API 보안 모범 사례:
API 보호를 위한 핵심 고려 사항

위험에 노출되었다고 느끼십니까? 보이지 않는 위협을 극복하는 방법을 알아보십시오.

소개

API(Application Programming Interface)는 디지털 에코시스템들이 서로 통신을 주고받으며 상호 작용하는 방식을 간소화하기 위해 개발됐습니다. 본질적으로 API는 여러 개별 시스템을 연결하는 데 따른 복잡성을 추상화합니다. 이를 통해 개발자들은 신속하고 손쉽게 자체 애플리케이션들에 타사 콘텐츠 또는 서비스를 통합하고, 일상적인 작업들을 자동화하며, 온라인 쇼핑, 원스톱 여행 계획 등 다양한 디지털 서비스들을 위한 편리성을 높일 수 있습니다.

점점 더 연결되는 오늘날 세계에서 API는 개발자들이 처음부터 다시 개발하지 않고도 신속하게 새로운 기능을 구현할 수 있도록 하는 수단으로 출발했으며, 이제 디지털 경제의 필수 구조로 자리매김하고 있습니다. 모던 애플리케이션들은 계속해서 발전을 거듭하면서 현대화를 향한 다리를 제공하기 위해 점차 타사 API에 의존하게 됐습니다. API는 이제 보다 완벽하고, 편리하며, 강력한 디지털 경험을 위한 토대가 되고 있습니다.

애플리케이션 개발을 위한 기본 구성 요소로서 API의 중요성이 높아지면서, 일련의 인프라 및 배포 시나리오들이 파생되었으며 특히 아키텍처가 급격하게 분산되었습니다. 하지만, API에 대한 의존도가 증가하면서 기존 보안 툴로는 감지하고 완화하기 어려운 침해, 남용 및 부정 행위에 대한 위험 경로도 크게 늘어났습니다.

탈중앙집중화된 분산 아키텍처 내 API 보호의 과제

API는 애플리케이션 간의 연결을 용이하게 만들고 새로운 콘텐츠와 서비스를 쉽게 통합할 수 있기 때문에 오늘날 디지털 경제의 토대로 자리매김했습니다. API는 기존 디지털 솔루션에 기능을 추가하는 가장 빠르고 가장 효율적인 방법 중 하나입니다. API는 기존 애플리케이션에 디지털 맵 서비스 또는 온라인 쇼핑몰 장바구니 등과 같은 새로운 서비스, 데이터, 기능 등을 쉽게 추가할 수 있도록 합니다. 그 결과, API는 레거시 애플리케이션들을 현대화하고 웹 서비스 개발의 속도와 효율성을 높이는 데 필수적입니다.

빠르고, 신속한 응대가 이루어지며, 마찰없는 디지털 경험이 오늘날 기업들을 결정짓는 요소이자 차별화 요인으로 작용하면서, API는 기업의 고객, 파트너, 공급업체, 직원 및 기타 사용자들을 연결하는 강력하고 보편적인 방법이 되었습니다.

또한, API는 모바일 앱, 싱글 페이지 애플리케이션(SPA, Single Page Application) 및 마이크로서비스가 주도하는 다양한 개발 플랫폼 또는 프레임워크에 데이터, 콘텐츠, 기능 등을 제공하는 안정적이고 효율적인 방법을 제공합니다. API는 에코시스템 전반에서 사용 용이성, 성능 및 신뢰성을 향상시켜 보다 우수한 사용자 경험과 귀중한 데이터 및 정보에 대한 신속한 액세스를 제공함으로써 고객 및 파트너와 함께 안정적이고 안전한 협업과 혁신을 수행할 수 있도록 합니다.

항공편 예약은 물론, 렌트카 서비스, 온라인 여행 가이드, 날씨 정보 및 실시간 비행 상태 등을 제공하는 여행 앱을 생각해 보십시오. 이들 각 서비스와 콘텐츠 리소스는 타사 API를 통해 제공됩니다. 개발자들은 보다 효율적인 애플리케이션 개발과 시장 출시 시간 단축을 위해 이들 API를 활용할 수 있으며, 커스텀 개발 작업 없이 부가 가치 높은 서비스를 제공함으로써 차별화된 고객 경험을 실현할 수 있습니다.

상호 연결 기능을 수행하는 API를 통해 애플리케이션들은 점차 분산되고 탈중앙집중화된 모델로 진화하고 있습니다. API는 일반적으로 외부에 노출되도록 설계되었기 때문에 기업이 보호해야 하는 민감한 데이터로 향하는 경로가 될 수 있습니다.

모던 API 딜리버리 설계는 혁신적이고 유동적이며 애플리케이션 구성 요소들이 여러 환경에 분산되어 있기 때문에 모든 요소들을 일관되게 관리하고 보호하는 것이 어렵습니다. API 구성 요소들은 여러 클라우드 제공업체 내에 위치하고 있을 수 있으며, 하이브리드 데이터센터와 프라이빗 및 퍼블릭 클라우드 등이 조합된 환경 내에 있을 수 있습니다. 또한, 컨테이너와 서버리스 시스템은 보호가 필요한 임시 구성 요소들을 생성합니다. 점점 더 분산되는 이러한 모델 각각에서 이기종 인프라 전반에 분산된 API들은 중앙 집중식 보안 제어 범위 밖에 있기 때문에 악의적인 해커의 공격 가능성과 잠재적인 진입 지점이 늘어나고 있습니다.

