

Differ:

LLM을 이용한 코드 변화 보안 검수

w/ Amazon Bedrock

홍성진

홍성진 aka. nisam

- 샌드버드 Staff Security Engineer
- AWS 한국 사용자모임 보안 소모임 운영진
- 前 네이버 Security Engineer
- 비오비 4기



#AppSec #DevSecOps #ThreatModeling #BugBounty #CloudSec
#SecureCoding #☕ #🎾 #🏆



발표자료에 도움을 주신 분 조흥기, 박우현

TABLE OF CONTENTS

01 | 문제 정의

02 | 시스템
아키텍처

03 | 성능 평가

04 | 프론트
엔지니어링

05 | 주요 인사이트

06 | Q&A

01

문제 정의

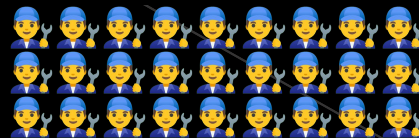
보안 엔지니어의 업무 중:

모든 제품 수정 및 새로운 기능 구현에 대해 보안 평가를
수행합니다.

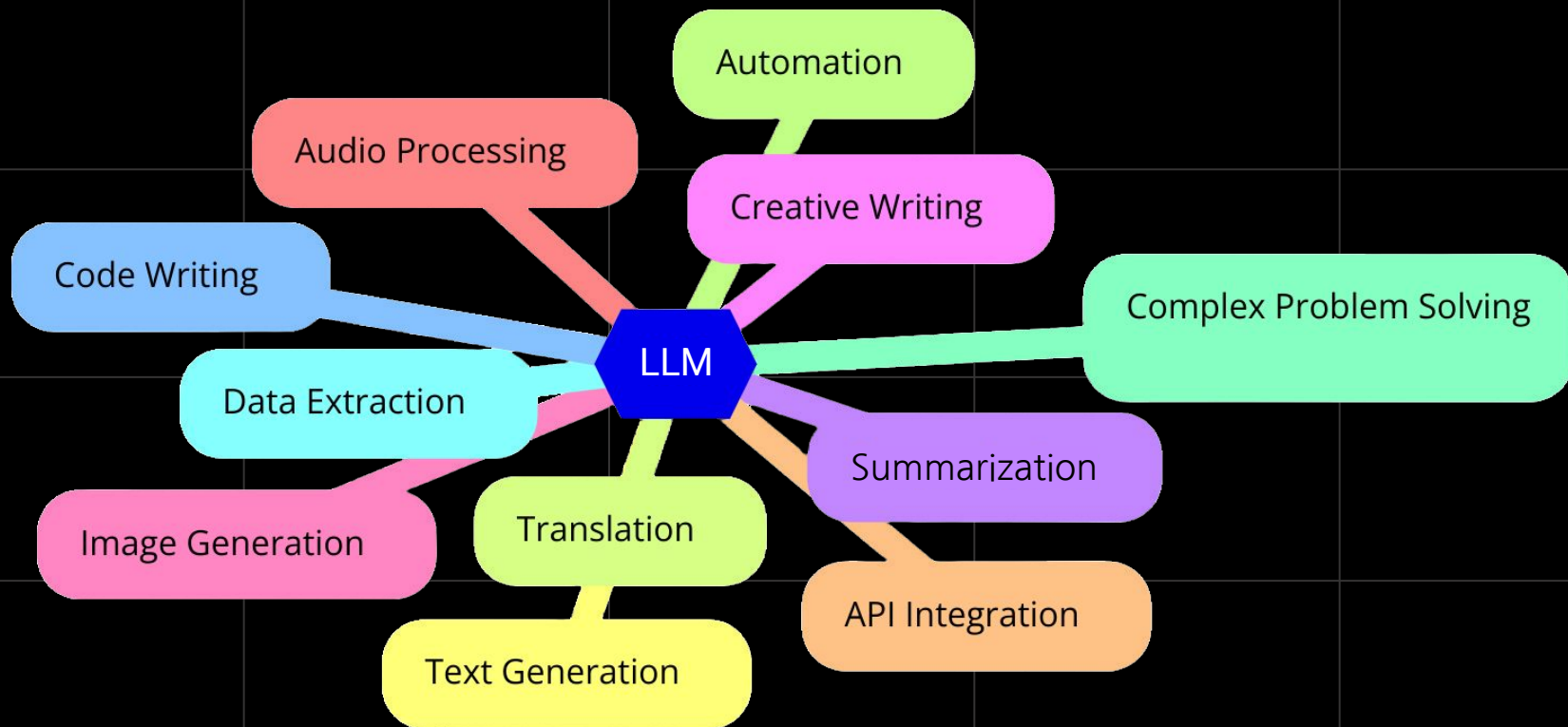
+

컴플라이언스 요구사항 이기도 함

VS



하지만 LLM이 출동하면 어떨까?



LLM들이 취약점을 찾을 수 있을까?

A Comprehensive Study of the Capabilities of Large Language Models for Vulnerability Detection

Benjamin Steenhoek
Iowa State University
Ames, Iowa, USA
benjis@iastate.edu

Md Mahbubur Rahman
Iowa State University
Ames, Iowa, USA
mdrahman@iastate.edu

Monoshi Kumar Roy
Iowa State University
Ames, Iowa, USA
monoshi@iastate.edu

Mirza Sanjida Alam
Iowa State University
Ames, Iowa, USA
sanjida@iastate.edu

Earl T. Barr
University College London
London, UK
e.barr@ucl.ac.uk

Wei Le
Iowa State University
Ames, Iowa, USA
weile@iastate.edu

Abstract—Large Language Models (LLMs) have demonstrated great potential for code generation and other software engineering tasks. Vulnerability detection is of crucial importance to maintaining the security, integrity, and trustworthiness of software systems. Precise vulnerability detection requires reasoning about the code, making it a good case study for exploring the limits of LLMs' reasoning capabilities. Although recent work has applied LLMs to vulnerability detection using generic prompting techniques, their full capabilities for this task and the types of errors they make when explaining identified vulnerabilities remain unclear.

reasoning. Pattern-matching on code structures is insufficient to produce precise analyses [67, 53], especially for real-world code. For example, to precisely detect a buffer overflow, we cannot only scan for `strcpy` or `malloc` statements. We need to identify the statements that update the strings and buffers, reason about the lengths of the strings after the changes at these statements, and also understand the bounds-check code to judge whether the protection is sufficient.

