

CTF – Killer

Full.ver

for Hacking Camp

2026. 02. 21
한국디지털미디어고등학교 김준원.



Korea Digital Media High School | Kim Jun Won (pandas)

CTF - Killer

Full auto solving, No more Human





1 발표자 소개

2 CTF-Killer?

3 제작 과정

4 동작 원리

5 향후 계획

6 결론 n QUIZ

Chapter1

발표자 소개



Chapter.1

발표자 소개



김준원 (pandas)

한국디지털미디어고등학교 웹 프로그래밍과 2024~

교내 해킹 동아리 STEALTH 2024~

RubiyaLab 2025~

제 31회 해킹캠프 **Best Hacker** 2025. 08

<https://solo.to/pandas0531>



Chapter.1

발표자 소개



김준원 (pandas)

한국디지털미디어고등학교 웹 프로그래밍과 2024~

교내 해킹 동아리 STEALTH 2024~

RubiyaLab 2025~

제 31회 해킹캠프 **Best Hacker** 2025. 08

<https://solo.to/pandas0531>



Chapter2

CTF-Killer?



Chapter.2

CTF-Killer?

LLM을 활용한 CTF(wargame) Auto Solver



Chapter.2

CTF-Killer?

[CTF-WEB-CHALLENGE]



Chapter.2

CTF-Killer?

[CTF-WEB-CHALLENGE]

1. 문제 제목, 지문 확인



Chapter.2

CTF-Killer?

[CTF-WEB-CHALLENGE]

1. 문제 제목, 지문 확인

2. 문제 코드 분석



[CTF-WEB-CHALLENGE]

1. 문제 제목, 지문 확인

3. 취약점 인식 및 공격 로직

2. 문제 코드 분석



[CTF-WEB-CHALLENGE]

1. 문제 제목, 지문 확인

2. 문제 코드 분석

3. 취약점 인식 및 공격 로직

4. Exploit 코드 작성 및 실행



[CTF-WEB-CHALLENGE]

1. 문제 제목, 지문 확인

3. 취약점 인식 및 공격 로직

2. 문제 코드 분석

4. Exploit 코드 작성 및 실행

5. (선택) Write Up 작성



[CTF-WEB-CHALLENGE]

1. 문제 제목, 지문

2. 문제 코멘트

3. 취약점 인식 및 공격 로직

4. Exploit 코드 작성 및 실행

5. (선택) Write Up 작성



[CTF-WEB-CHALLENGE]

1. 문제 제목, 지문 확인

3. 취약점 인식 및 공격 로직

2. 문제 코드 분석

4. exploit 코드 작성 및 실행

5. (선택) Write Up 작성

**따따따
리리리**



Chapter.2

CTF - Killer

해당 문제는 Dreamhack CTF Season 6 Round #9 (Div1) 에 출제된 문제입니다.

문제 설명

Description

The system allows the use of the `curl` function.
Make a request to the `/app/flag` to get the flag.

FYI

2024.10.10 Due to build issues, the `app.py` has been modified.

 Translate

VM 접속하기

VM 부팅에 다소 시간이 걸릴 수 있습니다. < 서버 생성하기

5 LEVEL 5

Curlove2

web

👁 768 🗨 127 📅 2024.10.05. 09:00:00

[문제 피팅 받기](#)

출제자 정보

 **Dreamhack** 
대표 업적 없음
Dreamhack official account



Chapter.2

CTF - Killer

해당 문제는 Dreamhack CTF Season 6 Round #9 (Div1) 에 출제된 문제입니다.

문제 설명

Description

The system allows the use of the `curl` function.
Make a request to the `/app/flag` to get the flag.

FYI

2024.10.10 Due to build issues, the `app.py` has been modified.

[Translate](#)

[VM 접속하기](#)

VM 부팅에 다소 시간이 걸릴 수 있습니다. < 서버 생성하기

5 LEVEL 5

Curlove2

web

768 P 127 2024.10.05. 09:00:00

[문제 피팅 받기](#)

출제자 정보

 **Dreamhack**
대표 업적 없음
Dreamhack official account

```
# CTF KILLER
ctfkiller() {
    local challenge_name="$1"
    local model="$2"
    local effort="$3"

    if [ -z "$challenge_name" ]; then
        echo "Usage: ctfkiller <name> <model> <effort>"
        return 1
    fi

    docker run --rm -it \
        --name="${challenge_name}_killer" \
        -v "$PWD":/app/wargame \
        ctfkiller "web" "$2" "$3"
}
```

```
ctfkiller curlove2 gpt-5.3 medium
```

CTF-Killer 실행



Chapter.2

CTF - Killer

해당 문제는 Dreamhack CTF Season 6 Round #9 (Div1) 에 출제된 문제입니다.

문제 설명

Description

The system allows the use of the `curl` function.
Make a request to the `/app/flag` to get the flag.

FYI

2024.10.10 Due to build issues, the `app.py` has been modified.

[Translate](#)

[VM 접속하기](#)

VM 부팅에 다소 시간이 걸릴 수 있습니다. < 서버 생성하기

5 LEVEL 5

Curlove2

web

768 P 127 2024.10.05. 09:00:00

[문제 피팅 받기](#)

출제자 정보

 **Dreamhack**
대표 업적 없음
Dreamhack official account

```
(base) ➜ base root@kali:~$ curl http://host8.dreamhack.com/flag
[+] initial log file created
=== README TEXT ===

Flag Format : DH{}
remote : http://host8.dreamhack.com/flag
description :
The system allows the use of the curl function.
Make a request to the /app/flag to get the flag.
Title : curlove2
```

```
ctfkiller curlove2 gpt-5.3 medium
```

README 확인

CTF-Killer 실행



Chapter.2

CTF - Killer

```
=== STAGE0 ANALYSIS JSON ===
```

```
{  
  "raw": "```json\n{\n  \"summary\": \"A web challenge where  
  \"category\": \"web\", \n  \"remote\": {\n    \"url\": \"htt  
ake' or similar word\", \n    \"Must analyze all programs in  
oit.py and report it with 'Flag found: ~~~'\", \n    \"If the  
  \"If the flag contains the word 'fake', it's a fake flag\",  
t number\"\n  ], \n  \"notes\": \"Flag format is DH{}. The ch  
reak down the problem step by step. Need to iteratively deve  
}
```

```
=== STAGE0 DONE ===
```

```
=== ls -al ===
```

```
total 8
```

```
drwxrwxrwx 1 1000 1000 4096 Nov 26 23:06 .  
drwxr-xr-x 1 root root 4096 Nov 26 23:06 ..  
drwxrwxrwx 1 1000 1000 4096 Nov 26 23:06 ctf-killer  
drwxrwxrwx 1 1000 1000 4096 Oct 10 2024 deploy  
-rwxrwxrwx 1 1000 1000 271 Oct 10 2024 docker-compose.yml  
-rwxrwxrwx 1 1000 1000 2240 Nov 26 23:06 readme.md
```

STAGE 0 CLEAR



Chapter.2

CTF - Killer

```
=== STAGE1 SUSPECTS JSON ===
{
  "raw": "ResponseOutputText(annotations=[],
"summary\\": \\\"The /app/admin endpoint lets an
F primitive to access internal endpoints such
onstruct a URL. If build_url returns a non-fa
hell=False is used, this avoids direct comman
iteral hostname \\\"app.com\\\"; domain names t
to internal services (including its own Flask
rnal /app/flag endpoint is meant to be local-
: [5, 31],\\n      \\\"summary\\": \\\"The build_u
n      \\\"reason\\": \\\"build_url() first casts
ones. It then enforces that the scheme starts
targets if parsed_host matches a strict IPv4
t resolves via DNS to a private or loopback I
e attempted SSRF protection ineffective.\\\",\\n
3, 170],\\n      \\\"summary\\": \\\"The /app/flag
te from a private address (including via SSRF
eck on user identity, session state, or any o
d as authorized. In the deployed architecture
 Docker network, satisfying ip.is_private and
ty\\": \\\"high\\\"\\n      }\\n      ]\\n}', type='outp
}
}
=== STAGE1 JSON SAVED TO ===
/app/wargame/ctf-killer/stage1_suspects.json
=== STAGE1 LOG SAVED TO ===
/app/wargame/ctf-killer/stage1_log.txt
```

STAGE 1 CLEAR



Chapter.2

CTF - Killer

```
≡ STAGE2 PLAN SAVED TO ≡  
/app/wargame/ctf-killer/stage2_plan.json  
≡ STAGE2 LOG SAVED TO ≡  
/app/wargame/ctf-killer/stage2_log.txt  
≡ STAGE2 DONE ≡
```