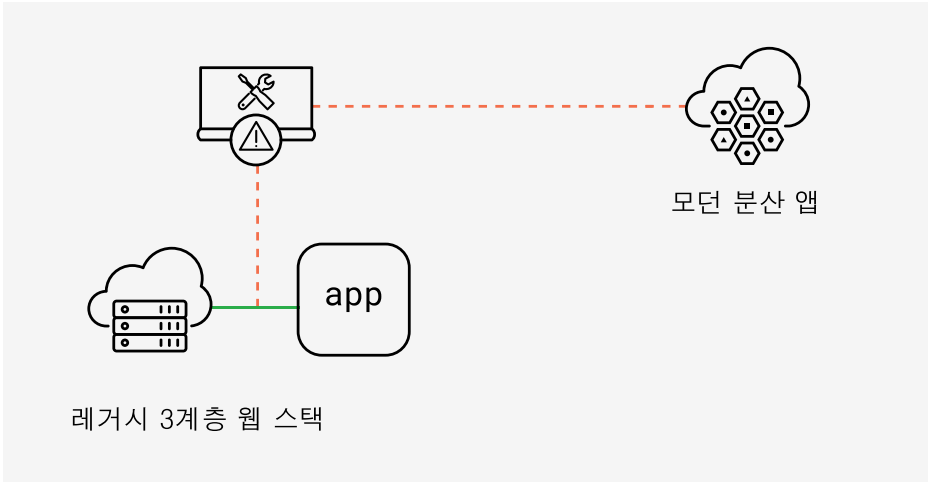


그림 1: 아키텍처 탈중앙화로 인해 초래된 복잡성은 위협 노출 가능성을 크게 높였습니다.

그 결과, API는 **레거시 애플리케이션들을 현대화하고** 웹 서비스 개발의 속도와 효율성을 높이는 데 필수적입니다.

API는 웹 애플리케이션과 동일한 공격을 받습니다.

API는 취약점 악용과 봇 오용을 비롯해 웹 애플리케이션들을 겨냥한 공격과 동일한 많은 공격에 취약하기 때문에 API 보안 사고들은 세간의 이목을 끄는 많은 데이터 침해의 원인이 되었습니다. 이들 유형의 공격에는 해커가 허가 없이 무단으로 컴퓨터 네트워크에 액세스하고 기밀 정보를 추출하는 데이터 침해가 포함됩니다.

API는 봇과 악성 자동화를 통한 공격에도 취약합니다. 봇은 인터넷에 널리 퍼져 있습니다. 실제로 F5 Labs 분석에 따르면, 사용자 대상 인증 방식에 대한 공격은 종종 봇에 의해 수행된 가장 빈번한 침해 원인인 것으로 나타났습니다.[1]

봇은 자동화되고 반복적이며 사전에 정의된 작업을 실행하는 방식으로 작동하며 인터넷에서 사람의 행동을 모방하는 소프트웨어 프로그램입니다. 고객 서비스를 자동화하거나, 소셜 네트워크에서 인간 커뮤니케이션을 시뮬레이션하거나, 검색 엔진을 색인화하는 등 유용한 기능을 실행할 수 있습니다. 하지만, 봇은 크리덴셜 스터핑(credential stuffing)을 통한 금융 데이터 및 개인 정보 유출과 계정탈취(ATO)를 통한 다른 형태의 사기 및 오용을 비롯해 악의적인 작업을 실행하라는 명령을 받을 수도 있습니다.

또한, 허위 네트워크 트래픽으로 시스템에 과부하를 발생시켜 정상적인 웹 서비스 네트워크나 API를 마비시키는 대규모 DDoS(Distributed Denial of Service) 공격을 감행하여 서비스 중단을 초래할 수 있습니다. 자동화된 DDoS 공격은 서버 및 네트워크 리소스를 소모하거나, 기업들이 비즈니스를 위해 의존하는 웹 서비스를 무력화시키거나 회사 네트워크에 과부하를 발생시켜 완전히 중지시킵니다.