LLMs have shown limited ability for complex reasoning [27]

LLMs Cannot Reliably Identify and Reason About Security Vulnerabilities (Yet?): A Comprehensive Evaluation, Framework, and Benchmarks

Saad Ullah
Boston University
saadu@bu.edu

Mingji Han
Boston University
mjhan@bu.edu

Saurabh Pujar
IBM Research
saurabh.pujar@ibm.com

Hammond Pearce
UNSW Sydney
hammond.pearce@unsw.edu.au

Ayşe Coskun
Boston University
acoskun@bu.edu

Gianluca Stringhini
Boston University
gian@bu.edu

Abstract—Large Language Models (LLMs) have been suggested for use in automated vulnerability repair, but benchmarks showing they can consistently identify security-related bugs are lacking. We thus develop SecLLMHolmes, a fully automated evaluation framework that performs the most detailed investigation to date on whether LLMs can reliably identify and reason about security-related bugs. We construct a set of 228 code scenarios and analyze eight of the most capable LLMs across eight different investigative dimensions using our framework. Our evaluation shows LLMs provide non-deterministic responses, incorrect and unfaithful reasoning, and perform poorly in real-world scenarios. Most importantly, our findings reveal significant non-robustness in even the most advanced models like ‘PaLM2’ and ‘GPT-4’: by merely changing function or variable names, or by the addition of library

especially as LLMs are not infallible in security-related tasks, for example introducing vulnerabilities into source code [8], [9] and software testing [10]. Unfortunately, there is no standardized and automated approach to evaluate the performance of LLMs at identifying vulnerable code. We fill this gap by introducing SecLLMHolmes, a generalized, fully automated, and scalable framework to systematically evaluate the performance (i.e., accuracy and reasoning capabilities) of LLMs for vulnerability detection. Our framework tests the capabilities of a given LLM as a security assistant across eight distinct dimensions: (1) deterministic response, (2) performance over range of parameters, (3) diversity of prompts, (4) faithful reasoning, (5) evaluation over variety of vulnerabilities, (6) assessment of various code difficulty levels, (7) robustness to code augmentations, and (8) use in

전용 딥 러닝 모델은 잘하는거 같은데...

- 코스트
- 러닝 커브
- 유지보수
- ...

VulDeePecker: A Deep Learning-Based System for Vulnerability Detection

Zhen Li^{*†}, Deqing Zou^{*‡§}, Shouhuai Xu[§], Xinyu Ou^{*}, Hai Jin^{*},
Sujuan Wang^{*}, Zhijun Deng^{*} and Yuyi Zhong^{*}

^{*}Services Computing Technology and System Lab, Big Data Technology and System Lab,
Cluster and Grid Computing Lab, School of Computer Science and Technology,
Huazhong University of Science and Technology
deqingzou@hust.edu.cn

[†]School of Cyber Security and Computer, Hebei University

[‡]Shenzhen Huazhong University of Science and Technology Research Institute

[§]Department of Computer Science, University of Texas at San Antonio

Abstract—The automatic detection of software vulnerabilities is an important research problem. However, existing solutions to this problem rely on human experts to define features and often miss many vulnerabilities (i.e., incurring high false negative rate). In this paper, we initiate the study of using deep learning-based vulnerability detection to relieve human experts from the tedious and subjective task of manually defining features. Since deep learning is motivated to deal with problems that are very different from the problem of vulnerability detection, we need some guiding principles for applying deep learning to vulnerability detection. In particular, we need to find representations of software programs that are suitable for deep learning. For this purpose, we propose using *code gadgets* to represent programs and then transform them into vectors, where a code gadget is a number of (not necessarily consecutive) lines of code that are semantically related to each other. This leads to the design and implementation of a deep learning-based vulnerability detection system, called **Vulnerability Deep Pecker (VulDeePecker)**. In order to evaluate VulDeePecker, we present the first vulnerability dataset for deep learning approaches. Experimental results show that VulDeePecker can achieve much fewer false negatives (with reasonable false positives) than other approaches. We further apply VulDeePecker to 3 software products (namely Xen, Seamonkey, and Libav) and detect 4 vulnerabilities, which are not reported in the National Vulnerability Database but were “silently” patched by the vendors when releasing later versions of these products; in contrast, these vulnerabilities are almost entirely missed by the other vulnerability detection systems we experimented with.

that the number of vulnerabilities registered in the Common Vulnerabilities and Exposures (CVE) was approximately 4,600 in 2010, and grew to approximately 6,500 in 2016 [4]. An alternate approach is to automatically detect vulnerabilities in software programs, or simply *programs* for short. There have been many static vulnerability detection systems and studies for this purpose, ranging from open source tools [6], [11], [52], to commercial tools [2], [3], [7], to academic research projects [19], [28], [32], [37], [38], [49], [59], [60]. However, existing solutions for detecting vulnerabilities have two major drawbacks: imposing *intense manual labor* and incurring *high false negative rates*, which are elaborated below.

On one hand, existing solutions for vulnerability detection rely on human experts to define features. Even for experts, this is a tedious, subjective, and sometimes error-prone task because of the complexity of the problem. In other words, the identification of features is largely an art, meaning that the quality of the resulting features, and therefore the effectiveness of resulting detection system, varies with the individuals who define them. In principle, this problem can be alleviated by asking multiple experts to define their own features, and then select the set of features that lead to better effectiveness or use a combination of these features. However, this imposes even more tedious work. As a matter of fact, it is always desirable to reduce, or even eliminate whenever possible, the reliance on the intense labor of human experts. This can be justified by the trend of cyber defense automation, which is catalyzed

공개 LLM 모델을 활용한다면 생기는 장점

- 낮은 러닝커브
- 손쉬운 유지보수
- 모델 출시에 따른 지속적인 성능향상
- 자연어 및 프로그래밍 언어 이해력
- 자체 모델 제작 및 운용에 비해 상대적 비용 절감

문제

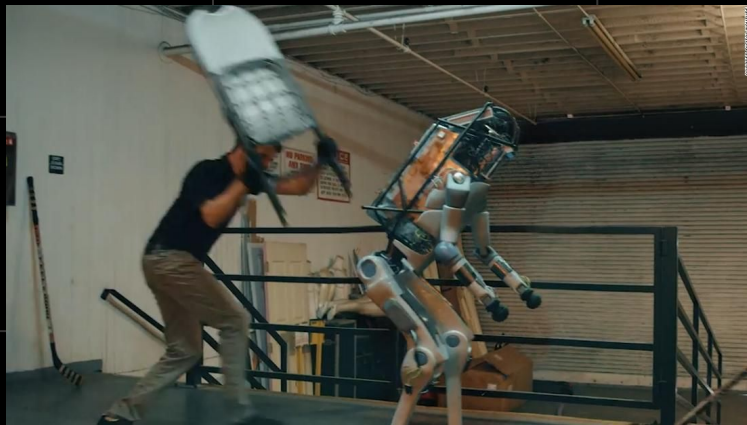
모든 제품 수정 및 새로운 기능 구현에 대해 보안 평가를 수행 할 시간과 인력이 부족하다.



