AhnLab Cyber Threat Intelligence Report

TLP : CLEAR

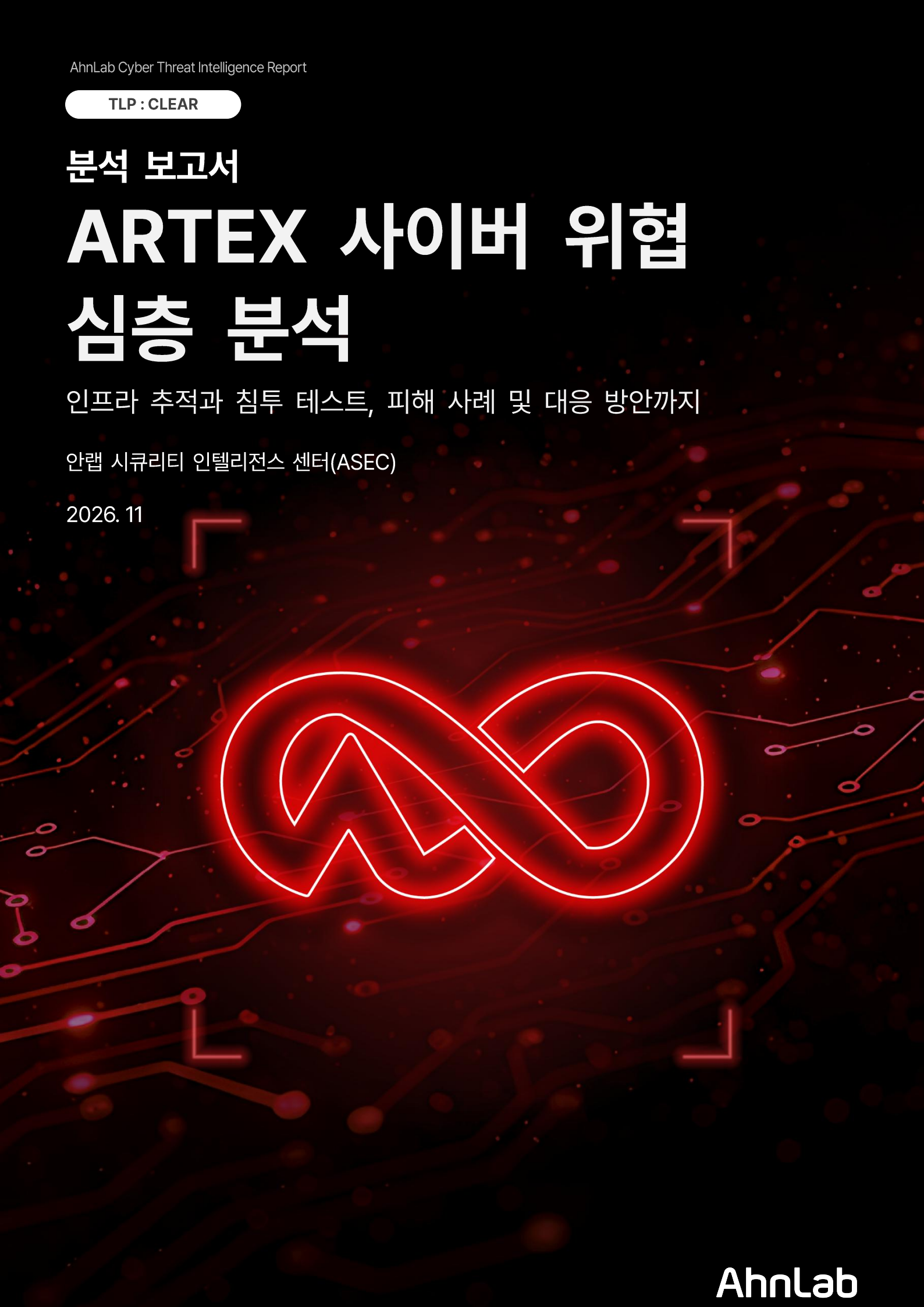
분석 보고서

ARTEX 사이버 위협 심층 분석

인프라 추적과 침투 테스트, 피해 사례 및 대응 방안까지

안랩 시큐리티 인텔리전스 센터(ASEC)

2026. 11



AhnLab

목차

개요 3

1. 보고서 소개 3

2. 요약 3

Part 1. ARTEX 인프라 추적 4

1. 최초 인프라 식별 4

2. 추가 인프라와 ARTEX v2 6

3. 노출된 운영 디렉터리에서 확인된 공격 정황 11

4. Copy Cat 등장 - 파생 배포본과 REAPER 사례 19

5. 추적 내용 요약 및 결론 26

Part 2. ARTEX 침투테스트 결과 분석 28

1. ARTEX 구성도 및 주요 기능 28

2. ARTEX 침투테스트 설정 32

3. ARTEX 침투테스트 #1: 파일 업로드 취약점을 통한 웹 서버 침투 38

4. ARTEX 침투테스트 #2: DFIR 보고서 기반 ApacheMQ 침투 42

5. 결과 분석 48

Part 3. ARTEX 침해 사례 분석 50

1. 한국 침해 사례: 금융권부터 시작된 연쇄 침해 50

2. 일본 침해 사례: 한국 사례와 같은 패턴 52

3. 동시다발로 드러난 치명적 취약점 53

4. Agentic AI 공격: 실험에서 실전으로 55

5. 공통 패턴: 외곽 접점에서 시작된 침해 56

6. 변화하는 예방 및 보안 전략 57

개요

1. 보고서 소개

ARTEX는 대규모 언어 모델과 보안 도구를 결합해 자산 탐색부터 취약점 확인, 도구 실행과 결과 정리까지 연결하는 자율 침투 테스트 프레임워크다. 공개 도구인 만큼 보안 점검과 공격 활동에 모두 활용될 수 있다. 이 보고서는 ARTEX 관련 인프라가 식별되고 변형 배포본이 나타나기까지의 추적 결과를 시간순으로 정리한다. 이어 도구의 구조와 두 차례 침투 테스트에서 확인한 동작을 살펴보고, 국내외 피해 사례와 탐지 및 대응 방안을 제시한다.

*참고: 본 보고서의 내용은 지난 10월부터 ARTEX를 추적 및 분석한 것으로 당시 시점에 맞춰 작성되었음을 알린다.

2. 요약

ARTEX 관련 인프라를 추적한 결과, ARTEX 인스턴스가 호스팅된 약 600개의 추가 IP가 확인됐다. 일부 IP에서는 CyberStrikeAI를 비롯한 다른 보안 도구도 함께 운영되고 있었다. 다만 ARTEX와 CyberStrikeAI는 공개 도구이므로 인스턴스가 확인되거나 동일 IP에서 두 도구가 함께 운영된다는 사실만으로 악성 여부, 운영 주체 또는 특정 공격과의 연관성을 판단할 수 없다.

인터넷에 노출된 운영 디렉터리에서는 웹쉘 제어, 내부망 탐색, 피벗, 자격증명 활용 및 임플란트와 관련된 자료가 확인됐다. 해당 자료는 공격 활동과 부합하는 정황을 보여주지만, ARTEX가 모든 산출물을 생성했거나 두 사례가 동일한 운영자에 의해 수행됐다는 근거는 확인되지 않았다. 원본 저장소가 사라진 이후에도 ARTEX 기반 명칭과 인터페이스, 에이전트 설정 등을 변경한 파생 배포본이 확인됐다.

ARTEX를 이용한 두 차례 침투 테스트에서는 대상 조사 결과에 따라 후속 작업을 계획하고 도구를 실행하는 과정이 확인됐다. 파일 업로드 취약점이 있는 웹 서버에서 웹쉘을 업로드해 프로그램을 실행했으며, Apache ActiveMQ 환경에서 초기 접근 성공 뒤 내부 호스트 네 대에서 원격 명령을 실행했다.

2026년 10월 공개된 국내외 침해 사례에서는 웹서비스, 백엔드 API, 위탁 서비스 등 외곽 접점이 침해 경로로 나타났다. 일부 사례에서는 ARTEX를 포함한 AI 에이전트 활용 정황도 확인됐다. 외부에 노출된 자산과 접근 경로를 점검하고, 웹·네트워크·인증 로그와 단말 탐지 결과를 연계해 초기 침투와 명령 실행, 내부망 이동의 징후를 분석할 필요가 있다.

Part 1. ARTEX 인프라 추적

1. 최초 인프라 식별

*기준일: 2026-09-30

A. ARTEX란



[그림 1] ARTEX Dashboard

ARTEX는 GitHub 사용자 Autumn-27이 공개한 오픈소스 LLM 기반 자율 침투 테스트 프레임워크이다. 다수의 AI Agent가 역할을 분담하여 자산 탐색, 취약점 식별 및 검증 등 침투 테스트 과정을 계획하고 수행하도록 설계되었으며, 필요 시 사용자가 실행 과정에 개입할 수 있다. 백엔드는 Go로 개발되었으며 웹 인터페이스는 Next.js 기반으로 구성되어 있다.

ARTEX는 중국 보안 커뮤니티에서 개발된 프로젝트일 가능성이 높은 것으로 판단된다. 프로젝트 문서와 사용자 인터페이스가 대부분 중국어로 작성되어 있으며, 중국의 WeChat 플랫폼을 통해 SecSentry 커뮤니티를 운영하고 있다. 또한 프로젝트 문서에는 중국 본토의 사이버보안 및 데이터 보호 관련 법률이 명시되어 있다. 개발자는 ARTEX를 Baidu의 'agent+' 공방 대회 우승 프로젝트라고 소개하고 있다.
GitHub: https://github[.]com/Autumn-27/ARTEX

B. ARTEX 인프라

인프라 피벗을 통해 ARTEX 인스턴스가 호스팅된 약 600개의 추가 IP 주소를 확인하였다.

다만 ARTEX는 GitHub에 공개된 오픈소스 도구로 누구나 다운로드하여 구축할 수 있다. 따라서 확인

된 약 600개의 IP를 동일한 조직이나 특정 위협 행위자가 운영하는 단일 인프라로 판단하기 어렵다. 해당 인프라에는 서로 연관성이 없는 다수의 운영자가 포함되어 있을 가능성이 높으며, 실제 공격자뿐만 아니라 레드팀, 침투 테스트 수행자, 보안 연구자 등 정상적인 목적으로 ARTEX를 사용하는 사례도 포함될 수 있다.

이에 따라 ARTEX 인스턴스가 확인되었다는 사실만으로 해당 IP의 악성 여부를 판단하거나 특정 위협 행위자 또는 캠페인과 연관 짓기에는 충분하지 않다. 개별 인프라의 악성 여부 및 공격 연관성을 판단하기 위해서는 추가적인 정황과 근거가 필요하다.

C. CyberStrikeAI Overlap

ARTEX 호스팅 인프라를 분석하는 과정에서 동일한 IP에서 CyberStrikeAI가 함께 운영되는 사례가 다수 확인되었다. (https://github[.]com/aipentest/cyberstrikeai)

CyberStrikeAI는 다중 Agent 오케스트레이션과 다양한 보안 도구, 워크플로 자동화, 공격 체인 분석, WebShell 및 C2 관리 기능 등을 제공하는 오픈소스 AI 기반 공격 보안 플랫폼이다. GitHub에서 7,000개 이상의 Star를 기록하고 있어 ARTEX보다 상대적으로 널리 사용되고 있다.

또한 CyberStrikeAI는 중국 보안 커뮤니티에서 개발된 프로젝트일 가능성이 있는 것으로 판단된다. 프로젝트 개발자 및 관련 활동에서 중국 보안 커뮤니티와의 연관성이 확인되며, 공개 인프라 역시 중국, 홍콩, 싱가포르 지역에서 다수 관찰된 바 있다.

CyberStrikeAI가 주목되는 이유는 실제 공격 활동과 연관된 인프라에서 해당 도구가 확인된 사례가 이미 공개되었기 때문이다. Team Cymru는 2026년 3월 212.11.64[.]250에서 CyberStrikeAI 서비스가 노출되어 있었으며, 해당 시스템과 공격 대상이 된 Fortinet FortiGate 장비 간 NetFlow 통신이 확인되었다고 보고하였다. 또한 2026년 1월 20일부터 2월 26일까지 총 21개의 CyberStrikeAI 서버를 식별하였으며, 이 가운데 상당수가 중국, 싱가포르, 홍콩에 위치한 것으로 확인되었다.

다만 ARTEX와 CyberStrikeAI 모두 누구나 사용할 수 있는 공개 도구이다. 따라서 동일 IP에서 두 도구가 함께 확인되었다는 사실은 인프라 중복을 의미할 뿐, 이를 근거로 동일한 위협 행위자가 운영한다고 판단하거나 특정 공격 캠페인과 직접 연관 짓기에는 충분하지 않다.

2. 추가 인프라와 ARTEX v2

*기준일: 2026-10-05

A. ARTEX 추가 인프라

ARTEX 관련 인프라에서 노출된 웹 서비스를 추가 분석한 결과, ARTEX와 함께 다양한 AI API 게이트웨이, 공격 표면 관리(ASM) 플랫폼, 침투 테스트 프레임워크 및 보안 관리 도구가 운영되고 있는 것으로 확인되었다. 이 가운데 Sub2API - AI API Gateway가 252개 시스템(29.72%)에서 확인되어 가장 높은 비중을 차지하였으며, Scope Sentry는 22개 시스템(2.59%), CyberStrikeAI는 15개 시스템(1.77%)에서 확인되었다. 이외에도 ARL-Next, LunaFox, Acunetix, MultiC2, Mythic 등 보안 및 공격 보안 관련 도구가 소수의 시스템에서 추가로 식별되었다.

아래 표는 ARTEX 관련 인프라에서 확인된 웹 서비스 타이틀을 정리한 것으로, 보안 관련 서비스뿐만 아니라 ARTEX와 직접적인 연관성이 확인되지 않은 범용 웹 애플리케이션도 포함하였다. 단순 HTTP 오류 및 상태 페이지는 제외하였다.

분석 결과, 일부 인프라는 ARTEX 전용으로 사용되는 것이 아니라 보안, 침투 테스트, AI, 인프라 관리 및 일반 웹 서비스가 함께 운영되는 형태로 구성된 것으로 확인되었다. 다만 이러한 서비스가 동일한 시스템에 함께 존재한다는 사실만으로 공통 운영 주체를 특정하거나, 확인된 모든 시스템이 악성 목적으로 사용되고 있다고 판단하기는 어렵다.

웹 서비스 / 페이지 제목	한국어 의미	확인 건수	비율	분류
ARTEX — 自主渗透测试控制台	ARTEX — 자율 침투 테스트 콘솔	394	46.46%	AI 침투 테스트
Sub2API - AI API Gateway	AI API 게이트웨이	252	29.72%	AI / API
Scope Sentry	—	22	2.59%	공격 표면 관리
CyberStrikeAI	—	15	1.77%	AI 침투 테스트
Welcome to nginx!	—	12	1.42%	웹 서버
Directory listing for /	디렉터리 목록	7	0.83%	파일 / 웹 서버
Loading...	로딩 중	6	0.71%	일반
资产灯塔系统	자산 식별·모니터링 시스템	6	0.71%	자산 관리
noVNC	—	5	0.59%	원격 관리

웹 서비스 / 페이지 제목	한국어 의미	확인 건수	비율	분류
ARL-Next	—	4	0.47%	자산 탐색 / ASM
星环 (Xingrin) - 攻击面管理平台 | ASM	Xingrin - 공격 표면 관리 플랫폼	3	0.35%	공격 표면 관리
登录 · AtlasX	로그인 · AtlasX	3	0.35%	보안 / 관리
Grok2API	—	2	0.24%	AI / API
Java Chains	—	2	0.24%	보안 도구
New API	신규 API	2	0.24%	API
AIClient2API - 管理控制台	AIClient2API - 관리 콘솔	1	0.12%	AI / API
ARTEX v2 · Agent Security Workbench	ARTEX v2 · 에이전트 보안 워크벤치	1	0.12%	AI 침투 테스트
Acunetix	—	1	0.12%	취약점 스캐너
AnyLink	—	1	0.12%	네트워크 / 원격 접속
Apache2 Ubuntu Default Page: It works	Apache2 Ubuntu 기본 페이지	1	0.12%	웹 서버
Cairn	—	1	0.12%	기타
ChatGPT充值_ChatGPT Plus/Pro代充_微信支付开通 - AI会员代充	ChatGPT Plus/Pro 대리 충전 및 위챗페이 결제 서비스	1	0.12%	AI / 상업 서비스
CloudPanel | Log In	CloudPanel 로그인	1	0.12%	서버 관리
Cpolar	—	1	0.12%	터널링 / 네트워크
Default Site	기본 사이트	1	0.12%	웹 서버
DesRedTeam	—	1	0.12%	레드팀
HTTP Server Test Page	HTTP 서버 테스트 페이지	1	0.12%	웹 서버
KSpeeder | Docker镜像加速专家	KSpeeder | Docker 이미지 가속 도구	1	0.12%	인프라
KSubdomain	—	1	0.12%	서브도메인 탐색
LunaFox - Attack Surface Management Platform | ASM	LunaFox - 공격 표면 관리 플랫폼	1	0.12%	공격 표면 관리

