피드백 루프 기반 AI Agent를 활용한 DNS 캐시 포이즈닝 공격 프레임워크 제안

우한별*, 노현민*, 조영호(교신저자)**
*국방대학교 국방관리대학원 국방과학학부 사이버컴퓨터공학과 석사과정
**국방대학교 국방관리대학원 국방과학학부 사이버컴퓨터공학과 교수
e-mail:cousinot@naver.com, erta455@naver.com, younghocho@korea.kr

A DNS Cache Poisoning Attack Framework Using AI Agents with Feedback Loop

Hanbyeol Woo*, HyeonMin No*, Youngho Cho**
*Master's Course, Dept. of Cyber Security and Computer Engineering, Korea National Defense University
***Professor, Dept. of Cyber Security and Computer Engineering, Korea National Defense University

요 약

본 논문은 대형 언어 모델(LLM: Large Language Model) 기반 AI 에이전트가 DNS(Domain Name System) 캐시 포이즈닝 공격을 자율 수행하는 프레임워크를 제안한다. 제안 프레임워크는 정찰(Recon), 코드 생성(Gen), 실행(Executor), 검증(Validator)의 4단계 에이전트 파이프라인으로 구성되며, 검증 에이전트의 피드백을 생성 단계에 즉각 반영하는 자율 피드백 루프를 통해 공격 전략을 점진적으로 최적화한다. Docker 격리 환경 내 Unbound와 BIND9 리졸버를 대상으로 방어 수준(D1~D8)에 따른 8개 조합에 대해 세 차례(v1~v3) 실험을 수행하였다. 실험 결과, 최종 v3에서 87.5%(7/8)의 공격 성공률(ASR: Attack Success Rate)을 달성하였으며, 특히 LLM이 자율 설계한 하이브리드 공격이 복합 방어 환경을 무력화 할 수 있음을 실증하였다.

1. 서론

DNS(Domain Name System)은 사람이 읽기 쉬운 도메인 이름을 컴퓨터가 인식하는 IP 주소로 변환해주는 시스템으로 인터넷 통신을 가능하게 하는 핵심 인프라이다. 2008년 Kaminsky 공격[1] 발생 이후 포트 랜덤화[2], 0x20 인코딩[3], DNS Cookie[4] 등 DNS에 대한 다층 방어가 도입되었다. 그러나 SAD DNS의 ICMP 사이드채널과 같이 새로운 우회 기법이 지속 발견되고 있다[5]. 본 논문에서는 공격자가 TX ID와 소스 포트를 블라인드로 추측해야 하는 캐시 포이즈닝 공격 시나리오를 연구 대상으로 한다.

최근 LLM을 활용한 자동공격 연구가 증가하고 있다. Fang 등[6]은 GPT-4 기반 에이전트가 1-day 취약점의 87%를 자율 공격할 수 있음을 보였다. 그러나 기존 연구는 웹 및 호스트 수준 공격에 집중되며, DNS 프로토콜 수준의 패킷 위조와 다층 방어 우회에 관한 실증 연구는 부재하다.

본 연구는 LLM 기반 AI 에이전트가 현대 DNS 리졸버의 다층 방어를 자율 분석하고 우회하는 공격 프레임워크를 제안한다. 제안 프레임워크는 4단계 에이전트 파이프라인과 자율 피드백 루프를 결합한다. 본 연구의 주요 공헌사항은 다음과 같다. (1) LLM 기반 자율 DNS 캐시 포이즈닝 공격 프레임워크를 설계·구현한다. (2) 다양한 방어 수준별(D1~D8) 실험을 통해 방어 효과를 분리 측정한다. (3) LLM이 설계한 하이브리드 공격이 전체 방어 환경을 돌파할 수 있음을 실증한다.

2. 배경지식 및 관련연구

2.1 DNS 캐시 포이즈닝 공격 및 방어

DNS 캐시 포이즈닝은 DNS 스푸핑을 통해 리졸버 캐시에 위조된 레코드를 주입하는 공격이다. 공격자가 전송한 스푸핑 응답이 리졸버가 권위 서버로부터 수신하는 정답 응답보다 먼저 도착하면 해당 레코드가 캐시에 저장되어, 이후 동일 도메인에 대한 피해자 쿼리에 위조된 결과가 반환된다.

