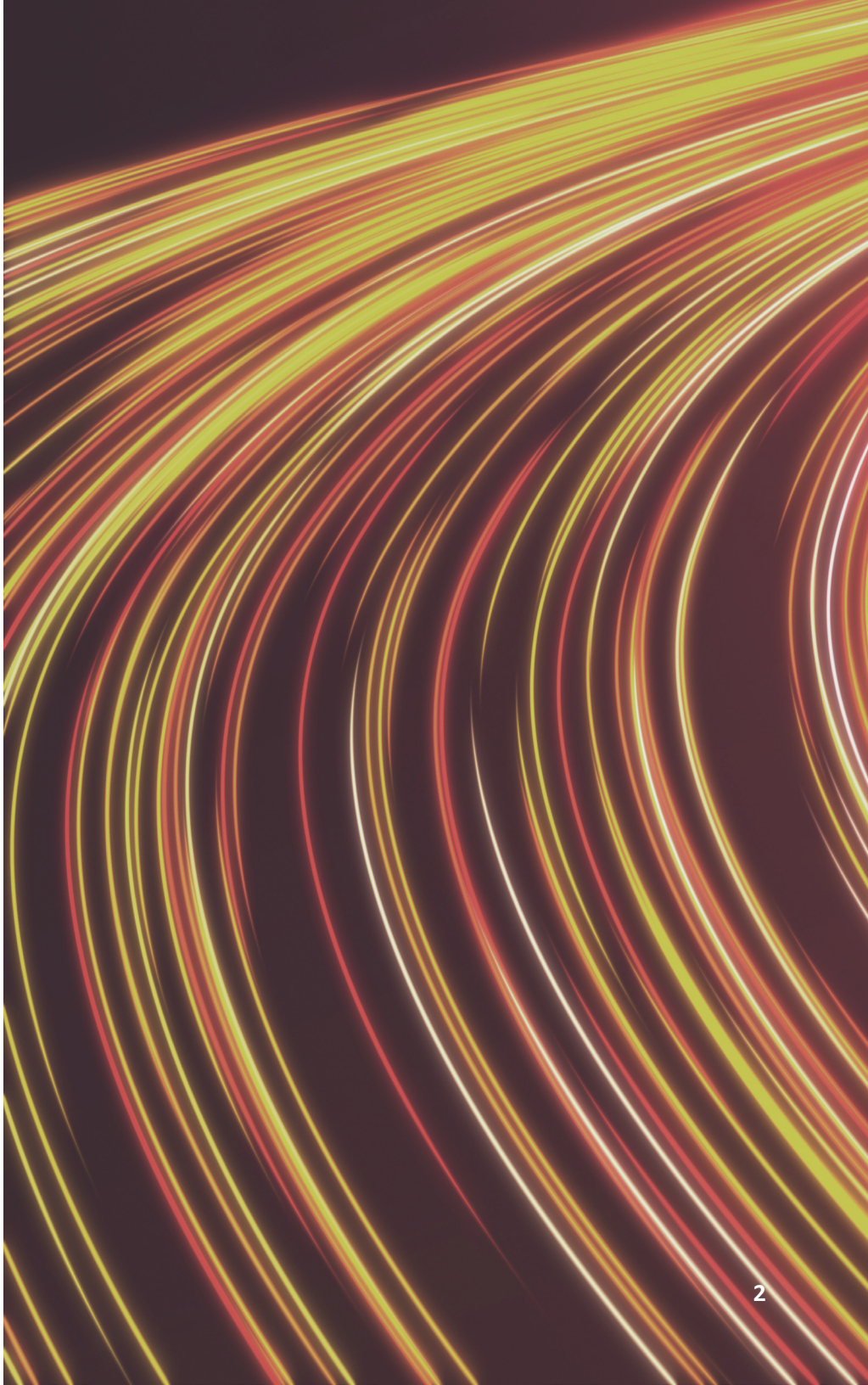

aws modern apps
현대적 Dev+Ops
모델 도입

현대적 Dev+Ops를 통해 조직은 개발 팀과 운영 팀의 간극을 좁힘으로써 고객의 진화하는 요구를 더 신속하고 일관적으로 충족할 수 있습니다.

DevOps는 세상을 변화시켰습니다. 매장의 상자 안, 디스크에 담겨 있던 소프트웨어를 AWS에 구축한 서비스의 형태로 디지털화하여 제공 방식을 바꾼 것은 Amazon과 인터넷이지만 그 소프트웨어의 업데이트 주기를 몇 개월, 몇 년에서 며칠로 바꾸고 개발 팀과 운영 팀을 긴밀하게 만든 것은 DevOps였습니다.

다양한 업종과 규모의 회사들이 워터폴 방식의 제품 관리 주기를 버리고, 혁신이 더욱 빠르고 보안과 성능, 탄력성이 향상되며 개발자와 고객 모두가 만족하는 민첩한 DevOps 방식을 도입하고 있습니다. 하지만 DevOps에는 이점이 많지만 완벽하지는 않습니다. DevOps는 클라우드에서 애플리케이션을 개발하고 호스트하는 것이 일반적인 관행으로 자리 잡기 이전에 생겨났기 때문에 오랜 시간 존재하기는 했지만 여전히 조직마다 DevOps를 다르게 해석할 여지가 있습니다. 예를 들어, 많은 조직에서는 아직까지도 DevOps가 하나의 전담 팀이라고 생각하며, 개발자가 모든 운영 업무를 담당하는 방식으로 생각하는 조직도 있습니다. 기술이 진화함에 따라 DevOps의 정의 또한 진화했습니다. 우리는 이 접근 방식을 현대적 Dev+Ops라고 부르며, 규정 준수, 관측성, 탄력성 및 인프라와 같은 운영 태스크를 개발 프로세스 초기에 공유하고 AI와 기계 학습으로 이를 개선함으로써 개발자와 운영을 더 가깝게 만드는 데 중점을 둡니다.