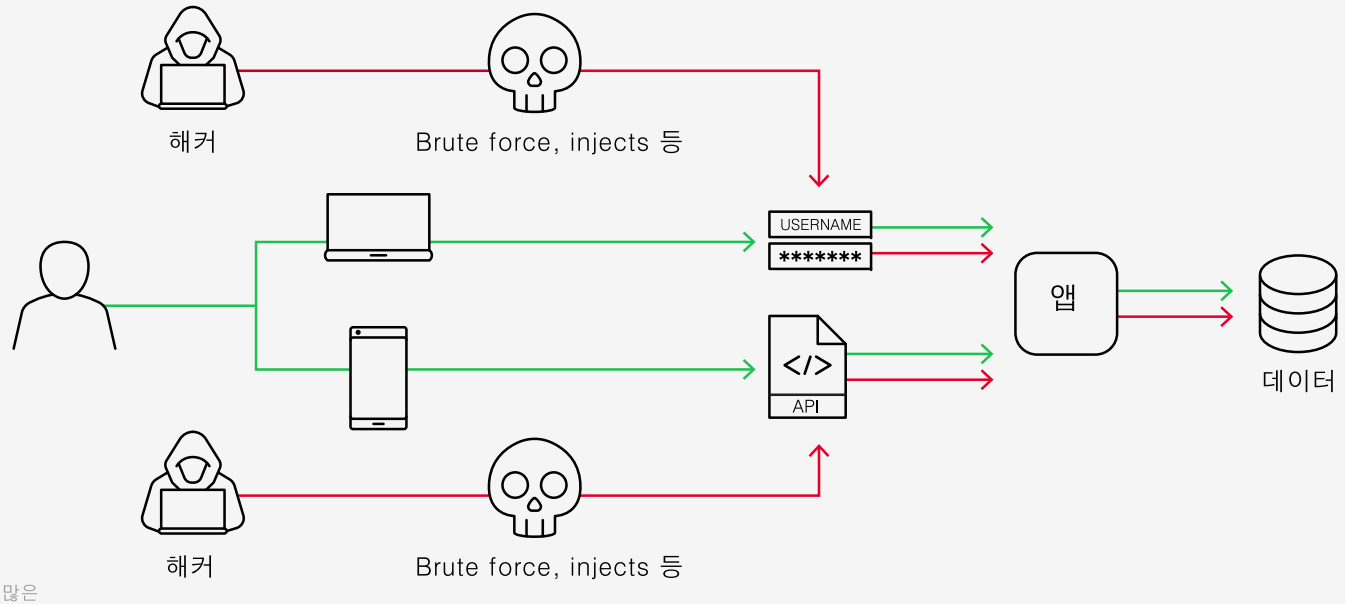
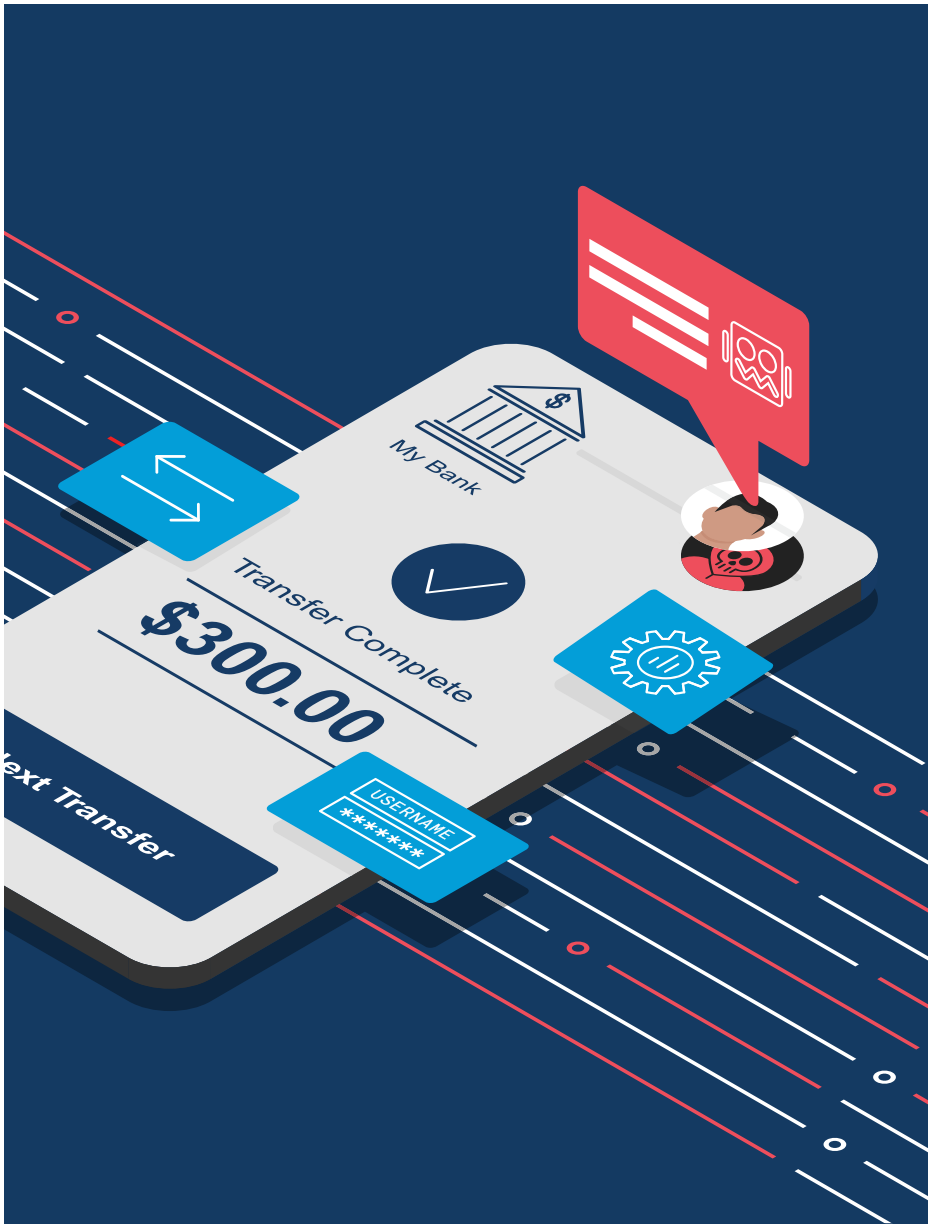


그림 2: API는 웹 애플리케이션과 동일한 많은 보안 취약점들에 공격을 받기 쉽습니다.



따라서, 리스크 관리 리더들은 레거시 웹 애플리케이션과 모던 API 패브릭 모두에 대해 가동 시간과 안정성에 영향을 미치는 것은 물론, 해킹과 데이터 침해로 이어지는 봇 기반 공격에 관심을 기울여야 합니다.

기존 보안 제어 기능으로는 보호할 수 없는 API

기본적인 웹방화벽(WAF), SIEM(Security Information and Event Management) 시스템 등과 같은 전통적인 보안 제어 기능들은 API에 대한 공격을 식별하고 차단하는 데 역부족입니다. 여기에는 대용량 M2M(machine-to-machine)이나 API-to-API 트래픽이 일부 원인이 되고 있습니다. 공격과 침해는 외견상 정상적인 애플리케이션 활동으로 보일 수 있지만, 이면에서는 API들이 악용되고 오용되어 너무 늦어버릴 때까지 해커들이 탐지를 피할 수 있도록 합니다.

