PyTorch Pickle 모델 로딩 안정성 확보를 위한 In-band와 Out-of-band 검증 비교

이선우[1], 장한재[2], 김수하[1], 박민서[1], 최우현[1], 윤승현[1]

[1]한국에너지공과대학교, [2]전남과학고등학교

{sunwoolee, water03, westpark, woohyunchoi, syoon}@kentech.ac.kr

jshs251317@h.jne.go.kr

Comparing In-Band and Out-of-Band Verification for Secure PyTorch Checkpoint Loading

Sunwoo Lee[1], Han Jae Jang[2], Suha Kim[1], Minseo Park[1], Woo-Hyun Choi[1], and Seunghyun Yoon[1]

[1]Korea Institute of Energy Technology (KENTECH)

[2]Jeonnam Science High School

요 약

AI 모델이 Hugging Face 등 외부 허브를 통해 파일로 유통되면서, PyTorch의 torch.load()는 공급망 공격의 주요 표면이 되고 있다. torch.load()는 pickle 기반 역직렬화를 수행하므로, 신뢰되지 않은 모델 파일 로딩만으로도 임의 코드 실행이 유발될 수 있다. 본 논문은 검증이 모델 내부에 위치하는 in-band와 모델 외부 로더에 위치하는 out-of-band가 형성하는 검증 경계를 비교한다. In-band는 자기 포함적 배포가 가능하지만 검증 로직이 역직렬화 흐름에 편입되므로 실행 범위 최소화, 외부 의존성 제거, 예외 포함 fail-closed가 필수 조건이다. Out-of-band는 포맷 변경 없이 정책 적용이 가능하나 로더·API 변조 가능성 때문에 검증의 핵심이 경로 강제로 이동한다. 이에 본 논문은 out-of-band에서 검증 완료 전 로딩 금지, 비위조·비재사용 허용 증표, 검증 수행과 로딩 집행의 분리를 강제 검증 경로의 설계 원칙으로 제시한다.

Ⅰ. 서 론

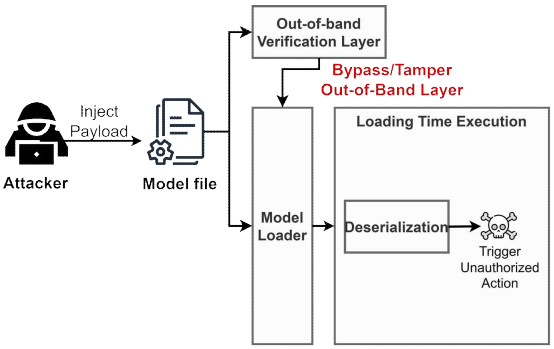
최근 AI 모델은 Hugging Face와 같은 외부 모델 허브를 통해 파일 형태로 배포·공유되는 공급망 환경이 확산되었고, PyTorch에서는 체크포인트를 *torch.load()*로 로딩하는 방식이 널리 사용된다. 그러나 *torch.load()*는 Python의 pickle 프로토콜에 기반한 역직렬화를 수행하므로, 신뢰되지 않은 출처에서 제공된 모델 파일을 로딩하는 행위 자체가 임의 코드 실행으로 연결될 수 있다[3]. 이러한 위험은 PyTorch의 로딩 경로와 연관된 CVE-2025-32434로도 확인된다[1]. 또한 모델 공급망에서 정적 분석 도구가 활용되지만, 파일 내부의 정적 표현 및 아카이브 처리에 의존하는 방식은 압축 컨테이너 조작 등으로 우회될 수 있으며, 실제로 picklescan의 탐지 우회 취약점 CVE-2025-1945이 보고된 바 있다[2]. 이와 같이 모델 파일은 데이터라는 전제가 성립하지 않는 환경에서, 역직렬화 단계의 실행 안전성을 보장하기 위한 검증 경계의 설정은 핵심 연구 과제가 된다.

본 논문은 PyTorch pickle 역직렬화 기반 공격을 대상으로, 모델 내부에 검증을 포함시키는 in-band 접근과 모델 외부 로더에서 검증을 수행하는 out-of-band 접근이 형성하는 보안 경계를 비교하고, 특히 out-of-band에서 결정적으로 발생하는 검증 주체 변조 문제를 중심으로 공격자가 임의로 수정할 수 없는 강제 검증 경로를 확보하기 위한 설계 원칙을 제시한다.

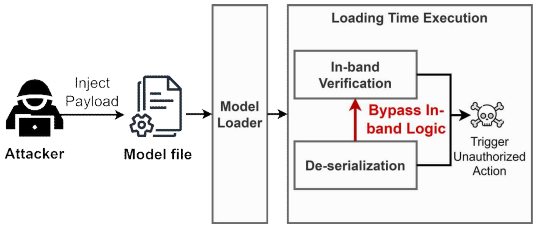
Ⅱ. 위협 모델

본 논문은 공격자가 모델 파일을 로딩하게 만드는 상황을 전제로 하며, 목표는 로딩 시점에 비인가 행위를 발생시키는 것이다. 공격자는 사용자가 로딩 대상으로 선택할 수 있는 모델 파일을 제공하거나, 로딩 전에 해당 파일을 교체·변조할 수 있다고 가정한다. 공격 성공은 로딩 과정에서 비인가 객체 생성 또는 비인가 호출 경로가 활성화되어 정책이 허용하지 않은 동작이 유발되는 경우로 정의한다.