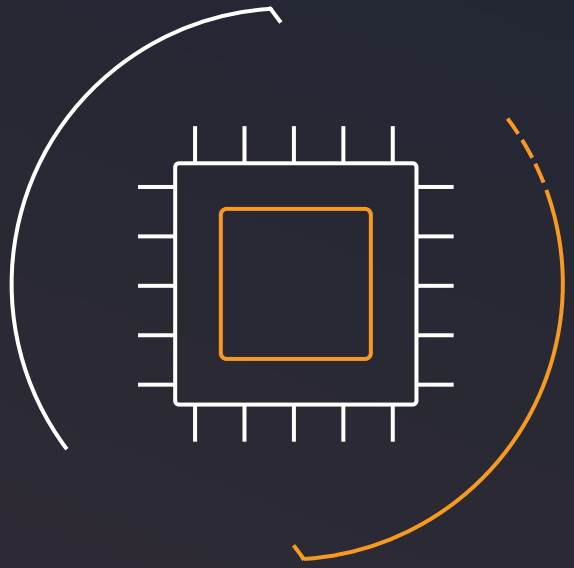
*State of DevOps Report*에 따르면 DevOps를 도입하고 배포 빈도를 주간/월간에서 시간/일간으로 변경한 회사는 리드 시간이 몇 달에서 며칠로 개선되었으며 변경 실패율도 46~60%에서 0~15%로 감소했습니다. 많은 고객에게 클라우드의 성공은 확장 가능한 온디맨드 인프라 도입에만 달린 것이 아니라 적어도 어느 정도는 개발과 운영 관행을 어떻게 전환했는지에도 달려 있습니다. 이제 Coca-Cola Argentina, 3M, Lululemon Athletica, The Washington Post와 같은 고객은 AWS Dev+Ops 서비스를 통한 현대적 Dev+Ops 접근 방식으로 전환하여 개발 팀과 운영 팀이 더욱 밀접하게 연계하도록 함으로써 DevOps의 더욱 큰 이점을 누리고 있습니다.

"6주씩 걸리던 것을 AWS를 통해 1주일에 하나씩 배포할 수 있게 되었습니다. 곧 하루에 몇 건씩 배포할 수 있으리라 예상합니다."[1]

- Rick Austin, 고급 기술 매니저
  3M Health Information Systems,

"AWS CloudFormation 템플릿과 AWS CodePipeline을 사용하여 새로운 프로덕션 계정을 구축하는 시간을 이틀에서 몇 분으로 단축했습니다. 그래서 설정하는 데 비용과 시간이 매우 적게 드는 작은 규모의 프로젝트를 시작할 수 있게 되었습니다. 이러한 민첩성을 발판으로 저희는 리소스가 감당할 수 있는 솔루션에 안주하기보다는 실험을 통해 최상의 솔루션을 구현할 수 있습니다. AWS 덕분에 새로운 기능과 애플리케이션을 이전보다 훨씬 빠르게 출시할 수 있습니다."[2]

- Sam Keen, lululemon Athletica
  제품 아키텍처 담당 이사



[1] *https://aws.amazon.com/solutions/case-studies/3M-health-information-systems/*

[2] *https://aws.amazon.com/solutions/case-studies/lululemon-athletica/*

더욱 발전된 DevOps 관행을 실행하는 팀

77%
사고 발생 후 하루 안에 서비스 복구

60%
보안 취약성 하루 안에 완전히 해결

3배
변화 관리의 효과

aws modern apps

아직 많은 고객이 Dev+Ops를 도입하지 않았지만 오류를 적게 내면서 더욱 빠르게 제공하는 데 유용한 솔루션을 찾는 과정에서 점차 Dev+Ops로 전환하고 있습니다. 어떤 고객은 Dev+Ops 여정을 시작했지만 기대했던 수준의 속도와 성공을 달성하는 데는 어려움을 겪습니다. 많은 경우에 고객이 Dev+Ops로의 전환을 너무 어려워하기 때문에 문제가 됩니다.

1. 시작하기가 너무 어렵습니다. 부담스럽게 느낄 수 있습니다. 변화를 위해서는 배워야 할 것도 많고 해야 할 일도 많기 때문입니다.
2. 현존하는 산업 솔루션이 기업의 보안과 규정 준수, 고가용성, 액세스 관리 등에 대한 요구를 충분히 고려하지 못합니다. 그 결과, DevOps 팀은 나중에 교체해야만 하는 솔루션을 구축하는 데 많은 시간을 할애하게 됩니다.
3. 많은 산업 솔루션이 개발 팀과 운영 팀을 쉽게 결합하도록 DevOps 팀에 충분히 도움을 주지 못합니다. 예를 들면 사전 예방적이 아니라 사후 대응적인 솔루션이 많습니다. 고급 경고와 최적의 진행 방법에 대한 제안을 통해 처음부터 문제를 예방하도록 돕는 것이 아니라 무언가 잘못되고 나서야 고객에게 알리는 것입니다.

문화를 바꿔야 한다는 점이 현대적 Dev+Ops의 가장 어려운 부분입니다. 모든 사람이 동일하게 이해하고 동의하게 하려면 리더십과 시간, 헌신이 필요합니다. 과거에는 운영 팀이 관리하던 업무에 개발자가 더 많이 참여하게 되면서 공동 책임이라는 의식이 필요하게 되었습니다. 다음은 문화적, 행동적 차원의 업무 방식으로, 이 방식을 도입하면 높은 성과를 내는 Dev+Ops 조직을 만들 메커니즘을 확보할 수 있습니다.

이 eBook에서는 AWS가 파악한 높은 성과를 내는 Dev+Ops 조직의 기본 원칙을 안내합니다.

