AWS Well-Architected 프레임워크

보안 요소

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

보안 요소: AWS Well-Architected 프레임워크

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon의 상표 및 트레이드 드레스는 Amazon 외 제품 또는 서비스와 함께, Amazon 브랜드 이미지를 떨어뜨리거나 고객에게 혼동을 일으킬 수 있는 방식으로 사용할 수 없습니다. Amazon이 소유하지 않은 기타 모든 상표는 Amazon과 제휴 관계이거나 관련이 있거나 후원 관계와 관계없이 해당 소유자의 자산입니다.

Table of Contents

요약 및 소개 1
소개 1
보안 기초 2
설계 원칙 2
정의 2
공동 책임 3
지배구조 5
AWS 계정 관리 및 분리 6
SEC01-BP01 계정을 사용하여 워크로드 분리 7
SEC01-BP02 계정 루트 사용자 및 속성 보호 10
안전한 워크로드 운영 15
SEC01-BP03 제어 목표 파악 및 검증 16
SEC01-BP04 보안 위협 및 권장 사항에 대한 최신 정보 파악 18
SEC01-BP05 보안 관리 범위 축소 20
SEC01-BP06 표준 보안 제어의 배포 자동화 22
SEC01-BP07 위협 모델을 사용하여 위협 식별 및 완화 조치의 우선순위 지정 25
SEC01-BP08 새로운 보안 서비스 및 기능을 정기적으로 평가 및 구현 28
자격 증명 및 액세스 관리 31
자격 증명 관리 31
SEC02-BP01 강력한 로그인 메커니즘 사용 32
SEC02-BP02 임시 자격 증명 사용 35
SEC02-BP03 안전하게 보안 암호 저장 및 사용 39
SEC02-BP04 중앙 집중식 ID 공급업체 사용 44
SEC02-BP05 정기적으로 자격 증명 감사 및 교체 48
SEC02-BP06 사용자 그룹 및 속성 사용 50
권한 관리 53
SEC03-BP01 액세스 요구 사항 정의 54
SEC03-BP02 최소 권한 액세스 부여 57
SEC03-BP03 긴급 액세스 프로세스 설정 61
SEC03-BP04 지속적으로 권한 축소 68
SEC03-BP05 조직에 대한 권한 가드레일 정의 70
SEC03-BP06 수명 주기에 따라 액세스 관리 73
SEC03-BP07 퍼블릭 및 크로스 계정 액세스 분석 75
SEC03-BP08 안전하게 조직과 리소스 공유 78

SEC03-BP09 안전하게 서드파티와 리소스 공유 81
탐지 86
SEC04-BP01 서비스 및 애플리케이션 로깅 구성 87
구현 지침 8
리소스 9
SEC04-BP02 표준화된 위치에서 로그, 조사 결과 및 지표 캡처 91
구현 지침 8
구현 단계 17
리소스 9
SEC04-BP03 보안 알림 보강 및 상관관계 지정 94
구현 지침 8
리소스 9
SEC04-BP04 규정 미준수 리소스 관련 문제 해결 시작 97
구현 지침 8
리소스 9
인프라 보호 101
네트워크 보호 102
SEC05-BP01 네트워크 계층 생성 103
SEC05-BP02 네트워크 계층 내 트래픽 흐름 제어 105
SEC05-BP03 검사 기반 보호 구현 108
SEC05-BP04 네트워크 보호 자동화 111
컴퓨팅 보호 113
SEC06-BP01 취약성 관리 수행 114
SEC06-BP02 강화된 이미지로부터 컴퓨팅 프로비저닝 116
SEC06-BP03 수동 관리 및 대화형 액세스 감소 119
SEC06-BP04 소프트웨어 무결성 검증 122
SEC06-BP05 컴퓨팅 보호 자동화 124
데이터 보호 127
데이터 분류 127
SEC07-BP01 데이터 분류 체계 이해 127
SEC07-BP02 데이터 민감도에 따라 데이터 보호 제어 적용 130
SEC07-BP03 식별 및 분류 자동화 132
SEC07-BP04 확장 가능한 데이터 수명 주기 관리 정의 135
저장 데이터 보호 137
SEC08-BP01 보안 키 관리 구현 138
SEC08-BP02 저장 시 암호화 적용 141

SEC08-BP03 저장 데이터 보호 자동화 144
SEC08-BP04 액세스 제어 적용 147
전송 중 데이터 보호 150
SEC09-BP01 보안 키 및 인증서 관리 구현 151
SEC09-BP02 전송 중 암호화 적용 154
SEC09-BP03 네트워크 통신 인증 156
인시던트 대응 161
AWS 인시던트 대응 161
클라우드 응답의 설계 목표 162
준비 163
SEC10-BP01 주요 직원과 외부 리소스 파악 164
SEC10-BP02 인시던트 관리 계획 개발 167
SEC10-BP03 포렌식 역량 확보 170
SEC10-BP04 보안 인시던트 대응 플레이북 개발 및 테스트 173
SEC10-BP05 액세스 권한 사전 프로비저닝 175
SEC10-BP06 도구 사전 배포 178
SEC10-BP07 시뮬레이션 실행 181
운영 183
인시던트 사후 활동 183
SEC10-BP08 인시던트로부터 학습하기 위한 프레임워크 구축 184
애플리케이션 보안 187
SEC11-BP01 애플리케이션 보안 교육 188
구현 지침 8
리소스 9
SEC11-BP02 개발 및 릴리스 수명 주기를 통한 테스트 자동화 191
..... 191
..... 191
구현 지침 8
리소스 9
SEC11-BP03 정기적인 침투 테스트 시행 194
구현 지침 8
리소스 9
SEC11-BP04 코드 검토 수행 196
구현 지침 8
리소스 9
SEC11-BP05 패키지 및 종속성 서비스의 중앙 집중화 199

구현 지침 8
리소스 9
SEC11-BP06 프로그래밍 방식으로 소프트웨어 배포 201
구현 지침 8
리소스 9
SEC11-BP07 정기적으로 파이프라인의 보안 속성 평가 205
구현 지침 8
리소스 9
SEC11-BP08 보안 소유권이 워크로드 팀에 귀속되도록 프로그램 구축 207
구현 지침 8
리소스 9
결론 209
기여자 210
참조 자료 212
문서 수정 213
고지 사항 216
AWS 용어집 217

보안 원칙 - AWS Well-Architected Framework

발행일: 2024년 11월 6일(문서 수정)

이 백서에서는 AWS Well-Architected Framework의 보안 원칙을 중점적으로 다룹니다. 안전한 AWS 워크로드 설계, 제공 및 유지 관리에 모범 사례 및 현재 권장 사항을 적용할 때 참조할 수 있는 지침을 제공합니다.

소개

AWS Well-Architected Framework를 활용하면 AWS에서 워크로드를 하면서 내리게 되는 결정의 장단점을 이해할 수 있습니다. 이 프레임워크를 사용하면 클라우드에서 신뢰할 수 있고 안전하며 효율적이고 경제적이면서 지속 가능한 워크로드를 설계하고 운영하기 위한 최신 아키텍처 모범 사례를 알아볼 수 있습니다. 또한 모범 사례와 비교하여 워크로드를 지속적으로 평가하고 개선할 영역을 파악할 수 있습니다. 워크로드를 제대로 설계하면 비즈니스 성공 가능성이 높아집니다.

이 프레임워크는 다음 6가지 원칙을 기반으로 합니다.

- 운영 우수성
- 보안
- 신뢰성
- 성능 효율성
- 비용 최적화
- 지속 가능성

이 백서에서는 보안 원칙을 중점적으로 다룹니다. 최신 AWS 권장 사항에 따라 비즈니스 및 규제 요구 사항을 충족하는 데 도움이 됩니다. 이 문서는 최고 기술 책임자(CTO), 최고 정보 보안 책임자(CSO/CISO), 아키텍트, 개발자와 같은 기술 업무 담당자와 운영 팀원을 위해 작성되었습니다.

이 백서의 내용을 확인하고 나면 보안을 염두에 두고 클라우드 아키텍처를 설계할 때 사용할 AWS의 현재 권장 사항과 전략을 파악할 수 있습니다. 구현 세부 정보나 아키텍처 패턴은 제공되지 않지만, 이 정보를 확인할 수 있는 적절한 리소스에 대한 참조는 포함되어 있습니다. 이 백서에서 설명하는 사례를 도입하면 데이터와 시스템을 보호하고, 액세스를 제어하며, 보안 이벤트에 자동으로 대응하는 아키텍처를 구축할 수 있습니다.

보안 기초

보안 원칙은 보안 태세를 개선할 수 있는 방식으로 클라우드 기술을 활용하여 데이터, 시스템 및 자산을 보호하는 방법을 설명합니다. 이 백서에서는 AWS에서 보안 워크로드를 설계하기 위한 심층 모범 사례 지침을 제공합니다.

설계 원칙

클라우드에는 워크로드 보안을 강화할 수 있는 여러 가지 원칙이 존재합니다.

- 강력한 자격 증명 기반 구현: 최소 권한 원칙을 구현하고 AWS 리소스와의 각 상호 작용에 대한 적절한 권한을 부여하여 업무를 분리합니다. 자격 증명 관리를 중앙 집중화하고 장기적인 정적 자격 증명에 대한 의존도를 해소하는 것을 목표로 합니다.
- 추적 기능 유지 관리: 실시간으로 환경에 대한 작업 및 변경 사항을 모니터링하고 알림을 전송하며 감사합니다. 로그 및 지표 수집을 시스템과 통합하여 자동으로 조사하고 조치를 취합니다.
- 모든 계층에 보안 적용: 여러 보안 제어와 함께 심층 방어 접근 방식을 적용합니다. 모든 계층(예: 네트워크 엣지, VPC, 로드 밸런싱, 모든 인스턴스 및 컴퓨팅 서비스, 운영 체제, 애플리케이션, 코드)에 적용됩니다.
- 보안 모범 사례의 자동 적용: 자동화된 소프트웨어 기반의 보안 메커니즘은 안전한 규모 조정 능력을 빠르고 비용 효율적으로 향상시킵니다. 버전 제어가 가능한 템플릿에서 코드로 정의되고 관리되는 제어 기능의 구현을 비롯한 보안 아키텍처를 생성합니다.
- 전송 중 데이터 및 보관 중인 데이터 보호: 데이터를 민감도 수준에 따라 분류하고 적절한 경우 암호화, 토큰화 및 액세스 제어와 같은 메커니즘을 사용합니다.
- 사람들이 데이터에 쉽게 접근할 수 없도록 유지: 데이터에 대한 직접 액세스 또는 수동 처리의 필요성을 줄이거나 없애기 위한 메커니즘 및 도구를 사용합니다. 이를 통해 민감한 데이터를 처리할 때 잘못된 취급이나 수정 및 수작업으로 인한 오류의 위험을 줄일 수 있습니다.
- 보안 이벤트에 대비: 조직의 요구 사항에 부합하는 인시던트 관리 및 조사 정책과 프로세스를 통해 사고에 대비합니다. 인시던트 대응 시뮬레이션을 실행하고 자동화된 도구를 사용하여 감지, 조사 및 복구 속도를 높입니다.

정의

클라우드의 보안은 다음의 7가지 영역으로 구성됩니다.

- 보안 기초

- 자격 증명 및 액세스 관리
- 탐지
- 인프라 보호
- 데이터 보호
- 인시던트 대응
- 애플리케이션 보안

공동 책임

보안과 규정 준수는 AWS와 고객의 공동 책임입니다. 이 공동 모델은 고객의 운영 부담을 더는 데 도움이 될 수 있습니다. 호스트 운영 체제 및 가상화 계층부터 서비스 운영 시설의 물리적 보안에 이르는 구성 요소를 AWS에서 운영, 관리 및 제어하기 때문입니다. 고객의 책임 및 관리 범위에는 AWS가 제공하는 보안 그룹 방화벽의 구성과 게스트 운영 체제(업데이트 및 보안 패치 포함) 및 기타 관련 애플리케이션 소프트웨어가 포함됩니다. 사용하는 서비스, 서비스를 IT 환경에 통합하는 과정 및 준거법과 규제에 따라 책임 범위가 다르기 때문에 고객은 선택하고자 하는 서비스에 대해 신중해야 합니다. 또한 이러한 공동 책임은 본질적으로 유연성을 제공하며 배포를 허용하는 제어 권한을 고객에게 부여합니다. 다음 차트에서 볼 수 있듯이 이 책임의 차이를 일반적으로 클라우드 '자체'의 보안과 클라우드 '내부'의 보안이라고 합니다.

AWS의 '클라우드 자체 보안' 책임 - AWS는 AWS 클라우드에서 제공되는 모든 서비스를 실행하는 인프라를 보호할 책임이 있습니다. 이 인프라는 AWS 클라우드 서비스를 실행하는 하드웨어, 소프트웨어, 네트워킹 및 시설로 구성됩니다.

고객의 '클라우드 내부 보안' 책임 - 고객의 책임은 고객이 선택하는 AWS 클라우드 서비스에 따라 결정됩니다. 서비스에 따라 고객이 보안 책임의 일환으로서 수행해야 하는 구성 작업의 양이 달라집니다. 예를 들어, Amazon Elastic Compute Cloud(Amazon EC2)와 같은 서비스는 서비스형 인프라(IaaS)로 분류되므로 고객이 필수 보안 구성 및 관리 작업을 모두 수행해야 합니다. Amazon EC2 인스턴스를 배포하는 고객의 경우 게스트 운영 체제 관리(업데이트 및 보안 패치 포함), 고객이 인스턴스에 설치하는 모든 애플리케이션 소프트웨어 또는 유틸리티, 각 인스턴스에 AWS에서 제공하는 방화벽 구성(보안 그룹이라고 함)에 대한 책임이 있습니다. Amazon S3 및 Amazon DynamoDB와 같은 추상화된 서비스의 경우 AWS는 인프라 계층, 운영 체제, 플랫폼을 작동하고, 고객은 엔드포인트에 액세스하여 데이터를 저장 및 검색합니다. 고객은 데이터를 관리하고(암호화 옵션 포함), 자산을 분류하며, IAM 도구를 사용하여 적절한 권한을 적용할 책임이 있습니다.

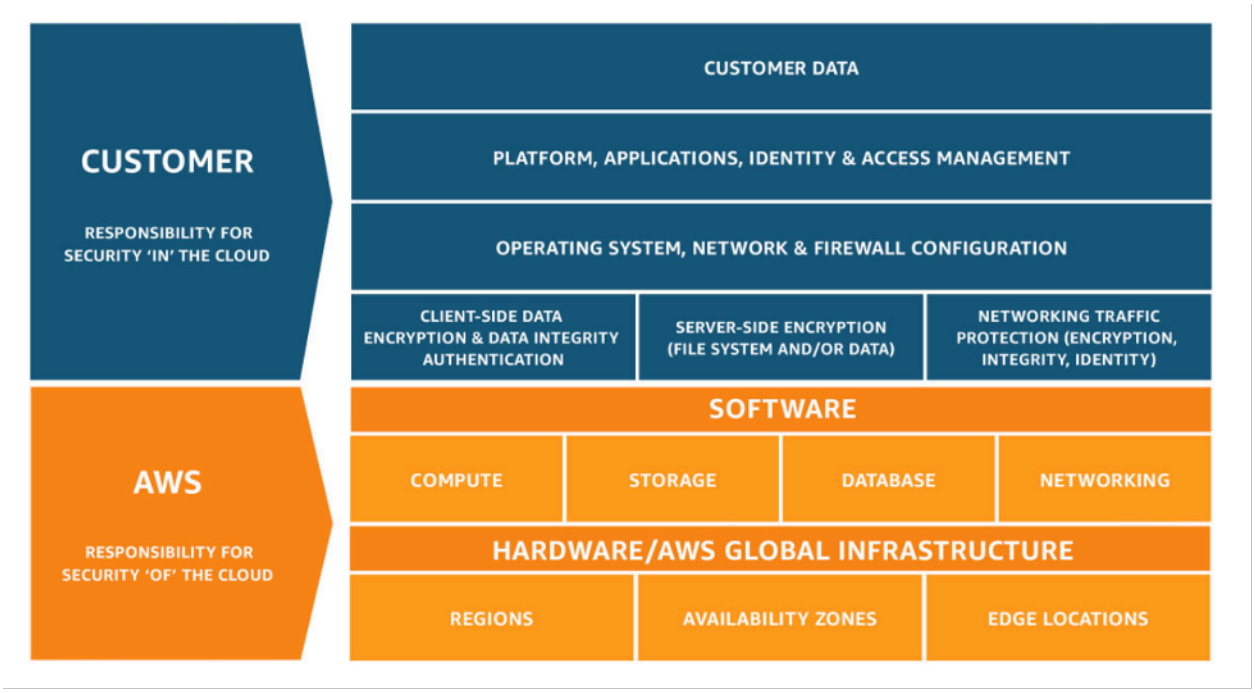


그림 1: AWS 공동 책임 모델.

이 고객/AWS 공동 책임 모델은 IT 제어로도 확대 적용됩니다. IT 환경을 운영하는 책임을 AWS와 고객이 공유하는 것과 마찬가지로, IT 제어의 관리, 운영, 확인 또한 공유됩니다. AWS는 이전에는 고객이 관리했던 AWS 환경에 배포된 물리적 인프라와 관련한 제어를 관리함으로써 운영 제어의 고객 부담을 완화하는 데 도움을 줄 수 있습니다. 고객마다 AWS에서 배포된 방법이 다르므로 고객은 특정 IT 제어 관리를 AWS로 전환하여 새로운 분산 제어 환경을 이용할 수 있습니다. 그런 다음 고객은 제공된 AWS 제어 및 규정 준수 설명서를 참조하여 필요한 제어 평가 및 검증 절차를 실시할 수 있습니다. 다음은 AWS, AWS 고객 또는 이 모두가 관리하는 제어의 예제입니다.

상속된 제어 - 고객이 AWS로부터 완전히 상속하는 제어입니다.

- 물리적 및 환경 제어

공유 제어 - 별개의 컨텍스트 또는 관점에서 인프라 계층 및 고객 계층 모두에 적용되는 제어입니다. 공동 제어에서 AWS는 인프라에 대한 요구 사항을 제공하고 고객은 AWS 서비스 사용과 관련한 자체적인 제어 구현을 제공해야 합니다. 그러한 예는 다음과 같습니다.

- 패치 관리 - AWS는 인프라 내의 패치 작업과 결함 수정에 대한 책임이 있지만, 고객은 게스트 운영 체제와 애플리케이션 패치 작업에 책임이 있습니다.
- 구성 관리 - AWS는 자체 인프라 디바이스의 구성을 유지 관리하지만, 고객은 자체 게스트 운영 체제, 데이터베이스, 애플리케이션 구성에 책임이 있습니다.

- 인식 및 교육 - AWS는 AWS 직원을 교육하며 고객은 자체 직원을 교육시켜야 합니다.

고객별 - AWS 서비스 내에서 배포하는 애플리케이션에 따라 고객이 전적으로 책임을 져야 하는 제어입니다. 그러한 예는 다음과 같습니다.

- 고객이 특정 보안 환경 내에서 데이터를 라우팅하거나 배치해야 하는 서비스 및 통신 보호 또는 영역 보안입니다.

지배구조

전반적인 접근 방식의 일부로 보안 거버넌스를 포함하는 것은 위험 관리에 도움이 되는 정책과 제어 목표를 정의하여 비즈니스 목표 달성을 지원하기 위한 것입니다. 보안 제어 목표에 계층화된 접근 방식을 사용하여 위험을 관리합니다. 각 계층은 이전 계층을 기반으로 구현합니다. AWS 공동 책임 모델을 이해하는 것이 가장 기본적인 계층입니다. 이 모델을 이해하면 고객 측에서 어떤 부분에 책임이 있는지, AWS로부터 어떤 책임을 상속하는지 명확히 알 수 있습니다. 이때 AWS 아티팩트가 유용한 리소스입니다. 이를 사용하면 AWS의 보안 및 규정 준수 보고서에 온디맨드로 액세스하고 온라인 계약을 선택할 수 있습니다.

다음 계층에서 대부분의 제어 목표를 달성하세요. 플랫폼 전반의 기능이 이 계층에 있습니다. 예를 들어, 이 계층에는 AWS 계정 벤딩 프로세스, AWS IAM Identity Center과 같은 자격 증명 제공업체와의 통합, 일반적인 탐지 제어가 포함됩니다. 플랫폼 거버넌스 프로세스의 결과 중 일부도 이 계층에 있습니다. 새로운 AWS 서비스를 사용하고자 할 때 AWS Organizations 서비스에서 서비스 제어 정책(SCP)을 업데이트하여 서비스를 처음 사용할 때 적용할 가드레일을 제공할 수 있습니다. 다른 SCP를 사용하여 일반적인 보안 제어 목표(보안 불변수라고 함)를 구현할 수 있습니다. 보안 불변수는 여러 개의 계정, 조직 단위 또는 전체 AWS 조직에 적용하는 제어 목표 또는 구성입니다. 일반적으로 인프라가 실행되는 리전의 수를 제한하거나 탐지 제어 비활성화를 차단하는 것을 예로 들 수 있습니다. 이 중간 계층에는 구성 규칙 또는 파이프라인 검사 등 코드화된 정책도 포함됩니다.

상단 계층은 제품 팀이 제어 목표를 충족하는 영역입니다. 제품 팀이 제어하는 애플리케이션에서 구현이 수행되기 때문입니다. 애플리케이션에 입력 검증을 구현하거나 마이크로서비스 간에 자격 증명이 정확히 전달되도록 하는 것을 예로 들 수 있습니다. 제품 팀에서 구성을 담당하지만, 여전히 중간 계층으로부터 일부 기능을 상속할 수 있습니다.

제어를 어디에 구현하든 목표는 같습니다. 바로 위험을 관리하는 것입니다. 위험 관리 프레임워크의 범위는 특정 업계, 리전 또는 기술에 적용됩니다. 기본 목표는 발생 가능성과 결과에 따라 위험을 강조하는 것입니다. 이를 내재된 위험이라고 합니다. 그런 다음 발생 가능성, 결과 또는 둘 다를 축소하는 제어 목표를 정의할 수 있습니다. 그러고 나서 제어 조치가 마련된 상태에서 위험의 결과가 무엇일지 볼 수

있습니다. 이를 잔존 위험이라고 합니다. 제어 목표는 하나의 워크로드나 여러 개의 워크로드에 적용할 수 있습니다. 다음 다이어그램은 일반적인 위험 매트릭스를 보여줍니다. 발생 가능성은 이전 발생 빈도를 기반으로 하며 결과는 이벤트로 인한 재정, 평판, 시간 비용을 기반으로 합니다.

Likelihood	Risk Level				
Very Likely	Low	Medium	High	Critical	Critical
Likely	Low	Medium	Medium	High	Critical
Possible	Low	Low	Medium	Medium	High
Unlikely	Low	Low	Medium	Medium	High
Very unlikely	Low	Low		Medium	High
Consequence	Minimal	Low	Medium	High	Severe

그림 2: 위험 수준 발생 가능성 매트릭스

AWS 계정 관리 및 분리

조직의 보고 구조를 미러링하는 대신 기능, 규정 준수 요구 사항 또는 공통 제어 요소를 기반으로 별도의 계정과 그룹 계정에 워크로드를 구성하는 것이 좋습니다. AWS에서 계정은 하드 경계를 가지고 있습니다. 예를 들어 프로덕션 워크로드를 개발 및 테스트 워크로드에서 격리하려면 계정 수준의 분리를 적극 권장합니다.

중앙에서 계정 관리: AWS Organizations는 AWS 계정 생성 및 관리 그리고 생성된 계정에 대한 제어를 자동화합니다. AWS Organizations를 통해 계정을 생성할 때는 사용할 이메일 주소를 신중히 선택해야 합니다. 이 이메일 주소가 암호를 재설정할 수 있는 루트 사용자가 되기 때문입니다. Organizations를 사용하면 여러 계정을 조직 단위(OU)로 그룹화하여 워크로드의 요구 사항과 용도에 따라 서로 다른 환경을 나타낼 수 있습니다.

중앙에서 제어 설정: 적절한 수준의 특정 서비스, 리전 및 서비스 작업만 허용하여 AWS 계정에서 수행할 수 있는 작업을 제어하세요. AWS Organizations에서는 서비스 제어 정책(SCP)을 사용하여 모든

AWS Identity and Access Management(IAM) 사용자 및 역할에 적용되는 조직, 조직 단위 또는 계정 수준에서 권한 가드레일을 적용할 수 있습니다. 예를 들어 사용자가 명시적으로 허용하지 않은 리전에서는 리소스를 시작하지 못하도록 제한하는 SCP를 적용할 수 있습니다. AWS Control Tower는 여러 계정을 간편하게 설정하고 관리하는 방법을 제공합니다. 이는 AWS Organizations에서 계정 설정 및 프로비저닝을 자동화하고 가드레일(예방 및 탐지 포함)을 적용하며 대시보드를 제공하여 가시성을 확보합니다.

중앙에서 서비스 및 리소스 구성: AWS Organizations를 사용하면 모든 계정에 적용되는 AWS 서비스를 구성할 수 있습니다. 예를 들어 AWS CloudTrail을 사용하면 조직 전체에서 수행되는 모든 작업을 중앙에서 로깅하도록 구성하고 구성원 계정이 로깅을 비활성화하지 않도록 할 수 있습니다. 또한 AWS Config을 사용하여 정의한 규칙에 대해 중앙에서 데이터를 집계할 수 있으므로 워크로드를 감사하여 규정 준수 여부를 확인하고, 변화에 신속하게 반응하도록 지원할 수 있습니다. AWS CloudFormation StackSets를 사용하면 조직 내의 여러 계정 및 OU에 걸쳐 AWS CloudFormation 스택을 중앙에서 관리할 수 있습니다. 이렇게 하면 보안 요구 사항에 부합하도록 새 계정을 자동으로 프로비저닝할 수 있습니다.

보안 서비스의 위임된 관리 기능을 사용하여 조직의 결제(관리) 계정에서 관리에 사용되는 계정을 분리합니다. GuardDuty, Security Hub, AWS Config 등의 여러 AWS 서비스에서는 관리 기능을 위해 특정 계정을 지정하는 등 AWS Organizations와의 통합을 지원합니다.

모범 사례

- SEC01-BP01 계정을 사용하여 워크로드 분리
- SEC01-BP02 계정 루트 사용자 및 속성 보호

SEC01-BP01 계정을 사용하여 워크로드 분리

다중 계정 전략을 통해 환경(예: 프로덕션, 개발 및 테스트)과 워크로드 간에 일반적인 가드레일 및 격리를 설정합니다. 계정 수준의 분리는 보안, 청구 및 액세스에 대한 강력한 격리 경계를 제공하므로 강력하게 권장됩니다.

원하는 성과: 클라우드 운영, 관련 없는 워크로드 및 환경을 별도의 계정으로 격리하여 클라우드 인프라 전반의 보안을 강화하는 계정 구조.

일반적인 안티 패턴:

- 데이터 민감도 수준이 서로 다른 관련 없는 여러 워크로드를 동일한 계정에 배치합니다.
- 잘못 정의된 조직 단위(OU) 구조입니다.

이 모범 사례 확립의 이점:

- 워크로드가 의도치 않게 액세스되는 경우 영향 범위가 감소합니다.
- AWS 서비스, 리소스 및 리전에 대한 액세스 권한의 중앙 거버넌스.
- 보안 서비스의 정책 및 중앙 집중식 관리를 통해 클라우드 인프라의 보안을 유지 관리합니다.
- 자동화된 계정 생성 및 유지 관리 프로세스.
- 규정 준수 및 규제 요구 사항에 대한 인프라의 중앙 집중식 감사.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 지침

AWS 계정은 서로 다른 민감도 수준에서 작동하는 워크로드 또는 리소스 간에 보안 격리 경계를 제공합니다. AWS는 다중 계정 전략을 통해 대규모로 클라우드 워크로드를 관리할 수 있는 도구를 제공하여 이 격리 경계를 활용할 수 있습니다. AWS에 대한 다중 계정 전략의 개념, 패턴 및 구현에 대한 지침은Organizing Your AWS Environment Using Multiple Accounts를 참조하세요.

중앙 관리 하에 여러 AWS 계정이 있는 경우 조직 단위(OU) 계층으로 정의된 계층 구조로 계정을 구성해야 합니다. 그런 다음 보안 제어를 구성하고 OU 및 구성원 계정에 적용하여 조직의 구성원 계정에 일관된 예방적인 제어를 설정할 수 있습니다. 보안 제어는 상속되므로 OU 계층 구조의 하위 수준에 있는 구성원 계정이 사용 가능한 권한을 필터링할 수 있습니다. 우수한 설계는 이 상속을 활용하여 각 구성원 계정에 대해 원하는 보안 제어를 달성하는 데 필요한 보안 정책의 수와 복잡성을 줄입니다.

AWS Organizations 및 AWS Control Tower는 AWS 환경에서 이 다중 계정 구조를 구현하고 관리하는 데 사용할 수 있는 두 가지 서비스입니다. AWS Organizations를 사용하면 하나 이상의 OU 계층으로 정의된 계층 구조로 계정을 구성할 수 있습니다. 각 OU에는 여러 구성원 계정이 포함되어 있습니다. 서비스 제어 정책(SCP)을 사용하면 조직 관리자가 구성원 계정에 대한 세분화된 예방 제어를 설정할 수 있으며, AWS Config를 사용하면 구성원 계정에 대한 사전 예방 및 탐지 제어를 설정할 수 있습니다. 많은 AWS 서비스가 AWS Organizations와 통합되어 위임된 관리 제어를 제공하고 조직의 모든 구성원 계정 전체의 서비스별 작업을 수행합니다.

AWS Organizations 위에 계층화된 AWS Control Tower에서는 랜딩 존이 있는 다중 계정 AWS 환경에 대한 원클릭 모범 사례 설정을 제공합니다. 랜딩 존은 Control Tower가 구축한 다중 계정 환경의 진입점입니다. Control Tower는 AWS Organizations에 비해 몇 가지 이점을 제공합니다. 개선된 계정 거버넌스를 제공하는 세 가지 이점은 다음과 같습니다.

- 조직에 참여하도록 승인된 계정에 자동으로 적용되는 통합 필수 보안 제어.

- 주어진 OU 세트에 대해 활성화 또는 비활성화할 수 있는 선택적 제어.
- AWS Control Tower Account Factory는 조직 내에서 미리 승인된 기준과 구성 옵션을 포함하는 계정의 자동화된 배포를 제공합니다.

구현 단계

1. 조직 단위 구조 설계: 적절하게 설계된 조직 단위 구조는 서비스 제어 정책 및 기타 보안 제어를 생성하고 유지 관리하는 데 필요한 관리 부담을 줄여줍니다. 조직 단위 구조는 비즈니스 요구 사항, 데이터 민감도 및 워크로드 구조에 맞춰 조정해야 합니다.
2. 다중 계정 환경을 위한 랜딩 존 생성: 랜딩 존은 조직이 워크로드를 신속하게 개발, 실행 및 배포할 수 있는 일관된 보안 및 인프라 기반을 제공합니다. 맞춤형으로 구축된 랜딩 존 또는 AWS Control Tower를 사용하여 환경을 오케스트레이션할 수 있습니다.
3. 가드레일 설정: 랜딩 존을 통해 환경에 일관된 보안 가드레일을 구현합니다. AWS Control Towerhttps://docs.aws.amazon.com/controltower/latest/userguide/mandatory-controls.html는 배포할 수 있는 필수 및 선택적 제어 목록을 제공합니다. 필수 제어는 Control Tower를 구현할 때 자동으로 배포됩니다. 적극 권장되는 선택적 제어 목록을 검토하고 요구 사항에 적합한 제어를 구현합니다.
4. 새로 추가된 리전에 대한 액세스 권한 제한: 새 AWS 리전의 경우 사용자 및 역할과 같은 IAM 리소스는 사용자가 지정하는 리전으로만 전파됩니다. 이 작업은 Control Tower를 사용할 때 콘솔을 통해 수행하거나 AWS Organizations에서 IAM 권한 정책을 조정하여 수행할 수 있습니다.
5. AWS CloudFormation StackSets 고려: StackSets를 사용하면 IAM 정책, 역할, 그룹을 포함한 리소스를 승인된 템플릿에서 다양한 AWS 계정 및 리전에 배포할 수 있습니다.

리소스

관련 모범 사례:

- SEC02-BP04 중앙 집중식 ID 공급업체 사용

관련 문서:

- AWS Control Tower
- AWS 보안 감사 지침
- IAM 모범 사례
- Use CloudFormation StackSets to provision resources across multiple AWS 계정 and regions

- Organizations FAQ
- AWS Organizations 용어 및 개념
- Best Practices for Service Control Policies in an AWS Organizations Multi-Account Environment
- AWS Account Management 참조 안내서
- Organizing Your AWS Environment Using Multiple Accounts

관련 비디오:

- Enable AWS adoption at scale with automation and governance
- Security Best Practices the Well-Architected Way
- Building and Governing Multiple Accounts using AWS Control Tower
- Enable Control Tower for Existing Organizations

SEC01-BP02 계정 루트 사용자 및 속성 보호

루트 사용자는 계정 내의 모든 리소스에 대한 전체 관리 액세스 권한이 있는 AWS 계정에서 가장 권한이 높은 사용자이며 경우에 따라 보안 정책의 제약을 받을 수 없습니다. 루트 사용자에 대한 프로그래밍 방식 액세스 권한을 비활성화하고, 루트 사용자에 대한 적절한 제어를 설정하며, 루트 사용자의 일상적인 사용을 방지하면 루트 자격 증명이 의도치 않게 노출되어 클라우드 환경이 손상될 위험을 줄일 수 있습니다.

원하는 성과: 루트 사용자를 보호하면 루트 사용자 자격 증명 남용으로 인해 우발적이거나 의도적인 손상이 발생할 가능성을 줄일 수 있습니다. 탐지 제어를 설정하면 루트 사용자를 사용하여 작업을 수행할 때 적절한 담당자에게 알릴 수도 있습니다.

일반적인 안티 패턴:

- 루트 사용자 자격 증명이 필요한 몇 가지 작업 이외의 작업에 루트 사용자를 사용합니다.
- 긴급 상황에서의 중요한 인프라, 프로세스 및 인력 기능을 확인하기 위한 비상 계획을 정기적으로 테스트하는 데 소홀합니다.
- 일반적인 계정 로그인 흐름만 고려하고 대체 계정 복구 방법을 고려하거나 테스트하는 데 소홀합니다.
- DNS, 이메일 서버 및 전화 공급자가 계정 복구 흐름에서 사용되므로 중요한 보안 경계의 일부로 처리하지 않습니다.

이 모범 사례 확립의 이점: 루트 사용자에 대한 액세스를 보호하면 계정의 작업이 제어되고 감사된다는 확신을 얻을 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 지침

AWS는 계정을 보호하는 데 도움이 되는 다양한 도구를 제공합니다. 그러나 이러한 조치는 대부분 기본적으로 활성화되어 있지 않으므로 이를 구현하기 위해서는 직접적인 조치를 취해야 합니다. AWS 계정 보호를 위한 기본 단계로 다음 권장 사항을 고려해 보세요. 이러한 단계를 구현할 때 보안 제어를 지속적으로 평가하고 모니터링하는 프로세스를 구축하는 것이 중요합니다.

AWS 계정을 처음 생성할 때 계정의 모든 AWS 서비스 및 리소스에 대한 완전한 액세스 권한이 있는 단일 자격 증명을 생성하는 것으로 시작합니다. 이 자격 증명을 AWS 계정 루트 사용자라고 합니다. 계정을 생성할 때 사용한 이메일 주소와 암호를 입력하여 루트 사용자로 로그인할 수 있습니다. AWS 루트 사용자에게 부여된 높은 액세스 권한으로 인해 특별히 이를 필요로 하는 작업을 수행하려면 AWS 루트 사용자의 사용을 제한해야 합니다. 루트 사용자 로그인 자격 증명은 철저하게 보호되어야 하며 AWS 계정 루트 사용자에 대해 항상 다중 인증(MFA)을 활성화해야 합니다.

사용자 이름, 암호 및 다중 인증(MFA) 디바이스를 사용하여 루트 사용자에 로그인하는 일반적인 인증 흐름 외에도 이메일 주소 및 계정과 연결된 전화번호에 대한 액세스 권한이 부여된 AWS 계정 루트 사용자에 로그인하는 계정 복구 흐름이 있습니다. 따라서 복구 이메일이 전송되는 루트 사용자 이메일 계정을 보호하는 것과 계정과 연결된 전화번호를 보호하는 것이 동일하게 중요합니다. 또한 루트 사용자와 연결된 이메일 주소가 동일한 AWS 계정의 이메일 서버 또는 도메인 이름 서비스(DNS) 리소스에서 호스팅되는 잠재적인 순환 종속성도 고려하세요.

AWS Organizations를 사용하는 경우 각각 루트 사용자가 있는 여러 AWS 계정이 있습니다. 하나의 계정이 관리 계정으로 지정되면 여러 계층의 구성원 계정을 관리 계정 아래 추가할 수 있습니다. 관리 계정의 루트 사용자 보호에 우선순위를 지정한 다음 구성원 계정 루트 사용자의 주소를 기재합니다. 관리 계정의 루트 사용자를 보호하기 위한 전략은 구성원 계정 루트 사용자와 다를 수 있으며, 구성원 계정 루트 사용자에게 예방 보안 제어를 적용할 수 있습니다.

구현 단계

루트 사용자에 대한 제어를 설정하려면 다음 구현 단계를 수행하는 것이 좋습니다. 해당하는 경우 권장 사항은 CIS AWS Foundations benchmark version 1.4.0에서 교차 참조됩니다. 이러한 단계 외에도 AWS 계정 및 리소스 보호를 위해 AWS 모범 사례 지침을 참조하세요.