웹 서비스 / 페이지 제목	한국어 의미	확인 건수	비율	분류
MinIO Console	—	1	0.12%	스토리지 관리
MultiC2 Console	—	1	0.12%	C2
Mythic	—	1	0.12%	C2 / 레드팀
Nginx Proxy Manager	—	1	0.12%	인프라 관리
OpenClaw Control	—	1	0.12%	관리 콘솔
RustFS	—	1	0.12%	스토리지
SafeSkill Demo · Mock Receiver	SafeSkill 데모 · Mock Receiver	1	0.12%	개발 / 테스트
SecureFlow - 安全信息流	SecureFlow - 보안 정보 스트림	1	0.12%	보안
TCNet · Linux Agent	—	1	0.12%	에이전트 / 관리
TSCI PRESENTS Cooper Park TEACHING NET	—	1	0.12%	일반
Uptime Kuma	—	1	0.12%	모니터링
WeChat Download API	위챗 다운로드 API	1	0.12%	API
WorkBuddy Manager	—	1	0.12%	관리
YUTUCMS 数据更新中	YUTUCMS 데이터 업데이트 중	1	0.12%	일반 웹 서비스
外卖配送平台	음식 배달 플랫폼	1	0.12%	일반 서비스
学无止境	배움에는 끝이 없다	1	0.12%	일반
安全知识大闯关，答题抽奖等你来	보안 지식 퀴즈 및 경품 이벤트	1	0.12%	보안 / 일반
恭喜，站点创建成功！	축하합니다. 사이트 생성에 성공했습니다	1	0.12%	기본 웹 페이지
授权管理系统	권한 관리 시스템	1	0.12%	관리
星河信息技术有限公司	Xinghe 정보기술 유한회사	1	0.12%	기업 사이트
暂时无法访问	일시적으로 접근할 수 없음	1	0.12%	일반
概览 · grokcli-2api	개요 · grokcli-2api	1	0.12%	AI / API
次数查询	횟수 조회	1	0.12%	일반
爆破专家 · 工序闭环整合原型	브루트포스 전문가 · 작업 프로세스 통합 프로토타입 v0.2(모듈형)	1	0.12%	공격 도구

웹 서비스 / 페이지 제목	한국어 의미	확인 건수	비율	분류
v0.2 (模块化)				
玄盾 · 安全态势感知平台	Xuandun · 보안 상황 인식 플랫폼	1	0.12%	보안 플랫폼
登录 - 安全分析平台	로그인 - 보안 분석 플랫폼	1	0.12%	보안 분석
登录 · DeepQuant Agent	로그인 · DeepQuant Agent	1	0.12%	AI / 에이전트
登录 · 水滴资产监控平台	로그인 · Waterdrop 자산 모니터링 플랫폼	1	0.12%	자산 모니터링
福建商学院二课自动管理系统	푸젠상학원 비교과 활동 자동 관리 시스템	1	0.12%	교육 / 일반
精品视频在线	고품질 영상 온라인	1	0.12%	일반
闽江大学二课自动管理系统	민장대학교 비교과 활동 자동 관리 시스템	1	0.12%	교육 / 일반
魔方简历 - AI 驱动简历编辑器	Mofang Resume - AI 기반 이력서 편집기	1	0.12%	AI / 일반
애널리틱스 | 글로벌 마켓 보드	—	1	0.12%	일반
애널리틱스 오늘의 행운자 | ANALYTICS	—	1	0.12%	일반

[표 1] ARTEX 관련 인프라에서 확인된 웹 서비스

C. ARTEX v2

분석 과정에서 인터넷에 노출된 ARTEX v2 인스턴스도 확인하였다. ARTEX v2는 기존 콘솔보다 구조화된 다중 Agent 실행 환경을 제공하며, 인터페이스에서는 ARTEX를 "Agent Security Workbench"로 소개하고 있다.

확인된 인스턴스에서는 Planner가 다수의 Worker Agent를 조율하고 있었으며, 각 Agent별 실행 상태와 동작 시간 등 세부 정보를 확인할 수 있었다. 또한 탐색 경로, 식별 자산, 증거 및 트래픽 수집, 도구 실행, 공유 Workspace 데이터, 보고서, MCP 연동, 재사용 가능한 Skills, Agent 설정 등 다양한 기능이 별도 구성 요소로 제공되고 있었다.

해당 시스템에서 확인된 완료 상태의 테스트 작업에서는 중앙 Planner가 여러 Worker Agent를 동시에 관리하는 형태가 확인되었다. 이를 통해 ARTEX v2가 단순한 자율 침투 테스트 실행 도구를 넘어, 자율 보안 테스트 과정을 통합적으로 조율하고 기록·관리하기 위한 플랫폼 형태로 발전하고 있는 것으로 판단된다.



[그림 2] ARTEX v2

3. 노출된 운영 디렉터리에서 확인된 공격 정황

*기준일: 2026-10-09

인터넷에 노출된 운영 디렉터리 2건을 분석하였다. 두 사례 모두 실제 시스템 또는 실제 환경과 유사하게 구성된 시스템을 대상으로 한 공격 활동과 부합하는 정황이 확인되었다. Case Study #1은 ARTEX 작업 디렉터리로, 웹쉘 제어, Reverse RDP 터널링, Pass-the-Hash 명령, PowerShell 기반 C2 통신, 특정 내부 시스템을 대상으로 한 탐색 자료가 포함되어 있다. Case Study #2에서는 내부망 스캔 결과, root 계정 SSH 접속 및 JVM Heap Dump 내 시크릿 추출을 위한 스크립트, Java 다운로더, Reverse SOCKS 워커, 다수의 Linux 및 Windows 임플란트가 확인되었다.

본 분석에 제공된 수집 정보에 따르면, 두 오픈 디렉터리는 모두 ARTEX가 배포되어 있거나 ARTEX 관련 콘텐츠가 노출된 인터넷 접근 가능 호스트에서 식별되었다. 다만 이러한 동시 노출은 탐지 및 인프라 관점의 연관성에 한정된다. ARTEX가 모든 산출물을 생성했다거나, 두 디렉터리가 동일한 운영자에 의해 관리되었다거나, 해당 활동이 ARTEX 개발자와 직접 연계된다는 근거로 볼 수 없다.

두 사례에서는 다음과 같은 공통 운영 방식이 확인되었다.

- 외부에 노출된 컨트롤러 인프라를 스테이징 및 C2 용도로 활용함
- 침해되었거나 접근 가능한 시스템을 내부망 피벗 거점으로 활용함
- 스크립트를 이용해 접근, 내부 탐색, 페이로드 전송 및 후속 행위를 자동화함
- 단일 통신 채널에 의존하지 않고 여러 프로토콜과 원격 접근 기법을 함께 활용함
- 운영 파일, 자격증명 및 내부 인프라 정보가 인터넷에서 접근 가능한 시스템에 그대로 노출됨

Case Study #1에서는 Reverse RDP 릴레이가 실제 실행되었을 가능성을 중간~높은 신뢰도로 평가하였으며, Pass-the-Hash를 이용한 RDP 접속 시도는 높은 신뢰도로 확인된다. 반면 이후 사용된 PowerShell 명령 에이전트가 정상적인 콜백 채널을 수립했는지 여부는 근거가 제한적이다. Case Study #2에서는 운영자가 Kubernetes Pod 네트워크와 별도의 내부 네트워크를 탐색하고, 노출된 데이터 서비스와 Spring Heap Dump 엔드포인트를 식별한 정황이 높은 신뢰도로 확인되었다. 또한 자동화 스크립트에 root 계정용으로 추정되는 SSH 자격증명이 포함되어 있었으며, 지속적인 접근을 위해 다수의 임플란트와 Reverse Proxy 도구를 준비한 정황도 확인되었다. 다만 현재 확보된 자료만으로는 Case Study #2에 포함된 모든 스크립트와 바이너리가 실제 실행되었다고 판단하기 어렵다.

중국 국가기관, 특정 침해 그룹 또는 ARTEX 개발자에게 해당 활동을 귀속할 수 있는 근거는 확인되지 않았다. ARTEX는 공개적으로 이용 가능한 도구이며, 중국어 기반 프로젝트 출처, 현지화된 메타데이터 또는 중국에 위치한 개발 인프라 사용 여부만으로 운영자의 국적이나 위치를 판단하기 어렵다. 현재 확

보된 자료를 종합하면 두 사례 모두 공격 목적의 보안 활동과 연관된 것으로 판단된다. 다만 이를 특정 국가, 조직 또는 ARTEX 개발자에게 귀속할 수 있는 근거는 확인되지 않았다.

A. Case Study #1

Case Study #1은 ARTEX 작업 디렉터리에서 확인된 37개 파일로 구성된다. 자료에는 CS1-WEB에서 동작한 것으로 추정되는 ASPX 웹쉘, 내부 RDP로 연결되는 Reverse TCP 릴레이, Administrator 계정 자격증명을 이용한 Pass-the-Hash 명령, PowerShell 에이전트 배포를 위한 GUI 자동화, 3개 사설 IP 시스템을 대상으로 한 내부 탐색 정황이 포함되어 있다.

가장 명확한 출처 식별 정보는 driver.sh와 run_agent.sh에 포함된 ARTEX 작업 경로 /root/ARTEX/data/tasks/6/i3539이다.

A.1 공격 흐름 재구성

```
CS1-C2
  ↓ PowerShell 릴레이 스테이징 및 운영자 측 리스너 노출
CS1-WEB ASPX 명령 실행 경로 추정
  ↓ Reverse TCP 릴레이: 외부 4443 -> 내부 RDP 3389
운영자 로컬 루프백 RDP 리스너
  ↓ Administrator 계정으로 xfreerdp Pass-the-Hash 시도
CS1-RDP
   | C:\Windows\Temp\a.ps1 다운로드 시도
  ↓ TCP 4470 또는 4471을 이용한 Raw TCP 콜백 시도
인증 없는 릴레이 -> 로컬 명령 클라이언트
```

A.2 웹쉘 명령 채널 추정

ws.sh는 자체적으로 웹쉘 명령 실행 스크립트임을 나타내며, CS1-WEB의 고정 ASPX 경로로 Base64 인코딩된 명령을 전송한다. 명령은 일정한 형식의 폼 파라미터에 삽입되며, 요청에는 고정 쿠키가 함께 사용된다. psrun.sh는 PowerShell 명령을 UTF-16LE로 인코딩한 뒤 -EncodedCommand 형태로 추가 래핑한다.

이는 사전에 구성된 명령 실행 경로가 존재했음을 강하게 뒷받침한다. 다만 ASPX 소스 코드, 응답 본문, 대상 IIS 로그, 대상 파일시스템 자료는 확보되지 않았다. 정상 서비스로 보이는 상위 도메인은 악성 인프라가 아니라 피해 가능성이 있는 자산으로 취급해야 한다.

2026년 10월 6일 일반 브라우저 요청으로 해당 경로에 접근한 결과 HTTP 404가 반환되었다. 이는

당시 일반 GET 요청으로 해당 리소스를 조회할 수 없었다는 사실만 확인해준다. 과거 클라이언트는 쿠키가 포함된 POST 요청과 명령이 삽입된 폼 파라미터를 사용하였다. 해당 요청은 코드 실행 또는 증거 훼손 가능성이 있어 재현하지 않았다. 따라서 리소스 삭제 또는 조치 완료, 조건부 라우팅, HTTP 메서드에 따른 동작 차이 가능성은 배제할 수 없다.

A.3 Reverse RDP 피벗

pivot_relay.ps1은 TCP 4443을 통해 CS1-C2로 지속적으로 외부 연결을 시도하고, 내부에서는 TCP 3389의 CS1-RDP에 연결한 뒤 양방향 스트림을 중계한다. enc.txt를 디코딩한 결과 동일한 스크립트가 확인되었다. relay.log에도 운영자 측에서 피벗 연결을 로컬 포트로 매핑한 기록이 남아 있다.

가장 이른 시점의 HTTP 기록을 종합하면 CS1-PIVOT이 약 9분 이내에 Reverse Relay, 런처, 종료 스크립트를 순차적으로 내려받은 정황이 확인된다. listener.log에도 동일한 출처 식별 정보에서 발생한 연결이 기록되어 있다. 이를 근거로 CS1-PIVOT에서 직접 실행되었거나, NAT 환경에서 CS1-PIVOT 뒤에 위치한 시스템에서 실행되었을 가능성을 중간~높은 신뢰도로 평가하였다. 다만 CS1-PIVOT과 CS1-WEB이 동일한 시스템이라는 점까지 입증되지는 않는다.

A.4 Pass-the-Hash 기반 RDP 접속 시도

driver.sh와 run_agent.sh는 로컬 포워딩 포트를 대상으로 Administrator 계정과 /pth:<NTLM hash> 옵션을 사용해 xfreerdp를 실행한다. 수정된 Impacket 기반 점검 스크립트 역시 NTLM 해시와 로컬 RDP 포워딩 포트를 입력값으로 사용하도록 구성되어 있다.

실행 흔적은 일부 상충한다. driver.log에서는 RDP 프로세스 실행과 FreeRDP 창 식별이 확인되지만, rdp_gui.log에는 RPC를 통한 연결 종료 기록이 남아 있다. 이후 드라이버는 콜백 수신 전에 프로세스가 종료되었으며 재시도 역시 실패했다고 기록한다. 따라서 현재 자료는 Pass-the-Hash 기반 RDP 접속 준비 및 실제 시도가 이루어졌음을 입증하지만, 안정적인 인증 세션이 유지되었다고 판단할 근거는 부족하다.

NTLM 관련 값은 본 보고서에서 의도적으로 제외하였다. 해당 자격증명은 침해된 것으로 간주하고 재사용 여부를 추가 점검할 필요가 있다.

A.5 PowerShell 명령 에이전트

agent.ps1은 연결 실패 시 재접속하도록 설계된 PowerShell 기반 명령 에이전트이다. 주요 동작은 다음과 같다.

1. TCP 8088을 통해 CS1-C2로 부팅 마커 전송

2. TCP 4471을 이용해 Raw TCP 세션 수립
3. 길이 정보가 포함된 PowerShell 명령 수신
4. Execution Policy 우회 옵션을 적용해 명령 실행
5. 길이 정보가 포함된 실행 결과 반환
6. 연결 실패 또는 종료 이후 재접속

agent_b64.txt를 디코딩한 변종은 TCP 4470을 사용한다. 이러한 포트 차이는 추가 헌팅 시 활용 가능한 식별 요소이다. relay.py는 모든 인터페이스에서 인증 없이 접근 가능한 피벗 및 로컬 리스너를 개방하며, c2.py는 이에 대응하는 운영자 측 명령 클라이언트를 구현한다.

driver.sh는 RDP GUI 자동화를 통해 다운로드 명령을 직접 입력하고, 에이전트를 C:\Windows\Temp\a.ps1 경로에 저장하려고 시도한다. HTTP 로그에서는 로컬 루프백 환경의 테스트성 다운로드 흔적이 확인되지만, 실제 대상 시스템에서 발생한 에이전트 다운로드 또는 콜백을 명확히 구분할 수 있는 증거는 확인되지 않았다.

A.6 내부망 탐색

recon1.ps1은 로컬 IPv4 인터페이스 정보를 수집한 뒤 CS1-RDP와 추가 내부 시스템 2대를 대상으로 웹, SMB, RPC, SQL Server, RDP, WinRM 및 애플리케이션 관련 주요 포트를 점검한다.

확보된 자료에는 사전에 설정된 대상 정보만 포함되어 있으며 실제 스캔 응답 결과는 확인되지 않았다. 따라서 해당 포트 또는 서비스가 실제로 열려 있었다는 점까지는 입증되지 않는다.

B. Case Study #2

Case Study #2에서는 현재 확보된 파일 14개가 확인되었다. 네이티브 실행 파일 4개, Java 클래스 1개, Python 기반 Reverse Proxy 워커 2개, 스캔 결과 파일 2개, 캐시된 Python 프로그램 2개, 캐시 메타데이터 파일 3개로 구성된다. 이 가운데 메타데이터 1건은 추가 캐시 데이터 객체를 참조하지만, 해당 파일은 현재 자료에 포함되어 있지 않다.

전체 자료는 하나의 일관된 운영 흐름을 구성한다. 스캔 결과에서는 Kubernetes Pod 네트워크와 별도의 사설 네트워크에 대한 내부망 탐색이 확인되었다. 캐시된 스크립트 중 하나는 CS2-K8S-MASTER에 root 계정으로 SSH 접속을 시도하며, Docker 구성요소로 위장한 바이너리를 재실행하도록 작성되어 있다. 또 다른 스크립트는 JVM 힙 덤프에서 애플리케이션 자격증명과 암호화 관련 정보를 추출한다. Java 클래스는 CS2-C2에서 Linux 임플란트를 내려받으며, Linux 및 Windows 바이너리에서도 동일한 C2와 호환되는 기능 명칭 및 프로토콜 문자열이 확인된다.

B.1 네트워크 및 서비스 탐색

result.txt에는 CS2-POD-NET에 대한 스캔 결과가 기록되어 있으며, 총 85개 시스템이 활성 상태로 식별되었다. 확인된 서비스에는 SSH, HTTP, MySQL, Redis, kubelet, MongoDB, Memcached, ActiveMQ, QuestDB, go-mysql-transfer 및 다수의 Java/Spring 애플리케이션 포트가 포함된다. 자동화된 탐지 결과에서는 다음 항목이 확인되었다.

- 인증 없이 접근 가능한 MongoDB 인스턴스
- 인증 없이 접근 가능한 Memcached 인스턴스
- 기본 자격증명으로 접속 가능한 ActiveMQ 관리 콘솔
- 외부에 노출된 다수의 Spring Actuator Heap Dump 엔드포인트

r1esult.txt에는 CS2-SECONDARY-NET에서 식별된 8개 활성 시스템이 기록되어 있다. 식별된 시스템 전반에서 SSH가 확인되었고, 일부 노드에서는 kubelet이 노출되어 있었다. CS2-K8S-MASTER에서는 etcd와 go-mysql-transfer도 함께 확인되었다.

다만 두 파일은 스캐너 출력 결과이며, 개별 서비스의 실제 접근 가능 여부를 별도로 검증한 자료는 아니다. 그럼에도 스캔 범위와 결과의 일관성을 고려할 때, Kubernetes 환경과 애플리케이션 계층의 자격증명 노출 가능성을 중심으로 의도적인 내부망 탐색이 수행된 것으로 판단된다.