- 책임 이행
- 스스로 측정
- 점진적 개선
- 전체 자동화
- 최대한 코드화
- 중복 백업 시스템 구축
- 도구 표준화

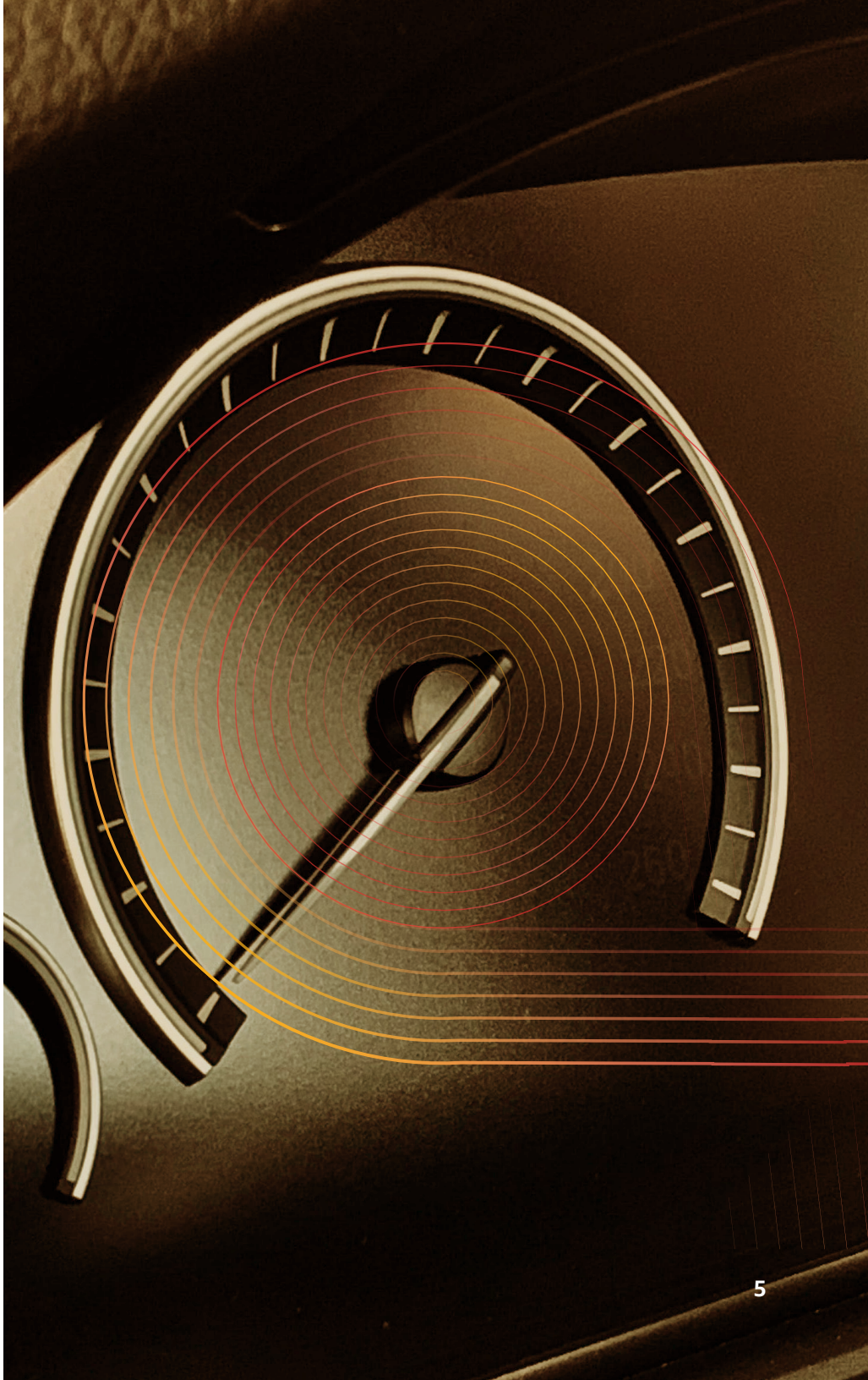
책임 이행

Dev+Ops의 목표는 소프트웨어 제공 속도일 수도 있지만 애초에 Dev+Ops의 의도는 개발 팀과 운영 팀을 밀접하게 연결하여 문제 해결에 협력하도록 하는 것이었습니다. 시간이 지남에 따라 조직 구조는 다양해졌지만 일반적인 솔루션은 책임의 문화에 집중하는 것입니다. 팀과 개인별로 특수 분야와 전문성이 있겠지만 문제를 해결하는 데는 공동으로 책임을 져야 합니다.

스스로 측정

Dev+Ops의 궁극적 목표는 업무를 빠르게 진행하는 것입니다. 하지만 어디에서 시작해야 할지 모르는 경우가 많습니다. 먼저 스스로 측정하는 데서 시작해 보세요. 즉, 현재의 릴리스 시간을 측정해 보는 것입니다. 소프트웨어 제공 프로세스의 어떤 부분에서 시간이 가장 오래 걸리며 그 이유는 무엇인가요? 개선할 수 있을까요? 성공에 중요한 다른 핵심 성과 지표는 무엇인가요? 프로세스에서 개발 팀과 운영 팀 간에 분산된 부분이 있나요? 먼저 이런 지표의 수치와 그 결과를 측정하면 집중해야 할 부분을 알 수 있으며 팀 내에서 신뢰할 수 있는 결과를 보여줄 수 있을 것입니다.

- 릴리스 시간
- 파이프라인 내의 병목 또는 방해 요소
- 릴리스 속도



점진적 개선

며칠, 몇 주 만에 갑자기 높은 성과를 내는 Dev+Ops 조직을 만들 수 있는 사람은 없습니다. 기본 원칙을 먼저 수립하고 작게 시작해서 꾸준하고 점진적으로 개선해 가는 것이 중요합니다. 마찬가지로, 큰 규모의 변화나 릴리스에만 매달려 있으면 소프트웨어 프로세스를 가속화할 수 없습니다. 사람은 크고 복잡한 변화보다 새로운 내용이 얼마 안 되는 작지만 점진적인 변화를 이해하기가 훨씬 쉽습니다.

전체 자동화

Dev+Ops의 목표는 모든 것을 지속적으로 진행하는 것입니다. 자동화를 통해 사람이 개입했을 때 나타나는 시간 지체와 오류를 줄일 수 있습니다. Dev+Ops 워크플로에 자동화를 추가하는 방법은 많지만 가장 기본적인 방법은 테스트, 배포, 롤백이 자동화된 Dev+Ops CI/CD 파이프라인을 추가하는 것입니다.

최대한 코드화

자동화와 Dev+Ops를 위한 또 다른 방법은 인프라와 정책 등을 코드화하여 사람과 기계 모두가 이해하도록 하는 것입니다. 가장 기본적인 사용 사례는 코드형 인프라(IaC)로, 클라우드 인프라를 사용자가 정의한 대로 프로비저닝하도록 자동화하여 사용자가 변경 사항을 추적하고 조직 전체에 모범 사례를 공유할 수 있도록 블루프린트를 제공합니다. 또 다른 예는 코드형 정책으로, 중앙 팀이 자동화가 실행할 규칙을 작성하는 방법입니다.

코드형 인프라 파일에는 다음이 포함됩니다.

- **컴퓨팅**
- **스토리지**
- **자격 증명, 액세스, 보안**
- **애플리케이션 리소스**

중복 백업 시스템 구축

모범 사례를 준수하는 데는 독려하는 방법과 강제하는 방법이 있습니다. 모범 사례 준수를 독려하려면 조직에 모범 사례 템플릿을 공유하여 사용하도록 하면 됩니다. 일반적으로 코드형 인프라에 이 방법이 사용됩니다. 모범 사례 준수를 강요하려면 코드형 정책과 같은 프로세스를 사용할 수 있습니다. 그러나 차세대 중복 백업 시스템은 규칙과 인공 지능을 사용하여 오류가 사용자에게 영향을 미치기 전에 잠재적인 오류를 찾고 Dev+Ops 팀에 모범 사례를 적용하는 방법을 안내합니다.

도구 표준화

표준화된 도구를 갖추면 조직은 확장이 가능합니다. 도구가 표준화되어 있지 않고 여러 종류라면 정책을 작성하고 모범 사례를 적용하기가 매우 어려우며 거의 불가능한 수준입니다. 소수의 공급 업체가 만들어 함께 작동하도록 설계된 도구를 사용한다면 어려움이 덜하겠지만 궁극적으로는 도구가 더 표준화될수록 시간도 절약되고 오류도 줄어듭니다.