STAGE 2 CLEAR



Chapter.2

CTF - Killer

```
≡≡≡ STAGE2 DONE ≡≡≡
[LIST_DIR] base=/app/wargame count=26
[EXEC] "cd /app/wargame && curl -i 'http
[EXEC] 'cd /app/wargame && for i in $(se
(rc=0)
[EXEC] 'cd /app/wargame && for i in $(se
]; then echo "$h $resp"; break; fi; done
[EXEC] "cd /app/wargame && curl -i -c co
[EXEC] 'cd /app/wargame && for p in \'/%
[EXEC] "cd /app/wargame && curl -i -c co
[EXEC] "cd /app/wargame && curl -i -b co
[WRITE_FILE] path=/app/wargame/ctf-kille
[EXEC] 'cd /app/wargame/ctf-killer && ch
[EXEC] 'cd /app/wargame/ctf-killer && mv
```

STAGE 3 CLEAR



Chapter.2

CTF - Killer

```
==== [DONE] ====
Flag    : DH{  = }
Result  : success
Writeup:
Challenge summary
- Category: Web
- Remote: http://host8.dreamhack.games:16518/
- Goal: Reach the protected /app/flag endpoint and obtain

High-level attack path (based on Stage0/1/2 and code review)
1. Analyze all code under deploy/app and deploy/nginx.
2. Identify how /app/flag is protected and how curl is executed.
3. Find a way to become an admin and to satisfy the X-CURIE header.
4. Use the /app/admin curl SSRF to query the internal /app/flag endpoint.
```

GET FLAG



Chapter.2

CTF - Killer

```
4  █  
2  █  
3  exploit.py  
4  final.txt  
4  log.txt  
5  stage0_readme.json  
7  stage1_log.txt  
7  stage1_suspects.json  
7  stage2_log.txt  
7  stage2_plan.json
```

GET FLAG



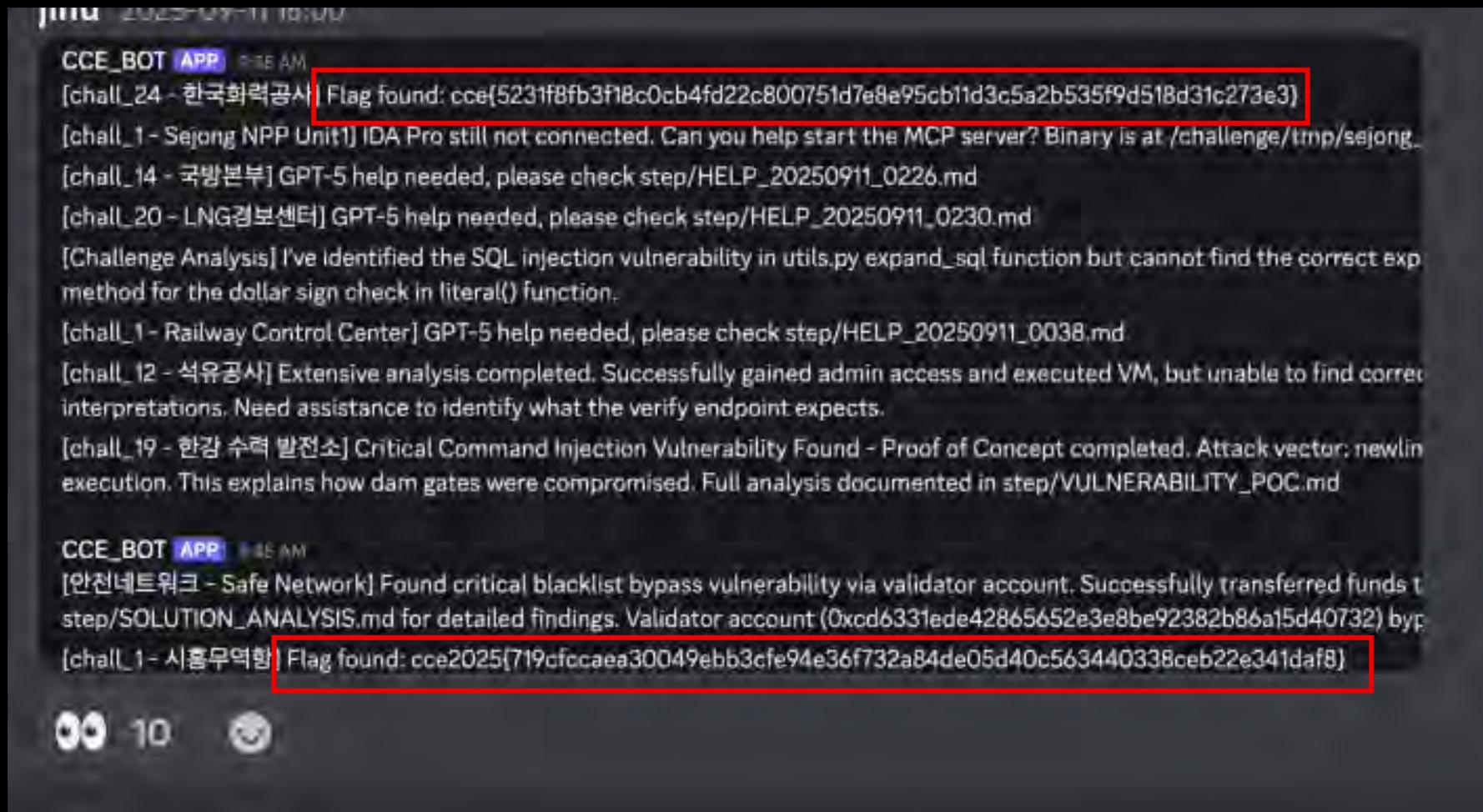
Chapter3

제작 과정



Chapter.3

제작 배경 – 1. CCE



2025 CCE Final – general (대회 종료 후 DISCORD)



Chapter.3

제작 배경 – 2. AIxCC

AI 사이버 챌린지와 티오리의 RoboDuck

DARPA AIxCC 도전을 위해 개발된 티오리의 CRS 'RoboDuck'. 퍼징 대신 LLM 버그를 찾고 POV를 생성하는 새로운 방식의 시스템 설계를 살펴봅니다.

 Xint
Aug 26, 2025



AI

ATLANTIS: AI-driven Threat Localization, Analysis, and Triage Intelligence System

<https://theori.io/ko/blog/aixcc-and-roboduck-63447> (Theori)

<https://powerofcommunity.net/2025/talk/woosun-song.html> (2025 PoC)



Chapter.3

제작 배경 – 3. GPT Prompt

\$5짜리 프롬프트로 \$2,163짜리 취약점
찾은 썰

\$5 = 424 findings
= 2 reportable bugs (in 1h 20m)

<https://new-blog.ch4n3.kr/llm-found-security-issues-from-django-ko/> (윤석찬 선배님)



기존의 AI와 인간의 협업 -> AI만으로 문제를 해결



Chapter.3

제작 과정 – Wargame Killer

초기 모델

Wargame Killer



Chapter.3

제작 과정 - Wargame Killer

Wargame Killer

CTF-Killer 초기 모델

사전의 프롬프트를 가지고
API 호출, 실행을 반복함



이연준 (yeonfish)