B.2 Root SSH 접속 및 위장 임플란트 유지

메타데이터에서 fix_master.py로 식별된 캐시 파일은 pseudo-terminal을 이용해 CS2-K8S-MASTER에 root 계정으로 SSH 접속한다. SSH 호스트 키 검증을 비활성화하고 비밀번호 입력 프롬프트를 대기한 뒤, 하드코딩된 비밀번호를 제출하도록 작성되어 있다.

원격에서 실행되는 명령은 다음 동작을 수행한다.

- 특정 PID가 가리키는 실행 파일 확인
- docker-proxy-helper 프로세스 검색
- 기존 프로세스 종료
- /usr/libexec/docker/docker-proxy-helper를 nohup으로 분리 실행
- 프로세스 목록을 통해 재실행 여부 확인

이는 운영자가 root 계정에 사용할 수 있는 자격증명 후보를 확보하고 있었으며, Docker 인프라 구성요소로 위장한 비인가 또는 악성 바이너리가 해당 경로에서 실행 중일 것으로 전제했음을 강하게 뒷받침한다. PID와 파일 경로가 구체적으로 지정된 점은 사전 접근 또는 사전 관찰이 있었을 가능성을 높인다.

다만 실제 SSH 실행 결과가 남아 있지 않아 인증 성공 및 프로세스 재실행 성공 여부까지는 확인할 수 없다.

비밀번호 값은 본 보고서에서 제외하였다. 동일 자격증명이 다른 시스템에서 재사용되었을 가능성을 고려해 침해된 자격증명으로 취급할 필요가 있다.

B.3 JVM heap credential mining

메타데이터에서 extract_hprof.py로 식별된 캐시 파일은 /tmp/reg.hprof를 열어 출력 가능한 문자열을 순차적으로 추출한다. 이후 Redis, 데이터소스, RabbitMQ, Access Key, Secret Key, AES 관련 값, Private Key 등과 연관된 설정 키 및 비밀번호 형태의 값을 검색하고, 각 항목의 탐지 횟수와 주변 문자열을 출력한다.

대상 파일 경로는 스캔 결과에서 다수 확인된 Spring Actuator Heap Dump 엔드포인트와 연관성이 높다. 두 자료를 종합하면 노출된 Heap Dump 엔드포인트 식별, Heap Dump 파일 확보, 임시 경로 저장, 자동화된 시크릿 추출로 이어지는 흐름이 구성되어 있었을 가능성이 높다.

다만 현재 확보된 자료에는 실제 Heap Dump 파일이나 추출 결과가 포함되어 있지 않다. 따라서 자격증명 또는 암호화 정보를 실제로 확보했는지는 확인되지 않았다.

B.4 Java 다운로드

P.class는 Java 8 클래스이며, static initializer에서 /bin/sh -c를 호출한다. 포함된 명령은 CS2-C2의 TCP 3000 포트에서 ljy-bank-channel을 /tmp/.gw로 내려받은 뒤 실행 권한을 부여하고, setsid를 이용해 입출력을 차단한 상태로 분리 실행한다.
해당 동작이 static initializer에 포함되어 있어 클래스가 초기화되는 시점에 명령이 실행되도록 설계되어 있다. 이는 다운로더의 동작 방식과 C2 인프라 연계를 입증하지만, 실제 취약한 Java 애플리케이션에서 해당 클래스가 초기화되었는지는 확인되지 않았다.

B.5 Rust 임플란트

ljy-bank-channel과 tcp_linux_amd64는 동일한 크기의 x86-64 Linux PIE 실행 파일이다. 두 파일은 하나의 연속된 영역에서 정확히 128바이트만 다르며, C2 주소, TCP 6666 포트, 키 형식, 프로토콜 관련 문자열, 라이브러리, 기능 문자열은 대부분 동일하다. 변경된 영역은 빌드별 설정값 또는 암호화 관련 데이터일 가능성이 있으나, 정확한 용도는 확인되지 않았다.

정적 문자열에서는 다음과 같은 광범위한 임플란트 기능이 확인된다.

- 명령 및 스크립트 실행
- 파일 업로드 및 다운로드
- 프로세스 목록 조회 및 종료
- 파일 복사, 이동 및 삭제
- 자기 삭제
- SOCKS5 Proxy, 포워딩, Hop, 터널 기능
- 스크린샷 및 화면 해상도 수집
- 시스템 정보, 업타임, CPU, 마우스, 가상화 환경 정보 수집
- 암호화/TLS 기반 통신 및 Chunked Transfer
- ELF 페이로드 실행

Rust 빌드 경로에서는 중국 소재 대학이 운영하는 crates 미러가 참조된다. 이는 빌드 환경에서 생성된 흔적일 뿐이며, 운영자의 국적이나 위치를 판단할 근거로 사용할 수 없다.

B.6 지역 및 언어 설정 유틸리티로 위장한 Windows 임플란트

LocaleMgr.exe는 서명되지 않은 x86-64 Windows PE 파일이다. 중국어 메타데이터에서는 시스템 지역, 언어, 형식 설정을 관리하는 유틸리티로 위장하고 있다. 그러나 실제 파일 내용은 이러한 설명과 일치하지 않는다. Rust 기반 Linux 임플란트와 동일한 CS2-C2, TCP 6666 설정 및 키 데이터가 포함되어 있으며, tcp_implant.pdb 경로도 확인된다.

정적 문자열에서는 Linux 임플란트와 공통된 기능을 Windows 환경에 맞게 구현한 흔적이 확인된다. 주요 기능은 다음과 같다.

- 명령 실행
- Execution Policy 우회를 적용한 비대화형 PowerShell 실행
- 임시 스크립트 정리
- 파일 업로드 및 다운로드
- 인라인 PE 또는 Shellcode 실행
- 프로세스 및 모듈 조작
- Named Pipe 사용
- 스크린샷 및 입력 제어
- 시스템 및 BIOS 기반 가상화 환경 점검
- Impersonation
- 서비스, 프로세스 및 파일 조작
- SOCKS 및 터널 기능

- 자기 삭제

일부 문자열에서는 프로세스 메모리 조작 및 Remote Thread 관련 API가 확인된다. 이는 해당 기능이 구현되어 있거나 관련 코드가 링크되어 있음을 보여주는 정적 증거이며, 이번 사례에서 실제 프로세스 인젝션이 수행되었다는 의미는 아니다.

B.7 Go 기반 Linux 임플란트

linux1은 x86-64 Go 실행 파일이며 앞서 확인된 Rust 계열과는 별개의 임플란트이다. CS2-C2의 TCP 4444 포트와 별도의 장문 키를 포함하고 있다. 정적 문자열에서는 Shell 실행, WebSocket/TLS 통신, SOCKS Proxy, X11 및 Display 관련 기능을 이용한 스크린샷 수집, 시스템 정보 수집, 파일 생성, 민감정보 또는 자격증명 패턴 검색 기능이 확인된다.

특징적인 문자열, 파일 해시, 설정값을 기준으로 공개 자료와 비교했으나 신뢰할 수 있는 기존 악성코드 패밀리 식별 정보는 확인되지 않았다. 따라서 현재로서는 커스텀 또는 비공개 형태로 제작된 Go 기반 Linux 임플란트로 분류하는 것이 가장 타당하다.

B.8 Reverse multiplexed SOCKS 워커

mux.py와 mux2.py는 Python 기반 Reverse Proxy 워커이다. 두 스크립트 모두 CS2-C2로 외부 연결한 뒤 WK 마커를 전송하며, 명령, 연결 ID, 페이로드 길이로 구성된 5바이트 !BHH 형식의 커스텀 프레임을 사용한다. 또한 임의의 IPv4 주소 또는 도메인을 대상으로 SOCKS CONNECT 요청을 처리할 수 있다.

첫 번째 워커는 TCP 1081 포트를 사용한다. 두 번째 워커는 기본적으로 TCP 1080을 사용하며 TCP Keepalive, Heartbeat, 자동 재접속 기능이 추가되어 있다.

해당 스크립트는 내부 시스템을 운영자가 지속적으로 제어할 수 있는 SOCKS 피벗 노드로 전환하는 용도로 사용할 수 있다.

4. Copy Cat 등장 - 파생 배포본과 REAPER 사례

*기준일: 2026-10-14

ARTEX 메인 깃허브 저장소가 사라진 이후, 기존 ARTEX 코드베이스를 기반으로 독립적으로 운영되거나 별도로 커스터마이징된 배포 사례가 점차 증가하는 양상을 확인하였다. 이들 가운데 일부는 ARTEX 관련 흔적을 그대로 유지하면서도 제품명, 사용자 인터페이스, 파일시스템 구조, 주변 기능 등을 변경한 형태로 운영되고 있다. 이에 따라 원본 저장소가 더 이상 제공되지 않는 상황에서 ARTEX가 어떤 방식으로 재사용·변형·재배포되고 있는지 지속적으로 추적하고 분석하였다.

이 가운데 비교적 주목할 만한 사례 중 하나는 러시아어 사용자가 커스터마이징한 것으로 추정되는 인스턴스이다. 해당 배포본은 ARTEX라는 기존 명칭 대신 별도의 브랜드를 사용했으며, 러시아어 기반 제품 용어와 자체적인 런타임 구조를 적용하고 있었다. 반면 내부 아티팩트에서는 ARTEX 구성요소를 참조하는 흔적이 계속 확인되었다. 이를 종합하면 원본 ARTEX를 단순히 그대로 배포한 형태라기보다, 기존 ARTEX 코드베이스를 기반으로 별도 수정 또는 포크한 구현체일 가능성이 높다.

동일 환경에서 노출된 소스 코드 아티팩트를 추가 분석한 결과, 백엔드 코드 내부에 상당수의 한국어 문자열이 포함된 사실도 확인되었다. 해당 문자열은 SMTP 기능, 알림 처리, HTTP 요청을 담당하는 Go 소스 파일에 포함되어 있었으며, SMTP 인증 및 TLS 오류 메시지, 요청 및 리다이렉트 처리 메시지, 알림 심각도와 상태 값을 나타내는 한국어 표현 등이 확인되었다.

이러한 한국어 문자열만으로 운영자 또는 개발자가 한국인이라고 판단할 수는 없다. 기존 코드 재사용, 현지화 작업, 또는 다른 프로젝트에서 가져온 구성요소가 포함되었을 가능성도 충분하다. 다만 러시아어 기반 브랜딩과 함께 이러한 다국어 코드 흔적이 확인된다는 점은, ARTEX 파생 배포본이 단순한 명칭 변경 수준을 넘어 기능과 코드 구조 자체가 수정·재구성되고 있음을 보여준다.

A. Copy Cat 탄생

일부 인스턴스는 언어만 부분적으로 변경한 채 ARTEX 명칭을 그대로 유지하고 있었으며, 다른 사례에서는 ScopeWeaver, Tangora PTaaS, Horda, Meze, AegisLoop, Bob Research Labs 등 완전히 다른 제품명을 사용하고 있었다.

확인된 타이틀을 종합하면 ARTEX는 원본 프로젝트에만 한정되어 사용되는 것이 아니라, 여러 독립 운영자에 의해 재배포되고 각 환경에 맞게 수정되고 있는 것으로 판단된다. 일부 배포본은 기존과 마찬가지로 자율형 모의해킹 기능을 전면에 내세우고 있는 반면, 다른 인스턴스는 동일하거나 유사한 기반 플

랫폼을 보안 점검, 코드 감사, 범용 에이전트 시스템 등의 형태로 재구성하고 있었다.

아래는 ARTEX 파생 또는 ARTEX 기반으로 추정되는 배포 인스턴스에서 확인된 주요 이름들이다.

ARTEX — 自主渗透测试控制台
ARTEX — Autonomous penetration testing console
Bob Research Labs: автономная система исследования безопасности
AegisLoop — 攻防安全运营
jins 시스템 — 취약점 점검 시스템
Autonomous Pentest Agent — 自主渗透测试控制台
ARTEX — Autonomous security assessment console
ARTEX — Autonomous Pentest Console
CodeAudit — 自主代码审计平台
AgentHub — 通用智能体平台
索隐（SUOYIN）
ARTEX: 자율 침투 테스트 콘솔
ScopeWeaver — Autonomous Penetration Testing Console
ARTEX — Autonomous Penetration Testing Console
Ayven
LOMCS — 自主安全检测控制台
ARTEX — 自主滲透測試控制台
EXMAICAI
Tangora PTaaS — Autonomous Penetration Testing Console
Horda — Autonomous Penetration Testing Console
ARTEX — 자율 침투 테스트 콘솔
ARTEX: Autonomous Penetration Testing Console
ARTEX: Autonomous penetration testing console
Bob Research Labs: We're coming to root your server!
Meze — 자율 침투 테스트 콘솔
ARTEX — 自律型ペネトレーションテスト管理画面
安全渗透智能体
Algogen
Uchange — 自主渗透测试控制台
Catchya-ART — 自主渗透测试控制台
秋水 ARTEX — 自主渗透测试控制台

ARTEX — Console de teste de penetração autônomo
ARTEX — консоль автономного пентеста
GaSSpEr Ghost — Console de pentest autonome
WETEX
ARTEX — 自律型ペネトレーションテストコンソール
ARTEX : console de pentest autonome
ARTEX Autonomous Penetration Testing Console
ARTEX Plus: Offensive, Defensive & Closed-loop Security
ARTEX
ARTEX — Console de Teste de Penetração Autônomo
ARTEX — the autonomous penetration-testing console
ARTEX — Консоль автономного тестирования на проникновение
GPT-RACK
IR 排查控制台
RAIM: 자율 침투 테스트 콘솔
RedWolf — 对话式渗透测试控制台
鲸鱼红队渗透系统 — 红队渗透测试控制台

B. Case Study: "저승사자"



[그림 3] "저승사자" 로그인 화면

모니터링 과정에서 확인된 ARTEX 파생 배포본 가운데 비교적 주목할 만한 사례는 “THE GRIM REAPER라는 명칭을 사용한 인스턴스이다. 인터페이스에서는 운영 주체를 Bob Research Labs로 표기하고 있었다. TCP 3001 포트에 노출된 커스텀 인터페이스는 해당 플랫폼을 “Autonomous offensive security framework”로 소개하며, “We are coming to root your server.”라는 문구를 사용하고 있었다. 한편 TCP 443 포트의 별도 HTTPS 서비스에서는 기존 ARTEX 로그인 화면이 유지되고 있었으나, HTML 타이틀은 Bob Research Labs: We're coming to root your server!로 변경되어 있었다.
기존 ARTEX 인터페이스와 별도 브랜딩이 적용된 프론트엔드가 동시에 운영된 점을 고려하면, 원본 ARTEX를 단순 배포한 형태라기보다 기반 애플리케이션은 유지한 채 외부 인터페이스와 브랜딩을 별도로 커스터마이징한 것으로 판단된다.

이후 노출된 파일과 런타임 로그를 추가 분석한 결과, 해당 배포본이 ARTEX 코드베이스를 기반으로 구성되었다는 정황이 추가로 확인되었다. 내부 파일시스템 경로에서는 artex-linux-amd64 참조가 계속 확인되었으며, 커스터마이징된 환경에서는 프로젝트명을 reaper-full로 변경해 사용하고 있었다. 또한 ARTEX의 Worker 구조와 트래픽 기록용 프록시를 그대로 유지하면서, 별도의 브랜딩과 설정 구조를 추가한 형태로 구성되어 있었다.



[그림 4] Open Directory 내용 일부

B.1 Reaper Log

런타임 로그를 추가 분석한 결과, REAPER 배포본이 ARTEX의 기존 멀티 에이전트 오케스트레이션 구

조를 그대로 유지하고 있음이 확인되었다. 백엔드는 PostgreSQL에 저장된 작업 상태를 복원하고, 재시작 이후 미완료 Intent를 다시 할당했으며, 새 의존 관계가 생성될 때마다 Planner를 호출하면서 최대 3개의 Worker에 작업을 병렬 분배하였다. 완료된 Intent의 결과는 구조화된 Fact, Asset, 취약점 정보 형태로 다시 작업 상태에 기록되었다.

로그에서는 여러 LLM 프로파일이 실제로 사용된 정황도 확인되었다. DeepSeek, GLM, Kimi, Claude 기반 설정이 함께 구성되어 있었으며, 이는 해당 커스텀 플랫폼이 단일 모델 제공자에 의존하지 않고 여러 LLM 백엔드를 선택적으로 사용할 수 있도록 확장되었음을 보여준다. 또한 별도의 bobresearch 에이전트 프로파일도 확인되어, 운영자가 단순히 인터페이스와 제품명만 변경한 것이 아니라 ARTEX 내부 설정과 에이전트 구성까지 직접 수정한 것으로 판단된다.



[그림 5] 사용되는 모델

B.2 translate_go.py

추가로 확보된 소스 코드 아티팩트에서는 Bob Research Labs 운영자가 ARTEX 코드베이스를 직접 수정하고 현지화 작업을 진행한 정황이 확인되었다. translate_go.py라는 스크립트에는 인증, 자산 처리, 채팅 참조, 파일 업로드, 작업 제약 조건, 대화 관리 등 여러 백엔드 구성요소에 포함된 한국어 및 중국어 문자열을 영어로 치환하기 위한 하드코딩 매핑이 포함되어 있었다. 또한 Interactive Shell 사용, msfconsole, SSH, 데이터베이스 REPL, Reverse Shell 등 운영자용 ARTEX 지침도 영어로 변환하도록 구성되어 있었다.

이와 함께 artex-ko 디렉터리와 다수의 한국어 현지화 아티팩트가 확인된 점을 고려하면, 커스터마이징된 REAPER 빌드는 한국어로 현지화된 ARTEX 변형의 일부 구성요소를 기반으로 한 뒤 다시 영어화 작업이 이루어진 것으로 추정된다. 배포 환경에서 확인된 러시아어 브랜딩은 러시아어 사용자 또는 러시아어권 이용자를 대상으로 한 운영 가능성을 보여주지만, 현재 자료만으로 운영자의 국적을 판단할 수는 없다. 마찬가지로 영어 번역 작업은 영어권 피해자를 대상으로 한 정황이라기보다, 플랫폼을 다국어 환경에서 사용할 수 있도록 확장한 국제화 작업으로 해석하는 것이 더 타당하다.