AWS를 사용해야 하는 이유

AWS는 현대적 Dev+Ops 관행, 도구, 교육을 위한 핵심 세트로 고객의 역량을 강화하여 클라우드 트랜스포메이션을 빠르게 진행하도록 돕는 독보적인 위치에 있습니다. AWS를 사용하면 고객은 클라우드의 이점을 더욱 빠르게 실현하고 혁신을 가속화할 수 있습니다.

1

Amazon은 Dev+Ops와 혁신의 선구자입니다

Amazon은 현재 일반적으로 사용되는 많은 Dev+Ops 모범 사례를 개척했으며 성과가 높은 고객과 협력하여 고객의 요구에 맞는 솔루션을 제공하는 데 10년이 넘는 경력을 가지고 있습니다. AWS 서비스에는 기본적으로 Amazon 모범 사례가 포함됩니다.

2

AWS는 통합된 현대적 Dev+Ops 서비스의 포트폴리오를 가장 광범위하고 심층적으로 보유합니다

핵심 모니터링과 CI/CD 서비스에서 코드형 인프라 및 사전 예방적 조언과 권장 사항을 제공하기 위해 빅 데이터와 AI를 사용하는 최신 서비스까지 AWS에는 높은 성과를 내는 Dev+Ops 팀 또는 조직을 구축하는 데 필요한 모든 서비스가 있습니다.

3

보안이 모든 업무의 근간이고 그다음은 가용성입니다

AWS Dev+Ops 서비스는 기본적으로 보안 모범 사례를 근간으로 구축됩니다. 예를 들어 CI/CD 파이프라인은 기본적으로 프로젝트마다 격리되어 있습니다. 액세스가 필요한 사람에게만 액세스를 허가하기 쉽게 한 것입니다. 또한 AWS 서비스는 가용성이 높기 때문에 팀이 생산성을 발휘할 수 있습니다.

4

개발자와 조직을 위해 구축되었습니다

조직의 요구를 염두에 두지 않고 구축된 개발자 도구가 많습니다. AWS는 개발자와 조직 모두의 요구에 집중하며, 개발 팀이 빠르고 안전하게 업무를 진행하는 데 도움이 되는 거버넌스와 규정 준수를 쉽게 적용할 수 있게 합니다.

시행계획

그러면 AWS에서 Dev+Ops를 어떻게 시행하면 될까요? 다음 내용은 다양한 업종과 규모의 조직이 앞서 언급한 기본 원칙을 시행하기 위해 사용한 권장 모범 사례입니다.

GitOps

GitOps는 코드형 인프라가 Git 리포지토리에서 호스트되는 접근 방법으로, 애플리케이션 소프트웨어 코드와 동일한 병합 요청 프로세스를 따릅니다. GitOps는 애플리케이션의 인프라에서 대역 외 변경 사항을 제거하도록 설계되었습니다. 코드형 인프라는 주로 템플릿이나 파일을 사용하여 애플리케이션의 인프라를 코드 형식으로 정의하며, 인프라의 프로비저닝을 자동화하여 해당 파일의 정의와 일치하도록 하는 접근법입니다. GitOps를 사용하면 인프라 정의 파일이 Git 리포지토리에 호스트되고 각 커밋이 애플리케이션 소프트웨어 코드와 같은 병합 요청 프로세스를 따르며 지속적 통합 및 지속적 전달(CI/CD) 자동화를 통해 진행됩니다. GitOps의 경우 인프라 파일은 테스트를 거치고 인프라 프로비저닝은 자동화되며 오류가 발생하면 이전 상태로 롤백됩니다.

AWS는 AWS CloudFormation을 중심으로 구축된 완전한 GitOps 솔루션을 제공합니다. AWS CloudFormation을 통해 개발자와 운영 팀은 콘솔에서 일반적인 인프라 언어(YAML 또는 JSON) 또는 AWS Cloud Development Kit(AWS CDK)을 사용하여 코드형 인프라 파일을 손쉽게 정의, 프로비전, 관리할 수 있습니다. AWS CDK를 통해서는 CloudFormation 템플릿을 Python, TypeScript, C#, Java, JavaScript와 같은 일반적인 프로그래밍 언어로 정의할 수 있습니다. CloudFormation은 AWS CodePipeline과 통합되어 있어 CI/CD 워크플로를 자동화할 수 있으며 AWS CodeCommit, GitLab, GitHub와 같이 많이 사용되는 버전 관리 서비스와 통합됩니다. 새로운 인프라 정의 파일이 생성되면 버전이 관리되는 환경에 호스트됩니다. CloudFormation 템플릿에 변경 사항이 있는 경우 CI/CD 자동화가 트리거되며 개발자는 손쉽게 배포하거나 변경 사항을 롤백 또는 롤포워드할 수 있습니다.

"Rivian이 빠른 속도로 성장하고 있기 때문에 고도로 확장 가능한 시스템이 필요합니다. 예전에는 변경하는 데 5일이 걸렸다면 이제는 몇 분이면 완료됩니다."[3]