방어 측은 로딩 이전에 파일을 점검할 수 있으나, 본 논문은 사전 점검이 완전한 판별기라고 가정하지 않는다. 따라서 논의의 초점은 사전 점검의 완전성 여부가 아니라, 로딩 시점에서 강제 정책 집행이 성립하는 조건이다. 또한 검증 방식에 따라 신뢰 경계와 실패 모드는 달라진다. out-of-band 방식은 그림 1. (a)와 같이 로더·검증 계층이 정책 집행의 기준점이 되므로 해당 계층의 우회·변조 시 검증이 무력화될 수 있으며 in-band 방식은 그림 1. (b)와 같이 검증 로직이 역직렬화 과정에 포함되므로 검증 로직의 우회·악용 가능성이 있다. 본 논문은 이러한 양 측면의 위험을 포함하여, 검증 경계가 우회·변조되지 않도록 강제되는 조건을 논의한다.



(a) Out-of-band 방식의 위협



(b) In-band 방식의 위협

그림 1. AI 모델 로드 전 강제 검증 방식에 따른 위협

Ⅲ. 검증 경계 분석: In-band와 Out-of-band 비교

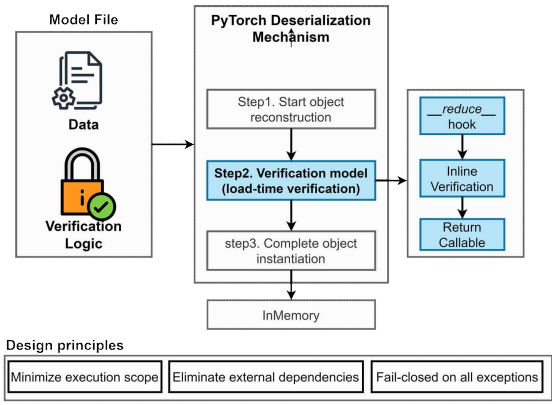
본 절은 PyTorch pickle 역직렬화 기반 공격을 대상으로, 검증 로직의 위치에 따라 형성되는 검증 경계의 성격을 분석한다. 여기서 검증 경계란 역직렬화 과정에서 정책 위반을 판정·차단하는 권한이 실제로 행사되는 지점이며, 공격자가 우회·변조를 시도하는 1차 표적이다. 본 논문은 검증이 모델 파일 내부에 포함되는 in-band와 모델 파일 외부의 로더·검증 계층에서 수행되는 out-of-band를 대비하여, 각 접근이 요구하는 신뢰 가정과 방어가 성립하는 조건을 정리한다.

In-band 검증에서 검증 경계는 직렬화된 객체 그래프 내부에 위치한다. 이 방식은 검증 로직이 모델과 함께 유통된다는 점에서 적용 형태가 단순해 보일 수 있으나, 검증 로직이 역직렬화 흐름에 편입되는 만큼 방어는 검증을 추가했기 때문에 안전하다가 아니라 검증 코드 자체가 공격 표면이 되더라도 안전하게 동작한다는 조건을 충족해야 성립한다. 즉 검증 코드는 역직렬화가 본격적으로 진행되기 전의 조기 단계에서 실행되어야 하며, 검증 완료 전에는 객체 재구성이 진행되지 않도록 순서가 고정되어야 한다. 또한 검증 로직의 실행 범위가 커질수록 공격 표면이 확대되므로, 검증은 최소한의 연산으로 한정되고 외부 의존성을 배제해야 한다. 마지막으로 검증 실패뿐 아니라 예외·불완전한 상태를 모두 로딩 중단으로 처리하는 fail-closed가 전제되어야 한다. out-of-band 검증에서 검증 경계는 모델 파일 외부의 로더 또는 검증 계층에 위치한다. 이 방식은 모델 파일을 변경하지 않고도 정책을 적용할 수 있어 적용 범위를 파일 내부 구조로부터 분리하고, 동일 정책을 다양한 체크포인트에 일관되게 적용하기 용이하다. 다만 방어 성립의 핵심은 검증 로직의 존재가 아니라 검증을 거치지 않고는 역직렬화를 시작할 수 없음이 보장되는지에 있다. 즉 로딩 호출자가 검증 계층을 우회하거나, 검증 계층 자체가 변조되어 검증이 형식화·생략될 수 있다면 out-of-band는 즉시 무력화된다. 따라서 out-of-band는 로딩 진입점과 정책 집행 경로가 강제로 고정되고, 검증 결과가 로딩 단계에서 위·변조 불가능한 형태로 결합되는 조건을 만족해야 한다.