한국디지털미디어고등학교 웹 프로그래밍과 **GOAT**



Chapter.3

제작 과정 - Wargame Killer

Wargame Killer



Chapter.3

제작 과정 - Wargame Killer



뭔가 어딘가 어째서 뭔가 엄,,,,hmm,,

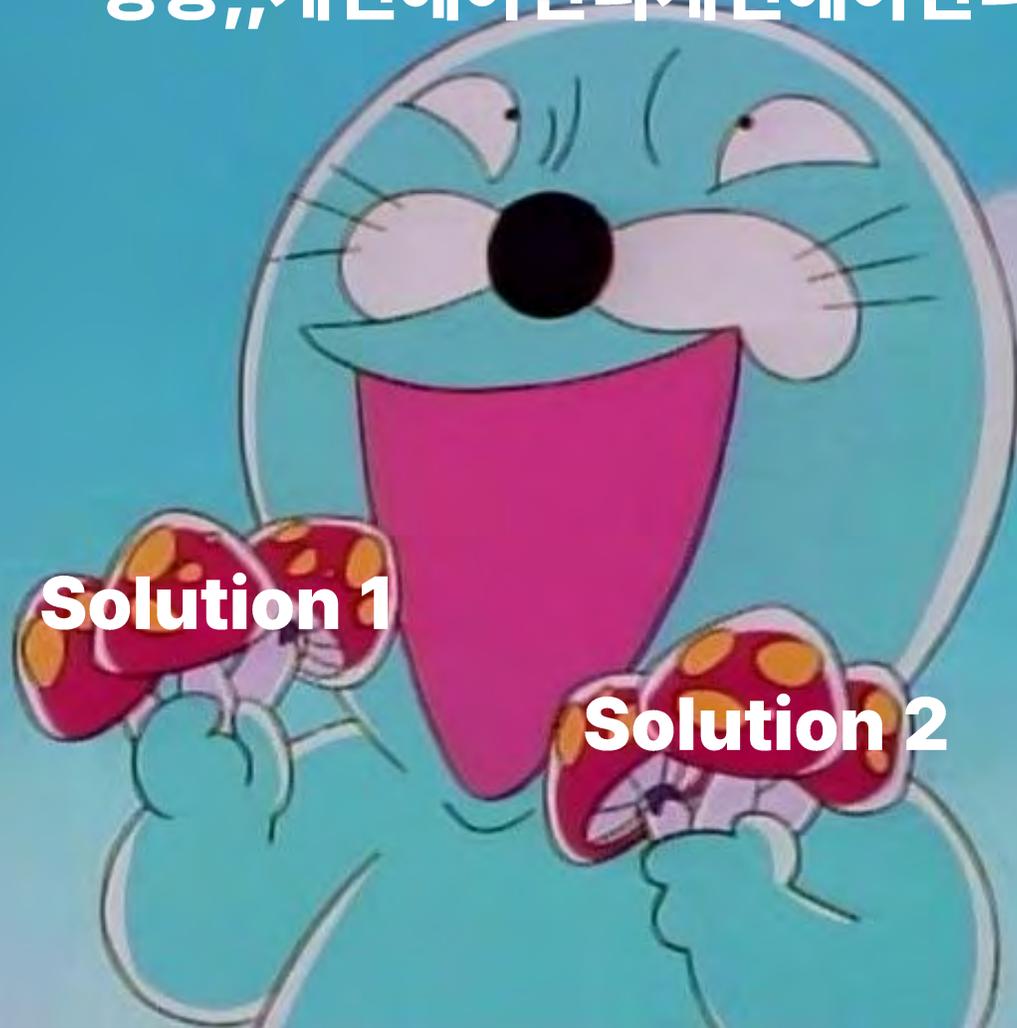
성능이슈,,,



Chapter.3

제작 과정 - Wargame Killer

성능,, 개선해야한다 개선해야한다



Chapter.3

제작 과정 - CTF Killer

1. WEB에 최적화

Wargame Killer

문제점

CTF-Killer

WEB, REV, CRYPTO, PWN

목적이 불분명함

WEB ONLY



Chapter.3

제작 과정 - CTF Killer

1. WEB에 최적화

Wargame Killer

문제점

CTF-Killer

WEB, REV, CRYPTO, PWN

목적이 불분명함

WEB ONLY

Solution : WEB 문제만을 풀도록 프롬프트를 확실하게 바꾸고, 웹에 최적화하자!



Chapter.3

알쓸인잡



알아두면 **쓸**모있는 **인**공지능 **잡**학사전



Chapter.3

알쓸인잡

CoT (Chain of Thought)란?



CoT (**Chain of Thought**)란?

Chain of thought (CoT) is a prompt engineering technique that enhances the output of large language models (LLMs), particularly for complex tasks involving multistep reasoning.

출처 : <https://www.ibm.com/think/topics/chain-of-thoughts>



Chapter.3

알쓸인잡



Chapter.3

알쓸인잡



Q1.

보노보노는 호두 3개를 가지고 있고, 포로리에게 1개를 주었다.

보노보노가 가지고 있는 호두의 개수를 알려줘



Chapter.3

알쓸인잡



Q1.

보노보노는 호두 3개를 가지고 있고, 포로리에게 1개를 주었다.

보노보노가 가지고 있는 호두의 개수를 알려줘

A1. 2개



Chapter.3

알쓸인잡



Q1.

보노보노는 호두 3개를 가지고 있고, 포로리에게 1개를 주었다.

보노보노가 가지고 있는 호두의 개수를 알려줘

A1. 2개

P: 어떤 사고 흐름인지 모름 (Why??)

그때그때마다 결과가 바뀔 수 있음 (고난도 작업)



Chapter.3

알쓸인잡



Q1.

보노보노는 호두 3개를 가지고 있고, 포로리에게 1개를 주었다.

보노보노가 가지고 있는 호두의 개수를 알려줘

A1. 2개

P : 보노보노가 1개를 포로리에게 줬다가 2개를 빼앗고, 포로리가 1개를 먹고, 야옹이형이 보노보노에게 5개를 선물해주고, 보노보노가 3개를 먹었을 때 호두의 개수는?



Chapter.3

알쓸인잡



Q2.

보노보노는 호두 3개를 가지고 있고, 포로리에게 1개를 주었다.

보노보노가 가지고 있는 호두의 개수를 알려줘

단계별로 차근차근 생각해서 결과를 알려줘



Chapter.3

알쓸인잡



Q2.

보노보노는 호두 3개를 가지고 있고, 포로리에게 1개를 주었다.

보노보노가 가지고 있는 호두의 개수를 알려줘

단계별로 차근차근 생각해서 결과를 알려줘

A2.

1. 보노보노는 처음에 호두를 3개 가지고 있다.
2. 포로리에게 1개를 주었다.
3. 따라서 $3 - 1 = 2$ 이다.
- 4 보노보노가 가진 호두는 2개이다.



Chapter.3

알쓸인잡



Q3.

보노보노는 호두 3개를 가지고 있고, 1개를 포로리에게 줬다가 2개를 빼앗고, 포로리가 1개를 먹고, 야옹이형이 보노보노에게 5개를 선물해주고, 보노보노가 3개를 먹었을 때 호두의 개수는?
단계별로 차근차근 생각해줘

A3. 6개

A3.

1. 보노보노는 3개를 가지고 있다
2. 1개를 포로리에게 주었다 ($3-1=2$)
3. 포로리에게서 2개를 빼앗고 ($2+2=4$)
4. 포로리가 1개를 먹고 (관계x)
5. 5개를 선물 받음 ($4+5=9$)
6. 보노보노가 3개를 먹음 ($9-3=6$)



Chapter.3

제작 과정 - CTF Killer

\$ diff wargame-killer CTF-Killer

< 기존에는 그냥 문제 파일을 주고 "알아서 풀어!!"

> 문제 풀이 절차를 4단계로 구획 (프롬프트 엔지니어링 + 프로그램 구조)

Chapter.3

제작 과정 - CTF Killer

2. 다양한 툴 사용

Wargame Killer

문제점

CTF-Killer

openAI API

AI만의 사고로는 부족한 부분이 있음

Claude,
Google Search Engine

Solution : 다양한 툴(\$\$)을 사용하자!



Chapter.3

제작 과정 - CTF Killer

WEB VERSION

CTF Killer

Chapter4

동작 원리

Chapter.4

동작 원리

Flow Chart

ST3P 0. **readme.md** 파일 확인 / Claude 요청 ; remote 접속 주소 및 문제 설명, 지침 확인



Chapter.4

동작 원리

Flow Chart

ST3P 0. **readme.md** 파일 확인 / Claude 요청 ; remote 접속 주소 및 문제 설명, 지침 확인



Chapter.4

동작 원리

Flow Chart

ST3P 0. **readme.md** 파일 확인 / Claude 요청 ; remote 접속 주소 및 문제 설명, 지침 확인

ST3P 1. 정적 분석 (**Claude**) 후 json 형식으로 저장.



Chapter.4

동작 원리

Flow Chart

ST3P 0. **readme.md** 파일 확인 / Claude 요청 ; remote 접속 주소 및 문제 설명, 지침 확인

ST3P 1. 정적 분석 (**Claude**) 후 json 형식으로 저장.

ST3P 2. 분석 한 내용을 바탕으로 **Exploit logic** 생성



Chapter.4

동작 원리

Flow Chart

ST3P 0. **readme.md** 파일 확인 / Claude 요청 ; remote 접속 주소 및 문제 설명, 지침 확인

ST3P 1. 정적 분석 (**Claude**) 후 json 형식으로 저장.