예방 제어

1. 계정의 정확한 연락처 정보를 설정합니다.
    a. 이 정보는 분실한 암호 복구 흐름, 분실한 MFA 디바이스 계정 복구 흐름 및 팀과의 중요한 보안 관련 커뮤니케이션에 사용됩니다.
    b. 회사 도메인에서 호스팅하는 이메일 주소(배포 목록 선호)를 루트 사용자의 이메일 주소로 사용합니다. 개인의 이메일 계정이 아닌 배포 목록을 사용하면 장기간에 걸쳐 루트 계정에 액세스할 수 있는 추가 중복성과 연속성이 제공됩니다.
    c. 연락처 정보에 표시된 전화번호는 이 목적을 위한 보안 전용 전화여야 합니다. 전화번호를 표시하거나 다름 사람과 공유해서는 안 됩니다.
2. 루트 사용자의 액세스 키를 생성하지 않습니다. 액세스 키가 있으면 제거합니다(CIS 1.4).
    a. 루트 사용자에 대한 장기 프로그래밍 방식 자격 증명(액세스 및 보안 암호 키)을 제거합니다.
    b. 루트 사용자 액세스 키가 이미 있는 경우 해당 키를 사용하여 AWS Identity and Access Management(IAM) 역할에서 임시 액세스 키를 사용하도록 프로세스를 전환한 다음, 루트 사용자 액세스 키를 삭제해야 합니다.
3. 루트 사용자의 자격 증명을 저장해야 하는지 여부를 결정합니다.
    a. AWS Organizations를 사용하여 새 구성원 계정을 생성하는 경우 새 구성원 계정에 대한 루트 사용자의 초기 암호가 사용자에게 노출되지 않는 임의의 값으로 설정됩니다. 필요한 경우 AWS 조직 관리 계정의 암호 재설정 흐름을 사용하여 구성원 계정에 대한 액세스 권한을 얻는 것이 좋습니다.
    b. 독립 실행형 AWS 계정 또는 관리 AWS 조직 계정의 경우 루트 사용자에 대한 자격 증명을 생성하고 안전하게 저장하는 것이 좋습니다. 루트 사용자에 대해 MFA를 사용합니다.
4. AWS 다중 계정 환경에서 구성원 계정 루트 사용자에 대한 예방 제어를 활성화합니다.
    a. 구성원 계정의 경우 루트 사용자에 대한 루트 액세스 키 생성 금지 예방 가드레일 사용을 고려하세요.
    b. 구성원 계정의 경우 루트 사용자로 작업 금지 예방 가드레일 사용을 고려하세요.
5. 루트 사용자에 대한 자격 증명이 필요한 경우:
    a. 복잡한 암호를 사용합니다.
    b. 루트 사용자, 특히 AWS Organizations 관리(지급인) 계정에 대해 다중 인증(MFA)을 활성화합니다(CIS 1.5).
    c. 단일 사용 디바이스는 MFA 코드가 포함된 디바이스가 다른 용도로 재사용될 가능성을 줄일 수 있으므로 복원력 및 보안을 위해 하드웨어 MFA 디바이스를 고려합니다. 배터리로 구동되는 하드웨어 MFA 디바이스가 정기적으로 대체되는지 확인합니다. (CIS 1.6)
        - 루트 사용자용 MFA를 구성하려면 가상 MFA 또는 하드웨어 MFA 디바이스를 생성하는 방법에 대한 지침을 따르세요.

d. 백업을 위해 여러 MFA 디바이스를 등록하는 것이 좋습니다. 계정당 최대 8개의 MFA 디바이스가 허용됩니다.
   - 루트 사용자에 대해 둘 이상의 MFA 디바이스를 등록하면 MFA 디바이스가 손실된 경우 계정을 복구하는 흐름이 자동으로 비활성화됩니다.

e. 암호를 안전하게 저장하고, 암호를 전자적으로 저장하는 경우 순환 종속성을 고려합니다. 암호를 획득하는 데 동일한 AWS 계정에 대한 액세스 권한이 필요한 방식으로 암호를 저장하지 않습니다.

6. 선택 사항: 루트 사용자에 대한 주기적인 암호 교체 일정을 설정하는 것이 좋습니다.
   - 자격 증명 관리 모범 사례는 규정 및 정책 요구 사항에 따라 다릅니다. MFA로 보호되는 루트 사용자는 단일 인증 요소로 암호에 의존하지 않습니다.
   - 주기적으로 루트 사용자 암호를 변경하면 의도치 않게 노출된 암호가 남용될 위험이 줄어듭니다.

탐지 제어

- 루트 자격 증명의 사용을 감지하는 경보를 생성합니다(CIS 1.7). Amazon GuardDuty에서는 RootCredentialUsage 조사 결과를 통해 루트 사용자 API 자격 증명 사용을 모니터링하고 알림을 제공할 수 있습니다.
- AWS Well-Architected Security Pillar conformance pack for AWS Config에 포함된 탐지 제어를 평가 및 구현합니다. 또는 AWS Control Tower를 사용하는 경우 Control Tower 내에서 사용 가능한 제어 기능이 강력히 권장됩니다.

운영 지침

- 조직에서 루트 사용자 자격 증명에 대한 액세스 권한을 갖는 담당자를 결정합니다.
  - 루트 사용자 액세스 권한을 획득하는 데 필요한 모든 자격 증명 및 MFA에 대한 액세스 권한을 한 명의 개인이 갖지 않도록 2인 규칙을 사용합니다.
  - 한 명의 개인이 아닌 조직에서 계정과 연결된 전화번호 및 이메일 별칭(암호 재설정 및 MFA 재설정 흐름에 사용됨)에 대한 제어 권한을 유지 관리하는지 확인합니다.
- 루트 사용자는 예외적으로만 사용합니다(CIS 1.7).
  - AWS 루트 사용자는 관리 작업이라 하더라도 일상 작업에는 사용하면 안 됩니다. 루트 사용자가 필요한 AWS 작업을 수행하려면 루트 사용자로만 로그인합니다. 다른 모든 작업은 적절한 역할이 있는 다른 사용자가 수행해야 합니다.
- 루트 사용자 자격 증명을 사용해야 하는 긴급 상황이 발생하기 전에 절차를 테스트할 수 있도록 루트 사용자에 대한 액세스 권한이 작동하는지 주기적으로 확인합니다.

- 계정과 연결된 이메일 주소와 대체 연락처에 나열된 이메일 주소가 작동하는지 주기적으로 확인합니다. <abuse@amazon.com>에서 수신된 보안 알림이 있는지 이러한 이메일 받은 편지함을 모니터링합니다. 또한 계정과 연결된 모든 전화번호가 작동하는지 확인합니다.
- 루트 계정 남용에 대응하기 위한 인시던트 대응 절차를 준비합니다. AWS 계정에서 인시던트 대응 전략 구축에 대한 자세한 내용은 AWS Security Incident Response Guide 및 보안 원칙 백서의 인시던트 대응 섹션에 나온 모범 사례를 참조하세요.

리소스

관련 모범 사례:

- SEC01-BP01 계정을 사용하여 워크로드 분리
- SEC02-BP01 강력한 로그인 메커니즘 사용
- SEC03-BP02 최소 권한 액세스 부여
- SEC03-BP03 긴급 액세스 프로세스 설정
- SEC10-BP05 액세스 권한 사전 프로비저닝

관련 문서:

- AWS Control Tower
- AWS 보안 감사 지침
- IAM 모범 사례
- Amazon GuardDuty – root credential usage alert
- Step-by-step guidance on monitoring for root credential use through CloudTrail
- AWS에서 사용하도록 승인된 MFA 토큰
- AWS에서 break glass access 구현
- Top 10 security items to improve in your AWS 계정
- AWS 계정에서 승인되지 않은 활동이 감지되면 어떻게 해야 하나요?

관련 비디오:

- Enable AWS adoption at scale with automation and governance
- Security Best Practices the Well-Architected Way

- Limiting use of AWS root credentials from AWS re:inforce 2022 – Security best practices with AWS IAM

안전한 워크로드 운영

안전한 워크로드 운영에는 설계에서 구축, 실행, 지속적 개선까지 워크로드의 전체 수명 주기가 포함됩니다. 클라우드에서 안전하게 운영하는 역량을 향상하는 한 가지 방법은 거버넌스에 조직적 접근 방식을 취하는 것입니다. 거버넌스는 관여된 사람의 선의의 판단에 전적으로 의존하지 않고 일관되게 의사 결정을 이끄는 방식입니다. 거버넌스 모델과 프로세스는 '특정 워크로드에 대한 제어 목표가 충족되며 제어 목표가 해당 워크로드에 적합한지 어떻게 알 수 있는가?'라는 질문에 대답을 제시합니다. 의사 결정에 일관된 접근 방식을 적용하면 워크로드 배포의 속도가 빨라지고 조직 내 보안 역량의 기준을 높이는 데 도움이 됩니다.

워크로드를 안전하게 운영하려면 모든 보안 영역에 중요한 모범 사례를 적용해야 합니다. 운영 우수성에 대해 조직 및 워크로드 수준에서 정의한 요구 사항 및 프로세스를 모든 영역에 적용하세요. AWS, 업계 권장 사항 및 위협 인텔리전스를 최신 상태로 유지하면 위협 모델 및 제어 목표를 발전시키는 데 도움이 됩니다. 보안 프로세스, 테스트 및 검증을 자동화하면 보안 작업 규모를 조정하는 데 도움이 됩니다.

자동화를 통해 프로세스의 일관성과 반복 가능성을 확보할 수 있습니다. 사람은 많은 일에서 뛰어난 역량을 보입니다. 하지만 그러한 사람이라도 동일한 업무를 반복해서 수행할 때 전혀 실수를 하지 않으리라 보장할 수는 없습니다. 런북이 잘 작성되었더라도 사람들이 반복적인 작업을 일관적으로 수행하지 않을 것이라는 위험이 있습니다. 사람들의 책임이 서로 다르며 익숙하지 않은 알림에 대응해야 할 때는 그 위험이 특히 더 큽니다. 하지만 자동화는 매번 같은 방식으로 반응합니다. 애플리케이션을 배포하는 가장 좋은 방법은 자동화를 사용하는 것입니다. 배포를 실행하는 코드를 테스트한 후 배포를 수행하는 데 사용할 수 있습니다. 그러면 변경 프로세스에 대한 신뢰도가 높아지고 실패한 변경에 대한 위험이 낮아집니다.

구성이 제어 목표에 부합하는지 확인하기 위해 자동화와 배포된 애플리케이션을 비프로덕션 환경에서 먼저 테스트합니다. 그러면 자동화가 모든 단계를 올바르게 수행했는지 입증할 수 있습니다. 또한 개발 및 배포 주기의 초기에 피드백을 얻어 재작업을 줄일 수 있습니다. 배포 오류의 가능성을 줄이려면 사람이 아닌 코드를 사용하여 구성을 변경합니다. 애플리케이션을 다시 배포해야 한다면 자동화를 사용하는 경우 배포가 훨씬 쉽습니다. 추가 제어 목표를 정의할 때 모든 워크로드에 적용되도록 자동화에 쉽게 추가하면 됩니다.

개별 워크로드 담당자가 워크로드별 보안을 위해 노력하게 하는 대신 공동의 기능과 공유 구성 요소를 사용하여 시간을 절약하세요. 여러 팀이 사용할 수 있는 서비스의 예로는 AWS 계정 생성 프로세스, 인력을 위한 중앙 집중화된 자격 증명, 일반적인 로깅 구성, AMI 및 컨테이너 기반 이미지 생성이 있습니

다. 이 접근 방식을 사용하면 빌더가 워크로드 주기 시간을 개선하고 일관적으로 보안 제어 목표를 달성하는 데 도움이 됩니다. 팀의 일관성이 높아지면 제어 목표를 검증하고 이해관계자에게 제어 태세와 위험 포지션을 더 효과적으로 보고할 수 있습니다.

모범 사례

- SEC01-BP03 제어 목표 파악 및 검증
- SEC01-BP04 보안 위협 및 권장 사항에 대한 최신 정보 파악
- SEC01-BP05 보안 관리 범위 축소
- SEC01-BP06 표준 보안 제어의 배포 자동화
- SEC01-BP07 위협 모델을 사용하여 위협 식별 및 완화 조치의 우선순위 지정
- SEC01-BP08 새로운 보안 서비스 및 기능을 정기적으로 평가 및 구현

SEC01-BP03 제어 목표 파악 및 검증

위협 모델에서 식별된 규정 준수 요구 사항 및 위험을 기준으로, 워크로드에 적용해야 하는 제어 목표와 제어 항목을 도출하고 검증합니다. 제어 목표 및 제어에 대한 지속적인 검증은 위험 완화의 효과를 측정하는 데 도움이 됩니다.

원하는 성과: 비즈니스의 보안 제어 목표가 잘 정의되고 규정 준수 요구 사항에 맞게 조정됩니다. 제어는 자동화와 정책을 통해 구현 및 시행되며 목표 달성의 효율성 측면이 지속적으로 평가됩니다. 특정 시점과 일정 기간의 효과 증명을 감사자에게 쉽게 보고할 수 있습니다.

일반적인 안티 패턴:

- 비즈니스와 관련하여 확실한 보안에 대한 규제 요구 사항, 시장 기대치 및 업계 표준을 제대로 이해하고 있지 않습니다.
- 사이버 보안 프레임워크와 제어 목표가 비즈니스 요구 사항에 맞지 않습니다.
- 제어 구현이 측정 가능한 방식으로 제어 목표와 밀접하게 일치하지 않습니다.
- 자동화를 통해 제어의 효과를 보고하지 않습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 지침

보안 제어 목표의 기초를 형성할 수 있는 일반적인 사이버 보안 프레임워크가 많이 있습니다. 비즈니스에 대한 규제 요구 사항, 시장 기대치, 업계 표준을 고려하여 요구 사항을 가장 잘 지원하는 프레임워크

를 결정하세요. 예를 들어, AICPA SOC 2, HITRUST, PCI-DSS, ISO 27001, NIST SP 800-53이 포함됩니다.

제어 목표를 파악한 경우, 사용하는 AWS 서비스가 해당 목표를 달성하는 데 어떤 도움이 되는지 이해해야 합니다. AWS Artifact를 사용하여 AWS에서 포괄하는 책임 범위를 설명하는 대상 프레임워크에 맞게 조정된 문서와 보고서 그리고 사용자 책임에 속하는 나머지 범위에 대한 지침을 찾아보세요. 다양한 프레임워크 제어 설명에 부합하는 추가적인 서비스별 지침은 AWS Customer Compliance Guides를 참조하세요.

목표 달성을 위한 제어를 정의할 때 예방적 제어를 바탕으로 실행을 체계화하고 감지 제어를 사용하여 완화를 자동화하세요. 서비스 제어 정책(SCP)을 사용하여 AWS Organizations에서 리소스 구성 및 작업이 규정을 준수하지 못하는 일이 없도록 방지할 수 있습니다. 규정 미준수 리소스를 모니터링하고 보고한 다음, 해당 동작이 확실해지면 규칙을 실행 모델로 전환하도록 AWS Config에서 규칙을 구현합니다. 사이버 보안 프레임워크에 맞게 미리 정의된 관리형 규칙 세트를 배포하려면 첫 번째 옵션으로 AWS Security Hub 표준 사용을 평가합니다. AWS Foundational Security Best Practices(FSBP) 표준과 CIS AWS Foundations Benchmark는 여러 표준 프레임워크에서 공유되는 많은 목표에 부합하는 제어 기능을 갖추고 있어 좋은 출발점이 될 수 있습니다. Security Hub에 기본적으로 원하는 제어 감지 기능이 없는 경우 AWS Config 규정 준수 팩을 사용하여 보완할 수 있습니다.

AWS 글로벌 보안 및 규정 준수 가속화(GSCA) 팀이 권장하는 APN 파트너 번들을 사용하여 보안 고문, 컨설팅 기관, 증거 수집 및 보고 시스템, 감사자 및 필요한 경우 기타 보완 서비스의 도움을 받을 수 있습니다.

구현 단계

1. 일반적인 사이버 보안 프레임워크를 평가하고 선택한 목표에 맞게 제어 목표를 조정합니다.
2. AWS Artifact를 사용하는 프레임워크에 대한 지침 및 책임 관련 문서를 얻습니다. 공동 책임 모델에서 AWS가 부담하는 규정 준수 부분과 사용자의 책임인 부분을 이해합니다.
3. SCP, 리소스 정책, 역할 신뢰 정책 및 기타 가드레일을 사용하여 리소스 구성 및 작업이 규정을 준수하지 못하는 일이 없도록 합니다.
4. 제어 목표에 맞는 Security Hub 표준 및 AWS Config 규정 준수 팩 배포를 평가합니다.

리소스

관련 모범 사례:

- SEC03-BP01 액세스 요구 사항 정의
- SEC04-BP01 서비스 및 애플리케이션 로깅 구성

- SEC07-BP01 데이터 분류 체계 이해
- OPS01-BP03 거버넌스 요구 사항 평가
- OPS01-BP04 규정 준수 요구 사항 평가
- PERF01-BP05 정책 및 참조 아키텍처 사용
- COST02-BP01 조직 요구 사항에 따라 정책 개발

관련 문서:

- AWS Customer Compliance Guides

관련 도구:

- AWS Artifact

SEC01-BP04 보안 위협 및 권장 사항에 대한 최신 정보 파악

업계 위협 인텔리전스 간행물과 업데이트를 위한 데이터 피드를 모니터링하여 최신 위협 및 방어 조치를 파악하세요. 최신 위협 데이터를 기반으로 자동 업데이트되는 관리형 서비스 제품을 평가합니다.

원하는 성과: 업계 간행물에 최신 위협 및 권장 사항이 업데이트되므로, 지속적으로 정보를 얻을 수 있습니다. 자동화를 사용하여 새로운 위협을 식별할 때 잠재적 취약성과 노출을 탐지합니다. 이러한 위협에 대해 완화 조치를 취합니다. 최신 위협 인텔리전스로 자동 업데이트되는 AWS 서비스를 채택합니다.

일반적인 안티 패턴:

- 최신 위협 인텔리전스에 대한 정보를 지속적으로 파악할 수 있는 신뢰할 수 있고 반복 가능한 메커니즘이 없습니다.
- 잠재적 취약성 및 노출에 대해 인적 검토가 필요한 기술 포트폴리오, 워크로드, 종속성에 대한 인벤토리를 수동으로 유지 관리합니다.
- 워크로드 및 종속성을 알려진 위협 완화 기능을 제공하는 최신 지원 버전으로 업데이트할 수 있는 메커니즘이 없습니다.

이 모범 사례 확립의 이점: 위협 인텔리전스 소스를 사용하여 최신 상태를 유지하면 비즈니스에 영향을 미칠 수 있는 위협 환경의 중요한 변화를 놓칠 위험이 줄어듭니다. 워크로드에 잠재적인 취약성이나

노출이 존재하는 부분과 그 종속성을 스캔, 탐지 및 해결하기 위한 자동화 기능을 마련하면 수동 대안에 비해 위험을 빠르고 예측 가능하게 완화하는 데 도움이 될 수 있습니다. 이를 통해 취약성 완화와 관련된 시간과 비용을 제어할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 지침

신뢰할 수 있는 위협 인텔리전스 발행물을 검토하여 위협 상황을 파악합니다. 알려진 적대적 전술, 기법 및 절차(TTP)에 대한 문서는 MITRE ATT&CK 기술 자료를 참조하세요. MITRE의 Common Vulnerabilities and Exposures 목록을 검토하여 사용 중인 제품의 알려진 취약성에 대한 최신 정보를 확인합니다. 널리 알려진 OWASP(Open Worldwide Application Security Project)의 인기 OWASP Top 10 프로젝트를 통해 웹 애플리케이션에 대한 중대한 위험을 이해합니다.

CVE용 AWS 보안 공지를 참조하여 AWS 보안 이벤트 및 권장되는 수정 단계에 대한 최신 정보를 확인합니다.

최신 상태를 유지하는 데 드는 전반적인 수고를 덜고 오버헤드를 줄이려면 시간이 지남에 따라 새로운 위협 인텔리전스를 자동으로 통합하는 AWS 서비스를 사용하는 것이 좋습니다. 예를 들어, Amazon GuardDuty에서는 계정 내에서 비정상적인 동작과 위협 서명을 탐지하기 위한 업계 위협 인텔리전스를 최신 상태로 유지할 수 있습니다. Amazon Inspector에서는 연속 스캔 기능에 사용하는 CVE의 데이터베이스를 자동으로 최신 상태로 유지합니다. AWS WAF 및 AWS Shield Advanced 모두 새로운 위협이 발생하면 자동으로 업데이트되는 관리형 규칙 그룹을 제공합니다.

자동화된 플릿 관리 및 패치 적용의 경우 Well-Architected 운영 우수성 원칙을 검토하세요.

구현 단계

- 비즈니스 및 업계와 관련된 위협 인텔리전스 간행물의 최신 소식을 구독합니다. AWS 보안 공지를 구독합니다.
- Amazon GuardDuty 및 Amazon Inspector와 같이 새로운 위협 인텔리전스를 자동으로 통합하는 서비스 채택을 고려해 보세요.
- Well-Architected 운영 우수성 원칙의 모범 사례에 부합하는 플릿 관리 및 패치 적용 전략을 배포합니다.

리소스

관련 모범 사례:

- SEC01-BP07 위협 모델을 사용하여 위협 식별 및 완화 조치의 우선순위 지정
- OPS01-BP05 위협 환경 평가
- OPS11-BP01 지속적인 개선을 위한 프로세스 마련

SEC01-BP05 보안 관리 범위 축소

특정 제어 기능의 관리를 AWS(관리형 서비스)로 전환하는 AWS 서비스를 사용하여 보안 범위를 줄일 수 있는지를 결정합니다. 이러한 서비스는 인프라 프로비저닝, 소프트웨어 설정, 패치 적용, 백업과 같은 보안 유지 관리 작업을 줄이는 데 도움이 될 수 있습니다.

원하는 성과: 워크로드에 맞는 AWS 서비스를 선택할 때 보안 관리 범위를 고려합니다. 관리 오버헤드 및 유지 관리 작업에 드는 비용(총 소유 비용(TCO))은 다른 Well-Architected 고려 사항 외에도 선택한 서비스 비용을 기준으로 산정됩니다. AWS 제어 및 규정 준수 설명서를 제어 평가 및 검증 절차에 통합합니다.

일반적인 안티 패턴:

- 선택한 서비스에 대한 공동 책임 모델을 완전히 이해하지 않고 워크로드를 배포합니다.
- 상응하는 관리형 서비스를 평가하지 않고 가상 머신에서 데이터베이스 및 기타 기술을 호스팅합니다.
- 관리형 서비스 옵션과 비교할 때 보안 관리 작업을 가상 머신의 호스팅 기술에 대한 총 소유 비용에 포함하지 않습니다.

이 모범 사례 확립의 이점: 관리형 서비스를 사용하면 운영 보안 제어를 관리하는 데 따르는 전반적인 부담을 낮추고 보안 위험과 총 소유 비용을 줄일 수 있습니다. 특정 보안 작업에 걸리는 시간을 비즈니스에 더 많은 가치를 제공하는 작업에 재투자할 수 있습니다. 또한, 관리형 서비스는 일부 제어 요구 사항을 AWS로 전환하여 규정 준수 요구 사항의 범위를 줄일 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 지침

AWS 기반 워크로드의 구성 요소를 통합하는 방법에는 여러 가지가 있습니다. Amazon EC2 인스턴스에 기술을 설치하고 실행하려면 사용자가 전체 보안 책임의 가장 큰 부분을 맡아야 하는 경우가 많습니다. 특정 제어 기능을 운영하는 데 따르는 부담을 줄이려면 사용자가 지는 공동 책임 모델의 범위를 줄이는 AWS 관리형 서비스를 식별하고, 기존 아키텍처에서 이를 사용하는 방법을 이해해야 합니다. 예를 들어 데이터베이스 배포에 Amazon Relational Database Service(RDS) 사용, 컨테이너 오

케스트레이션에 Amazon Elastic Kubernetes Service(Amazon EKS) 또는 Amazon Elastic Container Service(Amazon ECS) 사용 또는 서버리스 옵션 사용이 포함됩니다. 새 애플리케이션을 구축할 때는 보안 제어를 구현하고 관리하는 데 필요한 시간과 비용을 줄이는 데 어떤 서비스가 도움이 될 수 있는지 생각해 보세요.

규정 준수 요구 사항도 서비스를 선택할 때 고려할 요인이 될 수 있습니다. 관리형 서비스는 일부 요구 사항의 규정 준수를 AWS로 전환할 수 있습니다. 운영 및 관리하는 서비스의 양상을 감사하고 관련 AWS 감사 보고서의 제어 설명을 수락하는 데 있어 규정 준수 팀이 어느 정도 편안함을 느끼는지 규정 준수 팀과 논의하세요. AWS Artifact에서 발견된 감사 아티팩트를 감사 기관이나 규제 기관에 AWS 보안 통제의 증거로 제공할 수 있습니다. 또한 일부 AWS 감사 아티팩트에서 제공하는 책임 지침을 AWS Customer Compliance Guides와 함께 사용하여 아키텍처를 설계할 수 있습니다. 이 지침은 시스템의 특정 사용 사례를 지원하기 위해 적용해야 하는 추가 보안 제어를 결정하는 데 도움이 됩니다.

관리형 서비스를 사용할 경우 리소스를 최신 버전으로 업데이트하는 프로세스를 숙지해야 합니다(예: Amazon RDS에서 관리하는 데이터베이스 버전 또는 AWS Lambda 함수의 프로그래밍 언어 런타임 업데이트). 관리형 서비스가 이 작업을 대신 수행할 수도 있지만, 업데이트 시기를 구성하고 운영에 미치는 영향을 이해하는 것은 사용자의 책임입니다. AWS Health와 같은 도구를 사용하면 환경 전체에서 이러한 업데이트를 추적하고 관리할 수 있습니다.

구현 단계

1. 관리형 서비스로 대체할 수 있는 워크로드 구성 요소를 평가하세요.
    a. 워크로드를 AWS로 마이그레이션하는 경우 워크로드를 리호스팅, 리팩터링, 리플랫포밍, 재구축 또는 교체해야 하는지 평가할 때 관리에 드는 시간 및 비용 절감과 위험 감소를 고려해야 합니다. 때로는 마이그레이션을 시작할 때 추가적으로 투자하여 장기적으로 상당한 비용을 절감할 수 있습니다.
2. 자체 기술 배포를 설치하고 관리하는 대신 관리형 서비스(예: Amazon RDS)를 구현하는 것을 고려해 보세요.
3. AWS Artifact의 책임 지침을 참조하여 워크로드에 적용해야 하는 보안 제어를 결정하세요.
4. 사용 중인 리소스의 인벤토리를 유지하고 최신 서비스와 접근 방식을 최신 상태로 유지하여 범위를 줄일 수 있는 새로운 기회를 찾아보세요.

리소스

관련 모범 사례:

- PERF02-BP01 워크로드에 가장 적합한 컴퓨팅 옵션 선택

- PERF03-BP01 데이터 액세스 및 스토리지 요구 사항을 가장 잘 지원하는 목적별 데이터 스토어 사용
- SUS05-BP03 관리형 서비스 사용

관련 문서:

- Planned lifecycle events for AWS Health

관련 도구:

- AWS Health
- AWS Artifact
- AWS Customer Compliance Guides

관련 비디오:

- How do I migrate to an Amazon RDS or Aurora MySQL DB instance using AWS DMS?
- AWS re:Invent 2023 - Manage resource lifecycle events at scale with AWS Health

SEC01-BP06 표준 보안 제어의 배포 자동화

AWS 환경 전반에서 표준인 보안 제어 기능을 개발하고 배포할 때 최신 DevOps 사례를 적용하세요. 코드형 인프라(IaC) 템플릿을 사용하여 표준 보안 제어 및 구성을 정의하고, 버전 제어 시스템에서 변경 사항을 캡처하고, CI/CD 파이프라인의 일부로 변경 사항을 테스트하며, AWS 환경에 변경 사항을 자동으로 배포할 수 있습니다.

원하는 성과: IaC 템플릿이 표준화된 보안 제어를 캡처하고 이를 버전 제어 시스템에 적용합니다. CI/CD 파이프라인은 변경 사항을 탐지하고 AWS 환경 테스트 및 배포를 자동화하는 곳에 마련되어 있습니다. 배포를 진행하기 전에 템플릿의 구성 오류를 탐지하고 알림을 보내기 위한 가드레일이 있습니다. 워크로드는 표준 제어 기능이 적용되는 환경에 배포됩니다. 팀은 셀프 서비스 메커니즘을 통해 승인된 서비스 구성을 배포할 수 있습니다. 제어 구성, 스크립트 및 관련 데이터를 위한 안전한 백업 및 복구 전략이 마련되어 있습니다.

일반적인 안티 패턴:

- 웹 콘솔 또는 명령줄 인터페이스를 통해 표준 보안 제어 기능을 수동으로 변경합니다.

- 개별 워크로드 팀에 의존하여 중앙 팀이 정의한 제어 기능을 수동으로 구현합니다.
- 워크로드 팀의 요청에 따라 중앙 보안 팀에 의존하여 워크로드 수준 제어 기능을 배포합니다.
- 동일한 개인 또는 팀이 적절히 업무를 분담하거나 점검하며 균형을 맞추지 않고 보안 제어 자동화 스크립트를 개발, 테스트 및 배포할 수 있습니다.

이 모범 사례 확립의 이점: 템플릿을 사용하여 표준 보안 제어를 정의하면 버전 제어 시스템을 통해 시간 경과에 따른 변경 사항을 추적하고 비교할 수 있습니다. 자동화를 사용하여 변경 사항을 테스트하고 배포하면 표준화와 예측 가능성을 높여 배포가 성공할 확률이 커지고 수동 반복 작업을 줄일 수 있습니다. 워크로드 팀이 승인된 서비스 및 구성을 배포할 수 있는 셀프 서비스 메커니즘을 제공하면 구성 오류 및 오용의 위험을 줄일 수 있습니다. 또한, 개발 프로세스 초기에 제어 기능을 통합하는 데도 도움이 됩니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 지침

SEC01-BP01 계정을 사용하여 워크로드 분리에서 설명한 사례에 따라 AWS Organizations를 사용하여 관리하는 여러 환경에 여러 AWS 계정이 필요합니다. 이러한 각 환경과 워크로드에는 별도의 보안 제어가 필요할 수 있지만, 조직 전체에서 일부 보안 제어를 표준화할 수 있습니다. 예로는 중앙 집중식 ID 제공업체 통합, 네트워크 및 방화벽 정의, 로그 저장 및 분석을 위한 표준 위치 구성 등이 있습니다. 코드형 인프라(IaC)를 사용하여 인프라 프로비저닝에 동일하게 엄격한 애플리케이션 코드 개발을 적용할 수 있는 것과 마찬가지로, IaC를 통해 표준 보안 제어를 정의하고 배포할 수도 있습니다.

가능하면 AWS CloudFormation과 같은 선언적인 방법으로 보안 제어를 정의하고 소스 제어 시스템에 저장합니다. DevOps 관행을 바탕으로 보다 예측 가능한 릴리스를 위해 제어 기능을 자동으로 배포하고, AWS CloudFormation Guard와 같은 도구를 사용하여 자동화된 테스트를 수행하며, 배포된 제어 기능과 원하는 구성 간의 차이를 탐지합니다. AWS CodePipeline, AWS CodeBuild, AWS CodeDeploy와 같은 서비스를 사용하여 CI/CD 파이프라인을 구성할 수 있습니다. Organizing Your AWS Environment Using Multiple Accounts의 지침 참조하여 다른 배포 파이프라인과 분리된 자체 계정에서 이러한 서비스를 구성하세요.

템플릿을 정의하여 AWS 계정, 서비스 및 구성을 정의하고 배포하는 것을 표준화할 수도 있습니다. 이 기술을 사용하면 중앙 보안 팀에서 정의를 관리하고 셀프 서비스 접근 방식을 통해 워크로드 팀에 제공할 수 있습니다. 이를 달성하는 한 가지 방법은 워크로드 팀이 자체 파이프라인 배포에 통합 가능한 제품으로 템플릿을 게시할 수 있는 Service Catalog를 사용하는 것입니다. AWS Control Tower를 사용하는 경우 일부 템플릿과 제어 기능을 시작점으로 삼을 수 있습니다. 또한 Control Tower는 Account Factory 기능을 제공하여 워크로드 팀이 정의한 표준을 통해 새로운 AWS 계정 계정을 만들 수 있도록

합니다. 이 기능을 사용하면 중앙 팀에 종속되지 않고 워크로드 팀에서 필요하다고 판단한 경우 새 계정을 승인하고 생성할 수 있습니다. 이러한 계정은 제공하는 기능, 처리 중인 데이터의 민감도 또는 동작과 같은 이유에 따라 다양한 워크로드 구성 요소를 분리하는 데 필요할 수 있습니다.

구현 단계

1. 버전 관리 시스템에서 템플릿을 저장하고 유지 관리하는 방법을 결정합니다.
2. CI/CD 파이프라인을 생성하여 템플릿을 테스트하고 배포합니다. 잘못된 구성이 있는지 점검하고 템플릿이 기업 표준을 준수하는지 확인하는 테스트를 정의합니다.
3. 워크로드 팀이 요구 사항에 따라 AWS 계정을 배포하고 서비스할 수 있도록 표준화된 템플릿 카탈로그를 구축합니다.
4. 제어 구성, 스크립트 및 관련 데이터에 대한 안전한 백업 및 복구 전략을 구현합니다.

리소스

관련 모범 사례:

- OPS05-BP01 버전 관리 사용
- OPS05-BP04 구축 및 배포 관리 시스템 사용
- REL08-BP05 자동화를 통한 변경 사항 배포
- SUS06-BP01 지속 가능성 개선을 신속하게 도입할 수 있는 방법 채택

관련 문서:

- Organizing Your AWS Environment Using Multiple Accounts

관련 예제:

- Automate account creation, and resource provisioning using Service Catalog, AWS Organizations, and AWS Lambda
- Strengthen the DevOps pipeline and protect data with AWS Secrets Manager, AWS KMS, and AWS Certificate Manager

관련 도구:

- AWS CloudFormation Guard

- Landing Zone Accelerator on AWS

SEC01-BP07 위협 모델을 사용하여 위협 식별 및 완화 조치의 우선순위 지정

위협 모델링을 수행하여 워크로드에 대한 잠재적 위협 및 관련 완화 조치의 최신 등록을 식별하고 유지 관리합니다. 보안 위협 우선순위를 지정하고 보안 제어 완화 조치를 조정하여 방지, 감지 및 대응합니다. 워크로드와 진화하는 보안 환경에 맞춰 이를 보완하고 유지 관리합니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 지침

위협 모델링이란 무엇인가요?

"위협 모델링은 가치 있는 것을 보호한다는 맥락에서 위협과 완화 조치를 식별, 전달 및 이해하기 위해 작동합니다." – The Open Web Application Security Project (OWASP) Application Threat Modeling

위협 모델을 사용해야 하는 이유는 무엇인가요?

시스템은 복잡하고 시간이 지남에 따라 점점 더 복잡해지고 기능이 향상되어 더 많은 비즈니스 가치를 제공하고 고객 만족도와 참여도를 향상시킵니다. 즉, IT 설계를 결정할 때는 계속해서 증가하는 사용 사례를 고려해야 합니다. 이러한 복잡성과 사용 사례 순열의 수는 일반적으로 위협을 찾고 완화하는 데 구조화되지 않은 접근 방식을 비효율적으로 만듭니다. 대신 시스템에 대한 잠재적인 위협을 열거할 뿐만 아니라 조직의 제한된 리소스가 시스템의 전체 보안 태세를 개선하는 데 최대한 영향을 미칠 수 있도록 완화 조치를 고안하고 우선순위를 지정하는 체계적인 접근 방식이 필요합니다.

위협 모델링은 수명 주기 후반에 비해 상대적으로 완화 조치 관련 비용과 노력이 적게 필요한 설계 프로세스 초기에 문제를 찾아 해결하는 것을 목표로 이러한 체계적인 접근 방식을 제공하도록 설계되었습니다. 이 접근 방식은 시프트-레프트 보안이라는 업계 원칙과 일치합니다. 궁극적으로 위협 모델링은 조직의 위험 관리 프로세스와 통합되며 위협 기반 접근 방식을 사용하여 구현할 제어에 대한 결정을 내리는 데 도움이 됩니다.

위협 모델링은 언제 수행해야 하나요?

워크로드의 수명 주기에서 가능한 한 빠르게 위협 모델링을 시작합니다. 그러면 식별한 위협에 대해 유연성을 높일 수 있습니다. 소프트웨어 버그와 마찬가지로 위협을 조기에 식별할수록 위협을 해결하는 것이 더 비용 효율적입니다. 위협 모델은 최신 상태를 유지해야 하는 문서로 워크로드가 변경됨에 따라 계속 진화해야 합니다. 주요 변경 사항이 있거나 위협 환경이 변경되거나 새로운 기능 또는 서비스를 채택하는 경우를 포함하여 시간이 지남에 따라 위협 모델을 보완합니다.

구현 단계

위협 모델링을 어떻게 수행할 수 있나요?

위협 모델링을 수행하는 방법에는 여러 가지가 있습니다. 프로그래밍 언어와 마찬가지로 각각 장단점이 있으므로 자신에게 가장 적합한 방법을 선택해야 합니다. 한 가지 접근 방식은 Shostack's 4 Question Frame for Threat Modeling으로 시작하는 것입니다. 여기에서는 위협 모델링 실습에 대한 구조를 제공하기 위해 개방형 질문을 제시합니다.

1. 어떤 작업을 하고 있나요?

