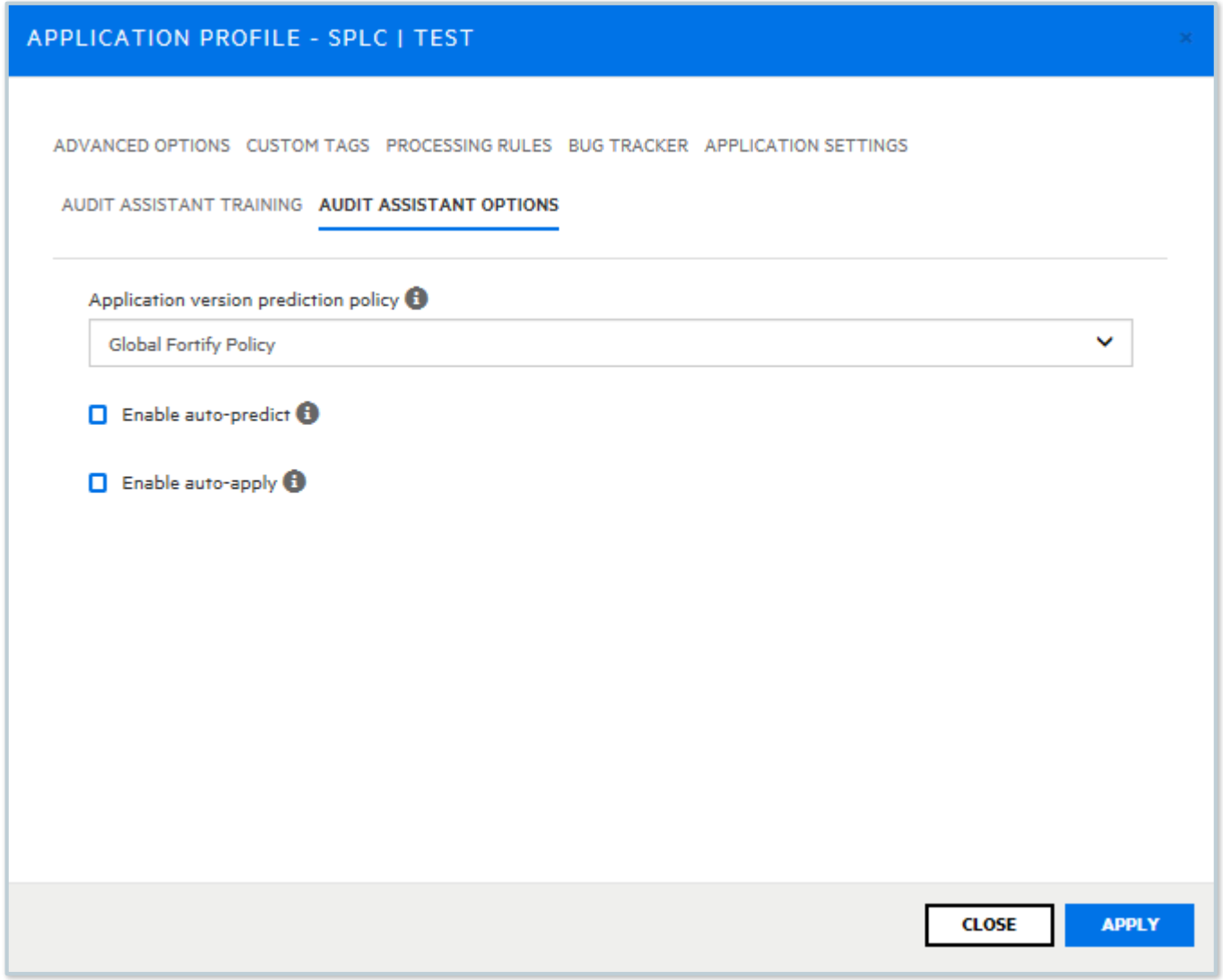


- SSC를 통한 머신러닝 설정
- 기존의 포티파이 지원 토큰 및 URL 사용 가능
  - Global Fortify Policy
- 원할 경우 어플리케이션 단에서 API를 통한 정책 Override 가능