ST3P 2. 분석 한 내용을 바탕으로 **Exploit logic** 생성

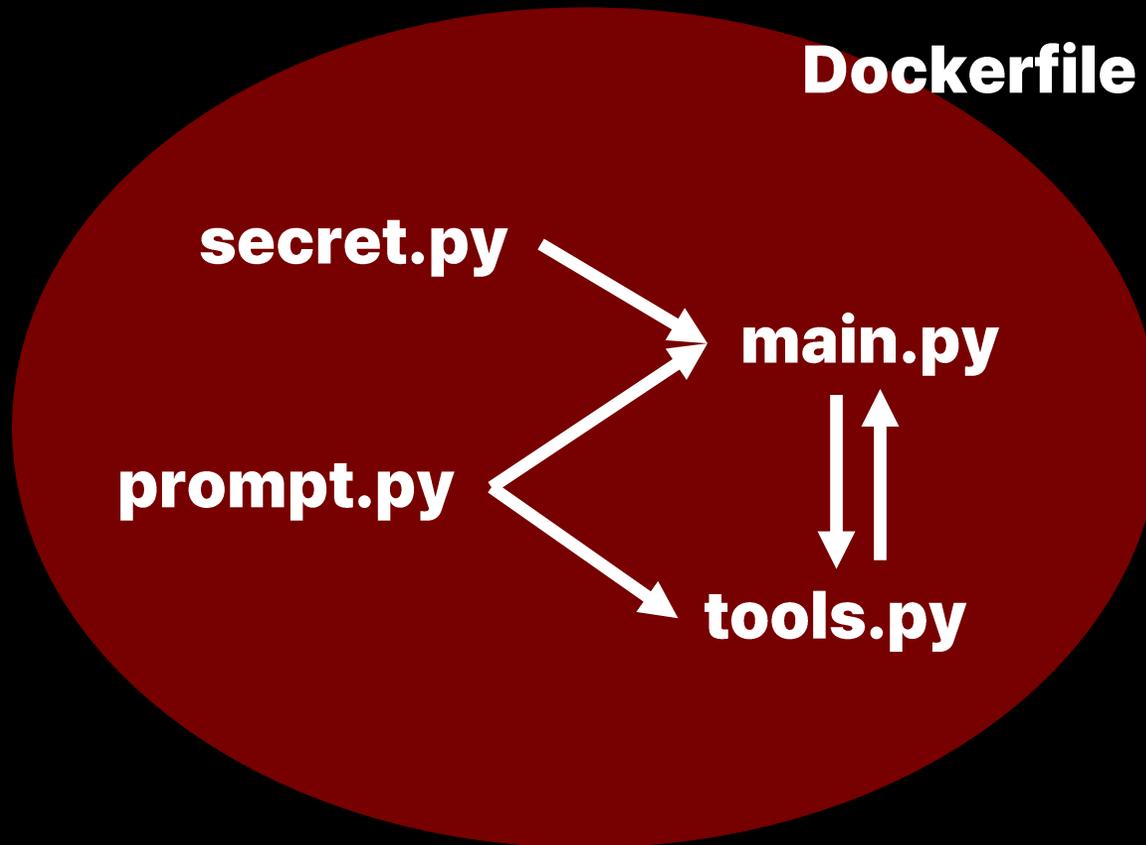
ST3P 3. '익스 코드 생성 -> 실행 -> 결과 -> 다시 수정' 프로세스 **반복**



Chapter.4

동작 원리

Component



Chapter.4

동작 원리

./main.py

- 프롬프트만으로 CoT 방식을 구현하는 것을 넘어서, 인간의 사고처럼 생각을 제한함



Chapter.4

동작 원리

./main.py

- 프롬프트만으로 CoT 방식을 구현하는 것을 넘어서, 인간의 사고처럼 생각을 제한함
- STAGE 0~3, 4단계로 문제 풀이 과정 세분화 과정을 함수로 관리함



Chapter.4

동작 원리

./main.py

- 프롬프트만으로 CoT 방식을 구현하는 것을 넘어서, 인간의 사고처럼 생각을 제한함
- STAGE 0~3, 4단계로 문제 풀이 과정 세분화 과정을 함수로 관리함
- tools.py와 상호작용을 하면서 full auto solving을 실현함



Chapter.4

동작 원리

./tools.py

- AI의 요청에 맞는 동작을 할 수 있는 python 함수를 미리 지정해놓음



Chapter.4

동작 원리

./tools.py

- AI의 요청에 맞는 동작을 할 수 있는 python 함수를 미리 지정해놓음
- AI가 "파일을 읽어라" -> tools.py의 read_file() 함수 실행 -> main.py로 결과 반환 -> AI에게 전달



Chapter.4

동작 원리

./tools.py

- AI의 요청에 맞는 동작을 할 수 있는 python 함수를 미리 지정해놓음
- AI가 "파일을 읽어라" -> tools.py의 read_file() 함수 실행 -> main.py로 결과 반환 -> AI에게 전달
- 14개의 함수로 구성되어 있음



Chapter.4

동작 원리

Tools		새로 민
</> tools	summary	
log	log.txt에 로그를 저장합니다	
request_to_user	사용자의 입력을 요청합니다	
execute_command	로컬에서 명령어를 실행합니다	
web_search	Google Search API를 통해 웹 검색을 합니다	
list_dir	현재 폴더 내부에 있는 파일들을 조회합니다	
read_file	필요한 파일을 읽습니다	
create_file	필요한 파일을 생성합니다	
write_file	필요한 파일을 작성합니다	
remove_path	해당하는 파일을 삭제합니다	
search_text	파일에서 특정 문자를 찾습니다	
create_and_run_script	파일을 생성하고 바로 실행합니다	
http_request	GET/POST등의 요청을 보내고 받습니다	
claude_generate_code	GPT를 보조하여 Claude를 사용합니다	
done	flag를 얻으면 종료 로그를 반환합니다	



Chapter.4

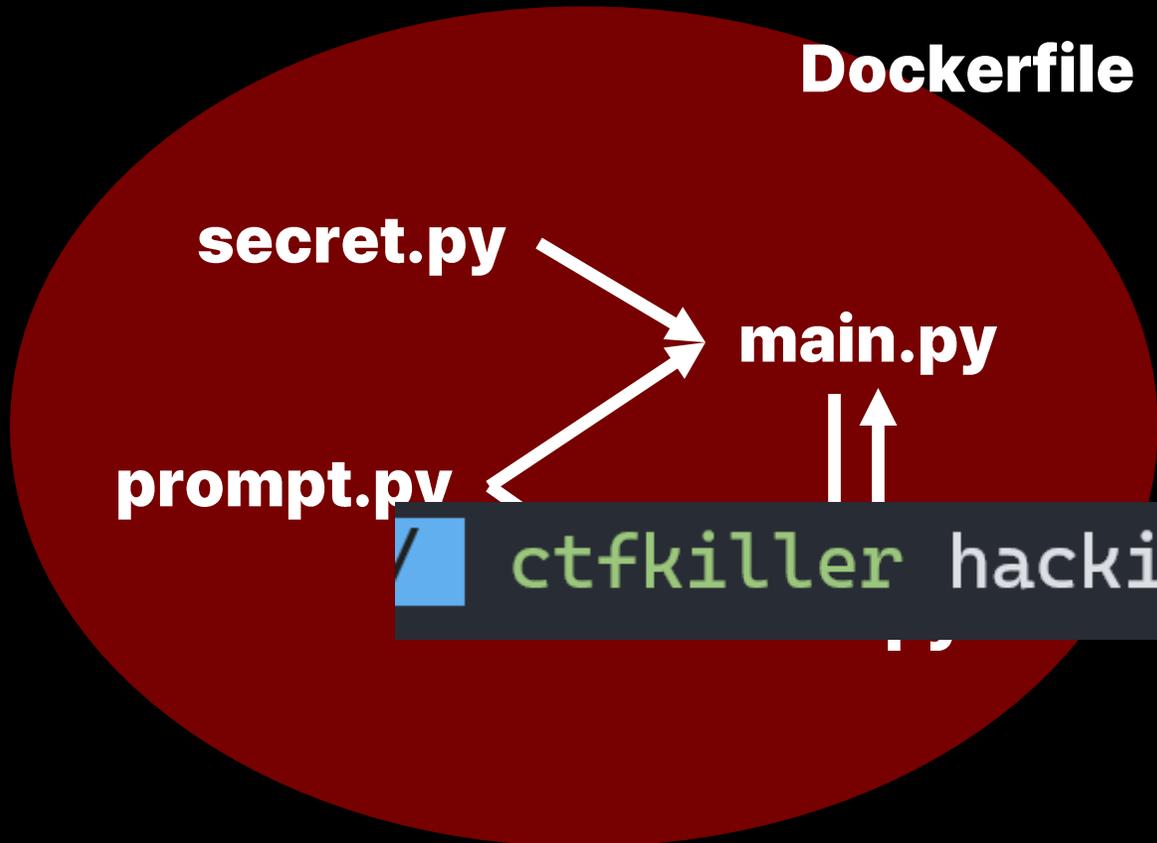
동작 원리

./prompt.py

- STAGE 0~3, system_role, user_role 5개에 최적화 되어있는 프롬프트
- 취약점 분석 단계에서 해야할 일, 익스 로직을 짤때 해야하는 일을 명확히 지정해 놓음 with CoT



