Falco 탐지 한계 분석 및 우회

표창우, 한승재, 주현석*,김창훈**

*대구대학교 사이버보안전공 (학부생) **대구대학교 사이버보안전공 (교신저자)

Analysis and bypass of Falco detection limits

Pyo Changwoo, Han Seungjae, Joo Hyunsuk*
Kim Chang Hoon **

*Dept. of cyber security Daegu Univ(College student)
**Dept. of cyber security Daegu Univ(corresponding author)

요 약

클라우드 환경이 보편화되면서 Falco가 커널 이벤트 기반 런타임 위협 탐지 도구로 널리 활용되고 있다. Falco는 커널에서 동작하므로 사용자 공간에서의 악성 행위를 제대로 분석하지 못하는 구조적 한계를 가진다. 프로세스 위장, 간접 실행, 암호화된 C2(Command and Control Server) 통신 등 사용자 공간에서 일어나는 일을 탐지하지 못하는 취약점을 본 연구팀은 발견했다. 본 논문에서는 Falco의 탐지 한계를 확인하고, 이를 분석한 결과를 제시한다.

I. 서론

클라우드 환경이 보편화되면서, Apple, IBM 등 여러 다국적 기업이 런타임 보안 애플리케이션으로 Falco를 채택하고 있다 [1]. Falco가 이렇게 널리 사용되는 이유는 크게 세 가지 측면에서 설명할 수 있다. 첫째, Kubernetes와 완벽한 통합으로 컨테이너 환경에서의 운영이 용이하다는 점, 둘째, 리눅스 커널 수준의 시스템 호출 모니터링을 통해 저수준의 위협까지 탐지할 수 있는 기술적 우위, 셋째, 오픈소스로서 무료로 사용할 수 있을 뿐만 아니라 활발한 커뮤니티를 통해 풍부한 자료와 지속적인 발전이 보장된다는 점이다.

이처럼 Falco가 클라우드 보안 시장에서 차지하는 높은 점유율은 보안성에 대한 실증적 연구의 필요성을 제기한다. 비슷한 기능을 제공하는 여러 애플리케이션 가운데 Falco를 연구 대상으로 선정한 이유는 실제 사용 빈도와 영향력을 고려했기 때문이다.

본 연구를 통해 Falco의 감시 체계가 사용자 공간에서 발생하는 행위를 제대로 탐지하지 못하는 취약점을 발견하였고, 한계를 구체적으로 기술했다.

II. 관련 연구

2.1 Falco 개요

Falco는 리눅스 커널에서 발생하는 시스템 호출과 플러그인으로부터 수집된 다양한 이벤트를 실시간으로 관찰·정규화한 뒤, 사전에 정의된 규칙을 기반으로 이상 행위를 탐지하고 경보를 발생시킨다 [2]. Falco는 컨테이너 런타임 및 쿠버네티스 메타데이터와 결합하여 이벤트에 식별 문구를 부여하고, 탐지된 결과는 시스템 로그, 표준 출력 등 다양한 형태로 전달된다. 또한 쿠버네티스 Audit 로그, AWS CloudTrail, 등 여러 플러그인을 통해 클라우드 제어판 이벤트와 계정 활동 로그 등을 함께 수집함으로써, kubectl exec, IAM 변경, 컨테이너 생성·삭제, 정책 설정 변경과 같은 호스트 외부

행위까지 상관 분석할 수 있는 확장된 관찰 체계를 제공한다. 이와 같은 기능적 확장을 통해 Falco는 클라우드 환경에서 널리 활용되고 있다.