   이 질문의 목적은 구축 중인 시스템과 보안과 관련된 해당 시스템에 대한 세부 정보를 이해하고 동의하는 데 도움을 주기 위한 것입니다. 모델이나 다이어그램을 생성하는 것은 가령 데이터 흐름 다이어그램을 사용하여 구축 항목을 시각화하는 데 도움이 되므로 이 질문에 답하는 가장 일반적인 방법입니다. 시스템에 대한 권한 수임과 중요한 세부 정보를 작성하는 것도 범위를 정의하는 데 도움이 됩니다. 이를 통해 위협 모델에 기여하는 모든 담당자가 동일한 작업에 집중하고 범위 이외의 주제(시스템의 오래된 버전 포함)로 벗어나 시간을 허비하는 것을 방지할 수 있습니다. 예를 들어 웹 애플리케이션을 구축하는 경우, 사용자 설계를 통해 영향을 미칠 수 없기 때문에 브라우저 클라이언트에 대한 운영 체제의 신뢰할 수 있는 부팅 시퀀스에 대해 위협 모델링을 수행할 필요가 없습니다.
2. 잘못되면 어떻게 되나요?

   이 질문을 통해 시스템에 대한 위협을 식별합니다. 위협은 원치 않는 영향을 미치고 시스템의 보안에 영향을 미칠 수 있는 우발적이거나 의도적인 작업 또는 이벤트입니다. 무엇이 잘못될 수 있는지에 대한 명확한 이해 없이는 위협에 대해 대응할 수 있는 방법이 아무 것도 없습니다.

   무엇이 잘못될 수 있는지에 대한 표준 목록은 없습니다. 이 목록을 작성하려면 팀 내 모든 개인과 위협 모델링 수행에 관여하는 관련 대상자 간의 브레인스토밍과 협업이 필요합니다. STRIDE와 같은 위협 식별 모델을 사용하여 브레인스토밍을 지원할 수 있습니다. 이 모델은 스푸핑, 변조, 거부, 정보 공개, 서비스 거부 및 권한 상승과 같이 평가할 다양한 범주를 제안합니다. 또한 OWASP Top 10, HiTrust Threat Catalog, 조직의 자체 위협 카탈로그를 포함하여 아이디어를 얻을 수 있는 기존 목록과 연구를 검토하여 브레인스토밍을 지원할 수 있습니다.
3. 이에 대해 무엇을 할 수 있나요?

   이전 질문의 경우와 마찬가지로 가능한 모든 완화 조치에 대한 표준 목록은 없습니다. 이 단계에 대한 입력은 이전 단계에서 식별된 위협, 행위자 및 개선 영역입니다.

   보안과 규정 준수는 AWS와 고객의 공동 책임입니다. '이에 대해 무엇을 할 수 있나요?'라고 질문할 때 '이에 대한 책임은 누구에게 있나요?'라고 질문하는 것임을 이해하는 것이 중요합니다. 사용자와 AWS 간의 책임 균형을 이해하면 위협 모델링 수행의 범위를 관리되는 완화 조치로 지정하는 데 도

움이 됩니다. 이러한 완화 조치는 일반적으로 AWS 서비스 구성 옵션과 자체 시스템별 완화 조치의 조합으로 이루어집니다.

공동 책임에서 AWS가 맡은 부분의 경우 AWS 서비스가 많은 규정 준수 프로그램의 범위에 속해 있다는 것을 알 수 있습니다. 이러한 프로그램을 통해 클라우드의 보안 및 규정 준수를 유지하기 위해 AWS에 마련된 강력한 제어 기능을 이해할 수 있습니다. AWS 고객은 AWS Artifact에서 이러한 프로그램의 감사 보고서를 다운로드할 수 있습니다.

사용 중인 AWS 서비스에 관계없이 항상 고객 책임의 요소가 있으며 이러한 책임에 맞는 완화 조치가 위협 모델에 포함되어야 합니다. AWS 서비스 자체에 대한 보안 제어 완화 조치를 위해 자격 증명 및 액세스 관리(인증 및 권한 부여), 데이터 보호(저장 데이터 및 전송 중 데이터), 인프라 보안, 로깅, 모니터링과 같은 도메인을 포함한 도메인 전반에서 보안 제어 구현을 고려해야 합니다. 각 AWS 서비스에 대한 설명서에는 완화 조치로 고려할 보안 제어에 대한 지침을 제공하는 보안 전용 장이 있습니다. 작성 중인 코드와 해당 코드 종속성을 고려하고 이러한 위협을 해결하기 위해 마련할 수 있는 제어에 대해 생각하는 것이 중요합니다. 이러한 제어는 입력 검증, 세션 처리 및 경계 처리와 같은 기능에 해당될 수 있습니다. 대부분의 취약성은 사용자 지정 코드에서 발생하는 경우가 많으므로 이 영역에 집중하세요.

4. 잘 수행했나요?

   목표는 팀과 조직이 위협 모델의 품질과 시간이 지남에 따라 위협 모델링을 수행하는 속도를 모두 개선하는 것입니다. 이러한 개선은 연습, 학습, 지도, 검토의 조합에서 비롯됩니다. 더 자세히 알아보고 실습하려면 사용자와 팀이 Threat modeling the right way for builders 교육 과정 또는 워크숍을 완료하는 것이 좋습니다. 또한 위협 모델링을 조직의 애플리케이션 개발 수명 주기에 통합하는 방법에 대한 지침은 AWS 보안 블로그에서 How to approach threat modeling 게시물을 참조하세요.

Threat Composer

위협 모델링을 수행하는 데 도움과 안내를 받으려면 위협 모델링 시 가치 창출 시간을 단축하는 것을 목표로 하는 Threat Composer 도구를 사용하는 것이 좋습니다. 이 도구는 다음과 같은 작업을 수행하는 데 도움이 됩니다.

- 자연스러운 비선형 워크플로우에서 작동하는 위협 문법에 맞게 유용한 위협 설명을 작성합니다.
- 사람이 읽을 수 있는 위협 모델을 생성합니다.
- 기계가 읽을 수 있는 위협 모델을 생성하여 위협 모델을 코드로 취급할 수 있습니다.
- Insights Dashboard를 사용하여 품질 및 커버리지 개선 영역을 빠르게 식별할 수 있도록 도와줍니다.

자세한 내용을 보려면 Threat Composer를 방문하여 시스템 정의 Example Workspace로 전환합니다.

리소스

관련 모범 사례:

- SEC01-BP03 제어 목표 파악 및 검증
- SEC01-BP04 보안 위협 및 권장 사항에 대한 최신 정보 파악
- SEC01-BP05 보안 관리 범위 축소
- SEC01-BP08 새로운 보안 서비스 및 기능을 정기적으로 평가 및 구현

관련 문서:

- How to approach threat modeling (AWS Security Blog)
- NIST: Guide to Data-Centric System Threat Modelling

관련 비디오:

- AWS Summit ANZ 2021 - How to approach threat modelling
- AWS Summit ANZ 2022 - Scaling security – Optimise for fast and secure delivery

관련 교육:

- Threat modeling the right way for builders – AWS Skill Builder virtual self-paced training
- Threat modeling the right way for builders – AWS 워크숍

관련 도구:

- Threat Composer

SEC01-BP08 새로운 보안 서비스 및 기능을 정기적으로 평가 및 구현

워크로드의 보안 상태를 개선할 수 있는 AWS 및 AWS 파트너의 보안 서비스와 기능을 평가하고 구현합니다.

원하는 성과: AWS 및 AWS 파트너가 출시한 새로운 기능 및 서비스를 알려주는 표준 관행이 마련되어 있습니다. 이러한 최신 기능이 환경과 워크로드에 대한 현재 제어 설계와 새로운 제어 설계에 어떤 영향을 미치는지 평가합니다.

일반적인 안티 패턴:

- AWS 블로그와 RSS 피드를 구독하여 관련 새 기능 및 서비스를 신속하게 알아보지 않습니다.
- 2차 소스에서 제공하는 보안 서비스 및 기능에 대한 소식 및 업데이트를 적극 활용합니다.
- 조직의 AWS 사용자에게 최신 업데이트에 대한 새로운 정보를 계속 파악하도록 권장하지 않습니다.

이 모범 사례 확립의 이점: 새로운 보안 서비스 및 기능을 잘 파악하면 클라우드 환경 및 워크로드의 제어 구현에 대해 정보에 입각한 결정을 내릴 수 있습니다. 이러한 소스는 진화하는 보안 환경과 새롭게 등장하는 위협으로부터 AWS 서비스를 보호하는 데 사용할 수 있는 방법에 대한 인식을 높이는 데 도움이 됩니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 낮음

구현 지침

AWS는 다음과 같은 여러 채널을 통해 고객에게 새로운 보안 서비스 및 기능을 안내합니다.

- AWS의 새로운 소식
- AWS 뉴스 블로그
- AWS 보안 블로그
- AWS 보안 공지
- AWS 문서 개요

Amazon Simple Notification Service(SNS)를 통해 AWS 일간 기능 업데이트 주제를 구독하여 업데이트에 대한 포괄적인 일일 요약 정보를 확인할 수 있습니다. Amazon GuardDuty, AWS Security Hub와 같은 일부 보안 서비스는 해당 서비스에 대한 새로운 표준, 조사 결과 및 기타 업데이트 관련 정보를 파악할 수 있도록 자체 SNS 주제를 제공합니다.

나아가 매년 전 세계에서 개최되는 컨퍼런스, 이벤트, 웨비나를 통해 새로운 서비스와 기능에 대해 자세히 발표하고 설명합니다. 특히 주목할 것은 연례 AWS re:Inforce 보안 컨퍼런스와 보다 일반적인 AWS re:Invent 컨퍼런스입니다. 앞서 언급한 AWS 뉴스 채널은 이러한 컨퍼런스에서 보안 및 기타 서비스에 대해 발표한 내용을 공유합니다. YouTube의 AWS Events 채널에서 온라인으로 심층 분석 교육 브레이크아웃 세션을 볼 수 있습니다.

또한 AWS 계정 팀에 최신 보안 서비스 업데이트 및 권장 사항에 대해 문의할 수 있습니다. 직접 연결되는 연락처 정보가 없는 경우 영업 지원 양식을 통해 팀에 문의할 수 있습니다. 마찬가지로, AWS

Enterprise Support를 구독한 경우 TAM(Technical Account Manager)으로부터 매주 업데이트를 받고 정기적인 검토 회의 일정을 잡을 수 있습니다.

구현 단계

1. 좋아하는 RSS 리더와 함께 다양한 블로그와 게시판을 구독하거나 일간 기능 업데이트 SNS 주제를 구독합니다.
2. 어떤 AWS 이벤트에 참석해야 하는지 평가하여 새로운 기능과 서비스에 대해 직접 알아보세요.
3. 보안 서비스 및 기능 업데이트에 관한 질문이 있으면 AWS 계정 팀과 회의를 진행하세요.
4. Enterprise Support를 구독하여 TAM(Technical Account Manager)과 정기적으로 상담하는 것을 고려해 보세요.

리소스

관련 모범 사례:

- PERF01-BP01 사용 가능한 클라우드 서비스 및 기능 학습 및 이해
- COST01-BP07 새로운 서비스 릴리스로 최신 상태 유지

자격 증명 및 액세스 관리

AWS 서비스를 사용하려면 사용자와 애플리케이션에 AWS 계정 내 리소스에 대한 액세스 권한을 부여해야 합니다. AWS에서 더 많은 워크로드를 실행할 경우 적절한 사람이 적절한 조건에서 적절한 리소스에 액세스할 수 있도록 강력한 자격 증명 관리 및 권한이 필요합니다. AWS는 인력 및 시스템 자격 증명과 그 권한을 관리하는 데 도움이 되는 다양한 기능을 제공합니다. 이러한 기능에 대한 모범 사례는 두 가지 주요 영역으로 나뉩니다.

주제

- 자격 증명 관리
- 권한 관리

자격 증명 관리

보안 AWS 워크로드를 운영할 때는 두 가지 유형의 ID를 관리해야 합니다.

- 인적 자격 증명: AWS 환경 및 애플리케이션에 액세스해야 하는 인적 자격 증명은 인력, 서드파티 및 사용자라는 세 그룹으로 분류할 수 있습니다.

  인력 그룹에는 조직의 구성원인 관리자, 개발자 및 운영자가 포함됩니다. 이들이 AWS 리소스를 관리, 구축 및 운영하려면 액세스 권한이 필요합니다.

  서드파티는 계약업체, 공급업체 또는 파트너와 같은 외부 협력자입니다. 이들은 계약의 일환으로 AWS 리소스와 상호 작용합니다.

  사용자는 애플리케이션의 소비자입니다. 이들은 웹 브라우저, 클라이언트 애플리케이션, 모바일 앱 또는 대화형 명령줄 도구를 통해 AWS 리소스에 액세스합니다.
- 머신 자격 증명: 워크로드 애플리케이션, 운영 도구 및 구성 요소에서 AWS 서비스에 요청을 하려면(예: 데이터 읽기) 자격 증명이 필요합니다. 이러한 자격 증명에는 AWS 환경에서 실행되는 머신이 포함됩니다(예: Amazon EC2 인스턴스 또는 AWS Lambda 함수). 외부 당사자 또는 AWS 환경에 액세스해야 하는 AWS 외부 머신의 머신 자격 증명을 관리할 수도 있습니다.

모범 사례

- SEC02-BP01 강력한 로그인 메커니즘 사용
- SEC02-BP02 임시 자격 증명 사용
- SEC02-BP03 안전하게 보안 암호 저장 및 사용

- SEC02-BP04 중앙 집중식 ID 공급업체 사용
- SEC02-BP05 정기적으로 자격 증명 감사 및 교체
- SEC02-BP06 사용자 그룹 및 속성 사용

SEC02-BP01 강력한 로그인 메커니즘 사용

로그인(로그인 자격 증명을 사용한 인증)은 다중 인증(MFA)과 같은 메커니즘을 사용하지 않을 때, 특히 로그인 자격 증명이 의도치 않게 공개되었거나 쉽게 추측되는 상황에서 위험을 초래할 수 있습니다. 강력한 로그인 메커니즘을 사용하면 MFA 및 강력한 암호 정책을 요구하여 이러한 위험을 줄일 수 있습니다.

원하는 성과: AWS Identity and Access Management(IAM) 사용자, AWS 계정 루트 사용자, AWS IAM Identity Center, 서드파티 ID 제공업체에 대한 강력한 로그인 메커니즘을 사용하여 AWS의 자격 증명에 대한 의도치 않은 액세스 위험을 줄입니다. 즉, MFA를 요구하고 강력한 암호 정책을 적용하며 비정상적인 로그인 동작을 감지합니다.

일반적인 안티 패턴:

- 복잡한 암호 및 MFA를 포함하여 자격 증명에 대한 강력한 암호 정책을 적용하지 않습니다.
- 다른 사용자 간에 동일한 자격 증명을 공유합니다.
- 의심스러운 로그인에 대한 탐지 제어를 사용하지 않습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 지침

인적 자격 증명으로 AWS에 로그인하는 방법에는 몇 가지가 있습니다. AWS에 인증할 때 페더레이션(AWS IAM과 중앙 집중화된 IdP 간의 직접 SAML 2.0 페더레이션 또는 AWS IAM Identity Center 사용)을 사용하는 중앙 집중식 ID 제공업체를 사용하는 것이 AWS 모범 사례입니다. 이 경우 ID 제공업체 또는 Microsoft Active Directory를 사용하여 보안 로그인 프로세스를 설정합니다.

AWS 계정을 처음 열면 AWS 계정 루트 사용자로 시작합니다. 루트 사용자 계정은 사용자(및 루트 사용자가 필요한 작업)에 대한 액세스 권한을 설정할 때만 사용해야 합니다. AWS 계정을 개설한 직후에는 계정 루트 사용자에 대해 다중 인증(MFA)을 활성화하고 AWS 모범 사례 가이드를 사용하여 루트 사용자를 보호하는 것이 중요합니다.

AWS IAM Identity Center는 인력 사용자를 위해 설계되었으며 서비스 내에서 사용자 ID를 생성 및 관리하고 MFA로 로그인 프로세스를 보호할 수 있습니다. 반면 AWS Cognito는 애플리케이션의 외부 사

용자 ID에 대한 사용자 풀 및 ID 제공업체를 제공하는 고객 ID 및 액세스 관리(CIAM)용으로 설계되었습니다.

AWS IAM Identity Center에서 사용자를 생성하는 경우 해당 서비스에서 로그인 프로세스를 보호하고 MFA를 켭니다. 애플리케이션의 외부 사용자 ID의 경우 Amazon Cognito 사용자 풀을 사용하고 해당 서비스에서 로그인 프로세스를 보호하거나 Amazon Cognito 사용자 풀이 지원하는 ID 제공업체 중 하나를 사용할 수 있습니다.

또한 AWS IAM Identity Center의 사용자의 경우 AWS 리소스에 대한 액세스 권한을 부여하기 전에 AWS Verified Access를 사용하여 사용자의 자격 증명 및 디바이스 태세를 확인하여 추가 보안 계층을 제공할 수 있습니다.

AWS Identity and Access Management(IAM) 사용자를 사용하는 경우 IAM을 사용하여 로그인 프로세스를 보호합니다.

AWS IAM Identity Center와 직접 IAM 페더레이션을 동시에 사용하여 AWS에 대한 액세스를 관리할 수 있습니다. IAM 페더레이션을 사용하여 AWS Management Console 및 서비스에 대한 액세스를 관리하고 IAM Identity Center를 사용하여 QuickSight 또는 Amazon Q Business와 같은 비즈니스 애플리케이션에 대한 액세스를 관리할 수 있습니다.

로그인 방법에 관계없이 강력한 로그인 정책을 적용하는 것이 중요합니다.

구현 단계

다음은 일반적인 강력한 로그인 권장 사항입니다. 구성하는 실제 설정은 회사 정책에 따라 설정하거나 NIST 800-63과 같은 표준을 사용해야 합니다.

- MFA 필수(Require MFA). 사람의 ID 및 워크로드에 MFA를 요구하는 것이 IAM 모범 사례입니다. MFA를 활성화하면 사용자가 로그인 자격 증명과 일회용 암호(OTP) 또는 하드웨어 디바이스에서 암호로 확인 및 생성된 문자열을 제공해야 하는 추가 보안 계층이 제공됩니다.
- 암호 강도의 기본 요소인 최소 암호 길이를 적용합니다.
- 암호 복잡성을 적용하여 암호를 추측하기 어렵게 만듭니다.
- 사용자에게 자신의 암호를 변경할 수 있도록 허용
- 공유 자격 증명 대신 개별 자격 증명을 생성합니다. 개별 자격 증명을 생성하여 각 사용자에게 고유한 보안 자격 증명 세트를 제공할 수 있습니다. 개별 사용자는 각 사용자의 활동을 감사할 수 있는 기능을 제공합니다.

IAM Identity Center 권장 사항:

- IAM Identity Center는 암호 길이, 복잡성 및 재사용 요구 사항을 설정하는 기본 디렉터리를 사용할 때 미리 정의된 암호 정책을 제공합니다.
- MFA를 활성화하고 자격 증명 소스가 기본 디렉터리, AWS Managed Microsoft AD 또는 AD Connector인 경우 MFA에 대한 컨텍스트 인식 또는 상시 설정을 구성합니다.
- 사용자가 자신의 MFA 디바이스를 등록하도록 허용합니다.

Amazon Cognito 사용자 풀 디렉터리 권장 사항:

- 암호 강도 설정을 구성합니다.
- 사용자에게 MFA를 요청합니다.
- Amazon Cognito 사용자 풀의 고급 보안 설정을 사용하여 의심되는 로그인을 차단할 수 있는 적응형 인증과 같은 기능을 사용합니다.

IAM 사용자 권장 사항:

- IAM Identity Center 또는 직접 페더레이션을 사용하는 것이 좋습니다. 그러나 IAM 사용자가 필요할 수 있습니다. 이 경우 IAM 사용자에 대한 암호 정책을 설정합니다. 암호 정책을 사용하여 최소 길이 또는 알파벳 이외 문자 포함 여부 등과 같은 요구 사항을 정의할 수 있습니다.
- MFA 로그인을 적용하도록 IAM 정책을 생성합니다. 그러면 사용자가 자신의 암호와 MFA 디바이스를 관리할 수 있습니다.

리소스

관련 모범 사례:

- SEC02-BP03 안전하게 보안 암호 저장 및 사용
- SEC02-BP04 중앙 집중식 ID 공급업체 사용
- SEC03-BP08 안전하게 조직과 리소스 공유

관련 문서:

- AWS IAM Identity Center 암호 정책
- IAM 사용자의 암호 정책
- AWS 계정 루트 사용자 암호 설정

- Amazon Cognito password policy
- AWS 보안 인증 정보
- IAM 보안 모범 사례

관련 비디오:

- Managing user permissions at scale with AWS IAM Identity Center
- Mastering identity at every layer of the cake

SEC02-BP02 임시 자격 증명 사용

모든 유형의 인증을 수행할 때 자격 증명이 의도치 않게 공개되거나 공유되거나 도난당하는 위험을 줄이거나 제거하기 위해 장기 자격 증명 대신 임시 자격 증명을 사용하는 것이 가장 좋습니다.

원하는 성과: 장기 자격 증명의 위험을 줄이려면 가능한 한 인적 자격 증명과 시스템 자격 증명 모두에 대해 임시 자격 증명을 사용합니다. 장기 자격 증명은 퍼블릭 리포지토리에 대한 업로드를 통한 노출과 같은 많은 위험을 초래합니다. 임시 자격 증명을 사용하면 자격 증명이 손상될 가능성이 크게 줄어듭니다.

일반적인 안티 패턴:

- 개발자가 페더레이션을 사용하여 CLI에서 임시 자격 증명을 획득하는 대신 IAM 사용자의 장기 액세스 키를 사용합니다.
- 개발자가 장기 액세스 키를 코드에 포함하고 해당 코드를 퍼블릭 Git 리포지토리에 업로드합니다.
- 개발자가 앱 스토어에서 사용할 수 있도록 장기 액세스 키를 모바일 앱에 포함합니다.
- 사용자가 다른 사용자와 장기 액세스 키를 공유하거나 직원이 장기 액세스 키를 소유한 상태로 퇴사합니다.
- 임시 자격 증명을 사용할 수 있는 경우에도 시스템 자격 증명에 장기 액세스 키를 사용합니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

모든 AWS API 및 CLI 요청에 대해 장기 자격 증명 대신 임시 자격 증명을 사용합니다. AWS 서비스에 대한 API 및 CLI 요청은 거의 모든 경우에 AWS 액세스 키를 사용하여 서명해야 합니다. 이러한 요청은 임시 또는 장기 자격 증명으로 서명할 수 있습니다. 장기 액세스 키라고도 하는 장기 자격 증명을 사용

해야 하는 유일한 경우는 IAM 사용자 또는 AWS 계정 루트 사용자를 사용하는 경우입니다. 다른 방법을 통해 AWS에 페더레이션하거나 IAM 역할을 수임할 때 임시 자격 증명이 생성됩니다. 로그인 자격 증명을 사용하여 AWS Management Console에 액세스하는 경우에도 AWS 서비스를 직접 호출할 수 있도록 임시 자격 증명이 생성됩니다. 장기 자격 증명이 필요한 경우는 거의 없으며 임시 자격 증명을 사용하여 대부분의 작업을 수행할 수 있습니다.

임시 자격 증명을 선호하여 장기 자격 증명 사용을 피하는 것은 페더레이션 및 IAM 역할과 함께 IAM 사용자의 사용을 줄이는 전략과 함께 수행해야 합니다. 과거에는 IAM 사용자가 인적 자격 증명과 시스템 자격 증명 모두에 사용되었지만 이제는 장기 액세스 키 사용의 위험을 피하기 위해 사용하지 않는 것이 좋습니다.

구현 단계

인적 자격 증명

직원, 관리자, 개발자, 운영자와 같은 인적 자격 증명의 경우:

- 중앙 집중식 ID 제공업체를 사용해야 하며 인적 사용자가 임시 자격 증명을 사용하여 AWS에 액세스하려면 ID 제공업체와의 페더레이션을 사용해야 합니다. 사용자에 대한 페더레이션은 각 AWS 계정에 대한 직접 페더레이션을 사용하거나 AWS IAM Identity Center 및 선택한 ID 제공업체를 사용하여 수행할 수 있습니다. 페더레이션은 장기 자격 증명을 제거하는 것 외에도 IAM 사용자를 사용하는 것보다 많은 이점을 제공합니다. 사용자는 직접 페더레이션을 위해 명령줄에서 또는 IAM Identity Center를 사용하여 임시 자격 증명을 요청할 수도 있습니다. 즉, IAM 사용자 또는 사용자에 대한 장기 자격 증명이 필요한 사용 사례가 거의 없습니다.

서드파티 자격 증명의 경우:

- 서비스형 소프트웨어(SaaS) 공급자와 같은 서드파티에 AWS 계정의 리소스에 대한 액세스 권한을 부여할 때 크로스 계정 역할 및 리소스 기반 정책을 사용할 수 있습니다. 또한 B2B SaaS 고객 또는 파트너를 위한 Amazon Cognito OAuth 2.0 권한 부여 클라이언트 자격 증명 흐름을 사용할 수 있습니다.

웹 브라우저, 클라이언트 애플리케이션, 모바일 앱 또는 대화형 명령줄 도구를 통해 AWS 리소스에 액세스하는 사용자 자격 증명:

- 소비자 또는 고객에게 AWS 리소스에 대한 액세스 권한을 부여해야 하는 경우 Amazon Cognito 자격 증명 풀 또는 Amazon Cognito 사용자 풀을 사용하여 임시 자격 증명을 제공할 수 있습니다. 자격 증명의 권한은 IAM 역할을 통해 제어됩니다. 인증되지 않은 게스트 사용자의 권한이 제한된 개별 IAM 역할을 정의할 수도 있습니다.

기계 자격 증명

시스템 자격 증명의 경우 장기 자격 증명을 사용해야 할 수 있습니다. 이 경우 AWS에 액세스하려면 워크로드에 IAM 역할이 있는 임시 자격 증명을 사용하도록 요구해야 합니다.

- Amazon Elastic Compute Cloud(Amazon EC2)의 경우 Amazon EC2의 역할을 사용할 수 있습니다.
- https://docs.aws.amazon.com/lambda/latest/dg/lambda-intro-execution-role.htmlAWS를 사용하면 임시 자격 증명을 사용하여 AWS Lambda 작업을 수행할 수 있는 서비스 권한을 부여하도록 Lambda 실행 역할을 구성할 수 있습니다. IAM 역할을 사용하여 임시 자격 증명을 부여하는 AWS 서비스에 대한 다른 유사한 모델이 많이 있습니다.
- IoT 디바이스의 경우 AWS IoT Core 자격 증명 공급자를 사용하여 임시 자격 증명을 요청할 수 있습니다.
- AWS 리소스에 액세스해야 하는 AWS 외부에서 실행되는 시스템 또는 온프레미스 시스템의 경우 IAM Roles Anywhere를 사용할 수 있습니다.

임시 자격 증명이 지원되지 않는 시나리오가 있으며, 이 경우 장기 자격 증명을 사용해야 합니다. 이러한 상황에서는 자격 증명을 주기적으로 감사 및 교체하고 정기적으로 액세스 키를 교체합니다. 매우 제한된 IAM 사용자 액세스 키의 경우 다음과 같은 추가 보안 조치를 고려하세요.

- 매우 제한된 권한 부여:
  - 최소 권한 원칙을 준수합니다(작업, 리소스 및 조건을 구체적으로 지정).
  - IAM 사용자에게 하나의 특정 역할에 대한 AssumeRole 작업만 부여하는 것을 고려합니다. 온프레미스 아키텍처에 따라 이 접근 방식은 장기 IAM 자격 증명을 격리하고 보호하는 데 도움이 됩니다.
- IAM 역할 신뢰 정책에서 허용되는 네트워크 소스 및 IP 주소를 제한합니다.
- 사용량을 모니터링하고 미사용 권한 또는 오용에 대한 알림을 설정합니다(AWS CloudWatch Logs 지표 필터 및 경보 사용).
- 권한 경계를 적용합니다(서비스 제어 정책(SCP) 및 권한 경계가 서로를 보완함 - SCP는 개괄적이지만 권한 경계는 세분화됨).
- 자격 증명을 프로비저닝하고 안전하게 저장하는 프로세스를 구현합니다(온프레미스 볼트에 저장).

장기 보안 인증이 필요한 시나리오에 대한 몇 가지 다른 옵션은 다음과 같습니다.

- 자체 토큰 벤딩 API를 구축합니다(Amazon API Gateway 사용).
- 장기 자격 증명을 사용해야 하는 상황이나 데이터베이스 로그인과 같이 AWS 액세스 키 이외의 자격 증명을 사용해야 하는 시나리오에서 AWS Secrets Manager와 같은 보안 암호 관리를 처리하도록 설

계된 서비스를 사용할 수 있습니다. Secrets Manager는 암호화된 보안 암호의 관리, 교체 및 보안 저장을 간소화합니다. 많은 AWS 서비스가 Secrets Manager와의 직접 통합을 지원합니다.

- 멀티 클라우드 통합의 경우 소스 자격 증명 서비스 공급자(CSP) 자격 증명을 기반으로 ID 페더레이션을 사용할 수 있습니다(AWS STSAssumeRoleWithWebIdentity 참조).

장기 자격 증명 교체에 대한 자세한 내용은 rotating access keys를 참조하세요.

리소스

관련 모범 사례:

- SEC02-BP03 안전하게 보안 암호 저장 및 사용
- SEC02-BP04 중앙 집중식 ID 공급업체 사용
- SEC03-BP08 안전하게 조직과 리소스 공유

관련 문서:

- Temporary Security Credentials
- AWS 보안 인증
- IAM 보안 모범 사례
- IAM 역할
- IAM Identity Center
- Identity Providers and Federation
- Rotating Access Keys
- 보안 파트너 솔루션: 액세스 및 액세스 제어
- AWS 계정 루트 사용자
- Access AWS using a Google Cloud Platform native workload identity
- How to access AWS resources from Microsoft Entra ID tenants using AWS Security Token Service

관련 비디오:

- Managing user permissions at scale with AWS IAM Identity Center
- Mastering identity at every layer of the cake

SEC02-BP03 안전하게 보안 암호 저장 및 사용

워크로드에는 데이터베이스, 리소스 및 서드파티 서비스에 대한 자격 증명을 증명하는 자동화된 기능이 필요합니다. 이는 API 액세스 키, 암호, OAuth 토큰과 같은 보안 암호 액세스 자격 증명을 사용하여 수행됩니다. 특별 제작된 서비스를 사용하여 이러한 자격 증명을 저장, 관리 및 교체하면 해당 자격 증명이 손상될 가능성을 줄이는 데 도움이 됩니다.

원하는 성과: 다음 목표를 달성하는 애플리케이션 자격 증명을 안전하게 관리하기 위한 메커니즘을 구현합니다.

- 워크로드에 필요한 보안 암호를 식별합니다.
- 가능한 경우 장기 자격 증명을 단기 자격 증명으로 대체하여 필요한 장기 자격 증명의 수를 줄입니다.
- 나머지 장기 자격 증명의 안전한 저장 및 자동 교체를 설정합니다.
- 워크로드에 존재하는 보안 암호에 대한 액세스 권한을 감사합니다.
- 개발 프로세스 중에 소스 코드에 보안 암호가 포함되어 있지 않은지 확인하기 위해 지속적으로 모니터링합니다.
- 자격 증명이 의도치 않게 공개될 가능성을 줄입니다.

일반적인 안티 패턴:

- 자격 증명을 교체하지 않습니다.
- 소스 코드 또는 구성 파일에 장기 자격 증명을 저장합니다.
- 자격 증명을 저장 시 암호화되지 않은 상태로 저장합니다.

이 모범 사례 확립의 이점:

- 보안 암호는 저장 시 및 전송 중 암호화된 상태로 저장됩니다.
- 자격 증명에 대한 액세스 권한은 API를 통해 제한됩니다(자격 증명 자판기와 같음).
- 자격 증명에 대한 액세스 권한(읽기 및 쓰기 모두)은 감사되고 로깅됩니다.
- 우려 사항 분리: 자격 증명 교체는 아키텍처의 나머지 부분과 분리될 수 있는 별도의 구성 요소에 의해 수행됩니다.
- 보안 암호는 필요에 따라 소프트웨어 구성 요소에 자동으로 배포되며 중앙 위치에서 교체가 수행됩니다.
- 자격 증명에 대한 액세스 권한은 세분화된 방식으로 제어할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

과거에는 데이터베이스, 서드파티 API, 토큰 및 기타 보안 암호에 인증하는 데 사용되는 자격 증명이 소스 코드나 환경 파일에 포함되었을 수 있습니다. AWS는 이러한 자격 증명을 안전하게 저장하고 자동으로 교체하며 사용을 감사하는 여러 메커니즘을 제공합니다.

보안 암호 관리에 접근하는 가장 좋은 방법은 제거, 대체 및 교체 지침을 따르는 것입니다. 가장 안전한 자격 증명은 저장, 관리 또는 처리할 필요가 없는 자격 증명입니다. 안전하게 제거할 수 있는 워크로드 기능에 더 이상 필요하지 않은 자격 증명이 있을 수 있습니다.

워크로드의 적절한 기능을 위해 여전히 필요한 자격 증명의 경우 장기 자격 증명을 임시 또는 단기 자격 증명으로 대체할 기회가 있을 수 있습니다. 예를 들어 AWS 비밀 액세스 키를 하드 코딩하는 대신 IAM 역할을 사용하여 해당 장기 자격 증명을 임시 자격 증명으로 대체하는 것이 좋습니다.

일부 장기 보안 암호는 제거하거나 대체하지 못할 수 있습니다. 이러한 보안 암호는 AWS Secrets Manager와 같은 서비스에 저장될 수 있습니다. 이를 통해 중앙에서 저장 및 관리되고 정기적으로 교체될 수 있습니다.

워크로드의 소스 코드 및 구성 파일을 감사하면 여러 유형의 자격 증명을 확인할 수 있습니다. 다음 표에는 일반적인 유형의 자격 증명을 처리하기 위한 전략이 요약되어 있습니다.

자격 증명 유형	설명	제안된 전략
IAM 액세스 키	워크로드 내에서 IAM 역할을 수임하는 데 사용되는 AWS IAM 액세스 및 비밀 키	교체: 대신 컴퓨팅 인스턴스에 할당된 IAM 역할을 사용합니다(예: Amazon EC2 또는 AWS Lambda). AWS 계정의 리소스에 대한 액세스가 필요한 서드파티와의 상호 운용성을 위해 AWS 크로스 계정 액세스를 지원하는지 문의합니다. 모바일 앱의 경우 Amazon Cognito 자격 증명 풀(페더레이션 자격 증명)을 통한 임시 자격 증명 사용을 고려하세요. AWS 외부에서 실행되는 워크로드의 경우 IAM Roles Anywhere 또는

자격 증명 유형	설명	제안된 전략
		AWS Systems Manager 하이브리드 정품 인증을 고려하세요. 컨테이너는 Amazon ECS 작업 IAM 역할 또는 Amazon EKS node IAM role을 참조하세요.
SSH 키	수동으로 또는 자동화된 프로세스의 일부로 Linux EC2 인스턴스에 로그인하는 데 사용되는 Secure Shell 프라이빗 키	교체: AWS Systems Manager 또는 EC2 Instance Connect를 사용하여 IAM 역할을 통해 EC2 인스턴스에 대한 프로그래밍 방식의 액세스 및 인적 액세스를 제공합니다.
애플리케이션 및 데이터베이스 자격 증명	암호 - 일반 텍스트 문자열	교체: AWS Secrets Manager에 자격 증명을 저장하고 가능하면 자동 교체를 설정합니다.
Amazon RDS 및 Aurora 관리자 데이터베이스 자격 증명	암호 - 일반 텍스트 문자열	교체: Amazon RDS와의 Secrets Manager 통합 또는 Amazon Aurora를 사용합니다. 또한 일부 RDS 데이터베이스 유형은 일부 사용 사례에서 암호 대신 IAM 역할을 사용할 수 있습니다(자세한 내용은 IAM 데이터베이스 인증 참조).
OAuth 토큰	보안 암호 토큰 - 일반 텍스트 문자열	교체: AWS Secrets Manager에 토큰을 저장하고 자동 교체를 구성합니다.
API 토큰 및 키	보안 암호 토큰 - 일반 텍스트 문자열	교체: AWS Secrets Manager에 저장하고 가능하면 자동 교체를 설정합니다.

일반적인 안티 패턴은 소스 코드, 구성 파일 또는 모바일 앱 내부에 IAM 액세스 키를 포함하는 것입니다. AWS 서비스와 통신하는 데 IAM 액세스 키가 필요한 경우 임시 (단기) 보안 자격 증명을 사용합니다. 이러한 단기 자격 증명은 EC2 인스턴스의 경우 IAM 역할, Lambda 함수의 경우 실행 역할, 모바일 사용자 액세스의 경우 Cognito IAM 역할, IoT 기기의 경우 IoT Core 정책을 통해 제공할 수 있습니다. 서드파티와 연결할 때 IAM 사용자를 구성하고 서드파티에 해당 사용자의 비밀 액세스 키를 보내는 것보다 계정 리소스에 필수 액세스 권한이 있는 IAM 역할에 대한 액세스 권한을 위임하는 것이 좋습니다.

워크로드에 다른 서비스 및 리소스와 상호 운용하는 데 필요한 보안 암호를 저장해야 하는 경우가 많습니다. AWS Secrets Manager는 이러한 자격 증명은 물론, API 토큰, 암호 및 기타 자격 증명의 저장, 사용 및 교체를 안전하게 관리하기 위해 특별히 제작되었습니다.