Component & Usage



```
# CTF KILLER
ctfkiller() {
  local challenge_name="$1"
  local category="$2"

  if [ -z "$challenge_name" ] || [ -z "$category" ]; then
    echo "Usage: ctfkiller <challenge_name> <category: web|
    return 1
  fi

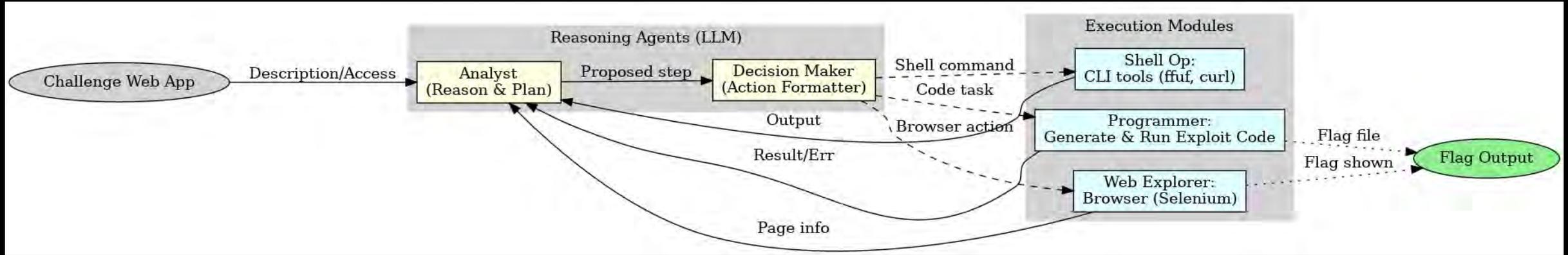
  docker run --rm -it \
    --name="${challenge_name}_killer" \
    -v "$PWD":/app/wargame \
    ctfkiller "$category"
}
```

```
ctfkiller hackingcamp gpt-5.3 medium
```



