SK 쉴더스

Threat Intelligence Report

EQST INSIGHT

EQST(이큐스트)는 'Experts, Qualified Security Team' 이라는 뜻으로
사이버 위협 분석 및 연구 분야에서 검증된 최고 수준의 보안 전문가 그룹입니다.

2026
01

SK 쉴더스

Contents

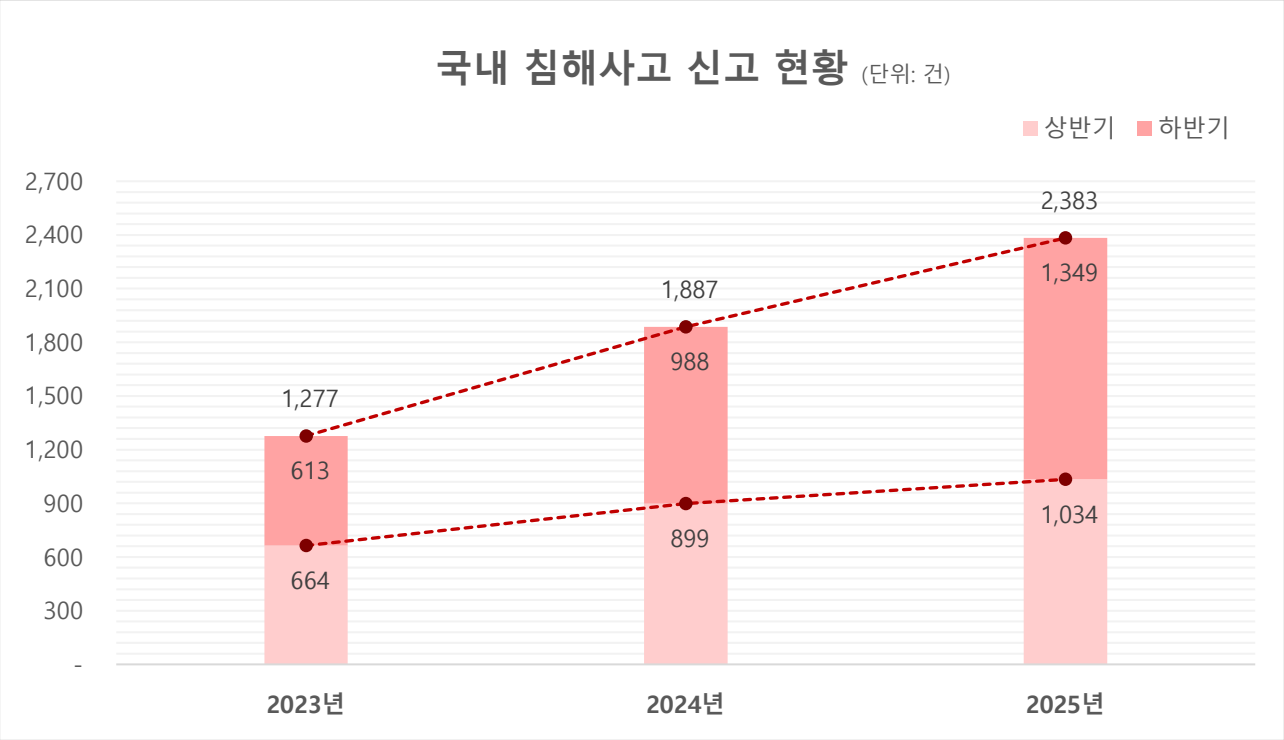
Headline

선제적 보안과 레드팀 기반 사이버 면역 체계 구축 전략

EQST 전략사업팀 이건희 선임

■ 사이버 보안을 위한 두가지 패러다임, 선제적 보안과 사이버 복원력

디지털 전환이 가속화되고, AI 기술이 고도화됨에 따라 사이버 공격은 더욱 정교하고 지능적인 양상으로 진화하고 있으며, 이에 따라 침해사고 발생 빈도 역시 해마다 증가하고 있다. 2023 년부터 2025 년까지 국내 침해사고 신고 건수는 각각 1277 건, 1887 건, 2383 건으로 늘어나고 있으며, 이러한 추세는 앞으로도 지속될 것으로 예견된다.



출처 : 과학기술정보통신부·한국인터넷진흥원

그림 1. 국내 침해사고 신고 현황(2023 년~2025 년)

지금까지 사이버 보안의 초점은 위협 발생 이후 이를 탐지하고 대응하는 것에 맞춰져 왔으며, 침해사고 발생시 스트레스를 최소화하고, 신속한 복구를 통해 업무의 연속성을 확보할 수 있는 역량인 사이버 복원력(Cyber Resilience)의 중요성이 강조돼 왔다.

하지만 피해를 최소화하려는 노력만으로는 충분치 않다. 사고 발생 이후에 행동하는 보안 체계에서는 침해사고가 빈번해질수록 피해 규모와 복구 비용이 비례하여 증가할 수밖에 없기 때문이다. 또한 침해사고의 발생은 그 자체로도 조직의 평판을 감소시키며, 이는 결국 고객 이탈과 매출 감소로 이어져 장기적인 악영향을 끼치게 된다.

최근에는 사이버 공격이 현실화되기 전에 잠재적 취약점과 침투 경로를 사전에 식별·보완함으로써 침해사고 발생 가능성을 최소화할 수 있는 '예방' 중심의 사이버 보안 전략인 선제적 보안(Preemptive Cybersecurity)이 새로운 패러다임으로 주목받고 있다. 가트너는 2026 년 10 대 기술 전략 트렌드 중 하나로 선제적 보안을 꼽았으며, 2030 년까지 선제적 보안을 위한 지출이 전체 보안 지출의 절반 수준까지 확대될 것으로 전망하고 있다.

선제적 보안과 사이버 복원력은 각각 공격적 성격과 방어적 성격을 지니고 있어 서로 상반된 개념으로 보일 수 있지만, 이들을 완전히 분리해서 바라보는 시각은 올바르지 않다. 선제적 보안 중심의 공격적 보안 전략은 위협을 최소화할 수는 있지만 완전히 제거하기란 현실적으로 어려운 반면, 사이버 복원력 중심의 방어적 보안 전략은 피해를 최소화하고 재발을 방지할 수 있지만 예상을 벗어난 위협을 사전에 대비하기는 어렵기 때문이다.

구분	선제적 보안	사이버 복원력
관점	공격자 관점	방어자 관점
목적	위협 예측 및 사전 차단(예방)	신속한 공격 탐지와 복구(탐지 및 대응)
장점	침해사고 발생 가능성 최소화 가능	침해사고 발생에 따른 피해 최소화 가능
단점	위협의 완전한 제거 어려움	예상하지 못한 위협에 대비하기 어려움

표 1. 선제적 보안과 사이버 복원력의 차이점

이제는 이 두 가지 패러다임을 상호 보완적으로 결합해 나가야 한다. 이러한 접근 방식을 통해 조직은 사이버 공격에 유연하게 대처할 수 있을 뿐만 아니라, 침해사고 발생 가능성 자체를 최소화함으로써 전방위적 사이버 면역체계를 갖출 수 있을 것이다.

■ 선제적 보안을 위한 보안 조직, 레드팀

아직까지는 국내에서 '레드팀(Red Team)'이라는 개념이 다소 생소한 것이 사실이다. 그럼에도 불구하고 최근 대기업을 중심으로 레드팀을 도입하려는 움직임이 확산되고 있으며, 많은 매체에서는 선제적 보안을 현실화하기 위한 방안으로 레드팀 도입을 언급하고 있다.

물론 레드팀이 선제적 보안을 위한 핵심 요소인 것은 분명하다. 하지만 레드팀이 무엇이고 어떤 활동을 하며, 어떻게 운영해 나가야 하는지를 이해하지 못한 채 도입만 서두르게 된다면 레드팀 도입이라는 선택이 자칫하면 선제적 보안 체계를 갖추고 있다는 느낌만 주게 되는 '보안 극장(Security Theater)'으로 전락할 수도 있음을 간과해서는 안된다.

레드팀을 한줄로 정의하자면, "공격자의 사고(Adversarial Thinking)"를 기반으로 현재 조직의 공격 표면(Attack Surface)에서 발생 가능한 위협을 사전에 식별하고, 취약성을 보완하기 위한 현실적인 방향성을 제시하는 보안 조직"이다.

레드팀은 컴플라이언스 대응을 위한 취약점 진단(Vulnerability Assessment)뿐만 아니라, 침투 테스트(Penetration Test), 피싱 캠페인(Phishing Campaign), 물리적 보안 테스트(Physical Security Test) 등 기술적·관리적·물리적 공격 표면(Attack Surface)에 대한 보안 활동을 수행하며, 운영 주체에 따라 인하우스(In-house)형태의 레드팀과 아웃소싱(Outsourcing)형태의 레드팀으로 분류할 수 있다.

먼저 인하우스 레드팀은 조직 내부에서 상시적으로 운영되는 팀으로, 조직 내부 시스템과 비즈니스 프로세스에 대한 깊은 이해를 보유하고 있다. 이들은 조직의 내부 구성원으로 구성되며 외부에 공개하기 민감한 시스템에 대한 보안 점검이나 타 부서와의 긴밀한 협조가 필요한 보안 활동 등 높은 신뢰도가 요구되는 과업을 효과적으로 수행할 수 있다. 하지만 이들은 조직 내부의 문화나 의사결정 구조에 영향을 받기 쉬워 편향이 발생할 가능성이 있고, 외부자 시각에서의 비판적 접근이 어려울 수도 있다는 한계점이 존재한다.

반면 아웃소싱 레드팀은 조직 외부의 전문가 집단으로 이루어져, 장기 또는 단기 프로젝트 성으로 운영된다. 이들은 다양한 고객과 산업군을 대상으로 한 풍부한 수행 경험과 실전형 침투 전략을 바탕으로 외부자 관점에서 조직의 보안 체계를 객관적으로 평가할 수 있다는 장점이 있다. 하지만 아웃소싱 레드팀은 외부 인력으로 구성되기 때문에 보안 활동 수행 과정에서 획득한 인사이트를 조직에게 내재화하기 위한 지식 이전이 불완전할 수 있으며, 인하우스 레드팀에 비해 신뢰하기 어려울 수 있다는 단점도 있다.

구분	인하우스 레드팀	아웃소싱 레드팀
구성	내부 직원	외부 전문가
장점	높은 조직 이해도와 신뢰성 보유	실전형 침투 전략과 창의적 시각 보유
단점	편향 발생 가능, 비판적 접근 어려움	지식 이전 불완전, 비교적 낮은 신뢰도

표 2. 인하우스형 레드팀과 아웃소싱형 레드팀의 차이점

레드팀 도입을 고려하고 있다면, 조직의 보안 성숙도를 충분히 고려하는 것이 중요하다. 체계적인 보안 운영 환경이 확보되지 않은 상태에서는 레드팀의 도입 효과를 극대화하기 어려우며, 보안 역량이 충분히 성숙하지 않은 조직에는 레드팀 도입이 오히려 비효율적인 보안 지출 요인으로 작용할 수 있기 때문이다.

만약 조직의 보안 성숙도가 낮거나 자체적으로 평가하기 어려운 상태라면 아웃소싱 레드팀 서비스를 통해 공격 표면 관리 역량을 강화한 뒤 인하우스 레드팀을 도입하는 것이 좋다.

반면, 보안 성숙도가 높으며 이미 인하우스 레드팀을 운영중인 조직이라면 아웃소싱 레드팀을 보다 적극적으로 활용할 수도 있다. 앞서 설명했듯이 아웃소싱 레드팀은 외부 전문가로 이루어져 내부 조직이 가진 지식적 한계를 보완할 수 있다. 이에 입각하여 아웃소싱 레드팀 활동을 통해 최신 위협 동향과 공격 트렌드를 반영한 위협 시나리오를 도출하고 인하우스 레드팀은 이를 조직의 환경과 운영 특성에 맞게 최적화하는 하이브리드 형태의 레드팀을 운영하게 된다면 레드팀 활동의 효율을 크게 향상시킬 수 있을 것이다.

■ 레드팀의 구성과 역할

레드팀은 역할과 책임에 따라 리더(Leader), 오퍼레이터(Operator), 그리고 엔지니어(Engineer)로 구성할 수 있다.

레드팀 리더는 레드팀 활동을 총괄하며, 기술적 실행보다는 주요 이해 관계자들과의 소통과 전략적 의사결정에 집중하는 관리자의 역할을 담당한다. 리더는 레드팀 활동이 원활하게 수행될 수 있는 기술적·문화적 환경을 조성하고, 레드팀과 이해관계자들이 효과적으로 소통할 수 있도록 지원해야 한다. 이에 따라 리더에게는 전략적 사고, 팀 관리 능력, 협력적 사고방식과 효과적인 커뮤니케이션 스킬이 필수적으로 요구된다.

레드팀 오퍼레이터는 레드팀 활동의 최전선에서 리더의 지시 아래 실제 공격 테스트를 수행하는 역할을 한다. 오퍼레이터는 모의 공격의 전 과정에 걸친 실제 공격자들의 TTPs(Tactics, Techniques, Procedures)를 정확히 이해하고, 조직의 보안 상태에 가장 적합한 공격을 수행할 수 있어야 한다. 그렇기 때문에 이들에게 가장 중요시되는 역량은 깊이 있는 오펜시브 시큐리티(Offensive Security) 지식과 네트워크 및 데이터

분석 능력이다. 뿐만 아니라 이들은 수행 결과를 이해관계자들이 이해하기 쉽게 설명하거나 문서화할 수도 있어야 하기 때문에 소프트 스킬도 이들이 보유해야 할 중요한 역량 중 하나다.

레드팀 엔지니어는 팀의 규모에 따라 오퍼레이터가 겸하는 경우도 있다. 이들은 레드팀 오퍼레이터가 실전에서 활용할 수 있는 공격 툴을 개발하고, C2(Command & Control) 프레임워크와 같은 공격자 인프라를 설계·구축하는 핵심 개발 전문가다. 이들은 조직의 방어 체계를 넘어설 수 있는 정교한 침투 도구를 반복적으로 개발할 수 있어야 하기 때문에, 매우 높은 수준의 개발 능력과 오펜시브 시큐리티 지식이 필수적으로 요구된다.

역할	필요 역량
레드팀 리더	전략적 사고, 팀 관리 능력, 협력적 사고 방식 및 커뮤니케이션 스킬
레드팀 오퍼레이터	오펜시브 시큐리티 지식, 네트워크 및 데이터 분석 능력
레드팀 엔지니어	오펜시브 시큐리티 지식, 높은 수준의 개발 능력

표 3. 레드팀 구성원 별 필요 역량 예시

■ 레드팀 활동 수행 절차

레드팀 활동 수행 절차는 계획(Planning)과 수행(Execution), 보고(Reporting) 세 가지 단계로 구분할 수 있다.

레드팀 활동 착수를 위한 첫 번째 과정인 계획 단계에서는 레드팀 리더와 이해관계자들간의 사전 미팅을 통해 목표와 정의, 성공 기준 등 레드팀 활동 계획 수립을 위한 제반 사항에 대한 협의가 이루어지며, 이 과정에서 레드팀과 이해관계자 모두가 준수해야 할 일종의 수행 지침인 교전 규칙(Rules of Engagement, ROE)에 협의사항을 빠짐없이 기록하게 된다.

미국 국립표준기술연구소(National Institute of Standards and Technology, NIST)의 NIST 800-115 가이드에서는 ROE 템플릿에 포함되어야 할 내용들을 아래와 같이 정의하고 있다.

항목	내용(예시)
Purpose	문서의 목적, 보안 테스트 대상 조직, 실행 조직, 목표
Scope	테스트 범위 및 유형, 제외 범위 및 산출물
Assumptions and Limitations	제약사항과 가정사항
Risks	내재적 위험 및 완화 기법
Document Structure	ROE 문서의 구조
Personnel	보안 테스트와 연관된 모든 조직과 인원 목록
Test Schedule	테스트 스케줄과 마일스톤
Test Site	테스트가 허가된 장소, 제한 구역
Test Equipment	보안 테스트에서 사용될 하드웨어, 소프트웨어, 미 허가 장비
General Communication	의사 소통 계획(일정, 장소, 주기 등)
Incident Handling and Response	사고 대응 및 복구 절차
Target System/Network	테스트 대상(시스템과 네트워크 대역 등)
Nontechnical Test Components	비기술적 테스트 활동(인터뷰, 리뷰 등)
Technical Test Components	기술적 테스트 활동(네트워크 스캔, 정보 수집, 침투테스트 등)
Data Handling	테스트 데이터 수집, 저장, 전송, 파기 방법과 절차
Reporting	보고서 요구사항, 보고 주기
Signature Page	이해관계자 및 조직 고위 경영진(CSO, CISO, CIO) 서명

출처 : 미국 국립표준기술연구소(National Institute of Standards and Technology, NIST)

표 4. ROE 항목 예시

ROE 작성과 서명이 완료되면 작성된 내용을 준수하여 레드팀 활동을 수행하게 된다. 이 단계에서 레드팀 오퍼레이터는 정찰(Reconnaissance), 초기 침투(Initial Access), 실행(Execution), 지속성 확보(Persistence), 유출(Exfiltration)등 다양한 TTPs 를 활용하여 목표를 달성하기 위한 모의 공격을 수행하게 된다. 또한 레드팀 엔지니어는 이 과정에서 오퍼레이터와 협업하며 효과적인 목표 달성을 위해 지원한다.

수행 과정에서 가장 핵심적인 요소 중 하나는 바로 오퍼레이터 로그(Operator Log) 기록이다. 오퍼레이터 로그는 모의해킹 과정의 투명성과 재현성, 후속 분석을 위한 핵심 산출물로, 타임스탬프와 행위자, 이벤트 유형, 결과 등의 내용이 포함된다.

만약 수행 과정에서 보안 체계가 견고하여 초기 침투가 불가능하다고 예상될 경우, 레드팀 리더의 판단과 ROE 에 작성된 예외 처리 절차에 따라 조직의 특정 자산(내부 서버 또는 계정 등)이 이미 침해되었다고 가정하고 후속 공격(Post-Exploitation)을 집중적으로 테스트하는 침해 사고 가정 시나리오(Assumed Breach Scenario) 형태로 전환하여 진행할 수도 있다.

필드	값
Timestamp	20260118_121411
Source	10.10.10.22
Destination	192.168.1.12
Target	TARGET-LINUX01
Event Type	Active Scanning
Command	nmap -sT -Pn 192.168.1.12
Result	Ports: 80/open, 443/open, 8443/open

표 5. Operator Log 예시

모든 테스트가 종료되면 레드팀 리더는 오퍼레이터의 활동 결과를 분석하고 개선 사항을 정리하여 보고서를 작성한다. 보고서에는 일반적으로 경영진을 위한 핵심 요약(Executive Summary)부터 식별된 취약점과 성공한 위협 시나리오, 위협 평가 결과와 함께 탐지 및 대응 성과를 정량적으로 표현할 수 있는 핵심 성과 지표(Key Performance Indicator)가 반드시 포함되어야 한다.

항목	내용(예시)
ASR(Attack Success Rate)	전체 공격 횟수와 성공한 공격 횟수에 따른 성공률
Detection Coverage	전체 공격 횟수와 탐지된 공격 횟수에 따른 탐지 비율
MTTC (Mean Time to Compromise)	공격 시작부터 목표에 침투를 성공할 때까지 걸린 시간
MTTD (Mean Time to Detect)	공격자가 침투한 시점부터 이를 탐지할 때까지 걸린 시간
MTTR (Mean Time to Respond):	침투 탐지 후 대응 완료할 때까지 걸린 시간

표 6. 핵심 성과 지표(KPI) 예시

보고서 작성이 완료되면 경영진과 운영팀, 탐지와 대응을 담당하는 블루팀(Blue Team) 등 이해관계자들에게 이를 배포하고, AAR(After Action Review) 또는 디브리핑(Debriefing) 세션을 개최하여 레드팀 활동을 리뷰한다. 이후 조직에서는 확인된 취약 사항에 대한 보완 조치를 진행하게 되며, 조치가 완료되면 조치 여부를 재 검증하기 위한 후속 활동인 이행 점검(Remediation Verification)을 진행하는 것으로 레드팀 활동이 마무리된다.

■ 침해 및 공격 시뮬레이션을 통한 레드팀과 블루팀의 통합

레드팀 활동은 대부분 공격 행위가 단발성으로 이루어진다. 하지만 만약 위협 시나리오를 반복적으로 테스트할 수 있다면 블루팀은 위협 시나리오에 대한 탐지 능력이나 대응 속도를 점진적으로 향상시킬 수 있을 것이다.

침해 및 공격 시뮬레이션(Breach and Attack Simulation, BAS)은 이러한 한계점을 극복할 수 있는 레드팀 활동으로, TTPs 단위로 구분된 위협 시나리오를 반복적으로 시뮬레이션 할 수 있도록 구조화하고 필요에 따라 이를 재시뮬레이션하는 방식으로 진행된다.