해결책?
(인력 추가, 자동화 툴[SAST/DAST/...], 등)

문제 해결

모든 제품 수정 및 새로운 기능 구현에 대해서 사람이 평가할 수 없다면,
LLM을 이용해 사람이 검수 할 내용을 찾아내자



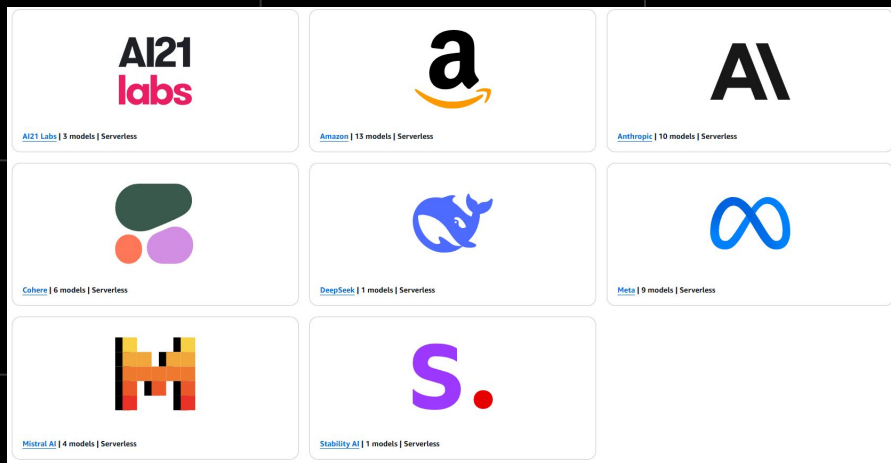
우리는 주로 어떤 것을 검수하는가

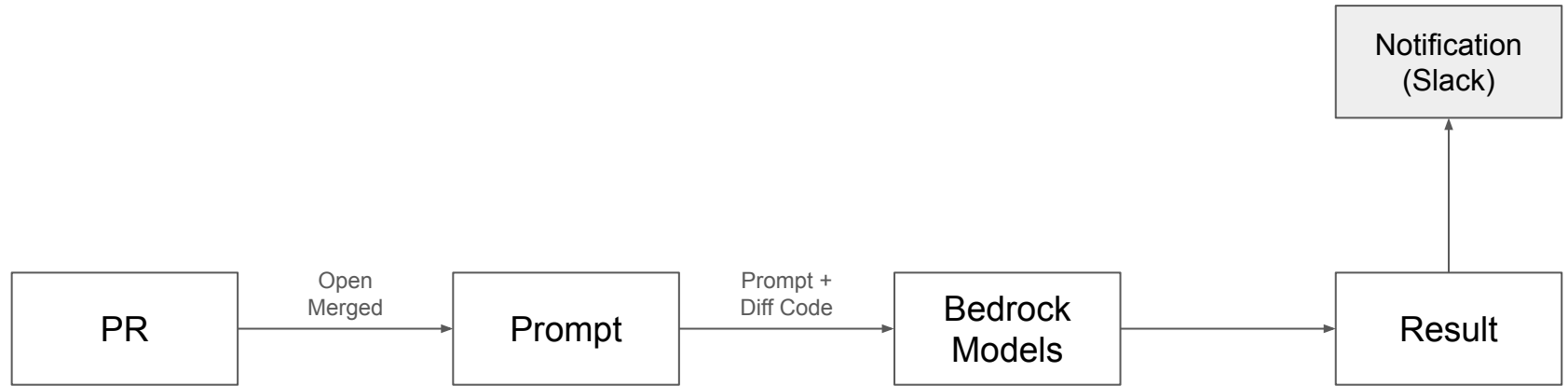
- 새로운 엔드포인트, 파라미터
- 취약점
- 주요 정보 혹은 시크릿 데이터
- 개인 정보 수집
- 새로운 패키지 사용
- 기타