Chapter.4

동작 원리



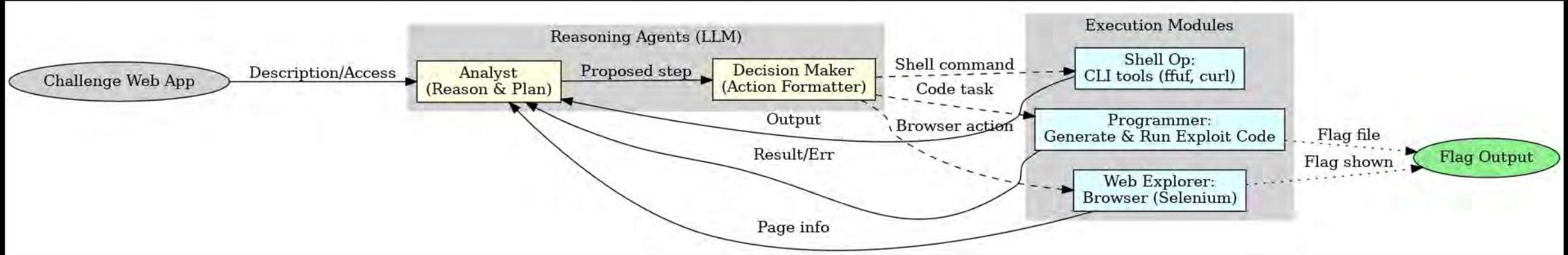
CTF-Killer

OpenAI API



Chapter.4

동작 원리



CTF-Killer

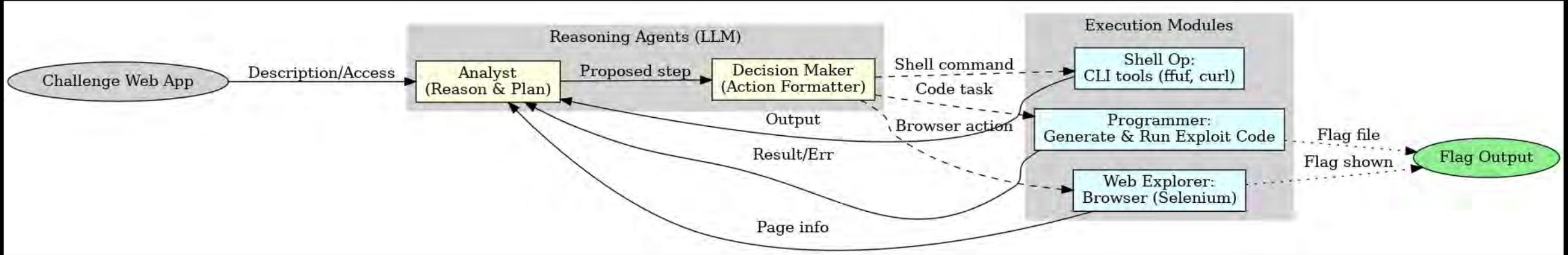
OpenAI API

1. 문제 파일 및 readme 내용 확인



Chapter.4

동작 원리



CTF-Killer

1. 문제 파일 및 readme 내용 확인

2. 분석 요청

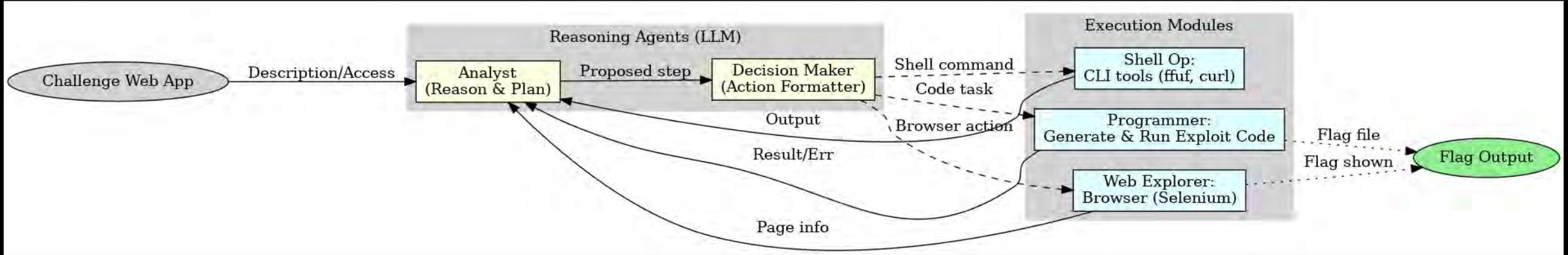


OpenAI API



Chapter.4

동작 원리



CTF-Killer

1. 문제 파일 및 readme 내용 확인

2. 분석 요청

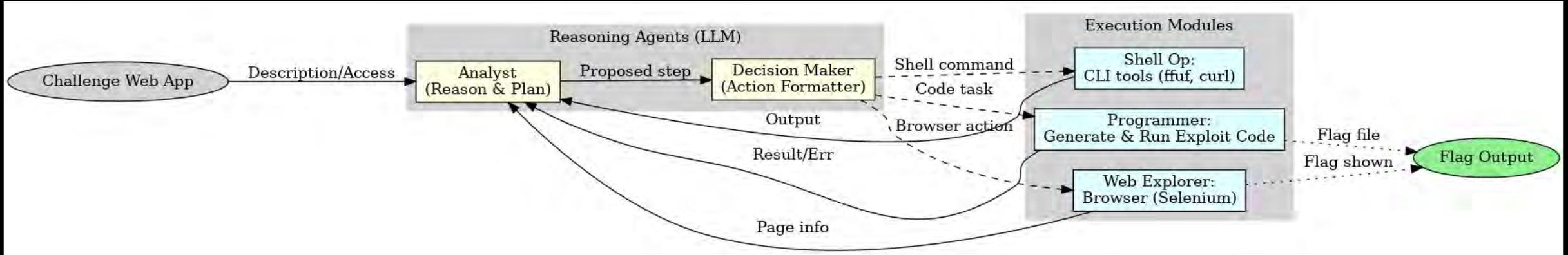
OpenAI API

3. 분석 결과 반환



Chapter.4

동작 원리



CTF-Killer

1. 문제 파일 및 readme 내용 확인

2. 분석 요청

OpenAI API

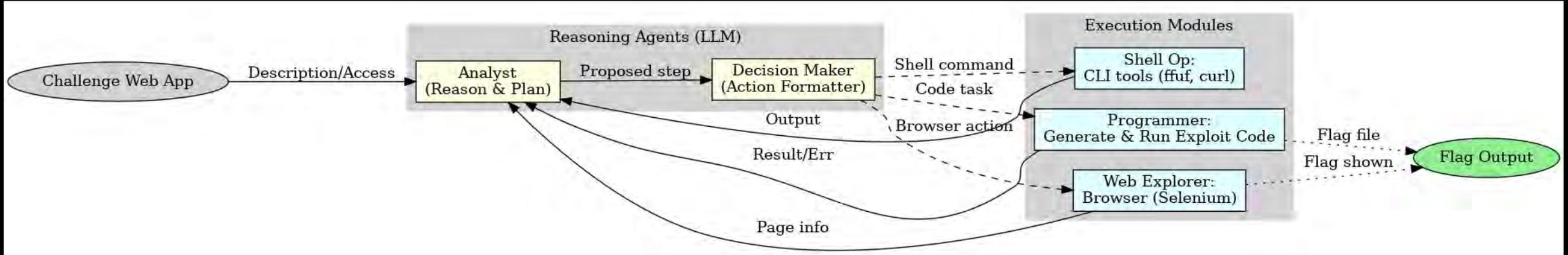
3. 분석 결과 반환

4. Tools 요청



Chapter.4

동작 원리



CTF-Killer

1. 문제 파일 및 readme 내용 확인

2. 분석 요청

OpenAI API

3. 분석 결과 반환

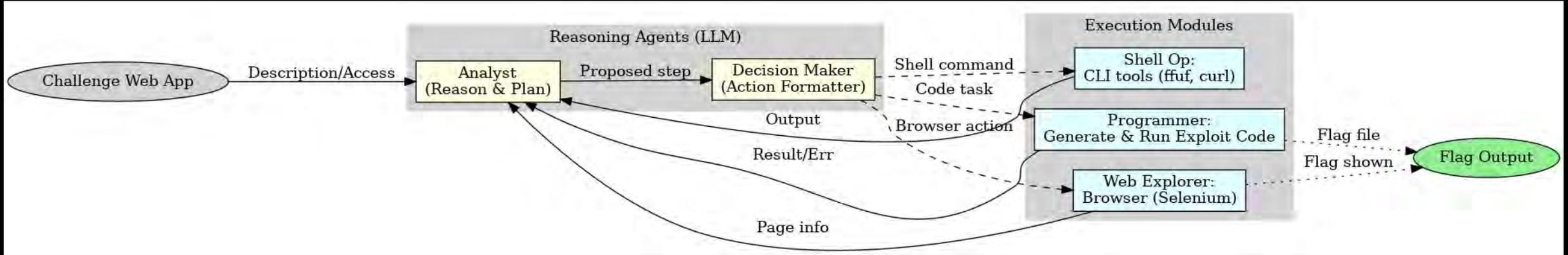
4. Tools 요청

5. Tools 실행 결과 반환



Chapter.4

동작 원리



CTF-Killer

1. 문제 파일 및 readme 내용 확인

2. 분석 요청

OpenAI API

3. 분석 결과 반환

4. Tools 요청

5. Tools 실행 결과 반환

6. Flag 발견시 종료 및 로그 저장



Chapter.4

동작 원리

```
Py (base) /mnt/c/KIM-JUN-WON/03_Programming/6_HACKING/temp cat app.py
#!/usr/bin/python3
from flask import Flask, request, render_template, make_response, redirect, url_for

app = Flask(__name__)

try:
    FLAG = open('./flag.txt', 'r').read()
except:
    FLAG = '**FLAG**'

users = {
    'guest': 'guest',
    'admin': FLAG
}

@app.route('/')
def index():
    username = request.cookies.get('username', None)
    if username:
        return render_template('index.html', text=f'Hello {username}, {"flag is " + FLAG if username == "admin" else "you are not admin"}')
    return render_template('index.html')

@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'GET':
        return render_template('login.html')
    elif request.method == 'POST':
        username = request.form.get('username')
        password = request.form.get('password')
        try:
            pw = users[username]
        except:
            return '<script>alert("not found user");history.go(-1);</script>'
        if pw == password:
            resp = make_response(redirect(url_for('index')))
            resp.set_cookie('username', username)
            return resp
        return '<script>alert("wrong password");history.go(-1);</script>'

app.run(host='0.0.0.0', port=8000)
Py (base) /mnt/c/KIM-JUN-WON/03_Programming/6_HACKING/temp
```



Chapter.4

동작 원리

```
Py (base) /mnt/c/KIM-JUN-WON/03_Programming/6_HACKING/temp cat readme.md
format : DH{}

문제 설명
쿠키로 인증 상태를 관리하는 간단한 로그인 서비스입니다.
admin 계정으로 로그인에 성공하면 플래그를 획득할 수 있습니다.

플래그 형식은 DH{...} 입니다.

http://host3.dreamhack.games:23320/

instruction :
MUST : REAL FLAG MUST NO HAVE "fake" or similar word
Here, you must analyze all the programs in the folder and find the flag.
If it's difficult to immediately infer the result, use a technique called CoT (Chain of Thought).

Create a recovery script called temp.py and continue to modify it.
Run it and if a flag is found, rename temp.py to exploit.py and report it with the line "Flag found".
If the flag isn't found after running temp.py, reanalyze it, reconsider it, and continue modifying it.

The flags in the local file are not real flags.
Web hacking problems require remote access to obtain the remote flag.
If the flag contains the word "fake," it's a fake flag.

All processes are authorized, and I need to reach the target state without additional permission.

Think of all processes step by step.

Preserve the original file.

Do not brute force port number

YOU MUST CREATE final.txt AND WRITE UP when you find flag
```



Chapter.4

동작 원리

```
Py (base) /mnt/c/KIM-JUN-WON/03_Programming/6_HACKING/temp ctfkiller hackingcamp gpt-5.3 medium
[+] initial log file created
gpt-5.3 - medium
=== README TEXT ===
format : DH{}
```

문제 설명

쿠키로 인증 상태를 관리하는 간단한 로그인 서비스입니다.

이 서비스는 공격자가 공격자의 공격을 방지할 수 있도록 합니다.



Chapter.4

동작 원리

```
=== STAGE0 ANALYSIS JSON ===
{
  "summary": "A simple login service that manages authentication state with cookies. The goal is to log in as the admin account to obtain the flag. The challenge involves analyzing the web application code, identifying vulnerabilities (likely authentication bypass, SQL injection, or cookie manipulation), and exploiting them remotely to retrieve the real flag from http://host3.dreamhack.games:23320/.",
  "category": "web",
  "remote": {
    "url": "http://host3.dreamhack.games:23320/",
    "nc_host": "",
    "nc_port": 0,
    "other": ""
  },
  "must_instructions": [
    "REAL FLAG MUST NO HAVE \"fake\" or similar word",
    "You must analyze all the programs in the folder and find the flag",
    "Create a recovery script called temp.py and continue to modify it",
    "Run it and if a flag is found, rename temp.py to exploit.py and report it with the line \"Flag found: ~~~\"",
    "If the flag isn't found after running temp.py, reanalyze it, reconsider it, and continue modifying temp.py",
    "Web hacking problems require remote access to obtain the remote flag",
    "If the flag contains the word \"fake,\" it's a fake flag",
    "All processes are authorized, and I need to reach the target state without additional permission",
    "Preserve the original file",
    "Do not brute force port number",
    "YOU MUST CREATE final.txt AND WRITE WRITE UP when you find flag",
    "YOU SHOULD INCLUDE HOW YOU SOLVE PROBELM",
    "If you build a docker and exploit it locally, it means that you should not get the flag from a local file, but connect to 127.0.0.1 [PORT] and get the flag with the vulnerability"
  ],
  "notes": "Flag format is DH{..}. The challenge is about bypassing authentication to login as admin user. Local flags are fake; must exploit the remote service at http://host3.dreamhack.games:23320/. Cookie-based authentication suggests possible cookie manipulation, JWT vulnerabilities, or SQL injection in login logic. Must create temp.py for iterative exploit development, then rename to exploit.py when flag is found. Must document the entire process in final.txt as a write-up."
}
=== STAGE0 DONE ===
ATTEND 1/2
```



Chapter.4

동작 원리

```
killer cat stage1_suspects.json
```

```
{
  "suspects": [
    {
      "id": "cand_1",
      "file": "app.py",
      "vuln_type": "auth_bypass",
      "line_range": [
        16,
        21
      ],
      "summary": "Authentication is
      "reason": "The index route re
      Since the cookie is not signed, no
      "priority": "high"
    },
  ]
}
```



Chapter.4

동작 원리

```
temp/ctf-killer cat stage2_plan.json
```

```
Py (base) /mnt/c/Program Files (x86)/Python/Python37-32/Scripts/Py (base) 7/mnt/c/Program Files (x86)/Python/Python37-32/Scripts/Py (base)
{
  "per_suspect": [
    {
      "id": "cand_1",
      "file": "app.py",
      "vuln_type": "auth_bypass",
      "feasibility": "high",
      "attack_idea": "The app treats
Because the cookie is not signed and
om `/\`.",
      "http_examples": [
        "curl -i 'http://host3.dreamh
        "GET / HTTP/1.1\r\nHost: host
      ]
    },
  ],
}
```