BAS 는 위협 시나리오를 반복해서 테스트할 수 있기 때문에, 침해사고 사후 검증 목적으로도 활용할 수 있다. 만약 조직에서 침해사고가 발생했다면 사고 원인을 분석하고 재발 방지를 위한 대책을 수립할 것이다. 이때 재발 방지 대책이 적용된 이후 BAS 를 활용하여 침해사고 시나리오를 시뮬레이션한다면, 적용된 보안 대책의 실효성을 평가할 수 있을 것이다.

BAS 는 앞서 설명한 레드팀 활동 이행 점검과 같이 수동으로 진행할 수도 있지만, 자동화 BAS 플랫폼을 도입한다면 점검 과정을 완전히 자동화할 수 있어 생산성을 극대화시킬 수 있다. 대부분의 자동화 BAS 플랫폼들은 다양한 보안 시스템들과 연동할 수 있는 기능과 탐지 성공률, 대응 시간 등의 핵심 지표를 실시간으로 파악할 수 있는 직관적인 인터페이스를 제공한다. 이는 곧 조직이 SOC(Security Operations Center)나 MSSP(Managed Security Service Platform)/MDR(Managed Detection and Response)을 고도화하는 데에도 큰 힘이 되며, 경영진은 이를 참고하여 ROI(Return on Investment)를 측정하거나 전략적인 의사결정을 내리는 데 도움을 받을 수도 있다.

이처럼 BAS 를 활용하면 선제적 보안과 사이버 복원력, 레드팀과 블루팀을 효과적으로 통합할 수 있다. 레드팀은 발생 가능성이 존재하는 위협 시나리오를 도출한 뒤 BAS 를 통해 시나리오를 구조화하고 반복적인 시뮬레이션 환경을 구축한 뒤, 블루팀은 이를 통해 탐지·대응 체계를 개선해 나간다. 그리고 이러한 과정을 반복함으로써, 레드팀과 블루팀이 통합된 퍼플팀(Purple Team) 운영체계를 구축해 나갈 수 있을 것이다.

그림 2. 지속적 공격 검증 절차

하지만 이면에는 현실적인 어려움 또한 존재한다. 레드팀은 위협 시나리오의 현실성을 검증하기 위해 적극적이고 실험적으로 접근하고자 하는 성향이 강한 반면, 블루팀은 실시간 보안 모니터링, 탐지와 사고 대응 등 운영의 안정성과 연속성 유지에 초점이 맞춰져 있어 보수적인 성향이 강하다. 따라서 블루팀은 레드팀의 공격 시나리오가 실제 운영 환경에 영향을 미칠 수 있다는 우려로 정보 공유와 협력을 주저하기도 하는데, 이러한 구조적 차이가 두 팀의 통합을 어렵게 하는 주요한 원인이다.

이러한 어려움을 해결하기 위해서는 경영진의 강력한 의지와 주도적인 움직임이 필요하다. "보안의 사각지대를 확인하고 개선하는 것이 우리의 목적"이라는 메시지를 조직 전반에 명확히 전달하고, 레드팀과 블루팀 협업을 적극적으로 지원하는 문화가 조성되어야 한다. 물론 단기간에 완벽한 협업 환경을 구축하기는 어려울 수 있지만 단계적인 접근과 지속적인 노력을 통해 이를 실현해 나가는 것이 바람직하다.

■ 맺음말

흔히 서로 정반대에 서 있다고 여겨지는 것들이 실제로는 서로를 보완하며 더 강한 조화를 이루는 경우가 많다. 예를 들어, 바람과 돛의 관계가 그렇다. 바람은 배를 흔들고 방향을 바꾸게 하지만, 돛이 없다면 그 어떤 배도 앞으로 나아갈 수 없다. 바람이 위협처럼 보여도 돛과 함께할 때는 추진력이 된다.

마찬가지로, 미리 위험을 예측하고 대비하는 '선제적 보안'과, 공격을 받아도 빠르게 회복할 수 있는 '사이버 복원력'은 서로를 밀어내는 힘이 아니다. 이들은 함께 있을 때야 비로소 조직을 올바른 방향으로 나아갈 수 있게 만들어주는 동력이 될 것이며, 이들이 조화롭게 어우러짐으로써 조직은 '예방', '탐지', '대응'이 유기적으로 연결된 사이버 면역 체계를 갖출 수 있을 것이다.

SK 쉴더스는 국내 최대 규모 화이트 해커 그룹 EQST(Experts, Qualified Security Team)를 통해 고객이 '선제적 보안' 과 '사이버 복원력'의 통합을 위해 나아가야 할 방향성을 제시할 수 있는 전문 모의해킹 컨설팅 서비스를 제공하고 있다.

EQST 는 20 여년간 공공·기업·금융·제조 등 산업군별 B2B 사업을 수행해왔으며, New ICT 분야를 선도하는 내부 연구조직에서 분석한 최신 사이버 보안 트렌드와 SK 쉴더스 통합 관제 플랫폼인 시큐디움(Secudium)의 방대한 위협 인텔리전스(Threat Intelligence, T.I)를 바탕으로 전문성과 트렌드, 노하우가 결합된 EQST 위협 시나리오 DB(EQST Threat Scenario Database)와 자체적인 모의해킹 방법론을 설계·구현하였다.

그리고 EQST 는 이를 기반으로 각종 컴플라이언스에 대응 가능한 취약점 진단 컨설팅부터 ASM(Attack Surface Management)과 모의해킹을 결합한 보안 점검 및 실전형 아웃소싱 레드팀 서비스까지 폭넓은 보안 컨설팅 서비스를 제공하고 있다. 만약 독자가 조직의 사이버 보안 성숙도 향상을 위해 '선제적 보안 전략'을 고려하고 있다면, 독자적인 전문성을 보유한 EQST 보안 컨설팅 서비스를 통해 선제적 보안을 위한 최적화된 지원을 받을 수 있을 것이다.

■ 참고문헌

[1] Gartner, "Gartner Identifies the Top Strategic Technology Trends for 2026"

[2] Gartner, "Gartner Top 10 Strategic Technology Trends for 2026"

[3] 과학기술정보통신부·한국인터넷진흥원, "25년 사이버 위협 하반기 동향 및 26년 전망"

[4] NIST, "Technical Guide to Information Security Testing and Assessment"

Keep up with Ransomware

Sinobi 랜섬웨어와 Lynx 그룹과의 연계 정황 분석

■ 개요

2025 년 12 월 랜섬웨어 피해 사례 수는 지난 11 월(740 건) 대비 약 15% 상승한 854 건으로 집계됐다.

프랑스 내무부는 지난 12 월 내부 이메일 서버가 사이버 공격을 받아 다수의 이메일 계정과 기밀문서에 무단 접근이 이루어졌다고 밝혔다. 조사 결과, 해커는 경찰관의 이메일 계정을 탈취해 최초로 침투한 후 평문으로 공유된 비밀번호를 확보한 것으로 밝혀졌다. 확보한 비밀번호로 내부 인증 시스템을 침투하는 과정에서, 경찰 데이터베이스에도 접근한 정황이 확인됐다. 한편 해킹 포럼인 BreachForums 에서 운영자로 추정되는 Indra 계정에 이번 공격을 주장하는 글이 올라왔다. 프랑스 당국이 ShinyHunters 와 연관된 인물들을 체포한 것에 대한 보복을 언급하며, 내무부 침해와 대규모 데이터 접근을 과시하는 내용이 포함됐다. 게시글에는 약 1,640 만 건 규모의 데이터 접근을 시사하는 내부 시스템 검색 결과에 대한 스크린샷과 일부 경찰 기록 관련 신원 정보가 담긴 이미지와 함께 프랑스 정부를 향해 일주일 내로 협상하라는 요구 메시지도 담겼다. 이번 사건은 계정 탈취와 더불어 기본적인 보안 원칙을 준수하지 않아 발생한 것으로 알려졌다. 계정 관리와 인증 체계를 강화하고 접근 권한 최소화 및 인증, 로그인 로그 기반 이상 징후 탐지 체계를 고도화할 필요가 있음을 시사했다.

공격자가 직접 침투하는 방식이 아닌, 보안 사고 대응 분야 종사자가 공격자와 공모한 정황이 사법당국 수사로 드러난 사례가 확인됐다.

미국 법무부는 2025 년 12 월, BlackCat(ALPHV) 랜섬웨어 조직과 공모한 미국인 Ryan Goldberg 와 Kevin Martin 을 미국 내 피해자를 공격해 금전을 갈취한 혐의로 기소했다. 수사 내용에 따르면 Ryan Goldberg 는 사이버 보안 기업에서 사고 대응 업무 담당자였고, Kevin Martin 은 랜섬웨어 사고 대응을 지원하는 기업에서 협상 업무를 맡았던 것으로 확인됐다. 이들은 보안 기업에서 습득한 사이버 보안 훈련과 경험을 바탕으로 기업의 네트워크를 암호화하고 몸값을 요구했다. 수익 분배의 경우 사전에 합의한 뒤 피해자가 지급한 몸값 중 일부를 운영진에게 전달하는 방식으로 공모 관계를 유지한 것으로 밝혀졌다.

2025 년 12 월 5 일 React2Shell 취약점(CVE-2025-55182)을 악용한 Weaxor 랜섬웨어 배포 사례도 있었다. 해당 취약점은 인증 절차 없이 취약한 서버에서 원격 코드 실행이 가능하며, 공격자는 PowerShell 을 통해 CobaltStrike[1] 기반 C&C[2] 서버를 확보한 뒤, Weaxor 랜섬웨어를 실행해 파일을 암호화했다. 공격자는 보안 업데이트가 적용되지 않은 취약한 시스템을 노렸으며, 특히 해당 취약점이 12 월 3 일 공개된 뒤 불과 이틀 만인 12 월 5 일에 악용된 점을 고려하면, 소프트웨어 및 보안 장비에 대한 신속한 취약점 대응이 필수적이다.

[1] Cobalt Strike: 공격자가 원격 명령 실행, 추가 페이로드 투하, 내부 정찰 및 수평 이동을 수행하기 위해 사용하는 C&C 프레임워크

[2] C&C(Command & Control) 서버: 감염된 시스템과 통신하며 명령 전달, 추가 페이로드 배포, 정보 수집 등 공격자의 원격 제어를 수행하는 서버

■ 랜섬웨어 뉴스

프랑스 내무부 이메일 서버 침해 정황

- 프랑스 내무부의 내부 이메일 서버가 공격을 받아 다수의 계정과 기밀 문서에 대한 무단 접근 발생
- 해커는 경찰관 이메일 계정을 탈취한 뒤 이메일로 주고받은 비밀번호를 확보해 내부 인증 시스템을 우회
- BreachForums에 운영자로 추정되는 계정이 침해를 주장하며 대규모 데이터 접근을 과시하고 협상을 요구

미국 법무부 BlackCat 공모 미국인 2명 유죄 인정 발표

- 미국 법무부 랜섬웨어 조직과 공모해 금전을 갈취한 미국인 2명(Ryan Goldberg, Kevin Martin)이 유죄를 인정했다고 발표
- Ryan Goldberg와 Kevin Martin은 미국 기업 네트워크를 암호화하고 몸값을 요구한 혐의를 인정
- 사고 대응, 협상 업무 경험을 악용해 수익 분배를 합의 후 피해자가 지급한 몸값 일부를 운영진에게 전달 정황 확인

React2Shell 악용 Weaxor 랜섬웨어 배포 사례 확인

- React2Shell 취약점(CVE-2025-55182) 악용으로 취약 서버에서 원격 코드 실행 후 랜섬웨어가 실행된 정황 확인
- 공격자는 침투 직후 PowerShell로 Cobalt Strike 기반 C&C 채널을 확보한 뒤 랜섬웨어를 실행해 파일 암호화를 수행

12월 신생 그룹 7개 등장

- 12월에 등장한 신규 랜섬웨어 그룹 7곳 모두 자체 다크웹 유출 사이트를 운영
- 이 중 Osiris, MS13-089, MintEye를 제외하면 현재까지 유출 사이트에 게시된 피해 사례는 확인되지 않음

그림 1. 랜섬웨어 동향

■ 랜섬웨어 위협

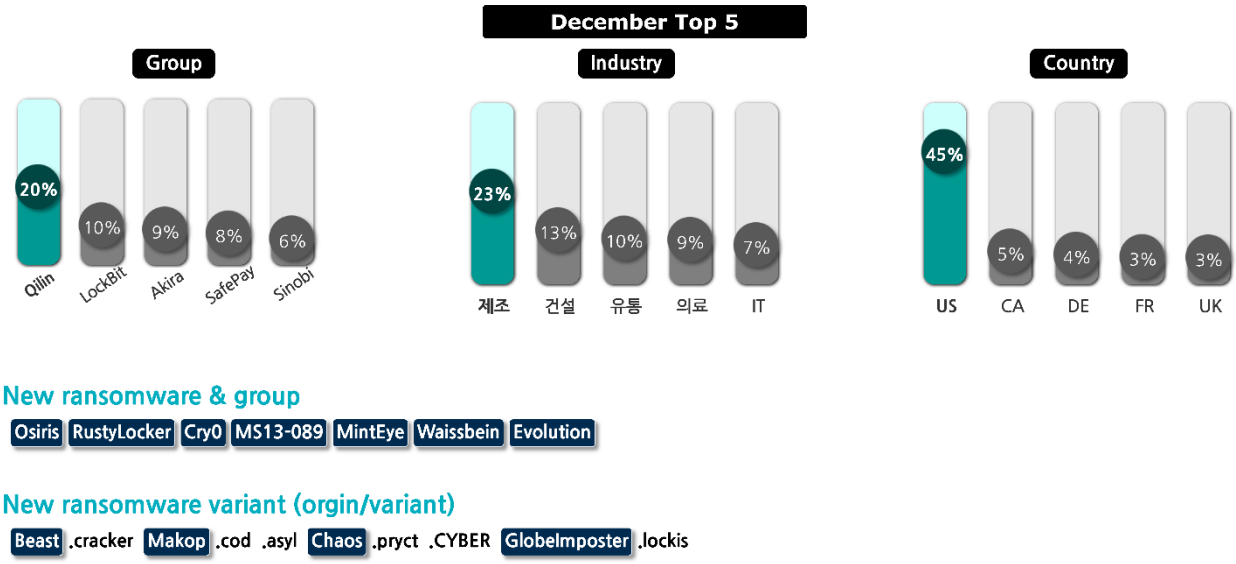


그림 2. 2025 년 12 월 랜섬웨어 위협 현황

새로운 위협

12 월에는 총 7 개의 신규 랜섬웨어 그룹이 등장했다. 이들 모두 다크웹 유출 사이트를 운영하지만, 현재까지 피해 게시물을 업로드한 사례는 일부에 그친다. Osiris 는 1 건, MS13-089 는 2 건, MintEye 는 5 건의 피해 게시물을 유출 사이트에 게시했다. 그 외 나머지 그룹은 다크웹 유출 사이트만 개설해 둔 상태로, 피해 사례는 아직 확인되지 않고 있다.

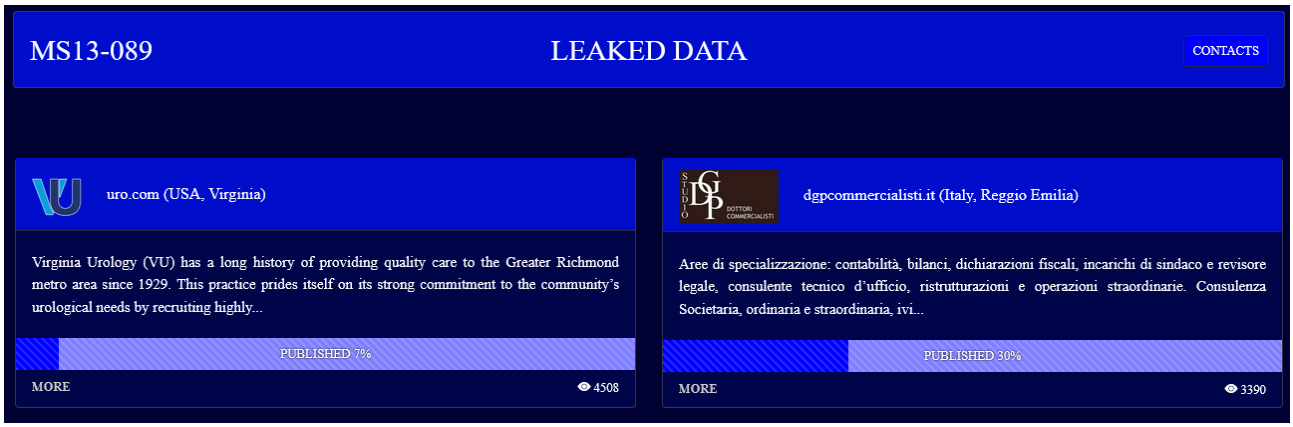


그림 3. MS13-089 의 다크웹 유출 사이트

2025 년 12 월에 등장한 MS13-089 그룹은 현재까지 총 2 건의 피해 게시물을 업로드했다. 또한 데이터를 공개하는 과정에서 공개 진행률을 백분율(%)로 구체적으로 표기해 피해자를 압박하는 특징이 확인된다.

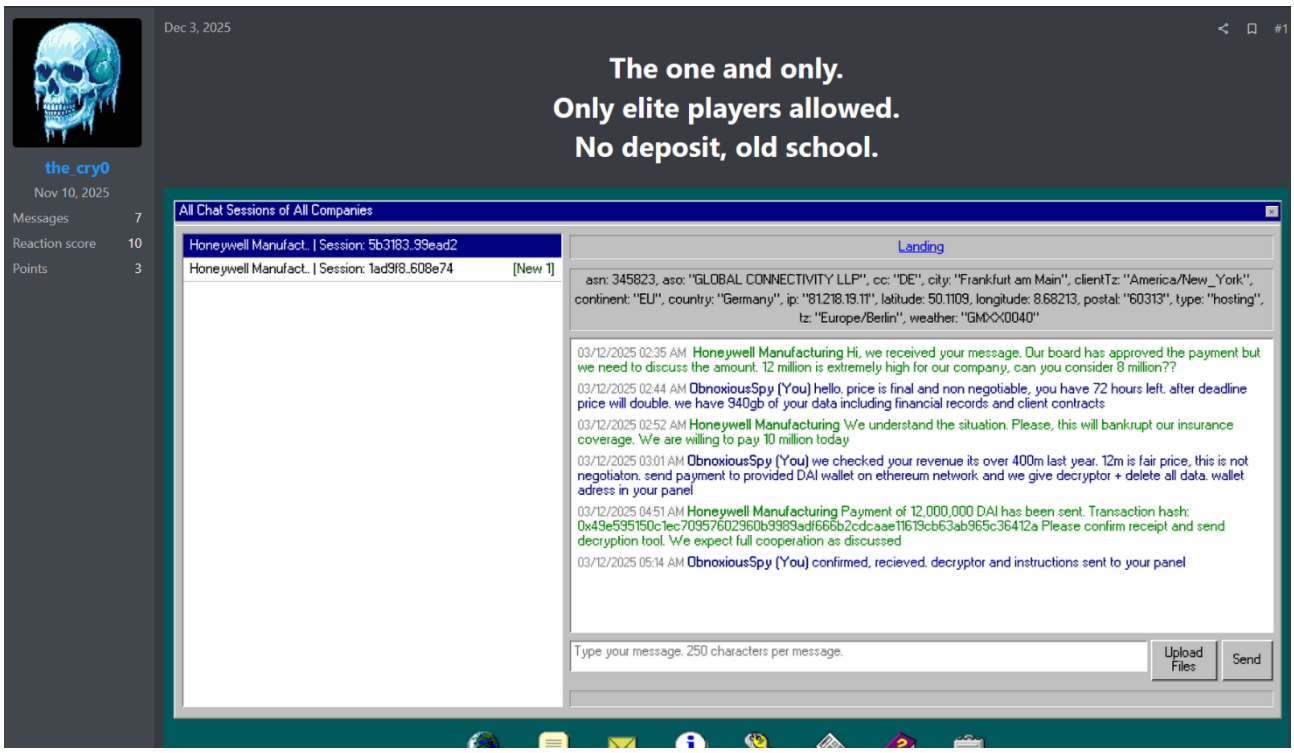


그림 4. Cry0 랜섬웨어 그룹의 RaaS[3] 홍보글

Cry0 그룹은 2025 년 12 월 3 일부터 러시아 해킹 포럼인 RAMP 에서 구성원을 모집하기 시작했다. 이들은 기업별 협상 세션을 관리하는 채팅 패널 화면과 실제 협상 로그를 게시해 운영 능력을 홍보했으나, 현재까지 다크웹 유출 사이트에 게시된 실제 피해자 명단은 확인되지 않고 있다.