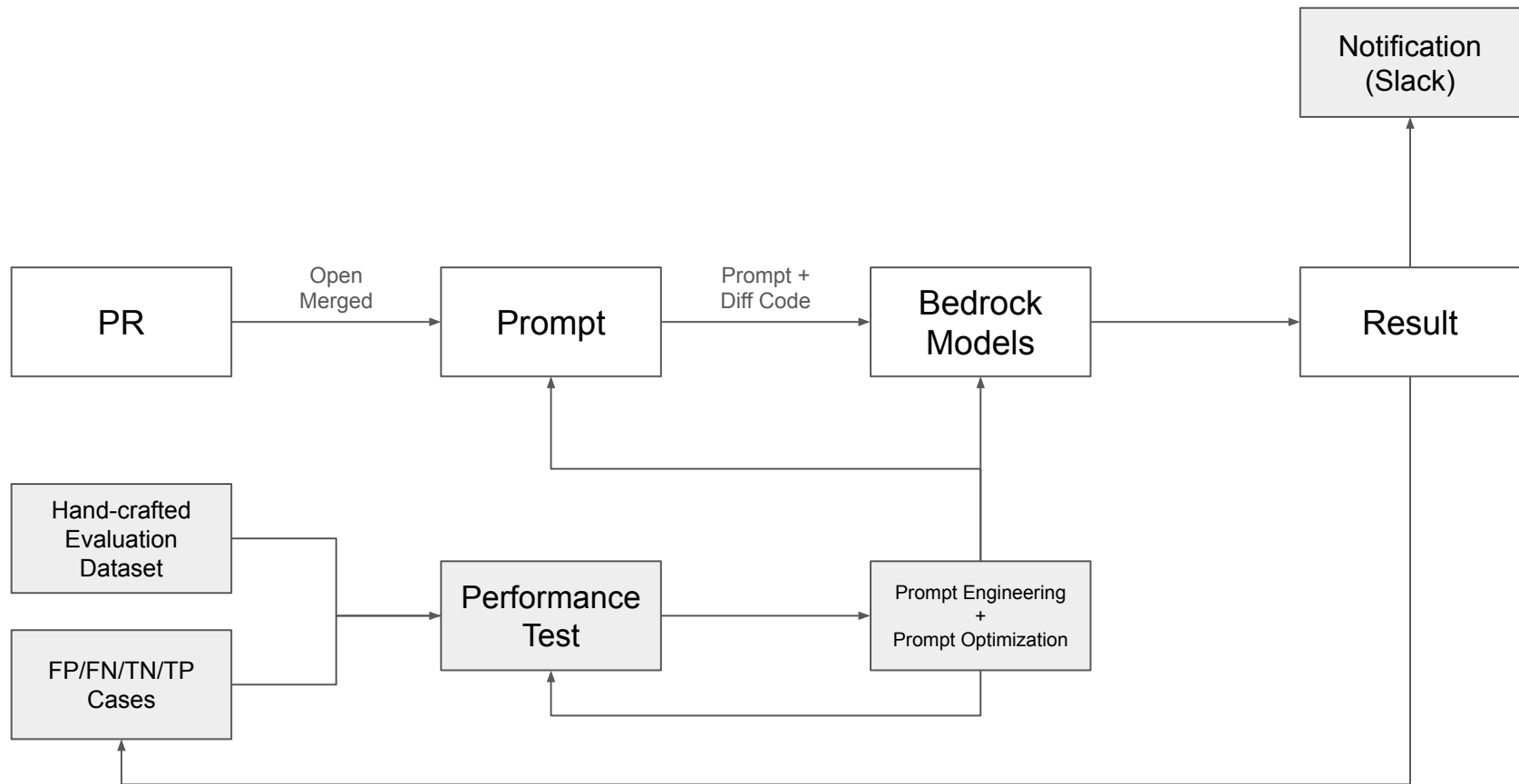
02 | 시스템 아키텍처

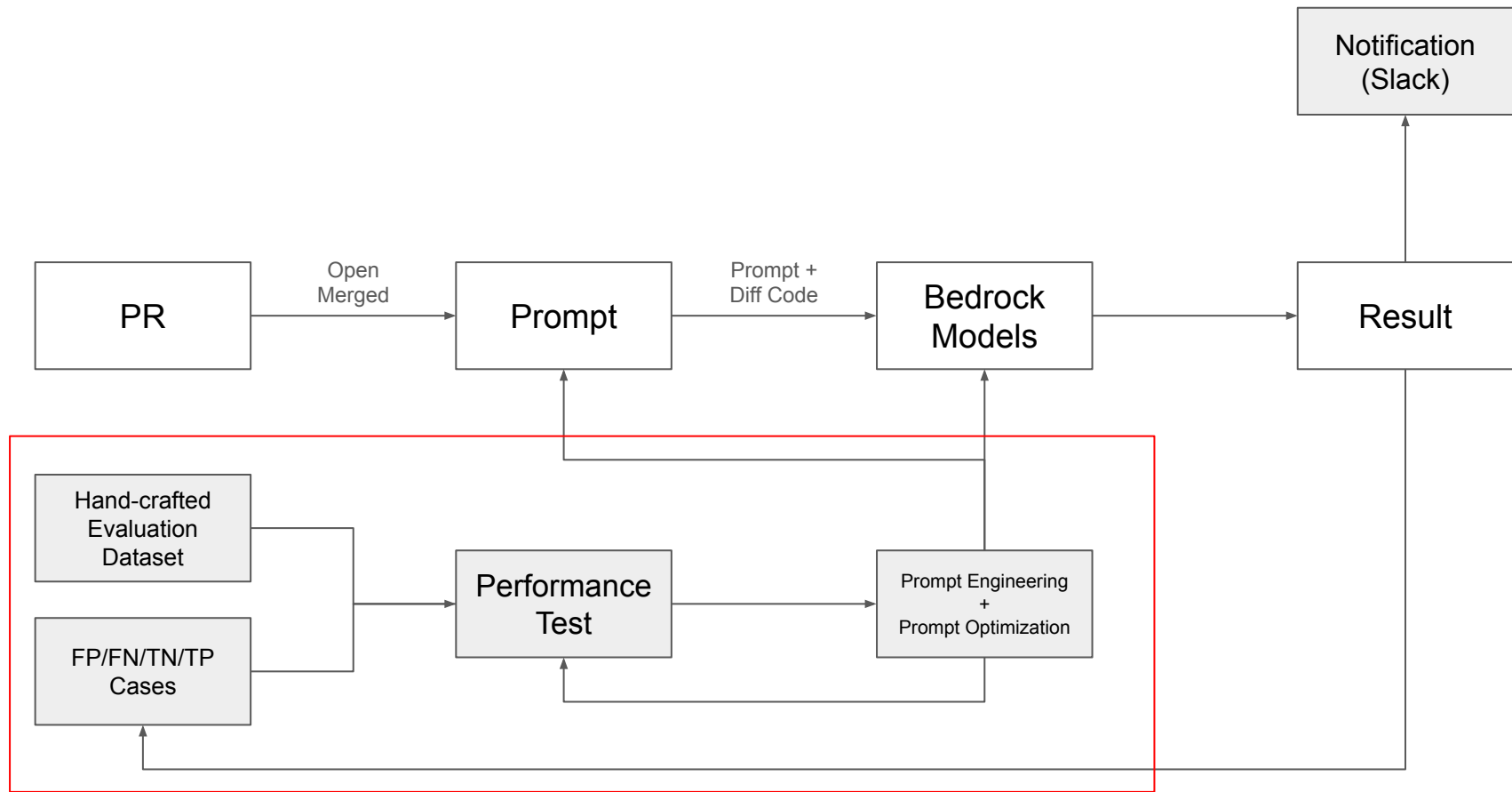
Amazon Bedrock

Amazon Bedrock은 AI 기업과 Amazon의 고성능 모델을 통합 API를 통해 사용할 수 있게 해주는 완전 관리형 서비스









03

| 성능 평가

성능 평가 공식

- Accuracy = $(TP+TN) / (TP+TN+FP+FN)$
- Precision = $TP / (TP+FP)$
- Recall = $TP / (TP+FN)$
- F1 Score = $TP / TP + 1/2(FP+FN)$

	0	1
0	True negative (TN)	False positive (FP)
1	False negative (FN)	True positive (TP)

성능 평가 공식

- Accuracy = $(TP+TN) / (TP+TN+FP+FN)$
- Precision = $TP / (TP+FP)$
- Recall = $TP / (TP+FN)$
- F1 Score = $TP / TP + 1/2(FP+FN)$



성능 평가용 수작업 케이스

총 40건의 테스트케이스 생성

E-new_component-2 #35

Conversation 0 Commits 1 Checks 0 Files changed 3

Changes from all commits File filter Conversations 0

0 / 3 files viewed Review in codespace Review changes

Filter changed files

- django.nv-3.0-safe
 - requirements.txt
- taskManager
 - urls.py
- django.nv-3.0/taskManager
 - settings.py

3 django.nv-3.0-safe/requirements.txt

```
@@ -1,4 +1,5 @@
1 Django==3.2.16
2 BeautifulSoup==4.11.1
3 advocate==1.0.0
4 - pyyaml==6.0.2
5 + drf-yasg==1.21.7
```

19 django.nv-3.0-safe/taskManager/urls.py

```
@@ -13,6 +13,9 @@
13 #
14
15 from django.urls import re_path
16
17 from taskManager.views import (
18     accounts,
19     otag
20 )
21
22 + schema_view = get_schema_view(
23 +     openapi.Info(
24 +         title="Snippets API",
```

성능 평가용 수작업 케이스

type	counts
Cryptographic_failures	2
New_endpoint	4
Nothing	5
Path_traversal	2
Hardcoded_secret	2
PII	3
Auths	2
SSRF	2
Misconfiguration	4
IDOR	3
SQLi	3
CSRF	2
XSS	3
New_component	3
Sum	40