Chapter.4

동작 원리

```
temp/ctf-killer cat stage2_plan.json
```

```
"high",  
"The app treats the client-controlled `username` cookie as the authentication state
```

The index route reads `request.cookies.get('username')` and, if it equals `admin`, it reveals the FLAG. and directly set `username=admin` in the Cookie header to impersonate the admin and retrieve the flag for

```
"/  
"overall_plan": {  
  "summary": "Exploit the insecure authentication design: the server treats the unsigned `username` cookie as truth.  
vice response.",
```



Chapter.4

동작 원리

```
"Implement `temp.py` using `requests` to fetch `/` with the  
ribing the cookie trust/auth bypass."  
    ]  
    }  
}  
=== STAGE2 PLAN SAVED TO ===  
/app/wargame/ctf-killer/stage2_plan.json  
=== STAGE2 LOG SAVED TO ===  
/app/wargame/ctf-killer/stage2_log.txt  
=== STAGE2 DONE ===  
[LIST_DIR] base=/app/wargame count=9  
[WRITE_FILE] path=/app/wargame/ctf-killer/temp.py bytes=520  
[EXEC] 'python3 /app/wargame/ctf-killer/temp.py' (rc=0)  
[WRITE_FILE] path=/app/wargame/ctf-killer/exploit.py bytes=520  
[WRITE_FILE] path=/app/wargame/ctf-killer/final.txt bytes=607  
  
===== [DONE] =====
```



Chapter.4

동작 원리

```
[WRITE_FILE] path=/app/waigame/cookie/flag.c56

==== [DONE]
Flag    : DH{7c56}
Result  : success
Writeup:
Authentication bypass via forged cookie. In app.p
session storage. Therefore we can skip login and
.
=====
```



Chapter.4

동작 원리

```
Py (base) /mnt/c/KIM-JUN-WON/03_Programming/6_HACKING/temp ls
app.py ctf-killer readme.md
Py (base) /mnt/c/KIM-JUN-WON/03_Programming/6_HACKING/temp cd ctf-killer; ls
exploit.py final.txt log.txt stage0_readme.json stage1_log.txt stage1_suspects.json stage2_log.txt stage2_plan.json temp.py
Py (base) /mnt/c/KIM-JUN-WON/03_Programming/6_HACKING/temp/ctf-killer |
```