데이터센터 내 서버와 애플리케이션 간 네트워크 통신인 east-west API 트래픽은 종종 중앙집중식 제어를 우회합니다. 이는 데이터센터의 경계로 들어오고 나가는 클라이언트-서버 간 통신을 일컫는 north-south 네트워크 트래픽과 달리, 일반적으로 여러 유형의 방화벽, IDS(Intrusion Detection Systems), DDoS 완화 장치, 그리고 데이터센터 입구의 기타 다계층 보호 기능을 통해 보호됩니다. 심지어 대기업들의 경우, 데이터센터와 여러 프라이빗 및 퍼블릭 클라우드 간에 흐르는 대량의 east-west 트래픽이 발생하고 있지만, east-west 트래픽과 관련하여 악의적인 활동을 모니터링하는 단일 탐지 지점이 없습니다.

입력 유효성 검사는 때로 데이터 유효성 검사로도 불리며, 기업들이 자체 애플리케이션을 보호하기 위해 시도하는 또 다른 방법입니다. 입력 유효성 검사는 사용자가 제공한 모든 입력을 검사하는 리스크 제어 방법으로서, 부적절하게 작성된 데이터가 비즈니스 로직을 실행하는 것을 방지합니다. 하지만, API는 M2M(machine-to-machine) 통신과 데이터 교환을 위해 설계되었기 때문에, 기업의 가장 민감한 데이터로 향하는 직접적인 경로를 나타낼 수 있습니다. 따라서, API에 대한 입력 유효성 검사는 사용자 대상 인식 웹 양식만큼 강력하지 않거나, 철저하게 테스트되지 않을 수 있습니다.

입력 유효성 검사는 사용자가 제공한 모든 입력을 검사하는 리스크 제어 방법으로서, 부적절하게 작성된 데이터가 비즈니스 로직을 실행하는 것을 방지합니다.

멀티 클라우드 아키텍처와 여러 데이터센터를 사용하는 아키텍처는 포괄적인 관리 전략 없이 API가 널리 배포되는 경우, 보안을 복잡하게 만들고 무분별한 API 확산(sprawl)을 초래할 수 있습니다. 이러한 아키텍처들은 보호하기 어려우며, 환경 전반에 예측 가능하고 신뢰할 수 있는 관리가 어렵다는 점이 그 이유 중 하나로 작용합니다. 여러 클라우드와 데이터센터들을 포괄하는 애플리케이션 배포를 유지하는 데 따른 복잡성 때문에 많은 이러한 아키텍처에서 일관성 없는 공격 감지와 정책 적용 문제가 흔히 나타나고 있으며, 이는 알려지지 않은 위험과 심각한 보안 격차를 초래할 수 있습니다.

예기치 못한 위험을 야기하는 모던 애플리케이션 라이프사이클

많은 기업들이 코드 변경과 새로운 소프트웨어 버전을 신속하고 안정적으로 배포할 수 있도록 하는 CI/CD(Continuous Improvement/Continuous Delivery) 개발 프로세스를 사용하고 있습니다. 하지만, 모든 소프트웨어 변경은 위험을 초래하거나 증가시킬 수 있습니다. 애플리케이션 개발 팀들이 보다 신속한 혁신을 위해 계속해서 CI/CD 파이프라인을 사용함에 따라, 점차 더 많은 API 호출이 비즈니스 로직 내에 깊이 숨겨져 검색과 식별이 극히 어려워지고 타사 API를 통한 공격 위험이 증가할 수 있습니다.