[3] RaaS (Ransomware-as-a-Service): 랜섬웨어를 서비스 형태로 제공해서 누구나 쉽게 랜섬웨어를 만들고 공격할 수 있도록 하는 비즈니스 모델

Top5 랜섬웨어

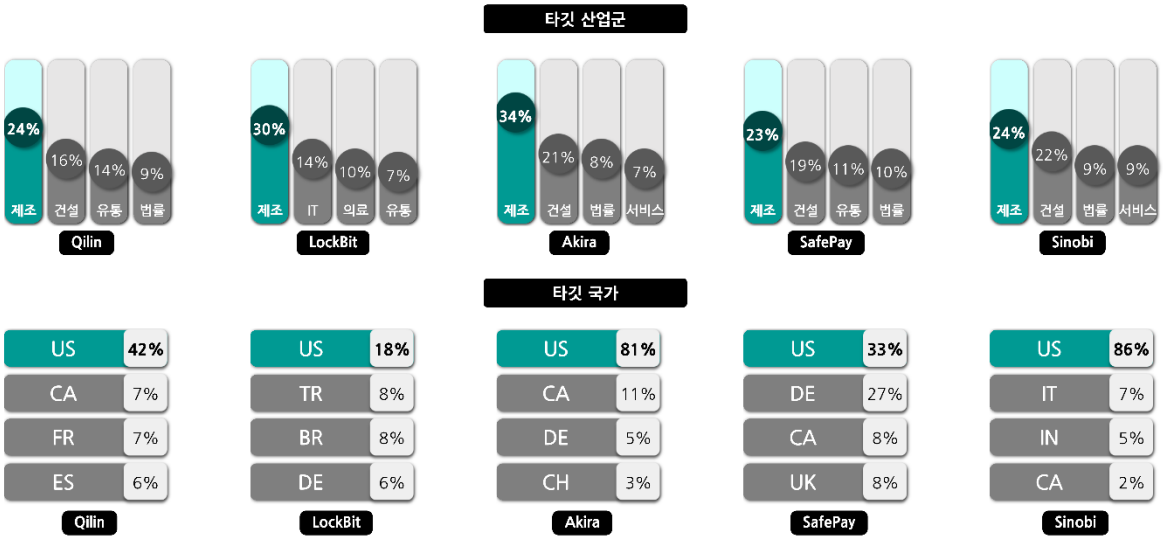


그림 5. 산업/국가별 주요 랜섬웨어 공격 현황

12 월 가장 많은 피해를 발생시킨 랜섬웨어 그룹은 Qilin 그룹이다. 대표적으로 Qilin 그룹은 12 월 29 일 미국 코네티컷주의 Goodwin University 를 공격해 학생 및 직원의 개인정보를 포함한 약 140GB 규모의 데이터를 탈취한 후, 그룹 다크웹 유출 사이트에 공개했다.

LockBit 은 2025 년 9 월 LockBit 5.0 을 출시하며 재정비를 예고했지만, 이후 한동안 뚜렷한 활동은 관측되지 않았다. 그러나 12 월부터는 피해자를 연달아 게시하며 활동 강도를 높였다. 대표적으로 12 월 19 일 미국 아이오와주의 Clarinda Regional Health Center 를 공격한 뒤 환자와 직원들의 개인정보 공개를 예고했다. 12 월 22 일에는 브라질 상파울루의 Colégio Miguel de Cervantes 를 공격해 학생 성적, 행정 서류 등의 정보를 게시했다.

Akira 그룹은 2025 년 SonicWall 방화벽의 접근 제어 미흡 취약점(CVE-2024-40766)을 집중적으로 악용했다. 다수의 사례가 이를 주요 침투 경로로 삼은 것이 확인됐다. 공격자는 해당 취약점을 통해 유효한 VPN 세션을 탈취하거나 자격 증명을 확보해 내부망에 침투한 뒤, 랜섬웨어를 실행하고 데이터를 탈취했다. 이와 별개로 Akira 그룹은 2025 년 12 월 24 일 미국 미네소타주의 전력협동조합인 Agralite Electric Cooperative 를 공격해 사회보장번호와 세금 문서 등 민감정보가 포함된 약 136GB 규모의 데이터를 탈취했다. 또한 12 월 25 일 덴마크의 건축 설계 회사 Friis & Moltke Architects 를 공격해 도면 파일과 계약서 등이 포함된 약 12GB 규모의 데이터를 유출하겠다고 협박했다.

SafePay 그룹은 2025 년 12 월 29 일 영국의 유통 노동조합 Usdaw 를 공격해 조합원의 개인정보와 내부 문건이 포함된 데이터를 탈취했다. 또한 같은 날 아르헨티나의 의료기업 Investigaciones Médicas 를 공격해 환자 검진 데이터와 의료 자료 등의 민감한 정보를 취득했다고 주장했다.

Sinobi 그룹은 2025 년 12 월 7 일 미국의 에너지 산업 서비스 업체 Quality Companies 를 공격해 업무 문서와 계약 정보 등이 포함된 약 40GB 규모의 데이터를 탈취한 뒤, 이를 다크웹 유출 사이트에 공개했다. 또한 미국의 전기 설계 업체 Homestead Electrical Contracting 을 공격해 내부 자료를 유출하겠다고 협박했다.

■ 랜섬웨어 집중 포커스

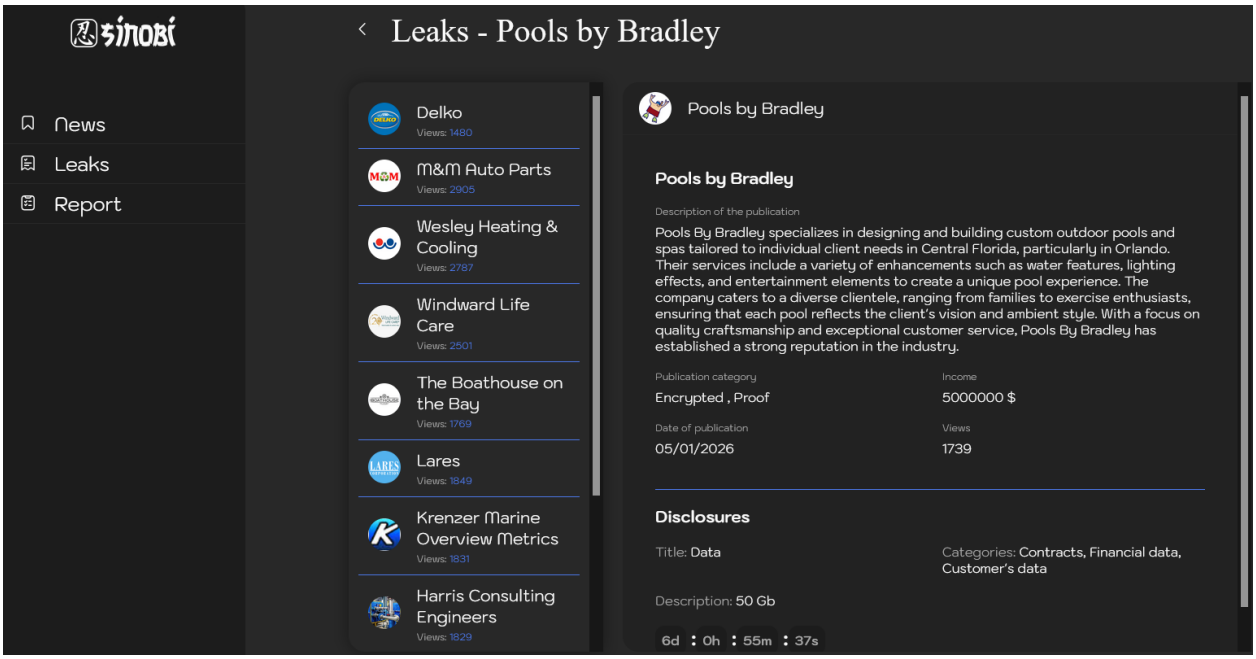


그림 6. Sinobi 그룹의 다크웹 유출 사이트

Sinobi 랜섬웨어 그룹은 2025 년 7 월에 발견되었다. 현재까지 다크웹 유출 사이트에 피해 조직 221 곳을 공개했다. 유출 게시물에는 피해 조직의 이름과 정보, 게시 일자뿐 아니라 탈취 자료의 종류와 샘플 데이터를 함께 제시하고 있다. 파일 암호화와 데이터 탈취 공개를 병행하는 이중 갈취 방식으로 피해 조직을 협박한다. 또한, 피해 조직마다 몸값을 다르게 요구하는데 적게는 500 만(한화 약 72 억) 달러에서 최고 4,400 만 달러(한화 약 640 억)으로 확인돼 평균 약 2,400 만 달러(한화 약 349 억)의 금액 규모다.

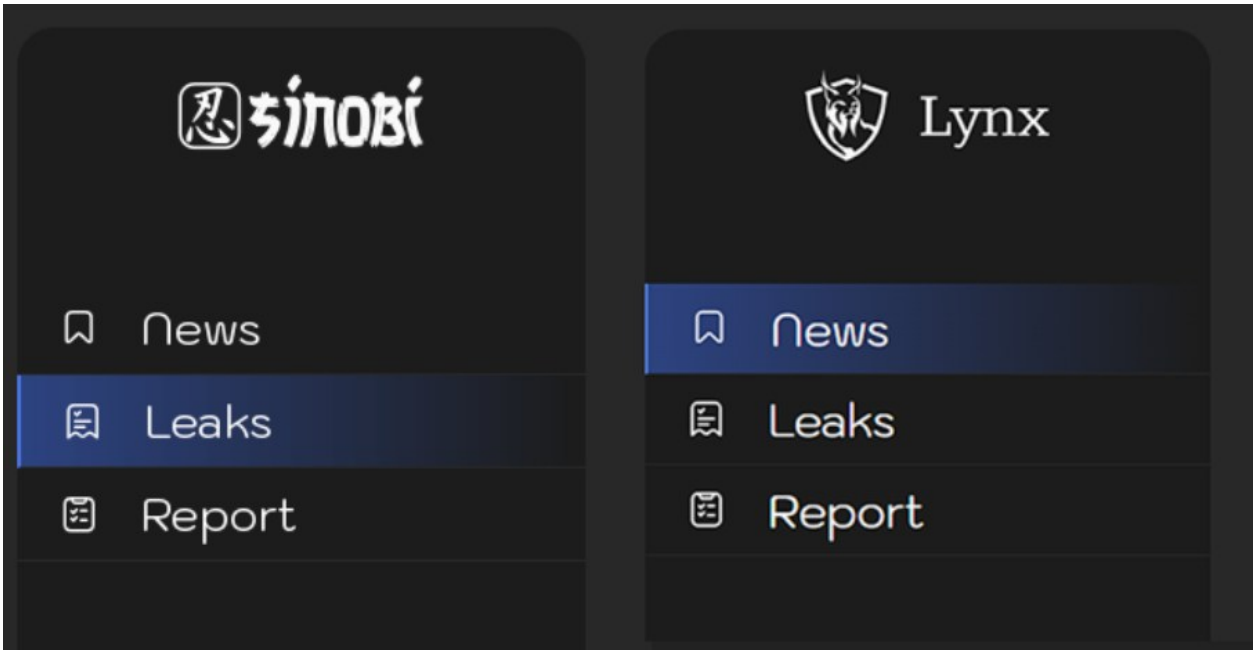


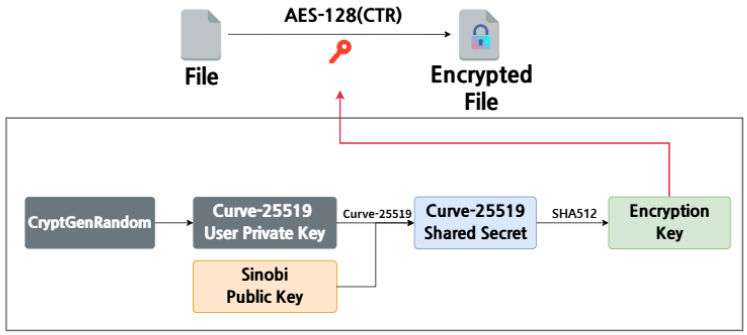
그림 7. 다크웹 유출 사이트 비교(좌: Sinobi, 우: Lynx)

또한 Sinobi 그룹은 INC 소스 코드 거래 정황 이후 등장한 Lynx 계열과 유사성을 보인다. Lynx 는 INC 랜섬웨어에서 확인된 코드 구성 및 기능 흐름과 유사한 특징을 보이며, Sinobi 역시 Lynx 와 암호화 방식 및 실행 인자 구조가 유사한 특징이 확인된다. 즉, INC-Lynx-Sinobi 세 그룹은 코드와 구조적 측면에서 연관성이 있다. 특히, Lynx 와 Sinobi 의 다크웹 유출 사이트는 UI 와 메뉴 구성이 유사해 동일 운영 주체 또는 협력 관계 가능성도 제기된다. 이에 따라 본 보고서는 다가오는 위협에 대비하기 위해 INC–Lynx–Sinobi 간 연관 정황을 종합하고, Sinobi 랜섬웨어의 상세 분석 내용을 공유하고자 한다.

Sinobi Ransomware

암호화 키

Curve-25519 알고리즘으로 암호화 키를 생성하며 AES-128(CTR) 알고리즘으로 암호화

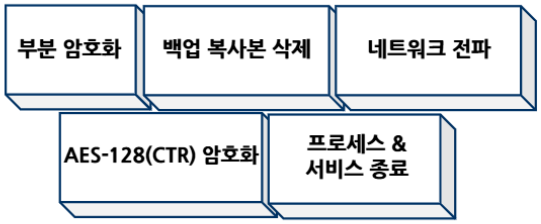


암호화 방식

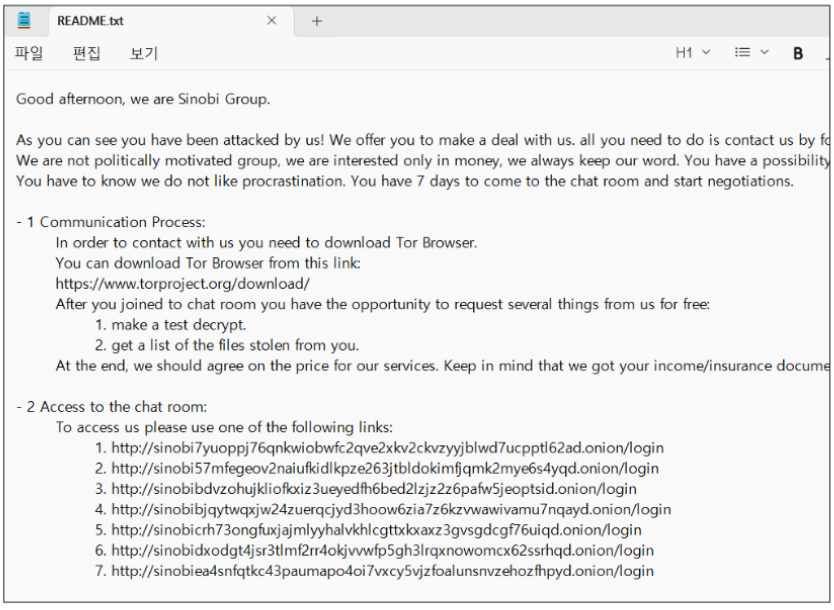
Files ≤ 1MB: 전체 암호화

Files > 1MB: 선택한 모드(entire / fast / medium / Slow)에 따라
→ 파일 크기의 100% / 5% / 15% / 25%만 부분 암호화

특징



랜섬노트


README.txt

파일 편집 보기

Good afternoon, we are Sinobi Group.

As you can see you have been attacked by us! We offer you to make a deal with us. all you need to do is contact us by fo
We are not politically motivated group, we are interested only in money, we always keep our word. You have a possibility
You have to know we do not like procrastination. You have 7 days to come to the chat room and start negotiations.

- 1 Communication Process:
  In order to contact with us you need to download Tor Browser.
  You can download Tor Browser from this link:
  https://www.torproject.org/download/
  After you joined to chat room you have the opportunity to request several things from us for free:
    1. make a test decrypt.
    2. get a list of the files stolen from you.
  At the end, we should agree on the price for our services. Keep in mind that we got your income/insurance docume

- 2 Access to the chat room:
  To access us please use one of the following links:
    1. http://sinobi7yuoppj76qnkwiobwfc2qve2xkv2ckvzyyjblwd7ucpptl62ad.onion/login
    2. http://sinobi57mfegeov2naiufkidlkpze263jtbldokimfjqmk2mye6s4yqd.onion/login
    3. http://sinobibdvzohujkliofkxiz3ueyedfh6bed2lzjz2z6pafw5jeoptsid.onion/login
    4. http://sinobibjqytwqxjw24zuerqcjyd3hoow6zia7z6kzvwawivamu7nqayd.onion/login
    5. http://sinobicrh73ongfuxjajmlyyhalvkhlcgttxkxaxz3gvsgdcgf76uiqd.onion/login
    6. http://sinobidxodgt4jsr3tlmf2rr4okjvvwfp5gh3lrqxnowomcx62ssrhqd.onion/login
    7. http://sinobiea4snfqtkc43paumapo4oi7vxcy5vjzfoalunsnvzehozfhpyd.onion/login

README.txt

변경 확장자

.SINOBI

제작 언어

C/C++

그림 8. Sinobi 랜섬웨어 개요

랜섬웨어 전략

그림 9. 랜섬웨어 공격 전략

Sinobi 랜섬웨어는 다양한 실행 인자를 사용해 암호화 대상이나 방식을 정밀하게 제어할 수 있도록 설계되어 있어, 공격자는 암호화 대상과 전체 혹은 부분 암호화를 수행할 수 있는 모드 등을 설정할 수 있다. 이때 사용되는 인자와 기능은 아래 표와 같다.

인자	설명
--file <filePath>	암호화 대상 파일 지정
--dir <dirPath>	암호화 대상 폴더 지정
--mode <mode>	암호화 모드 설정 (Entire / Fast / Medium / Slow)
--help	실행 도움 메시지 출력
--verbose	로그 출력
--silent	확장자 미변경, 랜섬노트 미생성
--hide-cmd	콘솔창 숨기기
--no-background	배경화면 변경 기능 비활성화
--no-print	랜섬노트 출력 기능 비활성화
--stop-processes	파일 암호화 직전, 대상 파일이 실행중인 경우 프로세스 종료
--kill	특정 프로세스 및 서비스 종료
--encrypt-network	네트워크 공유까지 암호화 대상으로 포함
--load-drives	숨겨진 드라이브 마운트
--safe-mode	안전모드 부팅

표 1. 랜섬웨어 실행인자

Sinobi 랜섬웨어와 INC 및 Lynx 랜섬웨어의 실행 인자를 비교한 결과, 세 랜섬웨어는 상당 부분에서 공통된 기능을 공유하고 있으나 일부 인자 구성에서는 차이가 확인된다. 먼저 암호화 모드 설정 기능은 Sinobi와 INC에서 제공되지만 Lynx에서는 확인되지 않았다. Sinobi는 --mode 인자를 통해 암호화 모드를 설정할 수 있고 INC 역시 --mode 로 암호화 모드를 설정할 수 있다. 반면 Lynx 는 이를 제어하는 인자가 확인되지 않는다. 또한 확장자 변경 및 랜섬노트 생성 생략 기능은 Sinobi 에서만 제공된다. Sinobi 는 --silent 인자를 사용하면 확장자 변경과 랜섬노트 생성을 수행하지 않지만, Lynx 와 INC 에서는 동일한 기능의 인자가 확인되지 않았다. 이러한 차이점을 제외하면, 세 랜섬웨어는 실행 인자 수준에서 제공하는 나머지 주요 기능이 전반적으로 동일한 구성을 보인다.

```
if ( DeviceIoControl(FileW, 0x53C028u, &InBuffer, 0x18u, 0, 0, &BytesReturned, 0) )
{
  if ( byte_140033C78 )
    printf_1(L"[+] Successfully delete shadow copies from %c:/\n", i);
```

그림 10. 백업 복사본 삭제

Sinobi 랜섬웨어는 Lynx 랜섬웨어와 동일한 방식으로, 암호화 작업을 수행하기 전 복구 방해를 위해 백업 복사본을 삭제한다. 이를 위해 DeviceIoControl 함수를 호출해 백업 복사본이 저장되는 공간의 최대 용량을 낮게 재설정한다. 그 결과 시스템은 저장 공간이 부족하다고 판단하고, 공간을 확보하는 과정에서 기존에 생성된 백업 복사본을 자동으로 삭제한다.

또한 랜섬웨어는 --kill 인자를 사용하면 원활한 파일 암호화를 위해 특정 프로세스와 서비스를 종료한다. 종료 대상 프로세스 및 서비스는 아래 표와 같으며, 해당 목록은 Lynx 랜섬웨어에서 확인된 항목과 동일한 것으로 확인된다.

