# 클라우드 컴퓨팅
# 공격 및 방어 플랫폼 개발


부산대학교
정보컴퓨터 공학부

지도교수 김 태 운

팀 명 DDALPI

팀 원 201513137 이강빈
202155599 장진영
202055505 강수민

# 목차


1 프로젝트 개요 ..... 3

1.1 프로젝트 개요 ..... 3

1.2 과제 목표 및 기대효과 ..... 4

2 YoYo Attack 방식 ..... 5

2.1 Network – Layer Attack ..... 5

2.2 Application – Layer Attack ..... 5

3 YoYo Attack Detect 방식 ..... 6

3.1 Attacker 측 Detect 방식 ..... 6

3.2 Defender (Target) 측 Detect 방식 ..... 6

4 프로젝트 소개 및 설계 ..... 7

4.1 개발 일정 / 기술 스택 ..... 7

4.2 시스템 동작 방식 설명 ..... 8

5 한계 및 제약 사항, 대응되는 대책 ..... 10

5.1 한계 및 제약 사항 ..... 10

5.2 대책 ..... 10

6 개발 일정 계획 및 담당 업무 ..... 11

6.1 개발 일정 ..... 11

6.2 담당 업무 ..... 12

7 참고 문헌 ..... 13

# 1. 프로젝트 개요

## 1.1 배경 및 필요성

과거보다 현재 IT 기기들의 전반적인 컴퓨팅 파워가 상승하면서 해커들이 공격을 위한 좀비 PC 자원을 쉽게 확보할 수 있게 되었다. [i] 이는 시중에서 활동하는 해커들이 DDoS 공격대행 의뢰 등을 통해 나타난다. [ii] 이러한 서비스를 통해 개인 방송인들과 특정 게임 리그의 실시간 서비스가 중단되는 일이 다수 발생하고 있다. [iii]

최근 클라우드 컴퓨팅 서비스 시장이 빠르게 확대되고 있으며, 많은 기업과 개인 사용자들이 이를 채택하고 있다. 클라우드 기반의 서비스는 접근성, 유연성, 비용 효율성 측면에서 뛰어나기 때문에 널리 사용된다. 특히 데이터 저장, 애플리케이션 개발, 인공지능 및 머신러닝 등의 분야에서 클라우드의 활용도가 높아지고 있다. 이러한 추세는 향후에도 지속될 것으로 예상되며, 클라우드는 IT 인프라의 핵심 요소로 자리 잡고 있다. [iv]

이에 클라우드 서비스의 애플리케이션 레이어를 공격하여 DDoS 처럼 서비스를 마비시키지 않고 Auto-Scaling 만을 유도한 뒤 상대 서비스에 금전적인 피해를 입히는 사례들이 발견되고 있다. 이를 EDoS(Economic Denial of Service) 또는 YoYo Attack 이라고 명명한다. (이후 YoYo Attack 으로 통칭한다)[v]

YoYo Attack 은 클라우드 서비스의 과금 구조를 악용하여 상대방의 비용을 증가시키는 새로운 형태의 공격 방식이다. 클라우드 서비스는 자원 사용량에 따라 과금되기 때문에, 공격자는 인위적으로 트래픽을 유발하여 과도한 비용을 청구하게 함으로써 경제적 타격을 준다.[vi]

## 1.2 과제 목표 및 기대효과

본 과제는 YoYo Attack 플랫폼과 Auto-Scaling 이 유발될 수 있는 클라우드 가상 환경의 방어 플랫폼을 구현한다.

첫 번째로, Raspberry Pi(이후 RPI 으로 통칭한다.) Cluster 로 공격 플랫폼을 구축하여 가상의 좀비 PC 역할을 수행하도록 한다. Web Application 을 통해 공격 방식을 제어하고, YoYo Attack 의 다양한 공격 시나리오를 설정한다. 또한, Auto-Scaling 을 유발할 수 있는 최적의 공격량을 탐색하여 효율적인 공격 전략을 수립한다. 이를 통해 공격 자원을 효과적으로 관리하고 제어한다.