AWS Secrets Manager는 민감한 자격 증명의 안전한 저장 및 처리를 보장하는 5가지 주요 기능, 즉 저장 시 암호화, 전송 중 암호화, 종합적 감사, 세분화된 액세스 제어, 확장 가능한 자격 증명 교체를 제공합니다. AWS 파트너의 기타 보안 암호 관리 서비스 또는 유사한 기능과 보증을 제공하는 현지 개발 솔루션도 허용됩니다.

보안 암호를 검색할 때 Secret Manager 클라이언트 측 캐싱 구성 요소를 사용하여 나중에 사용할 수 있도록 캐싱할 수 있습니다. 캐싱된 보안 암호를 검색하는 것이 Secret Manager에서 검색하는 것보다 빠릅니다. 또한 Secrets Manager API를 호출하는 데는 비용이 발생하므로 캐시를 사용하면 비용을 줄일 수 있습니다. 암호를 검색할 수 있는 모든 방법은 Get secrets을 참조하세요.

> **Note**
>
> 일부 언어에서는 클라이언트 측 캐싱을 위해 자체 메모리 내 암호화를 구현해야 할 수 있습니다.

구현 단계

1. Amazon CodeGuru와 같은 자동화된 도구를 사용하여 하드 코딩된 자격 증명이 포함된 코드 경로를 식별합니다.
   a. Amazon CodeGuru를 사용하여 코드 리포지토리를 스캔합니다. 검토가 완료되면 CodeGuru에서 유형=보안 암호를 필터링하여 문제가 있는 코드 줄을 찾습니다.
2. 제거하거나 대체할 수 있는 자격 증명을 식별합니다.
   a. 더 이상 필요하지 않은 자격 증명을 식별하고 제거하도록 표시합니다.
   b. 소스 코드에 포함된 AWS 보안 암호 키의 경우 필요한 리소스와 연결된 IAM 역할로 대체합니다. 워크로드의 일부가 AWS 외부에 있지만 AWS 리소스에 액세스하기 위해 IAM 자격 증명이 필요한 경우 IAM Roles Anywhere 또는 AWS Systems Manager Hybrid Activations을 고려합니다.

3. 교체 전략을 사용해야 하는 서드파티의 장기 보안 암호의 경우 Secrets Manager를 코드에 통합하여 런타임 시 서드파티 보안 암호를 검색합니다.
   a. CodeGuru 콘솔은 검색된 자격 증명을 사용하여 Secrets Manager에서 보안 암호를 자동으로 생성할 수 있습니다.
   b. Secrets Manager의 보안 암호 검색을 애플리케이션 코드에 통합합니다.
      i. 서버리스 Lambda 함수는 언어에 구애받지 않는 Lambda 확장을 사용할 수 있습니다.
      ii. EC2 인스턴스 또는 컨테이너의 경우 AWS는 널리 사용되는 여러 프로그래밍 언어로 Secrets Manager에서 보안 암호를 검색하기 위한 클라이언트 측 코드 예제를 제공합니다.
4. 주기적으로 코드 베이스를 검토하고 다시 스캔하여 코드에 추가된 새 보안 암호가 없는지 확인합니다.
   a. 소스 코드 리포지토리에 새로운 보안 암호가 커밋되지 않도록 git-secrets와 같은 도구 사용을 고려하세요.
5. 예상치 못한 사용, 부적절한 보안 암호 액세스 또는 보안 암호 삭제 시도가 있는지 Secrets Manager 활동을 모니터링합니다.
6. 자격 증명에 대한 인적 노출을 줄입니다. 자격 증명을 읽고 쓰고 수정할 수 있는 액세스 권한을 이 목적을 위한 전용 IAM 역할로 제한하고, 해당 역할을 수임할 수 있는 액세스 권한을 일부 운영 사용자에게만 제공합니다.

리소스

관련 모범 사례:

- SEC02-BP02 임시 자격 증명 사용
- SEC02-BP05 정기적으로 자격 증명 감사 및 교체

관련 문서:

- AWS Secrets Manager 시작하기
- Identity Providers and Federation
- Amazon CodeGuru Introduces Secrets Detector
- AWS Secrets Manager의 AWS Key Management Service 활용 방식
- Secret encryption and decryption in Secrets Manager
- Secrets Manager 블로그 항목
- Amazon RDS, AWS Secrets Manager와의 통합 발표

관련 비디오:

- Best Practices for Managing, Retrieving, and Rotating Secrets at Scale
- Find Hard-Coded Secrets Using Amazon CodeGuru Secrets Detector
- Securing Secrets for Hybrid Workloads Using AWS Secrets Manager

관련 워크숍:

- Store, retrieve, and manage sensitive credentials in AWS Secrets Manager
- AWS Systems Manager Hybrid Activations

SEC02-BP04 중앙 집중식 ID 공급업체 사용

직원 ID(직원 및 계약업체)의 경우 중앙 위치에서 ID를 관리할 수 있는 ID 제공업체를 이용하세요. 이렇게 하면 단일 위치에서 액세스를 생성, 관리 및 취소하므로 여러 애플리케이션과 서비스에 대한 액세스를 더 쉽게 관리할 수 있습니다.

원하는 성과: 중앙 집중식 ID 제공업체를 통해 직원, 인증 정책(예: 다중 인증(MFA) 요구), 시스템 및 애플리케이션에 대한 권한 부여(예: 사용자의 그룹 구성원 자격 또는 특성에 따른 액세스 할당)를 중앙에서 관리할 수 있습니다. 직원은 중앙 ID 제공업체에 로그인하고 내부 및 외부 애플리케이션에 페더레이션(sSingle Sign On)하므로 사용자가 여러 자격 증명을 기억할 필요가 없습니다. ID 제공업체는 인사(HR) 시스템과 통합되므로 직원 변경 사항이 ID 제공업체와 자동으로 동기화됩니다. 예를 들어, 누군가가 조직을 떠나는 경우 페더레이션된 애플리케이션 및 시스템(AWS 포함)에 대한 액세스를 자동으로 취소할 수 있습니다. ID 제공업체에서 세부 감사 로깅을 활성화했으며 이러한 로그를 모니터링하여 비정상적인 사용자 동작이 있는지 확인하고 있습니다.

일반적인 안티 패턴:

- 페더레이션 및 Single Sign-On은 사용하지 않습니다. 직원이 여러 애플리케이션과 시스템에서 별도의 사용자 계정과 자격 증명을 생성합니다.
- ID 제공업체를 HR 시스템에 통합하는 등 직원의 ID 수명 주기를 자동화하지 않습니다. 사용자가 조직을 떠나거나 역할을 변경하면 수동 프로세스에 따라 여러 애플리케이션 및 시스템에서 기록을 삭제하거나 업데이트합니다.

이 모범 사례 확립의 이점: 중앙 집중식 ID 제공업체를 사용하면 직원 ID 및 정책을 한 곳에서 관리하고, 사용자와 그룹에 애플리케이션 액세스 권한을 할당하며, 사용자 로그인 활동을 모니터링할 수 있습니다. 인사(HR) 시스템과 통합하면 사용자가 역할을 변경하면 이러한 변경 내용이 ID 제공업체와 동기화

되고 할당된 애플리케이션 및 권한이 자동으로 업데이트됩니다. 사용자가 조직을 떠나면 ID 제공업체에서 해당 ID가 자동으로 비활성화되어 페더레이션된 애플리케이션 및 시스템에 대한 액세스 권한이 취소됩니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 지침

AWS에 액세스하는 인력 사용자를 위한 지침 조직의 직원 및 계약직과 같은 인력 사용자는 직무를 수행하기 위해 AWS Management Console 또는 AWS Command Line Interface(AWS CLI)를 사용하여 AWS에 대한 액세스 권한이 필요할 수 있습니다. 중앙 집중식 ID 제공업체에서 두 가지 수준으로 AWS에 페더레이션하여 직원에게 AWS 액세스 권한을 부여할 수 있습니다. 즉, 각 AWS 계정에 대해 직접 페더레이션하거나 AWS 조직의 여러 계정에 페더레이션할 수 있습니다.

인력 사용자를 각 AWS 계정과 직접 페더레이션하려면 중앙 집중식 ID 제공업체를 사용하여 해당 계정에서 AWS Identity and Access Management에 페더레이션할 수 있습니다. IAM의 유연성을 통해 각 AWS 계정에 대해 별도의 SAML 2.0 또는 OIDC(Open ID Connect) ID 제공업체를 지원하고 액세스 제어를 위해 페더레이션 사용자 속성을 사용할 수 있습니다. 직원은 웹 브라우저를 사용하여 자격 증명(예: 암호 및 MFA 토큰 코드)을 제공하여 ID 제공업체에 로그인합니다. ID 제공업체가 브라우저에 SAML 어설션을 발행하면 이는 AWS Management Console 로그인 URL에 제출되고 사용자가 IAM 역할을 수임하여 AWS Management Console에 Single Sign-On으로 로그인할 수 있습니다. 또한 사용자는 ID 제공업체로부터 SAML 어설션을 사용해 IAM 역할을 수임하여 AWS STS로부터 AWS CLI 또는 AWS SDK에서 사용할 임시 AWS API 자격 증명을 확보할 수 있습니다.

직원을 AWS 조직의 여러 계정과 페더레이션하려면 AWS IAM Identity Center를 사용하여 AWS 계정 및 애플리케이션에 대한 직원의 액세스를 중앙에서 관리할 수 있습니다. 조직의 Identity Center를 활성화하고 자격 증명 소스를 구성합니다. IAM Identity Center는 사용자 및 그룹을 관리하는 데 사용할 수 있는 기본 자격 증명 소스 디렉터리를 제공합니다. 또는 SAML 2.0을 사용하여 외부 ID 제공업체에 연결하고 SCIM을 사용하여 사용자 및 그룹을 자동으로 프로비저닝하거나 AWS Directory Service를 사용하여 Microsoft AD 디렉터리에 연결함으로서 외부 자격 증명 소스를 선택할 수도 있습니다. 자격 증명 소스가 구성되면 권한 집합에 최소 권한 정책을 정의하여 AWS 계정 계정에 대한 사용자 및 그룹의 액세스 권한을 할당할 수 있습니다. 직원은 중앙 ID 제공업체를 통해 인증하여 AWS 액세스 포털에 로그인하고 할당된 클라우드 애플리케이션 및 AWS 계정에 Single Sign On으로 로그인할 수 있습니다. 사용자는 Identity Center를 통해 인증하고 AWS CLI 명령을 실행하기 위한 자격 증명을 얻기 위해 AWS CLI v2를 구성할 수 있습니다. 또한 Identity Center는 Amazon SageMaker AI Studio 및 AWS IoT Sitewise Monitor 포털과 같은 AWS 애플리케이션에 대한 Single Sign On을 허용합니다.

위의 지침을 따른 후에는 작업자가 AWS에서 워크로드를 관리할 때 정상적인 작업을 위해 더 이상 IAM 사용자와 그룹을 사용할 필요가 없습니다. 대신 사용자와 그룹은 AWS 외부에서 관리되며 사용자는 페

더레이션형 ID로 AWS 리소스에 액세스할 수 있습니다. 페더레이션 ID는 중앙 ID 제공업체가 정의한 그룹을 사용합니다. AWS 계정에서 더 이상 필요하지 않은 IAM 그룹, IAM 사용자 그리고 장기 사용자 자격 증명(암호 및 액세스 키)을 식별하고 제거해야 합니다. IAM 자격 증명 보고서를 사용하여 사용하지 않은 자격 증명을 찾고, 해당 IAM 사용자를 삭제하며, IAM 그룹을 삭제할 수 있습니다. 조직에 서비스 제어 정책(SCP)를 적용하여 새로운 IAM 사용자 및 그룹 생성을 방지하고 페더레이션된 ID를 통해 AWS에 액세스하도록 할 수 있습니다.

> **Note**
>
> 사용자는 Automatic provisioning 설명서에 설명된 대로 SCIM 액세스 토큰의 교체를 처리할 책임이 있습니다. 또한 ID 페더레이션을 지원하는 인증서를 교체할 책임이 있습니다.

애플리케이션 사용자를 위한 지침 Amazon Cognito를 중앙 집중식 ID 제공업체로 사용하여 모바일 앱과 같은 애플리케이션 사용자의 자격 증명을 관리할 수 있습니다. Amazon Cognito는 웹 및 모바일 앱에 대한 인증, 권한 부여 및 사용자 관리를 지원합니다. Amazon Cognito는 수백만 명의 사용자로 확장 가능한 자격 증명 스토어를 제공하고, 소셜 및 엔터프라이즈 ID 페더레이션을 지원하며, 고급 보안 기능을 제공하여 사용자와 비즈니스를 보호할 수 있도록 지원합니다. 사용자 지정 웹 또는 모바일 애플리케이션을 Amazon Cognito에 통합하여 몇 분 만에 애플리케이션에 사용자 인증 및 액세스 제어를 추가할 수 있습니다. SAML 및 OIDC(Open ID Connect)와 같은 개방형 ID 표준을 기반으로 구축된 Amazon Cognito는 다양한 규정 준수 규정을 지원하고 프론트엔드 및 백엔드 개발 리소스와 통합됩니다.

구현 단계

직원이 AWS에 액세스하는 단계

- 다음 접근 방식 중 하나인 AWS 사용하여 직원을 중앙 집중식 ID 제공업체를 사용하도록 통합하세요.
  - IAM Identity Center를 사용하여 ID 제공업체와 페더레이션하여 AWS 조직 내 여러 AWS 계정에 대한 Single Sign On을 지원합니다.
  - IAM을 통해 ID 제공업체를 각 AWS 계정에 직접 연결하여 페더레이션된 세분화된 액세스를 지원합니다.
- 페더레이션 ID로 대체되는 IAM 사용자 및 그룹을 식별 및 제거합니다.

애플리케이션 사용자를 위한 단계

- Amazon Cognito를 애플리케이션에 대한 중앙 집중식 ID 제공업체로 사용합니다.

- OpenID Connect 및 OAuth를 사용하여 사용자 지정 애플리케이션을 Amazon Cognito에 통합합니다. 인증을 위해 Amazon Cognito와 같은 다양한 AWS 서비스와 통합할 수 있는 간단한 인터페이스를 제공하는 Amplify 라이브러리를 사용하여 사용자 지정 애플리케이션을 개발할 수 있습니다.

리소스

관련 모범 사례:

- SEC02-BP06 사용자 그룹 및 속성 사용
- SEC03-BP02 최소 권한 액세스 부여
- SEC03-BP06 수명 주기에 따라 액세스 관리

관련 문서:

- AWS의 ID 페더레이션
- IAM의 보안 모범 사례
- AWS Identity and Access Management Best practices
- Getting started with IAM Identity Center delegated administration
- How to use customer managed policies in IAM Identity Center for advanced use cases
- AWS CLI v2: IAM Identity Center credential provider

관련 비디오:

- AWS re:Inforce 2022 - AWS Identity and Access Management (IAM) deep dive
- AWS re:Invent 2022 - Simplify your existing workforce access with IAM Identity Center
- AWS re:Invent 2018: Mastering Identity at Every Layer of the Cake

관련 예제:

- 워크숍: Using AWS IAM Identity Center to achieve strong identity management

관련 도구:

- AWS 보안 컴피턴시 파트너: 자격 증명 및 액세스 관리

- saml2aws

SEC02-BP05 정기적으로 자격 증명 감사 및 교체

자격 증명을 주기적으로 감사하고 교체하여 리소스에 액세스하는 데 자격 증명을 사용할 수 있는 기간을 제한합니다. 장기 자격 증명은 많은 위험을 초래하며 이러한 위험은 장기 자격 증명을 정기적으로 교체하여 줄일 수 있습니다.

원하는 성과: 자격 증명 교체를 구현하여 장기 자격 증명 사용과 관련된 위험을 줄입니다. 자격 증명 교체 정책 미준수를 정기적으로 감사하고 개선합니다.

일반적인 안티 패턴:

- 자격 증명 사용을 감사하지 않습니다.
- 장기 자격 증명을 불필요하게 사용합니다.
- 장기 자격 증명을 사용하고 정기적으로 교체하지 않습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 지침

임시 자격 증명을 사용할 수 없으며 장기 자격 증명이 필요한 경우 자격 증명을 감사하여 정의된 제어(예: 다중 인증(MFA))가 적용되고 정기적으로 교체되며 적절한 액세스 수준을 보유하는지 확인합니다.

올바른 제어 기능이 적용되는지 확인하려면 주기적인 검증(가능한 자동화된 도구 사용)을 실시해야 합니다. 인적 자격 증명의 경우, 사용자가 주기적으로 암호를 변경하고 액세스 키 사용을 중지하며 그 대신 임시 자격 증명을 사용하도록 규정해야 합니다. AWS Identity and Access Management(IAM) 사용자에서 중앙 집중식 ID로 전환하면서 자격 증명 보고서를 생성하여 사용자를 감사할 수 있습니다.

또한 ID 제공업체에서 MFA를 적용하고 모니터링하는 것이 좋습니다. 사용자가 MFA를 구성한 경우 AWS Config 규칙를 설정하거나 AWS Security Hub 보안 표준을 사용할 수 있습니다. 시스템 자격 증명에 대한 임시 자격 증명을 제공하려면 IAM Roles Anywhere를 사용하는 것이 좋습니다. IAM 역할 및 임시 자격 증명을 사용할 수 없는 상황에서는 빈번한 감사 및 교체 액세스 키가 필요합니다.

구현 단계

- 정기적으로 자격 증명 감사: ID 제공업체 및 IAM에 구성된 자격 증명을 감사하면 승인된 자격 증명만 워크로드에 액세스하도록 보장할 수 있습니다. 이러한 자격 증명에는 IAM 사용자, AWS IAM

Identity Center 사용자, Active Directory 사용자 또는 다른 업스트림 ID 제공업체의 사용자가 포함될 수 있지만 이에 국한되지 않습니다. 예를 들어 퇴사하는 사람을 제거하고 더 이상 필요하지 않은 크로스 계정 역할을 제거합니다. IAM 엔터티가 액세스하는 서비스에 대한 권한을 정기적으로 감사하는 프로세스가 있어야 합니다. 이렇게 하면 사용되지 않는 권한을 제거하기 위해 수정해야 하는 정책을 식별하는 데 도움이 됩니다. 자격 증명 보고서 및 AWS Identity and Access Management Access Analyzer를 사용하여 IAM 자격 증명 및 권한을 감사합니다. AWS 환경에서 직접 호출되는 특정 API 직접 호출에 대해 경보를 설정하도록 Amazon CloudWatch를 사용할 수 있습니다. Amazon GuardDuty는 예상치 못한 활동을 알릴 수도 있습니다. 이때 IAM 자격 증명에 대한 지나치게 허용적인 액세스 또는 의도하지 않은 액세스를 나타낼 수 있습니다.

- 정기적으로 자격 증명 교체: 임시 자격 증명을 사용할 수 없는 경우 장기 IAM 액세스 키를 정기적으로 교체합니다(최대 90일마다). 자신도 모르게 액세스 키가 의도치 않게 공개된 경우 자격 증명을 사용하여 리소스에 액세스할 수 있는 기간이 제한됩니다. IAM 사용자의 액세스 키 교체에 대한 자세한 내용은 액세스 키 교체를 참조하세요.
- IMA 권한 검토: AWS 계정의 보안을 개선하려면 모든 IAM 정책을 정기적으로 검토하고 모니터링합니다. 정책이 최소 권한 원칙을 준수하는지 확인합니다.
- IAM 리소스 생성 및 업데이트 자동화 고려: IAM Identity Center는 역할 및 정책 관리와 같은 많은 IAM 작업을 자동화합니다. 또는 AWS CloudFormation을 사용하면 템플릿을 확인하고 버전을 제어할 수 있으므로, 역할 및 정책을 포함한 IAM 리소스 배포를 자동화하여 인적 오류가 발생할 가능성을 줄일 수 있습니다.
- IAM Roles Anywhere를 사용하여 IAM 사용자를 시스템 ID로 교체: IAM Roles Anywhere를 사용하면 온프레미스 서버와 같이 기존에는 불가능했던 영역에서 역할을 사용할 수 있습니다. IAM Roles Anywhere는 신뢰할 수 있는 X.509 인증서를 사용하여 AWS에 인증하고 임시 자격 증명을 받습니다. IAM Roles Anywhere를 사용하면 장기 자격 증명이 온프레미스 환경에 더 이상 저장되지 않으므로 이러한 자격 증명을 교체할 필요가 없습니다. 만료가 가까워지면 X.509 인증서를 모니터링하고 교체해야 합니다.

리소스

관련 모범 사례:

- SEC02-BP02 임시 자격 증명 사용
- SEC02-BP03 안전하게 보안 암호 저장 및 사용

관련 문서:

- AWS Secrets Manager 시작하기

- IAM 모범 사례
- Identity Providers and Federation
- 보안 파트너 솔루션: 액세스 및 액세스 제어
- Temporary Security Credentials
- AWS 계정의 자격 증명 보고서 가져오기

관련 비디오:

- Best Practices for Managing, Retrieving, and Rotating Secrets at Scale
- Managing user permissions at scale with AWS IAM Identity Center
- Mastering identity at every layer of the cake

SEC02-BP06 사용자 그룹 및 속성 사용

사용자 그룹 및 속성에 따라 권한을 정의하면 정책의 수와 복잡성을 줄여 최소 권한 원칙을 보다 간단하게 구현할 수 있습니다. 사용자 그룹을 사용하여 조직에서 수행하는 기능에 따라 한 곳에서 여러 사용자의 권한을 관리할 수 있습니다. 부서, 프로젝트 또는 위치와 같은 속성은 사용자가 유사한 기능을 수행하는 리소스의 다른 하위 세트에 대해 권한 범위 계층을 추가로 제공할 수 있습니다.

원하는 성과: 기능에 따른 권한 변경 사항을 해당 기능을 수행하는 모든 사용자에게 적용할 수 있습니다. 그룹 멤버십과 속성은 사용자 권한을 관리하므로, 개별 사용자 수준에서 권한을 관리할 필요가 줄어듭니다. ID 제공업체(idP)에서 정의한 그룹 및 속성은 AWS 환경에 자동으로 전파됩니다.

일반적인 안티 패턴:

- 개별 사용자의 권한을 관리하고 여러 사용자에 걸쳐 복제합니다.
- 지나치게 개괄적으로 그룹을 정의하여 너무 광범위한 권한을 부여합니다.
- 그룹을 너무 세밀하게 정의하여 멤버십에 대한 중복과 혼란을 야기합니다.
- 속성을 대신 사용할 수 있는 경우 리소스의 하위 세트에서 권한이 중복된 그룹을 사용합니다.
- AWS 환경에 통합되어 있는 표준화된 ID 제공업체를 통해 그룹, 속성 및 멤버십을 관리하지 않습니다.
- AWS IAM Identity Center 세션을 사용할 때 역할 체인 사용

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

AWS 권한은 사용자, 그룹, 역할 또는 리소스와 같은 보안 주체와 관련된 정책이라는 문서에서 정의됩니다. 직무, 워크로드 및 SDLC 환경을 기반으로 권한 할당(그룹, 권한, 계정)을 구성하여 권한 관리를 확장할 수 있습니다. 직원의 경우 액세스 중인 리소스가 아닌 조직에서 사용자가 수행하는 기능을 기반으로 그룹을 정의할 수 있습니다. 예를 들어 WebAppDeveloper 그룹에는 개발 계정 내에서 Amazon CloudFront와 같은 서비스를 구성하기 위한 정책이 연결되어 있을 수 있습니다. `AutomationDeveloper` 그룹에는 `WebAppDeveloper` 그룹과 일부 중첩 권한이 있을 수 있습니다. 두 기능을 수행하는 사용자가 모두 `CloudFrontAccess` 그룹에 속하도록 하는 대신 이러한 일반적인 권한을 별도의 정책으로 캡처하여 두 그룹에 연결할 수 있습니다.

그룹 외에도 속성을 사용하여 액세스 범위를 확대할 수 있습니다. 예를 들어, `WebAppDeveloper` 그룹의 사용자에 대해 Project 속성을 지정하여 프로젝트 관련 리소스에 대한 액세스 범위를 설정할 수 있습니다. 이 기술을 사용하면 권한이 동일할 경우 상이한 프로젝트에서 작업하는 애플리케이션 개발자용 그룹을 서로 다르게 만들 필요가 없습니다. 권한 정책에서 속성을 참조하는 방법은 해당 속성이 페더레이션 프로토콜(예: SAML, OIDC 또는 SCIM)의 일부로 정의되었는지, 사용자 지정 SAML 어설션으로 정의되었는지, IAM Identity Center 내부에 설정되었는지에 관계없이 해당 소스를 기반으로 합니다.

구현 단계

1. 그룹 및 속성을 정의할 위치를 설정합니다.
   a. SEC02-BP04 중앙 집중식 ID 공급업체 사용의 지침에 따라 그룹 및 속성을 정의할 때 ID 제공업체 내부 또는 IAM Identity Center 내부에서 그룹 및 속성을 정의해야 하는지 아니면 특정 계정의 IAM 사용자 그룹을 사용해야 하는지 결정할 수 있습니다.
2. 그룹을 정의합니다.
   a. 기능 및 필요한 액세스 범위를 기준으로 그룹을 결정합니다. 계층 구조 또는 명명 규칙을 사용하여 그룹을 효과적으로 구성하는 것이 좋습니다.
   b. IAM Identity Center 내에 정의하는 경우 그룹을 만들고 권한 집합을 사용하여 원하는 수준의 액세스를 연결합니다.
   c. 외부 ID 제공업체 내에서 정의하는 경우 제공업체가 SCIM 프로토콜을 지원하는지 확인하고, IAM Identity Center 내에서 자동 프로비저닝을 활성화하는 방법을 고려하세요. 이 기능은 제공업체와 IAM Identity Center 간에 그룹 생성, 멤버십 및 삭제를 동기화합니다.
3. 속성을 정의합니다.
   a. 외부 ID 제공업체를 사용하는 경우 SCIM 및 SAML 2.0 프로토콜 모두 기본적으로 특정 속성을 제공합니다. `https://aws.amazon.com/SAML/Attributes/PrincipalTag` 속성 이름을 사용하는 SAML 어설션을 통해 추가 속성을 정의하고 전달할 수 있습니다. 사용자 지정 속성 정의 및 구성에 대한 지침은 ID 제공업체의 설명서를 참조하세요.

b. IAM Identity Center 내부에서 역할을 정의하는 경우 속성 기반 액세스 제어(ABAC) 기능을 활성화하고 필요에 따라 속성을 정의합니다. 조직의 구조 또는 리소스 태그 지정 전략에 맞는 속성을 고려합니다.

IAM Identity Center를 통해 수임된 IAM 역할에서 IAM 역할 체인이 필요한 경우 `source-identity` 및 `principal-tags`와 같은 값은 전파되지 않습니다. 자세한 내용은 액세스 제어를 위한 속성 활성화 및 구성을 참조하세요.

1. 그룹 및 속성을 기반으로 권한 범위를 지정합니다.
   a. 권한 정책에 보안 주체의 속성을 액세스 중인 리소스의 속성과 비교하는 조건을 포함하는 것을 고려해 보세요. 예를 들어 `PrincipalTag` 조건 키의 값이 같은 이름의 `ResourceTag` 키 값과 일치하는 경우에만 리소스에 대한 액세스를 허용하도록 조건을 정의할 수 있습니다.
   b. ABAC 정책을 정의할 때는 ABAC 권한 부여 모범 사례 및 예시의 지침을 따르세요.
   c. 조직의 요구가 변화함에 따라 그룹 및 속성 구조를 정기적으로 검토하고 업데이트하여 최적의 권한 관리를 보장합니다.

리소스

관련 모범 사례:

- SEC02-BP04 중앙 집중식 ID 공급업체 사용
- SEC03-BP02 최소 권한 액세스 부여
- COST02-BP04 그룹 및 역할 구현

관련 문서:

- IAM 모범 사례
- Manage Identities in IAM Identity Center
- What Is ABAC for AWS?
- ABAC In IAM Identity Center
- ABAC 정책 예시

관련 비디오:

- Managing user permissions at scale with AWS IAM Identity Center

- Mastering identity at every layer of the cake

권한 관리

AWS 및 워크로드에 액세스해야 하는 인적 자격 증명 및 시스템 자격 증명에 대한 액세스를 제어하는 권한을 관리합니다. 권한을 통해 누가 어떤 조건에서 무엇에 액세스할 수 있는지를 제어할 수 있습니다. 구체적인 인적 및 머신 자격 증명에 권한을 설정하여 특정 리소스의 특정 서비스 작업에 대한 액세스 권한을 부여합니다. 또한 액세스 권한을 부여하려면 충족되어야 하는 조건을 지정할 수 있습니다.

서로 다른 유형의 리소스에 액세스 권한을 부여하는 방법에는 여러 가지가 있습니다. 그중 하나는 서로 다른 정책 유형을 사용하는 것입니다.

IAM의 자격 증명 기반 정책은 관리형 또는 인라인 형태이며, 사용자, 그룹 또는 역할을 포함한 IAM 자격 증명에 연결됩니다. 이러한 정책으로 자격 증명이 수행할 수 있는 작업(권한)을 지정할 수 있습니다. 자격 증명 기반 정책은 하위 범주로 분류할 수 있습니다.

관리형 정책 - AWS 계정에 속한 다수의 사용자, 그룹 및 역할에게 독립적으로 연결할 수 있는 자격 증명 기반 정책입니다. 두 가지 유형의 관리형 정책이 있습니다.

- AWS 관리형 정책 - AWS에서 생성 및 관리하는 관리형 정책입니다.
- 고객 관리형 정책 - 사용자가 자신의 AWS 계정에서 생성 및 관리하는 관리형 정책입니다. 고객 관리형 정책은 AWS 관리형 정책에 비해 정책을 더 세밀하게 제어할 수 있습니다.

관리형 정책은 권한 적용에 선호되는 방법입니다. 그러나 하나의 사용자, 그룹 또는 역할에 직접 추가하는 인라인 정책을 사용할 수도 있습니다. 인라인 정책은 정책과 자격 증명을 정확히 1대 1 관계로 유지합니다. 자격 증명을 삭제하면 인라인 정책도 삭제됩니다.

대부분의 경우 최소 권한 원칙에 따라 고객 관리형 정책을 직접 생성해야 합니다.

리소스 기반 정책은 리소스에 연결됩니다. 리소스 기반 정책의 예로는 S3 버킷 정책이 있습니다. 이 정책은 리소스와 같거나 다른 계정에 있을 수 있는 보안 주체에 권한을 부여합니다. 리소스 기반 권한을 지원하는 서비스 목록은 IAM으로 작업하는 AWS 서비스를 참조하세요.

권한 경계를 사용하여 관리자가 설정할 수 있는 최대 권한을 설정할 수 있습니다. 이렇게 하면 IAM 역할 생성과 같은 권한을 생성하고 관리할 수 있는 능력을 개발자에게 위임할 수 있지만, 개발자가 생성한 권한을 사용하여 권한을 에스컬레이션할 수 없도록 제한할 수 있습니다.

AWS의 속성 기반 액세스 제어(ABAC)에서는 태그라고 불리는 속성을 기반으로 권한을 부여할 수 있습니다. 이러한 태그는 IAM 보안 주체(사용자 또는 역할) 및 AWS 리소스에 연결될 수 있습니다. 관리자

는 IAM 보안 주체의 속성을 기반으로 권한을 적용하는, 재사용 가능한 IAM 정책을 만들 수 있습니다. 예를 들어, 관리자는 조직의 개발자에게 개발자의 프로젝트 태그와 일치하는 AWS 리소스에 대한 액세스 권한을 부여하기 위해 단일 IAM 정책을 사용할 수 있습니다. 개발자 팀이 프로젝트에 리소스를 추가하면 속성에 따라 권한이 자동으로 적용되어 새로운 리소스마다 정책 업데이트를 할 필요가 없습니다.

조직 서비스 제어 정책(SCP)에서는 조직 또는 조직 단위(OU)의 계정 구성원에 대한 최대 권한을 정의합니다. SCP는 자격 증명 기반 정책이나 리소스 기반 정책을 통해 계정 내 엔터티(사용자나 역할)에 부여하는 권한을 제한하지만, 권한을 부여하지는 않습니다.

세션 정책은 역할 또는 페더레이션 사용자를 맡습니다. AWS CLI 또는 AWS API를 사용할 때 세션 정책을 전달합니다. 세션 정책은 역할 또는 사용자의 자격 증명 기반 정책이 세션에 부여하는 권한을 제한합니다. 세션 정책은 생성된 세션에 대한 권한을 제한하지 않지만, 권한을 부여하지도 않습니다. 자세한 정보는 세션 정책을 참조하세요.

모범 사례

- SEC03-BP01 액세스 요구 사항 정의
- SEC03-BP02 최소 권한 액세스 부여
- SEC03-BP03 긴급 액세스 프로세스 설정
- SEC03-BP04 지속적으로 권한 축소
- SEC03-BP05 조직에 대한 권한 가드레일 정의
- SEC03-BP06 수명 주기에 따라 액세스 관리
- SEC03-BP07 퍼블릭 및 크로스 계정 액세스 분석
- SEC03-BP08 안전하게 조직과 리소스 공유
- SEC03-BP09 안전하게 서드파티와 리소스 공유

SEC03-BP01 액세스 요구 사항 정의

관리자, 최종 사용자 또는 기타 구성 요소별로 워크로드의 각 구성 요소 또는 리소스에 액세스해야 합니다. 각 구성 요소에 대한 액세스 권한 부여 대상을 명확하게 정의하고 적절한 ID 유형과 인증 및 권한 부여 방법을 선택합니다.

일반적인 안티 패턴:

- 애플리케이션에 보안 암호를 하드 코딩 또는 저장합니다.
- 각 사용자에 대한 사용자 지정 권한을 부여합니다.
- 장기 자격 증명을 사용합니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

관리자, 최종 사용자 또는 기타 구성 요소별로 워크로드의 각 구성 요소 또는 리소스에 액세스해야 합니다. 각 구성 요소에 대한 액세스 권한 부여 대상을 명확하게 정의하고 적절한 ID 유형과 인증 및 권한 부여 방법을 선택합니다.

조직 내 AWS 계정에 대한 정기적인 액세스는 페더레이션 액세스 또는 중앙 집중식 ID 제공업체를 사용하여 제공되어야 합니다. 또한 자격 증명 관리를 중앙 집중화하고, 직원 액세스 수명 주기에 대한 AWS 액세스를 통합하기 위해 정립된 사례가 있는지 확인해야 합니다. 예를 들어, 직원이 다른 액세스 수준의 직무로 변경할 경우 해당 그룹 멤버십 또한 변경하여 새로운 액세스 요구 사항을 반영해야 합니다.

비인적 자격 증명에 대한 액세스 요구 사항을 정의할 경우 어떤 애플리케이션 및 구성 요소가 액세스해야 하는지 그리고 어떻게 권한이 부여되는지 결정합니다. 권장되는 접근 방식은 최소 권한 액세스 모델을 통해 구축된 IAM 역할을 사용하는 것입니다. AWS 관리형 정책에서는 대부분의 일반적인 사용 사례를 다루는 미리 정의된 IAM 정책을 제공합니다.

AWS Secrets Manager 및 AWS Systems Manager Parameter Store와 같은 AWS 서비스를 사용하면 IAM 역할을 사용할 수 없는 경우 애플리케이션 또는 워크로드에서 보안 정보를 안전하게 분리할 수 있습니다. Secrets Manager에서 자격 증명의 자동 교체를 설정할 수 있습니다. Systems Manager를 사용하여 스크립트, 명령, SSM 문서, 구성 및 자동화 워크플로의 파라미터를 참조할 수 있으며 이 경우 파라미터를 생성할 때 지정한 고유한 이름을 사용합니다.

AWS IAM Roles Anywhere를 사용하여 AWS 외부에서 실행되는 워크로드에 대한 IAM의 임시 보안 자격 증명을 얻을 수 있습니다. 워크로드는 AWS 애플리케이션에서 AWS 리소스에 액세스하는 데 사용하는 것과 동일한 IAM 정책 및 IAM 역할을 사용할 수 있습니다.

가능한 경우, 장기적이고 정적인 자격 증명보다는 단기적이고 임시적인 자격 증명을 사용하는 것이 좋습니다. 사용자가 프로그래밍 방식 및 장기 자격 증명을 사용해야 하는 시나리오의 경우 액세스 키의 마지막 사용 정보를 사용하여 액세스 키를 교체하고 제거합니다.

사용자가 AWS Management Console 외부에서 AWS와 상호 작용하려면 프로그래밍 방식의 액세스가 필요합니다. 프로그래밍 방식으로 액세스를 부여하는 방법은 AWS에 액세스하는 사용자 유형에 따라 다릅니다.

사용자에게 프로그래밍 방식 액세스 권한을 부여하려면 다음 옵션 중 하나를 선택합니다.

프로그래밍 방식 액세스가 필요한 사용자는 누구인가요?	To	액세스 권한을 부여하는 사용자
작업 인력 ID (IAM Identity Center가 관리하는 사용자)	임시 보안 인증 정보를 사용하여 AWS CLI, AWS SDK 또는 AWS API에 대한 프로그래밍 요청에 서명합니다.	사용하고자 하는 인터페이스에 대한 지침을 따릅니다. • AWS CLI에 대해서는 AWS Command Line Interface 사용 설명서에서 AWS IAM Identity Center을 사용하도록 AWS CLI 구성을 참조하세요. • AWS SDK, 도구,AWS API에 대해서는 AWS SDK 및 도구 참조 가이드에서 IAM Identity Center 인증을 참조하세요.
IAM	임시 보안 인증 정보를 사용하여 AWS CLI, AWS SDK 또는 AWS API에 대한 프로그래밍 요청에 서명합니다.	IAM 사용자 설명서의 AWS 리소스와 함께 임시 보안 인증 정보 사용에 나와 있는 지침을 따르세요.
IAM	(권장되지 않음) 장기 보안 인증 정보를 사용하여 AWS CLI, AWS SDK 또는 AWS API에 대한 프로그래밍 요청에 서명합니다.	사용하고자 하는 인터페이스에 대한 지침을 따릅니다. • AWS CLI에 대해서는 AWS Command Line Interface 사용 설명서에서 IAM 사용자 보안 인증 정보를 사용한 인증을 참조하세요. • AWS SDK와 도구에 대해서는 AWS SDK 및 도구 참조 가이드에서 장기 보안 인증 정보를 사용한 인증을 참조하세요.