프로세스	서비스
sql, veeam, backup, exchange, java, Notepad	sql, veeam, backup, exchange

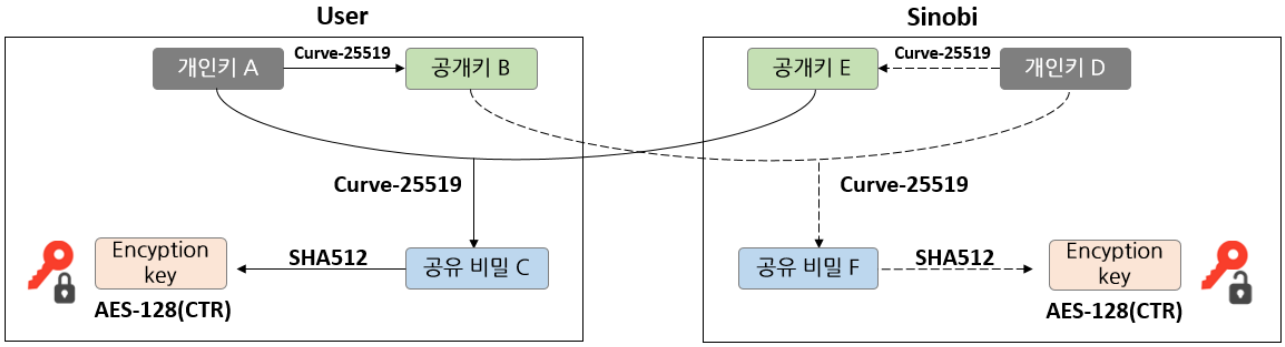
표 2. 프로세스 및 서비스 종료 대상

파일 암호화는 --file 인자 사용 시 특정 파일만 암호화하며, --dir 인자 사용 시에는 지정한 경로에 존재하는 파일만 암호화한다. 두 인자가 모두 입력되지 않으면 예외 대상을 제외한 모든 파일을 암호화하며, 확인된 예외 대상은 아래 표와 같다.

암호화 제외 경로	확장자 및 파일명
Windows, Program Files, Program Files (x86), $RECYCLE.BIN, AppData	.exe, .msi, .dll, .SINOBI, README.txt

표 3. 암호화 예외 대상

Sinobi 랜섬웨어와 Lynx 랜섬웨어의 암호화 예외 대상은 대부분 동일하나, 일부 항목에서 차이가 확인된다. Sinobi 랜섬웨어는 이미 암호화된 파일을 중복으로 암호화하지 않기 위해 “.SINOBI” 확장자를 예외 대상으로 추가한 반면, Lynx 랜섬웨어는 “.lynx” 확장자를 예외 대상으로 사용한다.



공유 비밀 C = 공유 비밀 F

그림 11. 암호화 키 생성 방식

Sinobi 랜섬웨어는 파일 암호화를 위해 파일마다 고유한 32 바이트 개인키(A)를 생성한다. 이후 하드코딩된 공격자의 공개키(B)와 Curve-25519 연산을 수행해 공유 비밀(C)을 생성한다. 이때 공유 비밀이란, Curve-25519 알고리즘에서 양측이 각자의 개인키와 상대방의 공개키만으로 동일하게 계산되는 값을 의미한다. 즉 피해자의 개인키(A)와 공격자의 공개키(B)로 계산한 값(C)은, 공격자의 개인키(D)와 피해자의 공개키(E)로 계산한 값(F)과 동일하며, 이 동일한 값(C, F)을 공유 비밀이라고 한다. 이때 생성된 공유 비밀은 바로 사용되지 않고 SHA-512 로 해시되어 파생키로 변환되며, 최종적으로 이 파생키를 사용해 AES-128(CTR) 알고리즘으로 파일 암호화를 수행한다. 암호화가 완료되면 Sinobi 는 파일의 끝에 피해자의 공개키(E)를 저장한다. 공격자는 이 공개키(E)와 자신이 보유한 개인키(D)를 이용해 공유 비밀을 다시 계산할 수 있으며, 같은 방식으로 해시를 적용해 파생키를 생성함으로써 해당 파일을 복호화할 수 있다.

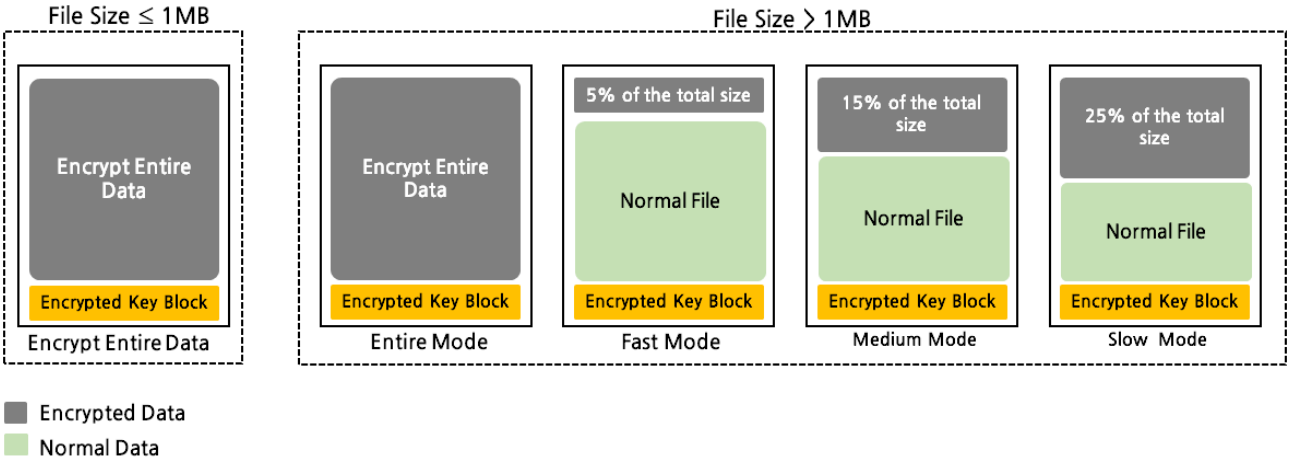


그림 12. 파일 암호화 방식

파일 암호화 범위는 파일 크기와 사용되는 인자에 따라 달라진다. 크기가 1MB 이하인 파일은 인자와 상관없이 전체 데이터가 암호화된다. 1MB 를 초과하는 파일의 경우에는 인자에 따라 파일 크기의 일부만 암호화한다. Entire 모드는 파일 전체가 암호화되고, Fast 모드는 파일의 5%, Medium 모드는 15%, Slow 모드는 25%를 암호화하는 방식이다. 암호화가 완료되면, 파일 크기나 암호화 모드와 관계없이 모든 파일의 끝에는 고정된 메타데이터 블록이 추가된다. 이 블록에는 복호화에 사용되는 공개키와 해당 파일에 적용된 암호화 모드 정보, 그리고 감염 여부를 식별하기 위한 “SINOBI” 마커가 함께 저장된다.

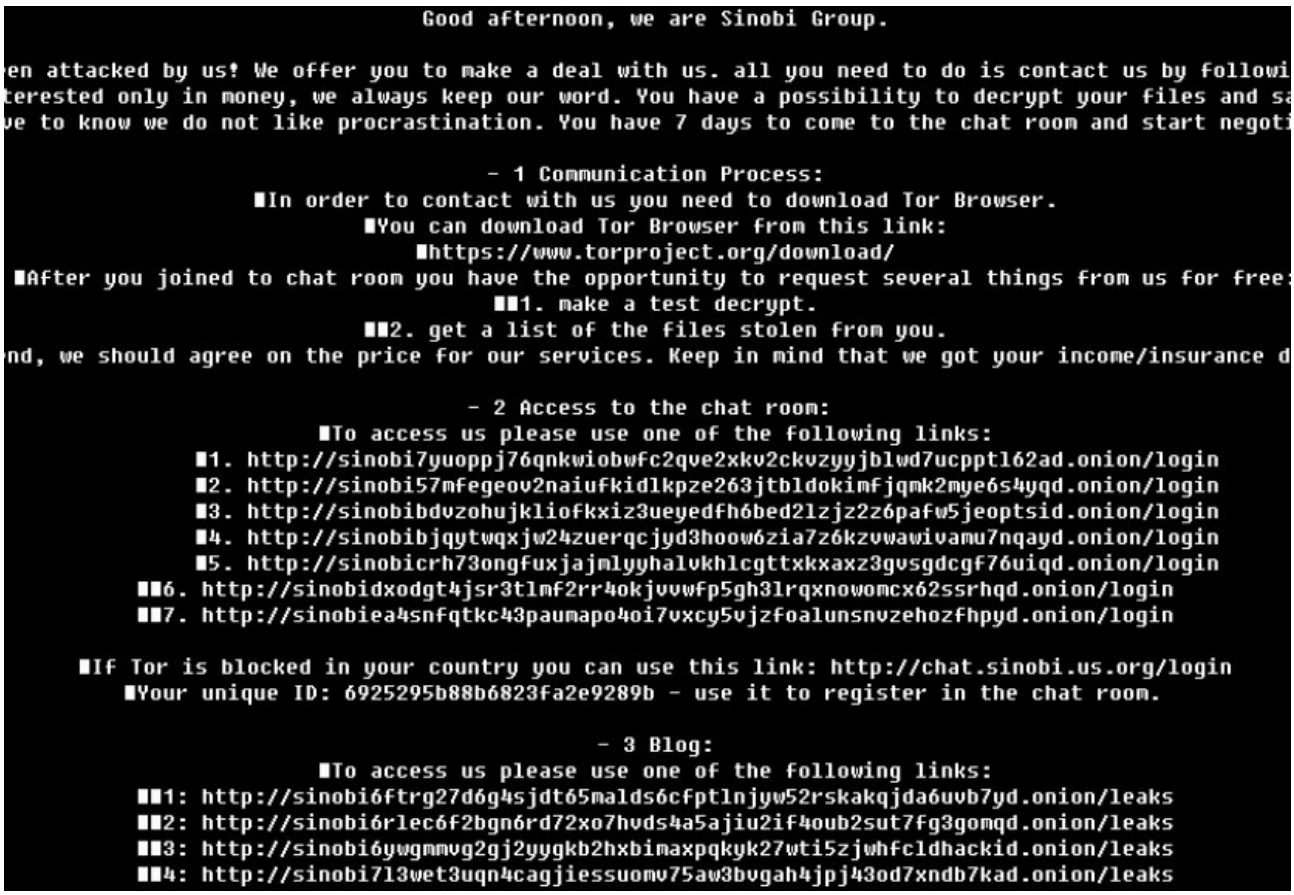


그림 13. 변경된 바탕화면

파일 암호화가 완료되면, 랜섬웨어는 실행 시점에 랜섬노트 텍스트를 포함한 바탕화면 이미지를 생성해 디스크에 저장한다. 이후 해당 이미지 경로를 HKCU\Control Panel\Desktop\Wallpaper 레지스트리 값에 설정하고 바탕화면을 변경함으로써, 피해자가 랜섬노트 내용을 즉시 확인하도록 유도한다.

랜섬웨어 대응방안

그림 14. 랜섬웨어 대응방안

Sinobi 랜섬웨어는 실행 시 복구를 방해하기 위해 VSS 와 백업과 관련된 프로세스 및 서비스를 종료한다. 이에 대비하기 위해 ASR[4] 규칙 활성화를 통해서 비정상적인 프로세스 동작을 차단하고, 랜섬웨어의 악성 행위를 막을 수 있다.

또한, EDR 솔루션 도입과 최신 보안 패치 적용으로 취약점을 악용한 침투나 비정상적인 행위를 신속히 식별하고 차단할 수 있도록 해야 한다. 백업 복사본은 별도의 네트워크 구간이나 외부 저장소, 오프라인 매체에 주기적으로 분산 백업해, 시스템이 암호화되더라도 데이터 복구가 가능하게 해야 한다. 이때 백업 장치 접근 권한을 최소화하고, 정기적으로 복구 테스트를 실시하여 백업 데이터의 무결성을 보장해야 한다.

마지막으로, Sinobi 랜섬웨어는 네트워크 공유 파일도 암호화하므로 네트워크 공유 자원의 접근권한을 최소화하거나 비활성화해 외부 리소스에 접근하지 못하도록 해야 한다.

[4] ASR (Attack Surface Reduction): 공격자가 사용하는 특정 프로세스와 실행 가능한 프로세스를 차단하는 보호 기능

IoCs

Hash(SHA-256)
9432B065C803BAA54F1FEFAC20D97AFFCE212DEC2BB9A597FC010064D391FC24
1B2A1E41A7F65B8D9008AA631F113CEF36577E912C13F223BA8834BBEFA4BD14
D4919A7402D7AE02516589FBDFB3CC436749544052843A37B5D36AC4B7385B18

■ 참고 사이트

• U.S. Department of the Justice(https://www.justice.gov/opa/pr/two-americans-plead-guilty-targeting-multiple-us-victims-using-alphv-blackcat-ransomware)

• Reuters(https://www.reuters.com/world/cyberattack-french-interior-ministrys-email-servers-compromised-more-than-20-2025-12-17/)

• Le Parisien(https://www.leparisien.fr/faits-divers/une-attaque-tres-grave-cinq-minutes-pour-comprendre-la-cyberattaque-qui-a-vise-le-ministere-de-linterieur-17-12-2025-ISX6EVWKDFCLLEZKHA2RBJFECA.php)

• S-RM(https://www.s-rminform.com/latest-thinking/react2shell-used-as-initial-access-vector-for-weaxor-ransomware-deployment)

JWT 서명키 유출이 초래하는 인증 위협과 리스크 대응 전략

■ 서론

JWT 는 인증에 필요한 정보를 하나의 토큰에 담아 전달하는 방식이다. 서버는 이 토큰을 통해 사용자를 확인하므로, 사용자 상태를 별도로 저장하지 않아도 인증을 처리할 수 있다. 이러한 특징 때문에 JWT 는 규모가 큰 서비스나 여러 서버로 구성된 환경에서 널리 사용된다.

JWT 기반 인증에서는 서버가 요청을 받을 때, 전달된 토큰이 정상적으로 생성된 것인지를 확인한다. 이때 서버가 확인하는 핵심 요소는 토큰의 서명이다.

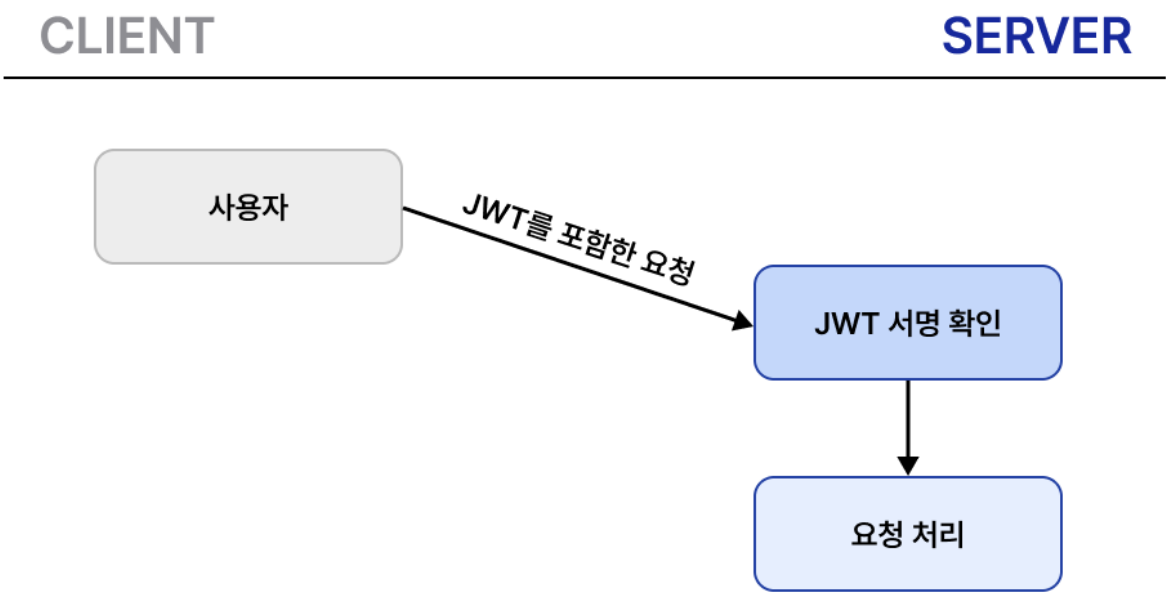


그림 1. JWT 기반 인증

이 구조에서는 서명 작성에 사용되는 서명키 하나에 인증의 신뢰가 집중된다. 따라서 서명키가 외부에 노출되거나 관리가 부실할 경우, 인증 체계 전체에 심각한 문제가 발생할 수 있다. 본 보고서는 JWT 의 이러한 특성을 바탕으로, 서명키 관리가 소홀할 때 어떤 문제가 발생하는지를 실제 사례를 통해 살펴보고자 한다.

■ JWT

1) JWT 란?

JWT 는 서버가 요청을 받을 때마다 “이 요청이 인증된 사용자로부터 온 정상 요청인지”, 그리고 해당 사용자가 “이 기능을 수행할 권한이 있는지”를 판단하기 위해 사용하는 토큰이다.

JWT 는 주로 로그인 성공 시 발급되어 이후 요청에 함께 전달되며, 서버는 이를 통해 사용자 식별과 권한 확인을 수행한다. 따라서 JWT 는 단순 로그인 상태 유지뿐 아니라 API 접근 제어, 서비스 간 인증(SSO)처럼 요청의 정당성을 증명해야 하는 환경에서 폭넓게 활용된다.

기존에는 서버가 “누가 어떤 상태인지”를 서버 내부에 저장해 두는 세션 방식을 통해, 요청이 올 때마다 저장된 정보를 조회하는 방식이 일반적이었다. 반면 JWT 방식은 서버가 상태를 계속 들고 있기보다는, 필요한 증명 정보를 토큰으로 묶어 사용자가 들고 다니게 한다. 이처럼 JWT 는 서버가 로그인 상태를 서버 메모리/DB 에 저장하지 않는 Stateless 구조를 전제로 하며, 요청이 올 때마다 토큰의 서명 검증 결과로 사용자를 확인한다.

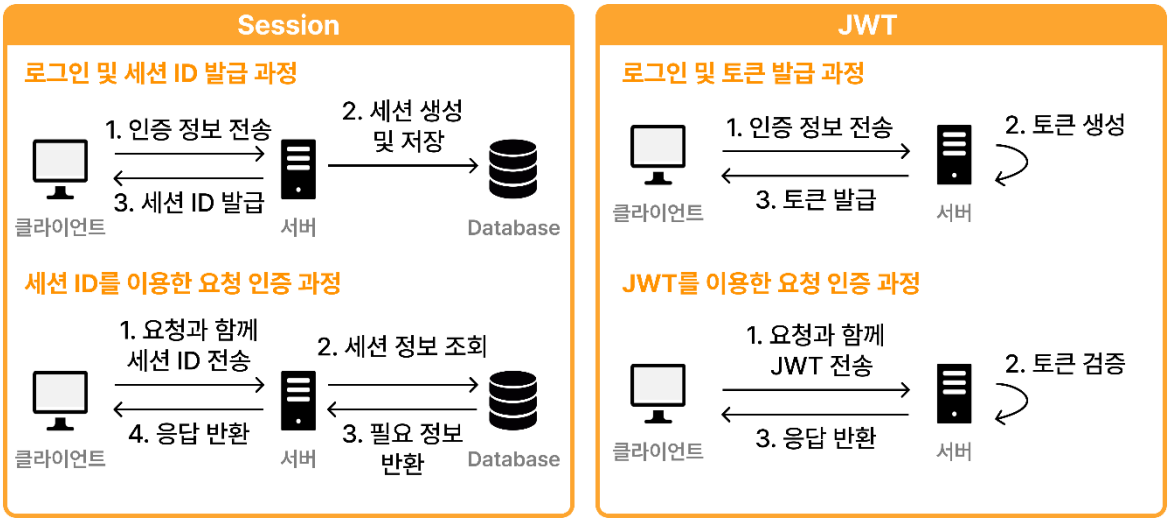


그림 2. 세션 및 JWT 발급 및 인증

2) JWT 구성 요소

JWT 는 점(.)으로 구분되는 세 개의 구성 요소로 이루어지며, 일반적으로 Header.Payload.Signature 형태로 표현된다. 각 구성 요소는 사람이 직접 읽기 위한 형식이 아닌, 데이터 전송 간 효율성을 위해 Base64Url 방식으로 인코딩된 문자열이다.