성능 평가 테스트 플랫폼 제작

ID	Model	Evaluation Time	Precision	W Precision	Recall	W Recall	F1 Score	W F1 Score	Accuracy
26	anthropic.claude-3-5-haiku-20241022-v1:0	238	0.961904761904762 (3)	0.9625 (3)	0.5894557823129251 (18)	0.5924107142857143 (18)	0.7179296893582608 (18)	0.7209623709623709 (18)	0.8693877551020408 (15)
28	anthropic.claude-3-5-sonnet-20241022-v2:0	372	0.9857142857142858 (1)	0.99375 (1)	0.578901257472686 (20)	0.5727137445887446 (20)	0.7159602302459446 (19)	0.7158882783882784 (19)	0.8693877551020408 (15)
29	anthropic.claude-3-haiku-20240307-v1:0	177	0.8095238095238094 (21)	0.875 (20)	0.40986394557823125 (22)	0.4367559523809524 (22)	0.5126984126984128 (22)	0.545138888888889 (22)	0.752100840361344 (22)
30	anthropic.claude-3-sonnet-20240229-v1:0	249	0.9476190476190477 (6)	0.95625 (6)	0.761904761904762 (5)	0.765625 (3)	0.813749742321171 (3)	0.8262536075036075 (3)	0.8991596638655462 (8)
31	anthropic.claude-3-sonnet-20240229-v1:0	261	0.9357142857142857 (10)	0.940625 (11)	0.660496249781964 (16)	0.6533596611721612 (16)	0.7425580267685531 (16)	0.7453144808047965 (14)	0.8693877551020408 (15)
32	anthropic.claude-3-sonnet-20240229-v1:0	261	0.9476190476190477 (6)	0.95625 (6)	0.7317503924646782 (11)	0.7260473901098901 (11)	0.7946322683164789 (11)	0.8011619228724491 (8)	0.889795918367347 (11)
33	anthropic.claude-3-sonnet-20240229-v1:0	282	0.9357142857142857 (10)	0.940625 (11)	0.7326500775748895 (10)	0.7411171157059315 (9)	0.7991058177815392 (8)	0.8128814097617154 (4)	0.9061224489795918 (4)
36	anthropic.claude-3-sonnet-20240229-v1:0	238	0.911904761904762 (15)	0.909375 (17)	0.7489177489177489 (7)	0.7418831168831169 (8)	0.7969866734572617 (9)	0.7974338978015448 (9)	0.889795918367347 (11)
42	anthropic.claude-3-sonnet-20240229-v1:0	263	0.9095238095238096 (17)	0.91875 (16)	0.761904761904762 (5)	0.75 (5)	0.802170868347339 (6)	0.8040747549019608 (6)	0.9020408163265307 (5)
46	anthropic.claude-3-sonnet-20240229-v1:0	269	0.911904761904762 (15)	0.909375 (17)	0.7460317460317459 (8)	0.7430555555555556 (7)	0.8021189336978811 (7)	0.8016846092503897 (7)	0.8938775510204081 (10)
48	anthropic.claude-3-sonnet-20240229-v1:0	266	0.9238095238095239 (13)	0.925 (14)	0.7448979591836735 (9)	0.7276785714285714 (10)	0.8023504273504275 (5)	0.7961404914529914 (10)	0.9020408163265307 (5)
52	anthropic.claude-3-sonnet-20240229-v1:0	319	0.9333333333333335 (12)	0.95 (10)	0.6373445930096177 (17)	0.6073481116584565 (17)	0.7302084811410492 (17)	0.7221241216937055 (17)	0.8693877551020408 (15)
53	anthropic.claude-3-sonnet-20240229-v1:0	264	0.9476190476190477 (6)	0.95625 (6)	0.765 (4)	0.7696875 (2)	0.8166179166179166 (2)	0.8300180862680863 (2)	0.9020408163265307 (5)
54	anthropic.claude-3-sonnet-20240229-v1:0	279	0.8880952380952382 (19)	0.878125 (19)	0.7052154195011339 (13)	0.6845238095238095 (13)	0.7476190476190476 (15)	0.732638888888889 (16)	0.8612244897959184 (20)
56	anthropic.claude-3-5-haiku-20241022-v1:0	237	0.961904761904762 (3)	0.9625 (3)	0.5776540919398061 (21)	0.5773268398268399 (19)	0.705779934351363 (20)	0.7084790209790209 (20)	0.8693877551020408 (15)
55	anthropic.claude-3-5-sonnet-20241022-v2:0	357	0.9857142857142858 (1)	0.99375 (1)	0.6841269841269841 (14)	0.6604166666666667 (15)	0.786646686466867 (12)	0.7741404428904428 (13)	0.8831168831168831 (14)
59	cohere.command-r-v1:0	833	0.8595238095238094 (20)	0.865625 (21)	0.5882086167800453 (19)	0.5095238095238095 (21)	0.6410636982065554 (21)	0.5913961038961039 (21)	0.8163265306122449 (21)
58	cohere.command-r-plus-v1:0	935	0.7761904761904762 (22)	0.75625 (22)	0.7687074829931972 (2)	0.7464285714285714 (6)	0.7550148264433979 (13)	0.7409569597069597 (15)	0.889795918367347 (11)
60	meta.llama3-1-405b-instruct-v1:0	760	0.961904761904762 (3)	0.9625 (3)	0.8214285714285714 (1)	0.7864583333333334 (1)	0.8720730397422127 (1)	0.8511748120300752 (1)	0.9495798319327731 (1)
63	us.meta.llama3-2-90b-instruct-v1:0	3445	0.906060606060606 (18)	0.9267045454545454 (13)	0.6729024943310657 (15)	0.6956845238095238 (12)	0.7500373190028363 (14)	0.7760906478578892 (12)	0.8949579831932774 (9)
64	meta.llama3-1-405b-instruct-v1:0	748	0.9476190476190477 (6)	0.95625 (6)	0.7079831927737109 (12)	0.6827001633986928 (14)	0.7965074679878931 (10)	0.7830100366965158 (11)	0.926530612244898 (2)
68	anthropic.claude-3-sonnet-20240229-v1:0	236	0.9238095238095239 (13)	0.925 (14)	0.7665732959850607 (3)	0.7561274509803921 (4)	0.8134183823838999 (4)	0.8106671824344237 (5)	0.9142857142857143 (3)

04

| 프롬프트
엔지니어링



Andrej Karpathy ✓

@karpathy

The hottest new programming language is English

9:14 AM · Jan 25, 2023 · **2.3M** Views

2,672 Retweets

408 Quotes

20.1K Likes

1,268 Bookmarks

System prompt:

You are a security engineer.

VS

You are a senior security engineer working for an IT
company.

System prompt:

You are a security engineer.

VS

You are a senior security engineer working for an IT
company.

사용한 프롬프트 엔지니어링 기술들

- Chain-of-thought
- Persona
- Order of words
- 영어 vs 한글
- Ensemble LLMs

Chain-of-thought

(a) Few-shot

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: The answer is 11.

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?

A:

(Output) The answer is 8. ✗

(c) Zero-shot

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?

A: The answer (arabic numerals) is

(Output) 8 ✗

(b) Few-shot-CoT

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. $5 + 6 = 11$. The answer is 11.

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?

A:

(Output) The juggler can juggle 16 balls. Half of the balls are golf balls. So there are $16 / 2 = 8$ golf balls. Half of the golf balls are blue. So there are $8 / 2 = 4$ blue golf balls. The answer is 4. ✓

(d) Zero-shot-CoT (Ours)

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?