- SURENDRA BALU Rivian 3DExperience 기술 리드

[3] *https://aws.amazon.com/solutions/case-studies/rivian-case-study/*

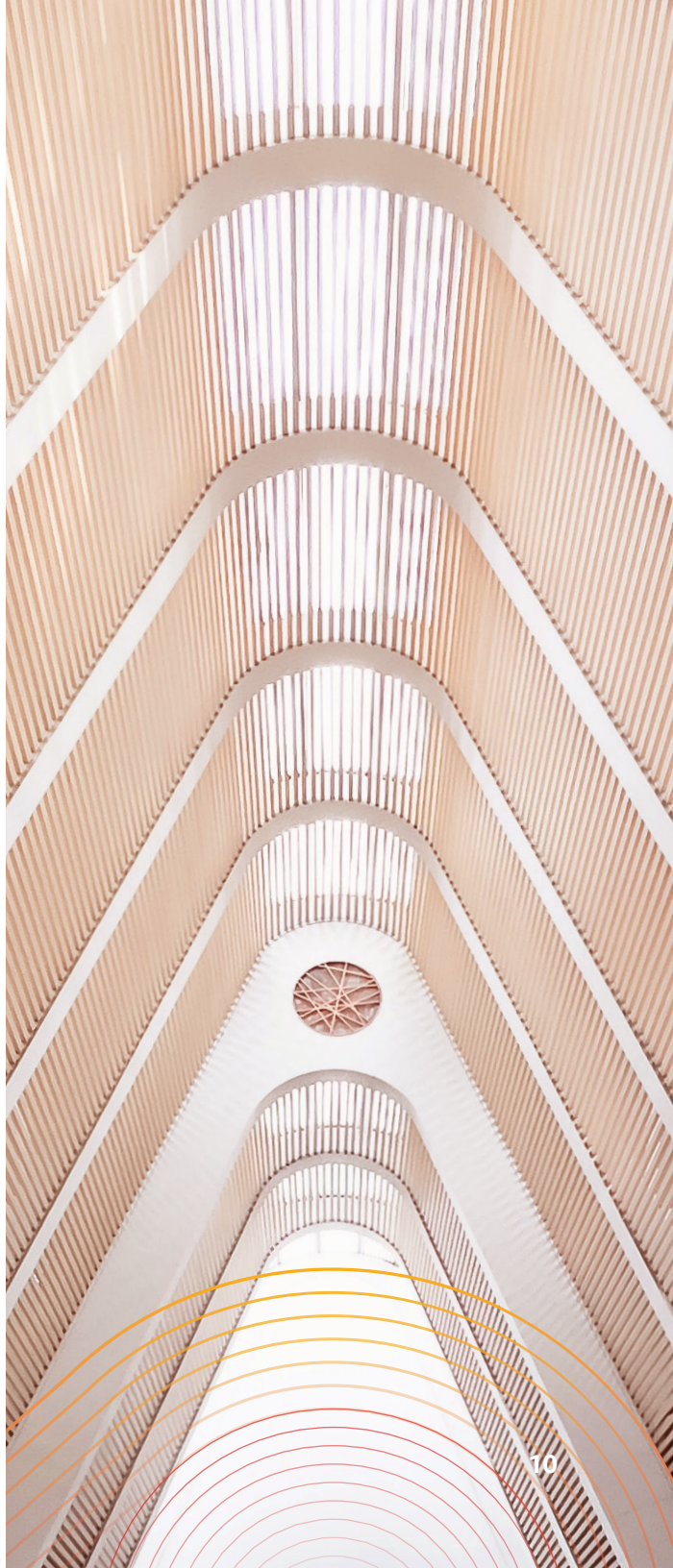
지속적 통합 및 지속적 전달

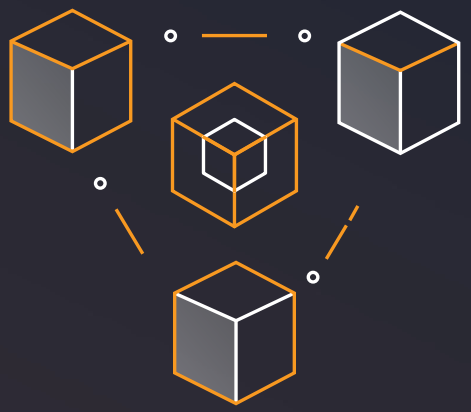
CI/CD는 구축과 테스트, 배포를 아우르는 파이프라인을 사용하여 릴리스를 위한 소프트웨어 준비를 자동화하는 기본적인 모범 사례입니다. CI/CD를 사용하면 오류가 많이 발생하는 수동 프로세스를 제거하고 소프트웨어 릴리스를 살필 필요가 없기 때문에 팀이 업무를 더욱 빠르게 진행하고 오류를 줄일 수 있습니다. 릴리스 시간이 긴 고객의 경우 CI/CD를 통해 시스템을 지속적으로 변경하여 문제를 해결할 수 있습니다. 엔지니어는 코드에서 잠재적인 오류나 위반 사항을 탐지하는 테스트를 쓰고 업데이트를 거부하여 팀에 돌려보냅니다. 버그가 프로덕션에서 나타날 경우 안정적인 버전으로 롤백을 자동화하고 가동 시간을 유지할 수 있습니다.

AWS는 기업 또는 스타트업에 CI/CD를 도입하는 완전관리형 솔루션을 제공합니다. AWS CodeBuild, AWS CodeArtifact, AWS CodePipeline, AWS CodeDeploy는 CI/CD를 도입하는 고객을 돕기 위해 특별히 구축된 연결형 Dev+Ops 서비스입니다. 이 서비스는 Amazon 내부의 CI/CD 도구에서 학습한 내용을 활용하며 기본적으로 모범 사례를 권장합니다. 예를 들어, 하나의 CodePipeline 파이프라인에 있는 각 프로젝트를 격리하면 보안을 중시하는 고객은 해당 프로젝트에 액세스가 필요한 사람에게만 손쉽게 허가를 줄 수 있기 때문에 공유된 빌드 서버의 취약성에 관해 걱정할 필요가 없습니다. 또한, AWS는 CI/CD를 용이하게 하며 전문성이 크게 필요치 않습니다. Amazon CodeGuru Reviewer와 같은 서비스는 개발 과정에서 심각한 문제나 보안 취약성, 찾기 힘든 버그를 알아내는 데 도움을 주는 기계 학습을 통해 코드 검토를 보강하기 때문에 소프트웨어 릴리스 주기의 가져오기 요청 단계에서 도움이 됩니다. 개발자는 안내를 받고 새로운 기능이 프로덕션에 배포되기 전에 문제를 해결함으로써 코드를 개선할 수 있습니다. 또한, AWS Proton은 엔지니어링 팀이 인프라 프로비저닝, 코드 배포, 모니터링, 개발 팀을 위한 업데이트에 필요한 모든 도구를 사전에 구성하는 데 사용할 수 있는 서비스 플랫폼입니다.

"애플리케이션을 구축하는 데 Jenkins를 사용했을 때는 한 시간까지 걸렸던 것이 AWS CodeBuild를 사용하니 이제 10분이면 됩니다. Jenkins에서 그런 성과를 얻으려면 아마 비용이 네 배는 들었을 겁니다. 구축을 그렇게 빠르게 끝내려면 Jenkins 인스턴스를 50개는 돌려야 했을 테니까요."

- Bryan Kane, Coursera 시니어 엔지니어





AIOps

AIOps는 더 자동화되고 더 사전 예방적인 메커니즘으로의 변화이며 이를 통해 안심하고 더 빠르게 혁신할 수 있습니다. 개발자에게 요구되는 지식의 양을 줄이고 Dev+Ops 워크플로에서 AI를 활용함으로써 개발자의 경험을 증강하도록 설계된 AIOps는 문제가 발생하기 전에 유용한 인사이트를 제공하고 팀이 사전 예방적으로 조치하도록 지원하며 기본적으로 모범 사례를 강제하고 궁극적으로 더 신속하게 혁신하도록 합니다. 예를 들어, 개발자는 모범 사례를 벗어난 상황과 동시성 문제, 기타 일반적인 코딩 버그가 시스템에 영향을 미치기 전에 탐지하고 시스템 성능을 향상하는 데 도움이 되는 실질적인 권장 사항을 받을 수 있습니다.