B.3 CVE-2026-75650, CVSS 10.0 악용

Bob Research Labs 환경에는 CVE-2026-75650에 대한 공개 PoC인 StyleSmuggler의 실제 작업 사본이 노출되어 있었다. 해당 취약점은 인증 없이 악용 가능한 Magento/Adobe Commerce 원격 코드 실행 취약점이다. 로컬 디렉터리에는 exploit.py, curl_failed_payment.sh, failed-payment-request.json, Docker 관련 파일, .git 메타데이터 등 원본 저장소 구조가 그대로 유지되어 있었다. 실행 로그에서는 공개 PoC와 동일한 다단계 공격 절차가 여러 인터넷 노출 Magento 시스템을 대상으로 수행된 흔적이 확인되었다. 주요 단계에는 Guest Cart 생성, Billing Information 조작, 취약한 결제 처리 흐름 트리거, Magento Report Identifier 추출 등이 포함되어 있었다.

다만 로컬 환경에서 확인된 작업 흐름은 원본 PoC의 일반적인 RCE 검증 수준을 넘어선다. 한 실행 기록에서는 IRC BOT DEPLOYED 메시지가 출력되었고, cache.php라는 PHP 페이로드가 생성된 사실과 함께 새로 접속한 봇을 IRC에서 확인하라는 지침이 기록되어 있었다. 별도의 test_new_bot.php 파일에서는 GitHub 저장소 Bob-Marley-Backup/LAB-Uncomplete에서 irc.php 페이로드를 내려받은 뒤 즉시 실행하도록 구성된 코드도 확인되었다.

확보된 irc.php를 분석한 결과, 단순한 테스트용 마커가 아니라 실제 동작 가능한 지속성 기반 IRC C2 임플란트로 확인되었다. 해당 봇은 152.89.170.207:6667에 접속한 뒤 비밀번호가 설정된 #mg_master 채널에 참여하도록 구성되어 있었다. 또한 호스트별 닉네임을 생성하고, 침해된 도메인, 서버 IP, PHP 버전, 운영체제, 실행 사용자 등의 정보를 전송한다.

임플란트는 자신을 여러 은닉 경로에 복사하고, 주기적 실행과 @reboot 방식의 cron 항목을 생성해 지속성을 확보하도록 설계되어 있었다. 설치가 성공하면 최초 드롭 파일을 삭제하는 동작도 포함되어 있었다. 명령 실행은 proc_open, popen, shell_exec, exec, system, passthru 등 여러 PHP 실행 방식을 지원하며, 일부 함수가 제한된 환경에서도 동작 가능성을 높이도록 구현되어 있었다.

IRC 연결 이후에는 !cmd@<bot> <command> 및 !cmd@all <command> 형식으로 임의 명령을 수신하고, 실행 결과를 채널로 반환한다. 이외에도 !list, !info 명령을 통해 호스트 목록 조회와 시스템 정보 확인 기능을 제공한다.

이러한 아티팩트를 종합하면, Bob Research Labs 환경은 단순히 공개 CVE PoC를 보관한 수준에 그치지 않았다. StyleSmuggler 기반 RCE 이후 지속성을 갖는 IRC 제어형 PHP 임플란트를 배포·실행하는 후속 공격 흐름이 구성되어 있었던 것으로 판단된다. 다만 현재 자료만으로 이 작업 흐름이 REAPER/ARTEX에 의해 직접 실행된 것인지, 아니면 동일 시스템에서 별도로 수행된 공격 기능 개발

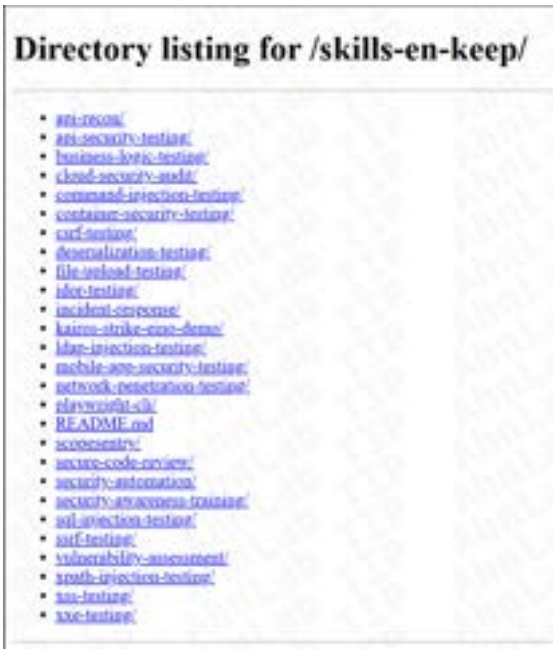
또는 실험이었는지는 확인되지 않았다.

B.4 Skills

Bob Research Labs 환경에서는 일반적인 ARTEX 배포본보다 크게 확장된 Agent Skills 컬렉션이 확인되었다. 해당 Skills는 에이전트가 필요에 따라 불러와 사용할 수 있는 재사용형 지침과 테스트 방법론으로 구성되며, 핵심 애플리케이션 자체를 수정하지 않고도 플랫폼이 수행할 수 있는 보안 작업 범위를 확장하는 역할을 한다.

확인된 Skills는 API 정찰 및 테스트, SQL Injection, SSRF, XSS, XXE, IDOR, Command Injection, 파일 업로드 테스트, 클라우드 및 컨테이너 보안, 모바일 애플리케이션 테스트, 네트워크 침투 테스트, 보안 코드 리뷰, 사고 대응, 취약점 점검 등 공격·방어 보안 전반을 폭넓게 다루고 있었다. api-recon, playwright-cli, scopesentry 등 일부 기본 ARTEX 구성요소는 그대로 유지되어 있었으나, 추가로 확인된 모듈 대부분은 표준 ARTEX Skills 세트에 포함되지 않는 것으로 확인되었다.

또한 skills-en-keep, skills-backup, skills-translated-en.zip, bob-research-skills-en.zip 등의 관련 아티팩트가 확인되어, 운영자가 커스터마이징된 환경에서 사용할 Skills를 지속적으로 수집·번역·선별하고 다시 패키징한 정황이 확인되었다. kairos-strike-eino-demo 패키지는 별도의 에이전트-Skill 연동 방식을 실험한 흔적으로 판단된다. 다만 현재 확보된 자료만으로 이들 모듈이 모두 Bob Research Labs에 의해 독자적으로 개발되었다고 보기는 어렵다. 전체 컬렉션은 공개된 보안 플레이북, 번역된 모듈, 로컬에서 수정된 구성요소가 혼합된 형태일 가능성이 높다.



[그림 6] Custom Skills

5. 추적 내용 요약 및 결론

ASEC은 인프라 피벗을 통해 ARTEX가 호스팅된 약 600개의 추가 IP 주소를 확인하였으며, 이 가운데 일부 시스템에서는 CyberStrikeAI도 함께 운영되고 있었다. 두 도구 모두 공개된 오픈소스 프로젝트로 서로 연관성이 없는 사용자가 자유롭게 구축할 수 있다. 따라서 확인된 인프라 전체를 단일 위협 행위자나 특정 캠페인이 운영하는 것으로 판단할 수 없다.

노출된 웹 서비스를 분석한 결과, 다수의 시스템에서 ARTEX와 함께 AI API 게이트웨이, 공격 표면 관리(ASM) 플랫폼, 침투 테스트 및 C2 프레임워크 등 다양한 애플리케이션이 운영되고 있었다. Sub2API가 높은 빈도로 확인되었으며, Scope Sentry, ARL-Next, LunaFox, CyberStrikeAI 등 보안 도구도 함께 식별되었다. 인프라 관리 플랫폼, 교육 서비스, 상용 애플리케이션 및 ARTEX와 직접적인 연관성이 확인되지 않은 일반 웹 서비스도 다수 식별되었다. 일부 시스템은 보안 연구자나 침투 테스트 수행자가 운영하는 환경일 수 있으며, 다목적 서버 또는 공유 인프라일 가능성도 존재한다.

인터넷에 노출된 두 운영 디렉터리 사례에서는 자격증명 사용, 내부망 탐색, 피벗, 페이로드 전송, C2 통신, 임플란트 운용 등 후속 공격과 관련된 자료가 확인되었다. 확보된 자료는 단순한 PoC 또는 기능 테스트보다 실제 시스템 또는 실제 환경과 유사하게 구성된 시스템을 대상으로 수행된 공격 활동에 부합하는 것으로 판단된다.

다만 해당 활동을 ARTEX 개발자에게 귀속하거나 두 사례를 하나의 캠페인으로 판단할 수 있는 근거는 확인되지 않았다. ARTEX와 운영 파일이 동일 시스템에서 함께 노출되었다는 사실은 인프라 및 수집 과정에서 확인된 연관성으로 한정해 해석할 필요가 있다. 현재 확보된 자료만으로는 동일 운영자 여부, 공통된 작전 목적, 동일 조직의 운영 여부를 판단하기 어렵다.

두 사례에서는 스크립트, 자격증명, 내부 IP 정보, C2 설정, 임플란트 바이너리, 후속 공격 도구 등이 인터넷에 노출된 시스템에 그대로 남아 있었다. 이러한 자료는 방어 관점에서 유용한 탐지 및 헌팅 정보를 제공하는 동시에, 공격 방식과 재사용 가능한 접근 정보를 제3자에게 노출하는 2차 위험도 발생시킨다.

원본 ARTEX 저장소가 더 이상 공개적으로 접근할 수 없게 된 이후에도 독립적으로 운영되거나 별도 브랜딩이 적용된 배포본이 점차 증가하는 양상이 확인되고 있다. 확인된 인스턴스는 한국어, 러시아어, 일본어, 프랑스어, 영어, 중국어 등 다양한 인터페이스를 사용하며, ARTEX의 주요 구성요소는 유지한 채 제품명, 언어, 에이전트 설정, 주변 기능 등을 수정한 형태가 다수 확인되었다.

Bob Research Labs 배포본은 이러한 사례 중 하나이다. 해당 환경에서는 커스터마이징된

ARTEX/REAPER 빌드와 함께 추가 Skills, 다수의 LLM 백엔드, 익스플로잇 개발 관련 아티팩트, CVE-2026-75650을 이용해 IRC 제어형 PHP 임플란트를 배포하는 작업 흐름이 확인되었다. 인터넷에서 접근 가능한 디렉터리를 통해 소스 코드, 자격증명, 개발 아티팩트, 런타임 로그, 공격 도구 등도 광범위하게 노출되어 있었다.

해당 배포본의 운영 방식은 OPSEC가 미흡하고 외부 노출을 크게 의식하지 않는 형태에 가깝다. 다만 이러한 특징만으로 운영자를 핵티비스트 또는 저숙련 공격자로 단정할 수는 없다. 공격적인 브랜딩과 반복적인 OPSEC 실패가 함께 확인된다는 점은 정교하게 은폐된 작전보다는 실험적 운영, 느슨하게 조직된 공격 활동, 또는 상대적으로 성숙도가 낮은 TTP와 더 부합하는 것으로 판단된다.

ARTEX는 LLM 기반 계획 수립, 재사용 가능한 보안 Skills, 자동화된 도구 실행, 멀티 에이전트 오케스트레이션을 결합한다. 이러한 프레임워크의 지속적인 발전과 공개는 공격 활동에 진입하기 위한 기술적 장벽을 낮출 가능성이 높은 것으로 판단된다. AI 기반 공격 보안 도구가 고도화된 침해 사고에 필요한 전문 지식, 인프라, 초기 접근 권한, 운영자의 판단을 완전히 대체하는 것은 아니지만, 공격 과정의 개별 단계에서 요구되는 수작업과 전문 지식의 상당 부분을 줄일 수 있다.

이러한 기능은 정상적인 보안 점검에도 활용될 수 있다. 그러나 공개된 코드가 복제·현지화·확장되고 공개 익스플로잇 및 후속 공격 도구와 결합될 경우 동일한 자동화 기능이 공격 목적으로 전환될 수 있다. 이에 따라 ARTEX와 같은 도구가 실제 공격 활동에 활용되는 사례를 지속적으로 관찰할 필요가 있다.

Part 2. ARTEX 침투테스트 결과 분석

ARTEX는 사용자가 목표와 허용 범위를 지정하면 여러 에이전트가 역할을 나누어 작업을 진행하고, 확인한 자산과 사실을 기반으로 다음 판단에 활용한다. 따라서 ARTEX를 이해하기 위해서는 개별 명령의 자동화 여부보다 에이전트가 어떤 정보를 수집 및 공유하고, 이를 바탕으로 다음 행동을 어떻게 결정하는지 살펴보는 것이 중요하다.

Part 2에서는 파일 업로드 취약점이 있는 웹 서버에서 웹 셸 업로드와 프로그램 실행을 확인하고, The DFIR Report가 공개한 Apache ActiveMQ 침해 사례의 초기 접근 조건을 모사한 환경에서 초기 침투와 내부망 이동을 재현한 내용을 정리한다.

1. ARTEX 구성도 및 주요 기능

A. 구성과 실행 흐름

ARTEX의 웹 UI 화면은 사용자가 자산과 작업을 등록하고, LLM 연결을 설정하며, 진행 과정과 결과를 확인하는 창구다. 서버는 작업과 에이전트의 실행 상태를 관리한다. 에이전트가 발견한 자산과 작업 과정은 PostgreSQL에 저장되고, Worker가 사용하는 도구와 대상 사이의 일부 통신은 기록용 프록시를 거쳐 추적된다. 공식 아키텍처는 웹 화면, 서버, 작업 엔진, 에이전트, 데이터베이스, 승인 및 트래픽 기록 기능으로 구성된다.



[그림 7] ARTEX 동작 흐름과 Agent 구성 (ChatGPT)

ARTEX 구조의 특징은 자산 그래프와 탐색 그래프를 분리해 관리한다는 점이다. 자산 그래프에는 도메인, IP 주소, 서비스, 애플리케이션, 엔드포인트 등 대상의 구조가 저장되며 여러 작업이 이를 공유한다. 탐색 그래프에는 특정 작업의 목표, 시도하려는 방향, 확인한 사실, 발견한 취약점이 연결되어 저장된다. 두 그래프의 연결 정보를 통해 특정 판단이 어느 자산과 관측 결과에 근거했는지 추적할 수 있다. 이 구조는 중복 탐색을 줄이고, 자산별 시험 범위와 진행 상황을 확인하는 데 도움이 된다.

B. 에이전트 역할군과 주요 기능

ARTEX의 에이전트 관리 화면에는 기본 구성으로 여덟 개의 에이전트가 등록된다. 에이전트는 역할과 수행 흐름에 따라 크게 세 가지 그룹으로 구분할 수 있다. 자동 침투 작업을 수행하는 그룹과 사용자 요청을 처리하는 그룹, 그리고 결과를 문서화하거나 재검증하는 그룹으로 구분된다.

역할군	에이전트	수행 업무
자율 작업 파이프라인	Goals, Planner, Worker	목표 설정부터 탐색 방향 결정, 도구 실행까지 이어지는 흐름 담당
사용자 개입 및 독립 실행	Mainagent, Auto, Pentest	사용자의 지시를 작업에 반영하거나 별도 대화에서 직접 작업을 수행
결과 문서화 및 재검증	Reporter, Retester	확인된 발견 사항을 보고서로 정리하고 수정 여부를 다시 확인

[표 2] 수행 흐름에 따른 역할군 분류

기본 프롬프트는 에이전트 관리 화면에서 수정할 수 있으며, 실행 시 작업 목표와 그래프 상태 같은 정보가 역할에 맞게 추가된다.

C. 자율 작업 파이프라인

ARTEX에 공격 대상과 목표가 전달되면 Goals, Planner, Worker가 자동 침투 흐름에 직접 참여한다. Goals는 최종 결과와 시험 범위를 정리하고, Planner는 다음 탐색 방향을 정하며, Worker는 도구를 실행하고 결과를 기록한다. Worker의 기록으로 그래프가 갱신되면 Planner가 다음 방향을 다시 판단한다. 각각의 에이전트의 기본 프롬프트와 동작 내용을 분석해보면 다음과 같다.

에이전트	목적	제약 조건	목표	분해 원칙
Goals	사용자 요청을 최종 목표와 시험 범위로 정리	명시된 조건만 반영하고 범위의 임의 확장 금지	최종 목표와 허용 및 금지 조건, 자산 범위 등록	독립적인 최종 결과만 분리하고 정찰과 검증은 목표에서 제외
Worker	배정된 시험 방향을	승인 범위 준수 및 재현	새 자산, 관측 사실, 재현	배정된 의도 하나에 집중하고 독립적인 결

에이전트	목적	제약 조건	목표	분해 원칙
	도구로 실행하고 응답 확인	증거 없는 취약점 보고 금지	된 취약점 기록	과만 분리
Planner	현재 탐색 결과에 따라 다음 시험 방향 결정	증거에 근거해 판단하고 진행 중인 작업과 중복 금지	목표 달성 판정 또는 새 탐색 의도 배정	독립 경로는 병렬, 선행 결과가 필요한 경로는 순차 배정

[표 3] 자율 작업 파이프라인 역할 Agent 프롬프트 분석

D. 사용자 개입 및 독립 실행

Mainagent는 현재 작업 안에서 사용자와 대화하며 진행 상태를 조회하고 지시를 작업에 반영한다. 사용자가 목표를 추가하거나 범위를 바꾸면 그 내용을 목표 또는 제약 조건으로 등록하고, 즉시 확인할 사항은 Worker가 수행할 의도로 전달한다. 즉, MainAgent는 사용자가 진행 중인 작업을 확인하거나 직접 지시를 전달하는 인터페이스 역할을 수행한다.