Kaminsky[1]의 TX ID 무작위 대입 공격 이후, 포트 랜덤화[2], 0x20 인코딩[3]과 DNS Cookie[4]가 방어 수단으로 도입되어 단순 포이즈닝 공격은 차단되었다. 하지만 SAD DNS[5]는 리눅스 커널의 ICMP rate limit을 활용한 사이드채널로 포트 랜덤화[2]를 우회하였고, 0x20 인코딩은 도메인의 알파벳 길이가 짧거나 일부 권위 서버가 대소문자를 보존하지 않는 경우 엔트로피 한계로 인해 케이스 무작위 대입을 통한 우회가 가능하다. DNS

Cookie 또한 BIND9의 예측가능한 PRNG 결함(CVE-2025-40780)이 공개되어 쿠키 값 사전 예측을 통한 우회가 가능함이 확인되었다.

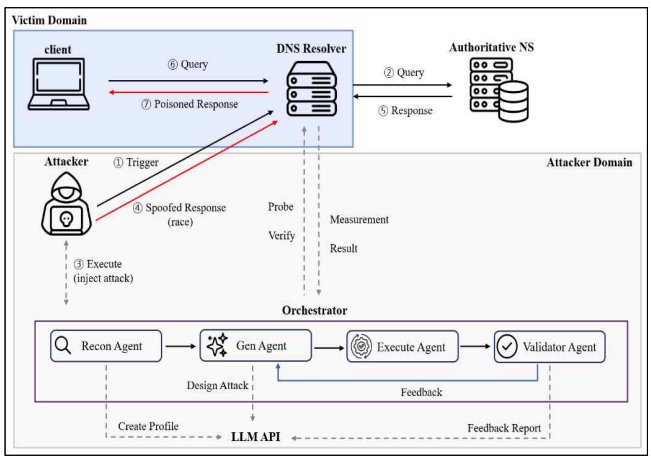
2.2 LLM 기반 사이버 공격 자동화

Fang 등[6]은 GPT-4 단일 에이전트로 1-day CVE 13개(87%)를 익스플로잇하였다. Iturbe 등[9]는 MITRE ATT&CK TTP를 prompt attribute로 주입하여 jailbreak 없이 공격 코드 생성이 가능함을 실증하였다. 그러나 이러한 연구들은 주로 웹 및 호스트 수준 공격을 대상으로 하며, 탐지기 우회 성능을 정량적으로 측정하지 않았다. 이와 다르게, 본 연구는 DNS 프로토콜 수준의 패킷 위조를 대상으로 하며, 탐지 환경에서의 우회 여부를 직접 측정한다는 점에서 차별화된다.

2.3 자율 에이전트 및 피드백 시스템

Reflexion[7]은 이전 실패 원인을 자연어로 분석하여 다음 시도에서 전략을 수정하는 피드백 구조를 제안하였다. Self-Refine[8]은 생성-피드백-개선 루프로 수학 추론, 코드 최적화, 대화 응답 생성 등 7개 과제에서 평균 20%p 향상을 달성하였다. 제안 프레임워크는 Validator의 피드백을 Reflexion의 자기반성으로, 반복 개선은 Self-Refine의 구조를 차용하여 DNS 프로토콜 공격에 적용한다.

3. 위협 모델 및 연구 문제 정의



[그림 1] 위협 모델 정의

본 연구에서의 위협 모델은 [그림 1]과 같으며 공격자(attacker), 피해자 리졸버(DNS Resolver), 권위 서버(Authoritative NS), 정상 클라이언트(client)로 구성된다. 공격자는 리졸버가 권위 서버에 쿼리를 보내는 짧은 시간(race window) 동안 스푸핑 응답을 주입하여 캐시를 오염시키며, 이후 정상 클라이언트의 질의에 오염 응답이 전달된다.

- **공격자**: 공격자는 LLM 에이전트 파이프라인을 통해 인간 개입 없이 피해자 도메인 분석, 공격 설계, 코드 실행, 결과 검증을 자율 수행한다. 이때, 피해자 리졸버와 상위 권위서버 간 통신을 관측할 수 없으며, MITM 등 네트워크 경로 제어 능력이 없는 것을 가정한다. TX ID와 소스 포트는 무작위 대입 또는 ICMP 사이드 채널을 통해 추론한다.
- **방어자**: DNS 리졸버(Unbound, BIND9)에 다양한 방어 수준(D1~D8)을 적용한다. 모든 방어(Port 랜덤화, 0x20 인코딩, DNS Cookie)가 비활성화된 Baseline(D1, D2)부터 모두 활성화된 방어(D7, D8)까지 [표 1]과 같이 정의한다.