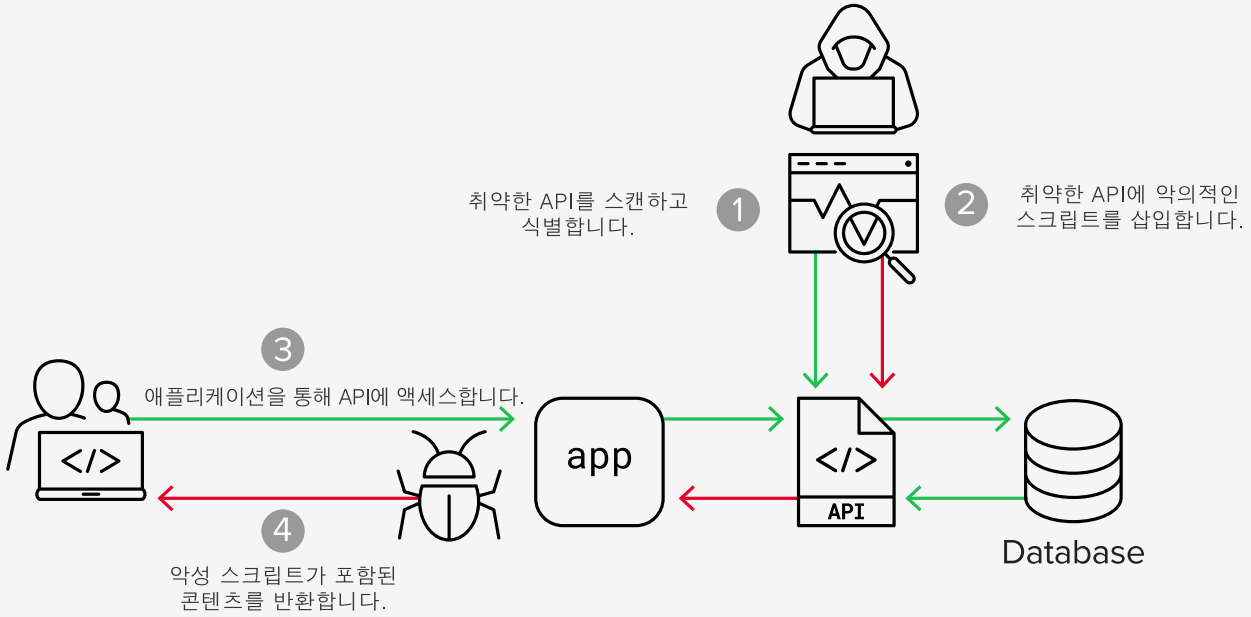
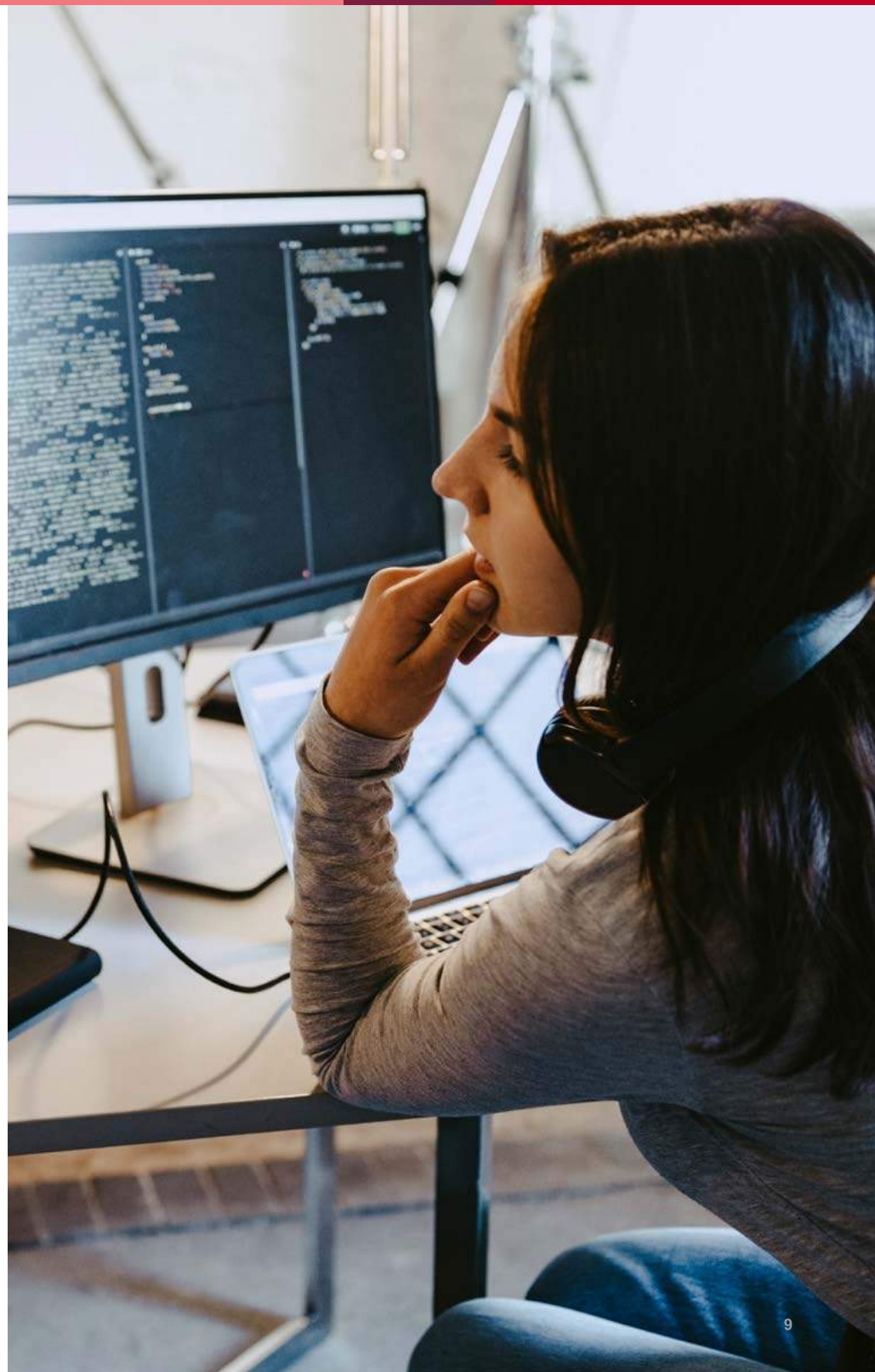


그림 3: API들은 비즈니스 로직 내 깊은 곳에 내장될 수 있지만, 여전히 공격에 취약합니다.

타사 API들을 이용함에 따라, 기업들은 많은 시간을 절약할 수 있으며 개발 프로세스를 단순화하고 다른 기업들이 개발한 특징과 서비스에 쉽게 액세스할 수 있습니다. 하지만, 조직들은 자체적으로 개발한 API를 제어할 수 있기 때문에, 타사 API를 이용하는 경우, 집 창문을 활짝 열어 사이버 범죄자들이 자신의 데이터와 애플리케이션에 쉽게 액세스하는 것을 허용하게 됩니다. 오픈소스 로깅 라이브러리(open-source logging library)와 같이 무해하게 보이는 기능이 치명적인 결함과 취약점의 원인이 될 수 있습니다.[2] 개발자들이 오픈소스 라이브러리를 통해 이용할 수 있는 것과 같은 퍼블릭 API를 배포하면서, 종종 규정된 보안 프로세스와 절차를 우회하는 경우가 많습니다.