A: **Let's think step by step.**

(Output) There are 16 balls in total. Half of the balls are golf balls. That means that there are 8 golf balls. Half of the golf balls are blue. That means that there are 4 blue golf balls. ✓

Chain-of-thought

ID	Model	Evaluation Time	Precision	W Precision	Recall
41	anthropic.claude-3-sonnet-20240229-v1:0	257	0.9238095238095239 (8)	0.925 (14)	0.8928571428571429 (1)
45	anthropic.claude-3-sonnet-20240229-v1:0	252	0.9119047619047619 (18)	0.909375 (18)	0.8499999999999999 (5)

Persona



Persona

39 You are a senior security engineer.

40 You are a senior security engineer working for an IT company.

41 You are a senior security engineer working for an IT company using GitHub

ID	Recall
<u>39</u>	0.8513888888888889 (3)
<u>40</u>	0.8598901098901098 (2)
<u>41</u>	0.8928571428571429 (1)

Order of words

프롬프트 결과물에 대한 출력 구조가 성능에 영향을 미침.

```
{"description": "", "result": true}
```

VS

```
{"result": true, "description": ""}
```

ID	Model	Evaluation Time	Recall
41	anthropic.claude-3-sonnet-20240229-v1:0	257	0.8928571428571429 (1)
43	anthropic.claude-3-sonnet-20240229-v1:0	249	0.780859010270775 (6)

영어 vs 한글

영어 프롬프트가 한글 프롬프트 보다 성능이 좋음.

79	meta.llama3-1-405b-instruct-v1:0	944	0.9380952380952382 (16)	0.93125 (21)	0.8571428571428571 (1)	0.8125 (1)	0.8840155945419104 (1)	0.8543494152046783 (1)	0.9428571428571428 (5)
96	meta.llama3-1-405b-instruct-v1:0	485	0.930952380952381 (22)	0.9343750000000001 (20)	0.7316326530612246 (20)	0.6894345238095237 (23)	0.8044191919191919 (12)	0.7821890782828284 (20)	0.9159663865546218 (15)

여러가지 모델을 섞어서 써보자

성능이 좋은 두가지 모델의 결과를 And 연산을 통해 오답을 상당히 높게 줄일 수 있었습니다.

ID	Model	Evaluation Time	Precision	W Precision	Recall	W Recall	F1 Score	W F1 Score	Accuracy
138	100 AND 137	0	0.9238095238095239 (16)	0.925 (19)	0.9432234432234432 (1)	0.9254807692307693 (1)	0.9317460317460318 (1)	0.9243055555555556 (1)	0.963265306122449 (1)

Llama + Claude

System: You are a security engineer working for an IT company. You have to answer the given question without making it up. Please answer questions, focusing mainly on the diff in the code.

Perform a comprehensive review of the provided diff(code changes), evaluating them with the questions.
Think step-by-step and then answer. Do not try to make up an answer.

Pay special attention to the following questions:

1. Identify new HTTP API endpoints or new user-input parameters to existing endpoints.
2. ...
3. ...

Below is the changed code, ignoring the headers in the diff, + is new code, and - is deleted code. Please focus on the newly added code:
<diff>{diff}</diff>

Below the associated files are the original code of the changed code. The code below is for reference only and should not be used to answer questions:
<associated_files>{associated_files}</associated_files>

IMPORTANT: RESPOND ****ONLY**** WITH THE JSON STRING IN THE FOLLOWING FORMAT, WITHOUT ANY MARKDOWN FORMATTING, CODE BLOCKS, OR ADDITIONAL TEXT.

Answer example:

```
<example>
{"new_endpoint": {"explanation": "detailed explanation", "result": boolean}, "vulnerability": {"explanation": "detailed explanation", "result": boolean}, "patch":
{"explanation": "detailed explanation", "result": boolean} ...}
</example>
```

모델 성능 평가

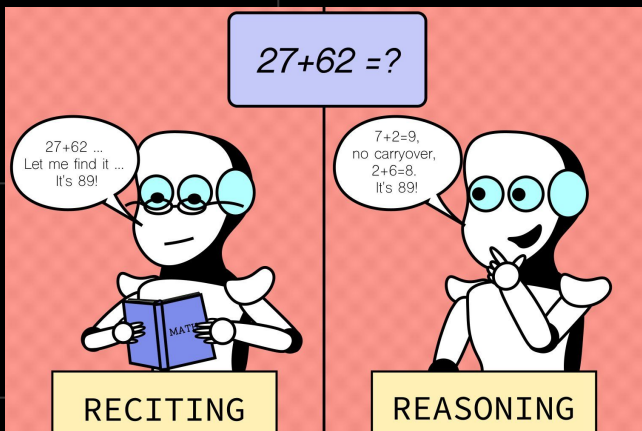
- anthropic.claude-3-haiku-20240307-v1:0
- anthropic.claude-3-sonnet-20240229-v1:0
- anthropic.claude-3-5-haiku-20241022-v1:0
- anthropic.claude-3-5-sonnet-20241022-v2:0
- anthropic.claude-3-7-sonnet-20250219-v1:0 (일반모델)
- cohere.command-r-plus-v1:0
- cohere.command-r-v1:0
- meta.llama3-1-405b-instruct-v1:0
- us.meta.llama3-2-90b-instruct-v1:0
- us.amazon.nova-pro-v1:0
- us.amazon.nova-lite-v1:0
- us.amazon.nova-micro-v1:0

모델 성능 평가

- anthropic.claude-3-haiku-20240307-v1:0
- anthropic.claude-3-sonnet-20240229-v1:0
- anthropic.claude-3-5-haiku-20241022-v1:0
- anthropic.claude-3-5-sonnet-20241022-v2:0
- anthropic.claude-3-7-sonnet-20250219-v1:0 (일반모델)
- cohere.command-r-plus-v1:0
- cohere.command-r-v1:0
- meta.llama3-1-405b-instruct-v1:0
- us.meta.llama3-2-90b-instruct-v1:0
- us.amazon.nova-pro-v1:0
- us.amazon.nova-lite-v1:0
- us.amazon.nova-micro-v1:0

Reasoning models

답변의 중간 단계가 있는 모델. 직접 결과까지 도달하는 과정에 추론을 함. 복잡한 문제에 적합하다고 함.



Good at

- + Deductive or inductive reasoning (e.g., riddles, math proofs)
- + Chain-of-thought reasoning (breaking down multi-step problems)
- + Complex decision-making tasks
- + Better generalization to novel problems

Bad at

- Fast and cheap responses (more inference time)
- Knowledge-based tasks (hallucination)
- Simple tasks ("overthinking")