[표 1] 방어 환경 정의

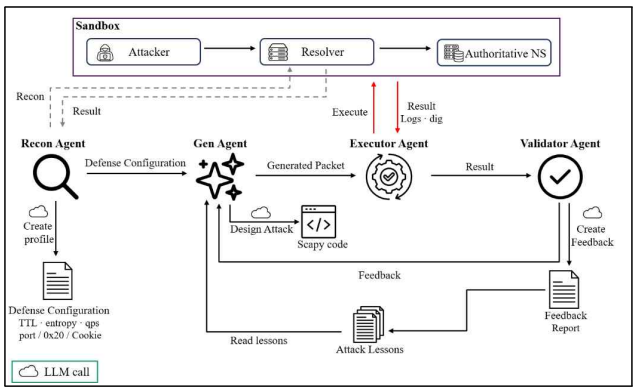
ID	Resolver	방어기능		
		Port 랜덤화	0x20	DNS Cookie
D1	Unbound	OFF	OFF	–
D2	Bind9	OFF	–	OFF
D3	Unbound	ON	OFF	–
D4	Bind9	ON	–	OFF
D5	Unbound	OFF	ON	–
D6	Bind9	OFF	–	ON
D7	Unbound	ON	ON	–
D8	Bind9	ON	–	ON

본 논문에서는 다음 세 가지 문제를 연구한다. 첫째, LLM 기반 에이전트 파이프라인이 다층 방어(D1~D8)를 자율 우회하여 캐시 포이즈닝을 달성할 수 있는지 확인한다. 둘째, 방어 수준별 ablation 실험으로 각 방어 메커니즘의 효과를 분리 측정할 수 있는지 검증한다. 셋째, LLM이 신규 공격 조합을 직접 설계하여 방어 환경을 돌파할 수 있는지 실증한다.

4. 제안 프레임워크

4.1 전체 아키텍처(그림 2)

제안 프레임워크는 4개 에이전트(Recon, Gen, Executor, Validator)로 구성된 파이프라인과 자율 피드백 루프로 구성된다. Recon, Gen, Validator 에이전트는 LLM API를 호출하여 방어 분석, 공격 코드 생성, 결과 검증을 수행하고, Executor 에이전트는 LLM 호출 없이 Gen agent 생성 코드를 Sandbox 컨테이너 내부의 Attacker를 통해 공격 시도한다.



[그림 2] 제안 프레임워크

- **정찰 에이전트(Recon)**: 타겟 리졸버에 실제 DNS 질의를 전송

하여 TX ID 엔트로피, 소스 포트 분포, 0x20 인코딩 여부, DNS Cookie 지원, TTL 패턴, 응답 속도 제한을 측정한다. LLM은 측정 결과를 해석하여 타겟 리졸버의 방어 구성을 파악하고 방어 프로파일을 생성한 후 생성 에이전트에 전달하여 공격 유형 선택의 입력으로 활용한다.

• **생성 에이전트(Gen)**: 정찰 프로파일, 이전 반복의 검증 피드백, 장기 메모리를 입력으로 받으며, LLM이 성공한 공격 유형을 축적하여 정리 공격 카탈로그를 참조한다. LLM은 이 카탈로그에서 대상 방어 구성에 적합한 유형을 선택하거나, 신규 조합을 자율 설계하며, 생성된 코드(Scapy)를 Executor로 전달한다.

• **실행 에이전트(Executor)**: Scapy 코드를 컨테이너에서 실행한다. 실행 전 AST 기반 코드 검증기가 함수명과 인자 오류를 자동 교정하며, 실행 중 pcap을 수집한다. 수집된 pcap과 실행 로그는 검증 에이전트에 전달된다.

• **검증 에이전트(Validator)**: 공격 결과를 2축으로 평가한다. 평가는 오염 여부(Victim의 dig 쿼리 기반), Snort IDS (커스텀 룰셋) 탐지 여부이다. 성공한 공격은 공격 카탈로그 메모리에 등록되어 다음 공격에 활용된다. 평가 결과는 다음 반복의 생성 에이전트에 전달되는 피드백으로 합성된다. 피드백은 이전 공격 유형, 결과, 탐지 패턴, 방어 우회 진척도를 포함하며, 이전과 동일한 공격 유형의 반복을 방지하기 위해 cross iteration 패턴 감지를 포함한다.

5. 초도 실험 결과

5.1 실험 목적

본 실험은 3장에서 제시한 세 가지 연구 문제를 검증하기 위해 설계하였다. 방어 수준(D1~D8)을 독립 변수로 설정하여 8개 조합에 대해 제안 프레임워크의 버전을 세 차례 업그레이드하며(v1~v3) 실험을 수행하였다. 버전별 피드백 구조 개선이 공격 성공률과 방어 난이도 돌파에 미치는 영향을 분석한다.