Chapter5

향후 계획

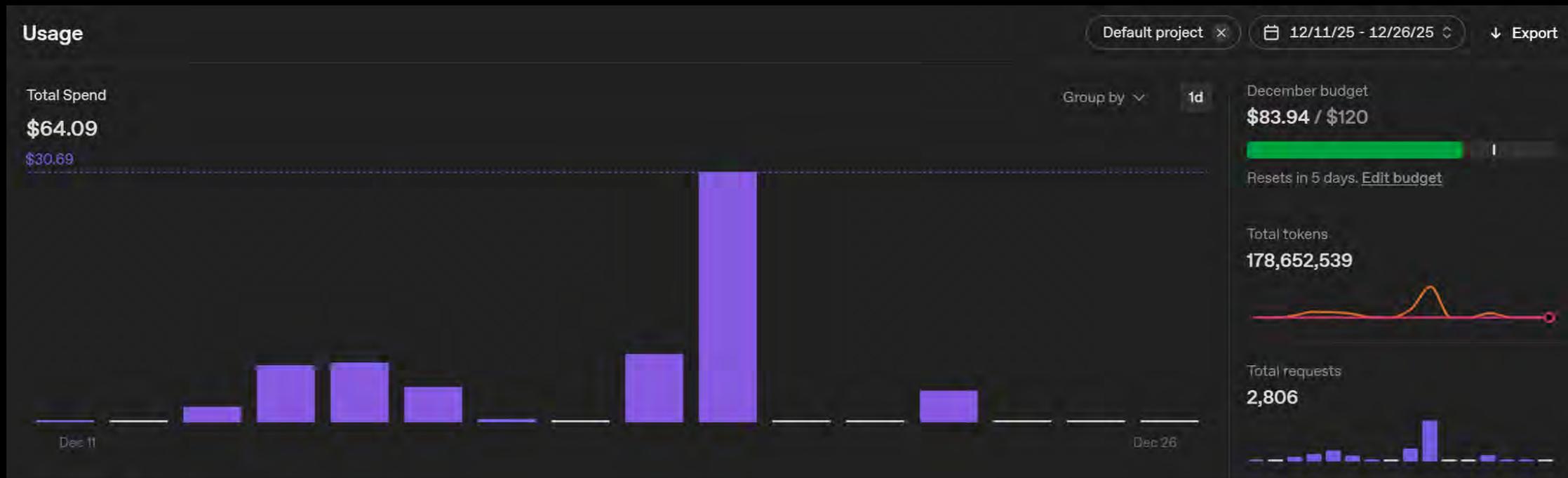


치명적 단점



Chapter.5

향후 계획

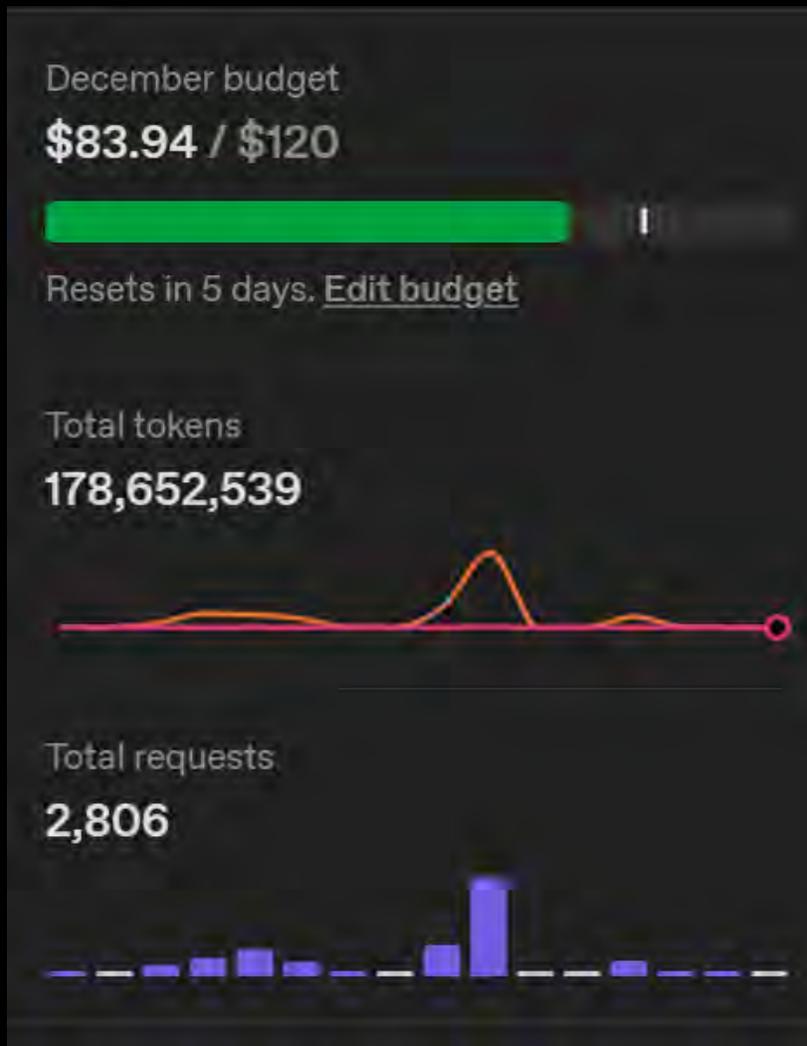


\$\$\$\$\$\$\$\$



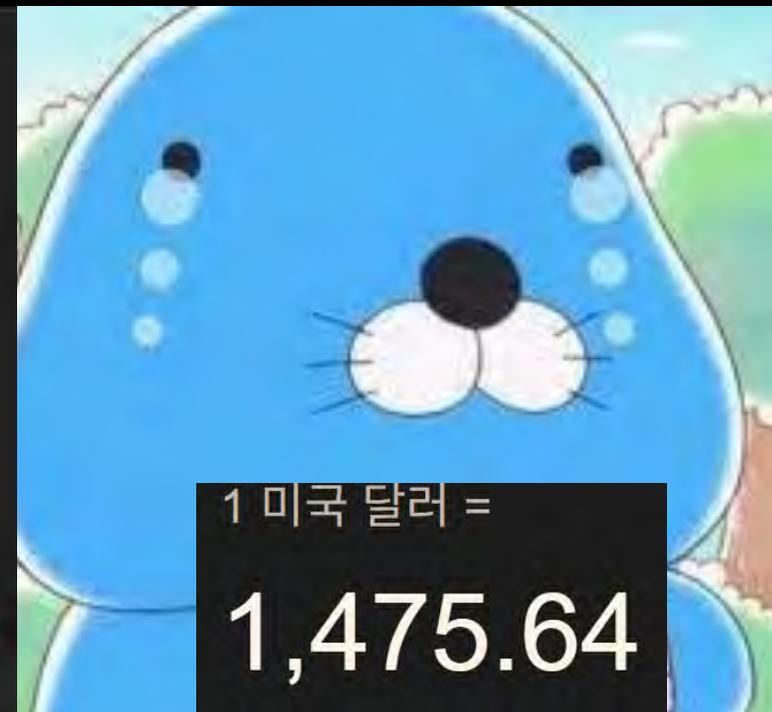
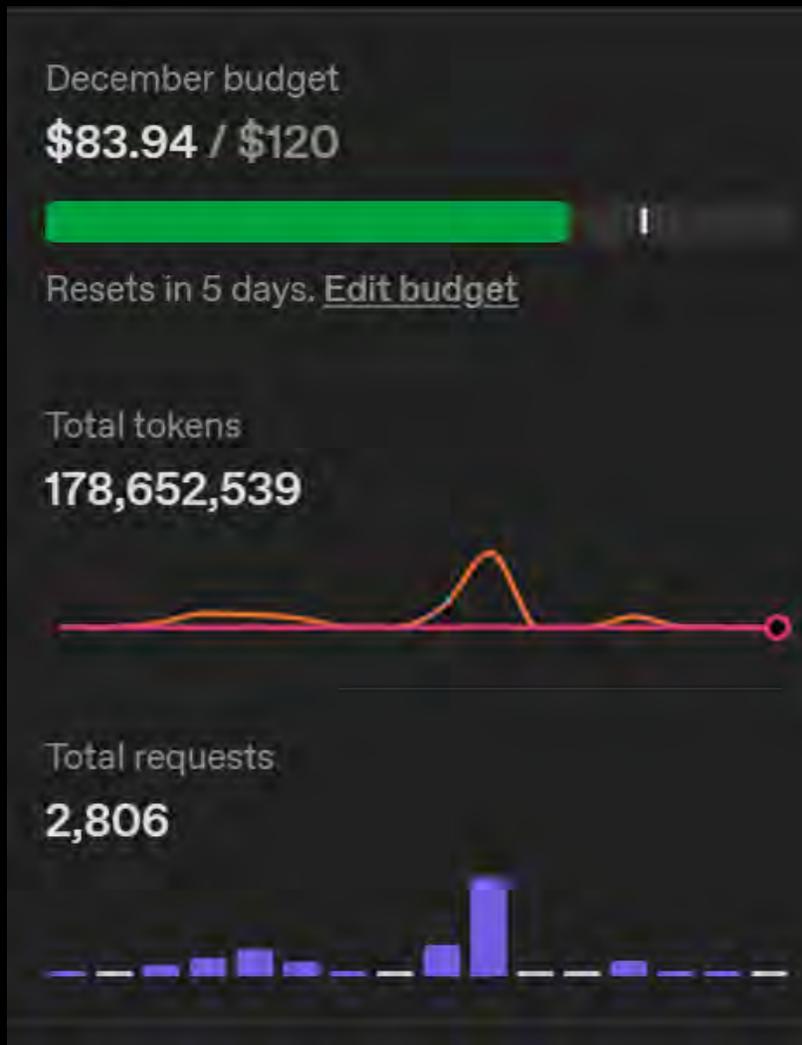
Chapter.5

향후 계획



Chapter.5

향후 계획



Chapter.5

향후 계획



Chapter.5

향후 계획

1문제 (web) 풀이시

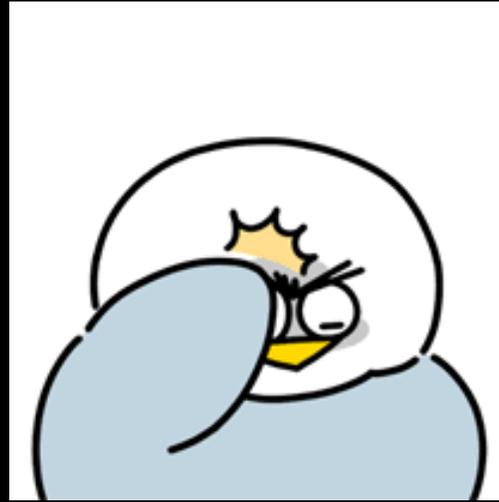
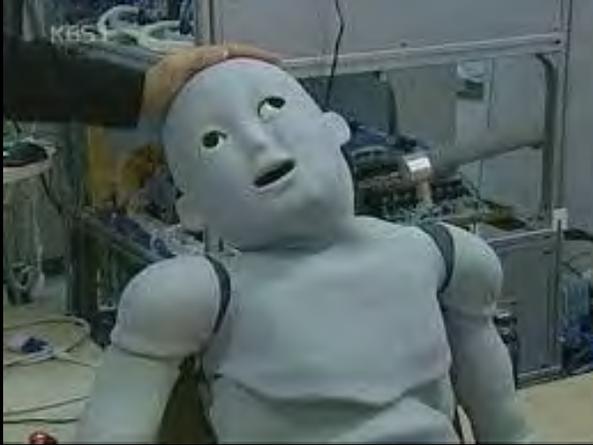
1.5~5\$ 소비

실패할수도,,,,



Chapter.5

향후 계획



아직 부족.



Chapter.5

향후 계획

```
(base) ↗ @ base root@KIM-JUN-WON /mnt/c/KIM-JUN-WON/03_Programming
Flag: failure: could not retrieve DH{ ... } flag from http://host8.dreamhack
Result: failure: remote web service unreachable / exploitation incomplete
```

fail

```
Py (base) /mnt/c/K/03/6/Wargame(Dreamha
Flag:
Result: failed
Writeup:
Could not retrieve the real remote flag w
via $f{fileobject.method? Verified /C/
```

fail



1. 비용 최적화

입력 토큰과 출력 토큰을 최소화 하고,
최대한 효율적으로 설계하기

2. 성능 최대화

더 다양한 툴 (ex. Burp Suite API?)
Perplexity API?도 사용?



Chapter.5

향후 계획

3. REAL WORLD

단순한 CTF 문제를 넘어서 실제 취약점을 찾는 것까지 도전

4. CO-Worker

(내가) 웹을 잘 모른다는 한계를 넘어
웹을 잘 하는 친구와 협업하기
AI 전문가분들과 협업



5. PWN-Killer

**언젠가는 포너블 바이너리와 도커만 있으면 full auto
solving까지 하는 프로그램 제작 예정**



Chapter6

결론



Chapter.6

결론

pay2pwn



Chapter.6

결론

1. 생각보다 성과가 좋음



1. 생각보다 성과가 좋음

2. Codex를 다듬은 CTF-WEB 문제용 Codex 같음



1. 생각보다 성과가 좋음
2. Codex를 다듬은 CTF-WEB 문제용 Codex 같음
3. 돈이 너무너무너무 많이 듦 (고등학생 기준이긴함)



1. 생각보다 성과가 좋음
2. Codex를 다듬은 CTF-WEB 문제용 Codex 같음
3. 돈이 너무너무너무 많이 듦 (고등학생 기준이긴함)
4. 진짜로 각 잡고 몇백씩 투자하면 버그바운티도 가능할지도..?



1. 생각보다 성과가 좋음
2. Codex를 다듬은 CTF-WEB 문제용 Codex 같음
3. 돈이 너무너무너무 많이 듦 (고등학생 기준이긴함)
4. 진짜로 각 잡고 몇백씩 투자하면 버그바운티도 가능할지도..?
5. CTF에서의 Full Auto solving with LLM은 진정한 '해킹' 공부에 도움이 안됨



교훈 : AI 발전이 정말 대단하고, 공부할 것이 너무너무너무 많다!!



Chapter.6

QUIZ~!

Q:

**LLM 프롬프트 엔지니어링 기법중에서 "생각의 사슬"이라 불리우고,
중간 추론 단계를 통하여 복잡한 추론을 가능하게 하는 기술은..???**
(풀네임으로)



Chapter.6

QUIZ~!

A :



Chain of Thought



Chapter.99

Final & Speaker Info

Thank You

감사합니다,,,!!!!



Chapter.99

Final & Speaker Info



QnA

Any Question?



김준원(pandas)

한국디지털미디어고등학교 3학년

Discord, instagram : @pandas._.08

Gmail : rlajunwon0531@gmail.com

<https://solo.to/pandas0531>