Custom 태그

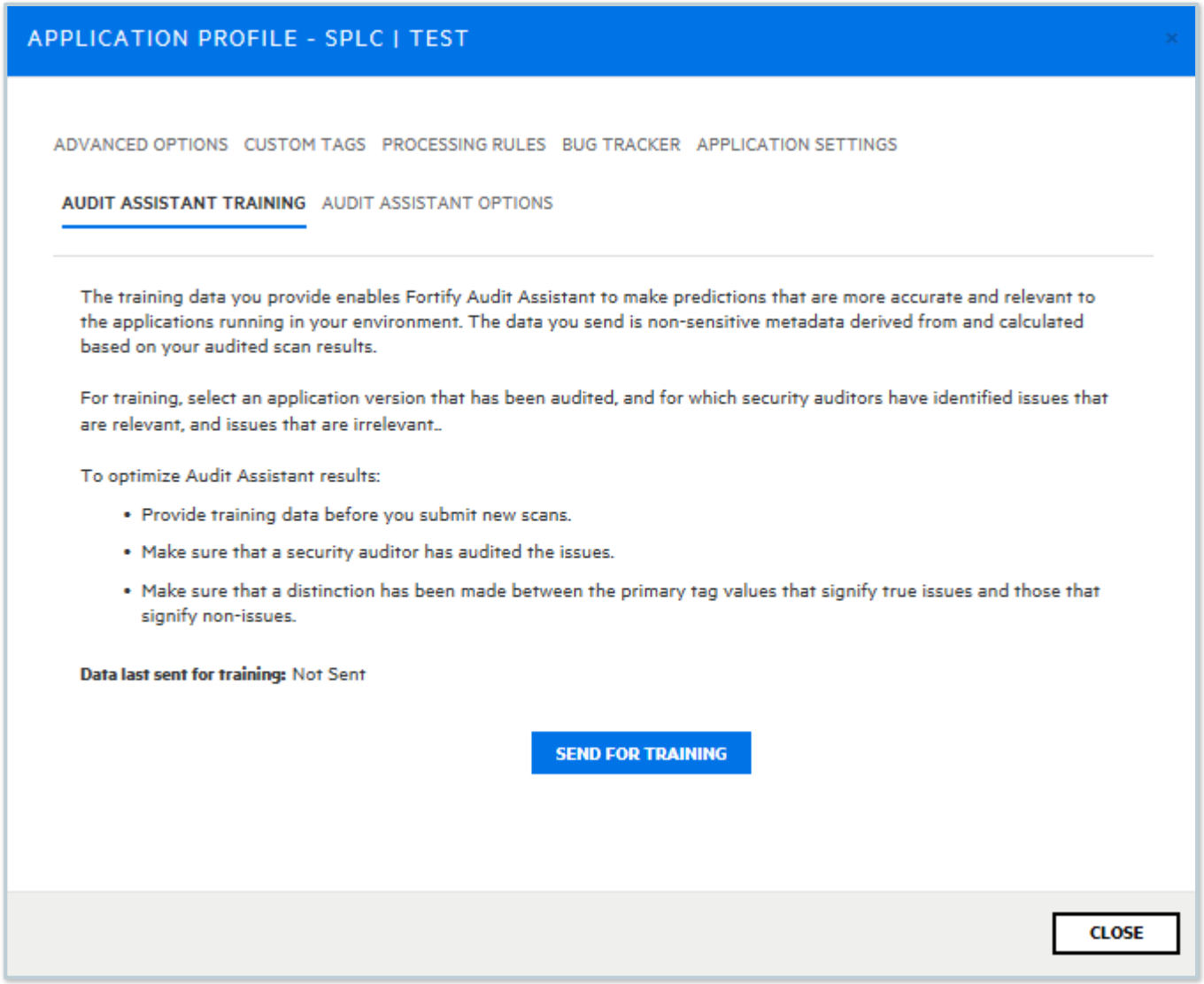

Custom Tags
Show Audit Assistant-compatible
Show Hidden
DELETE
NEW
Metrics & Tracking
Event Logs
Issue Age
Jobs
Performance Indicators
Rulepacks
Variables
Templates
Alerts
Attributes
Bug Filing
Custom Tags
Issue
Report Libraries
Reports
Users
Plugins
Configuration
AppSec Training
Audit Assistant
BIRT Reports
CloudScan
Core
Email
Issue Audit
Name
Description
Type
Extensible
Restricted
Hidden
Analysis
The analysis tag must be set for an issue to be counted as 'Audited.' This is encouraged to be the final action performed by an auditor.
LIST
Name
Analysis
Description
The analysis tag must be set for an issue to be counted as 'Audited.' This is encouraged to be the final action performed by an auditor.
