ARES: 서버리스 환경에서의 에이전틱 AI 기반 실시간 권한 우회 방지 시스템

신창희[1], 이승수[2]

*인천대학교(대학원생[1], 교수[2])

ARES: An Agentic AI Based Real-Time Privilege Bypass Prevention System in Serverless Environments

Changhee Shin[1], Seungsoo Lee[2]

*Incheon National University(Graduate student[1], Professor[2])

요 약

서버리스 컴퓨팅은 주요 클라우드 플랫폼에서 널리 채택되고 있으며, 각 함수의 자원 접근은 IAM 정책으로 제어된다. 그러나 서버리스 IAM은 최초 요청자의 권한을 실행 경로에 전파하지 않아, 최소 권한이 부여되더라도 함수 호출 경로를 통한 권한 우회가 발생할 수 있다. 본 논문에서는 에이전틱 AI 기반으로 이러한 우회를 실시간 탐지 및 차단하는 시스템 ARES를 제안한다. ARES는 클라우드 플랫폼에서 수집한 권한 정보를 관계 구조로 프로파일링하고, LAM이 호출 시점에 자율적으로 우회 여부를 추론하여 인라인 차단을 수행한다. 4개 주요 클라우드 플랫폼에서의 실험을 통해 모든 호출 유형에서 우회가 성공적으로 차단됨을 확인하였다.

I. Introduction

서버리스 컴퓨팅은 이벤트 기반의 함수 단위 실행 모델로 자동 확장과 운영 부담 절감을 제공하며, AWS Lambda, Google Cloud Functions, Azure Functions, Alibaba Function Compute 등 주요 클라우드 플랫폼에서 널리 채택되고 있다. 특히 최근 AI 모델 추론 워크로드로의 확장과 함께 서버리스 시장은 2025년 280억 달러에서 2034년 922억 달러 규모로 성장이 전망되며[1], 활용 범위가 빠르게 확대되고 있다.

서버리스 환경에서 각 함수의 자원 접근은 IAM 정책으로 제어되며, 함수별 최소 권한 부여가 권장된다. 그러나 최소 권한이 부여되더라도, 서버리스 IAM은 각 호출을 독립적으로 평가하며 최초 요청자의 권한을 실행 경로에 전파하지 않기 때문에, 함수 호출 경로를 통한 권한 우회가 발생할 수 있다. 기존 연구는 주로 최소 권한 추출 및 부여에 초점을 두고 있으나, 우회 위협에 대한 실시간 대응은 다루어지지 않고 있다[2, 3].

따라서 본 연구에서는 에이전틱 AI 기반으로 서버리스 환경의 권한 우회를 실시간으로 탐지 및 차단하는 시스템인 ARES를 제안한다. ARES는 클라우드 플랫폼에서 수집한 권한 정보로 우회 가능 경로를 사전에 프로파일링하고, 함수 호출 시점에 대규모 행동 모델(Large Action Model, LAM)이 자율적으로 우회 여부를 추론하여 인라인 차단을 수행한다.

본 논문의 기여 사항은 다음과 같다:

- 서버리스 IAM의 호출자 비인지(caller-unaware) 아키텍처에서 발생하는 권한 우회 위협을 분석하고 제시하였다.
- 에이전틱 AI 기반으로 우회 경로를 자율적으로 추론하고, 함수 실행 전에 인라인 차단을 수행하는 시스템 ARES를 구현하였다.
- 주요 클라우드 플랫폼의 서버리스 애플리케이션을 대상으로 실험을 수행하여, ARES의 우회 차단 정확도와 런타임 오버헤드의 효과성을 검증하였다.

II. Background and Motivation

2.1 Serverless IAM Mechanism

서버리스 컴퓨팅 환경에서 클라우드 리소스에 대한 접근은 IAM(Identity and Access Management) 정책을 통해 제어된다. IAM 정책은 기본적으로 허용 목록(allowlist) 방식으로 동작하며, 명시적으로 허용되지 않은 접근은 모두 거부된다. 정책의 핵심 요소는 Effect(허용/거부), Action(수행 가능한 작업), Resource(접근 대상), Principal(요청 주체), Condition(추가 조건)으로 구성된다.

서버리스 환경에서 IAM은 두 단계에서 적용된다. 첫째, 수신 정책(incoming policy)은 해당 함수를 누가 호출할 수 있는가를 정의한다. 이 정책은 허용된 주체(Principal)를 명시하여, 특정 사용자, 다른 함수, 또는 이벤트 소스만이 해당 함수를 트리거할 수 있도록 제한한다. 둘째, 실행 정책(outgoing policy)은 해당 함수가 어떤 자원에 어떤 연산을 수행할 수 있는가를 정의한다. 함수는 플랫폼이 부여한 서비스 자격 증명(service identity)으로 실행되며, 이 정책에 명시된 Action(연산)과 Resource(대상 자원)의 범위 내에서만 클라우드 자원에 접근할 수 있다.