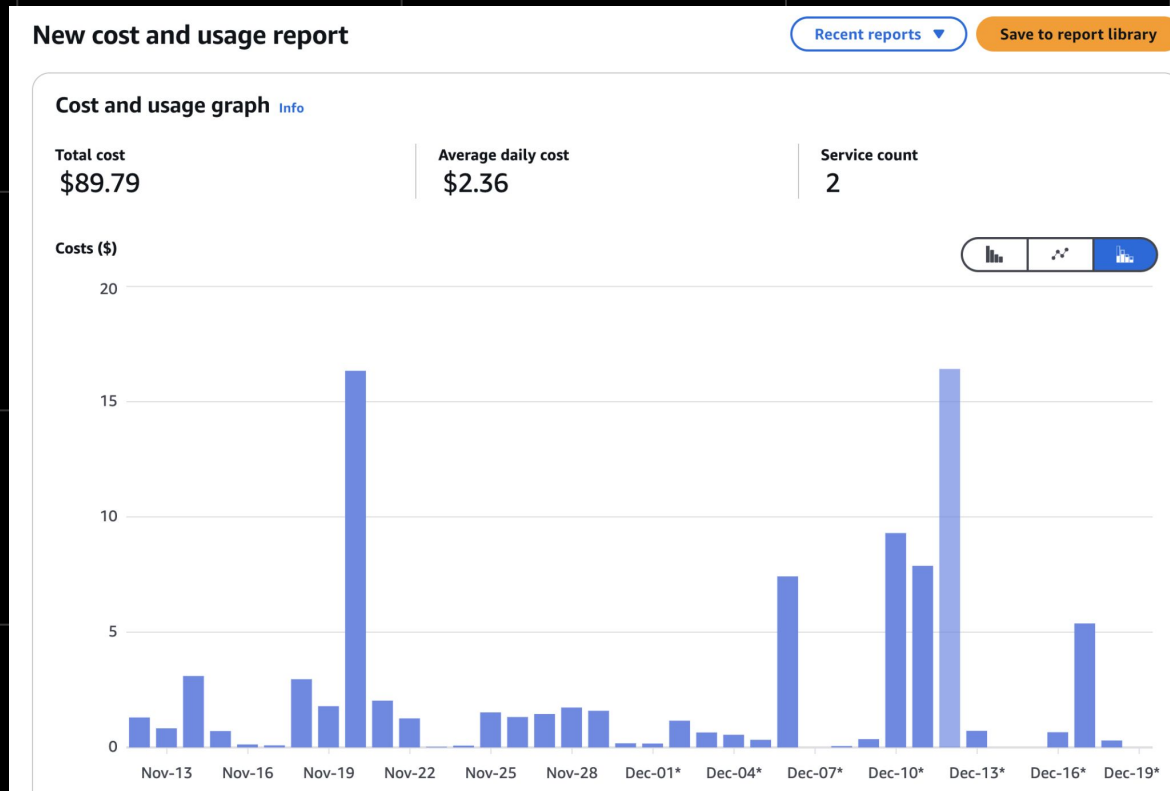
Reasoning models

비용이 상대적으로 비싸고 상대적으로 오래 걸렸지만, 특별히 성능이 나아지지 않았음. 별도의 톨 (별도 코드 조회 등)과 연동한다면 다른 결과가 나올 수도 있을 것 같음.

ID	Model	Evaluation Time	Precision	W Precision	Recall	W Recall	F1 Score	W F1 Score	Accuracy
152	us.anthropic.claude-3-7-sonnet-20250219-v1:0	1010	0.961904761904762 (7)	0.9625 (7)	0.8183640920483027 (22)	0.7741028708133971 (26)	0.8629800307219663 (18)	0.8326612903225806 (24)	0.92653061224489 (25)
153	us.deepseek.r1-v1:0	539	0.9380952380952381 (10)	0.93125 (22)	0.7899187768752987 (31)	0.7367683946488295 (38)	0.8262250354687329 (33)	0.784420359052712 (36)	0.89795918367346 (43)

05 | 주요 인사이트

비용(일반 모델 사용)



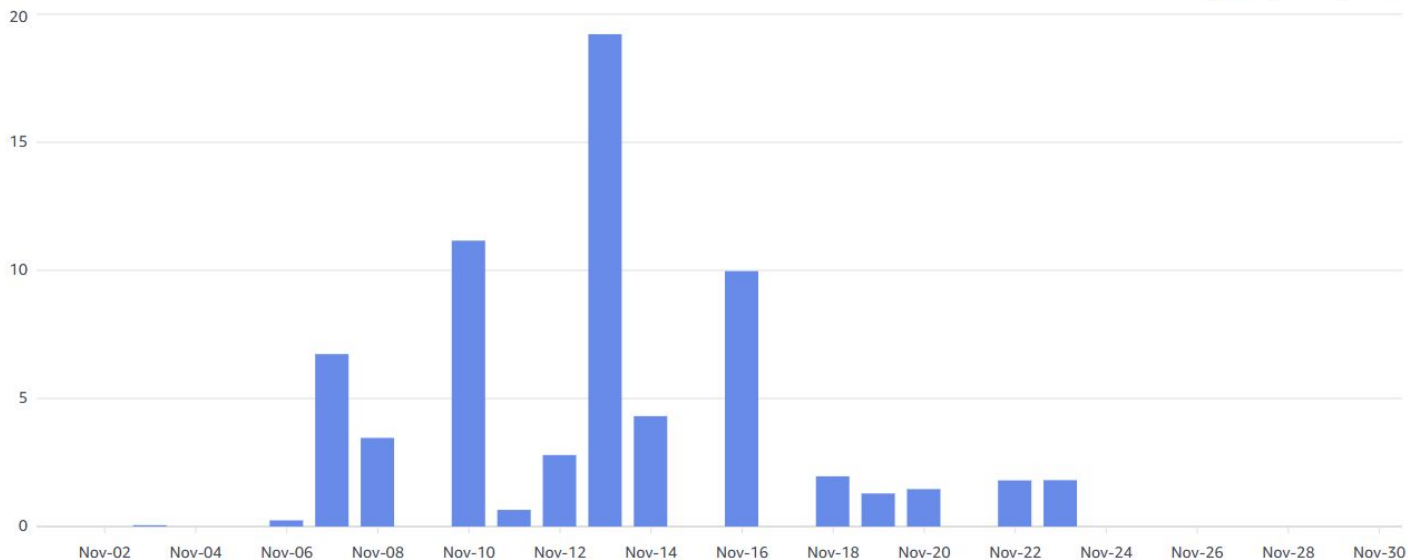
비용 - 테스트하려고 쓴돈

Cost and usage graph [Info](#)

Total cost
\$66.89

Average daily cost
\$2.23

Costs (\$)



Secrets



Brainiac WORKFLOW 11:58 PM

Improve fetch Og_tag for customer

<https://github.com/pull/19868>

No	Type	Answer	Accuracy	Reason
1	new_endpoint	Yes	95%	A new HTTP API endpoint is added in the form of the ' function, which makes a GET request to with various parameters.
2	vulnerability	Yes	100%	The 'scrapfly' function uses a <u>hardcoded API key ('scp-live-)</u> which could be a security vulnerability if exposed.
3	auths	No	90%	No significant changes or additions to authentication and authorization mechanisms are visible in this diff.
4	secrets	Yes	100%	The function contains a <u>hardcoded API key ('scp-live-)</u> which is sensitive information.
5	pii	No	85%	There doesn't appear to be any new collection of personal information in the added code.
6	new_package	No	90%	The diff does not show any changes to package manager files or explicit addition of new external packages.
7	security_patch	No	85%	The changes do not appear to be explicitly patching any security vulnerabilities, but rather adding new functionality.