Restricted
Extensible
Hidden
Value
Description
AA Mapping
Hidden
Not an Issue
Not an Issue
Reliability Issue
Not Predicted
Bad Practice
Indeterminate (Below Not An Issue threshold)
Suspicious
Indeterminate (Below Exploitable threshold)
Exploitable
Exploitable
Default Value
Audit Assistant Guidance
Non-Issue
Not an Issue
True Issue
Reliability Issue
Bad Practice
Suspicious
Exploitable
DELETE
EDIT

어플리케이션 별 머신러닝 정책 설정



- 어플리케이션 프로파일에서 정책, 자동 예측 및 적용 지정 가능

학습 데이터 전송



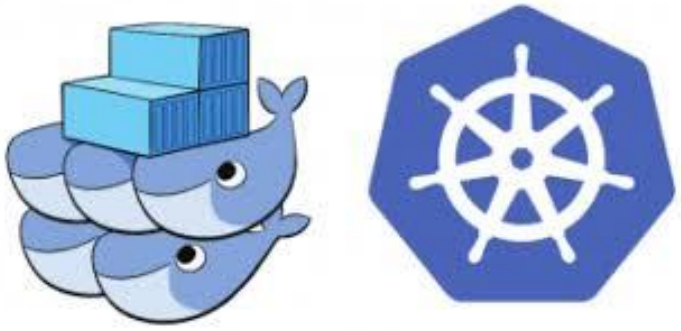
학습 데이터를
수동으로도 필요 시
전송 가능

포티파이 머신러닝 On-Premise!

고객의 요구가 높아 클라우드 형식으로 지원되던 **모든 머신러닝의 기능을 고객의 On-Premise에서도 지원**합니다!

- **Kubernetes를 통한 Docker 형태로 배포**됨으로써 **현재의 배포 트렌드를 따르고 관리 편의성**이 높습니다.
- 이 과정에서 새로운 PostgreSQL 백엔드를 개발하였습니다.