두 번째로, Auto-Scaling 을 통해 서비스의 성능 유지하는 방어 플랫폼을 구축하고, YoYo Attack 을 감지하는 알고리즘을 개발한다. 이를 통해 정상적인 접속은 허용하되, YoYo Attack 에 대해서는 Auto-Scaling 이 일어나지 않도록 한다. 또한 실시간 공격을 탐지하고, 이에 대응할 수 있도록 한다.

이를 통해 YoYo Attack 으로부터 서비스의 가용성을 보장하고, 서비스 제공자의 과도한 비용 지출을 막음으로써 경제적 피해를 줄일 수 있다. 또한, 공격 플랫폼을 통해 해당 서비스의 전반적인 보안 수준을 점검할 수 있다.

# 2. YoYo Attack 방식

## 2.1 Network - Layer Attack

### 2.1.1 SYN Flood

- TCP 프로토콜의 결함을 이용해 위조 TCP 연결 요청을 대량으로 전송하여 피공격자의 자원을 소진(CPU 풀로드 또는 메모리 부족)시킨다.

### 2.1.2 Land Attack

- 공격자가 희생자의 IP 로 IP 스푸핑을 하고 수 많은 ICMP 패킷을 전송한다. 그리고 희생자가 자기 자신에 대해 응답하여 서비스 거부 상태를 만든다.

### 2.1.3 UDP Flood

- 공격자가 대량의 UDP 패킷을 타켓 시스템에 보내서 네트워크나 시스템 자원을 과부화 상태로 만들어 정상적인 서비스를 방해한다.

## 2.2 Application - Layer Attack

### 2.2.1 HTTP GET Flood

- 공격자가 타켓 Web Application 에 다량의 HTTP GET 요청을 전송하는 공격이다.
- Web Application 서버는 다량의 GET 요청을 처리하는 과정에서 CPU, Memory 등의 자원을 소모하게 되고, 과부하가 걸리게 된다.

### 2.2.2 HTTP POST Flood

- 공격자가 타켓 Web Application 에 다량의 HTTP POST 요청을 전송하는 공격 방식이다.
- Web Application 서버는 다량의 POST 요청을 처리하는 과정에서 CPU, Memory 등의 자원을 소모하게 되고 과부하가 걸리게 된다.
- GET 요청을 처리하는 과정보다 POST 요청을 처리하는 과정이 복잡할 수 있으며, 효율적인 공격 방식이 된다.

# 3. YoYo Attack Detect 방식

## 3.1 Attacker 측 Detect 방식

### 3.1.1 공격 중인 Web Server Request Time 확인

- curl 명령어 또는 HTTP GET Request 를 활용하여 공격중인 Web Application 에 응답 시간을 확인한다.
- 공격 시작 전보다 응답시간이 지연되었는지 확인한다.

## 3.2 Defender (Target) 측 Detect 방식

### 3.2.1 네트워크 트래픽 도구 활용

- 비정상적으로 높은 트래픽이 발생했는지 확인한다.
- 특정 IP 주소에 트래픽이 집중되어 있는지 확인한다.

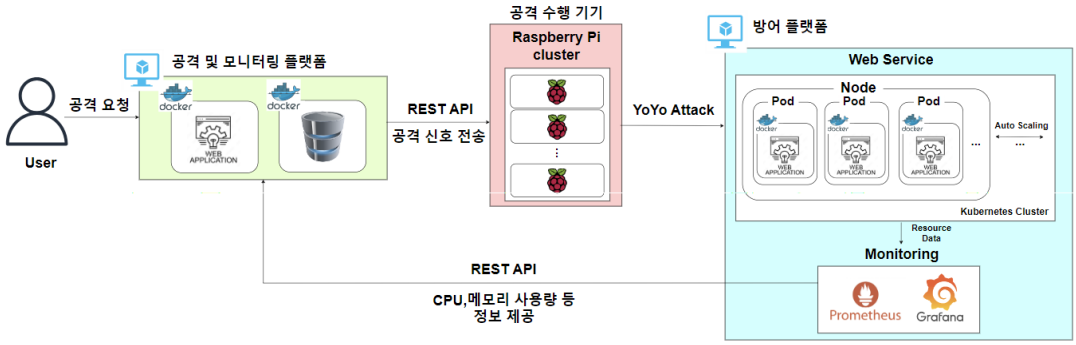
### 3.2.2 Server Resource 사용률 확인

- 다량의 요청을 처리하면 Server 의 CPU, Memory, 네트워크 대역폭 등의 사용률이 급증한다.
- 클라우드 시스템에서 CPU 의 사용량이 늘어나면 Auto-Scaling 을 진행한다.
- Server Resource 사용률을 지속적으로 모니터링 하고, 갑작스러운 증가가 있는지 확인한다.