JWT 예시
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkVRU1QiLCJhZG1pbiI6dHJ1ZSwiaWF0IjoxNTE2MjM5MDIyLCJodHRwczovL3d3dy5za3NoaWVsZHVzLmNvbS8iOnRydWUsInVzZXJuYW1lIjoiRVFTVCBMYWIifQ.PYq5fttdbnGoTpSsLiSI0rSBB49hsc1mAP85jG1wiGA

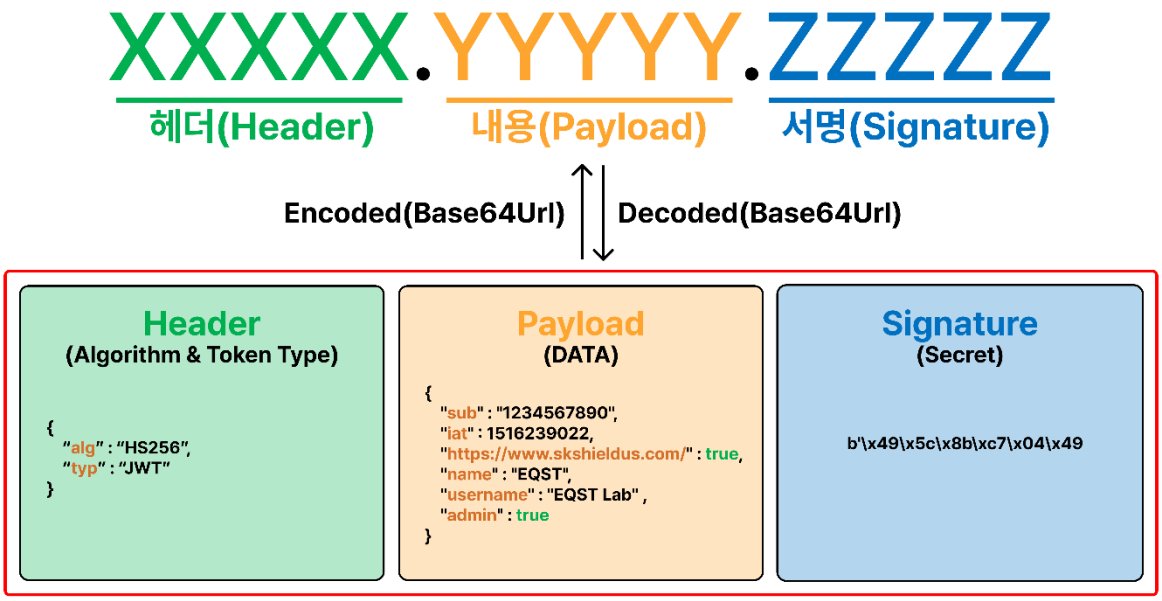


그림 3. JWT 구조

① Header(헤더)

헤더는 서명 알고리즘과 토큰 타입 정보를 포함하고 있다. 서버는 토큰을 검증할 때, 미리 허용해 둔 알고리즘 목록과 헤더에 명시된 알고리즘을 비교한 후, 일치하는 경우에만 해당 알고리즘을 사용해 시그니처(Signature)를 검증한다.

```
{
  "alg": "HS256",  // 서명 알고리즘
  "typ": "JWT"     // 토큰 타입
}
```

그림 4. JWT 헤더

② Payload(페이로드)

페이로드에는 토큰에 담을 실제 정보들이 포함되며, 이 정보의 각 항목을 클레임(Claim)이라 한다. JWT 표준(RFC 7519)에서는 클레임의 특성과 사용 범위에 따라 등록된 클레임, 공개 클레임, 비공개 클레임으로 구분된다.

```
{
  "sub": "1234567890",
  "iat": 1516239022,
  "https://www.skshieldus.com/": true,
  "name": "EQST",
  "team": "EQST Lab",
  "admin": true
}
```

그림 5. 디코딩 된 전체 페이로드

- **등록된 클레임(Registered Claim)**

등록된 클레임은 JWT 표준에서 의미와 용도가 사전에 정의된 클레임으로, 토큰 발급자, 유효 기간, 대상 등을 표현하기 위해 사용된다. 필수 항목은 아니지만, 토큰의 유효성 판단과 서비스 이용에 활용되기 때문에 실제 서비스 환경에서 사용되는 경우가 많다.

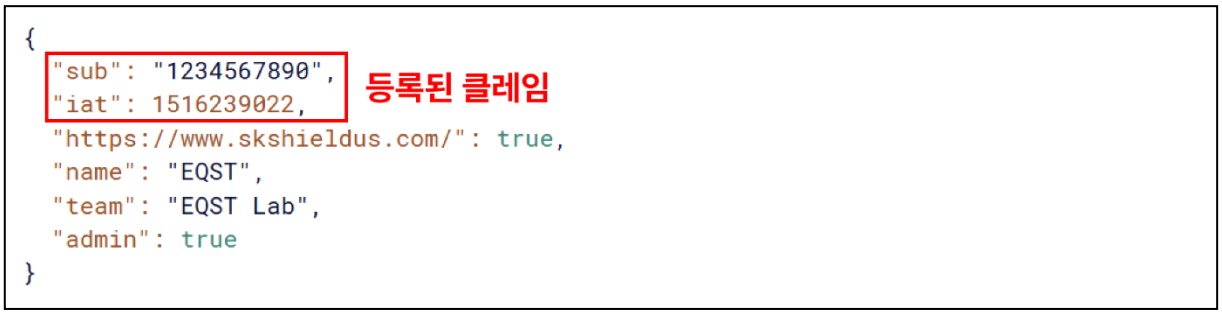


그림 6. 페이로드 내 등록된 클레임

JWT 표준에서 정의한 등록된 클레임 종류는 다음과 같다.

약어	클레임명(Full Name)	설명
iss	Issuer	토큰 발급자
sub	Subject	사용자 고유 식별자
aud	Audience	토큰 수신자
exp	Expiration Time	토큰 만료 시간(Unix Time)
nbf	Not Before	토큰 활성 시작 시간
iat	Issued At	토큰 발급 시간
jti	JWT ID	토큰 고유 식별자

표 1. 등록된 클레임 종류

• **공개 클레임(Public Claim)**

공개 클레임은 여러 서비스가 같은 JWT 를 함께 사용할 때, 서로 다른 서비스가 같은 이름의 클레임을 써서 충돌하는 일을 막기 위해 사용된다. 이를 위해 클레임 이름을 "https://www.skshieldus.com/"처럼 URI 형태로 길게 만들어 어느 조직 또는 서비스의 정의인지를 구분한다. 즉, 클레임 이름을 URI 형식으로 정의한 경우 해당 클레임은 공개 클레임으로 분류된다.

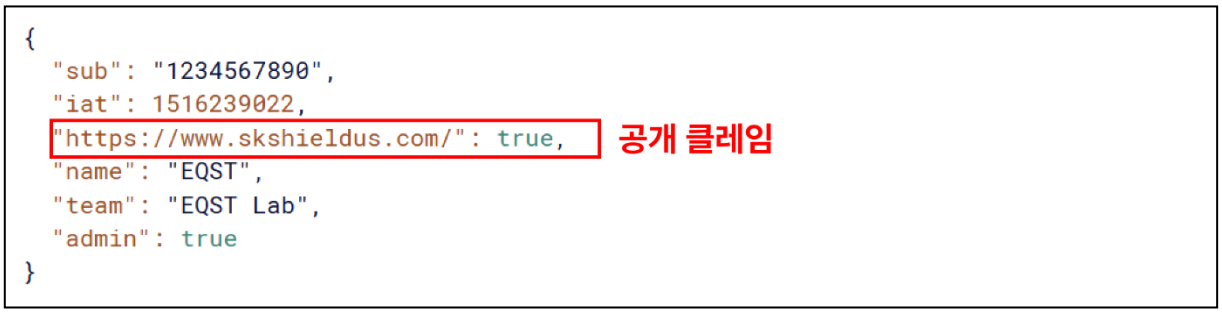


그림 7. 페이로드 내 공개 클레임

• **비공개 클레임(Private Claim)**

비공개 클레임은 JWT 표준에 의해 의미가 정의되어 있지 않아, 클레임 이름과 값 모두 서비스 요구사항에 따라 자유롭게 설계할 수 있다. 이로 인해 서버의 인증 및 인가 로직에 필요한 정보를 서비스 특성에 맞게 직접 포함하는 용도로 주로 활용된다.

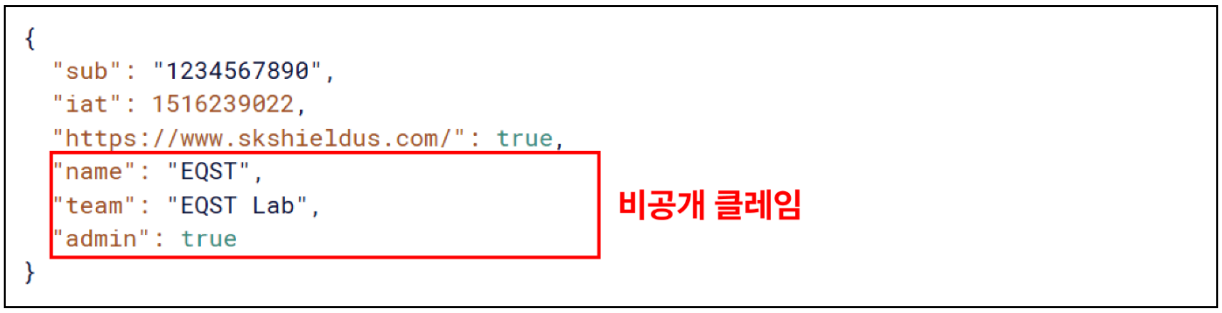


그림 8. 페이로드 내 비공개 클레임

③ Signature(시그니처)

시그니처는 JWT 가 전송 과정에서 위·변조되지 않았음을 확인하고, 서버가 신뢰하는 발급 주체가 생성한 토큰인지 판단하기 위한 서명 값이다. JWT 의 헤더와 페이로드는 누구나 쉽게 디코딩 해 내용을 확인할 수 있으므로, 토큰의 신뢰 여부는 시그니처 검증 결과에 의해 결정된다.

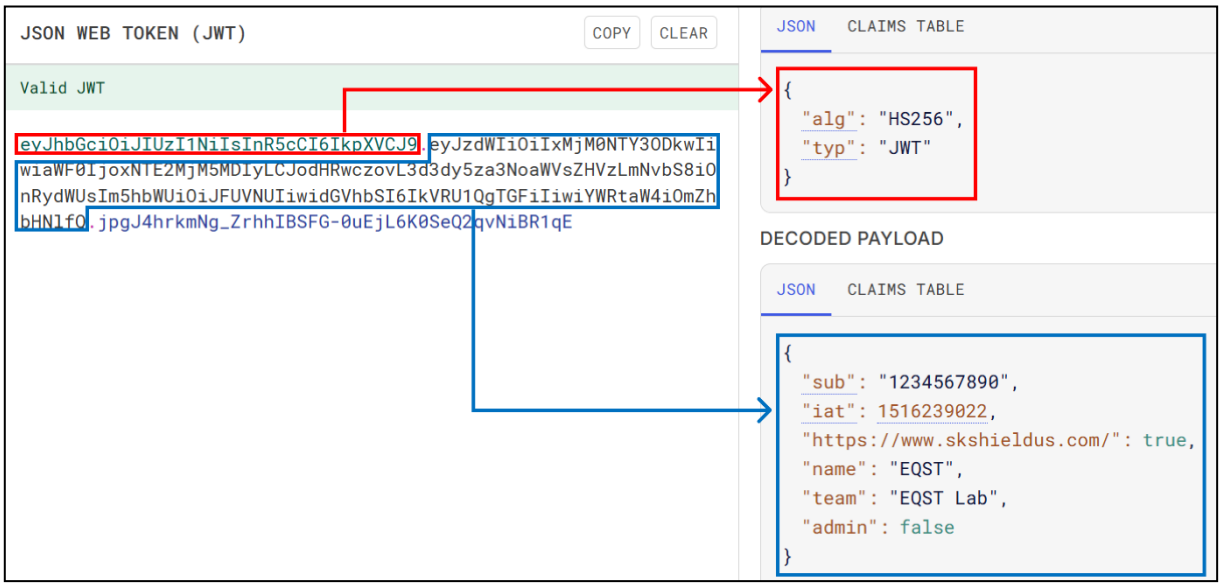


그림 9. 디코딩 된 JWT

시그니처는 JWT 의 헤더와 페이로드를 각각 Base64Url 인코딩한 뒤, 두 값을 점(.)으로 연결한 문자열을 해시 함수의 입력값 중 하나로 사용해 생성된다. 따라서 헤더나 페이로드가 조금이라도 변경되면 입력값이 달라지고, 서버의 검증 과정에서 시그니처가 일치하지 않아 토큰이 거부된다.

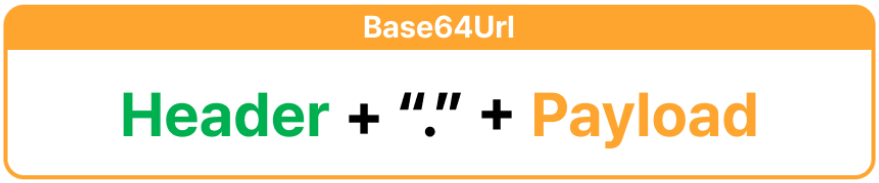


그림 10. 시그니처 입력값

이때 서버가 입력값이 바뀌었는지를 판단할 때 핵심이 되는 성질이 해시 함수의 특성이다. 해시 함수는 임의 길이의 입력 데이터를 고정 길이의 값으로 변환하며, 입력값이 조금만 바뀌어도 결과값이 완전히 달라진다. 그래서 토큰 내용이 변경되면 입력값이 달라지고, 결과적으로 시그니처 검증도 실패하게 된다.

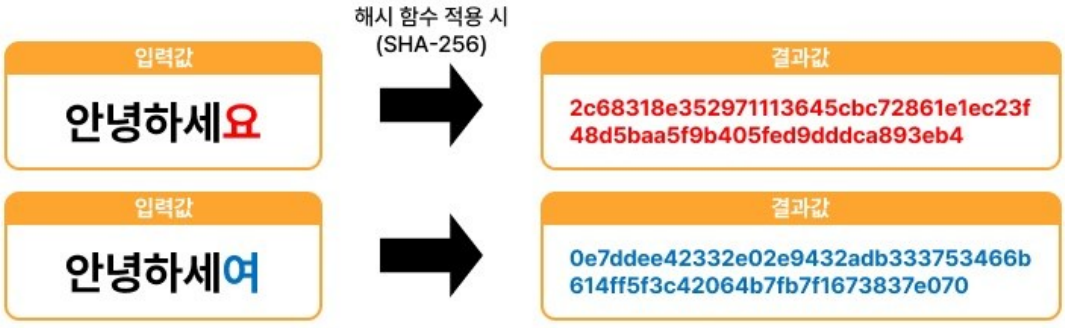


그림 11. 해시 예시

하지만 해시만으로는 인증을 성립시킬 수 없다. 해시는 입력값만 알면 누구나 같은 결과를 계산할 수 있으므로, 공격자도 페이로드를 바꾼 뒤 그에 맞는 해시 값을 다시 계산해 붙일 수 있기 때문이다. 따라서 JWT 는 단순 해시가 아니라, 키를 사용하는 서명 알고리즘으로 시그니처를 생성한다. 즉 "서버가 가진 키로만 만들 수 있는 값"을 시그니처로 사용함으로써, 토큰이 변조되지 않았을 뿐 아니라 서버가 신뢰하는 키로 서명된 토큰인지까지 확인한다.

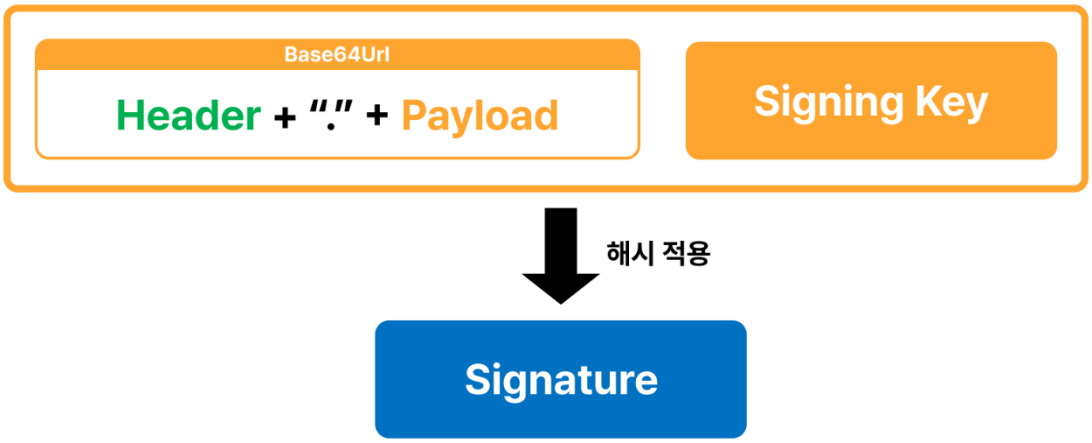


그림 12. JWT 시그니처 생성 과정

그렇기에 JWT 의 신뢰는 시그니처 생성 시 사용되는 키가 안전하게 보호된다는 전제에서만 유지된다. 서명키가 유출되는 등의 이유로 신뢰를 잃게 될 경우, JWT 인증 체계가 흔들릴 수 있으며, 이를 이어지는 내용에서 실제 사례와 시나리오를 통해 설명한다.

■ 서명키 유출로 인한 피해 사례

앞서 살펴본 것처럼 JWT 는 시그니처 검증 결과로 신뢰 여부를 판단한다. 이 때문에 시그니처를 생성하는 서명키의 보호와 관리가 보안의 핵심이 된다. 아래 사례들은 JWT 인증 기반 구조에서 서명키 관리에 실패했을 때 어떤 문제가 발생하는지 보여준다.

1) 쿠팡 사용자 개인정보 유출(2025) – API 인증 우회로 인한 대량 개인정보 유출

2025 년 쿠팡에서는 내부 인증 체계에 사용되던 JWT 서명키 관리 문제와 연관된 대규모 개인정보 유출 사고가 발생했다. 유출 규모는 약 3,370 만 건으로 알려졌다. 언론 보도에 따르면, 퇴사자가 재직 중 취득한 서명키가 악용되었고, 퇴사 이후 서명키 폐기 및 갱신 등의 관리가 미흡했던 점이 지적됐다.

공격자는 JWT 의 헤더와 페이로드를 임의로 구성한 뒤, 취득한 서명키로 서명을 생성함으로써 서버 측 검증을 통과하는 토큰을 만들어낼 수 있었고, 개인정보 조회 API의 인증을 우회하여 다수 계정의 개인정보에 접근할 수 있었다.

해당 접근은 단기간의 일회성 시도가 아닌 일정 기간 지속적으로 발생한 것으로 알려졌으며, 공격이 종료되기 전까지 위조된 JWT 를 이용한 비정상적인 API 호출이 즉각적으로 차단되지 않았다. 이 사건은 JWT 인증 구조에서 서명키가 유출될 경우, 서버가 해당 요청을 정상 사용자 요청과 구분하기 어렵다는 점을 실제 사고를 통해 보여준 사례이다.

2) Microsoft Storm-0558(2023) – 위조 토큰을 통한 메일함 열람

2023 년 Microsoft 는 “Storm-0558”이라는 중국계 위협 그룹이 Outlook 및 Exchange Online 서비스에 접근하기 위해 위조된 인증 토큰을 사용한 사실을 공식 발표했다. Microsoft 의 후속 보고에 따르면, 공격자는 Microsoft 개인 계정(MSA)에서 로그인 이후 발급되는 인증 토큰에 사용되던 서명키를 획득했고, 이를 이용해 정상 사용자로 로그인한 것처럼 보이는 인증 토큰을 직접 생성했다.

서비스는 해당 토큰을 정상적인 인증 요청으로 인식했기 때문에, 공격자는 특정 사용자의 권한으로 메일 조회 API 에 접근할 수 있었다. 이 사건은 사용자 인증 정보를 서버 세션이 아닌 서명된 토큰에 담아 처리하던 구조에서, 그 토큰을 생성하는 키가 유출되며 인증 체계 전체가 무력화된 사례이다.

이는 JWT 를 대표로 하는 stateless 토큰 인증 환경에서 서명키 하나가 인증 신뢰의 중심에 있으며, 해당 키가 노출될 경우 인증 전반이 붕괴될 수 있음을 보여준다.

3) Solorigate/Golden SAML[1] (2020) – SAML 토큰 위조를 통한 SSO 인증 우회

2020 년 공개된 SolarWinds 공급망 침해(Solorigate) 대응 과정에서, Microsoft 는 공격자가 일부 피해 조직의 ADFS[2] 를 장악한 뒤 ADFS 의 토큰 서명 인증서(개인키)를 탈취해 SAML 로그인 토큰을 위조하고 정상 인증처럼 통과시키는 행위를 확인했다고 안내했다. 공격자는 이 인증서를 이용해 임의 사용자로 발급된 것처럼 보이는 SAML 토큰을 생성하고 서명할 수 있으며, 서비스는 서명이 유효한 것으로 인식해 이를 정상 로그인 결과로 받아들이게 된다.