2.2 에이전틱 AI

에이전틱 AI는 선언적 목표를 자율적으로 분해하고, 필요한 정보를 수집하며, 적절한 도구를 활용하여 행동을 계획하고 실행하는 인지 시스템이다. 이러한 시스템은 독립적 실행 단위인 에이전트(agent)로 구성되며, 인식(perception), 추론(reasoning), 행동(action)의 연속적인 피드백 루프를 통해 환경 변화에 적응한다.

Agentic AI의 핵심 구성 요소로 대규모 행동 모델이 있다. 기존 LLM이 텍스트 생성에 초점을 둔다면, LAM은 외부 도구의 선택, 호출, 그리고 결과에 따른 후속 행동 결정까지를 모델의 출력 공간에 포함한다. 이를 통해 LAM은 상태 관찰, 계획 수립, 도구 호출, 결과 반영의 인지 루프를 자율적으로 반복하며, 추론의 순서와 깊이는 각 단계의 결과에 따라 동적으로 결정된다. 결과는 지식 베이스에 기록되어 이후 의사결정을 지원하며, 이를 통해 사전 정의된 규칙 없이 환경 변화에 대응할 수 있다.

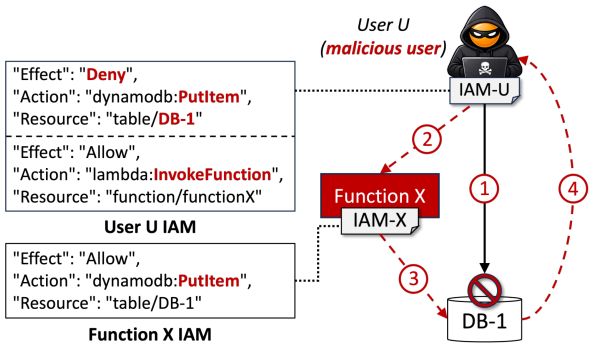

Fig. 1. Motivating example of permission bypass via function invocation under a caller-unaware IAM architecture.

2.3 Motivation

서버리스 환경에서는 함수별로 최소 권한이 올바르게 부여된 경우에도 권한 우회가 발생할 수 있다. 이는 서버리스 플랫폼의 IAM이 각 호출을 독립적으로 평가하며, 호출자의 권한만 검증하고 최초 요청자의 권한은 전파하지 않는 호출자 비인지(caller-unaware) 아키텍처에 기인한다. 이러한 구조에서는 개별 함수의 IAM 정책이 정밀하게 설정되더라도, 함수 간 호출 관계가 형성되면 최초 요청자의 권한 경계가 소멸한다.

Fig. 1.은 이러한 우회 시나리오를 보여준다. 악의적 사용자 U의 IAM 정책(IAM-U)은 DB-1에 대한 dynamodb:PutItem 연산을 거부(Deny)하는 동시에, Function X에 대한 lambda:InvokeFunction을 허용한다. 한편, Function X의 IAM 정책(IAM-X)은 DB-1에 대한 dynamodb:PutItem을 허용한다. 이 구성에서 사용자 U는 DB-1에 직접 접근할 수 없지만(①), 호출 권한을 이용하여 Function X를 트리거하고(②), Function X는 자신의 IAM 권한으로 DB-1에 쓰기를 수행한다(③). 그 결과가 사용자 U에게 반환되어(④), 거부된 자원에 간접 접근하게 된다. 이 과정에서 각 단계의 IAM 정책은 개별적으로 최소 권한을 준수하고 있으나, 플랫폼이 최초 요청자의 권한을 실행 경로에 전파하지 않기 때문에 우회가 성립한다. 나아가 이러한 우회는 이벤트 기반 호출을 통한 다중 홉 경로에서도 발생할 수 있으며, 이 경우 IAM 검증 자체가 적용되지 않아 최초 요청자와 완전히 분리된 실행 경로가 형성된다.

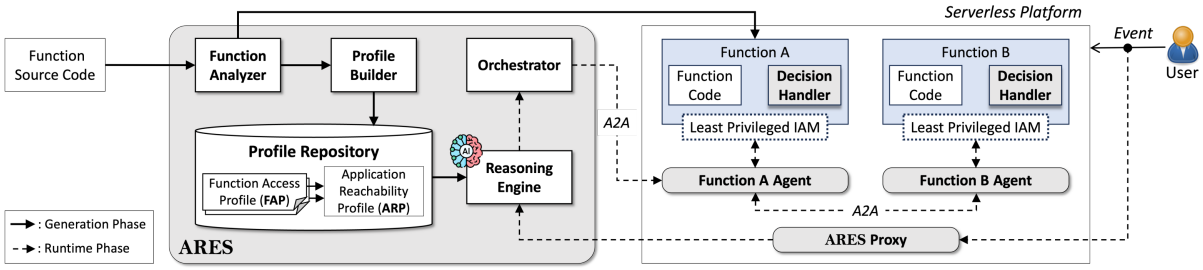

Fig. 2. Overall architecture of ARES with six key components and distributed agents.