5.2 실험 환경 및 평가지표

실험 환경은 [표 2]와 같다. fake_upstream은 응답 지연을 통해 race window를 제공하며, 이는 SAD DNS[5] 등 기존 연구와 동일한 표준 실험 구성이다.

[표 2] 실험 환경

구성요소	상세
LLM	Claude Sonnet 4.6 (Anthropic)
DNS 리졸버	Unbound 1.22.0 / BIND9 9.20.11
탐지기	Snort 3 + ML Detector (RF 6-feature)
네트워크	Docker 172.20.0.0/24 (8 컨테이너)
실험 규모	D0~D3 X 2 리졸버 = 8조합
실험 회차	v1, v2, v3 (3차, iter 상한 20)

버전 별 피드백 구조 차이는 다음과 같다.

• **v1 (베이스라인)**: 9종 고정 공격 카탈로그를 최대 20회 반복 시도한다. 검증 에이전트의 실패 원인만 다음 생성 에이전트에 전달한다.

• **v2 (하이브리드 설계)**: 공격 유형별 성공·실패 이력을 축적하고, 실패 원인을 4가지(TIME OUT, CODE ERROR, RACE LOST, DEFENSE BLOCKED)로 분류한다. 공격 카탈로그가 모두 실패하면 LLM이 하이브리드 공격을 새로 설계한다.

• **v3 (누적 학습)**: 피드백에 방어 구성 정보를 추가하고, 성공한 하이브리드 공격을 메모리에 등록한다. 생성 에이전트가 이를 재사용하여 누적 학습을 수행한다.

본 실험에서 사용한 평가지표는 다음과 같이 정의한다.

• **ASR(Attack Success Rate)**: 버전별 전체 방어 조합(8개) 중 캐시 오염에 성공한 조합의 비율로 범위는 [0,1]이다. (즉, ASR = 성공 조합 수 / 전체 방어 조합 수). Victim에서 dig 쿼리로 위조 레코드 존재를 확인한 경우 성공으로 판정한다.

• **성공 회차(iter to success)**: 각 방어 조합에서 공격이 성공한 반복 회차로, "성공 회차/최대 반복 한도(20회)" 형식으로 표기한다. 최대한도 내 성공하지 못한 경우 "실패"로 표기한다.

5.3 실험 결과

5.3.1 공격 카탈로그

Gen 에이전트가 참조하는 공격 카탈로그는 [표 3]과 같이 프레임워크 설계 간 축척 된 9종 기본 카탈로그와 LLM이 실험 중 설계한 Hybrid 2종으로 구성된다. 흥미로운 점으로 LLM은 실험간 사용된 모델의 학습 컷오프 (2025년 8월) 이후 공개된 CVE-2025-40780 관련 취약점을 피드백 과정에서 직접 발견하고 cookie preimage 공격을 설계하였다..

[표 3] 공격 카탈로그

번호	공격 유형	원리
1	kaminsky_txid	TX ID brute-force
2	sad_dns	ICMP rate-limit 사이드채널로 소스 포트 추론 후 race
3	negative_cache_poison	NXDOMAIN위조로 negative cache 오염
4	cookie_preimage	DNS Cookie 수집 후 위조 응답
5	port_sweep	권위서버 로그 기반 TX ID·포트 획득
6	case_mutation	0x20 대소문자 변형 전수 시도
7	birthday_flood	다중 서브도메인 병렬 질의
8	cname_chain	CNAME 체인 레코드 주입으로 리졸버 캐시 경로 변조
9	ns_injection	Authority/Additional 섹션에 위임 NS 레코드 주입
H1	cookie_preimage_kaminsky (v2)	DNS Cookie 수집·embed + TX ID 추측 결합
H2	cookie_negative_cache_hybrid (v3)	DNS Cookie 수집·embed + NXDOMAIN flood 결합

5.3.2 버전별 실험 결과

우선, [표 4]는 방어 수준별·리졸버별·버전별 공격 성공 회차이다. 동일 방어 조합에서도 버전마다 성공 회차가 달라지는데, 이는 TX ID race 등 확률적 공격의 특성과 피드백 누적에 따른 전략 선택이 결합된 결과이다.

[표4] 버전별 성공 회차(iter to success)

조합	v1	v2	v3
D1	3/20	3/20	3/20
D2	2/20	2/20	17/20
D3	6/20	6/20	2/20
D4	11/20	13/20	13/20
D5	실패	18/20	7/20
D6	4/20	11/20	8/20
D7	실패	실패	실패
D8	2/20	2/20	17/20
ASR	6/8 (75%)	7/8 (87.5%)	7/8 (87.5%)

다음으로, [표 5]는 프레임워크의 버전별로 성공한 공격 유형으로 버전마다 상이한 전략이 성공함을 보여준다. 이는 LLM이 공격을 반복하는 것이 아니라 정찰과 피드백에 기반해 공격의 설계를 실증하며, v2와 v3에서는 LLM이 추가로 설계한 하이브리드 공격에 성공하여 v1에서 실패한 D5에 대해 성공하였다.