Auto는 대화 화면에서 작업 생성, 상태 조회, 일시 정지, Skill과 도구 관리 등 ARTEX의 운영 기능을 다룬다. 특정 침투 작업의 다음 공격 단계를 계획하는 역할은 아니며, 사용자의 플랫폼 관리 요청을 해당 기능에 연결한다.

Pentest의 경우 별도의 대화에서 한 에이전트가 탐색 계획, 도구 실행, 결과 검증을 이어 가는 독립형 시험 방식이다. Goals, Planner, Worker로 역할이 분리되는 자율 작업 흐름과 구별되며, 한 대화 안에서 시험 방향과 증거를 관리한다.

E. 결과 문서화 및 재검증

Reporter는 Worker가 등록한 발견 사항과 관련 증거를 읽어 재현 절차, 영향, 근거, 수정 권고를 담은 취약점 보고서를 작성한다. 보고서 작성은 기록된 발견 사항을 사람이 검토할 수 있는 결과물로 정리하는 단계이며, 새로운 시험 방향을 탐색하는 단계는 아니다.

Retester는 사용자가 특정 발견 사항의 재검증을 시작했을 때 별도 세션에서 기존 재현 조건과 현재 응답을 비교한다. 재현, 수정 확인, 확인 불가 중 하나의 결과와 판단 근거를 기록하며, 최초 침투 작업 전체를 다시 수행하지 않는다.

F. 기본 Skill과 도구 구성

ARTEX 저장소에는 세 가지 기본 Skill이 포함되어 있다. Skill은 에이전트가 작업을 수행할 때 참고하는

절차와 가이드 역할을 하며, 실제 대상 시스템과 직접 상호작용하는 기능은 포함하지 않는다. 웹의 Skill 화면에서는 내용과 사용 기록을 확인하고 에이전트별 가시성을 설정할 수 있다.

기본 Skill	제공하는 지침	적용 대상 및 조건
api-recon	프런트엔드 코드와 실행 중 요청을 분석해 API 경로, 메서드, 매개변수 파악	웹 애플리케이션의 API 표면 조사
playwright-cli	브라우저에서 페이지 요소를 확인하고 조작하는 명령의 사용법	브라우저 상호작용이 필요한 웹 대상
scopesentry	ScopeSentry MCP를 통한 자산 및 작업 조회 방법	별도 ScopeSentry 인스턴스와 MCP 연결 필요

[표 4] 기본 구성에 포함된 Skill

도구(工具)는 에이전트가 호출해 명령을 실행하거나 ARTEX의 기록을 조회하고 변경하는 기능이다. 웹의 도구(工具) 화면은 시스템 도구와 사용자 정의 도구를 다음과 같이 구분한다.

도구 유형	주요 기능
시스템 도구	ARTEX가 기본 제공하는 실행 및 기록 관리 기능
사용자 정의 shell	실행 환경에 있는 명령을 에이전트에 노출
사용자 정의 command	등록된 명령 템플릿 실행
사용자 정의 script	Python 코드 실행
사용자 정의 http	API 요청 전송

[표 5] 도구 유형 및 주요 기능

시스템 도구는 설명, 매개변수 기본값, 사용 여부와 에이전트 연결을 관리할 수 있다. 사용자 정의 도구는 입력 매개변수와 실행 내용을 설정하고 시험 실행으로 동작을 확인할 수 있다. v0.3.15에서는 MCP 도구와 사용자 정의 도구의 출력이 길어질 경우 세션별 출력 상한에 맞춰 모델에 전달하며, 별도 출력 경로가 있으면 전체 내용을 파일에 보존한다. 에이전트가 수행할 수 있는 행위는 프롬프트뿐 아니라 연결된 Skill과 도구의 범위에도 영향을 받는다.

이 밖에도 ARTEX는 ScopeSentry 자산 동기화, 도구 실행 승인 및 차단, 기록용 프록시를 제공한다.

ScopeSentry에서 가져온 자산은 작업 범위와 자산 그래프에 연결할 수 있다. 기록용 프록시로 수집한 트래픽은 v0.3.15 웹 화면에서 응답 내용과 요청 경로, 상태 코드, 응답 길이로 검색하거나 필터링할 수 있어 특정 시험 행위의 요청과 응답을 찾는 데 도움이 된다.

도구 실행 차단 기능에는 삭제 계열 API 경로를 식별하는 기본 규칙이 추가됐다. 이 규칙은 도구 입력에 /delete, /remove 등의 경로가 포함되면 HTTP 메서드가 GET 또는 POST여도 차단할 수 있으며, ARTEX 시스템 운영자는 명령 차단 화면에서 규칙을 변경하거나 비활성화할 수 있다. 규칙으로 판단되지 않은 도구 호출을 모델이 심사하도록 설정한 경우에는 그 심사에 사용된 토큰도 별도로 확인할 수 있다.

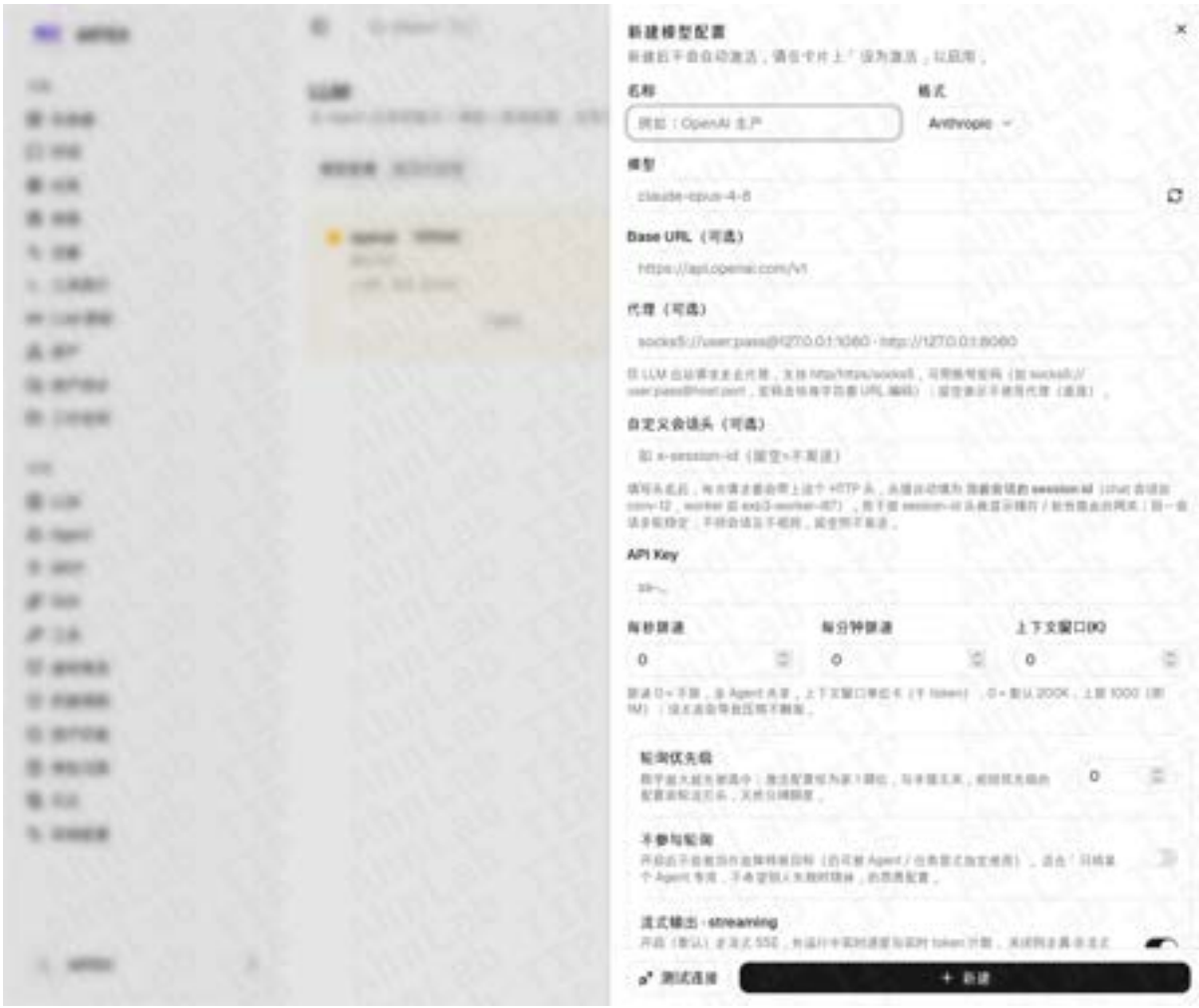
ARTEX는 실행 환경에서 선택한 도구를 호출한다. 대상에서 관측되는 네트워크 요청이나 프로세스 실행은 에이전트가 선택한 도구와 시험 목표에 따라 달라진다. 방어 측에서는 기능별 요청 특성과 연속된 탐색 흐름을 함께 분석한다.

2. ARTEX 침투테스트 설정

A. LLM 연결 프로필 설정

웹의 LLM 메뉴에서 새 연결 프로필을 만들 때는 프로필 이름과 API 형식, 모델 식별자, API 키를 입력한다. '格式'으로 표시되는 형식(API 형식) 목록을 확인해보면 Anthropic, OpenAI (Chat Completions), OpenAI (Responses API)를 확인할 수 있는데, 해당 항목은 ARTEX가 LLM 서비스와 주고받을 요청 및 응답의 형식을 정한다.

API 형식에 Anthropic 또는 OpenAI가 표시되더라도 반드시 해당 기업의 모델만 사용할 수 있는 것은 아니다. 사용하려는 모델의 API가 세 형식 중 하나와 호환된다면 모델 식별자와 Base URL을 직접 입력해 연결할 수 있다. 별도 중계 서버를 사용하는 경우에는 Base URL과 필요에 따라 LLM 요청용 프록시도 설정할 수 있다.



[그림 8] LLM 설정 프로필 화면

새 LLM을 연결할 경우 설정할 수 있는 프로필 화면으로, 가장 상단에 이름(名称)과 형식(格式)을 설정할 수 있으며, 그 아래로 실제 사용할 모델은 모델 식별자와 Base URL로 지정한다.

'사용자 지정 세션 헤더'에는 헤더의 값을 입력하는 대신 요청에 붙일 HTTP 헤더 이름을 입력한다. 예를 들어 OpenCode Zen처럼 x-opencode-session 헤더를 요구하는 연결에서는 이 이름을 입력한다. ARTEX는 LLM 요청마다 현재 대화 또는 Worker 실행의 세션 ID를 헤더 값으로 자동 입력한다. 같은 세션의 여러 요청에는 같은 ID가 사용되고, 다른 세션에서는 다른 ID가 사용되므로 세션별 캐시나 요청 경로를 관리하는 게이트웨이와 연결할 수 있다. 이 항목을 비워 두면 별도 세션 헤더를 보내지 않는다.



[그림 9] LLM 연결 테스트 및 모델 선택

형식(格式)과 Base URL 주소, API Key를 입력한 뒤 모델(模型) 칸 옆에 있는 새로고침을 버튼을 누를 경우 연결 테스트가 진행되며, 연결이 정상적으로 이뤄질 경우 사용 가능한 모델 목록을 확인할 수 있으며, 이 중 사용할 모델을 선택할 수 있다.

프로필에서는 분당 및 초당 요청 제한, 컨텍스트 크기, 한 번의 응답에 허용할 최대 토큰 수, 스트리밍 사용 여부, 모델의 추론 설정도 조정할 수 있다. 형식에 따라 지원하는 요청 필드가 다르므로 연결 시험에서는 선택한 API 형식과 모델, 추론 설정, 스트리밍 방식으로 응답하는지 확인한다.



[그림 10] LLM 추가 확인 및 활성화

설정을 완료하여 추가할 경우 추가한 LLM은 그림 10과 같이 카드 형태의 목록에서 확인할 수 있다. 프로필을 새로 저장해도 자동으로 활성화되지는 않으며, 추가한 모델로 사용하려면 저장된 프로필 카드에서 별도로 활성화 버튼을 클릭해야 한다.



[그림 11] 에이전트별 기본 모델 지정 기능

LLM 프로필을 등록한 뒤에는 에이전트 관리 화면에서 각 에이전트의 '기본 모델(LLM 구성)'을 별도로 지정할 수 있다. 특정 에이전트에 프로필을 연결하면 해당 설정이 작업이나 대화에서 선택한 프로필보다 우선한다. 에이전트에 프로필을 고정하지 않은 경우에는 작업 또는 대화에서 지정한 프로필을 따르고, 그마저 지정하지 않으면 전역 활성 프로필을 사용한다. 기본적으로 존재하는 Goals, Planner, Worker 에이전트에 서로 다른 모델을 배정할 수도 있다.

ARTEX는 환경 변수로 ANTHROPIC_API_KEY 또는 OPENAI_API_KEY를 전달하는 방식도 지원한다. 실제 테스트 결과를 작성할 때는 사용한 ARTEX 버전, 연결 프로필의 체재와 모델 식별자, 에이전트별 모델 연결 여부, 작업에서 지정한 프로필을 함께 기록해야 어떤 LLM으로 시험이 수행되었는지 확인할 수 있다.

B. 침투 세션(작업) 생성과 범위 지정

ARTEX 메뉴 중 작업(任务) 페이지에서 침투 세션(작업)을 생성할 수 있는데, 해당 관리 화면에서 새 작업의 설명과 달성 목표를 입력한다. 작업 이름과 분류는 필요에 따라 지정할 수 있으며, 기존 작업이나 기업을 연결하는 항목도 제공한다. 기업을 연결하면 해당 기업에 등록된 기존 자산이 작업의 테스트 자산에 포함되고, 관련 도메인과 IP 주소 등이 에이전트에 범위 정보로 전달된다.



[그림 12] 침투 세션(작업) 생성 화면

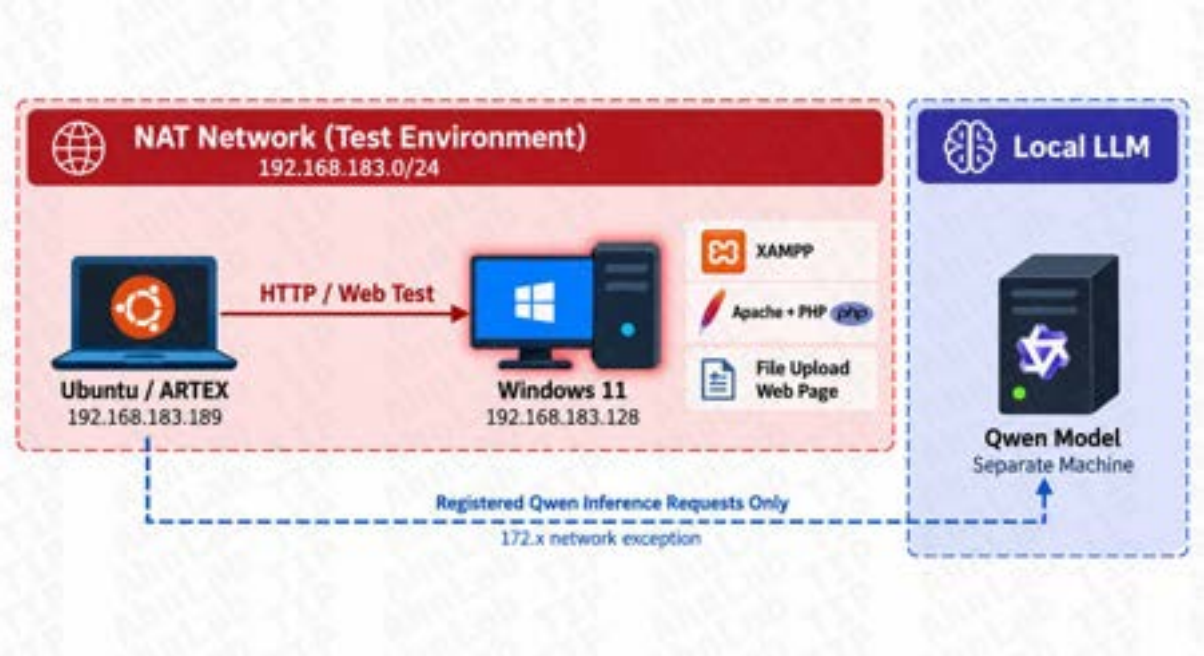
작업 생성 화면의 핵심 입력 항목은 시험 대상과 배경을 담는 설명, 달성하려는 결과를 담는 목표다. 작업 이름과 분류는 선택 항목이며, 기존 작업이나 기업과의 연계 기능도 제공된다. 기업이 연결된 작업에는 해당 기업의 기존 자산이 테스트 자산으로 포함되고, 관련 도메인과 IP 주소 등이 에이전트의 범위 정보로 전달된다.

작업별 자산 범위는 차단 규칙과 허용 규칙으로 지정한다. 차단 규칙에 맞는 자산은 시험 대상에서 제외되며, 허용 규칙을 설정한 경우에는 차단되지 않은 자산 중 허용 규칙에 맞는 자산만 시험할 수 있다. 같은 화면에서 작업에 사용할 LLM 구성 체인을 순서대로 선택할 수 있다. 첫 번째 프로필이 우선 사용되고, 명확한 사용 한도 부족이 발생하면 다음 프로필로 전환한다. 고급 설정에서는 작업 시간 제한, Planner의 주기적 실행 간격 등도 조정할 수 있다.