프로그래밍 방식 액세스가 필요한 사용자는 누구인가요?	To	액세스 권한을 부여하는 사용자
		• AWS API에 대해서는 IAM 사용자 설명서에서 IAM 사용자의 액세스 키 관리를 참조하세요.

리소스

관련 문서:

- ABAC(속성 기반 액세스 제어)
- AWS IAM Identity Center
- IAM Roles Anywhere
- AWS Managed policies for IAM Identity Center
- AWS IAM 정책 조건
- IAM 사용 사례
- 불필요한 자격 증명 제거
- 정책 작업
- How to control access to AWS resources based on AWS 계정, OU, or organization
- Identify, arrange, and manage secrets easily using enhanced search in AWS Secrets Manager

관련 비디오:

- Become an IAM Policy Master in 60 Minutes or Less
- Separation of Duties, Least Privilege, Delegation, and CI/CD
- Streamlining identity and access management for innovation

SEC03-BP02 최소 권한 액세스 부여

구체적인 조건에서 특정 리소스에 대해 일정한 작업을 수행하기 위해 사용자에게 필요한 액세스 권한만 부여합니다. 개별 사용자에 대한 권한을 정의하는 대신, 그룹 및 자격 증명 속성을 사용하여 대규모

로 권한을 동적으로 설정합니다. 예를 들어 개발자 그룹이 자체 프로젝트에 대한 리소스만 관리하도록 액세스 권한을 허용할 수 있습니다. 이렇게 하면 특정 개발자가 프로젝트에서 빠지게 될 경우 기본 액세스 정책을 변경하지 않고도 해당 개발자의 액세스 권한이 자동으로 해지됩니다.

원하는 성과: 사용자에게 자신의 특정 작업 기능에 필요한 최소 권한만 있습니다. 별도의 AWS 계정을 사용하여 개발자를 프로덕션 환경에서 격리합니다. 개발자가 특정 작업의 프로덕션 환경에 액세스해야 하는 경우 해당 작업을 수행하는 기간 동안에만 제한되고 제어된 액세스 권한이 부여됩니다. 프로덕션 액세스는 해당 개발자가 필요한 작업을 완료한 후 즉시 취소됩니다. 권한에 대한 정기적인 검토를 수행하고 사용자가 역할을 변경하거나 조직을 떠날 때와 같이 권한이 더 이상 필요하지 않은 경우 즉시 권한을 취소합니다. 관리자 권한을 신뢰할 수 있는 소규모 그룹으로 제한하여 위험 노출을 줄입니다. 머신 또는 시스템 계정에는 의도한 작업을 수행하는 데 필요한 최소 권한만 부여합니다.

일반적인 안티 패턴:

- 사용자에게 기본적으로 관리자 권한을 부여합니다.
- 일상 활동에 루트 사용자 계정을 사용합니다.
- 적절한 범위 조정 없이 지나치게 허용적인 정책을 생성합니다.
- 권한 검토는 자주 수행되지 않으므로 권한이 점차 커지는 상황이 발생합니다.
- 환경 격리 또는 권한 관리를 위해 속성 기반 액세스 제어에만 의존합니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 지침

최소 권한의 원칙은 자격 증명에서 특정 작업을 완수하는 데 필요한 최소한의 활동만 수행하도록 허용해야 한다고 규정합니다. 이를 통해 사용 편의성, 효율성 및 보안의 균형을 이룰 수 있습니다. 이 원칙에 따라 운영하면 의도하지 않은 액세스를 제한하고 누가 어떤 리소스에 액세스했는지 추적하는 데 도움이 됩니다. 기본적으로 IAM 사용자와 역할에는 어떠한 권한도 없습니다. 루트 사용자는 기본적으로 전체 액세스 권한을 가지므로 엄격하게 제어 및 모니터링하고 루트 액세스 권한이 필요한 작업에만 사용해야 합니다.

IAM 정책은 IAM 역할 또는 특정 리소스에 대한 권한을 명시적으로 부여하는 데 사용됩니다. 예를 들어, 자격 증명 기반 정책은 IAM 그룹에 연결하는 한편 S3 버킷은 리소스 기반 정책으로 제어할 수 있습니다.

IAM 정책을 생성할 때 AWS에서 액세스를 허용하거나 거부하려면 참이어야 하는 서비스 작업, 리소스 및 조건을 지정할 수 있습니다. AWS에서는 액세스 범위를 줄일 수 있도록 다양한 조건을 지원합니다.

예를 들어 PrincipalOrgID 조건 키를 사용하면 요청자가 AWS 조직에 속하지 않는 경우 요청자의 작업을 거부할 수 있습니다.

또한 CalledVia 조건 키를 사용하여 AWS Lambda 함수를 생성하는 AWS CloudFormation과 같이 사용자를 대신하여 AWS 서비스에서 제출하는 요청을 제어할 수 있습니다. 다양한 정책 유형을 계층화하여 심층 방어를 설정하고 사용자의 권한 전반을 제한할 수 있습니다. 나아가 어떤 조건에서 어떤 권한을 허용할지도 제한할 수도 있습니다. 예를 들어 워크로드 팀이 구축하는 시스템에 대해 자체 IAM 정책을 만들도록 허용하되, 부여할 수 있는 최대 권한을 제한하기 위해 권한 경계를 적용할 경우에만 허용할 수 있습니다.

구현 단계

- 최소 권한 정책 구현: IAM 그룹 및 역할에 최소 권한이 적용된 액세스 정책을 할당하여 사용자별로 정의한 역할 또는 기능을 반영합니다.
- 별도의 AWS 계정을 통해 개발 및 프로덕션 환경 격리: 개발 및 프로덕션 환경에 별도의 AWS 계정을 사용하고 서비스 제어 정책, 리소스 정책 및 자격 증명 정책을 사용하여 이들 간의 액세스를 제어합니다.
- API 사용에 대한 기본 정책: AWS CloudTrail 로그를 검토하여 필요한 권한을 결정하는 한 가지 방법입니다. 이 검토를 사용해 사용자가 AWS 내에서 실제로 수행하는 작업에 맞게 조정된 권한을 만들 수 있습니다. IAM Access Analyzer는 활동을 기반으로 IAM 정책을 자동으로 생성할 수 있습니다. 조직 또는 계정 수준에서 IAM Access Advisor를 사용하여 특정 정책에 대해 마지막 액세스 정보를 추적할 수 있습니다.
- 작업 함수에 AWS 관리형 정책 사용 고려: 세분화된 권한 정책을 생성하기 시작할 때 결제, 데이터베이스 관리자 및 데이터 과학자와 같은 일반적인 작업 역할에 AWS 관리형 정책을 사용하는 것이 도움이 될 수 있습니다. 이러한 정책은 최소 권한 정책을 구현하는 방법을 결정하는 동시에 사용자의 액세스 범위를 좁히는 데 도움이 될 수 있습니다.
- 불필요한 권한 제거: 미사용 IAM 엔터티, 자격 증명 및 권한을 감지하고 제거하여 최소 권한 원칙을 달성합니다. IAM Access Analyzer를 사용하여 외부 및 미사용 액세스를 식별할 수 있으며, IAM Access Analyzer 정책 생성은 권한 정책을 미세 조정하는 데 도움이 될 수 있습니다.
- 프로덕션 환경에 대한 사용자 액세스 제한: 사용자는 유효한 사용 사례가 있는 프로덕션 환경에만 액세스할 수 있어야 합니다. 사용자가 프로덕션 액세스 권한이 필요한 특정 작업을 수행한 후에는 액세스 권한을 해지해야 합니다. 프로덕션 환경에 대한 액세스를 제한하면 예기치 않게 프로덕션에 영향을 미치는 이벤트를 방지하고 의도하지 않은 액세스의 영향 범위를 줄일 수 있습니다.
- 권한 경계 고려: 권한 경계는 ID 기반 정책을 통해 IAM 엔터티에 부여할 수 있는 최대 권한을 설정하는 관리형 정책을 사용하는 기능입니다. 엔터티의 권한 경계는 자격 증명 기반 정책 및 관련 권한 경계 모두에서 허용되는 작업만 수행하도록 허용합니다.

- 속성 기반 액세스 제어 및 리소스 태그를 사용하여 액세스 구체화: 리소스 태그를 사용하는 속성 기반 액세스 제어(ABAC)를 사용하여 지원되는 경우 권한을 구체화할 수 있습니다. 보안 주체 태그를 리소스 태그와 비교하여 정의한 사용자 지정 차원에 따라 액세스를 구체화하는 ABAC 모델을 사용할 수 있습니다. 이 접근 방식은 조직의 권한 정책 수를 간소화하고 줄일 수 있습니다.
  - 보안 주체와 리소스를 모두 AWS 조직에서 소유한 경우에만 ABAC를 액세스 제어에 사용하는 것이 좋습니다. 외부 당사자는 자체 보안 주체 및 리소스에 대해 조직과 동일한 태그 이름 및 값을 사용할 수 있습니다. 외부 당사자 보안 주체 또는 리소스에 대한 액세스 권한을 부여하기 위해 이러한 이름-값 페어에만 의존하는 경우 의도하지 않은 권한을 제공할 수 있습니다.
- AWS Organizations에 서비스 제어 정책 사용: 서비스 제어 정책은 조직의 구성원 계정에 대해 사용 가능한 최대 권한을 중앙에서 제어합니다. 중요한 점은 서비스 제어 정책을 사용하여 구성원 계정의 루트 사용자 권한을 제한할 수 있다는 사실입니다. AWS Organizations를 보강하는 권장 관리 제어 기능을 제공하는 AWS Control Tower를 사용하는 것도 고려해 보세요. Control Tower 내에서 자체 제어 기능을 정의할 수도 있습니다.
- 조직을 위한 사용자 수명 주기 정책 수립: 사용자 수명 주기 정책은 사용자가 AWS에 온보딩하거나, 작업 역할 또는 범위가 변경되거나, 더 이상 AWS에 액세스할 필요가 없을 때 수행할 작업을 정의합니다. 사용자 수명 주기의 각 단계에서 권한 검토를 수행하여 권한이 적절하게 제한되는지 확인하고 권한이 커지는 상황이 없도록 해야 합니다.
- 권한을 검토하고 불필요한 권한을 제거하기 위한 정기 예약 설정: 사용자 액세스를 정기적으로 검토하여 사용자에게 과도하게 허용되는 액세스 권한이 없는지 확인해야 합니다. AWS Config 및 IAM Access Analyzer는 사용자 권한을 감사할 때 유용합니다.
- 작업 역할 매트릭스 설정: 작업 역할 매트릭스는 AWS 기반 내에서 필요한 여러 역할 및 액세스 수준을 시각화합니다. 작업 역할 매트릭스를 사용하여 조직 내 사용자 책임에 따라 권한을 정의하고 분리할 수 있습니다. 개별 사용자나 역할에 권한을 직접 적용하는 대신 그룹을 사용합니다.

리소스

관련 문서:

- 최소 권한 적용
- Permissions boundaries for IAM entities
- Techniques for writing least privilege IAM policies
- IAM Access Analyzer makes it easier to implement least privilege permissions by generating IAM policies based on access activity
- Delegate permission management to developers by using IAM permissions boundaries
- Refining Permissions using last accessed information

- IAM policy types and when to use them
- IAM 정책 시뮬레이터로 IAM 정책 테스트
- Guardrails in AWS Control Tower
- Zero Trust architectures: An AWS perspective
- How to implement the principle of least privilege with CloudFormation StackSets
- ABAC(속성 기반 액세스 제어)
- Reducing policy scope by viewing user activity
- View role access
- Use Tagging to Organize Your Environment and Drive Accountability
- AWS Tagging Strategies
- AWS 리소스에 태그 지정

관련 비디오:

- Next-generation permissions management
- Zero Trust: An AWS perspective

SEC03-BP03 긴급 액세스 프로세스 설정

중앙 집중식 ID 제공업체에 문제가 발생할 경우 예상치 못한 상황에서 워크로드에 긴급 액세스할 수 있는 프로세스를 만드세요.

긴급 상황을 초래할 수 있는 다양한 장애 모드에 대한 프로세스를 설계해야 합니다. 예를 들어, 일반적인 상황에서는 직원이 중앙 집중식 ID 제공업체를 사용하여 클라우드로 페더레이션함으로써(SEC02-BP04) 워크로드를 관리합니다. 그러나 중앙 집중식 ID 제공업체에 장애가 발생하거나 클라우드에서의 페더레이션 구성이 수정되면 직원이 클라우드로 페더레이션하지 못할 수 있습니다. 긴급 액세스 프로세스를 통해 권한 있는 관리자는 대체 수단(예: 대체 형태의 페더레이션 또는 직접 사용자 액세스)을 통해 클라우드 리소스에 액세스하여 페더레이션 구성 또는 워크로드 관련 문제를 해결할 수 있습니다. 긴급 액세스 프로세스는 일반 페더레이션 메커니즘이 복원될 때까지 사용됩니다.

원하는 성과:

- 긴급 상황으로 간주되는 장애 모드를 정의하고 문서화했습니다. 일반적인 상황과 사용자가 워크로드를 관리하기 위해 사용하는 시스템을 고려하세요. 이러한 각 종속성이 어떻게 실패하여 긴급 상황

을 초래할 수 있는지 생각해 보세요. 장애 가능성을 최소화하기 위해 장애 모드를 식별하고 보다 탄력적인 시스템을 설계하는 데 유용한 신뢰성 원칙의 질문 및 모범 사례를 찾아볼 수 있습니다.

- 장애를 긴급 상황으로 확인하기 위해 따라야 하는 단계를 문서화했습니다. 예를 들어 ID 관리자에게 기본 및 대기 ID 제공업체의 상태를 확인하고 둘 다 사용할 수 없는 경우 ID 제공업체 장애에 대한 긴급 이벤트를 선언하도록 요청할 수 있습니다.
- 각 유형의 긴급 또는 장애 모드에 맞는 긴급 액세스 프로세스를 정의했습니다. 구체적으로 설명하면 모든 유형의 긴급 상황에서 일반 프로세스를 과도하게 사용하려는 사용자의 유혹을 줄일 수 있습니다. 긴급 액세스 프로세스는 각 프로세스를 사용해야 하는 상황과 반대로 프로세스를 사용하지 않아야 하는 상황을 설명하고 적용될 수 있는 대체 프로세스를 가리킵니다.
- 프로세스는 빠르고 효율적으로 따를 수 있는 상세한 지침과 플레이북과 함께 잘 문서화되어 있습니다. 긴급 상황은 사용자에게 스트레스를 주는 시간이 될 수 있고 사용자들이 극심한 시간 압박을 받을 수 있다는 점을 기억하세요. 따라서 프로세스를 최대한 단순하게 설계하세요.

일반적인 안티 패턴:

- 긴급 액세스 절차가 제대로 문서화되고 테스트되지 않습니다. 사용자는 긴급 상황에 대비하지 않고 긴급 상황 발생 시 즉흥적인 프로세스를 따릅니다.
- 긴급 액세스 프로세스는 일반 액세스 메커니즘과 동일한 시스템(예: 중앙 집중식 ID 제공업체)을 기반으로 합니다. 즉, 이러한 시스템에 장애가 발생하면 정상 액세스 메커니즘과 긴급 액세스 메커니즘에 모두 영향을 미치고 장애 복구 능력이 저하될 수 있습니다.
- 긴급 액세스 프로세스는 비응급 상황에서 사용됩니다. 예를 들어, 사용자는 파이프라인을 통해 변경 사항을 제출하는 것보다 직접 변경하는 것이 더 쉽기 때문에 긴급 액세스 프로세스를 자주 오용합니다.
- 긴급 액세스 프로세스는 프로세스를 감사하기에 충분한 로그를 생성하지 않거나 프로세스의 오용 가능성에 대해 경고할 수 있도록 로그를 모니터링하지 않습니다.

이 모범 사례 확립의 이점:

- 잘 문서화되고 테스트를 거친 긴급 액세스 프로세스를 갖추면 사용자가 긴급 상황에 대응하고 해결하는 데 걸리는 시간을 줄일 수 있습니다. 이를 통해 가동 중지 시간이 줄어들고 고객에게 제공하는 서비스의 가용성이 향상될 수 있습니다.
- 각 긴급 액세스 요청을 추적하고, 프로세스를 긴급 상황이 아닌 이벤트에 악용하려는 무단 시도를 감지하고 경고를 보낼 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

이 섹션에서는 모든 장애 모드에 적용되는 공통 지침부터 시작하여 장애 모드 유형에 따른 구체적인 지침에 이어 AWS에 배포된 워크로드와 관련된 여러 장애 모드에 대한 긴급 액세스 프로세스를 만드는 방법에 대한 지침을 제공합니다.

모든 장애 모드에 대한 공통 지침

장애 모드에 대한 긴급 액세스 프로세스를 설계할 때는 다음 사항을 고려하세요.

- 프로세스의 사전 조건과 전제 조건, 즉 프로세스를 사용해야 하는 시기와 사용하지 말아야 하는 경우를 문서화하세요. 장애 모드를 자세히 설명하고 다른 관련 시스템의 상태와 같은 가정을 문서화하는 데 도움이 됩니다. 예를 들어 장애 모드 2의 프로세스에서는 ID 제공업체를 사용할 수 있지만 설정된 AWS 구성이 수정되었거나 만료된 것으로 가정합니다.
- 긴급 액세스 프로세스에 필요한 리소스를 미리 생성합니다(SEC10-BP05). 예를 들어, 모든 워크로드 계정에서 IAM 사용자 및 역할과 크로스 계정 IAM 역할을 사용하여 긴급 액세스 AWS 계정을 미리 생성합니다. 이를 통해 긴급 상황 발생 시 이러한 리소스가 준비되어 있고 사용할 수 있는지 확인할 수 있습니다. 리소스를 미리 생성하면 긴급 상황에서 사용할 수 없는 AWS 컨트롤 플레인 API(AWS 리소스 생성 및 수정에 사용됨)에 대한 종속성이 없습니다. 또한 IAM 리소스를 미리 생성하면 최종 일관성으로 인한 잠재적 지연을 고려할 필요가 없습니다.
- 사고 관리 계획의 일부로 긴급 액세스 프로세스를 포함하세요(SEC10-BP02). 긴급 상황이 어떻게 추적되고 동료 팀, 경영진 그리고 해당하는 경우 외부 고객 및 비즈니스 파트너와 같은 조직 내 다른 사람에게 전달되는지 문서화하세요.
- 기존 서비스 요청 워크플로 시스템(있는 경우)에서 긴급 액세스 요청 프로세스를 정의하세요. 일반적으로 이러한 워크플로우 시스템에서는 접수 양식을 만들어 요청에 대한 정보를 수집하고, 워크플로의 각 단계를 통해 요청을 추적하며, 자동 승인 단계와 수동 승인 단계를 모두 추가할 수 있습니다. 각 요청을 인시던트 관리 시스템에서 추적되는 해당 긴급 이벤트와 연관시키세요. 긴급 액세스를 위한 통일된 시스템을 구축하면 단일 시스템에서 이러한 요청을 추적하고, 사용 추세를 분석하며, 프로세스를 개선할 수 있습니다.
- 긴급 액세스 프로세스는 승인된 사용자만 시작할 수 있고 필요에 따라 사용자의 동료 또는 경영진의 승인이 필요한지 확인하세요. 승인 절차는 업무 시간 내외에서 모두 효과적으로 운영되어야 합니다. 1차 승인자를 사용할 수 없고 승인될 때까지 관리망에 에스컬레이션되는 경우 승인 요청을 2차 승인자가 허용할 수 있는 방법을 정의합니다.
- 긴급 액세스 프로세스 및 메커니즘에 대한 강력한 로깅, 모니터링 및 알림 메커니즘을 구현합니다. 모든 긴급 액세스 시도 성공 및 실패에 대한 자세한 감사 로그를 생성합니다. 인시던트 관리 시스템의 진행 중인 긴급 이벤트와 활동의 상관관계를 파악하고, 작업이 예상 기간을 벗어나거나 정상 운영 중에 긴급 액세스 계정이 사용되는 경우 알림을 시작합니다. 긴급 절차는 백도어로 간주될 수 있으므

로 긴급 상황에만 긴급 액세스 계정에 액세스해야 합니다. 보안 정보 및 이벤트 관리(SIEM) 도구 또는 AWS Security Hub와 통합하여 긴급 액세스 기간 동안 모든 활동을 보고하고 감사할 수 있습니다. 정상 작업으로 돌아가면 긴급 액세스 자격 증명을 자동으로 교체하고 관련 팀에 알립니다.

- 긴급 액세스 프로세스를 정기적으로 테스트하여 단계가 명확한지 확인하고 올바른 액세스 수준을 빠르고 효율적으로 부여하세요. 긴급 액세스 프로세스는 인시던트 대응 시뮬레이션(SEC10-BP07) 및 재해 복구 테스트(REL13-BP03)의 일부로 테스트해야 합니다.

장애 모드 1: AWS로 페더레이션에 사용된 ID 제공업체를 사용할 수 없는 경우

SEC02-BP04 중앙 집중식 ID 공급업체 사용에서 설명한 대로, 중앙 집중식 ID 제공업체를 통해 직원을 페데레이션하여 AWS 계정에 액세스 권한을 부여하는 것이 좋습니다. IAM Identity Center를 사용하여 AWS 조직 내 여러 AWS 계정으로 페더레이션하거나 IAM을 사용하여 개별 AWS 계정에 페더레이션할 수 있습니다. 두 경우 모두, 직원이 Single Sign On을 위해 AWS 로그인 엔드포인트로 리디렉션되기 전에 중앙 집중식 ID 제공업체를 통해 인증합니다.

드문 경우지만 중앙 집중식 ID 제공업체를 사용할 수 없는 경우 직원은 AWS 계정에 페더레이션하거나 워크로드를 관리할 수 없습니다. 이 긴급 상황에서 중앙 집중식 ID 제공업체가 다시 온라인 상태가 될 때까지 기다릴 수 없는 중요한 작업을 수행할 수 있도록 소수의 관리자가 AWS 계정에 액세스할 수 있는 긴급 액세스 프로세스를 제공할 수 있습니다. 예를 들어 ID 제공업체를 4시간 동안 사용할 수 없는 경우, 예상치 못한 고객 트래픽 급증에 대처하려면 Production 계정의 Amazon EC2 Auto Scaling 그룹 상한선을 수정해야 합니다. 긴급 관리자는 긴급 액세스 프로세스에 따라 특정 프로덕션 AWS 계정에 대한 액세스 권한을 얻고 필요한 사항을 변경해야 합니다.

긴급 액세스 프로세스는 긴급 액세스 AWS 계정에만 사용되고 긴급 액세스 프로세스를 지원하는 AWS 리소스(예: IAM 역할 및 IAM 사용자)가 있는 미리 생성된 긴급 액세스를 기반으로 합니다. 정상적으로 운영되는 동안에는 아무도 긴급 액세스 계정에 접속해서는 안 되며 이 계정의 오용을 모니터링하고 경고해야 합니다(자세한 내용은 이전 공통 지침 섹션 참조).

긴급 액세스 계정에는 긴급 액세스가 필요한 AWS 계정에서 크로스 계정 역할을 수임할 수 있는 권한이 있는 긴급 액세스 IAM 역할이 있습니다. 이러한 IAM 역할은 긴급 계정의 IAM 역할을 신뢰하는 신뢰 정책으로 미리 생성되고 구성됩니다.

긴급 액세스 프로세스는 다음 방법 중 하나를 사용할 수 있습니다.

- 긴급 액세스 계정에서 관련 강력한 암호 및 MFA 토큰을 사용하여 긴급 관리자용 IAM 사용자 세트를 미리 생성할 수 있습니다. 이러한 IAM 사용자는 IAM 역할을 수임할 권한을 보유하고 있습니다. 이를 통해 이후 긴급 액세스가 필요한 AWS 계정에 크로스 계정 액세스를 허용합니다. 가능한 한 적은 수의 사용자를 생성하고 각 사용자를 한 명의 긴급 관리자에게 할당하는 것이 좋습니다. 긴급 상황 발

생 시 긴급 관리자 사용자는 암호와 MFA 토큰 코드를 사용하여 긴급 액세스 계정에 로그인하고, 긴급 계정에서 긴급 액세스 IAM 역할로 전환하며, 마지막으로 워크로드 계정의 긴급 액세스 IAM 역할로 전환하여 긴급 변경 작업을 수행합니다. 이 접근 방식의 장점은 각 IAM 사용자가 한 명의 긴급 관리자로 할당되며 CloudTrail 이벤트를 검토하여 어떤 사용자가 로그인했는지 알 수 있다는 점입니다. 단점은 연결된 장기 암호와 MFA 토큰을 사용하여 여러 IAM 사용자를 유지 관리해야 한다는 점입니다.

- 긴급 액세스 AWS 계정 루트 사용자를 사용하여 긴급 액세스 계정에 로그인하고 긴급 액세스를 위한 IAM 역할을 수임하며 워크로드 계정에서 크로스 계정 역할을 수임할 수 있습니다. 루트 사용자에게는 강력한 암호와 여러 MFA 토큰을 설정하는 것이 좋습니다. 또한 강력한 인증 및 권한 부여를 시행하는 안전한 엔터프라이즈 자격 증명 볼트에 암호와 MFA 토큰을 저장하는 것이 좋습니다. 암호 및 MFA 토큰 재설정 요소를 보호해야 합니다. 계정의 이메일 주소를 클라우드 보안 관리자가 모니터링하는 이메일 배포 목록으로 설정하고 계정의 전화번호를 보안 관리자가 모니터링하는 공유 전화번호로 설정합니다. 이 접근 방식의 장점은 한 세트의 루트 사용자 자격 증명을 관리할 수 있다는 것입니다. 단점은 공유 사용자이기 때문에 여러 관리자가 루트 사용자로 로그인할 수 있다는 것입니다. 엔터프라이즈 볼트 로그 이벤트를 감사하여 루트 사용자 암호를 체크아웃한 관리자를 식별해야 합니다.

장애 모드 2: AWS의 ID 제공업체 구성이 수정되었거나 만료된 경우

인력 사용자가 AWS 계정에 페더레이션할 수 있도록 하려면 외부 ID 제공업체를 사용하여 IAM Identity Center를 구성하거나 IAM 자격 증명 공급자를 생성할 수 있습니다(SEC02-BP04). 일반적으로 ID 제공업체가 제공한 SAML 메타데이터 XML 문서를 가져와서 이를 구성합니다. 메타데이터 XML 문서에는 ID 제공업체가 SAML 어설션에 서명하는 데 사용하는 개인 키에 해당하는 X.509 인증서가 포함되어 있습니다.

AWS 측의 이러한 구성은 관리자가 실수로 수정하거나 삭제할 수 있습니다. 또 다른 시나리오에서는 AWS로 가져온 X.509 인증서가 만료되고 새 인증서가 포함된 새 메타데이터 XML을 AWS에 아직 가져오지 않은 경우가 있습니다. 두 시나리오 모두 인력 사용자에 대한 AWS 페더레이션을 중단하여 긴급 상황을 초래할 수 있습니다.

이러한 긴급 상황의 경우 ID 관리자에게 페더레이션 문제를 해결할 수 있는 AWS 액세스 권한을 제공할 수 있습니다. 예를 들어, ID 관리자는 긴급 액세스 프로세스를 사용하여 AWS 계정에 긴급 액세스로 로그인하고, Identity Center 관리자 계정의 역할로 전환하며, 페더레이션을 다시 활성화하기 위해 ID 제공업체로부터 최신 SAML 메타데이터 XML 문서를 가져와서 외부 ID 제공업체 구성을 업데이트합니다. 페더레이션이 수정되면 인력 사용자는 계속해서 일반 운영 프로세스를 사용하여 워크로드 계정에 페더레이션합니다.

이전 장애 모드 1에 설명된 접근 방식에 따라 긴급 액세스 프로세스를 만들 수 있습니다. ID 관리자에게 Identity Center 관리자 계정에만 액세스하고 해당 계정의 Identity Center에서 작업을 수행할 수 있는 최소 권한 권한을 부여할 수 있습니다.

장애 모드 3: Identity Center 중단

IAM Identity Center의 예상치 못한 상황이나 AWS 리전 중단이 발생할 경우 AWS Management Console에 임시 액세스를 제공하는 데 사용할 수 있는 구성을 설정하는 것이 좋습니다.

긴급 액세스 프로세스는 ID 제공업체로부터 긴급 계정의 IAM으로 직접 페더레이션을 사용합니다. 프로세스 및 설계 고려 사항에 대한 자세한 내용은 Set up emergency access to the AWS Management Console을 참조하세요.

구현 단계

모든 장애 모드에 대한 공통 단계

- 긴급 액세스 프로세스 AWS 계정 전용 프로세스를 생성합니다. 계정에 필요한 IAM 리소스(예: IAM 역할 또는 IAM 사용자) 및 선택적으로 IAM ID 제공업체를 미리 생성합니다. 또한 긴급 액세스 계정의 해당 IAM 역할과의 신뢰 관계를 사용하여 워크로드 AWS 계정에서 크로스 계정 IAM 역할을 미리 생성합니다. AWS Organizations에서 AWS CloudFormation StackSets를 사용하여 조직의 구성원 계정에서 이러한 리소스를 만들 수 있습니다.

- AWS Organizations 서비스 제어 정책(SCP)을 생성하여 구성원 AWS 계정에서 크로스 계정 IAM 역할의 삭제 및 수정을 거부합니다.
- 긴급 액세스 AWS 계정에 대해 CloudTrail을 활성화하고 로그 컬렉션 AWS 계정에서 중앙 S3 버킷으로 트레일 이벤트를 전송합니다. AWS Control Tower를 사용하여 AWS 다중 계정 환경을 설정하고 관리하는 경우 AWS Control Tower를 사용하여 생성하거나 AWS Control Tower에 등록한 모든 계정에서는 기본적으로 CloudTrail이 활성화되고 전용 로그 아카이브 AWS 계정의 S3 버킷으로 전송됩니다.
- 긴급 IAM 역할별로 콘솔 로그인 및 API 활동과 일치하는 EventBridge 규칙을 생성하여 긴급 액세스 계정의 활동을 모니터링합니다. 인시던트 관리 시스템에서 추적되는 진행 중인 긴급 이벤트 외부에서 활동이 발생하는 경우 보안 운영 센터에 알림을 보냅니다.

장애 모드 1: AWS로 페더레이션에 사용된 ID 제공업체를 사용할 수 없는 경우 및 장애 모드 2: AWS의 ID 제공업체 구성이 수정되었거나 만료된 경우의 추가 단계

- 긴급 액세스를 위해 선택한 메커니즘에 따라 리소스를 미리 생성하세요.

- IAM 사용자 사용: 강력한 암호 및 관련 MFA 디바이스를 사용하여 IAM 사용자를 미리 생성하세요.
- 긴급 계정 루트 사용자 사용: 강력한 암호로 루트 사용자를 구성하고 엔터프라이즈 자격 증명 저장소에 암호를 저장합니다. 여러 물리적 MFA 디바이스를 루트 사용자와 연결하고 긴급 관리자 팀원이 빠르게 액세스할 수 있는 위치에 디바이스를 저장합니다.

장애 모드 3: Identity Center 중단의 추가 단계

- AWS Management Console에 대한 긴급 액세스 설정에서 설명한 대로, 긴급 액세스 AWS 계정에서는 ID 제공업체로부터 직접 SAML 페더레이션을 활성화하도록 IAM ID 제공업체를 생성합니다.
- IdP에 구성원 없이 긴급 운영 그룹을 만드세요.
- 긴급 액세스 계정에서 긴급 운영 그룹에 해당하는 IAM 역할을 생성합니다.

리소스

관련 Well-Architected 모범 사례:

- SEC02-BP04 중앙 집중식 ID 공급업체 사용
- SEC03-BP02 최소 권한 액세스 부여
- SEC10-BP02 인시던트 관리 계획 개발
- SEC10-BP07 게임 데이 진행

관련 문서:

- Set up emergency access to the AWS Management Console
- SAML 2.0 페더레이션 사용자가 AWS Management Console에 액세스할 수 있게 하기
- Break glass access

관련 비디오:

- AWS re:Invent 2022 - Simplify your existing workforce access with IAM Identity Center
- AWS re:Inforce 2022 - AWS Identity and Access Management (IAM) deep dive

관련 예제:

- AWS Break Glass Role

- AWS customer playbook framework
- AWS incident response playbook samples

SEC03-BP04 지속적으로 권한 축소

팀에서 필요한 액세스 권한을 결정할 때 불필요한 권한을 제거하고 최소 권한을 부여하기 위한 검토 프로세스를 수립합니다. 인적 액세스와 시스템 액세스 모두에 대해 사용되지 않는 ID와 권한을 지속적으로 모니터링하고 제거합니다.

원하는 성과: 권한 정책은 최소 권한 원칙을 준수해야 합니다. 직무와 역할이 더 잘 정의됨에 따라 권한 정책을 검토하여 불필요한 권한을 제거해야 합니다. 이 접근 방식은 자격 증명이 의도치 않게 노출되거나 권한 부여 없이 액세스되는 경우 영향 범위를 줄입니다.

일반적인 안티 패턴:

- 사용자에게 기본적으로 관리자 권한을 부여합니다.
- 전체 관리자 권한은 아니지만 과도하게 허용적인 정책을 생성합니다.
- 더 이상 필요하지 않은 권한 정책을 유지합니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

팀과 프로젝트가 이제 막 시작되었으므로 허용 권한 정책을 사용하여 혁신과 민첩성을 확보할 수 있습니다. 예를 들어 개발 또는 테스트 환경에서 개발자에게 광범위한 AWS 서비스에 대한 액세스 권한을 부여할 수 있습니다. 액세스 권한을 지속적으로 평가하고 현재 작업을 완료하는 데 필요한 서비스 및 서비스 작업으로만 액세스 권한을 제한하는 것이 좋습니다. 인적 자격 증명과 시스템 자격 증명 모두에 대해 이 평가가 권장됩니다. 시스템 또는 서비스 계정이라고도 하는 시스템 자격 증명은 AWS에 애플리케이션 또는 서버에 대한 액세스 권한을 부여하는 자격 증명입니다. 지나친 허용 권한은 광범위한 영향을 미치고 잠재적으로 고객 데이터를 노출시킬 수 있으므로 이 액세스 권한은 프로덕션 환경에서 특히 중요합니다.

AWS는 사용되지 않는 사용자, 역할, 권한 및 자격 증명을 식별하는 데 도움이 되는 여러 방법을 제공합니다. AWS는 또한 연결된 액세스 키와 Amazon S3 버킷의 객체와 같은 AWS 리소스에 대한 액세스 권한을 포함하여 IAM 사용자 및 역할의 액세스 활동을 분석하는 데 도움이 될 수 있습니다. AWS Identity and Access Management Access Analyzer 정책 생성은 보안 주체가 상호 작용하는 실제 서비스 및 작업을 기반으로 제한적 권한 정책을 생성하는 데 도움이 될 수 있습니다. ABAC(속성 기반 액세스 제

어)는 권한 정책을 각 사용자에게 직접 연결하는 대신 속성을 사용하여 사용자에게 권한을 제공할 수 있으므로 권한 관리를 간소화하는 데 도움이 됩니다.

구현 단계

- AWS Identity and Access Management Access Analyzer 사용: IAM Access Analyzer를 사용하면 Amazon Simple Storage Service(S3) 버킷 또는 IAM 역할과 같은 조직 및 계정 내 리소스 중 외부 엔터티와 공유되는 리소스를 식별할 수 있습니다.
- IAM Access Analyzer 정책 생성 사용: IAM Access Analyzer 정책 생성을 통해 IAM 사용자 또는 역할의 액세스 활동에 따라 세분화된 권한 정책을 생성할 수 있습니다.
- 프로덕션 전 하위 환경의 권한 테스트: 먼저 덜 중요한 샌드박스 및 개발 환경을 사용하여 IAM Access Analyzer를 사용하여 다양한 작업 기능에 필요한 권한을 테스트하는 것으로 시작합니다. 그런 다음 프로덕션에 적용하기 전에 테스트, 품질 보증 및 스테이징 환경 전반에서 이러한 권한을 점진적으로 더 엄격하게 하고 검증합니다. 서비스 제어 정책(SCP)이 부여된 최대 권한을 제한하여 가드레일을 적용하므로 하위 환경은 처음에 더 완화된 권한을 가질 수 있습니다.
- IAM 사용자 및 역할에 대해 허용되는 기간 및 사용 정책 결정: 마지막으로 액세스한 타임스탬프를 사용하여 사용되지 않는 사용자 및 역할을 식별하고 제거합니다. 서비스 및 작업의 마지막 액세스 정보를 검토하여 특정 사용자 및 역할에 대한 권한을 식별하고 범위를 지정합니다. 예를 들어 마지막 액세스 정보를 사용하면 애플리케이션 역할에 필요한 특정 Amazon S3 작업을 식별하여 그러한 작업으로만 역할의 액세스 권한을 제한할 수 있습니다. 마지막 액세스 정보 기능은 AWS Management Console에서 제공되며, 프로그래밍 방식으로 인프라 워크플로 및 자동화된 도구에 손쉽게 통합할 수 있습니다.
- AWS CloudTrail에서 데이터 이벤트 로깅 고려: 기본적으로 CloudTrail은 Amazon S3 객체 수준 활동(예: `GetObject` 및 `DeleteObject`) 또는 Amazon DynamoDB 테이블 활동(예: `PutItem` 및 `DeleteItem`)과 같은 데이터 이벤트를 로깅하지 않습니다. 특정 Amazon S3 객체 또는 DynamoDB 테이블 항목에 액세스해야 하는 사용자 및 역할을 결정하려면 이러한 이벤트에 대한 로깅을 활성화하는 방법을 고려하세요.