오늘날 기업들은 끊임없이 변화하는 위험 환경에 직면하고 있으며 항상 새로운 위협 경로가 나타나고 있습니다. 대표적인 예로, 소비자와 판매자에게 대체 결제 옵션 또는 기타 금융 서비스를 제공하는 회사와 같은 제3자 결제대행기관(aggregator)을 들 수 있습니다. 제3자 결제대행기관들은 소비자와 판매자에게 중요한 서비스를 제공하지만, 새로운 차원의 위험도 초래하고 있습니다. 즉, 소비자의 개인정보를 유출하여 사기나 신원 도용을 초래할 수 있으며 판매자의 시스템을 있을 수 있는 공격에 노출시킬 수도 있습니다.

보안 측면에서, CI/CD 파이프라인 역시 많은 기업들에게 득실 양면이 있습니다. 기업들이 계속해서 자동화된 CI/CD 파이프라인의 사용을 통해 개발 및 배포 주기를 가속화하고 있지만, 상당 수는 소프트웨어 구성 요소의 검색과 평가를 위해 여전히 수동 프로세스를 사용하는 보안 정책을 가지고 있습니다. 이러한 접근 방식은 기본적으로 비실용적일 뿐만 아니라, 심지어 기업의 위험을 높일 수 있습니다. 한편, 시스템 내에 위협이 발생할 후 조치를 취하는 사후 대응식 보안은 소프트웨어 개발 및 출시 주기를 지연시키고 시장 출시 일정을 늦출 수 있습니다. 그 결과, 많은 기업들이 속도를 위해 보안을 포기하는 선택을 하면서 공격의 위험이 증폭되고 있습니다.



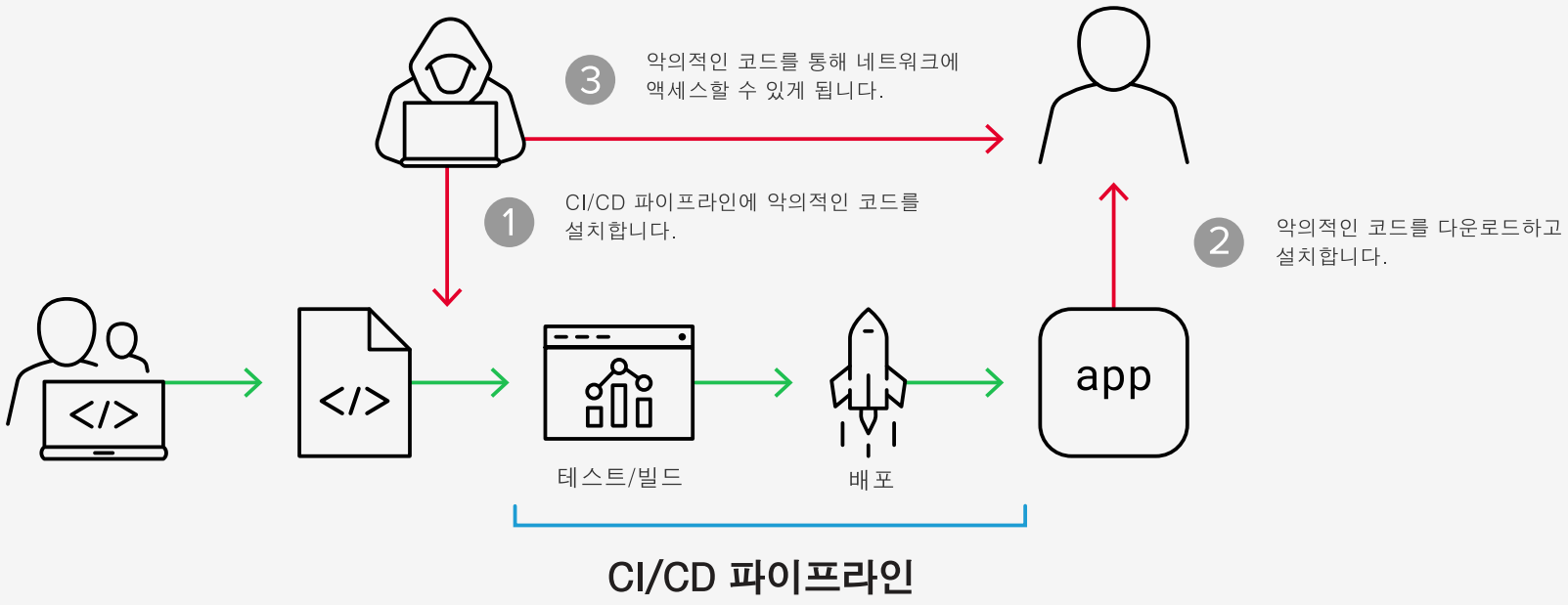


그림 4: CI/CD 파이프라인 자동화는 API 취약점들을 가릴 수 있습니다.

기본적으로 더 많은 위험을 초래하는 API

API는 많은 모던 애플리케이션의 기반이자, 여러 수준에서 혁신과 디지털 트랜스포메이션의 핵심 구성 요소입니다. API를 통해 개발자들은 타사 특징, 기능 및 서비스 등에 쉽게 액세스함으로써 이기종 시스템들이 원활하게 연동되도록 하고, 시장 출시 시간을 앞당기며, 고객 경험을 향상시킬 수 있습니다. 동시에 API는 해커가 기업 환경으로 들어올 수 있는 여러 창을 열기 때문에 불가피하게 새롭고 관리하기 어려운 위험을 초래합니다.