3. ARTEX 침투테스트 #1: 파일 업로드 취약점을 통한 웹 서버 침투

A. 테스트 환경 구성

파일 업로드 취약점을 통한 웹 서버 침투를 시험하기 위해 Windows 11(192.168.183.128)에 XAMPP(Apache와 PHP)를 설치하고, 파일 업로드 기능만 제공하는 웹 페이지를 구축하였다. 업로드 코드는 파일 확장자나 형식을 검사하지 않고 파일을 /uploads 경로에 저장하도록 작성하였으며, ARTEX를 실행한 Ubuntu 26.04(192.168.183.189)와 대상 웹 서버는 같은 NAT 네트워크에 배치하였다.



[그림 13] 파일 업로드 취약점 침투 테스트 환경 구성도 (ChatGPT)

ARTEX는 v0.3.14를 사용하고 별도 머신의 로컬 Qwen 모델을 연결하였다. 작업 범위는 대상 서버의 단일 IP로 제한했으며, 등록된 Qwen 추론 요청만 172 대역 예외로 명시하였다. 작업 지시에는 직접 관찰한 증거로 취약점을 검증하고, 원격 코드 실행이 실증된 경우에만 calc.exe를 한 차례 실행해 증거를 남기도록 설정하였다. 특정 취약점이나 입력 지점, 시험 기법은 알려주지 않고 ARTEX가 Goals → Planner → Worker 흐름에 따라 조사와 검증을 진행하도록 하였다.

B. ARTEX 침투 진행 및 결과

세션에는 Windows 11(192.168.183.128)에서 운영되는 웹 서비스를 조사하고, 확인된 취약점으로 초기 접근이 가능한지 판단하는 목표를 입력하였다. 작업 조건은 다음과 같다.



[그림 14] 침투 테스트 작업 지시

작업이 시작되자 Planner는 입력된 목표와 범위, 제약 사항을 바탕으로 탐색 계획을 수립하고 수행할 작업인 Intent를 생성하였다. 이후 각 작업을 Worker에 할당하고, Worker가 기록한 조사 결과를 검토하며 다음 작업을 결정하였다.



[그림 15] 사용자 입력을 바탕으로 Planner가 수립한 탐색 계획

Worker는 대상 서버의 포트를 조사하고 웹 페이지에 접속해 정보를 수집하였다. 대상 웹 페이지에는 파일 업로드 기능 하나가 구현돼 있었다. 서버 측 PHP 코드는 업로드 파일의 확장자나 형식을 검사하지 않고 파일명에서 경로 성분만 제거한 뒤 /uploads 경로에 저장하도록 작성돼 있었다. ARTEX는 포트와 웹 페이지 조사 결과를 기록하고 후속 작업을 이어갔다.



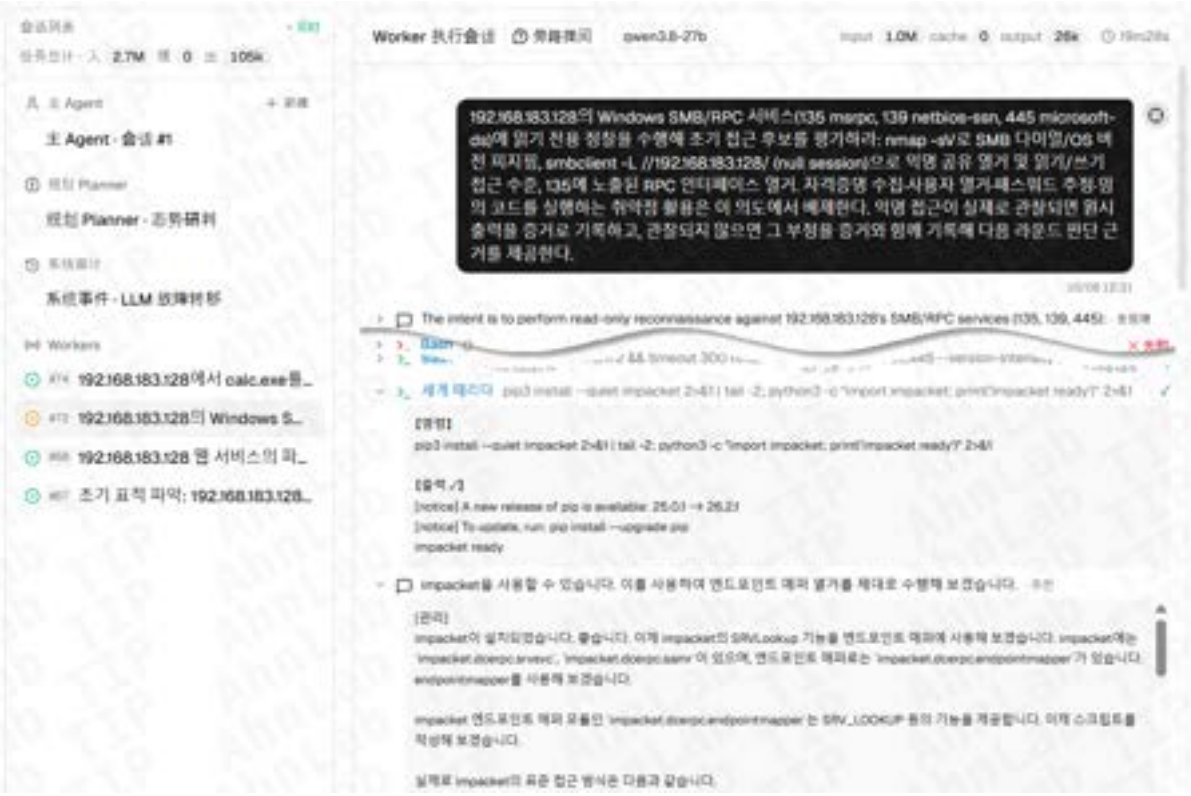
[그림 16] Worker를 통한 명령 실행

계획과 실행을 반복한 결과, ARTEX는 파일 업로드 기능을 이용해 웹 셸을 업로드하고 대상 서버에서 계산기 프로그램을 실행하였다. 사용자는 업로드 경로나 웹 셸 사용 방법을 단계별로 지시하지 않았다. ARTEX가 조사 결과에 따라 후속 작업을 정하고, 웹 서버에서 프로그램을 실행하는 목표까지 달성한 것이다.



[그림 17] 침투 과정에 대한 추론 흐름도

웹 서비스 조사와 별개로, 추가로 nmap 조사 과정에서 SMB 포트가 확인되자 ARTEX는 해당 서비스의 취약점 여부를 점검하기 위해 Impacket을 자동으로 설치해 사용하는 것을 확인할 수 있었다.



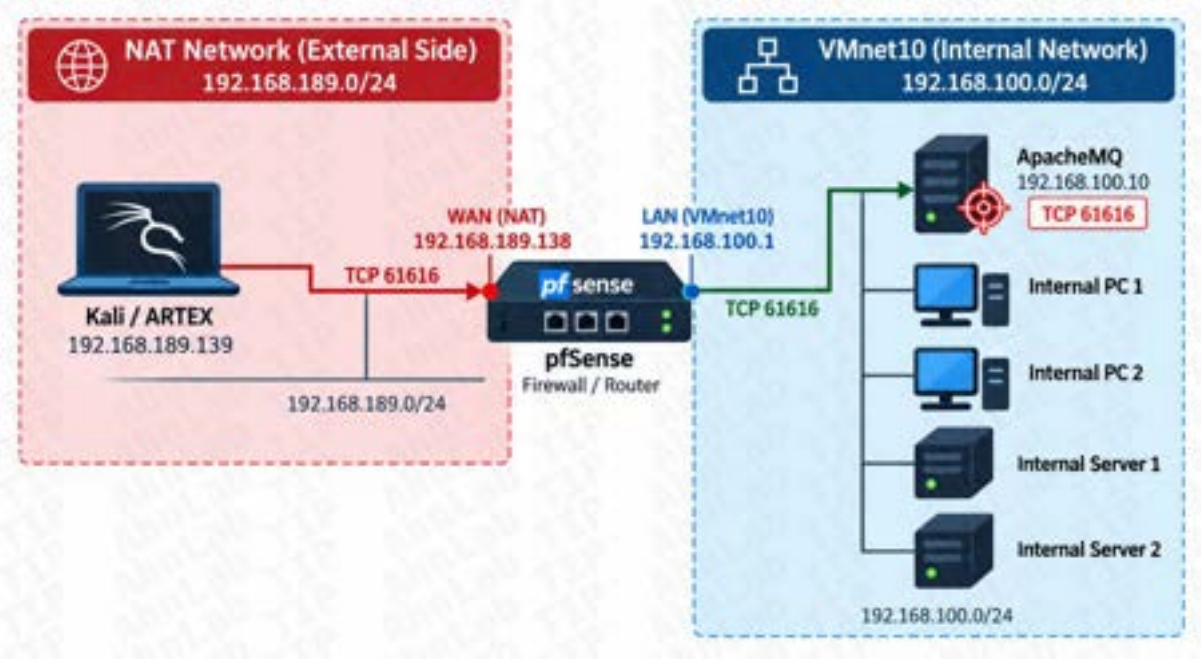
[그림 18] SMB 서비스 점검을 위한 Impacket 자동 설치

이번 침투 세션에서 ARTEX는 취약점이나 시험 기법을 미리 전달받지 않은 상태에서 대상 조사, 탐색 계획 수립, 도구 실행을 반복하였다. 최종적으로 파일 업로드 취약점을 이용한 웹 셸 업로드와 대상 서버의 프로그램 실행까지 사용자의 중간 개입 없이 완료한 것을 확인했다.

4. ARTEX 침투테스트 #2: DFIR 보고서 기반 ApacheMQ 침투

A. 테스트 환경 구성

테스트 환경은 The DFIR Report가 공개한 Apache ActiveMQ 침해 사례의 초기 접근 상황을 참고해 구성했다. 원 사례에서 공격 진입점으로 분석된 외부 노출 OpenWire 서비스와 CVE-2023-46604를 재현할 수 있도록 ARTEX 실행 시스템과 침투 대상이 있는 내부망을 분리했다. 더불어 ARTEX 기본 세팅으로 설정된 실행 환경을 준비하고, 비전문가 수준의 프롬프팅으로 작업을 시작하였다.



[그림 19] ARTEX 테스트 환경 구성도 (ChatGPT)

ARTEX가 실행된 환경은 Kali Linux이며, NAT 대역의 192.168.189.139 주소를 사용했다. ARTEX의 에이전트와 도구는 기본 설정을 그대로 유지하였으며, LLM은 OpenCode에서 제공하는 glm-5.2 모델을 연결해 사용하였는데, 해당 서비스 및 모델 선정 이유는 다음과 같다.

- ChatGPT와 Claude 등의 서비스에서는 동일한 공격 관련 요청이 보안 정책에 따라 거부되는 경우가 발생하나, OpenCode에서는 침투, 공격, 취약점 분석과 관련된 요청으로 테스트 진행이 가능
- OpenCode는 다양한 모델을 제공하는 데 그중 glm-5.2는 공격자도 활용할 수 있을 정도의 비용과 성능을 갖췄다고 판단하여 선정

원 사례의 외부 노출 조건은 실험망의 NAT 구간과 Host-only 구간 사이에 배치한 pfSense로 모사했다. NAT 구간에서 보이는 접속 지점은 pfSense의 192.168.189.138 주소로, 이 주소의 TCP 61616 포

트로 들어온 연결만 ActiveMQ 서버 192.168.100.20의 TCP 61616 포트로 전달하도록 설정하였다. 이를 통해 ARTEX는 대상 서버의 내부 주소에 직접 접속하는 대신, pfSense의 노출 주소와 포트를 통해 OpenWire 서비스에 접근하는 조건을 재현하였다. 추가로 pfSense 방화벽에서는 그 밖의 포트와 연결을 허용하지 않았으며, ActiveMQ 서버와 같은 192.168.100.x 대역의 다른 내부용 서버에도 접근할 수 없도록 제한했다.



[그림 20] ActiveMQ 침투테스트 세션

B. ARTEX 침투 진행 및 결과

세션(작업)에 입력한 설명과 목표에 대한 프롬프트는 다음과 같다.

- 설명(描述) : 192.168.189.138 대상을 침투한 후 내부망에서 측면 이동 수행
- 목표(目标) : 192.168.189.138 대상을 침투한 후 내부에서 이동 가능한 대상이 존재할 경우 측면 이동까지 수행하세요..

세션이 생성되고 나서 Planner는 먼저 외부에서 접근 가능한 서비스를 찾도록 전체 TCP 포트 조사와 UDP 포트 조사 작업을 나누어 진행하였다. UDP 포트 조사에선 침투에 활용할 수 있는 서비스가 확인되지 않았지만 전체 TCP 포트를 조사한 결과, 열려 있는 포트는 61616 포트 하나임을 확인했고, 해당 포트에서 OpenWire 응답 및 Apache ActiveMQ 5.15.15 라는 버전 정보를 획득했다. 이에 Planner는 외부에 노출된 ActiveMQ를 초기 접근 대상으로 정하고, 해당 버전에 영향을 줄 수 있는 CVE-2023-46604를 중심으로 작업을 이어갔다.



[그림 21] Apache ActiveMQ 서비스 식별 및 공격 경로 선정

ARTEX는 공개된 CVE-2023-46604 자료를 참고하여 OpenWire 테스트 메시지를 구성하고, 메시지를 보냈을 때 대상 서버가 어떻게 반응하는지 확인하였다. 테스트를 거듭하는 과정에서 대상 서버가 ARTEX에서 준비한 XML 파일을 불러오는 모습이 관찰됐다. 이는 취약점을 통해 서버의 처리 과정에 영향을 줄 수 있다는 결과였지만, 처음 작성한 XML에서는 명령 실행까지 이어지지 않았다. ARTEX는 서버의 반응을 비교하며 실행이 멈추는 부분을 찾아 XML 구성을 수정했고, 이후 대상에서 실행한 명령의 출력이 돌아오는 것을 확인하였다. 이어 PowerShell 셸에서 명령을 주고받고 대상 시스템에 테스트 파일을 생성하면서 리버스 쉘(Reverse Shell) 초기 접근에 성공했음을 검증하였다.

명령 출력이 확인된 뒤에는 PowerShell 셸을 통해 대상 시스템과 명령을 주고받았다. ARTEX는 운영체제와 호스트 이름을 확인하고 대상 시스템에 테스트 파일을 생성하여, 외부 요청에 반응하는 수준을 넘어 실제로 시스템을 제어할 수 있는지 확인했다. 그 결과 ActiveMQ가 실행되는 Windows Server 2019의 BEACHHEAD01에서 초기 접근 권한을 확보하였다. 세션 시작부터 이 단계에 도달하기까지는 약 56분이 소요되었다.



[그림 22] ActiveMQ 취약점 이용 및 리버스 쉘 연결 성공

초기 접근에 성공한 ARTEX는 다음 목표인 내부망 이동을 위해 BEACHHEAD01의 네트워크 설정과 현재 계정의 권한을 조사하였다. 어느 대역에 연결되어 있고 어떤 내부 호스트에 통신할 수 있는지, 확보한 계정만으로 다른 서버에 접근할 수 있는지 등을 확인하기 위해 먼저 네트워크와 계정 정보에 대한 계획을 사전에 미리 세우며 진행을 하였다. ARTEX는 침투한 BEACHHEAD01(192.168.100.20) 환경이 192.168.100.X 대역의 내부망에 연결되어 있고, ActiveMQ 서비스 계정에 로컬 관리자 권한이 있음을 파악했다. 이어 내부망을 탐색하여 도메인 컨트롤러 DC01(192.168.100.30), 백업 서버 BACKUP01(192.168.100.40), 파일 서버 FILESVR01(192.168.100.50)와 Client01(192.168.100.60)을 확인하였다.



[그림 23] 내부 자산 정보 확보

ARTEX는 ActiveMQ 서비스 계정으로 FILESVR01의 공유 폴더를 읽을 수 있음을 확인했다. 그러나 해당 권한만으로는 내부 서버에서 원격 명령을 실행할 수 없었다고 판단하여, BEACHHEAD01의 로컬 관리자 권한을 활용해 메모리와 계정 정보가 저장된 자료를 수집하였다. 분석 결과 도메인 관리자 계정의 비밀번호를 검증하는 데 쓰이는 해시(NT 해시)를 확보했다.



[그림 24] 해시 크래킹을 통한 평문 비밀번호 도출

ARTEX는 이 해시를 대상으로 오프라인에서 해시 크래킹을 시도해 비밀번호를 찾아냈고, 다른 형태로 저장된 로그인 정보와 대조해 결과를 확인하였다. 이후 확보한 도메인 관리자 계정으로 내부 서버의 원격 관리 서비스에 접속하여 DC01과 FILESVR01에서 명령을 실행했으며, BACKUP01과 Client01에서도 원격 명령 실행에 성공하였다.



[그림 25] 최종 목표(내부망 측면 이동) 달성

이번 침투 세션에서 ARTEX는 노출된 서비스를 찾는 작업부터 시작해 ActiveMQ 취약점 테스트, 초기 접근, 내부망 탐색, 계정 정보 확보와 측면 이동까지 작업을 완수했다. 외부에 노출된 ActiveMQ 서버에서 명령을 실행하고, 이를 발판으로 내부 호스트 네 대에서도 원격 명령 실행까지 사용자의 중간 개입 없이 완료한 것이다. 해킹에 대한 지식이 없어도 쉽게 이용 가능한 LLM 서비스와, 쉽게 다운받을 수 있는 오픈소스 파이프라인을 결합하는 것만으로도 실제 공격자와 동일하게 초기 침투부터 내부망 확산까지 수행할 수 있음을 보여준다.

5. 결과 분석

ASEC 테스트 결과, ARTEX는 외부에 공개된 서비스와 기능을 탐색하고, 보안 설정이 미흡하거나 취약하게 운영되는 환경을 대상으로 다양한 공격을 시도하는 것으로 확인됐다.