Amazon.com과 AWS 운영 우수성의 20년에 걸친 정보를 포함하고 있는 AWS Machine Learning 모델이 탑재된 기계 학습 기반의 개발자 도구 서비스, Amazon CodeGuru Reviewer는 코드 분석을 용이하게 하여 보안 모범 사례가 모든 곳에서 사용되도록 합니다. Amazon CodeGuru Profiler를 사용하는 고객은 애플리케이션 실행에서 비용이 많이 들고 비효율적인 방법을 파악하고 이를 해결하기 위한 실용적인 단계를 제안받아 결과적으로 비용을 절약할 수 있습니다. 또한, Amazon DevOps Guru를 사용하는 고객은 AIOps 솔루션을 도입하여 애플리케이션의 운영 성능과 가용성을 향상할 수 있습니다. DevOps Guru는 애플리케이션의 비정상적인 동작을 파악하고 잠재적으로 가동 중단이나 서비스 중단을 일으킬 수 있는 심각한 문제를 드러내며 문제를 해결하기 위한 권장 사항을 제공합니다.

"저희는 언제나 팀이 운영 문제를 해결하는 데 할애하는 시간을 절약할 방법을 찾고 있습니다. 이제는 Amazon DevOps Guru와 Amazon DevOps Guru의 기계 학습 기반 인사이트를 활용하여 운영 관련 문제를 빠르게 파악하고 상관 관계를 조사하여 해결합니다. Amazon DevOps Guru의 인사이트를 바탕으로 이제 문제의 근본 원인을 찾기 위해 처음부터 조사하지 않고도 문제를 빠르게 알아낼 수 있습니다. 저희 IT 팀의 평균 고장 수리 시간(MTTR)이 눈에 띄게 줄었으며 이제는 문제 해결에서 정말 많은 시간을 절약합니다. 동시에 고객 또한 최고의 최종 사용자 경험을 하고 있습니다."

- HCL Technologies

지속적 관측성

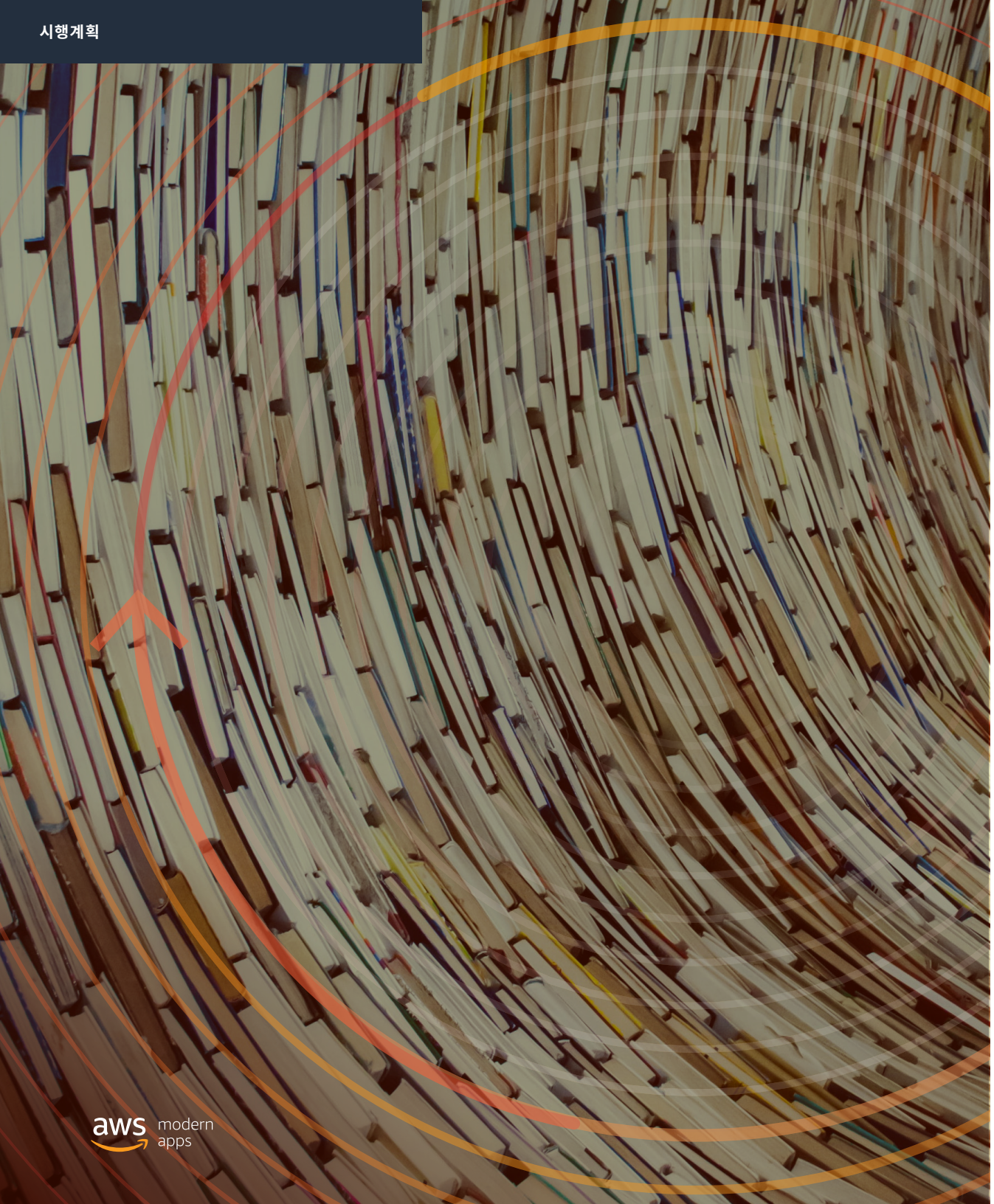
지속적 관측성이란 어느 시점에나 시스템의 상태를 이해할 수 있다는 것을 말합니다. 관측성이 확보되어야 문제를 탐지하고 조사하고 해결할 수 있습니다. 지속적 관측성을 통해 엔지니어링 팀은 애플리케이션의 상태와 성능에 대한 인사이트를 얻어 주목해야 할 문제를 해결함으로써 결과적으로 최종 사용자의 경험을 개선할 수 있습니다. 클라우드 앱과 리소스는 수십억 개의 지표와 로그, 흔적을 생성하여 끊임없는 데이터의 흐름을 만들어 냅니다. 따라서 배포된 앱의 성능을 분석하기가 쉽지 않을 수 있습니다. 관측성을 통해 어느 소스에서나 사용자에 대한 영향을 파악하고 손상되거나 비용이 많이 드는 코드 경로를 빠르게 찾아 개발자의 생산성을 개선하세요.

지속적인 관측성을 확보하는 첫 번째 단계는 Amazon CloudWatch(또는 선호도에 따라 Amazon Managed Service for Prometheus, Amazon Managed Service for Grafana, Amazon Distro for OpenTelemetry 중 하나)를 사용하여 관측성 대시보드를 구축하는 것입니다. 관측성 대시보드를 사용하면 클라우드 서비스에서 어떤 일이 벌어지는지 관리하는 어려움을 덜 수 있습니다. 시계열 지표, 로그, 흔적, 경보 데이터를 표시함으로써 시스템 동작을 간결하게 요약하는 이 대시보드를 통해 사람들은 시스템을 들여다볼 수 있습니다. 대시보드를 구축하고 나면 CloudWatch 경보를 설정하여 클라우드 환경에 잠재적인 문제가 발생했을 때 지속적으로 알람을 받아야 합니다.