III. System Design

3.1 ARES Overview

Fig. 2.는 ARES의 전체 구조를 나타낸다. ARES는 생성 단계(Generation Phase)와 런타임 단계(Runtime Phase)로 동작한다. 생성 단계에서는 Function Analyzer가 소스 코드의 언어와 벤더를 식별하고 에이전트 통신 코드를 삽입한 뒤, P-rofile Builder가 클라우드 API를 통해 수집한 권한 정보를 바탕으로 우회 가능성을 분석하여 Pro-file Repository에 저장한다. 런타임 단계에서는 ARES Proxy가 함수 호출을 수신하고, Reasoning Engine이 저장된 프로파일을 참조하여 우회 여부를 판단한다. 판단 결과는 Orchestrator를 통해 에이전트에 전달되며, 연쇄 호출 시에는 A2A(Agent-to-Agent) 프로토콜로 다음 에이전트에 전파된다.

3.2 Reachability Profile Generation

생성 단계에서 ARES는 클라우드 플랫폼 API를 통해 사용자의 IAM 정책에서 허용 및 거부된 자원 목록, 각 함수의 실행 역할(execution role)에 부여된 Action과 Resource, 그리고 함수의 수신 정책에 명시된 호출 가능 주체(principal) 정보를 수집한다. 수집된 정보는 ARP(Application Reach-ability Profile)라는 관계 구조로 재구성되어 프로파일 저장소에 유지된다.

ARP는 세 가지 관계 집합으로 구성된다. 첫째, User Invoke 집합은 사용자 u가 호출 가능한 함수 f의 쌍 (u, f)을 정의한다. 둘째, Service Reach 집합은 함수와 자원 간의 도달 가능성을 나타낸다. 함수 f의 principal 필드로부터 “누가 f를 호출하는가”를, resource 필드로부터 “f가 어떤 자원에 접근하는가”를 추출하여 연결한다. 셋째, User D-eny 집합은 사용자 u가 직접 접근이 금지된 자원 r의 쌍 (u, r)을 정의한다. 이 세 집합을 통해 우회 조건이 정의된다. 사용자 u가 함수 f를 호출하여 자원 r에 도달하는 경로에서, (u, r)이 User Deny에, (u, f)가 User Invoke에 속하고, f에서 r까지의 경로가 Service Reach의 전이적 폐포에 포함될 때, 해당 호출은 권한 우회로 정의된다.

3.3 Agentic AI Based Real-Time Enforcement

런타임 단계에서 함수 호출이 ARES 프록시를 통해 수신되면, 추론 엔진의 LAM은 호출 이벤트만을 입력으로 받아 추론을 개시한다. LAM은 MCP(Model Context Protocol)를 통해 ARP를 조회하며, 각 단계에서 어떤 쿼리를 실행할지를 스스로 결정한다. 예를 들어, 먼저 User Invoke 집합을 조회하여 해당 사용자의 호출 권한을 확인하고, User Deny 집합을 통해 접근 제한 자원을 파악한 뒤, Service Reach 집합을 반복 조회하여 다중 홉 경로를 탐색한다. 이 과정에서 쿼리의 순서와 깊이는 사전에 정해진 것이 아니라, 각 단계의 결과에 따라 동적으로 결정된다. 추론은 우회 경로가 확인되거나 더 이상 확장 가능한 경로가 없을 때 종료된다.

우회가 탐지되면, LAM은 차단 지점을 특정한다. 차단 결정은 오케스트레이터를 통해 에이전트에 전달되며, 에이전트는 세션 ID, 호출자 정보를 로컬 차단 테이블에 저장한다. 이후 함수 실행 시 Decision Handler가 호출자 정보를 차단 테이블과 대조하여, 일치 시 실행을 중단하고 불일치 시 정상 실행을 허용하며 호출자 정보를 다음 서비스에 전파한다. 동일한 우회 패턴이 반복될 때는 캐싱된 결정으로 즉시 처리된다. 연쇄적으로 호출되는 함수 구조에서는, 최초 에이전트가 A2A 프로토콜을 통해 다음 함수의 에이전트에 호출자 컨텍스트와 차단 결정을 전파하여 다중 홉 경로 전체에서 일관된 집행을 수행한다.