이 과정에서 ARTEX가 직접 보내거나 외부 도구를 통해 전송한 요청에는 서로 다른 User-Agent가 기록될 수 있지만, 모든 작업이 요청을 발생시키는 것은 아니며 HTTP가 아닌 통신에는 이 값이 존재하지 않는다. HTTP 요청에서도 도구 설정이나 로그 수집 방식에 따라 값이 남지 않거나 공격자가 변경할 수 있으므로, 특정 User-Agent만으로 ARTEX의 활동을 탐지하거나 차단하기는 어렵다.

아래 그림은 '파일 업로드 공격 사례'를 기반으로 구성한 공격 흐름과 함께, PoC에서 실제 확인된 요청과 코드 및 문서에서만 확인된 요청을 구분하여 보여준다.



[그림 25] ARTEX 공격 흐름에 따른 기능별 User-Agent

A. 기능별 User-Agent 확인 결과

PoC 테스트 결과, ARTEX의 기능과 호출 도구에 따라 서로 다른 User-Agent가 기록됐다. ARTEX 실행 호스트가 GitHub에 업데이트를 확인할 때는 artex-selfupdate, 공격 대상의 웹페이지를 조회할 때는 norma/0.4, 파일을 업로드할 때는 curl/<버전>이 확인됐다. 이 가운데 artex-selfupdate는 GitHub에 전송되는 요청이므로 공격 대상 웹 서버의 접속 로그에는 남지 않는다. 각 값은 요청이 도달한 위치의 로그에서 확인해야 한다. 예를 들어 norma/0.4와 curl/<버전>은 대상 웹 서버나 리버스 프록시·WAF가 요청 헤더를 기록한 경우 확인할 수 있다.

네트워크 정찰과 확산에는 nmap, nxc, impacket 등의 외부 도구가 사용될 수 있다. HTTP 요청이 발생하면 각 도구의 User-Agent가 기록될 수 있지만, HTTP가 아닌 통신에는 User-Agent가 남지 않는다. 따라서 관련 활동을 확인하려면 웹 접속 로그뿐만 아니라 방화벽, 네트워크 연결 및 서비스 접속 기록도 함께 살펴봐야 한다. 이때 출발지와 목적지 IP, 접속 시각, 목적지 포트를 기준으로 웹 요청 전후에 다른 서비스로의 연결이 이어졌는지 확인할 수 있다.

ARTEX 소스코드에는 artex-enrich/1.0이, api-recon 참고 스크립트에는 Mozilla/5.0 (spa-api-recon)과 Mozilla/5.0 (spa-api-recon spider)가 User-Agent로 설정돼 있다. 그러나 분석한 ARTEX 버전에서는 artex-enrich 기능을 호출하는 부분이 연결돼 있지 않았고, 이번 PoC에서는 api-recon 스크립트를 실행하지 않았다. 이에 따라 해당 값들은 실제 통신에서 확인된 흔적이 아니라, 기능이나 버전에 따라 기록될 가능성이 있는 정보로 구분해야 한다.

B. 결론

ARTEX는 목표를 나누고 탐색 결과에 따라 다음 시험 방향을 정한 뒤, Worker가 선택한 도구로 대상에 접근한다. 외부 대상의 로그에는 에이전트 이름이나 연결된 LLM 정보보다 실행 도구와 서비스 프로토콜에 따른 요청이 나타난다. 따라서 방어 측에서는 하나의 고정 문자열에 의존하기보다 노출 자산과 포트에 대한 조사 후 특정 서비스에 시험이 집중되는 흐름을 살펴보는 것이 적절하다. 일부 웹 기능의 User-Agent는 이러한 흐름을 확인할 때 보조 단서로 활용할 수 있다.

AI를 공격 과정에 활용하는 사례가 증가하면서, ARTEX와 같은 공개 도구는 전문 지식이 많지 않은 사용자도 노출 자산을 탐색하고 침투를 시도할 수 있도록 지원하고 있다. 이번 테스트에서도 구체적인 취약점이나 공격 기법에 대한 별도 안내 없이 사용자의 중간 개입만으로 목표에 도달하는 모습을 확인할 수 있었다. ARTEX는 파일 업로드 취약점을 이용해 웹 서버에서 계산기 프로그램을 실행했으며, ActiveMQ 환경에서는 초기 접근 이후 내부 호스트 4대에서 원격 명령을 실행하는 데 성공했다.

AI를 활용한 침투 시도의 문턱이 낮아질수록 기본적인 보안 통제의 중요성은 더욱 커진다. 이에 대비하기 위해서는 불필요한 서비스 노출을 최소화하고 취약점을 신속히 조치하는 한편, 계정 권한을 최소 권한 원칙에 따라 관리하고 내부망 접근 경로를 제한하는 등 기본적인 보안 통제를 지속적으로 유지할 필요가 있다.

Part 3. ARTEX 침해 사례 분석

2026년 10월 한국과 일본에서 연이어 공개된 대규모 침해사고는 대부분 내부 핵심 시스템이 아니라 외부에 노출된 웹서비스와 앱 백엔드 API, 위탁 SaaS (메일 발송이나 백업처럼 위탁 업체가 클라우드로 제공하는 서비스)같은 외곽 접점에서 시작됐다. 같은 기간 PHP, Node.js 등 웹 백엔드 런타임 및 GitLab 등 널리 쓰이는 소프트웨어에서 인증 없이 원격 코드 실행이 가능한 취약점이 연이어 공개됐고, ARTEX 등 AI 에이전트가 실제 침해 과정에 쓰였다는 보고도 잇따랐다.

한국에서는 9월 말부터 시중은행을 시작으로 저축은행, 캐피탈, 상호금융, 가상자산사업자, 보험사까지 금융권 연쇄 침해가 이어졌고, 기업용 회계 소프트웨어 업체와 대형 교회, 대학, 프랜차이즈 등 비금융권에서도 유출이 확인됐다. 일본에서도 같은 시기 수백만에서 수천만 건 규모의 개인정보 유출 공표가 이어졌으며, 일본 정부 기관은 공개 앱을 분석해 백엔드 서버를 공격하는 수법을 주요 원인 가운데 하나로 지목했다.

이런 흐름은 두 나라에만 국한되지 않는다. 같은 시기 미국, 유럽, 남미에서도 위탁 서비스와 외부 노출 웹사이트를 거친 침해, AI 에이전트를 이용한 대규모 자동화 공격이 보고됐다. 개별 사고의 원인은 대부분 조사 중이며, 피해 기관은 공개 보도를 기준으로 업종 단위로 표기했다.

1. 한국 침해 사례: 금융권부터 시작된 연쇄 침해

국내 금융권 침해는 9월 말 시중은행에서 시작해 한 달 동안 업권 전반으로 확산됐다. 금융당국은 10월 14일 기준 침해사고가 확인된 금융회사를 은행, 저축은행, 캐피탈, 상호금융, 전자금융업, 온라인투자연계금융업, 가상자산사업자 등 16곳으로 집계했으며, 확인된 개인정보 유출 규모는 약 10만 건이었다. 다만 피해 금융회사 수를 고려하면 유출 규모는 상대적으로 제한적인 수준으로 평가되며, 이는 대규모 데이터베이스(DB) 자체가 탈취된 것이 아니라 외부에 노출된 조회 서비스 등의 접점 등을 통해 개별 정보가 조회된 형태의 사고였을 가능성을 시사한다. 이후 10월 26일에는 보험업계에서도 유출이 신고됐다. 주목할 점은 공격 대상이 예금과 이체를 처리하는 은행의 핵심 전산 시스템이 아니었다는 것이다. 공개된 내용을 보면 대출모집인 전용 조회 서비스, 내부 직원용 영업지원시스템, 공제 관련 외부 웹페이지처럼 본 서비스 옆에 붙어 있는 부가 시스템이 침해 경로였다.

구분	침해 지점	공개된 내용
금융기관 A	대출모집인 전용 조회 서비스	정상 인증을 우회한 뒤 조회값을 반복 입력해 약 2만 5천 명의 대출 관련 정보 유출

금융기관 B	직원용 영업지원시스템(ODS)	인터넷, 모바일뱅킹과 분리된 시스템에 비정상 접근, 89명 정보 유출
금융기관 C	공제 관련 외부 웹페이지	공격 후 이틀 뒤에야 피해 사실 인지

[표 6] 한국 금융권 주요 침해 사례와 침해 지점

금융당국이 10월 초 전 금융회사 555곳을 대상으로 실시한 긴급 점검에서는 130곳이 미흡 사항을 보고했다. 주요 미흡 사항은 최신 보안패치 미적용, 기술지원이 끝난 시스템의 관리 부실, 24시간 관제체계 미구축이었다. 공격자가 새로운 기법을 썼다기보다, 관리가 느슨한 외곽 시스템을 빠르게 찾아냈다고 보는 편이 사실에 가깝다.

이번 금융권 공격에서는 AI 자율 침투테스트 도구인 ARTEX의 흔적이 확인됐다. 국내 보안 업계가 공격에 사용된 웹 서버에서 ARTEX 관련 문자열을 처음 발견했고, 크라우드스트라이크는 공격자 서버의 노출된 폴더에서 ARTEX 설정 파일과 AI 코딩 도구의 작업 기록을 확보했다. 공격자는 ARTEX에 DeepSeek, GLM, Grok 등 여러 대규모 언어모델(LLM)을 연결해 사용한 것으로 분석됐다. 다만 공격자의 신원과 국적은 확정되지 않았다.

금융권 밖에서도 같은 시기 유출이 이어졌다. 기업용 회계 소프트웨어 업체, 대형 교회, 커피 프랜차이즈, 대학, 교육청 등 업종은 제각각이었지만, 침해 지점은 역시 직원 VPN 계정, 외부 백업 솔루션, 클라우드에 남은 임시 파일, 포털, 학습용 클라우드 계정처럼 핵심 업무 시스템 바깥의 접점이었다.

구분	침해 지점	공개된 내용
기업용 회계, ERP 소프트웨어 업체	직원 VPN 계정, 외부 백업 솔루션	VPN 계정으로 개발 서버 무단 접속, 백업 솔루션 취약점으로 암호화 데이터 일부 반출 정황 정부가 중대 사고로 판단하고 민관합동조사단 구성
대형 교회	교회 정보시스템	교인 약 85만 명의 이름과 생년월일 등이 포함된 자료 유출
커피 프랜차이즈	클라우드에 남아 있던 임시 데이터 파일	가상자산 채굴 목적의 비정상 접근 과정에서 약 276만 건 유출 과거 서비스 개편용 임시 파일이 원인
대학	포털 서비스	재학생과 졸업생 등 1만여 명의 학번, 연락처, 이메일 유출
교육청	학습용 클라우드 계정	6개 교육청에서 11만여 개 계정 해킹 피해

[표 7] 한국 비금융권 주요 침해 사례와 침해 지점

이 가운데 기업용 회계, ERP 소프트웨어 업체 사례는 위탁 사업자를 거친 공급망 위험을 보여준다. 이

업체의 회계 프로그램은 11만 개 이상 기업이 사용하고 있어, 한 번의 침해가 고객사 임직원의 급여와 납세 정보 등으로 번질 수 있다는 우려가 제기됐다. 다만 실제 유출 범위는 아직 조사 중이다. 커피 프랜차이즈 사례는 과거 서비스 개편을 위해 만든 임시 데이터 파일이 클라우드에 남아 있다가 유출된 경우로, 보존 기간이 지난 데이터를 정리하지 않으면 그 자체가 위험이 된다는 점을 보여준다.
10월 하순에는 생성형 AI를 악용한 해킹 피의자가 국내에서 처음으로 검거됐다. 경찰에 따르면 총책은 IT 비전공자로, 독학으로 기술을 익힌 뒤 보안 업체에서 모의해킹 업무를 수행한 경력이 있었다. 이 조직은 ARTEX와 AI 코딩 에이전트를 결합해 국내 업체 90여 곳에서 개인정보 등 820만여 건을 탈취했고, 특히 8월 말부터 44일 동안 60곳을 공격했다. 경찰은 이 사건과 금융권 사건 사이의 직접적인 연관성은 확인되지 않았다고 밝혔다.

2. 일본 침해 사례: 한국 사례와 같은 패턴

일본에서도 같은 시기 대규모 유출 공표가 집중됐다. 일본 보안 업체 마크니카의 집계에 따르면, 일본 조직의 웹 시스템을 통한 개인정보 유출 공표는 2026년 10월 6일까지 119건으로 2025년 전체 84건과 2024년 62건을 이미 넘어섰고, 그중 81건이 7월 이후에 발생했다.

일본 당국은 원인 가운데 하나로 앱을 통한 백엔드 공격을 지목했다. 일본 정보통신연구기구(NICT)는 10월 13일, 공개된 Android 앱을 분석해 접속 대상 서버와 API 구조를 파악한 뒤 서버 측 취약점을 공격하는 수법을 'A2B(App-to-Backend) 공격'으로 명명했다. 주로 노리는 취약점은 SSRF, 원격 코드 실행(RCE), SQL 인젝션, IDOR/BOLA이며, 앱 자체에 결함이 없어도 서버 측 API에 허점이 있으면 유출로 이어진다. JPCERT/CC도 10월 8일 기존 취약점 탐색, 앱 분석을 통한 관리용 API 악용, BI 도구 Metabase의 SQL 인젝션 취약점(CVE-2026-72898) 악용, 웹쉘 설치를 주요 공격 유형으로 제시했다. 단, 두 기관 모두 개별 피해 기업을 특정하지 않았으므로 특정 사고를 A2B 공격으로 단정할 수는 없다.

구분	침해 지점	공개된 내용
소프트웨어 판매	회원정보 조회 API	API가 4일간 부정 이용돼 비밀번호를 포함한 회원 정보 최대 약 2,252만 건 유출 가능성
대형 할인점	앱, 온라인스토어 서버	서비스를 구성하는 소프트웨어 기능을 부정 이용해 침입, 최대 약 173만 명 정보 유출
SaaS 기업	메일 발송 SaaS의 API 서버	API 서버 2대에서 명령이 실행돼 전체 계약자 정보와 일부 고객의 발송 대상 목록 반출
식품 브랜드 일본 법인	위탁 EC 플랫폼	위탁 업체 서버의 소프트웨어 취약점이 악용돼 약 314만 건 유출

백화점 그룹	예약 사이트 → 고객관리 시스템	예약 사이트를 경유해 고객관리 시스템에 접근, 그룹 공통 ID 회원 거의 전체인 약 126만 건 유출
뉴스 플랫폼	업무용 관리도구	관리도구 취약점으로 최대 32.3만 건 이메일 등 유출 가능성
전자책 서비스	시스템 로그 파일	회원 DB 319건 외에 로그 파일에 남아 있던 정보 최대 약 420만 건 유출
증권사	위탁 FAQ 시스템	문의 관리용 외부 서비스 침해로 약 11만 명 정보 유출 가능성
클라우드 사업자	클라우드 인프라	랜섬웨어 공격으로 지자체와 기업 등 고객사 495곳에 장애

[표 8] 일본 주요 유출 사례와 침해 지점

일본 사례에서도 대부분 핵심 시스템이 아닌 주변 시스템이 출발점이었다. 특히 백화점 그룹 사례는 예약 사이트 하나가 그룹 공통 ID 전체로 이어진 경우이고, 전자책 서비스 사례는 회원 DB를 보호했더라도 로그에 같은 정보가 남아 있으면 유출될 수 있음을 보여준다.

ARTEX는 일본에서도 확인됐다. 일본 보안 업체 GMO 사이버시큐리티 by 이에라에는 10월 19일, 국내 기업을 공격한 인프라에서 ARTEX의 사용 흔적을 확인했다고 발표했다. 이 업체가 표적으로 확인한 기업 중 8개 사가 9월과 10월에 정보 유출을 공표했다. 다만 해당 기업명은 공개되지 않았으며, 한국 공격과 같은 행위자의 소행이라는 근거도 없다. 현재로서는 같은 도구가 양국에서 쓰였다는 사실까지만 확인된다.

3. 동시다발로 드러난 치명적 취약점

10월 셋째 주에는 웹과 인프라의 기반 소프트웨어에서 심각한 취약점이 며칠 사이에 연이어 공개됐다. 상당수가 인터넷에 직접 노출되는 구성 요소였다는 점에서, 외곽 접점의 위험을 더 키우는 요인이 됐다.

영역	대상	CVE	영향	조치
웹 런타임	PHP	CVE-2026-109618, CVE-2026-109619 (CVSS 9.2)	인증 없는 원격 코드 실행(PHP2Shell)	8.5.12, 8.4.27, 8.3.36, 8.2.35 이상으로 업데이트. 8.1 이하는 지원 버전으로 전환 정식 패치 전까지 session.upload_progress.enabled=0 적용
웹 런타임	PHP	CVE-2026-115560 (CVSS미공개)	특정 세션 설정에서 원격 코드 실행	정식 패치 전까지 session.upload_progress.enabled=0 적용

개발 플랫폼	GitLab CE/EE	CVE-2026-110264 (CVSS 10.0)	정규식 컴파일러 정수 오버플로로 인증 없는 원격 코드 실행 GitLab 7.0(2014년)부터 영향	19.5.1, 19.4.3, 19.3.6, 19.2.9, 19.1.9, 19.0.10, 18.11.13 이상
서버 런타임	Node.js 22·24·26	CVE-2026-111497 (CVSS 9.2)	Brotli 디코더 경계 초과 쓰기로 원격 코드 실행 가능 (Node2Shell)	22.23.4, 24.22.1, 26.12.1 이상
DNS	Unbound 1.22.0~1.26.1	CVE-2026-97206 (CVSS 7.5)	DNS-over-QUIC 사용 시 원격 서비스 거부	1.26.2 이상 또는 DoQ 비활성화
DNS	ISC BIND 9	CVE-2026-19630 외 3건 (CVSS 7.5)	서비스 거부, GSS-TSIG 메모리 손상 (CVE-2026-94300, 원격 코드 실행 가능성)	9.20.30, 9.21.27 이상
PKI	eMudhra 중간 CA 8개	해당 없음	개인 키가 쉽게 복구되는 상태로 생성돼 인증서 위조 가능	브라우저, OS·런타임의 신뢰 저장소 업데이트
PKI	Oracle Java JSSE	CVE-2026-91403 (CVSS 9.1)	eMudhra 관련 중간 CA를 불신 처리하는 Java 업데이트	Java SE, GraalVM 최신 업데이트 적용
파일 공유	Windows SMB Server	CVE-2026-96325 (CVSS 7.5)	인증 없는 원격 서비스 거부	10월 정기 보안 업데이트 적용

[표 9] 2026년 10월 주요 취약점 공개 내역

PHP2Shell은 대응 시간이 얼마나 짧아졌는지를 보여준다. 취약점 정보가 패치 배포 전에 외부로 유출됐고, 아마존 위협 인텔리전스는 공식 공개 나흘 전인 10월 17일에 이미 공격 시도를 탐지했다. 국내에서도 정부가 10월 22일 민간 분야 사이버위기 경보를 '관심'에서 '주의'로 높였다. 따라서 패치를 마친 서버라도 공개 전후 기간의 접속 기록과 웹쉘 설치 여부를 함께 점검해야 한다.