"저희는 CloudWatch를 사용해서 Amazon Simple Queue Service와 EC2 Auto Scaling 그룹의 크기 조정 작업에 대한 경보를 만듭니다. 이런 인사이트를 바탕으로 크기 조정에 대한 의사결정을 내려서 고객의 경험에는 영향을 주지 않으면서 AWS 컴퓨팅 리소스를 가장 효율적으로 사용할 수 있습니다. CloudFormation을 사용하면 고객용 애플리케이션에서 사용하는 AWS 리소스의 논리적 스택을 배포할 수 있어서 AWS 계정 전체에 반복 가능하고 일관적으로 배포가 가능합니다. 그 결과 오류가 잦고 비일관적일 수 있는 수동 구성에 의존하지 않고 코드형 구성을 사용하여 애플리케이션 스택을 쉽게 작성할 수 있습니다."

- Martin Costello, Just Eat 시니어 엔지니어

사실:
AWS는 매달 평균 1,000조의 지표 관측을 모니터링합니다.



운영 가시성을 위해 대시보드 구축

Amazon Builders' Library 기사에서 AWS Principal Engineer인 John O'Shea는 Amazon DevOps 팀이 시스템 상태를 파악하기 위해 대시보드를 구축하는 데 대해 어떻게 생각하는지 소개했습니다.

자세히 알아보기 ›

지속적 규정 준수

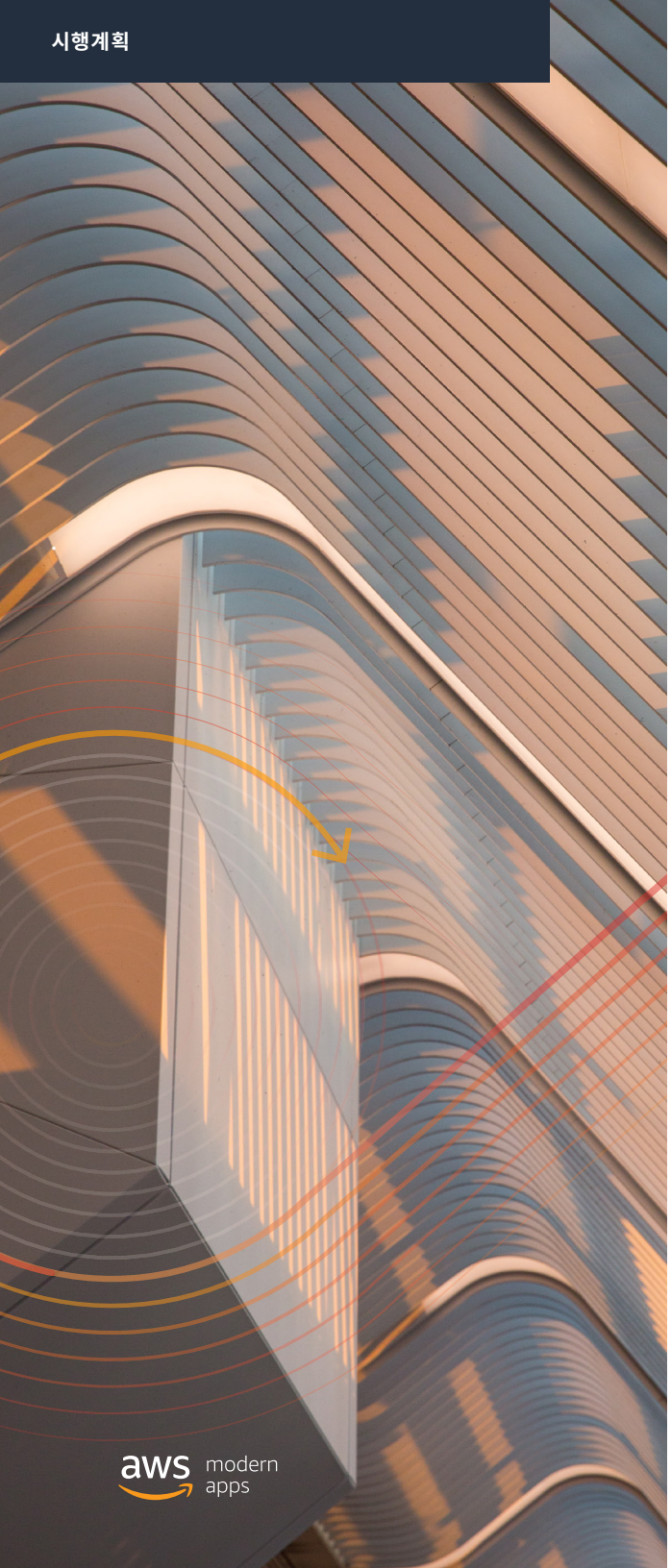
클라우드에서의 규정 준수는 패러다임이 완전히 다릅니다. 클라우드 리소스의 속도와 크기 때문에 전통적인 규정 준수 방법을 활용할 수 없습니다. 스프레드시트와 정적인 데이터베이스로는 리소스 이탈을 처리할 수 없습니다. 성공적으로 규정을 준수하려면 최대한 자동화하고 간소화해야 합니다. AWS는 고객이 클라우드 규정 준수를 간소화할 수 있도록 다양한 서비스를 제공합니다. 규정 준수에 어떤 서비스를 사용할지 파악하려면 내부 감사 관행에 있어서 국제적으로 인정받는 The Institute of Internal Auditors에서 제시하는 Three Lines Model을 적용해 보세요. 이 모델은 조직이 세 개의 방어선을 따라 특정 IT 역할과 책임을 통해 위험을 관리하도록 규정합니다. 첫 번째 방어선을 통해서는 위험을 관리하고, 두 번째로는 위험을 감독하고, 세 번째로는 위험 보증을 제공합니다.

AWS에는 세 개의 방어선 각각에 맞는 서비스가 다양하게 마련되어 있습니다. 예를 들면, 첫 번째 방어선인 위험 관리를 위한 서비스에는 AWS Config, AWS CloudTrail, AWS Systems Manager가 있습니다. 이 서비스를 통해 위험 관리를 위해 개별 관리 권한을 정의하고 배포할 수 있습니다. 두 번째 방어선을 위해서는 AWS Security Hub와 Amazon CloudWatch를 사용하면 전체 조직의 위험을 감독할 수 있습니다. 마지막으로 AWS Audit Manager는 세 번째 방어선인 위험 보증에 적합한 도구로, 감사를 위해 규정 준수의 증거를 수집하는 프로세스를 자동화합니다.