리소스

관련 문서:

- Grant least privilege
- 불필요한 자격 증명 제거
- What is AWS CloudTrail?
- 정책 작업

- DynamoDB의 로깅 및 모니터링
- Amazon S3 버킷 및 객체에 대한 CloudTrail 이벤트 로깅 사용
- AWS 계정의 자격 증명 보고서 가져오기

관련 비디오:

- Become an IAM Policy Master in 60 Minutes or Less
- Separation of Duties, Least Privilege, Delegation, and CI/CD
- AWS re:Inforce 2022 - AWS Identity and Access Management (IAM) deep dive

SEC03-BP05 조직에 대한 권한 가드레일 정의

권한 가드레일을 사용하여 보안 주체에 부여할 수 있는 사용 가능한 권한의 범위를 줄이세요. 권한 정책 평가 체인에는 권한 부여 결정을 내릴 때 보안 주체의 유효 권한을 결정하기 위한 가드레일이 포함됩니다. 계층 기반 접근 방식을 사용하여 가드레일을 정의할 수 있습니다. 가드레일 중 일부는 조직 전체에 광범위하게 적용하고 일부는 임시 액세스 세션에 세부적으로 적용하세요.

원하는 성과: 별도의 AWS 계정을 사용하여 환경을 명확하게 격리할 수 있습니다. 서비스 제어 정책(SCP)은 조직 전체의 권한 가드레일을 정의하는 데 사용됩니다. 더 포괄적인 가드레일은 조직 루트에 가장 근접한 계층 수준에서 설정되고, 더 엄격한 가드레일은 개별 계정 수준에 더 가깝게 설정됩니다.

지원되는 경우 리소스 정책은 보안 주체가 리소스에 액세스하기 위해 충족해야 하는 조건을 정의합니다. 또한, 리소스 정책은 적절한 경우 허용 가능한 작업 집합의 범위를 세분화합니다. 권한 경계는 워크로드 권한을 관리하는 보안 주체에 부여되어 권한 관리를 개별 워크로드 소유자에게 위임합니다.

일반적인 안티 패턴:

- AWS Organization 내에서 구성원 AWS 계정을 생성하지만, 루트 자격 증명에 이용할 수 있는 사용 및 권한을 제한하기 위해 SCP를 사용하지 않습니다.
- 최소 권한을 기준으로 권한을 할당하지만, 부여할 수 있는 최대 권한 집합에는 가드레일을 설정하지 않습니다.
- AWS IAM의 암묵적 거부 기반에 의존하여 권한을 제한하고, 정책이 원치 않는 명시적 허용 권한을 부여하지 않을 것이라고 신뢰합니다.
- 동일한 AWS 계정에서 여러 워크로드 환경을 실행한 후 VPC, 태그 또는 리소스 정책 등의 메커니즘에 의존하여 권한 경계를 적용합니다.

이 모범 사례 확립의 이점: 권한 가드레일은 권한 정책이 허용을 시도하더라도 원하지 않는 권한을 부여할 수 없다는 확신을 심어줍니다. 이를 통해 고려해야 할 권한의 최대 범위를 줄여 권한 정의 및 관리를 단순화할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

계층 기반 접근 방식을 사용하여 조직의 권한 가드레일을 정의하는 것이 좋습니다. 이 접근 방식은 추가 계층이 적용될 때 가능한 최대 권한 집합을 체계적으로 줄입니다. 이렇게 하면 최소 권한 원칙에 따라 액세스 권한을 부여하여 잘못된 정책 구성으로 인해 의도하지 않은 액세스가 발생할 위험을 줄일 수 있습니다.

권한 가드레일을 구축하는 첫 단계는 워크로드와 환경을 별도의 AWS 계정으로 분리하는 것입니다. 한 계정의 보안 주체는 명시적인 권한 없이는 다른 계정의 리소스에 액세스할 수 없습니다. 이는 두 계정이 동일한 AWS 조직 또는 동일한 조직 단위(OU)에 속하더라도 마찬가지입니다. OU를 사용하여 관리하려는 계정을 단일 단위로 그룹화할 수 있습니다.

다음 단계는 조직 구성원 계정 내에서 보안 주체에 부여할 수 있는 최대 권한 집합을 줄이는 것입니다. 이를 위해 OU 또는 계정에 적용 가능한 서비스 제어 정책(SCP)을 사용할 수 있습니다. SCP는 특정 AWS 리전에 대한 액세스 제한과 같은 일반적인 액세스 제어를 시행하여 리소스 삭제를 방지하거나 잠재적으로 위험한 서비스 작업을 비활성화하도록 할 수 있습니다. 조직의 루트에 적용하는 SCP는 구성원 계정에만 영향을 주고 관리 계정에는 영향을 주지 않습니다. SCP는 조직 내 보안 주체만 관리합니다. SCP는 리소스에 액세스하는 조직 외부의 보안 주체를 관리하지 않습니다.

AWS Control Tower를 사용하는 경우 제어 및 랜딩 존을 권한 가드레일 및 다중 계정 환경의 기반으로 활용할 수 있습니다. 랜딩 존은 다양한 워크로드 및 애플리케이션에 대한 별도의 계정을 갖춘 사전 구성된 안전한 기준 환경을 제공합니다. 가드레일은 서비스 제어 정책(SCP), AWS Config 규칙 및 기타 구성의 조합을 통해 보안, 운영 및 규정 준수에 대한 필수 제어를 적용합니다. 그러나 사용자 지정 조직 SCP와 함께 Control Tower 가드레일 및 랜딩 존을 사용하는 경우 충돌을 방지하고 적절한 거버넌스를 보장하기 위해 AWS 설명서에 설명된 모범 사례를 따르는 것이 중요합니다. Control Tower 환경 내에서 SCP, 계정 및 조직 단위(OU)를 관리하는 방법에 대한 자세한 권장 사항은 의 AWS Control Tower guidance for AWS Organizations 섹션을 참조하세요.

이러한 지침을 준수하면 Control Tower의 가드레일, 랜딩 존 및 사용자 지정 SCP를 효과적으로 활용하는 동시에 잠재적 충돌을 완화하고 다중 계정 AWS 환경에 대한 적절한 거버넌스 및 제어를 보장할 수 있습니다.

다음 단계는 IAM 리소스 정책을 사용하여 보안 주체 대행이 충족해야 하는 모든 조건과 함께 관리하는 리소스에서 수행할 수 있는 작업 범위를 지정하는 것입니다. 이는 보안 주체가 조직의 일원인 경우

(PrincipalOrgId 조건 키 사용) 모든 작업을 허용하는 것만큼 광범위할 수도 있고, 특정 IAM 역할의 특정 작업만 허용하는 것처럼 세밀할 수도 있습니다. IAM 역할 신뢰 정책의 조건을 사용하여 비슷한 접근 방식을 취할 수 있습니다. 리소스 또는 역할 신뢰 정책에서 해당 리소스나 정책이 관리하는 역할이나 리소스와 동일한 계정의 보안 주체를 명시적으로 지정하는 경우, 해당 보안 주체에는 같은 권한을 부여하는 연결된 IAM 정책이 필요하지 않습니다. 보안 주체가 리소스와 다른 계정에 있는 경우 보안 주체에 해당 권한을 부여하는 연결된 IAM 정책이 필요합니다.

워크로드 팀은 워크로드에 필요한 권한을 관리하고자 하는 경우가 많습니다. 이를 위해서는 새 IAM 역할 및 권한 정책을 만들어야 할 수도 있습니다. 팀이 IAM 권한 경계에서 부여할 수 있는 최대 권한 범위를 캡처하고, 이 문서를 팀이 IAM 역할 및 권한을 관리하는 데 사용할 수 있는 IAM 역할에 연결할 수 있습니다. 이러한 접근 방식을 통해 IAM 관리 액세스로 인한 위험을 완화하면서 작업을 완료하는 유연성을 제공할 수 있습니다.

보다 세분화된 단계는 권한 있는 액세스 관리(PAM) 및 임시 승격 액세스 관리(TEAM) 기술을 구현하는 것입니다. 보안 주체가 권한 있는 작업을 수행하기 전에 다중 인증을 수행하도록 요구하는 것을 PAM의 예로 들 수 있습니다. 자세한 내용은 Configuring MFA-protected API access를 참조하세요. TEAM에는 보안 주체에 상위 액세스 권한을 부여할 수 있는 승인 및 기간을 관리하는 솔루션이 필요합니다. 한 가지 방법은 액세스 권한이 승격된 IAM 역할에 대한 역할 신뢰 정책에 보안 주체를 임시로 추가하는 것입니다. 또 다른 방법은 정상적인 운영 상태에서 세션 정책을 사용하여 IAM 역할이 보안 주체에 부여한 권한의 범위를 좁힌 다음 승인된 기간에 이 제한을 일시적으로 해제하는 것입니다. AWS 및 일부 파트너가 검증한 솔루션에 대해 자세히 알아보려면 Temporary elevated access를 참조하세요.

구현 단계

1. 워크로드와 환경을 별도의 AWS 계정으로 분리합니다.
2. SCP를 사용하여 조직 구성원 계정 내에서 보안 주체에 부여할 수 있는 최대 권한 집합을 줄이세요.
    a. SCP를 정의하여 조직의 멤버 계정 내 보안 주체에게 부여할 수 있는 최대 권한 세트를 줄일 때 허용 목록 또는 거부 목록 접근 방식 중에서 선택할 수 있습니다. 허용 목록 전략은 허용되는 액세스를 명시적으로 지정하고 다른 모든 액세스를 암시적으로 차단합니다. 거부 목록 전략은 허용되지 않는 액세스를 명시적으로 지정하고 기본적으로 다른 모든 액세스를 허용합니다. 두 전략 모두 장단점이 있으며 적절한 선택은 조직의 특정 요구 사항과 위험 모델에 따라 달라집니다. 자세한 내용은 Strategy for using SCPs를 참조하세요.
    b. 또한 Service control policy examples도 검토하여 SCP를 효과적으로 구성하는 방법을 이해하세요.
3. IAM 리소스 정책을 사용하여 리소스에 허용된 작업의 범위를 좁히고 조건을 지정할 수 있습니다. IAM 역할 신뢰 정책의 조건을 사용하여 역할 수임에 대한 제한을 만드세요.

4. IAM 역할에 IAM 권한 경계를 할당하면 워크로드 팀이 자체 워크로드 IAM 역할 및 권한을 관리하는 데 사용할 수 있습니다.
5. 필요에 따라 PAM 및 TEAM 솔루션을 평가하세요.

리소스

관련 문서:

- Data perimeters on AWS
- Establish permissions guardrails using data perimeters
- 정책 평가 로직

관련 예제:

- Service control policy examples

관련 도구:

- AWS Solution: Temporary Elevated Access Management
- Validated security partner solutions for TEAM

SEC03-BP06 수명 주기에 따라 액세스 관리

조직 내 전체 수명 주기 동안 보안 주체(사용자, 역할, 그룹)에게 부여된 권한을 모니터링하고 조정합니다. 사용자의 역할이 변경됨에 따라 그룹 멤버십을 조정하고, 사용자가 조직을 떠나면 액세스 권한을 제거하세요.

원하는 성과: 조직 내 보안 주체의 수명 주기 전반에 걸쳐 권한을 모니터링하고 조정하여 불필요한 권한이 부여될 위험을 줄입니다. 사용자를 생성할 때 적절한 액세스 권한을 부여합니다. 사용자의 책임이 변경되면 액세스 권한을 수정하고, 사용자가 더 이상 활동하지 않거나 조직을 떠난 경우 액세스 권한을 제거합니다. 사용자, 역할 및 그룹에 대한 변경 사항을 중앙에서 관리합니다. 자동화를 사용하여 변경 사항을 AWS 환경에 전파합니다.

일반적인 안티 패턴:

- 처음에 필요한 것 이상으로 과도하거나 광범위한 액세스 권한을 자격 증명에 미리 부여합니다.

- 시간이 지남에 따라 자격 증명의 역할과 책임이 변경되어도 액세스 권한을 검토하고 조정하지 않습니다.
- 활동하지 않거나 퇴사한 직원 자격 증명의 활성 액세스 권한을 그대로 둡니다. 이로 인해 무단 액세스의 위험이 증가합니다.
- 자격 증명의 수명 주기를 관리하는 데 자동화를 활용하지 않습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

ID(예: 사용자, 역할, 그룹)의 수명 주기 전반에 걸쳐 부여한 액세스 권한을 신중하게 관리하고 조정하세요. 이 수명 주기에는 초기 온보딩 단계, 역할 및 책임의 지속적인 변경, 최종 오프보딩 또는 해고가 포함됩니다. 수명 주기 단계에 따라 액세스를 사전 예방적으로 관리하여 적절한 액세스 수준을 유지하세요. 최소 권한 원칙을 준수하여 과도하거나 불필요한 액세스 권한으로 인해 발생하는 위험을 줄입니다.

AWS 계정 내에서 직접 IAM 사용자의 수명 주기를 관리하거나 직원 ID 제공업체에서 AWS IAM Identity Center로의 페더레이션을 통해 관리할 수 있습니다. IAM 사용자의 경우 AWS 계정 내에서 사용자 및 관련 권한을 생성, 수정 및 삭제할 수 있습니다. 페더레이션 사용자의 경우 IAM Identity Center를 사용하여 System for Cross-domain Identity Management(SCIM) 프로토콜을 통해 조직 ID 제공업체의 사용자 및 그룹 정보를 동기화하여 수명 주기를 관리할 수 있습니다.

SCIM은 다양한 시스템에서 사용자 ID를 자동 프로비저닝하고 프로비저닝 해제하기 위한 개방형 표준 프로토콜입니다. SCIM을 통해 ID 제공업체와 IAM Identity Center를 통합하여 사용자 및 그룹 정보를 자동으로 동기화함으로써 조직의 권한 있는 ID 소스의 변경 사항에 따라 액세스 권한이 부여, 수정 또는 취소되는지 확인할 수 있습니다.

조직 내에서 직원의 역할과 책임이 변경되면 그에 따라 액세스 권한을 조정하세요. IAM Identity Center의 권한 집합을 사용하여 다양한 직무 역할 또는 책임을 정의하고 적절한 IAM 정책 및 권한에 연결할 수 있습니다. 직원의 역할이 변경되면 새로운 담당 업무를 반영하도록 지정된 권한 집합을 업데이트할 수 있습니다. 최소 권한 원칙을 준수하면서 필요한 액세스 권한이 부여되었는지 확인하세요.

구현 단계

1. 초기 액세스 권한 부여, 정기 검토, 오프보딩 절차를 비롯한 액세스 관리 수명 주기 프로세스를 정의하고 문서화합니다.
2. IAM 역할, 그룹 및 권한 경계를 구현하여 액세스 권한을 집합적으로 관리하고 최대 허용 액세스 수준을 적용합니다.

3. IAM Identity Center를 사용하여 사용자 및 그룹 정보의 권한 있는 소스로 페더레이션 ID 제공업체(예: Microsoft Active Directory, Okta, Ping Identity)와 통합합니다.
4. SCIM 프로토콜을 사용하여 ID 제공업체의 사용자 및 그룹 정보를 IAM Identity Center의 자격 증명 스토어로 동기화합니다.
5. 조직 내의 다양한 직무 역할 또는 책임을 나타내는 권한 집합을 IAM Identity Center에서 생성합니다. 각 권한 집합에 적합한 IAM 정책 및 권한을 정의합니다.
6. 정기적인 액세스 검토, 신속한 액세스 취소, 액세스 관리 수명 주기 프로세스의 지속적인 개선을 구현합니다.
7. 직원에게 액세스 관리 모범 사례를 주제로 교육을 제공하고 인식을 제고합니다.

리소스

관련 모범 사례:

- SEC02-BP04 중앙 집중식 ID 공급업체 사용

관련 문서:

- Manage your identity source
- Manage identities in IAM Identity Center
- AWS Identity and Access Management Access Analyzer 사용
- IAM Access Analyzer policy generation

관련 비디오:

- AWS re:Inforce 2023 - Manage temporary elevated access with AWS IAM Identity Center
- AWS re:Invent 2022 - Simplify your existing workforce access with IAM Identity Center
- AWS re:Invent 2022 - Harness power of IAM policies & rein in permissions w/Access Analyzer

SEC03-BP07 퍼블릭 및 크로스 계정 액세스 분석

퍼블릭 및 크로스 계정 액세스를 강조하는 결과를 지속적으로 모니터링합니다. 이 유형의 액세스가 필요한 리소스만 허용하도록 퍼블릭 액세스 및 크로스 계정 액세스를 줄입니다.

원하는 성과: 어떤 AWS 리소스가 누구와 공유되는지 파악합니다. 공유 리소스를 지속적으로 모니터링하고 감사하여 권한이 부여된 보안 주체와만 공유되는지 확인합니다.

일반적인 안티 패턴:

- 공유 리소스의 인벤토리를 유지하지 않습니다.
- 크로스 계정 또는 리소스에 대한 퍼블릭 액세스를 승인하는 프로세스를 따르지 않습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 낮음

구현 가이드

계정이 AWS Organizations에 있는 경우 전체 조직, 특정 조직 단위 또는 개별 계정에 리소스에 대한 액세스 권한을 부여할 수 있습니다. 계정이 조직의 구성원이 아닌 경우 개별 계정과 리소스를 공유할 수 있습니다. 리소스 기반 정책(예: Amazon Simple Storage Service(S3) 버킷 정책)을 사용하거나 다른 계정의 보안 주체가 사용자 계정의 IAM 역할을 수임하도록 허용하여 직접 크로스 계정 액세스 권한을 부여할 수 있습니다. 리소스 정책을 사용할 때 권한이 부여된 보안 주체에게만 액세스 권한이 부여되는지 확인합니다. 공개적으로 사용 가능해야 하는 모든 리소스를 승인하는 프로세스를 정의합니다.

AWS Identity and Access Management Access Analyzer에서는 증명 가능한 보안을 사용하여 계정 외부의 리소스에 대한 모든 액세스 경로를 식별합니다. 리소스 정책을 지속적으로 검토하고, 퍼블릭 또는 크로스 계정 액세스의 조사 결과를 보고하여 잠재적으로 광범위한 액세스를 간단하게 분석할 수 있습니다. 모든 계정에 대한 가시성을 확보할 수 있도록 AWS Organizations에서 IAM Access Analyzer를 구성하는 방법을 고려하세요. 또한 IAM Access Analyzer를 사용하면 리소스 권한을 배포하기 전에 조사 결과를 미리 볼 수 있습니다. 따라서 정책 변경 사항이 리소스에 대해 의도한 퍼블릭 및 크로스 계정 액세스만 부여하는지 확인할 수 있습니다. 다중 계정 액세스를 설계할 때는 신뢰 정책을 사용하여 역할을 수임할 수 있는 사례를 제어할 수 있습니다. 예를 들어 `PrincipalOrgId` 조건 키를 사용하여 AWS Organizations 외부에서 역할을 수임하려는 시도를 거부할 수 있습니다.

AWS Config에서 잘못 구성된 리소스를 보고하고 AWS Config 정책 검사를 통해 퍼블릭 액세스가 구성된 리소스를 감지할 수 있습니다. AWS Control Tower 및 AWS Security Hub와 같은 서비스는 AWS Organizations에서 탐지 제어 및 가드레일을 배포하기만 하면 공개적으로 노출된 리소스를 파악 및 개선할 수 있습니다. 예를 들어 AWS Control Tower에는 Amazon EBS 스냅샷이 AWS 계정에 의해 복원 가능한지 감지할 수 있는 관리형 가드레일이 있습니다.

구현 단계

- AWS Organizations에 대해 AWS Config 사용 고려: AWS Config를 사용하면 AWS Organizations 내의 여러 계정에서 찾은 조사 결과를 위임된 관리자 계정으로 집계할 수 있습니다. 이는 포괄적인 보

기를 제공하고 계정 전체에 AWS Config 규칙 규칙를 배포하여 퍼블릭 액세스가 구성된 리소스를 감지할 수 있습니다.

- AWS Identity and Access Management Access Analyzer 구성: IAM Access Analyzer는 조직 및 계정에서 Amazon S3 버킷 또는 IAM 역할과 같이 외부 엔터티와 공유되는 리소스를 식별하는 데 도움이 됩니다.
- AWS Config에서 자동 수정을 사용하여 Amazon S3 버킷의 퍼블릭 액세스 구성 변경에 대응: Amazon S3 버킷에 대한 퍼블릭 액세스 차단 설정을 자동으로 활성화할 수 있습니다.
- 모니터링 및 알림을 구현하여 Amazon S3 버킷이 공개되었는지 확인: Amazon S3 퍼블릭 액세스 차단이 비활성화된 시점과 Amazon S3 버킷이 공개되었는지 확인하기 위해 모니터링 및 알림 설정을 구현해야 합니다. 또한 AWS Organizations를 사용하는 경우 Amazon S3 퍼블릭 액세스 정책의 변경을 방지하는 서비스 제어 정책을 생성할 수 있습니다. AWS Trusted Advisor는 열기 액세스 권한이 있는 Amazon S3 버킷을 확인합니다. 모든 사용자에게 업로드 또는 삭제 액세스 권한을 부여하는 버킷 권한은 모든 사용자가 버킷 항목을 추가하거나, 수정하거나, 제거할 수 있도록 허용하여 잠재적 보안 문제가 발생하는 원인이 됩니다. Trusted Advisor 검사에서는 명시적인 버킷 권한뿐 아니라 버킷 권한을 재정의할 수 있는 관련 버킷 정책도 확인합니다. 또한 AWS Config를 사용하여 퍼블릭 액세스에 대해 Amazon S3 버킷을 모니터링할 수 있습니다. 자세한 내용은 How to Use AWS Config to Monitor for and Respond to Amazon S3 Buckets Allowing Public Access를 참조하세요.

Amazon S3 버킷에 대한 액세스 제어를 검토할 때 버킷 내에 저장된 데이터의 특성을 고려하는 것이 중요합니다. Amazon Macie는 개인 식별 정보(PII), 보호 대상 건강 정보(PHI), 프라이빗 키 또는 AWS 액세스 키 등의 자격 증명과 같은 민감한 데이터를 검색하고 보호하는 데 도움이 되도록 설계된 서비스입니다.

리소스

관련 문서:

- AWS Identity and Access Management Access Analyzer 사용
- AWS Control Tower controls library
- AWS Foundational Security Best Practices standard
- AWS Config 관리형 규칙
- AWS Trusted Advisor check reference
- Monitoring AWS Trusted Advisor check results with Amazon EventBridge
- Managing AWS Config Rules Across All Accounts in Your Organization
- AWS Config 및 AWS Organizations

- AMI를 Amazon EC2에서 공개적으로 사용할 수 있도록 설정

관련 비디오:

- Best Practices for securing your multi-account environment
- Dive Deep into IAM Access Analyzer

SEC03-BP08 안전하게 조직과 리소스 공유

워크로드 수가 증가함에 따라 해당 워크로드의 리소스에 대한 액세스 권한을 공유하거나 여러 계정에서 리소스를 여러 번 프로비저닝해야 할 수 있습니다. 개발, 테스트 및 프로덕션 환경과 같이 환경을 분리하는 구성이 있을 수 있습니다. 그러나 분리 구성을 적용한다고 해서 안전하게 공유하지 못하는 것은 아닙니다. 겹치는 구성 요소를 공유하면 운영 오버헤드를 줄일 수 있고 동일한 리소스를 여러 번 생성하는 동안 누락된 부분을 추측하지 않고도 일관된 경험을 제공할 수 있습니다.

원하는 성과: 안전한 방법을 사용하여 조직 내에서 리소스를 공유하고 데이터 손실 방지 이니셔티브를 지원하여 의도치 않은 액세스를 최소화합니다. 개별 구성 요소를 관리하는 것에 비해 운영 오버헤드를 줄이고 동일한 구성 요소를 수동으로 여러 번 생성할 때 발생하는 오류를 줄이며 워크로드의 확장성을 높입니다. 다중 장애 지점 시나리오에서 해결 시간을 단축할 수 있고 구성 요소가 더 이상 필요하지 않은 시기를 결정할 때 신뢰성을 높일 수 있습니다. 외부 공유 리소스 분석에 대한 권장 가이드는 SEC03-BP07 퍼블릭 및 크로스 계정 액세스 분석을 참조하세요.

일반적인 안티 패턴:

- 예상치 못한 외부 공유를 지속적으로 모니터링하고 자동으로 알리는 프로세스가 부족합니다.
- 공유해야 할 것과 공유하지 말아야 할 것에 대한 기준이 부족합니다.
- 필요할 때 명시적으로 공유하는 대신 광범위한 공개 정책을 기본으로 설정합니다.
- 필요할 때 겹치는 기본 리소스를 수동으로 생성합니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 지침

액세스 제어 및 패턴을 설계하여 공유 리소스의 소비를 신뢰할 수 있는 엔터티로만 안전하게 관리합니다. 공유 리소스를 모니터링하고 공유 리소스 액세스를 지속적으로 검토하고 부적절하거나 예상치 못한 공유에 대한 알림을 받습니다. 퍼블릭 및 크로스 계정 액세스 분석을 검토하면 필요한 리소스에 대

한 외부 액세스 권한을 줄이도록 거버넌스를 설정하고, 지속적으로 모니터링하고 자동으로 알리는 프로세스를 설정하는 데 도움이 됩니다.

AWS Organizations 내 크로스 계정 공유는 AWS Security Hub, Amazon GuardDuty, AWS Backup과 같은 여러 AWS 서비스에서 지원됩니다. 이러한 서비스를 통해 데이터를 중앙 계정과 공유하거나, 중앙 계정에서 액세스하거나, 중앙 계정에서 리소스 및 데이터를 관리할 수 있습니다. 예를 들어 AWS Security Hub는 조사 결과를 개별 계정에서 모든 조사 결과를 볼 수 있는 중앙 계정으로 전송할 수 있습니다. AWS Backup은 리소스를 백업하고 계정 간에 공유할 수 있습니다. AWS Resource Access Manager(AWS RAM)를 사용하여 기타 공통 리소스(예: VPC 서브넷 및 Transit Gateway Attachment, AWS Network Firewall 또는 Amazon SageMaker AI 파이프라인)를 공유할 수 있습니다.

조직 내에서만 리소스를 공유하도록 계정을 제한하려면 서비스 제어 정책(SCP)을 사용하여 외부 보안 주체에 대한 액세스를 방지합니다. 리소스를 공유할 때는 의도하지 않은 액세스를 방지하도록 자격 증명 기반 제어와 네트워크 제어를 결합하여 조직의 데이터 경계를 생성합니다. 데이터 경계는 신뢰할 수 있는 자격 증명만 예상 네트워크의 신뢰할 수 있는 리소스에 액세스하고 있는지 확인하는 데 도움이 되는 예방 가드레일입니다. 이러한 제어를 통해 어떤 리소스를 공유할 수 있는지에 대한 적절한 제한을 설정하고, 리소스의 허용되지 않은 공유 또는 노출을 방지할 수 있습니다. 예를 들어 데이터 경계의 일부로 VPC 엔드포인트 정책과 `AWS:PrincipalOrgId` 조건을 사용하여 Amazon S3 버킷에 액세스하는 자격 증명이 조직에 속하는지 확인할 수 있습니다. SCP는 서비스 연결 역할 또는 AWS 서비스 보안 주체에 적용되지 않는다는 점에 유의해야 합니다.

Amazon S3를 사용할 때 Amazon S3 버킷에 대한 ACL을 끄고 IAM 정책을 사용하여 액세스 제어를 정의합니다. Amazon CloudFront에서 Amazon S3 오리진에 대한 액세스 권한을 제한하려면 오리진 액세스 ID(OAI)에서 AWS Key Management Service를 통한 서버 측 암호화를 비롯한 추가 기능을 지원하는 오리진 액세스 제어(OAC)로 마이그레이션합니다.

경우에 따라 조직 외부에서 리소스 공유를 허용하거나 리소스에 대한 서드파티 액세스 권한을 부여할 수 있습니다. 외부에서 리소스를 공유하기 위한 권한 관리에 대한 권장 가이드는 권한 관리를 참조하세요.

구현 단계

1. AWS Organizations 사용: AWS Organizations은 사용자가 생성하고 중앙에서 관리하는 단일 조직으로 여러 AWS 계정을 통합하기 위해 사용할 수 있는 계정 관리 서비스입니다. 계정을 조직 단위(OU)로 그룹화하고 각 OU에 서로 다른 정책을 연결하여 예산, 보안 및 규정 준수 요구 사항을 충족할 수 있습니다. 또한 AWS 인공 지능(AI) 및 기계 학습(ML) 서비스가 데이터를 수집 및 저장하는 방법을 제어하고 조직과 통합된 AWS 서비스의 다중 계정 관리를 사용할 수 있습니다.
2. AWS Organizations을 AWS 서비스와 통합: AWS 서비스가 조직의 구성원 계정 내에서 사용자를 대신하여 작업을 수행하는 경우 AWS Organizations은 각 구성원 계정에서 해당 서비스에 대해 IAM

서비스 연결 역할(SLR)을 생성합니다. AWS Management Console, AWS API 또는 AWS CLI를 사용하여 신뢰할 수 있는 액세스를 관리해야 합니다. 신뢰할 수 있는 액세스 활성화에 대한 권장 가이드는 Using AWS Organizations with other AWS services 및 AWS services that you can use with Organizations를 참조하세요.

3. 데이터 경계 설정: 데이터 경계는 신뢰와 소유권의 명확한 경계를 제공합니다. AWS에서는 일반적으로 AWS 리소스에 액세스하는 온프레미스 네트워크 또는 시스템과 함께 AWS Organizations에서 관리하는 AWS 조직으로 표시됩니다. 데이터 경계의 목표는 자격 증명을 신뢰할 수 있고 리소스를 신뢰할 수 있으며 네트워크가 예상되는 경우 액세스가 허용되는지 확인하는 것입니다. 그러나 데이터 경계를 설정하는 것이 모든 상황에 적합한 접근 방식은 아닙니다. 특정 보안 위험 모델 및 요구 사항에 따라 Building a Perimeter on AWS 백서에 설명된 제어 목표를 평가하고 채택합니다. 고유한 위험 태세를 신중하게 고려하고 보안 요구 사항에 맞는 경계 제어를 구현해야 합니다.
4. AWS 서비스에서 리소스 공유 사용 및 적절히 제한: Amazon Machine Image(AMI) 및 AWS Resource Access Manager(AWS RAM)와 같이 많은 AWS 서비스에서는 다른 계정과 리소스를 공유하거나 다른 계정에서 리소스를 대상으로 지정할 수 있습니다. AMI를 공유하기 위해 신뢰할 수 있는 계정을 지정하도록 `ModifyImageAttribute` API를 제한합니다. 신뢰할 수 없는 자격 증명의 액세스를 방지하도록 AWS RAM을 사용할 때 `ram:RequestedAllowsExternalPrincipals` 조건을 지정하여 사용자 조직으로만 공유를 제한합니다. 권장 가이드 및 고려 사항은 Resource sharing and external targets를 참조하세요.
5. AWS RAM을 사용하여 한 계정 내에서 또는 다른 AWS 계정과 안전하게 공유: AWS RAM은 사용자가 생성한 리소스를 사용자 계정 및 다른 AWS 계정의 역할 및 사용자와 안전하게 공유하는 데 도움이 됩니다. 다중 계정 환경에서 AWS RAM을 사용하면 리소스를 한 번 생성하여 다른 계정과 공유할 수 있습니다. 이 접근 방식은 Amazon CloudWatch 및 AWS CloudTrail과의 통합을 통해 일관성, 가시성 및 감사 가능성을 제공하는 동시에 운영 오버헤드를 줄이는 데 도움이 됩니다. 이러한 이점은 크로스 계정 액세스를 사용할 경우 얻을 수 없습니다.

   이전에 리소스 기반 정책을 사용하여 공유한 리소스가 있는 경우 `PromoteResourceShareCreatedFromPolicy` API 또는 이에 상응하는 기능을 사용하여 리소스 공유를 전체 AWS RAM 리소스 공유로 승격할 수 있습니다.

   경우에 따라 리소스를 공유하기 위해 추가 단계를 수행해야 할 수도 있습니다. 예를 들어 암호화된 스냅샷을 공유하려면 AWS KMS 키를 공유해야 합니다.

리소스

관련 모범 사례:

- SEC03-BP07 퍼블릭 및 크로스 계정 액세스 분석

- SEC03-BP09 안전하게 서드파티와 리소스 공유
- SEC05-BP01 네트워크 계층 생성

관련 문서:

- 버킷 소유자가 자신의 소유가 아닌 객체에 크로스 계정 권한 부여
- How to use Trust Policies with IAM
- Building Data Perimeter on AWS
- AWS 리소스에 대한 서드파티 액세스 권한을 부여할 때 외부 ID를 사용하는 방법
- AWS services you can use with AWS Organizations
- Establishing a data perimeter on AWS: Allow only trusted identities to access company data

관련 비디오:

- Granular Access with AWS Resource Access Manager
- Securing your data perimeter with VPC endpoints
- Establishing a data perimeter on AWS

관련 도구:

- Data Perimeter Policy Examples

SEC03-BP09 안전하게 서드파티와 리소스 공유

클라우드 환경의 보안은 조직에 국한되지 않습니다. 조직은 서드파티를 이용하여 데이터의 일부를 관리할 수 있습니다. 서드파티 관리형 시스템에 대한 권한 관리는 임시 자격 증명으로 최소 권한 원칙을 사용하여 적시 액세스 방식을 따라야 합니다. 서드파티와 긴밀히 협력하면 영향 범위와 의도치 않은 액세스의 위험을 함께 줄일 수 있습니다.

원하는 성과: 액세스 키 및 보안 키와 같은 장기 AWS Identity and Access Management(IAM) 자격 증명은 오용될 경우 보안 위험이 있으므로 사용하지 않습니다. 대신, IAM 역할과 임시 자격 증명을 사용하여 보안 태세를 개선하고 장기 자격 증명 관리의 운영 오버헤드를 최소화합니다. 서드파티 액세스 권한을 부여할 때 IAM 신뢰 정책의 외부 ID로 Universally Unique Identifier(UUID)를 사용하고 최소 권한 액세스를 보장하기 위해 IAM 정책을 제어하에 역할에 연결해 둡니다. 외부에서 공유되는 리소스에 대한 권장 가이드는 SEC03-BP07 퍼블릭 및 크로스 계정 액세스 분석을 참조하세요.

일반적인 안티 패턴:

- 조건 없이 기본 IAM 신뢰 정책을 사용합니다.
- 장기 IAM 자격 증명 및 액세스 키를 사용합니다.
- 외부 ID를 재사용합니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 지침

AWS Organizations 외부에서 리소스 공유를 허용하거나 서드파티에 계정에 대한 액세스 권한을 부여할 수 있습니다. 예를 들어, 서드파티가 계정 내 리소스에 액세스해야 하는 모니터링 솔루션을 제공할 수 있습니다. 이 경우, 서드파티에만 필요한 권한이 포함된 IAM 크로스 계정 역할을 생성합니다. 또한 외부 ID 조건을 사용하여 신뢰 정책을 정의합니다. 외부 ID를 사용하는 경우 사용자 또는 서드파티는 각 고객, 서드파티 또는 테넌시용으로 고유한 ID를 생성할 수 있습니다. 고유한 ID는 생성된 후에는 사용자 외에는 누구도 제어해서는 안 됩니다. 서드파티는 안전하고 감사 가능하며 재현 가능한 방식으로 외부 ID를 고객과 연결하는 프로세스를 구현해야 합니다.

또한 IAM Roles Anywhere를 사용하여 AWS API를 사용하는 AWS 외부 애플리케이션에 대한 IAM 역할을 관리할 수 있습니다.

서드파티가 더 이상 환경에 액세스할 필요가 없으면 역할을 제거합니다. 장기 자격 증명을 서드파티에 제공하지 않아야 합니다. 다른 AWS 계정과 워크로드를 공유할 수 있도록 허용하는 AWS Well-Architected Tool 및 소유한 AWS 리소스를 다른 계정과 안전하게 공유할 수 있도록 지원하는 AWS Resource Access Manager 등 공유를 지원하는 다른 AWS 서비스를 지속적으로 파악하세요.

구현 단계

1. 크로스 계정 역할을 사용하여 외부 계정에 대한 액세스 권한을 제공합니다. 크로스 계정 역할을 사용하면 고객을 지원하기 위해 서드파티 및 외부 계정에서 저장하는 민감한 정보의 양을 줄일 수 있습니다. 교차 계정 역할을 사용하면 AWS 파트너 또는 조직의 다른 계정과 같은 서드파티에 계정의 AWS 리소스에 대한 액세스 권한을 안전하게 부여하는 동시에 해당 액세스를 관리 및 감사하는 기능을 유지할 수 있습니다. 서드파티는 하이브리드 인프라에서 서비스를 제공하거나 오프사이트 위치로 데이터를 가져올 수 있습니다. IAM Roles Anywhere를 사용하면 서드파티 워크로드에서 AWS 워크로드와 안전하게 상호 작용하고 추가적으로 장기 자격 증명의 필요성을 줄일 수 있습니다.