2.2 Falco의 구조적인 한계

Falco는 커널 수준의 시스템 호출을 중심으로 이벤트를 수집하여 런타임 행위를 분석하지만, 이러한 구조적 특성은 관찰 범위를 제한한다. eBPF(Extended Berkeley Packet Filter)는 리눅스 커널 내부에서 사용자 정의 코드를 실행해 시스템 이벤트를 추적할 수 있게 하는 확장 프레임워크다. Falco는 이러한 eBPF나 커널 모듈을 통해 이벤트를 수집하고, 사용자 공간으로 전달된 데이터를 정규화한다. 이 과정에서 Falco가 접근할 수 있는 정보는 커널 레벨의 이벤트에 한정되며, 사용자 공간에서 이루어지는 명령 파싱, 복호화, 실행 결정 등의 로직은 탐지 범위 밖에 놓인다. 예를 들어 TLS나 웹소켓과 같이 암호화된 통신으로 전달된 명령은 단순한 데이터 수신으로만 기록되기에 어떤 명령이 실행되었는지 Falco가 판단하기 어렵다.

또한 Falco는 규칙 평가 시 정규화된 필드에 의존하지만, 이러한 메타데이터는 런타임 조작이나 eBPF의 기술적 제약으로 인해 불완전할 수 있다. 프로세스 이름 변경이나 간접 실행은 실행 계보를 위장할 수 있으며 eBPF 수집 과정에서 인자 추출이 완전하지 않아 일부 정보가 누락 될 수 있다. 이에 따라 Falco의 이름 기반 규칙이나 단일 필드 조건은 오탐 또는 미탐을 발생시킬 가능성을 높인다.

마지막으로 Falco는 개별 시스템 호출 단위로 이벤트를 평가하기 때문에, 여러 단계로 분산된 공격 행위를 하나의 실행 체인으로 상관 분석하는 데 한계가 존재한다. 명령 수신, 실행, 결과 전송이 시간 적으로 분리되거나 암호화되어 이루어질 때 Falco는 이를 독립적인 정상 이벤트로 인식한다. 특히 공격자가 무작위 시간 지연을 적용하면 이벤트 간의 시간적 연속성이 약화되어 상관성 복원은 사실상 불가능하다.

III. Falco 우회 백도어 구현

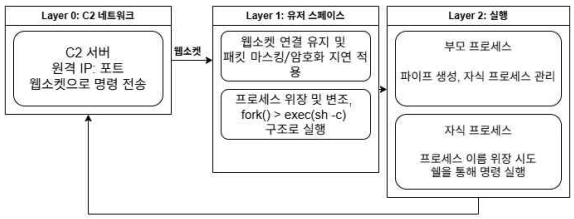
Falco의 관찰, 정규화, 규칙 평가 파이프라인이 결합할 때 발생하는 구조적 한계를 실증적으로 검증하기 위해, 본 연구에서는 네 가지 우회 기법을 결합한 백도어를 설계했다. 설계 목표는 다음과 같다.

(1) 런타임 시 프로세스 이름을 위장하여 이름 기반 규칙을 무력화한다.

(2) /bin/sh -c와 같은 간접 실행 형태를 이용하여 실제 명령을 정규화 단계에서 흐리게 만든다.

(3) 다수 프로세스 간의 관계를 규칙으로 포괄하기 어려운 점을 악용하여 프로세스 트리 분석을 무력화한다.

(4) 암호화된 제어 채널과 무작위 지연을 결합하여 네트워크 수신·명령 실행·결과 전송 간의 시간적 상관성을 약화한다. 백도어의 전체 동작 개요는 Fig 1 에 정리 하였다.



(Fig 1: 백도어의 전체 동작 개요)

공격자는 명령 수신, 복호화, 실행 결정을 모두 사용자 공간에서 수행한다는 가정하에 설계되었으며, 이를 통해 Falco가 커널 수준 이벤트만으로 판단하는 구조적 한계를 검증하도록 했다. 본 연구의 검증 목표는 Falco가 전체 행위 체인을 재구성하지 못함을 실증적으로 확인하는 데 있다.