그러나 이러한 도구를 사용한다고 규정 준수가 끝나는 것이 아닙니다. 지속적인 규정 준수를 통해 프로세스를 자동화해야 합니다. 지속적인 규정 준수를 위해서는 코드형 규정 준수라는 개념을 활용하기를 권장합니다. 코드형 규정 준수라는 개념을 도입하면 AWS CloudFormation와 같은 도구를 사용하여 Three Lines Model의 요소를 정의할 수 있습니다. 그런 다음 AWS CodePipeline과 같은 자동화된 파이프라인으로 그 요소를 테스트하고 배포합니다. 새로운 권리 권한과 증거가 필요해져도 각 계정에 수동으로 구현할 필요가 없습니다. 관리 권한 정의에서 코드가 업데이트되고 파이프라인을 통해 푸시되기 때문입니다. 따라서 위험 관리를 위한 새로운 관리 권한이 감사를 위해 즉시 증거로 추가됩니다. 지속적인 규정 준수를 구현하면 조직에서 관리 기능을 마련했고 증거 수집이 조작되지 않았다는 점을 실질적으로 보증할 수 있다는 또 다른 이점이 있습니다. Three Lines Model을 활용하고 지속적인 규정 준수로 프로세스를 자동화하는 것이 규정 준수를 대규모로 처리하는 가장 좋은 방법입니다.

"AWS를 사용하여 업무 방식을 완전히 바꾸었고 이제는 규정 준수 프로세스를 자동화하여 인력을 충원하지 않고도 크기를 조정할 수 있습니다. 규정 준수는 관리하기가 복잡할 수 있지만 AWS Config를 사용하면 규정 준수에 관해 해박하지 않은 저희 같은 사람도 기본 제공되는 템플릿으로 규정 준수를 구현할 수 있습니다. 대시보드는 사용자가 규정 미준수 리소스 등에 관한 정보를 얻어서 보안 태세를 향상하는 데 사용할 수 있고, 조직 전체와 함께 어느 부분에 집중하고 어떤 문제를 해결해야 할지 논의할 때 시작점 역할을 합니다. 디지털 그룹이 클라우드 기술을 통해 내부와 외부 팀에 기술 리더십을 제공할 때 중앙화된 가시성을 확보하고 전체 조직의 보안 태세를 통제하는 것이 중요합니다. 사전 패키징된 Config 규정 준수 팩은 이 프로세스를 간소화했고 저희가 클라우드로 마이그레이션하는 전반적인 속도를 높이는 데 핵심적인 역할을 했습니다."

- Andrew Clark, Baker Tilly Digital
  시니어 솔루션스 아키텍트



aws modern apps

공동의 서비스 플랫폼 구축

Dev+Ops를 구현하는 방법은 팀과 조직마다 조금씩 다릅니다. 많은 조직은 보안과 소프트웨어 제공, 모니터링, 네트워킹을 표준화하여 개발 팀을 지원하고 운영 부담을 줄이는 중앙화된 플랫폼을 갖추고 있습니다. 공동의 서비스 플랫폼은 개발자를 위해 코드 배포를 간소화한 맞춤형 셀프서비스 인터페이스입니다. 배포되는 모든 애플리케이션에 사용해야 하는 보안, 소프트웨어 제공, 모니터링, 네트워킹의 표준을 정의할 권한을 중앙 팀에서 가지고 있습니다. 따라서 개발자는 생산성을 높일 수 있고 플랫폼 팀에 더 많은 권한을 줄 수 있습니다.

AWS는 AWS에서 공동의 서비스 플랫폼을 구축하는 데 필요한 모든 도구를 제공합니다. 기본적인 프로덕션 사용 사례를 위해 AWS Copilot CLI를 사용하면 다중 계정 CI/CD, 보안 그룹, 기본 모니터링을 통해 '배터리 포함' 개발 환경을 쉽게 만들 수 있습니다. 더 복잡한 상황에서는 AWS에서 제공하는 완전관리형 서비스인 AWS Proton을 사용하면 됩니다. AWS Proton은 컨테이너와 서버리스 애플리케이션을 위한 최초의 완전관리형 제공 서비스입니다. 플랫폼 팀은 AWS Proton을 사용하여 인프라 프로비저닝, 코드 배포, 모니터링, 업데이트에 필요한 다양한 도구를 연결하고 함께 사용할 수 있습니다. AWS Proton은 개발자가 컨테이너와 서버리스 기술을 사용하여 쉽게 코드를 배포할 수 있게 하는 동시에 플랫폼 팀에 복잡성을 관리하고 일관적인 표준을 강제하는 데 필요한 도구를 제공하여 이를 해결합니다. 조직에서 공동의 서비스 플랫폼을 구축하는 데 더 많은 관리 기능이 필요하다면 많은 조직에서 AWS CloudFormation과 AWS Service Catalog로 구축하여 AWS에서 사용하도록 승인된 IT 서비스의 카탈로그를 만들고 관리하는 데 성공을 거두었다는 점을 참고하세요. 마찬가지로 플랫폼 팀이 다이내믹 프로그래밍 언어에 익숙한 고객은 AWS CDK를 사용하여 개발 팀을 위한 지원 플랫폼을 구축했습니다.

"저희 기술자들은 AWS Service Catalog의 셀프서비스 포털을 사용하여 티켓을 열지 않고 애플리케이션 빌드를 프로덕션으로 손쉽게 배포할 수 있습니다. 그래서 5분 만에 새로운 애플리케이션 스택을 시작할 수 있습니다. 예전에는 보통 몇 주가 걸렸던 일이죠. Wiley에서는 AWS Service Catalog 덕분에 DevOps와 자동화를 이루고 있습니다."[4]

- Meltem Dincer, Wiley 플랫폼 기능 부문 부회장

[4] *https://aws.amazon.com/solutions/case-studies/wiley/*

조직이 Dev+Ops 여정의 어디에 있든 AWS는 개발 팀과 운영 팀의 간극을 좁히기 위해 설계된 다양한 솔루션을 제공합니다.

AWS 개발자 도구는 시작하기 쉽고, 보안, 규정 준수, 고가용성, 액세스 관리 등 기업의 핵심적인 요구 사항을 충족하며, 고급 경고와 모범 사례에 따른 제안을 통해 문제가 발생하기 전에 파악하고 방지하는 사전 예방적 솔루션을 제공합니다. AWS와 함께라면 고객은 현대적 Dev+Ops 관행과 도구, 교육을 위한 핵심 세트를 통해 클라우드의 이점을 더욱 빠르게 누리고 혁신을 가속화할 수 있습니다.

현대적 DevOps에 관해 자세히 알아보기 ›