   외부 계정 액세스를 제공하기 위해 장기 자격 증명 또는 사용자와 연결된 액세스 키를 사용해서는 안 됩니다. 대신 크로스 계정 역할을 사용하여 크로스 계정 액세스를 제공합니다.

2. 실사를 수행하고 서드파티 SaaS 공급자에 대한 보안 액세스를 보장합니다. 서드파티 SaaS 공급자와 리소스를 공유할 때 철저한 실사를 수행하여 서드파티가 AWS 리소스에 액세스할 수 있는 안전하고 책임 있는 접근 방식을 갖추도록 하세요. 서드파티의 공동 책임 모델을 평가하여 서드파티가 제공하는 보안 조치와 나의 책임에 해당하는 조치를 파악합니다. SaaS 공급자가 외부 ID 및 최소 권한 액세스 원칙 사용을 포함하여 리소스에 액세스하기 위한 안전하고 감사 가능한 프로세스를 갖추고 있는지 확인합니다. 외부 ID를 사용하면 혼동된 대리자 문제를 해결하는 데 도움이 됩니다.

   보안 제어를 구현하여 서드파티 SaaS 공급자에 액세스 권한을 부여할 때 보안 액세스 및 최소 권한 원칙을 준수하도록 합니다. 여기에는 외부 ID, Universally Unique Identifiers(UUID) 및 엄격하게 필요한 것으로만 액세스를 제한하는 IAM 신뢰 정책의 사용이 포함될 수 있습니다. SaaS 공급자와 긴밀히 협력하여 안전한 액세스 메커니즘을 설정하고 AWS 리소스에 대한 SaaS 공급자의 액세스를 정기적으로 검토하며 감사를 수행하여 보안 요구 사항을 준수하는지 확인합니다.

3. 고객이 제공한 장기 자격 증명을 사용 중단합니다. 장기 자격 증명 사용을 중단하고 크로스 계정 역할 또는 IAM Roles Anywhere를 사용합니다. 장기 자격 증명을 사용해야 하는 경우 역할 기반 액세스로의 마이그레이션 계획을 수립합니다. 키 관리에 대한 자세한 내용은 Identity management를 참조하세요. 또한 AWS 계정 팀 및 서드파티와 협력하여 위험 완화 런북을 수립합니다. 보안 인시던트의 잠재적 영향에 대한 대응 및 완화에 대한 권장 가이드는 Incident response를 참조하세요.

4. 설정에 권장 가이드가 있는지 또는 자동화되어 있는지 확인합니다. 외부 ID는 보안 암호로 취급되지는 않지만 전화번호, 이름, 계정 ID와 같이 쉽게 추측할 수 있는 값이 아니어야 합니다. 설정을 가장할 목적으로 외부 ID를 변경할 수 없도록 외부 ID를 읽기 전용 필드로 만듭니다.

   사용자 또는 서드파티가 외부 ID를 생성할 수 있습니다. ID 생성 담당자를 결정하는 프로세스를 정의합니다. 외부 ID를 생성하는 엔터티에 관계없이 서드파티는 고객 간에 고유성과 형식을 일관되게 적용합니다.

   사용자 계정의 크로스 계정 액세스를 위해 생성된 정책은 최소 권한 원칙을 따라야 합니다. 서드파티는 역할 정책 문서를 제공하거나 AWS CloudFormation 템플릿 또는 이에 상응하는 템플릿을 사용하는 자동화된 설정 메커니즘을 제공해야 합니다. 이는 수동 정책 생성과 관련된 오류가 발생할 가능성을 줄이고 감사 가능한 트레일을 제공합니다. AWS CloudFormation 템플릿을 사용하여 교차 계정 역할을 생성하는 방법에 대한 자세한 내용은 Cross-Account Roles을 참조하세요.

   서드파티는 자동화되고 감사 가능한 설정 메커니즘을 제공해야 합니다. 그러나 필요한 액세스를 설명하는 역할 정책 문서를 사용하여 역할 설정을 자동화해야 합니다. AWS CloudFormation 템플릿 또는 이에 상응하는 템플릿을 사용하여 감사 방식의 일환으로 드리프트 감지를 사용하여 변경 사항을 모니터링해야 합니다.

5. 변경 사항을 고려합니다. 계정 구조, 서드파티에 대한 요구 사항 또는 제공되는 서비스 오퍼링이 변경될 수 있습니다. 변경 사항과 장애를 예상하고 적절한 인력, 프로세스 및 기술을 사용하여 그에 따

라 계획을 수립해야 합니다. 사용자가 제공하는 액세스 수준을 정기적으로 감사하고 감지 방법을 구현하여 예상치 못한 변경 사항을 알립니다. 외부 ID의 역할 및 데이터 스토어의 사용을 모니터링하고 감사합니다. 예상치 못한 변경 사항 또는 액세스 패턴의 결과로 일시적으로 또는 영구적으로 서드파티 액세스를 취소할 준비가 되어 있어야 합니다. 또한 수행하는 데 걸리는 시간, 관련된 담당자, 비용, 다른 리소스에 미치는 영향을 포함하여 취소 작업에 미치는 영향을 측정합니다.

탐지 방법에 대한 권장 가이드는 탐지 모범 사례를 참조하세요.

리소스

관련 모범 사례:

- SEC02-BP02 임시 자격 증명 사용
- SEC03-BP05 조직에 대한 권한 가드레일 정의
- SEC03-BP06 수명 주기에 따라 액세스 관리
- SEC03-BP07 퍼블릭 및 크로스 계정 액세스 분석
- SEC04 Detection

관련 문서:

- 버킷 소유자가 자신의 소유가 아닌 객체에 크로스 계정 권한 부여
- How to use trust policies with IAM roles
- Delegate access across AWS 계정 using IAM roles
- IAM을 사용하여 다른 AWS 계정의 리소스에 액세스하려면 어떻게 해야 합니까?
- IAM의 보안 모범 사례
- Cross-account policy evaluation logic
- How to use an external ID when granting access to your AWS resources to a third party
- Collecting Information from AWS CloudFormation Resources Created in External Accounts with Custom Resources
- Securely Using External ID for Accessing AWS Accounts Owned by Others
- Extend IAM roles to workloads outside of IAM with IAM Roles Anywhere

관련 비디오:

- How do I allow users or roles in a separate AWS 계정 access to my AWS 계정?
- AWS re:Invent 2018: Become an IAM Policy Master in 60 Minutes or Less
- AWS Knowledge Center Live: IAM Best Practices and Design Decisions

관련 예제:

- Amazon DynamoDB에 대한 크로스 계정 액세스 구성
- AWS STS Network Query Tool

탐지

탐지는 예기치 않거나 원치 않는 구성 변경 탐지와 예기치 않은 동작 탐지와 같은 두 부분으로 구성됩니다. 첫 번째 탐지는 애플리케이션 전달 수명 주기의 여러 위치에서 발생할 수 있습니다. 코드형 인프라(예: CloudFormation 템플릿)를 사용하면 CI/CD 파이프라인 또는 소스 제어에 검사를 구현하여 워크로드를 배포하기 전에 원치 않는 구성을 검사할 수 있습니다. 그런 다음 비프로덕션 및 프로덕션 환경에 워크로드를 배포할 때 기본 AWS, 오픈 소스 또는 AWS 파트너 도구를 사용하여 구성을 검사할 수 있습니다. 이러한 검사는 보안 원칙 또는 모범 사례를 준수하지 않는 구성을 찾거나 테스트된 구성 및 배포된 구성 사이에서 수행된 변경 사항을 찾기 위해 수행할 수 있습니다. 실행 중인 애플리케이션에 대해 알려진 배포 또는 자동화된 규모 조정 이벤트 외에 구성이 원치 않는 방식으로 변경되었는지 검사할 수 있습니다.

두 번째 탐지에 해당하는 예기치 않은 동작의 경우 도구를 사용할 수 있습니다. 또는 특정 API 직접 호출 유형이 증가하면 알림을 받을 수 있습니다. Amazon GuardDuty를 사용하면 예기치 않은 잠재적인 무단 또는 악성 활동이 AWS 계정에서 발생하면 알림을 받을 수 있습니다. 또한 워크로드에서 사용되리라 예상되지 않는 API 직접 호출의 변경과 보안 태세를 변경하는 API 직접 호출을 명시적으로 모니터링해야 합니다.

탐지를 통해 잠재적인 보안 구성 오류, 위협 또는 예기치 않은 동작을 식별할 수 있습니다. 이것은 보안 수명 주기의 핵심 부분으로서 품질 프로세스, 법률 또는 규정 준수 의무, 위협 식별 및 대응 과정을 지원하는 데 사용됩니다. 탐지 메커니즘에는 여러 가지 유형이 있습니다. 예를 들어 워크로드 로그를 분석하여 사용 중인 악용 항목을 알아낼 수 있습니다. 워크로드와 관련된 탐지 메커니즘을 정기적으로 검토하여 사내외 정책과 요구 사항에 부합하는지 확인해야 합니다. 자동 알림은 팀이나 도구가 조사에 착수할 수 있도록 정의된 조건을 기반으로 설정해야 합니다. 이러한 메커니즘은 조직 내에서 변칙적 활동 범위를 식별하고 파악하는 데 도움이 되는 중요한 대응 요소입니다.

AWS에는 탐지 메커니즘을 다룰 때 사용할 수 있는 방식이 아주 많습니다. 다음 섹션에서는 이러한 접근 방식을 사용하는 방법을 설명합니다.

모범 사례

- SEC04-BP01 서비스 및 애플리케이션 로깅 구성
- SEC04-BP02 표준화된 위치에서 로그, 조사 결과 및 지표 캡처
- SEC04-BP03 보안 알림 보강 및 상관관계 지정
- SEC04-BP04 규정 미준수 리소스 관련 문제 해결 시작

SEC04-BP01 서비스 및 애플리케이션 로깅 구성

서비스 및 애플리케이션의 보안 이벤트 로그를 유지합니다. 이는 감사, 조사 및 운영 사용 사례에 대한 보안의 기본 원칙이며 거버넌스, 위험 및 규정 준수(GRC) 표준, 정책 및 절차를 기반으로 하는 공통 보안 요구 사항입니다.

원하는 성과: 조직은 AWS 서비스 및 애플리케이션에서 보안 인시던트 대응과 같은 내부 프로세스 또는 의무를 이행해야 할 때 적시에 안정적이고 일관되게 보안 이벤트 로그를 검색할 수 있어야 합니다. 더 나은 운영 결과를 위해 로그를 중앙 집중화하는 것이 좋습니다.

일반적인 안티 패턴:

- 로그는 영구적으로 저장되거나 너무 빨리 삭제됩니다.
- 누구나 로그에 액세스할 수 있습니다.
- 로그 거버넌스 및 사용을 위해 수동 프로세스에 전적으로 의존합니다.
- 필요한 경우를 대비하여 모든 유형의 로그를 저장합니다.
- 필요한 경우에만 로그 무결성을 확인합니다.

이 모범 사례 확립의 이점: 보안 인시던트에 대한 근본 원인 분석(RCA) 메커니즘과 거버넌스, 위험 및 규정 준수 의무에 대한 증거 소스를 구현합니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

요구 사항에 따른 보안 조사 또는 기타 사용 사례 중에 관련 로그를 검토하여 인시던트의 전체 범위와 타임라인을 기록하고 이해할 수 있어야 합니다. 관심 있는 특정 작업이 발생했음을 나타내는 알림 생성에도 로그가 필요합니다. 쿼리 및 검색 메커니즘을 선택, 활성화, 저장 및 설정하고 경보를 설정하는 것이 중요합니다.

구현 단계

- 로그 소스를 선택하고 사용합니다. 보안 조사에 앞서 관련 로그를 캡처하여 AWS 계정의 활동을 소급하여 재구성해야 합니다. 워크로드와 관련된 로그 소스를 선택합니다.

  로그 소스 선택 기준은 비즈니스에 필요한 사용 사례를 기반으로 해야 합니다. AWS CloudTrail 또는 AWS Organizations 트레일을 사용하여 각 AWS 계정에 대한 트레일을 설정하고 이에 대한 Amazon S3 버킷을 구성합니다.

AWS CloudTrail은 AWS 서비스 활동을 캡처하는 AWS 계정에 대해 수행된 API 직접 호출을 추적하는 로깅 서비스입니다. AWS Management Console, AWS CLI 또는 AWS SDK를 사용하여 CloudTrail 이벤트 기록을 통해 검색할 수 있도록 기본적으로 관리 이벤트의 90일 보존으로 활성화됩니다. 데이터 이벤트를 더 오래 보존하고 가시성을 확보하려면 CloudTrail 트레일을 생성하고 이를 Amazon S3 버킷과 연결하고 선택적으로 Amazon CloudWatch 로그 그룹과 연결합니다. 또는 최대 7년 동안 CloudTrail 로그를 유지하고 SQL 기반 쿼리 기능을 제공하는 CloudTrail Lake를 생성할 수 있습니다.

AWS는 VPC를 사용하는 고객이 각각 VPC 흐름 로그 및 Amazon Route 53 Resolver 쿼리 로그를 사용하여 네트워크 트래픽 및 DNS 로그를 활성화하고 Amazon S3 버킷 또는 CloudWatch 로그 그룹으로 스트리밍할 것을 권장합니다. VPC, 서브넷 또는 네트워크 인터페이스에 대한 VPC 흐름 로그를 생성할 수 있습니다. VPC 흐름 로그의 경우 흐름 로그를 사용하는 방법과 위치를 선택하여 비용을 절감할 수 있습니다.

AWS CloudTrail 로그, VPC 흐름 로그 및 Route 53 Resolver 쿼리 로그는 AWS에서 보안 조사를 지원하는 기본 로깅 소스입니다. 또한 Amazon Security Lake를 사용하여 쿼리에 사용할 준비가 된 Apache Parquet 형식 및 OCSF(Open Cybersecurity Schema Framework)로 이 로그 데이터를 수집, 정규화 및 저장할 수 있습니다. Security Lake는 다른 AWS 로그와 서드파티 소스의 로그도 지원합니다.

AWS 서비스는 Elastic Load Balancing 로그, AWS WAF 로그, AWS Config 레코더 로그, Amazon GuardDuty 조사 결과, Amazon Elastic Kubernetes Service(Amazon EKS) 감사 로그, Amazon EC2 인스턴스 운영 체제 및 애플리케이션 로그와 같은 기본 로그 소스에서 캡처하지 않는 로그를 생성할 수 있습니다. 로깅 및 모니터링 옵션의 전체 목록은 AWS Security Incident Response Guide의 Appendix A: Cloud capability definitions – Logging and Events를 참조하세요.

- 각 AWS 서비스 및 애플리케이션에 대한 로깅 기능 연구: 각 AWS 서비스 및 애플리케이션은 각각 고유한 보존 및 수명 주기 기능이 있는 로그 스토리지 옵션을 제공합니다. 가장 일반적인 두 가지 로그 스토리지 서비스는 Amazon Simple Storage Service(S3) 및 Amazon CloudWatch입니다. 보존 기간이 긴 경우 비용 효율성과 유연한 수명 주기 기능을 위해 Amazon S3를 사용하는 것이 좋습니다. 기본 로깅 옵션이 Amazon CloudWatch Logs인 경우 액세스 빈도가 낮은 로그를 Amazon S3에 아카이브하는 방법을 고려해야 합니다.
- 로그 스토리지 선택: 로그 스토리지의 선택은 일반적으로 사용하는 쿼리 도구, 보존 기능, 친숙도 및 비용과 관련이 있습니다. 로그 스토리지의 기본 옵션은 Amazon S3 버킷 또는 CloudWatch 로그 그룹입니다.

Amazon S3 버킷은 선택적 수명 주기 정책을 통해 비용 효율적이고 내구성이 뛰어난 스토리지를 제공합니다. Amazon S3 버킷에 저장된 로그는 Amazon Athena와 같은 서비스를 사용하여 쿼리할 수 있습니다.

CloudWatch 로그 그룹은 CloudWatch 로그 인사이트를 통해 내구성이 뛰어난 스토리지와 기본 제공 쿼리 기능을 제공합니다.

- 적절한 로그 보존 식별: Amazon S3 버킷 또는 CloudWatch 로그 그룹을 사용하여 로그를 저장하는 경우 각 로그 소스에 적절한 수명 주기를 설정하여 저장 및 검색 비용을 최적화해야 합니다. 고객은 일반적으로 3개월에서 1년 사이의 로그를 쉽게 쿼리할 수 있으며 최대 7년 동안 보존할 수 있습니다. 가용성 및 보존에 대한 선택은 보안 요구 사항과 법적, 규제 및 비즈니스 의무의 조합과 일치해야 합니다.
- 적절한 보존 및 수명 주기 정책으로 각 AWS 서비스 및 애플리케이션에 대한 로깅 사용: 조직의 각 AWS 서비스 또는 애플리케이션에 대해 특정 로깅 구성 지침을 찾습니다.
  - AWS CloudTrail 트레일 구성
  - VPC 흐름 로그 구성
  - Amazon GuardDuty 조사 결과 내보내기 구성
  - AWS Config 레코딩 구성
  - AWS WAF 웹 ACL 트래픽 구성
  - AWS Network Firewall네트워크 트래픽 로그 구성
  - Elastic Load Balancing 액세스 로그 구성
  - Amazon Route 53 Resolver 쿼리 로그 구성
  - Amazon RDS 로그 구성
  - Amazon EKS 컨트롤 플레인 로그 구성
  - Amazon EC2 인스턴스 및 온프레미스 서버에 대해 Amazon CloudWatch 에이전트 구성
- 로그에 대한 쿼리 메커니즘 선택 및 구현: 로그 쿼리의 경우 CloudWatch 로그 그룹에 저장된 데이터에는 CloudWatch 로그 인사이트를 사용할 수 있고 Amazon S3에 저장된 데이터에는 Amazon Athena 및 Amazon OpenSearch Service를 사용할 수 있습니다. 보안 정보 및 이벤트 관리(SIEM) 서비스와 같은 서드파티 쿼리 도구를 사용할 수도 있습니다.

  로그 쿼리 도구를 선택하는 프로세스는 보안 작업의 인력, 프로세스 및 기술 측면을 고려해야 합니다. 운영, 비즈니스 및 보안 요구 사항을 충족하고 장기적으로 액세스 및 유지 관리 가능한 도구를 선택합니다. 로그 쿼리 도구는 스캔할 로그 수가 도구의 한도 내에서 유지될 때 최적으로 작동합니다. 비용이나 기술적 제약으로 인해 여러 쿼리 도구를 사용하는 것이 일반적입니다.

예를 들어 서드파티 보안 정보 및 이벤트 관리(SIEM) 도구를 사용하여 지난 90일 데이터에 대한 쿼리를 수행할 수 있지만, SIEM의 로그 수집 비용으로 인해 90일 이후 데이터에 대한 쿼리를 수행할 때는 Athena를 사용합니다. 구현에 관계없이, 특히 보안 이벤트 조사 중에 운영 효율성을 극대화하는 데 필요한 도구의 수를 최소화하는 접근 방식인지 확인합니다.

- 알림에 로그 사용: AWS는 여러 보안 서비스를 통해 알림을 제공합니다.
  - AWS Config의 경우 AWS 리소스 구성을 모니터링 및 기록하며, 원하는 구성을 기준으로 자동으로 평가하고 수정할 수 있습니다.
  - Amazon GuardDuty는 악성 활동 및 무단 행위를 지속적으로 모니터링하여 AWS 계정 계정 및 워크로드를 보호하는 위협 탐지 서비스입니다. GuardDuty는 AWS CloudTrail 관리 및 데이터 이벤트, DNS 로그, VPC 흐름 로그 및 Amazon EKS 감사 로그와 같은 소스에서 정보를 수집, 집계 및 분석합니다. GuardDuty는 CloudTrail, VPC 흐름 로그, DNS 쿼리 로그 및 Amazon EKS에서 직접 독립적인 데이터 스트림을 가져옵니다. Amazon S3 버킷 정책을 관리하거나 로그를 수집하고 저장하는 방식을 수정할 필요가 없습니다. 자체 조사 및 규정 준수 목적으로 이러한 로그를 보관하는 것이 좋습니다.
  - AWS Security Hub에서는 여러 AWS 서비스 및 서드파티 제품(선택 사항)의 보안 알림 또는 탐지 결과를 집계하고 정리하며 우선순위를 지정함으로써 보안 알림 및 규정 준수 상태를 종합적으로 파악할 수 있는 단일 장소를 제공합니다.

  이러한 서비스에서 다루지 않는 보안 알림 또는 환경과 관련된 특정 알림에 대해 사용자 지정 알림 생성 엔진을 사용할 수도 있습니다. 이러한 알림 및 탐지를 구축하는 방법에 대한 자세한 내용은 AWS Security Incident Response Guide의 Detection을 참조하세요.

리소스

관련 모범 사례:

- SEC04-BP02 표준화된 위치에서 로그, 조사 결과 및 지표 캡처
- SEC07-BP04 확장 가능한 데이터 수명 주기 관리 정의
- SEC10-BP06 도구 사전 배포

관련 문서:

- AWS Security Incident Response Guide
- Amazon Security Lake 시작하기
- Getting started: Amazon CloudWatch Logs

관련 비디오:

- AWS re:Invent 2022 - Introducing Amazon Security Lake

관련 예제:

- Assisted Log Enabler for AWS
- AWS Security Hub Findings Historical Export

SEC04-BP02 표준화된 위치에서 로그, 조사 결과 및 지표 캡처

보안 팀은 로그와 조사 결과를 바탕으로 무단 활동이나 의도하지 않은 변경을 나타낼 수 있는 이벤트를 분석합니다. 이 분석을 간소화하려면 표준화된 위치에서 보안 로그와 조사 결과를 캡처하세요. 이를 통해 관심 데이터 포인트를 상관관계 분석에 사용할 수 있고 도구 통합을 간소화할 수 있습니다.

원하는 성과: 로그 데이터, 조사 결과 및 지표를 수집, 분석 및 시각화하는 표준화된 접근 방식이 있습니다. 보안 팀은 서로 다른 시스템 전반의 보안 데이터를 효율적으로 분석 및 시각화하고 상관관계를 파악하여 잠재적 보안 이벤트를 발견하고 이상 징후를 식별할 수 있습니다. 보안 정보 및 이벤트 관리(SIEM) 시스템 또는 기타 메커니즘이 통합되어 보안 이벤트의 시기적절한 대응, 추적, 에스컬레이션을 위해 로그 데이터를 쿼리하고 분석합니다.

일반적인 안티 패턴:

- 팀이 조직의 로깅 전략과 일치하지 않는 로깅 및 지표 수집을 독립적으로 소유하고 관리합니다.
- 팀에 수집된 데이터의 가시성과 변경을 제한할 수 있는 적절한 액세스 제어 기능이 없습니다.
- 팀이 데이터 분류 정책의 일부로 보안 로그, 조사 결과 및 지표를 관리하지 않습니다.
- 팀이 데이터 수집을 구성할 때 데이터 주권 및 현지화 요구 사항을 무시합니다.

이 모범 사례 확립의 이점: 로그 데이터 및 이벤트를 수집하고 쿼리하는 표준화된 로깅 솔루션은 포함된 정보에서 도출되는 인사이트를 개선합니다. 수집된 로그 데이터의 자동화된 수명 주기를 구성하면 로그 스토리지로 인해 발생하는 비용을 줄일 수 있습니다. 팀에서 필요로 하는 데이터의 민감도와 액세스 패턴에 따라 수집된 로그 정보에 대한 세분화된 액세스 제어 기능을 구축할 수 있습니다. 도구를 통합하여 데이터의 상관관계를 파악하고, 데이터를 시각화하고, 데이터에서 인사이트를 도출할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 지침

조직 내 AWS 사용량이 증가하면 분산된 워크로드와 환경의 수가 늘어납니다. 이러한 각 워크로드 및 환경은 내부에 활동 데이터를 생성하므로, 보안 운영 측면에서 데이터를 로컬로 캡처하고 저장하기란 어렵습니다. 보안 팀은 보안 정보 및 이벤트 관리(SIEM) 시스템과 같은 도구를 사용하여 분산된 소스에서 데이터를 수집하고 상관관계 파악, 분석 및 대응 워크플로를 거칩니다. 이를 위해서는 다양한 데이터 소스에 액세스하기 위한 복잡한 권한 집합을 관리해야 하고 추출, 전환, 적재(ETL) 프로세스를 운영하는 데 추가 오버헤드가 듭니다.

이러한 문제를 해결하려면 Organizing Your AWS Environment Using Multiple Accounts에서 설명한 대로 보안 로그 데이터의 모든 관련 소스를 로그 아카이브 계정으로 집계하는 방법을 고려하세요. 여기에는 AWS CloudTrail, AWS WAF, Elastic Load Balancing, Amazon Route 53과 같이 AWS 서비스에서 생성하는 로그 및 워크로드의 모든 보안 관련 데이터가 포함됩니다. 적절한 크로스 계정 권한이 있는 별도의 AWS 계정에서 표준화된 위치의 데이터를 캡처하면 몇 가지 이점이 있습니다. 이러한 방식은 손상된 워크로드 및 환경 내에서 로그 변조를 방지하는 데 도움이 되며, 추가 도구를 위한 단일 통합 지점을 제공하고, 데이터 보존 및 수명 주기 구성을 위한 보다 간소화된 모델을 지원합니다. 데이터 주권, 규정 준수 범위 및 기타 규정의 영향을 평가하여 여러 보안 데이터 스토리지와 보존 기간이 필요한지 결정합니다.

로그와 조사 결과를 쉽게 캡처하고 표준화하려면 로그 아카이브 계정에서 Amazon Security Lake를 평가합니다. CloudTrail, Route 53, Amazon EKS, VPC 흐름 로그와 같은 일반적인 소스에서 자동으로 데이터를 수집하도록 Security Lake를 구성할 수 있습니다. 또한 Amazon GuardDuty 및 Amazon Inspector와 같은 기타 AWS 서비스에서 조사 결과와 로그 데이터를 연관시킬 수 있도록 Security Lake의 데이터 소스로 AWS Security Hub를 구성할 수 있습니다. 서드파티 데이터 소스 통합을 사용하거나 사용자 지정 데이터 소스를 구성할 수도 있습니다. 모든 통합은 데이터를 OSCF(Open Cybersecurity Schema Framework) 형식으로 표준화하고 Amazon S3 버킷에 Parquet 파일로 저장되므로, ETL 처리가 필요하지 않습니다.

보안 데이터를 표준화된 위치에 저장하면 고급 분석 기능이 제공됩니다. AWS는 AWS 환경에서 작동하는 보안 분석 도구를 로그 아카이브 계정과 별개인 보안 도구 계정에 배포할 것을 권장합니다. 이 접근 방식을 사용하면 로그와 로그 관리 프로세스에 액세스하는 도구와는 별개로 로그 및 로그 관리 프로세스의 무결성과 가용성을 보호하기 위한 제어 기능을 심층적으로 구현할 수 있습니다. Amazon Athena와 같은 서비스를 사용하여 여러 데이터 소스를 상관시키는 온디맨드 쿼리를 실행하는 방법을 고려하세요. QuickSight와 같은 시각화 도구를 통합할 수도 있습니다. AI 기반 솔루션은 점점 더 많이 사용되고 있으며, 조사 결과를 사람이 읽을 수 있는 요약 정보와 자연어 상호 작용으로 변환하는 등의 기능을 수행할 수 있습니다. 쿼리를 위한 표준화된 데이터 스토리지 위치를 사용하면 이러한 솔루션을 보다 쉽게 통합할 수 있는 경우가 많습니다.

구현 단계

1. 로그 아카이브 및 보안 도구 계정 생성
   a. AWS Organizations를 사용하여 보안 조직 단위 아래에 로그 아카이브 및 보안 도구 계정을 생성합니다. AWS Control Tower를 사용하여 조직을 관리하는 경우 로그 아카이브 계정 및 보안 도구 계정이 자동으로 생성됩니다. 필요에 따라 이러한 계정에 액세스하고 관리하기 위한 역할 및 권한을 구성합니다.
2. 표준화된 보안 데이터 위치 구성
   a. 표준화된 보안 데이터 위치를 만들기 위한 전략을 결정합니다. 일반적인 데이터 레이크 아키텍처 접근 방식, 서드파티 데이터 제품 또는 Amazon Security Lake와 같은 옵션을 통해 이를 달성할 수 있습니다. AWS는 적극적으로 사용하지 않을 때도 계정에 대해 옵트인한 AWS 리전에서 보안 데이터를 캡처할 것을 권장합니다.
3. 표준화된 위치에 데이터 소스 게시 구성
   a. 보안 데이터의 소스를 식별하고 표준화된 위치에 게시하도록 구성합니다. ETL 프로세스를 개발해야 하는 경우와 달리 원하는 형식으로 데이터를 자동으로 내보내는 옵션을 평가합니다. Amazon Security Lake를 사용하면 지원되는 AWS 소스와 통합된 서드파티 시스템에서 데이터를 수집할 수 있습니다.
4. 표준화된 위치에 액세스할 수 있는 도구 구성
   a. 표준화된 위치에 필요한 액세스 권한을 갖도록 Amazon Athena, QuickSight 또는 서드파티 솔루션과 같은 도구를 구성합니다. 해당하는 경우 로그 아카이브 계정에 대한 크로스 계정 읽기 권한이 있는 보안 도구 계정에서 작동하도록 관련 도구를 구성합니다. Amazon Security Lake에서 구독자를 생성하여 이러한 도구에 데이터 액세스 권한을 제공합니다.

리소스

관련 모범 사례:

- SEC01-BP01 계정을 사용하여 워크로드 분리
- SEC07-BP04 확장 가능한 데이터 수명 주기 관리 정의
- SEC08-BP04 액세스 제어 적용
- OPS08-BP02 워크로드 로그 분석

관련 문서:

- AWS Whitepapers: Organizing Your AWS Environment Using Multiple Accounts

- AWS Prescriptive Guidance: AWS Security Reference Architecture (AWS SRA)
- AWS Prescriptive Guidance: Logging and monitoring guide for application owners

관련 예제:

- Aggregating, searching, and visualizing log data from distributed sources with Amazon Athena and QuickSight
- How to visualize Amazon Security Lake findings with QuickSight
- Generate AI powered insights for Amazon Security Lake using Amazon SageMaker AI Studio and Amazon Bedrock
- Identify cybersecurity anomalies in your Amazon Security Lake data using Amazon SageMaker AI
- Ingest, transform, and deliver events published by Amazon Security Lake to Amazon OpenSearch Service
- CloudTrail Lake에서 자연어 쿼리 생성을 통한 AWS CloudTrail 로그 분석 간소화

관련 도구:

- Amazon Security Lake
- Amazon Security Lake Partner 통합
- Open Cybersecurity Schema Framework (OCSF)
- Amazon Athena
- QuickSight
- Amazon Bedrock

SEC04-BP03 보안 알림 보강 및 상관관계 지정

예상치 못한 활동은 여러 소스에서 다양한 보안 알림을 생성할 수 있으므로, 전체 컨텍스트를 이해하려면 추가 상관관계 분석 및 보강이 필요합니다. 자동화된 상관관계 분석을 구현하고 보안 알림을 보강하면 인시던트를 보다 정확하게 식별하고 대응할 수 있습니다.

원하는 성과: 활동이 워크로드 및 환경 내에서 다양한 알림을 생성하면 자동화된 메커니즘이 데이터의 상관관계를 파악하고 해당 데이터를 추가 정보로 보강합니다 이러한 사전 처리를 통해 이벤트를 보다 자세히 파악할 수 있으므로, 조사관이 이벤트의 심각성과 정식 대응이 필요한 인시던트인지를 판단하는 데 도움이 됩니다. 이 프로세스를 통해 모니터링 및 조사 팀의 업무가 줄어듭니다.

일반적인 안티 패턴:

- 업무 분담 요구 사항에서 달리 규정하지 않는 한, 여러 그룹의 사람들이 서로 다른 시스템에서 생성된 결과 및 알림을 조사합니다.
- 모든 보안 조사 결과 및 알림 데이터를 표준 위치에 퍼널링하지만, 조사관이 수동으로 상관관계 분석 및 보강 작업을 수행해야 합니다.
- 조사 결과를 보고하고 중요도를 설정하는 데 위협 탐지 시스템의 인텔리전스에만 의존합니다.

이 모범 사례 확립의 이점: 경보 보강 및 자동화된 상관관계 분석은 조사자의 전반적인 인지 부담과 데이터 준비 수작업을 으로 줄이는 데 도움이 됩니다. 이렇게 하면 이벤트가 인시던트인지를 판단하고 정식 대응을 시작하는 데 걸리는 시간을 줄일 수 있습니다. 또한, 추가 맥락 정보를 통해 이벤트의 실제 심각도를 정확하게 평가할 수 있습니다. 이벤트의 심각도는 특정 알림에서 제안하는 것보다 높거나 낮을 수 있기 때문입니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 낮음

구현 가이드

보안 알림은 다음을 포함하여 AWS 내부의 다양한 소스에서 비롯될 수 있습니다.

- Amazon GuardDuty, AWS Security Hub, Amazon Macie, Amazon Inspector, AWS Config, AWS Identity and Access Management Access Analyzer, Network Access Analyzer와 같은 서비스
- Amazon OpenSearch Service의 보안 분석과 같은 AWS 서비스, 인프라 및 애플리케이션 로그의 자동화된 분석에 기반하여 알림을 생성합니다.
- Amazon CloudWatch, Amazon EventBridge 또는 AWS Budgets와 같은 소스에서 청구 활동의 변경에 대응할 때 경보를 생성합니다.
- AWS Partner Network의 보안 파트너 솔루션 및 위협 인텔리전스 피드와 같은 서드파티 소스
- AWS Trust & Safety 또는 기타 소스(예: 고객 또는 내부 직원)를 통해 문의합니다.

누가(보안 주체 또는 자격 증명) 무엇(영향을 받는 리소스)에 대해 어떤 일(취해진 조치)을 수행하고 있는지에 대한 정보를 포함하는 것이 알림의 가장 기본적인 형식입니다. 각 소스에 대해 상관관계 분석을 수행하기 위한 토대로 이러한 ID, 작업, 리소스에 대한 식별자 간의 매핑을 생성할 수 있는 방법이 있는지 확인하세요. 이는 알림 소스를 보안 정보 및 이벤트 관리(SIEM) 도구와 통합하여 자동화된 상관관계 분석을 수행하거나, 자체 데이터 파이프라인 및 처리 과정을 구축하거나, 이 둘을 조합한 형태를 취할 수 있습니다.

사용자를 대신하여 상관관계 분석을 수행할 수 있는 서비스의 예로는 Amazon Detective가 있습니다. Detective는 다양한 AWS 및 서드파티 소스의 알림을 지속적으로 수집하고 여러 형태의 인텔리전스를 통해 관계를 시각적 그래프로 구성하여 조사를 지원합니다.

알림의 초기 중요도는 우선순위를 정하는 데 도움이 되지만, 알림이 발생한 맥락에 따라 실제 중요도가 결정됩니다. 예를 들어, Amazon GuardDuty는 워크로드 내 Amazon EC2 인스턴스가 예상치 못한 도메인 이름을 쿼리하고 있다고 알릴 수 있습니다. GuardDuty는 자체적으로 이 경고에 낮은 중요도를 할당할 수 있습니다. 그러나 알림이 발생한 당시 다른 활동과의 상관관계를 자동으로 분석하면 수백 개의 EC2 인스턴스가 동일한 ID로 배포되어 전체 운영 비용이 증가할 수 있습니다. 이 이벤트에서 이 상관관계가 있는 이벤트 맥락은 새 보안 알림을 게시하고 중요도는 높음으로 조정될 수 있으며, 이 경우 추가 조치를 신속하게 처리할 수 있습니다.

구현 단계

1. 보안 알림 정보의 소스를 식별합니다. 이러한 시스템의 알림이 ID, 작업 및 리소스를 어떻게 나타내는지 이해하여 상관관계 분석이 가능한 부분을 결정합니다.
2. 다양한 소스에서 알림을 캡처하기 위한 메커니즘을 설정합니다. 이를 위해 Security Hub, EventBridge, CloudWatch와 같은 서비스를 고려하세요.
3. 데이터 상관관계 분석과 보강을 위한 소스를 식별합니다. 예시 소스에는 AWS CloudTrail, VPC 흐름 로그, Route 53 Resolver 로그, 인프라 및 애플리케이션 로그가 포함됩니다. 이러한 로그의 일부 또는 전부는 Amazon Security Lake와의 단일 통합을 통해 사용될 수 있습니다.
4. 알림을 데이터 상관관계 분석 및 보강 소스와 통합하여 보다 상세한 보안 이벤트 맥락을 생성하고 중요도를 설정합니다.
    a. Amazon Detective, SIEM 도구 또는 기타 서드파티 솔루션은 특정 수준의 수집, 상관관계 분석, 보강을 자동으로 수행할 수 있습니다.
    b. AWS 서비스를 사용하여 직접 구축할 수도 있습니다. 예를 들어, AWS Lambda 함수를 간접 호출하여 AWS CloudTrail 또는 Amazon Security Lake에 대해 Amazon Athena 쿼리를 실행하고 결과를 EventBridge에 게시할 수 있습니다.

리소스

관련 모범 사례:

- SEC10-BP03 포렌식 역량 확보
- OPS08-BP04 실행 가능한 알림 생성
- REL06-BP03 알림 전송(실시간 처리 및 경보)

관련 문서:

- AWS Security Incident Response Guide

관련 예제:

- 계정 메타데이터로 AWS Security Hub 조사 결과를 보강하는 방법

관련 도구:

- Amazon Detective
- Amazon EventBridge
- AWS Lambda
- Amazon Athena

SEC04-BP04 규정 미준수 리소스 관련 문제 해결 시작

탐지 제어를 통해 구성 요구 사항을 준수하지 않는 리소스에 대해 알림을 보낼 수 있습니다. 프로그래밍 방식으로 정의된 수정을 수동 또는 자동으로 시작하여 이러한 리소스를 수정하고 잠재적 영향을 최소화할 수 있습니다. 수정을 프로그래밍 방식으로 정의하면 신속하고 일관된 조치를 취할 수 있습니다.