하지만, API 사용이 계속해서 급증하면서 2031년까지 API 수가 10억 개에 달할 것으로 예상되고 있으며, 공격 경로는 계속해서 확장되고 있습니다.[3] 기업들이 계속해서 API를 사용하여 자사 애플리케이션 포트폴리오를 혁신하고 현대화하면서 엔드포인트와 매개변수의 수가 증가하고 공격 위험도 높아지고 있습니다.

기본적으로 API는 공격자들에게 정찰 중에 수집할 수 있는 더 많은 정보를 제공합니다. API에 포함된 가용성과 정보는 시스템 내부에 대한 통찰력을 제공할 수 있으며, 이는 해커들이 표적 공격을 개발하는 데 도움이 될 수 있습니다. 특히, 익스플로잇 킷이 게시되자마자 공격자가 즉시 취약한 호스트를 찾기 위해 인터넷 스캐닝을 시작하는 log4j2와 같은 오픈소스 취약점의 경우 더욱 그렇습니다. 소프트웨어 공급망의 복잡성과 위험 평가의 어려움은 애플리케이션 및 보안 팀이 보다 긴밀하게 연계해야 하는 필요성을 강조합니다.[4]

API 보안을 위한 핵심 고려 사항

기업들이 API 보안을 강화하려고 준비할 때 고려해야 하는 몇 가지 중요한 사항들이 있습니다.

지속적인 API 엔트포인트 모니터링 및 보호

변화하는 애플리케이션 통합을 파악하고, 취약한 구성 요소들을 감지하며, 타사 통합을 통해 공격을 완화합니다.

포지티브 보안 (positive security) 모델 구현

OpenAPI 규격과 API Swagger 가져오기를 지원하고, 스키마 검증 및 프로토콜 준수를 시행하며, 정상 트래픽 패턴 기준을 자동으로 설정하는 것은 물론, 비정상적인 활동을 감지합니다.