이 사례는 JWT 가 아니라 SAML 형식 토큰에서 발생한 문제이다. 그러나 서명된 토큰을 신뢰하는 구조에서는, 토큰 형식을 불문하고 서명을 생성하고 검증하는 데 사용되는 키(서명키/서명 인증서) 관리가 중요하다는 것을 보여준다.

■ 유출된 서명키를 통한 JWT 위조

앞선 사례들은 서명키가 유출되면 공격자가 토큰을 직접 만들어 인증을 우회할 수 있음을 보여준다. 서버가 서명 검증을 신뢰의 핵심 기준으로 삼는 구조에서는, 외부에서 생성된 토큰이라도 동일한 키로 서명되면 정상 토큰과 구분하기 어렵다. 이 때문에 앞선 사례에서도 침해 사실이 즉시 인지되지 않았으며, 사후 조사 및 분석을 통해 확인되었다. 본 챕터에서는 이를 JWT 검증 기준과 위조 과정의 관점에서 설명한다.

서버는 일반적으로 서명 검증을 1 차 신뢰 기준으로 삼아 토큰의 위·변조 여부를 판단한다.

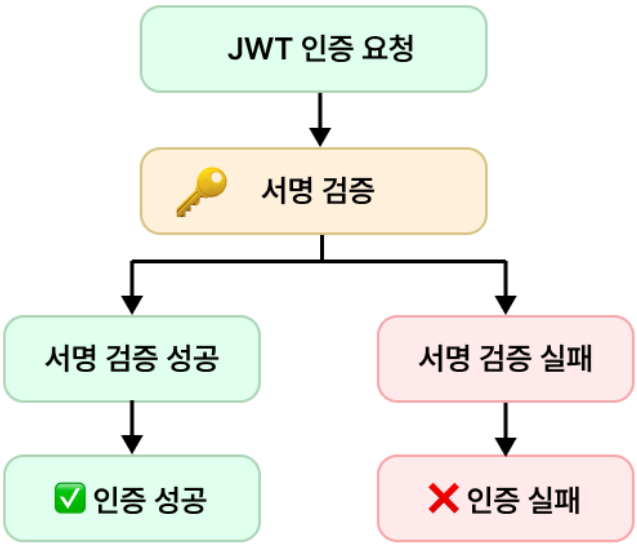


그림 13. JWT 검증 기준

[1] SAML (Security Assertion Markup Language): SSO 환경에서 인증 결과를 XML 기반 토큰으로 전달하는 표준

[2] ADFS (Active Directory Federation Service): 회사 계정으로 한번 로그인하면, 그 사실을 증명하는 토큰을 발급해 여러 서비스에 자동 로그인되게 해주는 윈도우 서버 기능

서명키가 유출된 상황에서는 공격자도 동일한 방식으로 서명값을 생성할 수 있게 된다. 이로 인해 JWT 기반 인증 환경에서는, 서명키 보유자가 곧 토큰 발급자와 동일한 권한을 갖게 된다. 정상적인 인증 서버에서 발급된 토큰과, 외부에서 동일한 서명키로 생성된 토큰을 구분할 수 없기 때문이다.

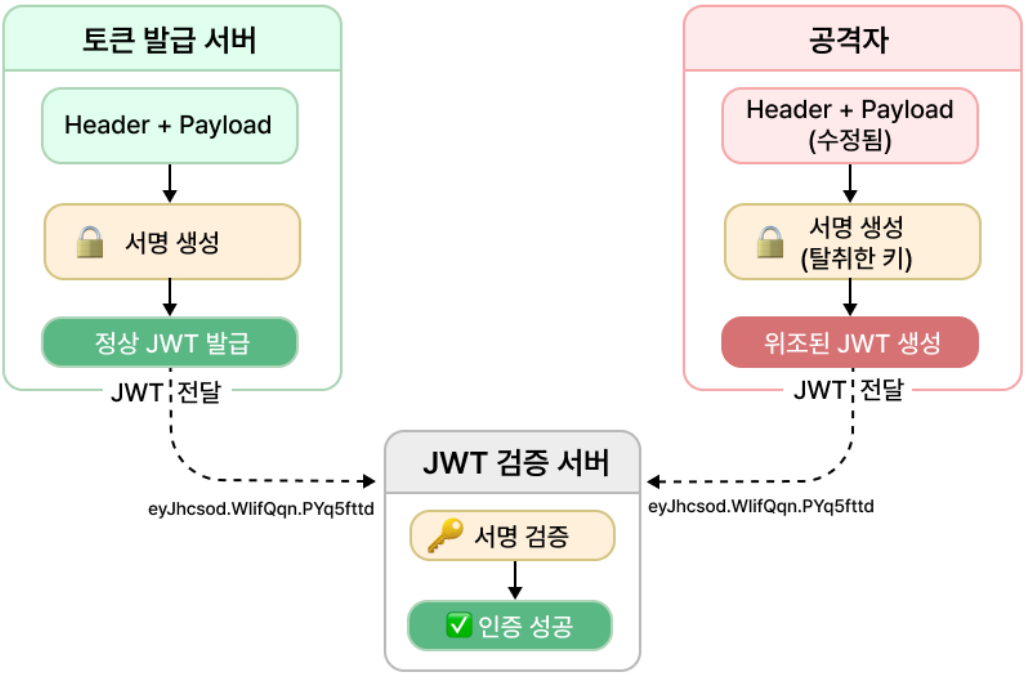


그림 14. 서명키를 이용한 JWT 위조(생성)과정

특히 JWT 의 페이로드는 암호화되지 않은 상태로 전달되므로, 토큰 구조를 파악하거나 내부 클레임 값을 확인할 수 있다.

```
{
  "sub": "1234567890",
  "iat": 1516239022,
  "https://www.skshieldus.com/": true,
  "name": "EQST",
  "team": "EQST Lab",
  "admin": false
}
```

그림 15. JWT 페이로드

공격자가 서명키를 확보한 경우, 기존 토큰의 페이로드를 참고하여 사용자 식별자 또는 권한 정보를 변경한 뒤, 동일한 키로 서명된 새로운 JWT 생성이 가능하다.

그림 16. 페이로드가 수정된 JWT 생성

이처럼 JWT 기반 인증은 서명 검증을 신뢰 기준으로 삼기 때문에, 서명키가 외부에 노출되면 동일 서명을 통한 토큰 위조 시도가 가능한 환경이 된다.

■ 공격 시나리오

JWT 가 공격자에 의해 위조될 경우, 다양한 문제들이 발생할 수 있다. 본 챕터에서는 그 예시를 설명한다.

1) API 인증 우회

쇼핑몰 서비스에서 개인정보 조회, 주문 내역 확인처럼 사용자별 데이터를 반환하는 API 는 요청에 포함된 JWT 로 요청자를 식별하는 경우가 많다. 실제로 아래 요청 헤더에서 확인할 수 있듯, 개인정보 조회 및 수정 API 호출 시 쿠키에 JWT 형태의 토큰이 포함되어 전달된다.

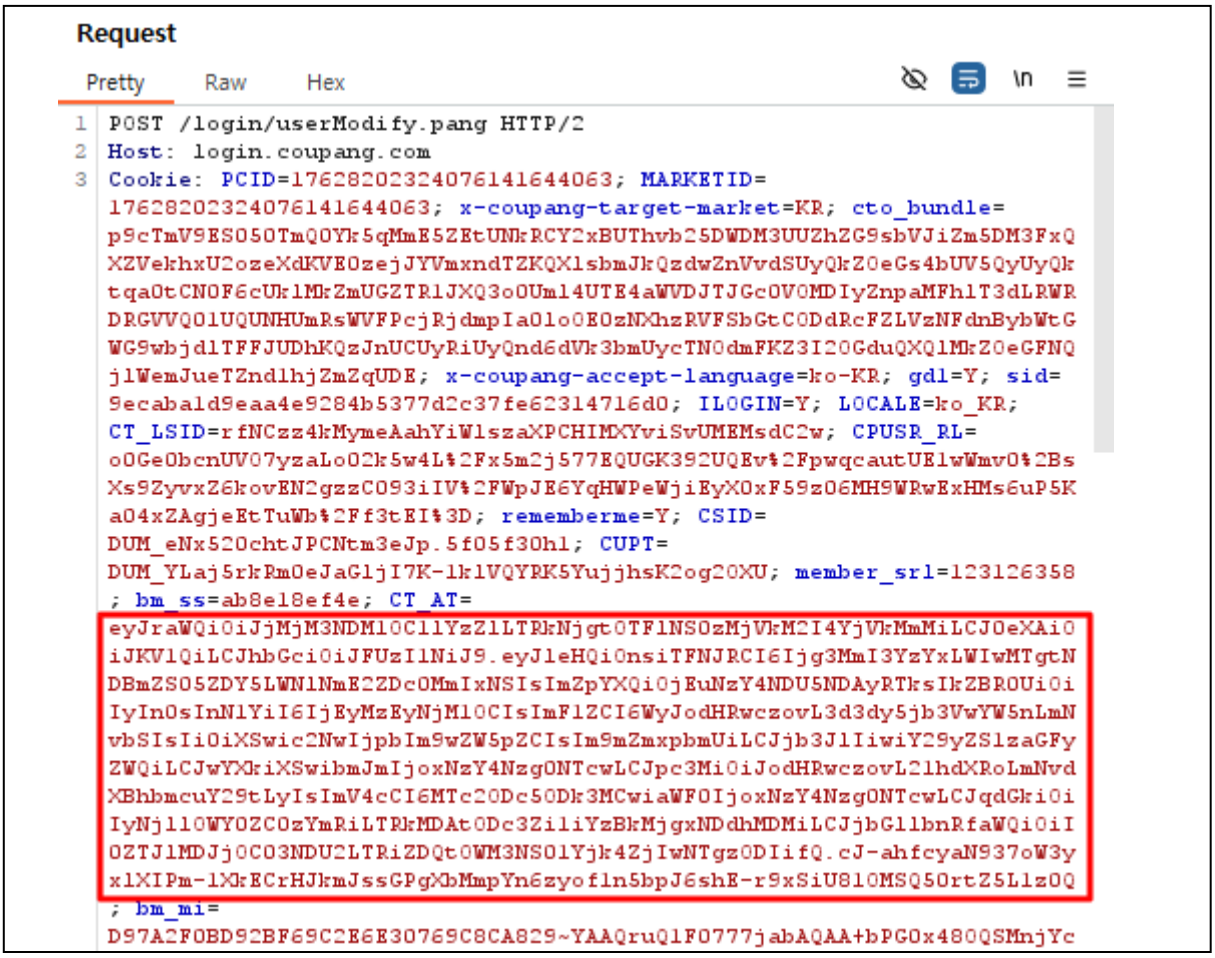


그림 17. 개인정보확인 API 요청에 존재하는 JWT

해당 JWT 를 디코딩하면 sub 와 같은 사용자 식별 정보가 포함되어 있음을 확인할 수 있다. 일부 구현에서는 서버가 요청 처리 과정에서 이 sub 값을 조회 대상 사용자 식별에 직접 사용한다.

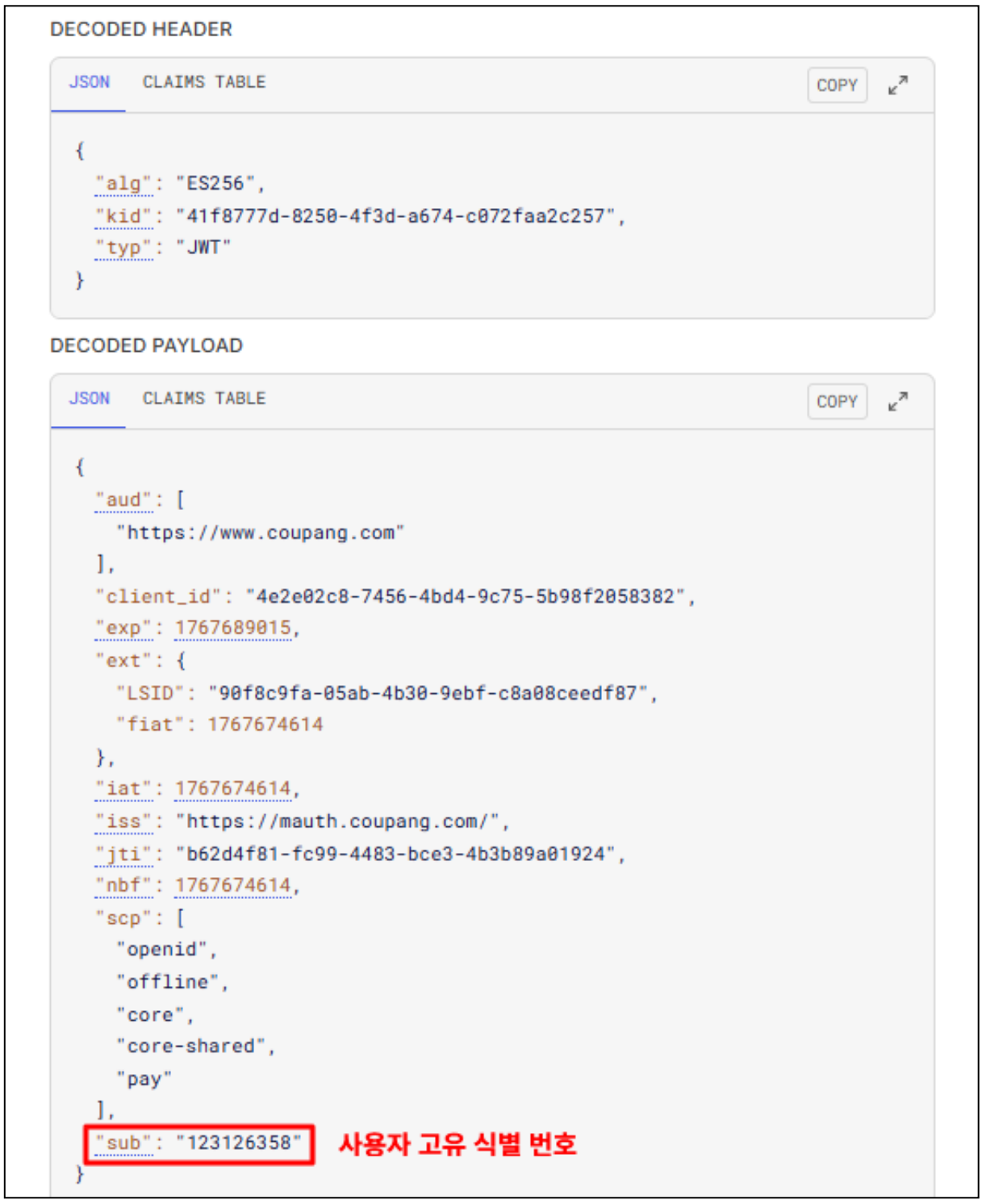
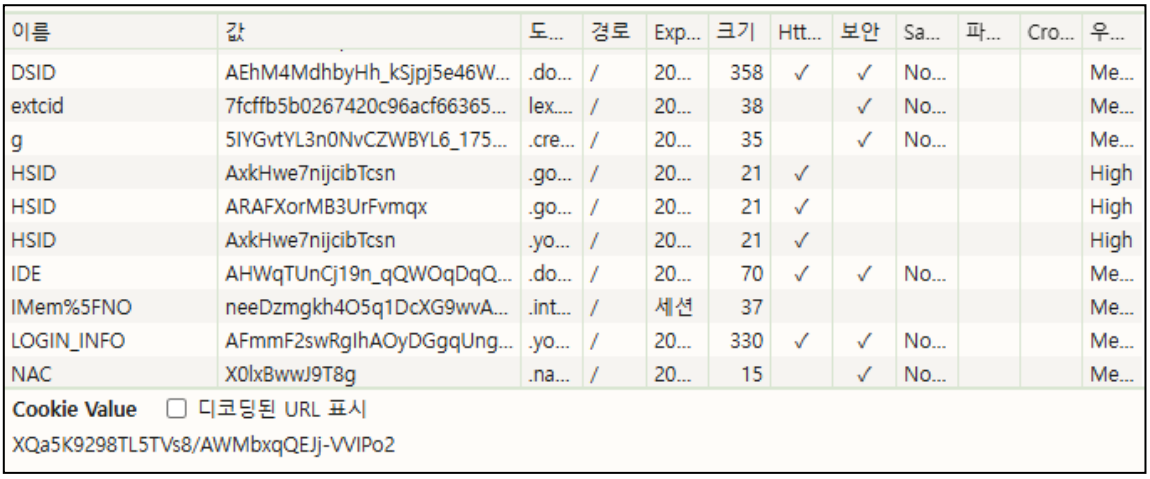


그림 18. JWT 디코딩 결과

따라서 서명키가 유출된 상황에서는 공격자가 sub 값을 다른 사용자 식별자로 변경한 뒤, 유효한 서명을 가진 JWT 를 새로 생성하여 동일 API 를 호출할 수 있다. 이러한 방식은 자동화와 결합되면 연속적인 사용자 식별자 대입을 통해 대량 조회로 확대될 수 있어, 개인정보 유출 규모가 급격히 커질 수 있다.

2) 로그인 우회

일부 서비스는 로그인 성공 후 JWT 를 쿠키에 저장하고, 이후 요청에서는 해당 토큰의 유효성만으로 로그인 상태를 판단한다. 로그인 전에는 아래와 같이 JWT 가 존재하지 않는다.



이름	값	도...	경로	Exp...	크기	Htt...	보안	Sa...	파...	Cro...	우...
DSID	AEhM4MdhbyHh_kSjpj5e46W...	.do...	/	20...	358	✓	✓	No...			Me...
extcid	7fcffb5b0267420c96acf66365...	lex....	/	20...	38		✓	No...			Me...
g	5IYGvtYL3n0NvCZWBYL6_175...	.cre...	/	20...	35		✓	No...			Me...
HSID	AxkHwe7nijcibTcsn	.go...	/	20...	21	✓					High
HSID	ARAFXorMB3UrFvmqx	.go...	/	20...	21	✓					High
HSID	AxkHwe7nijcibTcsn	.yo...	/	20...	21	✓					High
IDE	AHWqTUnCj19n_qQWOqDqQ...	.do...	/	20...	70	✓	✓	No...			Me...
IMem%5FNO	neeDzmgkh4O5q1DcXG9wvA...	.int...	/	세션	37						Me...
LOGIN_INFO	AFmmF2swRgIhAOyDGgqUng...	.yo...	/	20...	330	✓	✓	No...			Me...
NAC	X0lxBwwJ9T8g	.na...	/	20...	15		✓	No...			Me...

Cookie Value ☐ 디코딩된 URL 표시

XQa5K9298TL5TVs8/AWMbxqQEJj-VVIPo2

그림 19. 로그인 전 쿠키

로그인 후에는 id_token 이라는 JWT 가 새로 생성되어 쿠키에 저장되는 것을 확인할 수 있다.

이름	값	도...	경로	Exp...	크기	Htt...	보안	Sa...	파...	Cro...	우...
DSID	AEhM4MdhbyHh_kSjpj5e46W...	.do...	/	20...	358	✓	✓	No...			Me...
extcid	7fcffb5b0267420c96acf66365...	lex....	/	20...	38		✓	No...			Me...
g	5IYGvtYL3n0NvCZWBYL6_175...	.cre...	/	20...	35		✓	No...			Me...
HSID	AxkHwe7nijcibTcsn	.go...	/	20...	21	✓					High
HSID	ARAFXorMB3UrFvmqx	.go...	/	20...	21	✓					High
HSID	AxkHwe7nijcibTcsn	.yo...	/	20...	21	✓					High
id_token	eyJraWQiOiJfTmVvUDZlX1Nl...	.int...	/	세션	1032	✓	✓	Lax			Me...
IDE	AHWqTUnCj19n_qQWOqDqQ...	.do...	/	20...	70	✓	✓	No...			Me...
IMem%5FNO	neeDzmgkh4O5q1DcXG9wvA...	.int...	/	세션	37						Me...
interparkSNO	Vxv7t0P%2F85qVE2iXyuxx9p0...	.int...	/	세션	60		✓	Lax			Me...