# 4. 프로젝트 소개 및 설계

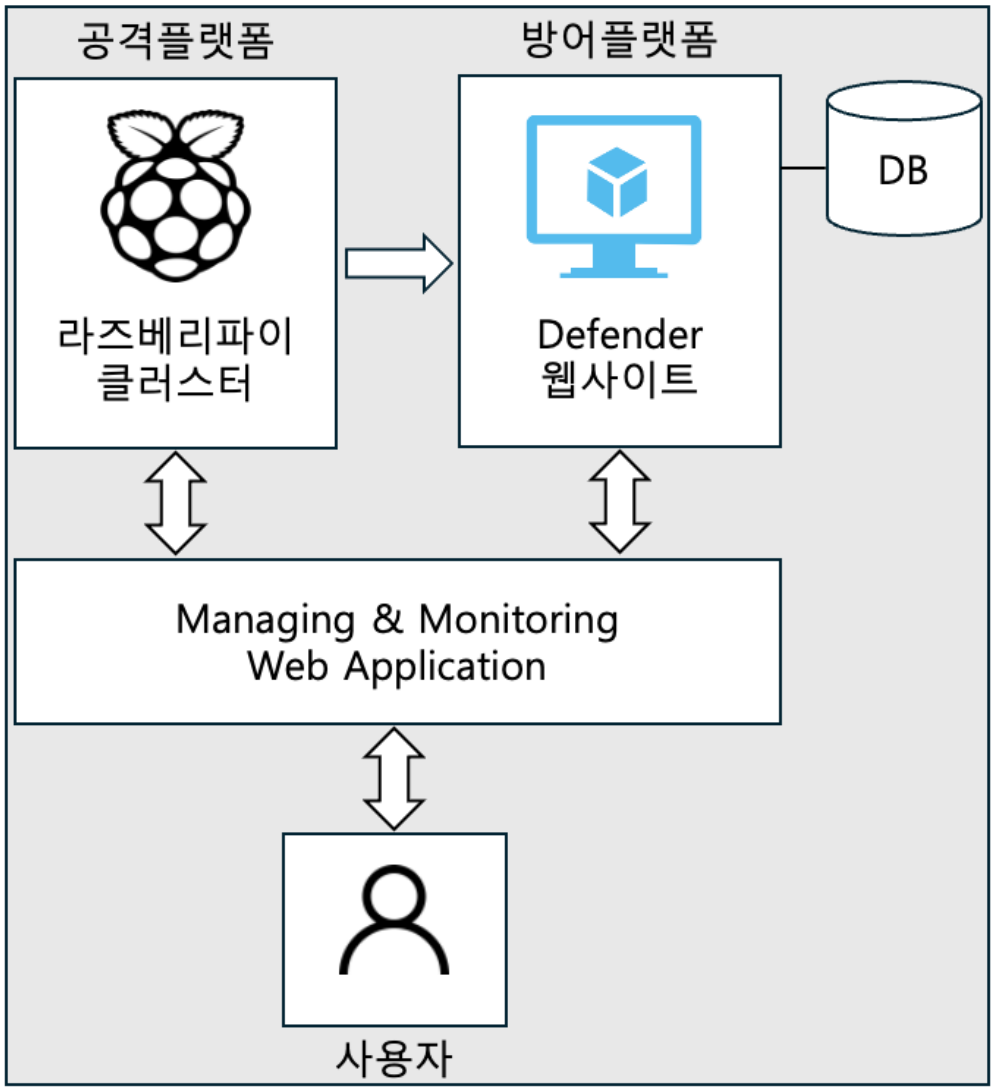
## 4.1 개발 환경 / 기술 스택

### 4.1.1 전체 구성도




| 요소 | 내용 |
|---|---|
| UI | HTML, CSS, Java Script 를 사용하여 공격 및 모니터링 화면, 방어 서비스 화면을 만든다. |
| 백엔드 서버 | YoYo Attack 수행 명령 및 방어 서비스 모니터링 등 사용자의 요청을 처리한다. |
| RPI Cluster | 방어 플랫폼에 YoYo Attack 을 수행한다. |
| 클라우드 서비스 플랫폼 | 공격 및 모니터링 플랫폼과 방어 플랫폼의 서버를 구축한다. |
| 데이터베이스 | 실제 서비스에서 발생하는 부하를 재현하기 위해서, 방어 플랫폼에서 사용한다. |
| Docker | 컨테이너 이미지를 기반으로 웹 애플리케이션을 다양한 환경에 배포한다. |
| Kubernetes | 컨테이너화 된 애플리케이션을 리소스 사용량에 따라 Auto-Scaling 한다. |
| Prometheus, Grafana | 시스템 모니터링 도구로 웹 애플리케이션의 상태 및 성능 지표 수집 및 시각화를 수행한다. |

## 4.2 시스템 동작 방식 설명




1. 사용자는 Web Application 을 이용해 동작을 선택할 수 있다.

- YoYo Attack Managing:

  YoYo Attack 을 시작 또는 중단할 수 있다.

- YoYo Attack Attacker Monitoring :

  사용자는 현재 공격이 어떻게 이루어지는지를 모니터링 할 수 있다.

- YoYo Attack Defender Monitoring :

  사용자는 현재 방어가 어떻게 이루어지는지를 모니터링 할 수 있다.

2. 공격 플랫폼

- Defender 웹사이트에 Network-Layer Attack, Application-Layer Attack 을 진행한다.

- 자신의 공격이 유효하게 진행되는지를 지속적으로 파악하고, 공격이 유효하지 않다면 공격 방식을 바꿔서 공격을 지속한다.