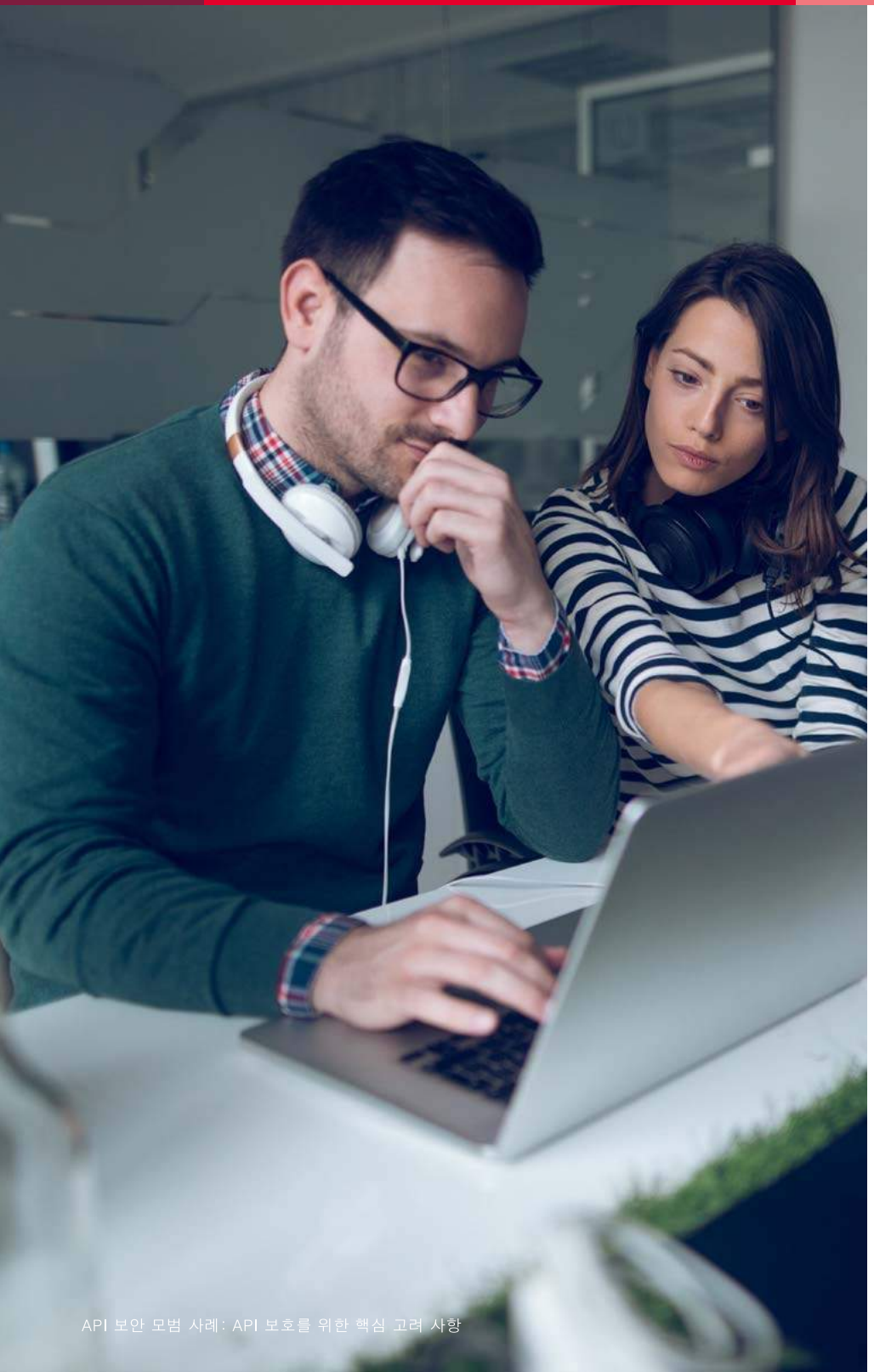
제로 트러스트(zero-trust) 및 위험 기반 보안 원칙 수용

이용 가능한 방법을 제한하고, 페이로드를 검사하며, 허가 받지 않은 데이터 노출을 방지하는 것은 물론, 개체 및 기능에 대한 액세스 제어와 위험 기반 인증을 구현합니다.

변화하는 애플리케이션 라이프사이클에 대응

변화하는 애플리케이션 라이프사이클에 대응 이기종 환경 전반에서 보안 구성 오류를 방지하고, 해킹과 DoS (Denial of Service) 공격을 야기할 수 있는 오용을 완화하며, 클라우드와 아키텍처 전반에서 일관되게 위협을 해결해야 합니다.

그림 5: 성공적인 API 보안 전략을 위해서는 다각적인 측면에서 경계가 이루어져야 합니다.



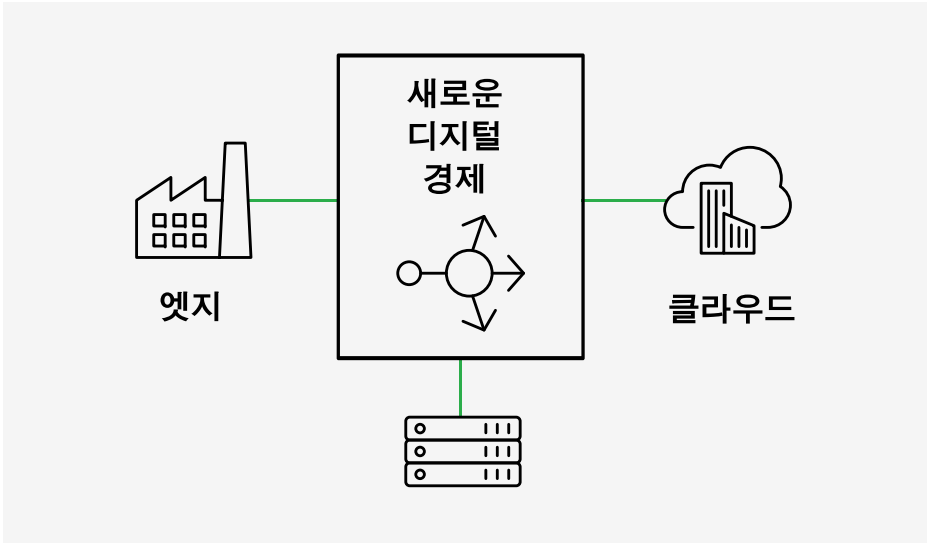


그림 6: API는 오늘날 탈중앙집중화되고 분산된 컴퓨팅 인프라의 핵심입니다.

결론

API는 디지털 비즈니스에서 중추적인 역할을 담당하고 있습니다. 하지만, 데이터센터의 커스텀 3계층 웹 스택과 같이 보다 예측 가능한 활용 사례가 있는 레거시 아키텍처에 비해, 그 특성상 안전하게 보호하는 것이 더욱 어렵습니다. API는 탈중앙집중화와 분산 아키텍처를 촉진하는 것은 물론, 타사 통합을 위한 무한한 기회를 제공하여 보안 및 리스크 팀의 셈법을 완전히 바꿔 놓고 있습니다.

부록

[1] Sander Vinberg and Raymond Pompon, "2022년 애플리케이션 보호 보고서: 유출에 대한 예상" F5 Labs (2022년 2월 15일) https://www.f5.com/labs/articles/threat-intelligence/2022-application-protection-report-in-expectation-of-exfiltration

[2] Andrew van der Stock, Brian Glas, Neil Smithline, Torsten Gigler, "2021년 OWASP Top 10: 새로운 위험의 물결, "OWASP (2021년 9월 24일) https://www.f5.com/services/resources/infographics/owasp-top-10-2021-infographic

[3] Rajesh Narayanan and Mike Wiley, "지속적인 API 확산: API 주도 경제의 과제와 기회" (2021년) https://www.f5.com/pdf/reports/f5-office-of-the-cto-report- continuous-api-sprawl.pdf

[4] Sander Vinberg, "Log4Shell: 이후의 재부팅(예전과 같은 방법) 보안 원칙," F5 Labs (2021년 12월 17일) https://www.f5.com/labs/articles/cisotociso/log4shell-rebooting-the-same-old-security-principles-in-its-wake

SECURE. SIMPLIFY. INNOVATE.

F5의 자동화, 보안, 성능 및 통찰력 기능은 고객들이 비용을 절감하고, 운영을 향상시키며 보다 효과적으로 사용자를 보호하는 적응형 애플리케이션을 개발하고 보호하며 운영할 수 있도록 지원합니다.

자세한 내용은 f5.com/solutions/api-security에서 확인해 보십시오.

US Headquarters: 801 5th Ave, Seattle, WA 98104 | 888-882-4447 // Americas : info@f5.com // Asia-Pacific: apacinfo@f5.com // Europe/Middle East/Africa: emeainfo@f5.com // Japan: f5j-info@f5.com