Audit Assistant™



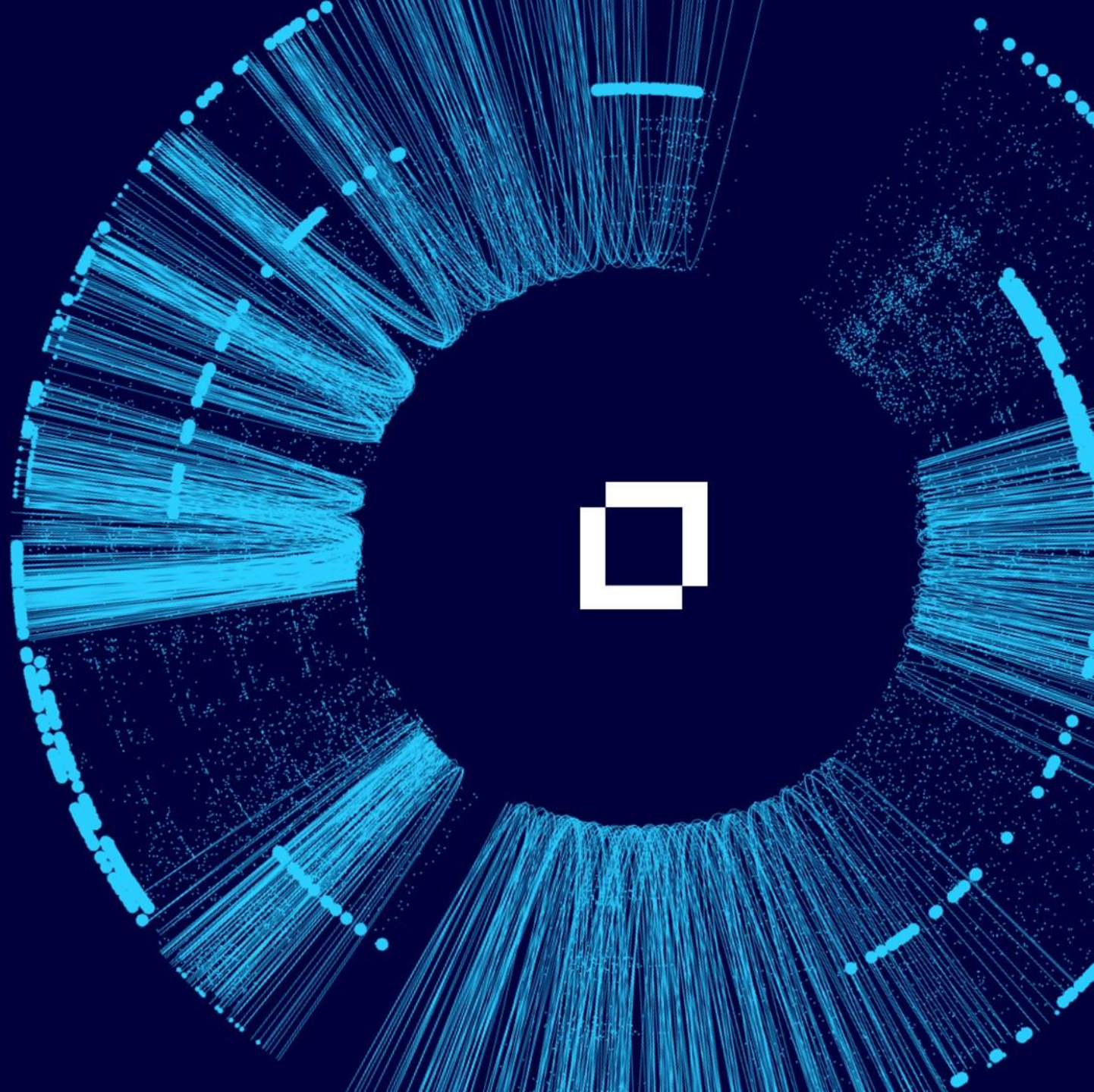
포티파이를 통한 Low-friction AppSec

- 사실 **False Positive를 줄이려는 노력은 개발자의 신뢰를 잃지 않으려는 포티파이의 많은 노력 중 하나**입니다.
- **포티파이 아키텍쳐**
  - **스캐닝 인프라를 공유하여 자원을 아끼고**
  - **빌드 서버의 Footprint를 줄이도록 재설계**되었습니다.
- 개발팀의 **AppSec 테스팅 보조**
  - 필요한 모든 곳에 **자동화를 적용하여 AppSec 테스팅의 자동화와 기민함**을 돕습니다.
- 보안 이슈 분석/감사/취약점 조치
  - **어플리케이션의 라이프 사이클에 걸친 보안 기준점 (Baseline)** 역할을 합니다.
  - **필터/커스텀 Rule/이슈 순서화 (Prioritization)** 그리고
  - **머신러닝 기술을 통해 False Positive를 줄여 실질적 취약점 조치**를 도와줍니다.

정리하며…

- (예외가 있겠지만) **AppSec 활동은 현재의 기술, 식견, App에 대한 가시성을 동원하여 향후에 일어날 수 있는 보안 침해 사례를 미리 막는 활동**입니다.
- 다량의 코드가 양산되고 잦은 배포가 일어나는 현업 현실에서 **코드 분석기가 신뢰를 잃으면 실제로 일어날 가능성이 높은 취약점마저 무시하게 만듭니다**.
- 포티파이는 **머신러닝 기술을 바탕으로 오탐을 줄이고, 미래에 발현할 가능성이 높은 실질적인 보안 취약점을 더 많이 탐지할 수 있도록 많은 노력을 기울이고 있습니다**.

Thank you.



#MicroFocusSecurityForum2019