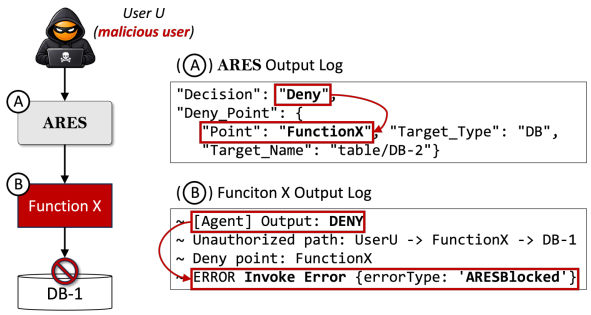

Fig. 3. ARES bypass detection and blocking.

IV. Evaluation

Implementation ARES는 Python으로 구현하였으며, 추론 엔진은 vLLM 기반 서버로 운영된다. 각 플랫폼 SDK에 호출 컨텍스트를 전파하는 패치 모듈을 구현하여, 에이전트 연동 코드를 자동 삽입하도록 하였다. 추론에는 Llama-3.1-8B 모델을 사용하였으며, ARP 조회를 위한 MCP 서버를 구현하여 LAM이 프로파일 저장소에 자율적으로 접근할 수 있도록 하였다.

Use Case 본 실험에서는 4개 플랫폼(AWS Lambda, Google Cloud Functions, Azure Functions, Alibaba Function Compute)에 최소 권한 구성의 함수를 배포하고 우회를 시도하였다. Fig. 3.은 Fig. 1.의 시나리오에 대한 차단 결과를 보여준다. 사용자 U가 Function X를 호출하면, ARES의 추론 엔진은 해당 호출이 DB-1에 대한 우회 경로임을 탐지하고 Deny 결정과 함께 차단 지점을 Function X로 특정한다(A). 이 결정을 수신한 Function X의 에이전트는 해당 요청을 ARESBlocked 오류와 함께 차단한다(B). 4개 플랫폼 모두에서 우회가 성공적으로 차단되었다.

Overhead 런타임 오버헤드는 각 벤더에서 공개한 오픈소스 서버리스 애플리케이션(AWS: Hello, Retail!, Google Cloud: Photosharing Workshop, Azure: Customer Car Reviews, Alibaba: E-commerce)을 대상으로 측정하였다. Fig. 4.는 ARES 적용 전후의 벤더별 평균 warm start 실행 시간을 나타낸다. ARES 적용 시 추가되는 오버헤드는 AWS 약 2.2ms, Google Cloud 약 1.3ms, Azure 약 1.5ms, Alibaba 약 4.0ms로, 전체적으로 5ms 이내의 미미한 수준으로, 실제 서비스 운영에 영향을 주지 않는 수준이다.

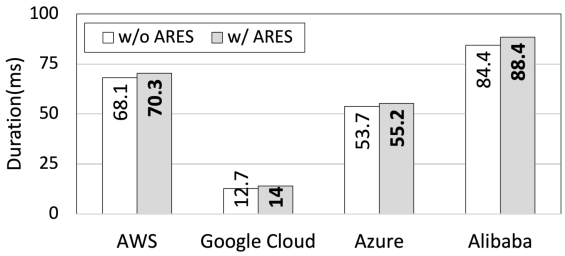

Fig. 4. Average warm start execution time per cloud vendor with and without ARES.

V. Conclusion

본 논문에서는 서버리스 환경의 caller-unaware IAM 아키텍처에서 발생하는 권한 우회 위협에 대응하기 위해 에이전틱 AI 기반 실시간 차단 시스템 ARES를 제안하였다. ARES는 클라우드 플랫폼에서 수집한 권한 정보를 관계 구조로 프로파일링하고, LAM이 함수 호출 시점에 자율적으로 우회 여부를 추론하여 인라인 차단을 수행한다. 4개 주요 클라우드 플랫폼에서의 실험을 통해 다양한 호출 유형의 우회 시도를 효과적으로 차단할 수 있음을 확인하며, 서버리스 환경에서의 실시간 권한 우회 방지에 대한 실효성을 입증하였다.

Acknowledgements

이 논문은 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원-학·석사연계ICT핵심인재양성 지원을 받아 수행된 연구임(IITP-2026-RS-2024-00437024)

[참고문헌]

[1] Precedence Research, "Serverless Computing Market Size, Share and Trends 2026 to 2034," https://www.precedenceresearch.com/serverless-computing-market, 2026.

[2] Gupta, Praveen, et al. "Growlithe: A Developer-Centric Compliance Tool for Serverless Applications." 2025 IEEE Symposium on Security and Privacy (SP). IEEE, 2025.

[3] Sankaran, Arnav, Pubali Datta, and Adam Bates. "Workflow integration alleviates identity and access management in serverless computing." Proceedings of the 36th Annual Computer Security Applications Conference. 2020.