IV. 실험

4.1 검증

검증은 외부 네트워크와 완전히 분리된 격리 환경에서 수행되었으며, 피해자 역할의 Ubuntu 시스템과 명령 전달용 C2 서버를 구성했다. 실험 환경의 세부 구성은 표 1에 제시했다. 평가 방법은 일반적인 명령어 실행과 백도어를 이용한 원격 명령어 실행을 각각 수행하여 Falco의

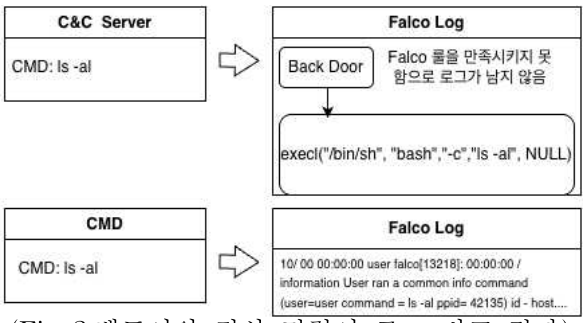
탐지 여부를 비교하는 대조 실험으로 진행했다. 측정 지표로는 로그 생성 여부의 차이를 활용했으며, 명령을 반복 수행하여 결과의 일관성을 검증했다.

(Table 1: 사용된 OS)

OS 및 소프트웨어	버전 정보
Windows 11	24H2 26100.6899
Linux	Ubuntu 22.04
Python	Python 3.11
Falco	0.42

4.2 실험 결과

실험 결과, 정상 쉘에서 직접 ls 명령을 실행했을 때 Falco는 일관되게 시스템콜 이벤트와 정규화된 필드를 기록하고 설정한 규칙에 따라 경고를 생성했다. 반면 동일한 최종 동작을 본 연구에서 구현한 백도어를 원격으로 전송·실행했을 때는 Fig 2 와 같이 Falco 로그에 유의미한 경보가 남지 않았다.



(Fig 2:백도어와 정상 명령어 로그 비교 결과)

V. 결론

본 연구는 Falco의 커널 기반 런타임 탐지 체계가 사용자 공간에서 수행되는 행위와 시간적 분산 공격에 대해 구조적 한계를 가진다는 사실을 실험으로 검증했다. 일반적인 셸 환경에서 명령어를 입력하였을 때 Falco가 시스템콜 이벤트를 정확히 기록하고 경고를 발생시켰으나, 본 연구에서 구현한 백도어를 통해 동일 명령을 간접 실행한 경우 탐지가 이루어지지 않았다. Falco가 커널 수준의 이벤트에만 의존하기 때문에 사용자 공간에서의 복호화·파싱 단계나 메타데이터 변조가 결합되면 Falco의 탐지 시스템에 명확한 공백이 발생하는 것을 확인했다. 이러한 탐지 공백을 보완하기 위해서는 사용자 공간 로그의 수집 및 상관 분석 강화, 프로세스 입력 인자값 무결성 확보, 보완 로그의 병행 수집 모듈이 필요하며, 다중 이벤트를 통합적으로 분석하는 시스템도 필요하다. 이를 통해 Falco의 탐지 신뢰성을 향상시키고 오탐·미탐을 줄일 수 있음을 제안한다.

(Acknowledgement)

본 논문은 과학기술정보통신부 및 정보통신기획평가원의 'SW중심대학사업'의 지원을 받아 제작되었습니다. (2025-0-00077)

[참고문헌]

[1] Sysdig, Sysdig Falco Graduates to CNCF, Sysdig Official Press Release, June, 2022. Available: https://www.sysdig.com/press-releases/sysdig-falco-graduates-cncf

[2] Falco Project, The Falco Project | Falco (What is Falco?), The Falco Project Documentation, Last modified March 12, 2025. Available: https://falco.org/docs/

[3] Falco Project, Kernel Events Architecture | Falco (Event Sources), The Falco Project Documentation, Last modified Dec. 13, 2024. Available: https://falco.org/docs/concepts/event-sources/kernel/architecture/

[4] Falco Project, Custom Ruleset | Falco (Custom Rules), The Falco Project Documentation, Last modified Oct. 22, 2025. Available: https://falco.org/docs/concepts/rules/custom-ruleset/