- YoYo Attack 을 통해 Scaling Out 이 일어나면 공격을 중단한다.

3. 방어 플랫폼

- 지속적으로 비정상적인 트래픽을 감지한다.

- 비정상적인 트래픽이 감지되면, YoYo Attack 을 방지하기 위해 내부적으로 여러 방어 알고리즘을 이용하여 공격을 피한다.

# 5. 한계 및 제약 사항, 대응되는 대책

## 5.1 한계 및 제약 사항

1. 공격 측에서 YoYo Attack 이 잘 이루어졌는지 탐지하는 방법이 제한적이다.

2. YoYo Attack 을 발생시키는 공격방식이 제한적이다.

3. Web Application 기능에 따라 Server 리소스 사용량이 달라진다.

4. Local 환경에서 무선 통신을 이용하면 다수의 PC 를 활용해도 한 번에 많은 패킷을 전달하기 힘들다.

5. Local 환경에서 다수의 PC 가 한 번에 많은 패킷을 전달해도 하나의 PC 가 한 번에 받아들이는 패킷량이 한정적이다.

6. Local PC 의 성능(CPU Clock, Memory)이 뛰어나 공격의 임계점이 높다.

7. 클라우드 시스템을 사용하여 공격 및 방어 플랫폼 서버를 운영할 때 비용이 많이 든다.

8. 실제 사용 서비스에 YoYo Attack 시도가 불가능하다.

## 5.2 대책

1. 공격 로그 데이터를 활용하여 머신 러닝 기법을 사용할 수 있다.

2, 3. 웹 퍼포먼스 테스트 툴을 활용하여 가장 공격 효율이 높은 요청을 찾아내고, 집중적으로 YoYo Attack 을 진행한다.

4. Local test 환경을 스위치를 이용해 NIC 를 유선으로 연결한다.

5. Defender 입장의 machine 의 NIC 대역폭이 크도록 준비한다.

6. PC 의 Resource 을 많이 사용하는 (ex, I/O Interrupt) Web Application 구성한다.

7, 8. Local 환경에서 최대한 실제 서비스 환경과 비슷한 환경을 구현하여 테스트하고 데모 시에만 클라우드를 사용 한다.