[표 5] 버전별 성공 공격

조합	v1	v2	v3
D1	port_sweep	port_sweep	negative_cache_poison
D2	port_sweep	port_sweep	sad_dns
D3	ns_injection	ns_injection	ns_injection
D4	kaminsky_txid	cookie_preimage_kaminsky	cname_chain
D5	실패	cookie_preimage	negative_cache_poison
D6	cookie_preimage	cookie_preimage	port_sweep
D7	실패	실패	실패
D8	negative_cache_poison	negative_cache_poison	cookie_negative_cache_hybrid

6. 결론

6.1 논의(Discussion)

본 연구는 LLM 기반 자율 DNS 캐시 포이즈닝 프레임워크를 제안하고, 리졸버 2종을 바탕으로 8개 조합에 대해 v1~v3 실험을 수행하였다. 주요 결과는 다음과 같다. 첫째, LLM이 공격 설계와 실행을 직접 수행할 수 있음을 확인하였다. 둘째, Validator의 피드백이 Gen agent의 입력으로 연결되는 구조를 통해, 공격이 성공하도록 최적화되어 타겟에 대한 공격의 성공을 실증하였다.

제안기술의 한계점은 세 가지 측면으로 정리할 수 있다. 첫째, 일부 방어 조합 공격에 실패하였으며, 공격 카탈로그가 표준 지식베이스를 사용하지 않고 자체 설계에 의존하였다. 둘째, 일부 공격 코드의 구현에 실패하여 코드 구현 단계에서의 한계점을 확인하였다. 셋째, 리졸버 자체의 방어 설정만을 대상으로 실험을 진행하여, 실제 네트워크 환경에 존재하는 외부 방어 구성(Snort IDS, ML 기반 탐지기)에 대한 우회 가능성은 실증하지 못하였다.

6.2 향후 연구

초도 실험을 바탕으로 한 향후 연구 방향은 다음과 같다.

첫째, 지식베이스 표준화를 통해 공격 카탈로그의 재현성과 외부 연구와의 호환성을 확보할 예정이다. Iturbe 등[8]의 MITRE attribute 기반 prompt 접근을 차용하여 표준화된 항목을 추가함으로써 자체 설계된 카탈로그의 한계를 보완한다.

둘째, 구현 피드백을 특화하여 생성되는 코드의 의도-실행 정합성을 자동 검증할 것이다. 코드 구현 단계에 특화된 피드백 구조를 도입하여 생성 코드가 의도한 공격을 실제로 수행하는지 자동 검증함으로써 코드 구현 단계의 실패를 개선한다.

셋째, 탐지기 확장을 통해 실제 운영 환경에 근접한 우회 성능을 실증할 것이다. Snort 표준 룰셋(ET Open)과 LLM 기반 의미 탐지기를 포함한 확장 탐지 환경에서 우회율을 재측정하여, 실험에서 다루지 못한 외부 방어에 대한 우회 가능성을 검증한다.

참고문헌

[1] D. Kaminsky, "Black Ops 2008: It's the End of the Cache As We Know It," in Proc. Black Hat USA, 2008.

[2] A. Hubert et al., "Measures for Making DNS More Resilient against Forged Answers," RFC 5452, IETF, 2009.

[3] D. Dagon et al., "Increased DNS Forgery Resistance Through 0x20-Bit Encoding," in Proc. ACM CCS, pp. 211-221, 2008.

[4] D. Eastlake et al., "Domain Name System (DNS) Cookies," RFC 7873, IETF, 2016.

[5] K. Man et al., "DNS Cache Poisoning Attack Reloaded: Revolutions with Side Channels," in Proc. ACM CCS, pp. 1337-1350, 2020.

[6] R. Fang et al., "LLM Agents Can Autonomously Exploit One-day Vulnerabilities," arXiv:2404.08144, 2024.

[7] N. Shinn et al., "Reflexion: Language Agents with Verbal Reinforcement Learning," in Proc. NeurIPS, 2023.

[8] A. Madaan et al., "Self-Refine: Iterative Refinement with Self-Feedback," in Proc. NeurIPS, 2023.

[9] E. Iturbe et al., "Unleashing Offensive Artificial Intelligence: Automated Attack Technique Code Generation," Computers & Security, vol. 147, 104077, 2024.