Cookie Value ☐ 디코딩된 URL 표시

eyJraWQiOiJfTmVvUDZlX1NlWkxkVnpQZk9NbUVRb09CUE9uR3ZNZmdQUUNZc0l6ZFJBIiwiYWxnIjoiUlMyNTYifQ.eyJhY2Nlc3Nf
dG9rZW4iOiJXeGtnd3g4akxtZ3hrWGE5WmlmWVVkS1BCZzJcL0F1aXVmRjR3d1VMbVBkVVdhRlhyV1hBTURGMWJqN0N4SlN3R
yIsInN1YiI6IjdYXC9RNlhRXC9jZExVYVdtZURlek9uZz09IiwiYXVkIjoidGlja2V0LXBjIiwicmVmcmVzaF90b2tlbiI6IkpsbDZJbmtlUzNLa
XdkTEh3WTl4SjdxcE1EOUkzOG1BbjhtenJmTHFaWng3RVh5SzdHYlBPYlVZRGQycDdsdUEiLCJpc3MiOiJodHRwczpcL1wvYWNjb3
VudHMuaW50ZXJwYXJrLmNvbSIsInJlbWVtYmVyX21lIjpmYWxzZSwiZXhwIjoxNzY4NTMyMjUxLCJpc19ub2xfY29ubmVjdGVkIjp0
nVlLCJpYXQiOjE3Njg1MjY4NTEsInZlcnNpb24iOiIyLjAiLCJtbm8iOiJuZWVEem1na2g0TzVxMURjWEc5d3ZBPT0iLCJzaWQiOiJWeH
Y3dDBQXC84NXFWRTJpWHl1eHg5cDBwM1NnRFVjb3JuOVk1ZDBicVljWT0ifQ.o9mos9k12ZFyrnEkm7Ls3xug3p3VG3-dm9B4DY
47J1rLNkJODAUxC1B8PileajHQD5WUlMDQNFUmCSr0xM5vBKnP5ZWa3FDZSA_o0e31J4yEGP5QSYMK7_zc_LE2ez25UX3QDdoc
dlMpMxKDfq2y3KGfYhJj39-Id-jqlf-_40PgefhIfeZCbAbCZyfHPPMyTZ-EmclCS2MjWUyf6thOztla7t-e3YBNaLZooP7etlS50_0yD-LC
RmRh38mb6ADd8Fz7mGrB768GXvQ32LnBc9Xk7rAF4gDyNhUZXTFokcxGnd9yPe69AFAJMKAh8vYDyfqjCgCiqAu-yPG5AiThA

그림 20. 로그인 후 JWT 토큰이 생성된 모습

이 id_token 를 디코딩해보면 sub/iat/exp 같은 기본 클레임 외에도 access_token, refresh_token, sid 처럼 로그인 상태를 구성하고, 유지하기 위한 값이 함께 포함된 것을 확인할 수 있다. 즉 이 토큰은 서비스가 로그인 상태를 판단하고 유지하는 데 사용하는 세션 대체 수단으로 동작할 수 있다.

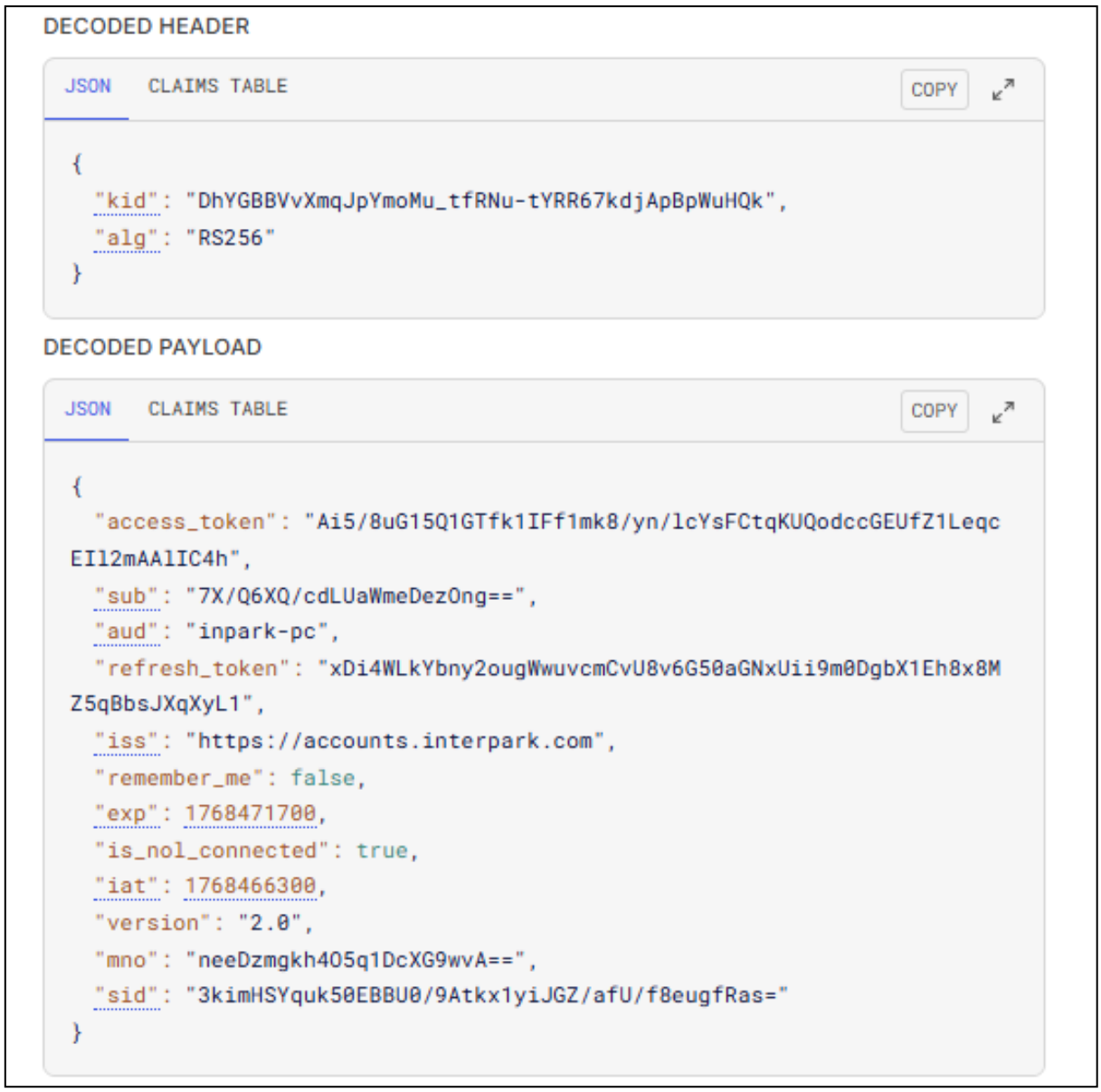


그림 21. JWT 디코딩 결과

서명키가 유출되면 공격자는 로그인 절차 없이도 유효한 id_token 을 생성해 주입할 수 있으며, 서비스는 이를 정상 로그인 결과로 인식한다. 즉, 로그인을 하지 않았음에도 로그인 한 것처럼 행동하는 것이 가능해진다.

3) 연계 로그인 우회

SSO(연계 로그인) 환경에서는 사용자가 서비스에 직접 ID/PW 입력하는 대신, 외부 인증 시스템이 로그인 결과를 토큰 형태로 발급하고 서비스는 이를 검증해 로그인 처리한다. 이때 로그인 결과 토큰이 JWT 형태로 전달되는 경우가 많으며, 서비스는 토큰의 서명 유효성 및 기본 클레임을 확인해 "정상 발급된 로그인 토큰"인지 판단한다.

만약 공격자가 SSO 토큰에 사용되는 서명키를 확보하거나, 토큰 검증 로직의 허점을 이용해 서명 검증을 우회할 수 있다면, 정상 인증 과정을 거치지 않고도 특정 사용자로 로그인한 것처럼 보이는 토큰을 만들어 서비스에 제출할 수 있다. 서비스는 서명이 유효한 토큰을 정상 로그인 결과로 받아들이기 때문에, 결과적으로 공격자는 SSO 로 연결된 계정의 권한으로 서비스에 접근하는 것이 가능해진다.

특히 SSO 는 여러 서비스가 동일한 인증 체계를 공유하는 구조인 경우가 많아, 하나의 서명키가 침해되면 단일 서비스에 그치지 않고 연동된 다른 서비스까지 영향이 확대된다. 이처럼 연계 로그인 환경에서의 JWT 위조는 "개별 서비스의 인증 우회"를 넘어 "인증 체계 전반의 신뢰 붕괴"로 이어질 수 있다.

4) 권한 상승

JWT 기반 인증 환경에서는 구현 방식에 따라 사용자 권한 정보가 페이로드의 클레임으로 포함될 수 있다. 예를 들어 일반 사용자와 관리자 계정을 구분하기 위해 role, isAdmin, authLevel 등의 클레임이 사용될 수 있으며, 서비스는 이후 요청에서 해당 값을 기준으로 접근 가능한 기능을 판단한다.

이처럼 권한 정보가 JWT 에 포함되는 구조에서는, 서명키가 유출될 경우 공격자가 기존 토큰의 페이로드를 수정해 권한 관련 클레임 값을 임의로 변경한 뒤, 유효한 서명으로 다시 JWT 를 생성할 수 있다. 예를 들어 role 값을 "ADMIN"으로 설정하거나 권한 레벨 값을 상향 조정한 토큰을 만들어 관리자 전용 API 를 호출하면, 관리자 페이지 접근, 사용자 계정 관리, 설정 변경, 로그 조회 등 권한 기능이 악용될 수 있다.

이러한 권한 상승은 단일 계정 침해를 넘어 서비스 운영 전반에 영향을 미칠 수 있다는 점에서 위험도가 크다. 특히 관리자 권한 범위가 넓을수록 데이터 변조, 대량 정보 유출, 서비스 장애 유발 등 2 차 피해로 확대될 수 있다.

■ 대응방안

이처럼 JWT 기반 인증 환경에서는 서명키 하나에 인증의 신뢰가 집중되며, 해당 키가 유출될 경우 토큰 형식이나 알고리즘과 무관하게 인증 체계 전반이 영향을 받는다. 이처럼 서명키에 대한 의존도가 높은 구조에서는, 서명키를 파일이나 환경변수, 문서 등에 평문으로 보관하거나 특정 관리자만 알고 있는 방식으로 운영할 경우 키 유출 위험을 완전히 배제하기 어렵다.

따라서 JWT 보안의 방향성은 서명키가 사람이나 애플리케이션 실행 환경에 노출되지 않도록 분리하여 보호하고, 침해 상황에서도 인증 피해를 최소화할 수 있는 구조를 마련하는 데 있다.

이를 위해 서명키 관리 관점에서 서명키 분리, 서명 권한 통제, 유출 가정 피해 제한 순으로 대응 방안을 정리한다.

1) 서명키 분리

본 챕터에서는 서명키 관리의 첫 단계로, 서명·검증 분리(대칭키/비대칭키)와 서명키 격리 운용을 통해 서명키가 사용되는 범위를 최소화하는 방안을 제시한다.

- **서명과 검증 분리**

JWT 기반 인증에서는 서버가 요청을 받을 때, 전달된 토큰이 정상적으로 발급된 것인지를 서명 검증 결과 하나로 판단한다. 즉, 서버는 "이 토큰이 어떤 키로 서명되었는가"만을 기준으로 인증 여부를 결정하며, 이 과정에서 사용되는 서명키는 토큰의 신뢰성을 결정하는 핵심 요소가 된다.

구분	서명 알고리즘	토큰 생성	토큰 검증
대칭키	HMAC-SHA256(HS256)	비밀키(Secret Key)	
비대칭키	RSA-SHA256(RS256)	개인키(Private Key)	공개키(Public Key)

표 2. 키 방식별 대표 JWT 서명 알고리즘

대칭키 기반 방식인 HMAC-SHA256(HS256)은 하나의 비밀키로 토큰 생성과 검증을 모두 수행한다. 구현이 단순하고 처리 속도가 빠르기 때문에, 단일 서비스나 비교적 단순한 인증 구조에서는 HS256 방식이 널리 사용되어 왔다. 그러나 인증 요청을 처리하는 모든 서버가 동일한 비밀키를 보유해야 하므로, 서비스가 확장될수록 키 배포 범위가 넓어지고 노출면이 커진다.

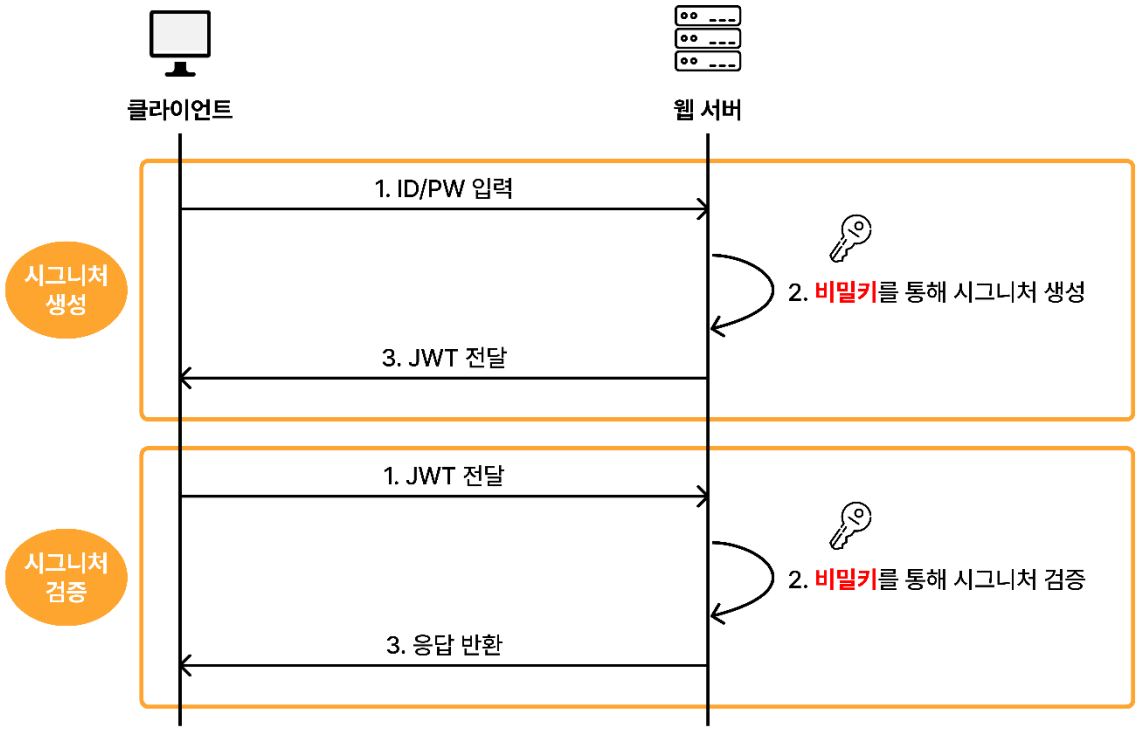


그림 22. 대칭키 기반 시그니처 생성 및 검증 과정

이 구조에서는 토큰을 검증하는 웹/API 서버가 비밀키를 직접 보유해야 하므로, 사용자 요청이 도달하는 영역 자체가 서명키 노출 지점이 된다. 따라서 서버가 침해되거나 키가 노출될 경우 공격자는 동일한 비밀키로 유효한 서명을 생성해 JWT 를 위조할 수 있다.

비대칭키 기반 방식인 RSA-SHA256(RS256)은 이러한 구조적 한계를 보완하기 위해, 토큰 생성에 사용되는 개인키와 검증에 사용되는 공개키를 분리한다. 개인키는 제한된 환경에서만 관리되며, 각 서비스 서버에는 토큰 검증에 필요한 공개키만 배포된다. 이로 인해 인증 요청을 처리하는 서버 수가 증가하더라도, 토큰을 생성할 수 있는 권한은 개인키를 보유한 서버로 제한된다.

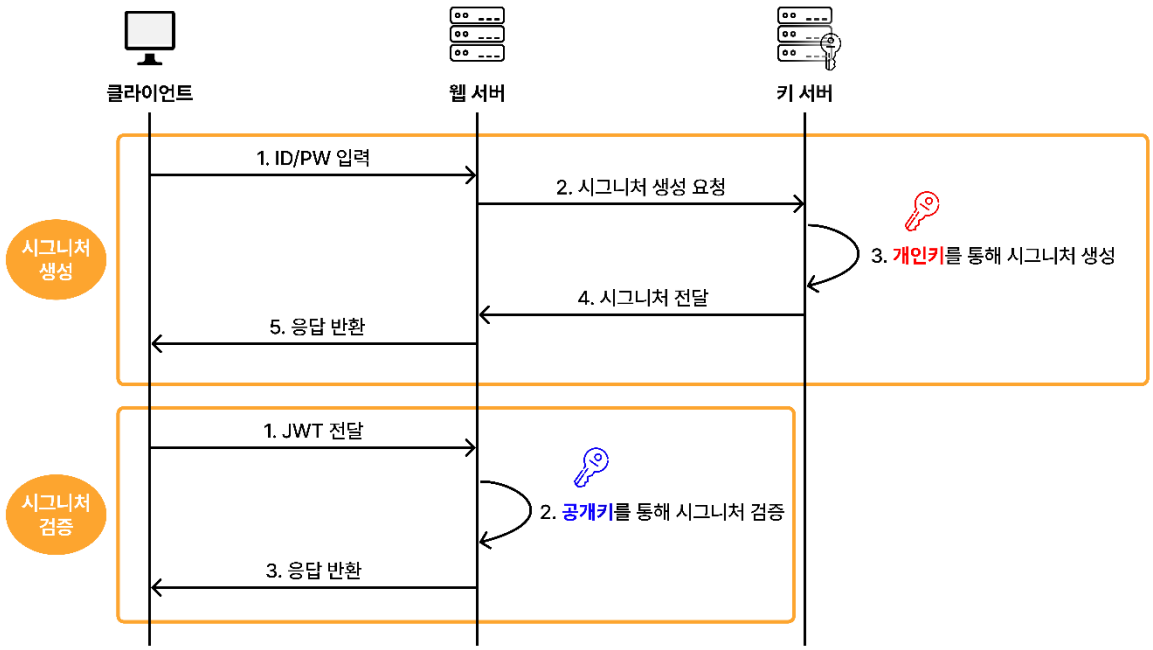


그림 23. 비대칭키 기반 시그니처 생성 및 검증 과정

이러한 구조에서는 검증 서버에 배포된 공개키가 노출되더라도 새로운 JWT 를 생성할 수 없다. 공개키는 토큰의 유효성을 판단하는 검증 용도로만 사용되며, 해당 키 자체로는 시그니처를 만들 수 없기 때문이다. 결과적으로 서명키가 포함된 환경과 검증 환경이 분리되어, 검증 서버의 키 노출이 토큰 위조로 이어지지 않는다.

또한 시그니처 검증은 서비스 규모와 관계없이 성능과 보안에 직접적인 영향을 미친다. 대칭키 기반 구조는 토큰 생성과 검증이 동일한 서버에 집중되어 트래픽 증가 시 성능 저하가 발생할 수 있으며, 침해 시 인증 위험이 확산된다. 반면 비대칭키 기반 구조는 시그니처 생성과 검증을 분리하여 서버 부하를 분산하고 생성 권한을 제한할 수 있다. 이러한 이유로 JWT 기반 인증에서는 비대칭키 기반 서명 방식이 구조적으로 더 안전한 선택으로 권장된다.

앞서 설명한 서명·검증 분리 구조는 서명키를 최소 범위에서만 사용하도록 제한하는 것을 목표로 하며, 이후 설명하는 HSM 과 같은 보안 장치는 이러한 구조에서 서명키를 보호하기 위한 수단으로 활용된다.

- **서명키 격리 운용**

JWT 환경에서 서명키 보호의 핵심은 키가 사람과 애플리케이션 실행 영역에 얼마나 노출되는지에 달려 있다. 서명키가 환경 변수, 설정 파일, 배포 산출물, 운영 문서 등에 포함되면 내부자 오남용이나 계정 탈취, 퇴사자 리스크 같은 현실적인 변수를 통해 키 유출 가능성이 생긴다. 따라서 서명키는 사람이 확인하거나 복사할 수 있는 값으로 취급하지 않고, 필요한 경우에만 제한된 방식으로 사용되도록 격리하는 운영 구조가 필요하다.

이를 위해 서명키 운영은 다음과 같은 방향으로 설계되어야 한다.

설계 방향	설명
중앙 관리 및 사람 개입 최소화	서명키를 코드/서버 설정에 포함하지 않고 중앙에서 관리하며, 배포·운영 과정에서 사람이 키 값을 직접 취급하는 절차를 최소화
키 미보관	애플리케이션이 키 값을 보관하지 않고 서명 대상 데이터만 전달하여 서명 연산을 요청하고 결과(시그니처)만 수신
사용 경로 제한 및 추적	서명키를 사용할 수 있는 주체와 경로를 제한해 노출 접점을 줄이고, 오·남용 탐지를 위해 서명 요청 이력을 남김

표 3. 서명키 노출 최소화를 위한 운영 원칙

이와 같은 키 노출 최소화 원칙을 기술적으로 쉽게 구현할 수 있도록 돕는 수단이 HSM(Hardware Security Module)이다. HSM 은 JWT 시그니처 생성 및 검증에 사용되는 중요한 키를 사람이나 애플리케이션이 직접 확인할 수 없도록 보호하기 위해 설계된 하드웨어 기반 보안 장치이다. 키는 HSM 내부에서만 생성·저장·사용되며, 외부 시스템은 키의 실제 값을 알지 못한 채 시그니처 생성과 같은 서명 연산을 요청하고 그 결과만을 전달받는다.