# 6. 개발 일정 계획 및 담당 업무

## 6.1 개발 일정

| 구분 | 작업일정 | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 5 월 | | | 6 월 | | | | 7 월 | | | | 8 월 | | | | | 9 월 | | | | 10 월 | | |
| | 3 | 4 | 5 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 5 | 1 | 2 | 3 | 4 | 1 | 2 | 3 |
| YoYo attack 스터디 | | | | | | | | | | | | | | | | | | | | | | | |
| VM 환경 YoYo attack 실습 | | | | | | | | | | | | | | | | | | | | | | | |
| VM 실습 환경 구축 | | | | | | | | | | | | | | | | | | | | | | | |
| 공격 알고리즘 구현 | | | | | | | | | | | | | | | | | | | | | | | |
| 방어 알고리즘 구현 | | | | | | | | | | | | | | | | | | | | | | | |
| 중간 보고서 작성 | | | | | | | | | | | | | | | | | | | | | | | |
| 실 서비스 환경 구축 | | | | | | | | | | | | | | | | | | | | | | | |
| UI/UX 개발 | | | | | | | | | | | | | | | | | | | | | | | |
| 모니터링 시스템 개발 | | | | | | | | | | | | | | | | | | | | | | | |
| 최종 시스템 구축 | | | | | | | | | | | | | | | | | | | | | | | |
| 디버깅 및 테스트 | | | | | | | | | | | | | | | | | | | | | | | |
| 배포 테스트 및 배포 | | | | | | | | | | | | | | | | | | | | | | | |
| 최종 보고서 | | | | | | | | | | | | | | | | | | | | | | | |

## 6.2 담당 업무

| 이름 | 분류 | 세부 역할 분담 |
|---|---|---|
| 이강빈 | Full Stack 개발 | - Management & Monitoring Web Application 개발<br>- Attacker 플랫폼 개발<br>- Defender 플랫폼 개발 |
| 장진영 | 알고리즘 개발 | - 공격 알고리즘 개발<br>- Application – Layer<br>- Network – Layer<br>- 공격 유효 확인 알고리즘<br>- 방어 알고리즘 개발<br>- YoYo Attack 감지 알고리즘 개발<br>- 공격 방어 알고리즘 개발 |
| 강수민 | 인프라 | - 구축<br>- Docker 설치<br>- Kubernetes 구성<br>- Prometheus, Grafana 구성<br>- RPI Cluster 구성 |
| 공통 | 자료 조사 | - YoYo Attack 공격 방식, 공격 탐지 방법 조사 |
| | 보고서 작성 | - 중간 보고서, 최종 보고서 |
| | 발표 및 시연 | - 발표자, 시연자, Q&A 담당자 |

# 7. 참고 문헌


[i] 보안뉴스, 《2021~2023 년, 디도스 최대 공격량 매년 100% 이상 폭발적 증가해》, 김영명 기자,2024.01.29 ,URL :https://www.boannews.com/media/view.asp?idx=126201&direct=mobile

[ii] 보안뉴스, 《 FBI, 디도스 공격 대행 서비스 13 곳 폐쇄시켜 》, 문가용 기자,2023.05.09 ,URL :https://www.boannews.com/media/view.asp?idx=117885&direct=mobile

[iii] YTN, 《 e 스포츠 덮친 '디도스 공격'...게임업계 시름 》, 김혜린 기자,2024.04.13 ,URL :https://www.ytn.co.kr/_ln/0103_202404130024114804

[iv] CIO korea, 《 1 분기 클라우드 시장 20% 성장… 빅 3 가 65% 점유 》, 김달훈 기자,2023.05.08 ,URL :https://www.ciokorea.com/news/289753

[v] Bremler-Barr, A., Brosh, E. and Sides, M. (2017). DDoS attack on cloud auto-scaling mechanisms. [online] IEEE Xplore. doi:https://doi.org/10.1109/INFOCOM.2017.8057010.

[vi] Ben David, R. and Barr, A. (2021). Kubernetes Autoscaling: YoYo Attack Vulnerability and Mitigation. Proceedings of the 11th International Conference on Cloud Computing and Services Science. doi:https://doi.org/10.5220/0010397900340044.