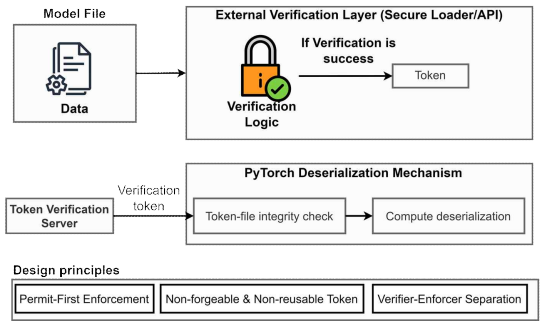
Ⅳ. 강제 검증 경로 설계 원칙

강제 검증 경로는 역직렬화가 시작되기 전에 정책 검증이 반드시 수행되고, 그 결과가 우회·변조되지 않도록 로딩 경로에 고정되는 것을 의미한다. 여기서 강제성은 검증 로직이 존재함이 아니라 검증을 거치지 않고는 역직렬화를 진행할 수 없음으로 정의되며, in-band와 out-of-band 모두에 동일하게 요구된다.

In-band 방식에서는 그림 2. (a)와 같이 검증 로직이 모델 파일 내부에 포함되어 역직렬화 흐름에 편입된다. 따라서 검증은 객체 재구성 이전의 조기 단계에서 실행되어야 하며, 검증 완료 전에는 실제 객체 인스턴스화가 진행되지 않도록 순서를 고정해야 한다. 또한 검증 로직 자체가 공격 표면으로 편입될 수 있으므로, 실행 범위를 최소화하고, 네트워크·파일 시스템·동적 로딩 등 외부 의존성을 제거해야 한다. 검증 실패나 예외, 검증 불가 상황은 모두 로딩 중단으로 처리해야 한다. out-of-band 방식에서는 그림 2.(b)와 같이 검증 경계가 모델 외부 로더·검증 계층에 위치하므로, 강제성의 핵심은 로딩이 검증 계층을 반드시 경유하도록 하는 데 있다. 이를 위해 로딩 진입점은 단일화되어야 하며, 사용자가 직접 역직렬화 API를 호출해 우회할 수 있는 경로를 허용하지 않아야 한다. 검증 결과는 단순한 불리언이 아니라 토큰으로 표현하고, 토큰에는 모델 식별자, 정책 버전, 유효기간을 포함해 바꿔치기·재사용·다운그레이드를 제한해야 한다. 특히 로더·API 변조 위험을 고려하면, 토큰의 유효성 검사는 외부의 신뢰된 검증 서버에서 수행하도록 분리하고, 로딩 단계에서는 토큰 및 파일 일치성 검사를 통해 허용 조건을 고정하는 것이 타당하다. 토큰 검증 실패, 검증 서버 도달 불가 등 검증 수행 불가도 로딩 중단으로 귀결되어야 한다.



(a) In-band 방식 설계 원칙



(b) Out-of-band 방식 설계 원칙

그림 2. 검증 방식 별 설계 원칙

Ⅴ. 결론

본 논문은 PyTorch 체크포인트 로딩 안전성이 개별 방어 기법의 존재보다, 검증 경계가 로딩 경로에 우회 불가능하게 고정되는 방식에 의해 성립함을 보였다. In-band 방식은 검증 로직이 역직렬화 흐름에 편입되는 만큼, 검증 코드의 실행 범위 최소화, 외부 의존성 제거, 그리고 모든 예외 상황에서 로딩을 중단하는 fail-closed가 필수 조건이다. Out-of-band 방식은 로더·API 변조 가능성을 전제로 하므로, 로딩이 검증을 선행하지 않으면 시작될 수 없는 permit-first enforcement, 위·변조 및 재사용이 불가능한 비위조 토큰, 그리고 검증 판단 주체와 로딩 집행 주체를 분리하는 verifier–enforcer 분리를 통해 검증 강제성을 확보해야 한다.

ACKNOWLEDGMENT

본 논문은 과학기술정보통신부 정보통신기획평가원에서 지원한 국산 AI 반도체 기반 마이크로 데이터센터 운영 및 확산 기술 개발 과제로 수행된 연구임 (과제번호: RS-2025-25457382)

참 고 문 헌

[1] National Vulnerability Database, “CVE-2025-32434,” NIST, Apr. 2025, [Online]. Available: https://nvd.nist.gov/vuln/detail/CVE-2025-32434. [Accessed: Jan. 11, 2026].

[2] National Vulnerability Database, “CVE-2025-1945,” NIST, Apr. 2025, [Online]. Available: https://nvd.nist.gov/vuln/detail/CVE-2025-1945. [Accessed: Jan. 11, 2026].

[3] W. Jiang, N. Synovic, R. Sethi, A. Indarapu, M. Hyatt, T. R. Schorlemmer, G. K. Thiruvathukal, and J. C. Davis, “An empirical study of artifacts and security risks in the pre-trained model supply chain,” in Proceedings of the ACM Workshop on Software Supply Chain Offensive Research and Ecosystem Defenses (SCORED), 2022, pp. 105–114.