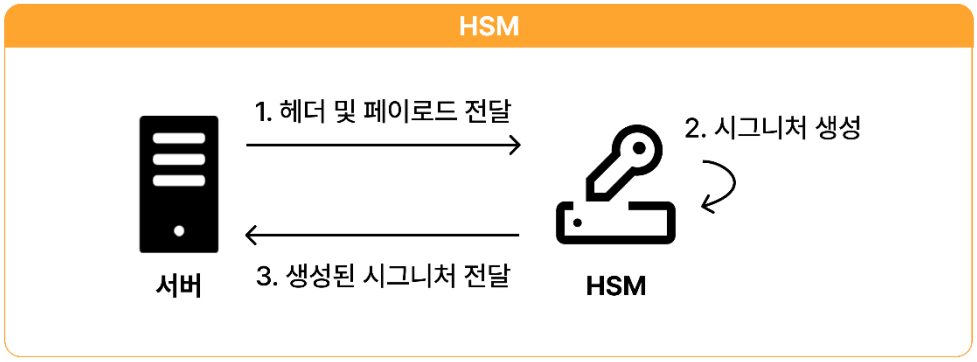


그림 24. HSM 사용 시 시그니처 생성 흐름

HSM 은 기술적으로 대칭키 방식에서 토큰 생성과 검증 모두에 사용될 수 있다. 그러나 모든 인증 요청에 HSM 을 연동하는 구조는 성능과 확장성 측면에서 운영 부담이 있다. 비대칭키 방식은 서명과 검증을 분리할 수 있어, HSM 을 서명키(개인키) 보호와 시그니처 생성에만 활용할 수 있다. 따라서 JWT 기반 인증 환경에서는 비대칭키 방식을 전제로, HSM 을 서명 전용으로 활용하는 방식을 권장한다.

HSM 은 목적에 따라 내장형, 네트워크형, 클라우드 기반 서비스 등으로 구분된다. 서버 간 인증이나 대규모 서비스 환경에서는 중앙에서 서명 연산을 처리할 수 있는 네트워크형 또는 클라우드 서비스 형태의 HSM 이 주로 사용된다. 반면 USB 형태의 HSM 은 주로 공인 인증서와 같은 개인 인증 용도로 활용되며, JWT 서명키 보호 목적에는 일반적으로 적합하지 않다.

구분	설명
내장형 HSM	서버 내부 슬롯(PCIe 등)에 직접 장착하는 카드 형태의 장치
네트워크형 HSM	독립된 전용 장비 형태로 네트워크를 통해 암호 연산 요청을 처리
USB형 HSM	휴대 가능한 USB 형태로 주로 개인 인증 및 서명 용도로 사용
칩형 HSM	반도체 칩 내부에 암호 연산 기능을 구현 (IoT 기기 등에 내장)
클라우드 HSM 서비스	클라우드 사업자(AWS, Azure 등)가 가상화된 환경에서 제공하는 HSM 서비스

표 4. HSM 종류

하지만 서명키 자체를 보호하더라도, 어떤 시스템이 서명 연산을 요청할 수 있는지에 대한 통제가 이루어지지 않으면 서명키가 유출된 것과 유사한 수준의 피해로 이어질 수 있다. 이러한 한계를 보완하기 위해, 다음에서는 서명 권한을 통제·추적하는 관리 체계를 중심으로 대응 방안을 살펴본다.

2) 서명 권한 통제

다음으로 서명 연산 요청 권한을 정책으로 제한하고 이력을 추적하는 통제 방안을 제시한다.

• KMS 기반 서명 권한 관리

HSM 을 통해 서명키의 외부 노출을 구조적으로 차단하더라도, 서명 연산을 요청할 수 있는 권한이 적절히 통제되지 않으면 보안 사고로 이어질 수 있다. 즉, 키 자체를 보호하는 것뿐 아니라, 누가 언제 어떤 목적으로 서명할 수 있는지에 대한 관리 역시 중요하다. 이러한 역할을 담당하는 것이 KMS(Key Management System)이다.

KMS 는 여러 애플리케이션과 서비스에서 사용하는 암호키를 중앙에서 관리하며, 키 생성부터 저장, 사용, 로테이션, 폐기까지의 전 과정을 정책 기반으로 통제한다. 이를 통해 관리자는 어떤 시스템이 어떤 키로 언제 서명 연산을 요청했는지를 확인할 수 있고, 서명 연산(키 사용) 권한을 서비스 또는 역할 단위로 세밀하게 제한할 수 있다. 또한 감사 로그와 모니터링 기능을 통해 비정상적인 서명 요청을 탐지하고 대응할 수 있다.

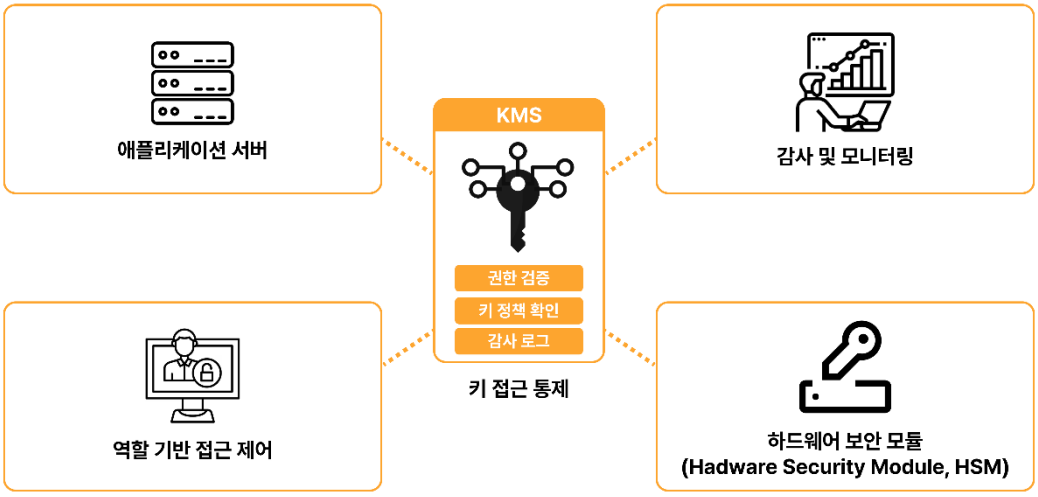


그림 25. KMS 기반 중앙 키 관리 아키텍처

주요 클라우드 서비스에서는 KMS 를 서비스 형태로 제공한다. 클라우드 KMS 는 IAM 기반 접근 제어, 자동 로테이션, 감사 로그 연동 등을 기본적으로 지원하며, 필요에 따라 클라우드 HSM 과 연계해 서명키를 하드웨어 수준에서 보호할 수 있다. 이 경우 애플리케이션은 KMS 를 통해 서명 요청을 수행하며, 실제 서명키는 클라우드 제공자의 보호된 환경 내에 유지된다.

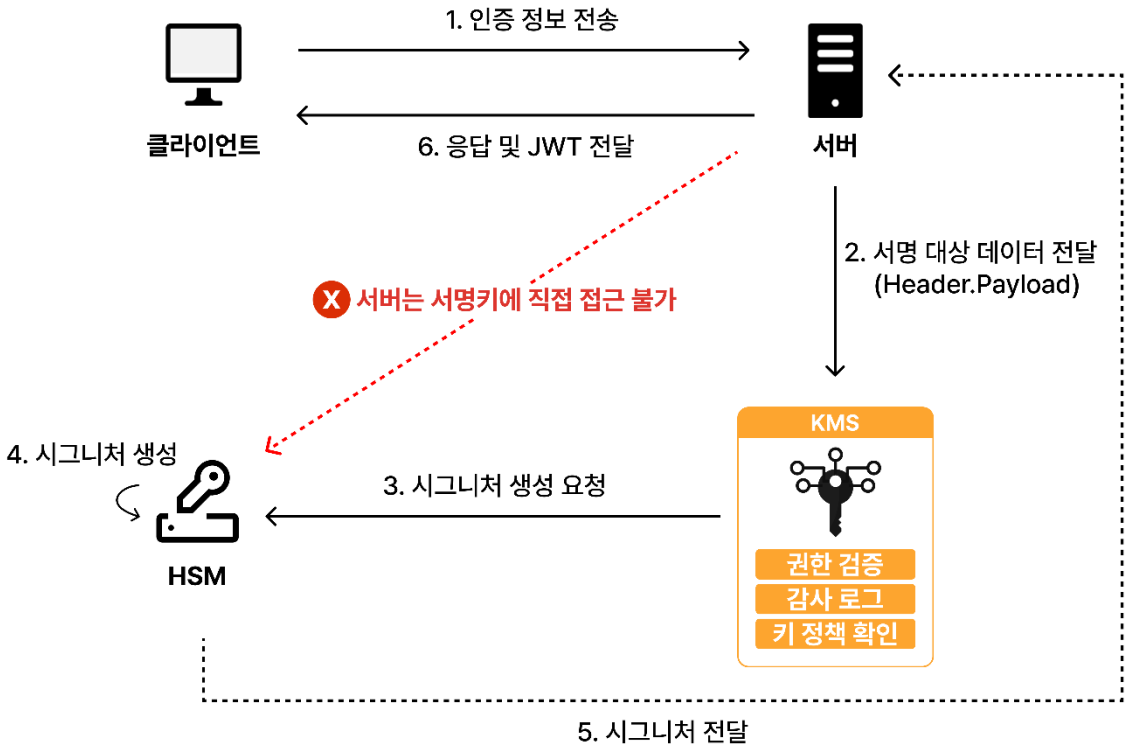


그림 26. KMS 와 HSM 을 통한 JWT 시그니처 생성 흐름

반면 금융권, 공공기관, 폐쇄망 환경과 같이 외부 클라우드 서비스 사용이 제한되거나 자체 보안 가이드라인을 따라야 하는 조직에서는, 동일한 개념을 온프레미스 환경에서 직접 구현해야 한다. 이 경우에도 대응의 핵심은 서명키를 중앙에서 관리하고, 사용 권한과 이력을 통제하는 구조를 갖추는 것이다.

구현 방식은 조직의 인프라와 보안 요구 수준에 따라 달라질 수 있지만, 서명 요청 권한의 분리, 사용 이력 기록과 감사, 주기적인 키 로테이션은 온프레미스 환경에서도 공통적으로 고려되어야 할 핵심 요소이다. 특히 키 로테이션은 동일 키의 장기 사용을 피함으로써 키 노출 시 영향 기간을 제한하고, 사고 대응 시 키 교체를 통해 피해 확산을 신속히 차단할 수 있게 한다.

이러한 구조를 기반으로 HSM 과 KMS 를 함께 적용하면 서명키 유출로 인한 위험을 효과적으로 완화할 수 있다. 그러나 보안 사고 가능성을 완전히 배제할 수는 없으므로, 다음으로는 이를 전제로 인증 피해의 확산을 최소화하기 위한 설계적 보완 방안을 살펴본다.

3) 서명키 유출 대비 피해 제한

마지막으로 서명키 유출 가능성을 전제로, sub 클레임 설계와 중요 기능 추가 검증을 통해 피해 확산 범위를 제한하는 완화 방안을 제시한다.

• sub 클레임 설계

앞선 대응 방안들은 서명키를 보호하고 관리하는 데 초점을 맞추고 있지만, 현실적으로 키 유출 가능성을 완전히 배제하는 것은 어렵다. 따라서 JWT 보안에서는 키 보호와 더불어, 서명키가 유출되었을 때 피해가 어디까지 확산될 수 있는지를 제한하는 설계적 보완 역시 함께 고려되어야 한다.

JWT 의 sub 클레임은 토큰이 어떤 사용자를 나타내는지를 식별하기 위한 값으로, 서버는 시그니처 검증이 완료된 토큰에 대해 sub 값을 기준으로 사용자 계정을 식별한다. 이때 sub 값이 내부 사용자 ID 와 같이 순차적이거나 의미를 가지는 정수값으로 구성되어 있을 경우, 서명키 유출 시 구조적인 위험이 발생할 수 있다. 공격자는 유효한 서명키로 토큰을 생성할 수 있는 상황에서 sub 값을 변경하며 다른 사용자로 가장한 JWT 를 반복적으로 만들어낼 수 있기 때문이다.

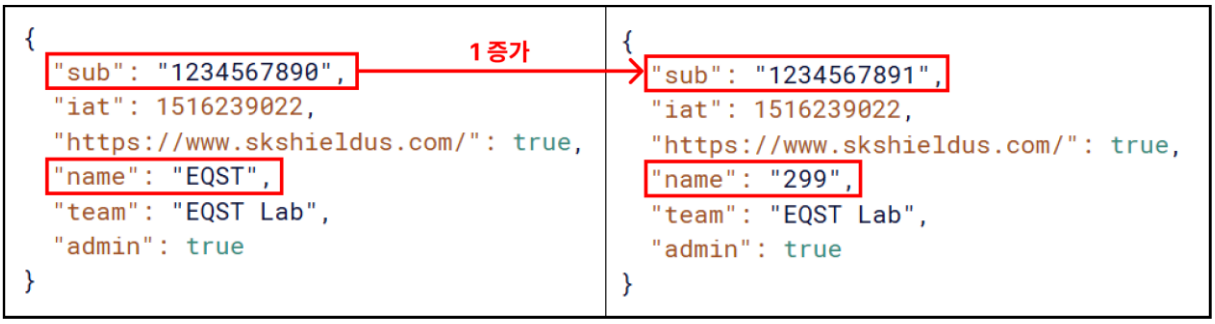


그림 27. sub 클레임 예시

이러한 구조에서는 개별 계정 침해를 넘어, 자동화된 방식으로 다수 계정을 순차적으로 시도하는 공격으로 확산될 가능성이 높아진다. 즉, sub 값의 설계 방식이 서명키 유출 시 피해 범위를 결정짓는 요소로 작용하게 된다.

이를 완화하기 위해 sub 클레임에는 외부에 노출되더라도 사용자 정보를 직접적으로 추정하기 어려운 비의미적 식별자를 사용하는 것이 바람직하다. UUID 와 같은 값은 사용자 식별 기능은 유지하면서도, 값의 연속성이나 범위를 기반으로 한 추측을 어렵게 만들어 자동화된 계정 전환 공격의 난이도를 높인다.

비의미적 식별자 적용은 서명키 유출을 근본적으로 차단하는 대응책은 아니지만, 침해 발생 시 공격자의 행위를 제한하고 피해 확산 속도를 늦추는 현실적인 보완 전략이다. JWT 표준에서 sub 클레임은 토큰 주체를 식별하기 위한 등록 클레임으로 정의되어 있으므로, 해당 역할을 전제로 외부 노출을 가정한 안전한 값 설계가 필요하다.

• **중요 기능 추가 검증**

서명키 보호와 sub 클레임 설계를 통해 토큰 위조와 계정 확산 위험을 완화하더라도, JWT 기반 인증 구조에서는 유효한 토큰을 보유한 요청을 정상 인증으로 처리한다는 특성 자체를 완전히 제거할 수는 없다. 따라서 키 유출이나 토큰 탈취가 발생한 상황을 전제로, 토큰 하나로 모든 기능 접근을 허용하지 않는 구조적 보완이 필요하다.

실제 서비스 환경에서는 개인정보 조회, 계정 정보 변경, 결제·환불과 같은 민감한 기능에 접근할 때, 로그인 상태와 별도로 비밀번호 재입력과 같은 추가 검증 절차를 요구하는 경우가 많다. 이는 인증 토큰의 신뢰 범위를 제한함으로써 단일 토큰 탈취로 인한 피해를 줄이기 위한 설계이다.

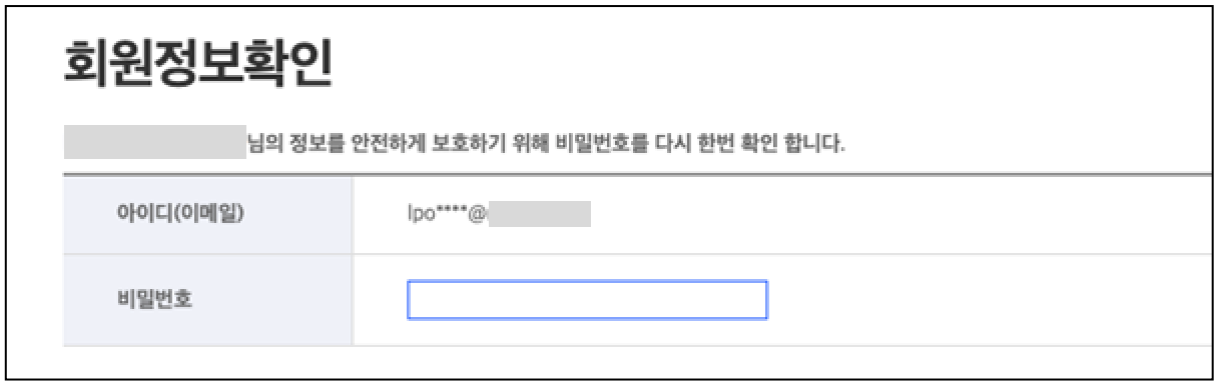


그림 28. 개인정보 페이지 접근 시

이와 같은 추가 검증은 JWT 의 서명 검증 이후에 수행되며, 토큰이 유효하더라도 특정 행위에 대해서는 사용자 재확인 또는 별도의 인증 단계를 거치도록 구성 가능하다. 이를 통해 공격자가 유효한 JWT 를 확보하더라도, 민감한 API 나 핵심 기능에 대한 직접적인 악용을 어렵게 만들 수 있다.

결과적으로 JWT 기반 인증 환경에서는 모든 기능을 동일한 신뢰 수준으로 처리하기보다, 기능의 중요도에 따라 인증 강도를 차등 적용하는 구조가 현실적인 보완 전략이 될 수 있다. 이는 서명키 유출이라는 최악의 상황에서도 서비스 전반의 피해를 제한하기 위한 추가적인 방어 계층으로 작용한다.

■ 마무리

JWT 환경에서 서명키가 유출되면 공격자는 정상 사용자와 구분하기 어려운 토큰을 생성해 사용할 수 있으며, 탐지가 지연될 경우 피해가 빠르게 확대될 수 있다. 따라서 JWT 보안의 핵심은 서명 알고리즘 선택 이전에 서명키를 어떻게 보호하고, 통제하는가에 있다.

특히 “관리자나 내부 인력은 알고 있어도 된다”는 운영 방식은 다양한 운영·접근 경로를 통해 키가 노출될 여지를 만든다. 따라서 서명키는 사람이 직접 보관·공유하는 대상이 아니라, 격리된 환경에서 필요 시에만 제한된 권한으로 사용되도록 설계되어야 한다. 이러한 원칙이 지켜질 때, 침해 상황에서도 인증 체계 전반으로 피해가 확산되는 위험을 실질적으로 줄일 수 있다.

■ 참고 사이트

• Microsoft Security (Storm-0558): https://www.microsoft.com/en-us/security/blog/2023/07/14/analysis-of-storm-0558-techniques-for-unauthorized-email-access/
• Microsoft (Golden SAML): https://www.microsoft.com/en-us/msrc/blog/2020/12/customer-guidance-on-recent-nation-state-cyber-attacks
• MITRE ATT&CK (Golden SAML): https://attack.mitre.org/techniques/T1606/002/
• RFC7519: https://datatracker.ietf.org/doc/html/rfc7519
• JWT
 -https://www.jwt.io/introduction#what-is-json-web-token-structure
 -https://cheatsheetseries.owasp.org/cheatsheets/JSON_Web_Token_for_Java_Cheat_Sheet.html
 -https://learn.microsoft.com/en-us/entra/identity-platform/id-token-claims-reference
• AWS KMS: https://docs.aws.amazon.com/kms/latest/developerguide/overview.html
• AWS CloudHSM: https://docs.aws.amazon.com/cloudhsm/latest/userguide/introduction.html
• Azure key-vault: https://learn.microsoft.com/en-us/azure/key-vault/general/basic-concepts
• Coupang:
 -https://pages.coupang.com/f/160047
 -https://news.coupang.com/archives/58857/
 -https://v.daum.net/v/20251202212010142

EQST INSIGHT

2026.01

SK쉴더스

SK쉴더스(주) 13486 경기도 성남시 분당구 판교로227번길 23, 4&5층
https://www.skshieldus.com

발행인 SK쉴더스 EQST사업그룹
제 작 SK쉴더스 마케팅그룹