신규 엔드포인트 & 파라미터



Brainiac WORKFLOW 5:57 PM

Added `is_force_update` for updating channel and mcls manually

<https://github.com/>

model_id: meta.llama3-1-405b-instruct-v1:0

No	Type	Answer	Reason
1	new_endpoint	True	<u>A new user-input parameter 'is_force_update' has been added to the existing endpoint 'api_private_channels_info'.</u>
2	vulnerability	False	No security vulnerabilities are detected in the new code.
3	auths	False	No changes or additions to authentication and authorization are detected.
4	secrets	False	No sensitive information is present in the new code.
5	pii	False	No collection of personally identifiable information is present in the new code.
6	new_package	False	No new external packages have been added.
7	security_patch	False	No changes to security vulnerabilities have been made.



Brainiac WORKFLOW 9:30 AM

remove sandbox push-providers from api response

<https://github.com/>

model_id: meta.llama3-1-405b-instruct-v1:0

No	Type	Answer	Reason
1	new_endpoint	True	<u>New HTTP API endpoints have been added, including '/applications/sandbox/push/{?P<token_type>(fcm apns huawei apns-voip fcm-voip huawei-voip calls-remote))/?\$'.</u>
2	vulnerability	False	No obvious security vulnerabilities are present in the new code.
3	auths	True	Changes have been made to authentication and authorization, including the addition of a new 'is_sandbox' parameter.
4	secrets	False	No sensitive information or hard-coded secrets are present in the new code.
5	pii	False	No collection of personally identifiable information is present in the new code.
6	new_package	False	No new external packages have been added.
7	security_patch	False	No changes have been made to patch security vulnerabilities.

개인정보수집



Brainiac WORKFLOW 12:00 PM

whatsapp integration

<https://github.com/>

model_id: meta.llama3-1-405b-instruct-v1:0

No	Type	Answer	
Reason			
1	new_endpoint	True	No
New HTTP API endpoints are added: /bots/{bot_userid}/integrations/, /bots/{bot_userid}/integrations/{integration_id}/, and /omega/webhook/applications/{app_id}/bots/{bot_id}/channels/{channel_id}/channel_bindings/{channel_binding_key}/			
2	vulnerability	False	No
obvious security vulnerabilities are detected in the new code			
3	auths	True	Changes are made to
authentication and authorization in the new code, specifically in the [REDACTED]-integration.py file			
4	secrets	False	Personal
No hard-coded secrets are found in the new code			
5	pii	True	Personal
information is collected in the new code, <u>specifically user IDs and phone numbers</u>			
6	new_package	False	No
New external packages are not added in the package manager			
7	security_patch	False	No
changes are made to patch security vulnerabilities in the new code			

취약점



Brainiac WORKFLOW 5:18 PM


Update cron trusted relationships (dev)

<https://github.com/>

model_id: meta.llama3-1-405b-instruct-v1:0

No	Type	Answer
Reason		
1	new_endpoint	False
input parameters are introduced in the provided diff.		No new HTTP API endpoints or user-
2	vulnerability	True
The change in the condition test from <u>'StringEquals'</u> to <u>'StringLike'</u> and the introduction of wildcards in the values may potentially introduce a security vulnerability by allowing more permissive access, but without more context, it's difficult to determine the likelihood of exploitation.		
3	auths	False
authentication and authorization mechanisms in the provided diff.		There are no changes or additions to
4	secrets	False
are introduced in the provided diff.		No hard-coded secrets
5	pii	False
of personal information in the new code.		There is no collection
6	new_package	False
packages are added in the package manager.		No new external
7	security_patch	False
explicitly patch security vulnerabilities.		There are no changes that

Auths

 **Brainiac** WORKFLOW 6:14 PM

fix typo

<https://github.com/>

model_id: meta.llama3-1-405b-instruct-v1:0

No	Type	Answer
Reason		
1	new_endpoint	False
endpoints or user-input parameters are added.		No new HTTP API
2	vulnerability	False
The change is a permission update, which does not introduce a vulnerability.		No security vulnerabilities are detected in the new code.
3	auths	True
<u>updating the permission from</u> MODERATION_SUPERGROUPCHANNEL_VIEW to MODERATION_SUPERGROUPCHANNEL_VIEW.		
4	secrets	False
hard-coded secrets are found in the new code.		No sensitive information or
5	pii	False
information is detected in the new code.		No collection of personal
6	new_package	False
packages are added in the package manager.		No new external
7	security_patch	False
to patch security vulnerabilities.		No changes are made

New package



Brainiac WORKFLOW 12:38 PM

Announcement on EKS (#19805)

<https://github.com/>

model_id: meta.llama3-1-405b-instruct-v1:0

No	Type	Answer	Reason
1	new_endpoint	False	No new HTTP API endpoints or user-input parameters are introduced in the provided diff.
2	vulnerability	False	No security vulnerabilities are detected in the new code.
3	auths	True	Changes are made to authentication and authorization, specifically in the <code>class</code> and the <code>file</code> .
4	secrets	False	No sensitive information or hard-coded secrets are found in the new code.
5	pii	False	No collection of personal information is detected in the new code.
6	new_package	True	<u>A new external package <code>kubernetes</code> is added in the <code>requirements-py3.8.txt</code> and <code>requirements.in</code> files.</u>
7	security_patch	False	No changes are made to patch security vulnerabilities.

Security patch



Brainiac WORKFLOW 5:00 AM

<https://github.com/>

model_id: meta.llama3-1-405b-instruct-v1:0

No	Type	Answer	Reason
1	new_endpoint	False	No new HTTP API endpoints or user-input parameters are added.
2	vulnerability	False	No security vulnerabilities are detected in the new code. The change is a simple conditional statement update.
3	auths	False	No changes or additions to authentication and authorization are made.
4	secrets	False	No sensitive information or hard-coded secrets are found in the new code.
5	pii	False	No collection of personal information is detected in the new code.
6	new_package	False	No new external packages are added in the package manager.
7	security_patch	True	<u>The change seems to be related to a security patch, as indicated by the comment referencing a security issue (SECURE-522).</u>

결론

- 생각보다 유지 비용이 저렴하고, 쓴 만큼 지불하기에 언제든지 멈출 수 있음.
- 취약점을 찾는건 아직 잘못함. 실제로도 오탐이 많았음.
- 하지만 코드를 이해하고 그것에 대한 질문은 굉장히 잘 수행 함(권한, 신규 엔드포인트 등)
- 언어나 프레임워크 등의 제약을 받지 않기에 확장이 자유로움.
- 프롬프트 엔지니어링에 따라 성능이 올라가는것을 논리적으로 이해하기 어려웠음.
- RAG/Tool calling 등 다른 기술들을 사용해 성능을 올리는 시도도 해보고 싶음.
- 실제로 보안 엔지니어가 봐야할 PR의 숫자를 줄여줌. ~~사람을 정말 적게 뽑아도~~
~~관찰자 없을까란 생각아...~~

Q&A

출처

Page 7. <https://www.akooda.co/blog/large-language-models-explained>

Page 43. <https://sebastianraschka.com/blog/2025/understanding-reasoning-llms.html>

Page 43. <https://news.mit.edu/2024/reasoning-skills-large-language-models-often-overestimated-0711>