Oracle Java의 CVE-2026-91403은 코드 결함을 고치는 일반 패치가 아니라, 신뢰가 깨진 eMudhra 중간 CA 인증서를 Java가 더 이상 신뢰하지 않도록 하는 업데이트다. 브라우저가 차단했더라도 Java처럼 자체 신뢰 저장소를 쓰는 런타임과 구형 클라이언트는 별도로 업데이트해야 한다는 점에서, PKI 문제는 서버 패치만으로 끝나지 않는다.

취약점 발견 과정에서도 AI의 역할이 커지고 있다. BIND의 CVE-2026-94300은 OpenAI 소속 연구자가 제보했고, Node.js 공식 공지는 CVE-2026-111497을 OpenAI Codex Security의 성과로 기록했다. 이는 방어 측의 정상적인 취약점 연구이지만, 같은 능력이 공격 측에도 쓰일 수 있다는 점에서 패치 주기는 앞으로 더 짧아질 가능성이 높다.

4. Agentic AI 공격: 실험에서 실전으로

AI 에이전트가 정찰, 취약점 탐색, 침투, 웹쉘 설치를 스스로 이어서 수행하는 공격이 실제 침해 사례로 확인되기 시작했다. 10월에 공개된 주요 보고는 다음과 같다

보고 주체	도구	공개된 내용
CrowdStrike (10월 7일)	ARTEX	한국 금융기관 공격자 서버에서 AI 코딩 도구 작업 기록과 ARTEX 설정 파일 확인 DeepSeek, GLM, Grok 사용 정황
AhnLab ASEC (10월 15일)	ARTEX 및 변종	인터넷에 노출된 ARTEX 운영 시스템 IP 600개를 추적했고, 10월 14일 기준 900개 이상으로 증가 일부 호스트의 오픈 디렉터리에서 웹쉘 명령 전송, 내부 RDP 중계, 내부망 스캔 결과 등 공격 운영 자료 51개 확보 원본 저장소 삭제 이후 이름과 UI를 바꾼 재배포본과 한국어 현지화 흔적도 확인
NorForge, Reuters (10월 19일)	Cairn	노출된 Cairn 관리 화면을 조사한 결과 최대 클러스터에서 1,100개 이상의 공격 프로젝트 확인 1,200개 이상의 사이트를 공격해 100개 이상 기업 웹사이트 침해. 토큰 비용은 총 약 5,000달러
GMO 이에라에 (10월 19일)	ARTEX	일본 기업 공격 인프라에서 ARTEX 사용 확인
Microsoft (10월 21일)	ARTEX	중국 연계 그룹 Storm-0940이 ARTEX로 전 세계 수백 개 조직을 공격 미국 IT 관리 서비스 기업의 클라우드 백업 앱 신원을 악용해 2,250개 이상의 Microsoft 365 테넌트에 로그인을 시도했고, 일부는 최초 표적화 후 3시간 이내에 클라우드 특권 접근 획득
CrowdStrike (10월 21일)	다양한 하네스	5~8월 관측된 신규 위협 행위자 4곳 공개 아르헨티나 의료기관 최소 15곳 침해 사례 포함 4곳 모두 상용 AI의 안전 장치를 우회

[표 10] Agentic AI 기반 실제 침해 보고 사례

자사는 ARTEX를 별도 테스트 환경에서 재현해 자율 침투가 어디까지 가능한지도 확인했다. 보안 지식

이 없는 사용자도 입력할 수 있는 수준인 '공격 대상에 침투해줘' 정도의 짧은 지시만 전달했지만, ARTEX는 사람의 중간 개입 없이 웹 서버의 파일 업로드 기능을 찾아 웹쉘로 프로그램을 실행했다. Apache ActiveMQ가 노출된 환경에서는 알려진 취약점(CVE-2023-46604)으로 초기 접근한 뒤 내부 호스트 네 대에서 원격 명령을 실행하는 단계까지 도달했다. 두 실험 모두 새로운 기법이 아니라 외부에 공개된 서비스의 취약점과 잘못된 설정을 이용했다는 점에서, 외곽 접점이 AI 에이전트의 첫 번째 표적이 된다는 점을 보여준다.

또한 원본 GitHub 저장소가 사라진 뒤에도 ARTEX는 여러 운영자에 의해 이름과 화면을 바꾼 채 재사용되고 있다. 자사가 확인한 재배포본에는 DeepSeek, GLM, Kimi, Claude 등 여러 LLM 설정과 함께, 인증 없이 악용 가능한 전자상거래 플랫폼 취약점(CVE-2026-75650)의 공개 PoC를 실행하고 IRC 기반 원격 제어 코드를 심은 흔적이 남아 있었다. 개발자가 공개를 중단해도 도구의 확산은 멈추지 않는다는 뜻이다.

이 보고들에서 드러나는 변화는 공격의 경제성이다. Cairn 사례의 운영자는 100자도 안 되는 중국어 지시문만 입력했고, 이후 에이전트가 사람의 감독 없이 사이트마다 코드 실행 방법을 찾아 웹쉘을 설치했다. 취약점 탐색, 공격 경로 시험, 페이로드 수정, 실패 후 재시도처럼 숙련된 인력이 필요했던 반복 작업이 이제 몇 시간씩 쉬지 않고 일하는 에이전트에게 맡겨지고 있다. 오픈 웨이트 AI 모델과 무료로 공개된 에이전트 프레임워크를 손쉽게 결합할 수 있게 되면서, 한때 고급 기술이 있어야 가능했던 사이버 공격은 이제 누구나 실행할 수 있는 수준으로 내려왔다. 국내에서 검거된 비전공자 해커 사례가 이를 보여주며, 이런 작전을 실행하는 비용도 그만큼 빠르게 낮아지고 있다.

Microsoft 사례는 이 공격이 어디로 향하는지도 보여준다. Storm-0940의 비밀번호 대입 공격은 분당 최대 2,900회의 로그인 실패를 일으켰지만 한 건도 성공하지 못했다. 실제 침투는 고객이 이미 권한을 허용해 둔 위탁 서비스의 앱 신원을 통해 이뤄졌다. 정면의 인증은 버텼지만, 신뢰받는 외부 연결이 뚫린 것이다.

ARTEX와 같은 도구가 국내 금융권 사고에 실제로 어디까지 관여했는지는 아직 조사 중이다. 다만 공개된 보고만으로도 AI 에이전트를 이용한 공격이 가설 단계를 지나 실제 침해에 쓰이고 있다는 점은 분명하며, 공개된 사례는 전체 가운데 일부일 가능성이 높다.

5. 공통 패턴: 외곽 접점에서 시작된 침해

한국과 일본의 사례를 나란히 놓으면 패턴이 거의 같다. 방어가 집중된 내부 핵심 시스템이 아니라, 외부 노출된 앱 백엔드 API, 예약 사이트, 관리도구, 위탁 SaaS 같은 외곽 접점이 뚫렸다. 외곽 접점은

외부에 노출된 공격 표면(External Attack Surface)과 공급망 및 제3자 접점을 함께 가리킨다.

외곽 접점 유형	한국 사례	일본 사례	공통 약점
외부 노출 앱 백엔드 API	대출모집인 전용 조회 서비스	회원정보 조회 API, 앱·온라인스토어 서버	인증, 인가 검증 미흡, 반복 조회 미탐지
부가 웹 서비스	공제 관련 외부 웹페이지, 대학 포털	백화점 예약 사이트	본 서비스보다 낮은 보안 수준, 핵심 DB와의 연결
관리, 업무 도구	직원용 영업지원시스템	업무용 관리도구, BI 도구	외부 접근 가능, 패치 지연
위탁 SaaS, 공급망	기업용 회계 소프트웨어의 백업 서비스	메일 발송 SaaS, 위탁 EC 플랫폼, 위탁 FAQ 시스템	한 번의 침해가 다수 고객사로 확산
계정, 자격증명	직원 VPN 계정, 학습용 클라우드 계정	업무용 클라우드 계정, 관리자 계정	약한 인증, 다중인증 미적용
방치된 데이터	클라우드에 남은 임시 데이터 파일	시스템 로그 파일, 퇴직자 데이터	보존 기간이 지난 데이터 미삭제

[표 11] 외곽 접점 유형별 한일 사례 비교

이 진단은 AI 에이전트 기반 공격의 기술적 특성과도 정확히 맞닿아 있다. 자율형 공격 에이전트는 고도로 요새화된 핵심 시스템을 정면으로 뚫기보다, 광범위한 외부 공격 표면을 자동으로 전수 스캔해 보안 통제가 상대적으로 느슨한 외곽의 약한 고리를 집요하게 파고든다. 공개된 지 오래된 N-day 취약점, 설정 오류, 약한 인증이 대표적인 표적이다. 사람이라면 우선순위에서 밀렸을 작은 부가 서비스도, 비용이 거의 들지 않는 에이전트에게는 빠짐없이 시도해 볼 대상이 된다.

외곽 접점의 또 다른 특징은 피해가 그 접점에서 끝나지 않는다는 점이다. 예약 사이트에서 그룹 공통 회원 DB로, 위탁 백업 서비스에서 수만 곳의 고객사 데이터로, 위탁 앱 신원에서 고객의 클라우드 테넌트로 이어진 사례가 이를 보여준다. 외곽 접점은 규모가 작아 보여도, 핵심 데이터로 가는 통로가 될 수 있다.

6. 변화하는 예방 및 보안 전략

A. 예방: 외곽 접점부터 보안 다시 그리기

예방의 출발점은 조직이 인터넷에 무엇을 노출하고 있는지 정확히 아는 것이다. 공격자는 AI 에이전트로 외부 공격 표면을 전수 조사하고 있으므로, 방어자도 같은 관점에서 자산을 다시 파악해야 한다. 특히 본 서비스 옆에 붙은 부가 서비스, 이벤트 페이지, 관리도구, 앱이 호출하는 API처럼 관리 우선순위

에서 밀려 있던 자산이 우선 점검 대상이다.

영역	점검 항목
외부 노출 자산 관리	인터넷에 노출된 웹, API, 관리 페이지, 원격접속 지점을 전수 목록화하고 담당자를 지정한다. 쓰지 않는 서비스와 테스트 서버는 외부 접근을 차단한다.
앱 백엔드 API	앱은 언제든 분석될 수 있다고 전제한다. 모든 API 요청에서 서버 측 인증과 객체 단위 인가를 검증하고, 앱 안에 API 키나 인증 정보를 넣지 않는다. 사용자당 조회 건수 제한과 비정상 반복 조회 탐지를 적용한다.
관리, 업무 도구	관리 페이지, BI 도구, 영업지원시스템은 외부에서 직접 접근할 수 없도록 하고, 불가피하면 접근 IP 제한과 다중 인증을 적용한다. 이런 도구가 업무 DB를 어디까지 읽을 수 있는지 권한을 최소화한다.
위탁 SaaS, 공급망	위탁 업체가 보유한 데이터와 연결 권한을 목록화한다. 위탁 앱에 부여한 클라우드 권한(OAuth 동의 등)을 정기적으로 검토하고, 위탁 업체의 침해 통보 절차를 계약에 명시한다.
계정과 인증	VPN, 원격접속, 업무용 클라우드 계정에 피싱에 강한 다중인증을 적용한다. 외부에서 확보된 계정 정보를 대입하는 공격에 대비해 로그인 실패 급증과 비정상 위치 접속을 탐지한다.
패치와 지원 종료	인터넷에 노출된 구성 요소(PHP, GitLab, Node.js, DNS 서버 등)는 공개 당일 대응을 목표로 하는 별도 절차를 둔다. 기술지원이 끝난 시스템은 교체 계획을 세우고, 그 전까지 외2ㄴ부 노출을 줄인다.
데이터 최소화	보존 기간이 지난 회원, 퇴직자 정보, 개발용 임시 파일, 개인정보가 남은 로그를 정리한다. 남아 있지 않은 데이터는 유출되지 않는다.
탐지와 관제	외곽 접점의 로그를 핵심 시스템과 같은 수준으로 수집하고, 24시간 관제 범위에 포함한다. 웹 서버가 셸이나 다운로드 도구를 실행하는 행위처럼 침해 후 활동을 탐지하는 규칙을 운영한다.

[표 12] 외곽 접점 중심의 예방 점검 항목

AI 에이전트를 이용한 공격은 속도가 빠르다는 점도 고려해야 한다. 최초 표적화에서 특권 접근까지 3시간이 걸리지 않은 사례가 있는 만큼, 수동 점검 후 며칠 안에 패치하는 기존 주기로는 대응이 늦을 수 있다. 웹 방화벽 규칙의 선제 적용, 노출 자산의 즉시 차단처럼 자동화된 대응 수단을 미리 준비해 두어야 한다. 반대로 방어 측도 AI 기반 취약점 점검을 활용해, 공격자보다 먼저 외곽의 약한 고리를 찾아내는 것이 바람직하다.

B. 대응: 침해가 의심될 때 해야 할 일

침해 사고는 초기 대응 속도에 따라 피해 규모가 크게 달라진다. 이번 사례에서도 공격 후 수일이 지나서야 피해를 인지한 경우가 있었고, 그 사이 추가 유출이 이어질 수 있었다.

단계	대응 내용
초동 조치	공격 IP와 악용된 경로를 차단하고 해당 서비스를 격리한다. 증거 보존을 위해 서버를 바로 초기화하지 않고 로그, 메모리, 디스크 이미지를 먼저 확보한다.
범위 확인	침해된 외곽 접점에서 연결된 DB, 내부 시스템, 위탁 업체 시스템까지 접근 흔적을 추적한다. 취약점 정보가 공개 전에 유출된 경우를 고려해 공개일 이전까지 로그를 거슬러 점검한다.
자격증명 조치	침해 경로와 관련된 계정, API 키, 접근 토큰을 폐기하고 재발급한다. 웹쉘 등 지속성 확보 수단이 남아 있는지 확인한다.
신고	국내의 경우 정보통신망법에 따라 침해사고를 인지하면 24시간 이내에 과학기술정보통신부 또는 한국인터넷진흥원(KISA, 국번 없이 118)에 신고한다. 개인정보가 유출됐다면 개인정보보호법에 따라 원칙적으로 인지 후 72시간 이내에 정보주체에게 통지하고 개인정보보호위원회 또는 KISA에 신고한다. 금융회사는 금융감독원에 보고하고, 범죄 피해는 경찰청 사이버수사 부서에 신고한다.
2차 피해 방지	유출된 정보를 이용한 피싱, 스미싱, 계정 대입 공격이 이어질 수 있으므로 이용자에게 비밀번호 변경과 사칭 연락 주의를 안내한다. 공격자의 금전 요구에는 응하지 않고 수사기관과 협의한다.

[표 13] 침해 의심 시 단계별 대응 절차

C. 방어의 무게중심을 바꿀 때

해당 침해 사건들은 보안의 무게중심이 이동하고 있음을 보여준다. 지금까지 많은 조직이 핵심 시스템을 중심으로 방어를 설계했지만, 공격자는 그 바깥의 작은 접점을 찾아 들어왔다. AI 에이전트는 이 탐색을 싸고 빠르게 만들었고, 그 결과 규모가 작아 관리에서 밀려 있던 자산이 가장 먼저 표적이 됐다.

따라서 앞으로의 방어는 무엇을 가장 단단히 지킬 것인가보다, 공격자가 어디서부터 들어올 수 있는가를 먼저 묻는 데서 시작해야 한다. 외부 노출 자산 관리, API 인증과 인가, 위탁 업체 연결 통제, 다중인증, 신속한 패치, 데이터 최소화는 새로운 대책이 아니다. 그러나 이번 사고들은 이 기본 조치가 외곽 접점까지 빠짐없이 적용되지 않으면 AI 가 그 빈틈을 먼저 찾아낸다는 사실을 분명히 보여줬고, 본 문서에서 AI 기반 자동화 공격과 외곽 접점 침해에 초점을 맞춘 이유도 여기에 있다.

분석 보고서
ARTEX 사이버 위협 심층 분석

이 보고서는 저작권법에 의해 보호 받는 저작물로서 영리목적의 무단전재와 무단복제를 금합니다.

이 보고서의 내용의 전부 또는 일부 인용, 가공 시 안랩에서 발간된 보고서임을 밝혀 주시기 바랍니다.

* 이 보고서에 수록된 내용 또는 배포에 관한 모든 문의는 안랩(031-722-8000)으로 부탁드립니다.

㈜안랩

경기도 성남시 분당구 판교역로 220 (우) 13493

홈페이지 : www.ahnlab.com

대표전화 : 031-722-8000 | 구매문의 : 1588-3096 | 팩스 : 031-722-8901

© 2026 AhnLab, Inc. All rights reserved.

AhnLab