자동화는 보안 운영을 개선할 수 있지만, 자동화를 신중하게 구현하고 관리해야 합니다. 적절한 감독 및 제어 메커니즘을 마련하여 자동 대응이 효과적이고 정확하며 조직의 정책과 위험을 바라보는 관점에 부합하는지 확인하세요.

원하는 성과: 리소스가 규정을 준수하지 않는 것으로 탐지될 때 이를 수정하기 위한 단계와 함께 리소스 구성 표준을 정의합니다. 가능하면 수동으로 또는 자동화를 통해 시작할 수 있도록 프로그래밍 방식으로 수정을 정의했습니다. 규정 미준수 리소스를 식별하고 보안 담당자가 모니터링하는 중앙 집중식 도구에 알림을 게시할 수 있는 탐지 시스템이 마련되어 있습니다. 이러한 도구는 수동 또는 자동으로 프로그래밍 방식의 수정 실행을 지원합니다. 자동 수정에 사용을 관리하기 위한 적절한 감독 및 제어 메커니즘이 마련되어 있습니다.

일반적인 안티 패턴:

- 자동화를 구현했지만, 수정 조치를 철저하게 테스트하고 검증하지 못했습니다. 이로 인해 정상적인 비즈니스 운영이 중단되거나 시스템이 불안정해지는 등 의도하지 않은 결과가 발생할 수 있습니다.

- 자동화를 통해 대응 시간과 절차를 개선할 수 있지만, 필요한 경우 사람이 개입하고 판단할 수 있는 적절한 모니터링 기능과 메커니즘이 없습니다.
- 보다 광범위한 인시던트 대응 및 복구 프로그램의 일부로서 수정이 아닌 일반적인 수정에만 의존합니다.

이 모범 사례 확립의 이점: 자동 수정은 수동 프로세스를 사용할 때보다 잘못된 구성에 더 빠르게 대응할 수 있으므로, 비즈니스에 미칠 수 있는 영향을 최소화하고 의도하지 않은 사용에 투입된 잠재적인 기간을 줄일 수 있습니다. 수정을 프로그래밍 방식으로 정의하면 일관되게 적용되어 인적 오류 위험이 줄어듭니다. 자동화는 또한 많은 양의 알림을 동시에 처리할 수 있는데, 이는 대규모로 운영되는 환경에서 특히 중요합니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

SEC01-BP03 제어 목표 파악 및 검증에서 설명한 대로, AWS Config 및 AWS Security Hub와 같은 서비스를 사용하면 요구 사항을 준수하는지 계정의 리소스 구성을 모니터링할 수 있습니다. 규정을 준수하지 않는 리소스가 감지되면 AWS Security Hub와 같은 서비스가 알림을 적절하게 라우팅하고 문제를 해결하는 데 도움이 될 수 있습니다. 이러한 솔루션은 보안 조사관이 문제를 모니터링하고 시정 조치를 취할 수 있는 중앙 장소를 제공합니다.

일부 리소스가 규정을 준수하지 않는 고유한 문제가 발생하여 수정하려면 사람의 판단이 필요한 경우도 있지만, 프로그래밍 방식으로 정의할 수 있는 표준 대응이 효과가 있는 상황도 있습니다. 예를 들어, 잘못 구성된 VPC 보안 그룹에 대한 표준 대응은 허용되지 않는 규칙을 제거하고 소유자에게 알리는 것일 수 있습니다. 응답은 AWS Lambda 함수, AWS Systems Manager Automation 문서에서 정의하거나 원하는 다른 코드 환경을 통해 정의할 수 있습니다. 수정 조치에 필요한 최소한의 권한으로 IAM 역할을 사용하여 환경이 AWS에 인증할 수 있는지 확인하세요.

원하는 수정 조치를 정의한 후에는 이를 시작하는 데 원하는 방법을 결정할 수 있습니다. AWS Config에서는 자동으로 수정을 시작할 수 있습니다. Security Hub를 사용하는 경우 사용자 지정 작업을 통해 이 작업을 수행할 수 있습니다. 그러면 조사 결과 정보를 Amazon EventBridge에 게시합니다. 그러면 EventBridge 규칙에 따라 수정이 시작될 수 있습니다. Security Hub에서 해결 방법을 자동 또는 수동으로 실행하도록 구성할 수 있습니다.

프로그래밍 방식의 수정을 위해서는 수행된 조치와 결과에 대해 포괄적인 로그와 감사를 수행하는 것이 좋습니다. 이러한 로그를 검토 및 분석하여 자동화된 프로세스의 효과를 평가하고 개선 영역을 식별합니다. Amazon CloudWatch Logs에서 로그를 캡처하고 Security Hub에서 조사 결과 노트로 결과를 캡처합니다.

시작점으로 Automated Security Response on AWS를 고려하세요. 여기에는 일반적인 보안 구성 오류를 해결하기 위한 수정 방법이 미리 구축되어 있습니다.

구현 단계

1. 알림을 분석하고 우선순위를 지정합니다.
   a. 다양한 AWS 서비스의 보안 알림을 Security Hub에 통합하여 중앙 집중식 가시성, 우선순위 지정 및 문제 해결을 제공합니다.
2. 수정 방안을 개발합니다.
   a. Systems Manager 및 AWS Lambda 등의 서비스를 사용하여 프로그래밍 방식의 수정을 실행할 수 있습니다.
3. 수정 시작 방법을 구성합니다.
   a. Systems Manager를 사용하여 조사 결과를 EventBridge에 게시할 사용자 지정 작업을 정의합니다. 이러한 작업이 수동 또는 자동으로 시작되도록 구성합니다.
   b. 또한 필요한 경우 Amazon Simple Notification Service(SNS)를 사용하여 관련 이해관계자(예: 보안 팀 또는 인시던트 대응 팀)를 대상으로 수동 개입 또는 에스컬레이션에 대한 알림 및 경보를 보낼 수도 있습니다.
4. 수정 로그를 검토 및 분석하여 효과와 개선 사항을 확인합니다.
   a. CloudWatch Logs로 로그 출력을 전송합니다. Security Hub에서 조사 결과를 결과로 캡처합니다.

리소스

관련 모범 사례:

- SEC06-BP03 수동 관리 및 대화형 액세스 감소

관련 문서:

- AWS Security Incident Response Guide - Detection

관련 예제:

- Automated Security Response on AWS
- Monitor EC2 instance key pairs using AWS Config
- Create AWS Config custom rules by using AWS CloudFormation Guard policies

- Automatically remediate unencrypted Amazon RDS DB instances and clusters

관련 도구:

- AWS Systems Manager Automation
- Automated Security Response on AWS

인프라 보호

모범 사례와 업계 규정 또는 규제 의무를 준수하기 위해서는 인프라 보호가 필요하며, 여기에는 심층 방어와 같은 제어 방법이 포함됩니다. 클라우드에서 작업을 계속 성공적으로 수행하려면 이러한 방법을 사용해야 합니다.

인프라 보호는 정보 보안 프로그램의 핵심 요소입니다. 인프라 보호 기능을 사용하면 의도하지 않은 무단 침입 및 잠재적 취약성으로부터 워크로드 내의 시스템과 서비스를 보호할 수 있습니다. 예를 들어 신뢰 경계(예: 네트워크 및 계정 경계), 시스템 보안 구성 및 유지 관리(예: 강화, 최소화 및 패치 적용), 운영 체제 인증 및 권한 부여(예: 사용자, 키 및 액세스 수준) 및 기타 적절한 정책 적용 지점(예: 웹 애플리케이션 방화벽 및/또는 API 게이트웨이)을 정의할 수 있습니다.

리전, 가용 영역, AWS 로컬 영역, AWS Outposts

AWS 보안 글로벌 인프라를 구성하는 리전, 가용 영역, AWS 로컬 영역, AWS Outposts에 친숙해져야 합니다.

AWS는 데이터 센터를 클러스터링하는 세계 곳곳의 물리적 로케이션을 의미하는 리전이라는 개념을 사용합니다. 각각의 논리적 데이터 센터 그룹을 가용 영역이라고 합니다. 각 AWS 리전은 지리적 영역 내에서 물리적으로 분리된 여러 개의 격리된 AZ로 구성됩니다. 데이터 레지던시 요구 사항이 있는 경우 원하는 위치와 가까운 AWS 리전을 선택할 수 있습니다. 데이터가 물리적으로 위치한 리전에 대한 완전한 통제와 소유권을 보유합니다. 이 경우 지역별 규정 준수 및 데이터 레지던시 요구 사항을 충족하는 데 도움이 될 수 있습니다. 각 AZ에서 전원, 냉각 및 물리적 보안이 독립되어 있습니다. 여러 AZ에 걸쳐 애플리케이션이 분할된 경우 정전, 낙뢰, 토네이도, 지진 등의 문제로부터 더 효과적으로 격리되고 보호됩니다. AZ는 모두 서로 100km(60마일) 내에 있지만 다른 AZ와 의미 있는 거리(킬로미터 단위)만큼 물리적으로 분리됩니다. AWS 리전의 모든 AZ는 완전히 중복된 전용 메트로 섬유를 사용하여 고대역폭과 짧은 지연 시간의 네트워킹으로 상호 연결되어 있기 때문에 AZ 간에 높은 처리량과 짧은 지연 시간을 지원합니다. AZ 간 모든 트래픽은 암호화됩니다. 고가용성에 중점을 둔 AWS 고객은 내결함성을 더욱 강화하기 위해 여러 AZ에서 실행되도록 애플리케이션을 설계할 수 있습니다. AWS 리전은 가장 높은 수준의 보안, 규정 준수, 데이터 보호 요건을 충족합니다.

AWS 로컬 영역은 컴퓨팅, 스토리지, 데이터베이스 및 기타 엄선된 AWS 서비스를 최종 사용자에게 더 가깝게 배치합니다. AWS 로컬 영역을 사용하면 미디어 및 엔터테인먼트 콘텐츠 제작, 실시간 게임, 저장소 시뮬레이션, 전자 설계 자동화 및 기계 학습과 같이 최종 사용자 측에서 지연 시간이 한 자릿수 밀리초 단위여야 하는 매우 까다로운 애플리케이션도 쉽게 실행할 수 있습니다. 각 AWS 로컬 영역 위치는 최종 사용자와 지리적으로 가까운 곳에서 Amazon EC2, Amazon VPC, Amazon EBS, Amazon File Storage 및 Elastic Load Balancing과 같은 AWS 서비스를 사용하여 지연 시간에 민감한 애플리케이션을 실행할 수 있는 AWS 리전의 확장 기능입니다. AWS 로컬 영역은 로컬 워크로드와 AWS 리전에서

실행되는 워크로드 간에 고대역폭의 보안 연결을 제공하므로 동일한 API와 도구 세트를 통해 리전의 전체 서비스에 원활하게 연결할 수 있습니다.

AWS Outposts는 네이티브 AWS 서비스, 인프라 및 운영 모델을 사실상 모든 데이터 센터, 코로케이션 공간 또는 온프레미스 시설로 옮길 수 있습니다. 온프레미스 시설과 AWS 클라우드 전반에서 동일한 AWS API, 도구 및 인프라를 사용하여 진정으로 일관된 하이브리드 경험을 제공할 수 있습니다. AWS Outposts는 커넥티드 환경을 위해 설계되었으며 짧은 지연 시간 또는 로컬 데이터 처리 요구 사항으로 인해 온프레미스에 있어야 하는 워크로드를 지원하는 데 사용할 수 있습니다.

AWS에서는 다양한 방식으로 인프라를 보호할 수 있습니다. 다음 섹션에서는 이러한 접근 방식을 사용하는 방법을 설명합니다.

주제

- 네트워크 보호
- 컴퓨팅 보호

네트워크 보호

직원과 고객 측 모두에서 사용자는 어디에나 위치할 수 있습니다. 네트워크에 액세스할 수 있는 모든 사람을 신뢰하는 기존 모델에서 벗어나야 합니다. 모든 계층에 보안을 적용한다는 원칙을 따를 때는 제로 트러스트 접근 방식을 채택하게 됩니다. 제로 트러스트 보안은 애플리케이션 구성 요소 또는 마이크로서비스가 서로 분리된 것으로 간주되고 구성 요소나 마이크로서비스가 다른 구성 요소나 마이크로서비스를 신뢰하지 않는 모델입니다.

워크로드 내의 리소스에 격리와 경계를 적용하려면 기본적으로 네트워크 설계를 철저하게 계획하고 관리해야 합니다. 워크로드의 많은 리소스가 VPC에서 작동하고 보안 속성을 상속하기 때문에 자동화된 검사 및 보호 메커니즘을 기반으로 설계하는 것이 중요합니다. 마찬가지로, 순전히 엣지 서비스 및/또는 서버리스를 사용하여 VPC 외부에서 작동하는 워크로드의 경우에는 모범 사례가 좀 더 단순화된 접근 방식으로 적용됩니다. 서버리스 보안에 대한 구체적인 지침은 AWS Well-Architected Serverless Applications Lens를 참조하세요.

모범 사례

- SEC05-BP01 네트워크 계층 생성
- SEC05-BP02 네트워크 계층 내 트래픽 흐름 제어
- SEC05-BP03 검사 기반 보호 구현
- SEC05-BP04 네트워크 보호 자동화

SEC05-BP01 네트워크 계층 생성

데이터 민감도 및 액세스 요구 사항에 따라 워크로드 구성 요소의 논리적 그룹을 기반으로 네트워크 토폴로지를 여러 계층으로 세분화합니다. 인터넷에서 인바운드 액세스를 필요로 하는 구성 요소(예: 퍼블릭 웹 엔드포인트)와 내부 액세스만 필요한 구성 요소(예: 데이터베이스)를 구분합니다.

원하는 성과: 네트워크 계층은 워크로드의 자격 증명 인증 및 권한 부여 전략을 보완하는 보안에 대한 통합 심층 방어 접근 방식의 일부입니다. 데이터 민감도 및 액세스 요구 사항에 따라 적절한 트래픽 흐름 및 제어 메커니즘과 함께 계층이 배치됩니다.

일반적인 안티 패턴:

- 단일 VPC 또는 서브넷에 모든 리소스를 생성합니다.
- 데이터 민감도 요구 사항, 구성 요소 동작 또는 기능을 고려하지 않고 네트워크 계층을 구성합니다.
- 모든 네트워크 계층 고려 사항의 기본값으로 VPC와 서브넷을 사용하며, AWS 관리형 서비스가 토폴로지에 미치는 영향을 고려하지 않습니다.

이 모범 사례 확립의 이점: 네트워크 계층 구축은 네트워크를 통한 불필요한 경로, 특히 중요한 시스템 및 데이터로 이어지는 불필요한 경로를 제한하는 첫 번째 단계입니다. 이를 통해 승인되지 않은 행위자가 네트워크에 액세스할 권한을 얻어 내부의 추가 리소스를 탐색하기가 더 어려워집니다. 개별 네트워크 계층은 침입 탐지 또는 맬웨어 방지와 같은 검사 시스템의 분석 범위를 효과적으로 줄입니다. 이렇게 하면 오탐과 불필요한 처리 오버헤드가 생길 가능성이 낮아집니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

워크로드 아키텍처를 설계할 때는 보통 구성 요소를 책임에 따라 여러 계층으로 분리합니다. 예를 들어, 웹 애플리케이션에는 프레젠테이션 계층, 애플리케이션 계층, 데이터 계층이 있을 수 있습니다. 네트워크 토폴로지를 설계할 때도 비슷한 접근 방식을 사용할 수 있습니다. 기본 네트워크 제어는 워크로드의 데이터 액세스 요구 사항을 적용하는 데 도움이 될 수 있습니다. 예를 들어, 3계층 웹 애플리케이션 아키텍처에서는 정적 프레젠테이션 계층 파일을 Amazon S3에 저장하고 Amazon CloudFront 등의 콘텐츠 전송 네트워크(CDN)에서 제공할 수 있습니다. 애플리케이션 계층에는 프라이빗 서브넷에 배포된 백엔드 서비스를 통해 Application Load Balancer(ALB)가 Amazon VPC 퍼블릭 서브넷(DMZ와 유사함)에서 지원하는 퍼블릭 엔드포인트가 있을 수 있습니다. 데이터베이스, 공유 파일 시스템과 같은 리소스를 호스팅하는 데이터 계층은 애플리케이션 계층의 리소스와는 다른 프라이빗 서브넷에 있을 수 있습니다. 각 계층 경계(CDN, 퍼블릭 서브넷, 프라이빗 서브넷)에서 승인된 트래픽만 해당 경계를 통과하도록 허용하는 제어 기능을 배포할 수 있습니다.

워크로드 구성 요소의 기능적 목적을 기반으로 네트워크 계층을 모델링하는 것과 마찬가지로, 처리 중인 데이터의 민감도도 고려하세요. 웹 애플리케이션 예제를 사용하면 모든 워크로드 서비스가 애플리케이션 계층 내에 있을 수 있지만, 서비스마다 민감도 수준이 다른 데이터를 처리할 수 있습니다. 이 경우 데이터 민감도 수준별로 여러 프라이빗 서브넷, 동일한 AWS 계정의 다른 VPC 또는 다른 AWS 계정의 다른 VPC를 사용하여 애플리케이션 계층을 나누는 것이 제어 요구 사항에 따라 적합할 수 있습니다.

네트워크 계층에 대해 추가로 고려해야 할 사항은 워크로드 구성 요소의 동작 일관성입니다. 계속해서 예를 들자면, 애플리케이션 계층에는 최종 사용자의 입력을 받아들이는 서비스 또는 다른 서비스에 대한 입력보다 본질적으로 더 위험한 외부 시스템 통합이 있을 수 있습니다. 파일 업로드, 실행할 코드 스크립트, 이메일 스캔 등을 그 예로 들 수 있습니다. 이러한 서비스를 자체 네트워크 계층에 배치하면 주위에 더 강력한 격리 경계가 형성되고, 검사 시스템에서 이러한 서비스의 고유한 동작으로 인해 오탐 알림이 발생하는 것을 방지할 수 있습니다.

설계의 일부로 AWS 관리형 서비스 사용이 네트워크 토폴로지에 어떤 영향을 미치는지 고려해 보세요. Amazon VPC Lattice와 같은 서비스를 통해 네트워크 계층 전반에서 워크로드 구성 요소의 상호 운용성을 더 쉽게 지원하는 방법을 알아보세요. AWS Lambda를 사용할 때는 특별한 이유가 없다면 VPC 서브넷에 배포합니다. VPC 엔드포인트와 AWS PrivateLink가 인터넷 게이트웨이에 대한 액세스를 제한하는 보안 정책을 간편하게 준수할 수 있는 위치를 확인합니다.

구현 단계

1. 워크로드 아키텍처를 검토합니다. 제공하는 기능, 처리 중인 데이터의 민감도, 동작을 기반으로 구성 요소와 서비스를 논리적으로 그룹화하세요.
2. 인터넷 요청에 응답하는 구성 요소의 경우 로드 밸런서 또는 기타 프록시를 사용하여 퍼블릭 엔드포인트를 제공하는 것을 고려해 보세요. 퍼블릭 엔드포인트를 호스팅하기 위해 CloudFront, Amazon API Gateway, Elastic Load Balancing, AWS Amplify와 같은 관리형 서비스를 사용하여 변화하는 보안 제어를 살펴보세요.
3. Amazon EC2 인스턴스, AWS Fargate 컨테이너 또는 Lambda 함수와 같은 컴퓨팅 환경에서 실행되는 구성 요소의 경우 첫 번째 단계부터 그룹을 기반으로 이러한 구성 요소를 프라이빗 서브넷에 배포합니다.
4. Amazon DynamoDB, Amazon Kinesis 또는 Amazon SQS와 같은 완전관리형 AWS 서비스의 경우 프라이빗 IP 주소를 통한 액세스의 기본값으로 VPC 엔드포인트를 사용하는 방법을 고려하세요.

리소스

관련 모범 사례:

- REL02 네트워크 토폴로지 계획
- PERF04-BP01 네트워킹이 성능에 미치는 영향 파악

관련 비디오:

- AWS re:Invent 2023 - AWS networking foundations

관련 예제:

- VPC 예시
- AWS Fargate, AWS PrivateLink 및 Network Load Balancer를 사용하여 Amazon ECS에서 컨테이너 애플리케이션에 비공개로 액세스
- Serve static content in an Amazon S3 bucket through a VPC by using Amazon CloudFront

SEC05-BP02 네트워크 계층 내 트래픽 흐름 제어

네트워크 계층 내에서 추가 세분화를 사용하여 각 워크로드에 필요한 흐름으로만 트래픽을 제한할 수 있습니다. 먼저, 워크로드와 사용자 환경에 대한 인터넷 또는 기타 외부 시스템 간의 트래픽(남북 트래픽)을 제어하는 데 중점을 둡니다. 그런 다음 서로 다른 구성 요소와 시스템 간의 흐름(동서 트래픽)을 살펴보세요.

원하는 성과: 워크로드 구성 요소가 서로 통신하고 해당 클라이언트 및 해당 클라이언트가 의존하는 다른 서비스와 통신하는 데 필요한 네트워크 흐름만 허용합니다. 설계에서 프라이빗 송수신과 비교한 퍼블릭 송수신, 데이터 분류, 지역 규정, 프로토콜 요구 사항 등의 고려 사항을 고려합니다. 가능한 경우 최소 권한 원칙 설계의 일환으로 네트워크 피어링보다 지점 간 흐름을 선호합니다.

일반적인 안티 패턴:

- 네트워크 보안에 대한 경계 기반 접근 방식을 취하고 네트워크 계층 경계에서의 트래픽 흐름만 제어합니다.
- 네트워크 계층 내의 모든 트래픽이 인증되고 승인되었다고 가정합니다.
- 수신 트래픽과 송신 트래픽 중 하나에 제어 기능을 적용하지만, 둘 다에 적용하지는 않습니다.
- 트래픽을 인증하고 승인하는 데 워크로드 구성 요소 및 네트워크 제어에만 의존합니다.

이 모범 사례 확립의 이점: 이 방법은 네트워크 내 무단 이동의 위험을 줄이고 워크로드에 추가 인증 계층을 더하는 데 도움이 됩니다. 트래픽 흐름 제어를 수행하면 보안 인시던트의 영향 범위를 제한하고 탐지 및 대응 속도를 높일 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 지침

네트워크 계층은 유사한 기능, 데이터 민감도 수준 및 동작을 제공하는 워크로드 구성 요소 주변의 경계를 설정하는 데 도움이 됩니다. 다만 최소 권한 원칙을 따르는 이러한 계층 내 구성 요소를 추가로 분할하는 기법을 사용하여 훨씬 더 세분화된 수준의 트래픽 제어를 구성할 수 있습니다. AWS에서 네트워크 계층은 주로 Amazon VPC 내 IP 주소 범위에 따라 서브넷을 사용하여 정의됩니다. 다양한 VPC를 사용하여 계층을 정의할 수도 있습니다(예: 비즈니스 도메인별로 마이크로서비스 환경을 그룹화하는 경우). 여러 VPC를 사용하는 경우 AWS Transit Gateway를 통해 라우팅을 조정합니다. 이 경우 보안 그룹 및 라우팅 테이블을 사용하여 계층 4 수준(IP 주소 및 포트 범위)의 트래픽 제어를 제공하지만, AWS PrivateLink, Amazon Route 53 Resolver DNS Firewall, AWS Network Firewall, AWS WAF와 같은 추가 서비스를 사용하여 또 다른 제어 기능을 확보할 수 있습니다.

연결 개시 당사자, 포트, 프로토콜 및 네트워크 계층 측면에서 워크로드의 데이터 흐름 및 커뮤니케이션 요구 사항을 이해하고 인벤토리를 작성합니다. 연결 설정 및 데이터 전송에 사용할 수 있는 프로토콜을 평가하여 보호 요구 사항(예: HTTP가 아닌 HTTPS)을 충족하는 프로토콜을 선택합니다. 네트워크 경계와 각 계층 내 모두에서 이러한 요구 사항을 파악합니다. 요구 사항이 확인되면 각 연결 지점에서 필요한 트래픽만 흐르도록 허용하는 옵션을 살펴보세요. 탄력적 네트워크 인터페이스(ENI)를 사용하는 리소스(예: Amazon EC2 인스턴스, Amazon ECS 작업, Amazon EKS 포드 또는 Amazon RDS 데이터베이스)에 연결할 수 있으므로, 처음에는 VPC 내에서 보안 그룹을 사용해서 시작하는 것이 좋습니다. 계층 4 방화벽과 달리 보안 그룹에는 식별자별로 다른 보안 그룹의 트래픽을 허용하는 규칙이 있어 시간이 지남에 따라 그룹 내 리소스가 변경될 때 업데이트를 최소화할 수 있습니다. 또한, 보안 그룹을 통해 인바운드 규칙과 아웃바운드 규칙을 모두 사용하여 트래픽을 필터링할 수 있습니다.

트래픽이 VPC 간에 이동할 때 단순 라우팅에 VPC 피어링을 사용하거나 복잡한 라우팅에 AWS Transit Gateway를 사용하는 것이 일반적입니다. 이러한 접근 방식을 사용하면 소스 네트워크와 대상 네트워크의 IP 주소 범위 간 트래픽이 원활하게 흐르도록 할 수 있습니다. 하지만 워크로드에 서로 다른 VPC의 특정 구성 요소 간 트래픽 흐름만 필요한 경우에는 AWS PrivateLink를 통해 지점 간 연결을 사용하는 것이 좋습니다. 이를 위해서는 생산자 역할을 해야 하는 서비스와 소비자 역할을 해야 하는 서비스를 식별해야 합니다. 생산자에 대해 호환 가능한 로드 밸런서를 배포하고, 그에 따라 PrivateLink를 설정한 다음, 소비자의 연결 요청을 수락합니다. 그러면 생산자 서비스에 소비자 VPC의 프라이빗 IP 주소가 할당되며, 소비자는 이를 사용하여 후속 요청을 할 수 있습니다. 이 접근 방식을 사용하면 네트워크를 피어링할 필요가 줄어듭니다. PrivateLink 평가의 일부로 데이터 처리 및 로드 밸런싱에 드는 비용을 포함합니다.

보안 그룹과 PrivateLink는 워크로드의 구성 요소 간 흐름을 제어하는 데 도움이 되지만, 리소스에 액세스할 수 있는 DNS 도메인(있는 경우)을 제어하는 방법도 또 다른 주요 고려 사항입니다. VPC의 DHCP 구성에 따라 이를 위해 두 가지 다른 AWS 서비스를 고려할 수 있습니다. 대부분의 고객은 CIDR 범위의 +2 주소에 있는 VPC에서 사용할 수 있는 기본 Route 53 Resolver DNS 서비스(Amazon DNS 서버 또는 AmazonProvidedDNS라고도 함)를 사용합니다. 이 접근 방식을 사용하면 DNS 방화벽 규칙을 생성하고, 이를 VPC에 연결하여 제공한 도메인 목록에 대해 수행해야 할 작업을 결정할 수 있습니다.

Route 53 Resolver를 사용하지 않거나 도메인 필터링 외에도 심층 검사 및 흐름 제어 기능으로 Resolver를 보완하려면 AWS Network Firewall 배포를 고려해 보세요. 이 서비스는 상태 비저장 또는 상태 저장 규칙을 통해 개별 패킷을 검사하여 트래픽을 거부할지, 아니면 허용할지 결정합니다. AWS WAF를 사용하여 퍼블릭 엔드포인트에 대한 인바운드 웹 트래픽을 필터링할 때도 비슷한 접근 방식을 취할 수 있습니다. 이러한 서비스에 대한 추가 지침은 SEC05-BP03 검사 기반 보호 구현을 참조하세요.

구현 단계

1. 워크로드 구성 요소 간에 필요한 데이터 흐름을 식별합니다.
2. 보안 그룹과 라우팅 테이블 사용을 포함하여 인바운드 및 아웃바운드 트래픽 모두에 대해 심층 방어 접근 방식으로 다중 제어 기능을 적용합니다.
3. 방화벽을 사용하여 Route 53 Resolver DNS 방화벽, AWS Network Firewall, AWS WAF와 같은 VPC 내부, 외부 및 전체의 네트워크 트래픽에 대한 세분화된 제어를 정의합니다. AWS Firewall Manager를 사용하여 조직 전체의 방화벽 규칙을 중앙에서 구성하고 관리하는 방법을 고려하세요.

리소스

관련 모범 사례:

- REL03-BP01 워크로드를 세그먼트화하는 방법 선택
- SEC09-BP02 전송 중 암호화 적용

관련 문서:

- VPC에 대한 보안 모범 사례
- AWS Network Optimization Tips
- Guidance for Network Security on AWS
- Secure your VPC's outbound network traffic in the AWS 클라우드

관련 도구:

- AWS Firewall Manager

관련 비디오:

- AWS Transit Gateway reference architectures for many VPCs
- Application Acceleration and Protection with Amazon CloudFront, AWS WAF, and AWS Shield
- AWS re:Inforce 2023: Firewalls and where to put them

SEC05-BP03 검사 기반 보호 구현

네트워크 계층 간에 트래픽 검사 지점을 설정하여 전송 중인 데이터가 예상 범주 및 패턴과 일치하는지 확인하세요. 트래픽 흐름, 메타데이터, 패턴을 분석하여 이벤트를 보다 효과적으로 식별, 탐지 및 대응할 수 있습니다.

원하는 성과: 네트워크 계층 사이를 이동하는 트래픽을 검사하고 승인합니다. 허용 및 거부 결정은 명시적 규칙, 위협 인텔리전스 및 기준 행동과의 편차를 기반으로 합니다. 트래픽이 민감한 데이터에 가까워질수록 보호가 더욱 엄격해집니다.

일반적인 안티 패턴:

- 포트 및 프로토콜에 기반한 방화벽 규칙에만 의존합니다. 지능형 시스템을 활용하지 않습니다.
- 변경될 수 있는 특정 최신 위협 패턴을 기반으로 방화벽 규칙을 작성합니다.
- 프라이빗 서브넷에서 퍼블릭 서브넷으로 이동하는 트래픽 또는 퍼블릭 서브넷에서 인터넷으로 전송되는 트래픽만 검사합니다.
- 네트워크 트래픽의 기본 뷰를 통해 이상 동작을 비교할 수 없습니다.

이 모범 사례 확립의 이점: 검사 시스템을 사용하면 트래픽 데이터에 특정 조건이 있는 경우에만 트래픽을 허용하거나 거부하는 등의 지능형 규칙을 작성할 수 있습니다. 시간이 흘러 위협 환경이 변화함에 따라 최신 위협 인텔리전스를 기반으로 AWS 및 파트너의 관리형 규칙 집합을 활용할 수 있습니다. 이를 통해 규칙을 유지 관리하고 보안 침해 지표를 조사하는 데 드는 오버헤드가 줄어 오탐이 발생할 가능성이 낮아집니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

AWS Network Firewall 또는 Gateway Load Balancer(GWLB) 이면에 배포할 수 있는 기타 방화벽 및 침입 방지 시스템(IPS)을 AWS Marketplace에서 사용하여 상태 저장 및 상태 비저장 네트워크 트래픽을 세밀하게 제어할 수 있습니다. AWS Network Firewall은 워크로드를 보호하는 데 도움이 되는 Suricata 호환 오픈 소스 IPS 사양을 지원합니다.

GWLB를 사용하는 AWS Network Firewall 및 공급업체 솔루션 모두 서로 다른 인라인 검사 배포 모델을 지원합니다. 예를 들어, VPC별로 검사를 수행하거나, 검사 VPC를 중앙 집중화하거나, 검사 VPC를 통해 동서 트래픽이 흐르고 VPC별로 인터넷 수신이 검사되는 하이브리드 모델에 배포할 수 있습니다. 또 다른 고려 사항은 솔루션이 전송 계층 보안(TLS) 언래핑을 지원하여 어느 방향에서든 시작된 트래픽 흐름에 대한 심층 패킷 검사를 지원하는지 여부입니다. 이러한 구성에 대한 심층적인 세부 정보와 자세한 내용은 AWS Network Firewall Best Practice guide를 참조하세요.

무차별 모드로 작동하는 네트워크 인터페이스의 패킷 데이터에 대한 pcap 분석과 같이 대역 외 검사를 수행하는 솔루션을 사용하는 경우 VPC 트래픽 모니터링을 구성할 수 있습니다. 미러링된 트래픽은 인터페이스의 가용 대역폭에 포함되며 미러링되지 않은 트래픽과 동일한 데이터 전송 요금이 부과됩니다. 이러한 어플라이언스의 가상 버전을 AWS Marketplace에서 사용할 수 있는지 확인할 수 있으며, 이를 통해 GWLB 이면에서 인라인 배포를 지원할 수 있습니다.

HTTP 기반 프로토콜을 통해 트랜잭션하는 구성 요소의 경우 웹 애플리케이션 방화벽(WAF)을 통해 일반적인 위협으로부터 애플리케이션을 보호합니다. AWS WAF는 Amazon API Gateway, Amazon CloudFront, AWS AppSync 또는 Application Load Balancer로 전송하기 전에 구성 가능한 규칙과 일치하는 HTTP(S) 요청을 모니터링하고 차단하는 웹 애플리케이션 방화벽입니다. 일부 방화벽에서는 트래픽 검사 전에 TLS를 종료해야 하므로, 웹 애플리케이션 방화벽의 배포를 평가할 때는 심층 패킷 검사를 고려해 보세요. AWS WAF를 시작하려면 AWS Managed Rules를 자체 규칙과 함께 사용하거나 기존 파트너 통합을 사용할 수 있습니다.

AWS Firewall Manager를 사용하여 AWS 조직 전반에 걸쳐 AWS WAF, AWS Shield Advanced, AWS Network Firewall, Amazon VPC 보안 그룹을 중앙에서 관리할 수 있습니다.

구현 단계

1. 검사 VPC를 통해서처럼 검사 규칙의 범위를 광범위하게 지정할 수 있는지 또는 VPC별로 좀 더 세분화된 접근 방식이 필요한지 결정합니다.
2. 인라인 검사 솔루션의 경우:
    a. AWS Network Firewall을 사용하는 경우 규칙, 방화벽 정책 및 방화벽 자체를 생성합니다. 구성이 완료되면 트래픽을 방화벽 엔드포인트로 라우팅하여 검사를 활성화할 수 있습니다.

   b. Gateway Load Balancer(GWLB)와 함께 서드파티 어플라이언스를 사용하는 경우 하나 이상의 가용 영역에 어플라이언스를 배포하고 구성합니다. 그런 다음 GWLB, 엔드포인트 서비스, 엔드포인트를 생성하고 트래픽에 대한 라우팅을 구성합니다.

3. 대역 외 검사 솔루션의 경우:

   1. 인바운드 및 아웃바운드 트래픽을 미러링해야 하는 인터페이스에서 VPC Traffic Mirroring을 활성화합니다. Amazon EventBridge 규칙을 사용하여 새 리소스가 생성될 때 인터페이스에서 트래픽 모니터링을 활성화하는 AWS Lambda 함수를 간접 호출할 수 있습니다. Traffic Mirroring 세션이 트래픽을 처리하는 어플라이언스 앞에 있는 Network Load Balancer를 가리키도록 합니다.

4. 인바운드 웹 트래픽 솔루션의 경우:

   a. AWS WAF를 구성하려면 먼저 웹 액세스 제어 목록(웹 ACL)을 구성합니다. 웹 ACL은 순차적으로 처리된 기본 작업(ALLOW 또는 DENY)이 포함된 규칙 모음으로, WAF가 트래픽을 처리하는 방식을 정의합니다. 자체 규칙 및 그룹을 만들거나 웹 ACL에서 AWS 관리형 규칙 그룹을 사용할 수 있습니다.

   b. 웹 ACL이 구성되면 웹 ACL을 AWS 리소스(예: Application Load Balancer, API Gateway REST API 또는 CloudFront 배포)와 연결하여 웹 트래픽 보호를 시작합니다.

리소스

관련 문서:

- What is Traffic Mirroring?
- Implementing inline traffic inspection using third-party security appliances
- AWS Network Firewall example architectures with routing
- Centralized inspection architecture with AWS Gateway Load Balancer and AWS Transit Gateway

관련 예제:

- Gateway Load Balancer 배포 모범 사례
- TLS inspection configuration for encrypted egress traffic and AWS Network Firewall

관련 도구:

- AWS Marketplace IDS/IPS

SEC05-BP04 네트워크 보호 자동화

코드형 인프라(IaC) 및 CI/CD 파이프라인과 같은 DevOps 사례를 사용하여 네트워크 보호 배포를 자동화합니다. 이러한 관행은 버전 제어 시스템을 통해 네트워크 보호의 변경 사항을 추적하고, 변경 사항을 배포하는 데 걸리는 시간을 줄이며, 네트워크 보호가 원하는 구성과 다른지 탐지하는 데 도움이 될 수 있습니다.

원하는 성과: 템플릿을 사용하여 네트워크 보호를 정의하고 이를 버전 제어 시스템에 커밋합니다. 테스트 및 배포를 오케스트레이션하는 새로운 변경이 발생하면 자동화된 파이프라인이 시작됩니다. 배포 전에 변경 사항을 검증하기 위한 정책 검사 및 기타 정적 테스트가 마련되어 있습니다. 스테이징 환경에 변경 사항을 배포하여 제어 기능이 예상대로 작동하는지 확인합니다. 제어가 승인되면 프로덕션 환경으로의 배포도 자동으로 수행됩니다.

일반적인 안티 패턴:

- 개별 워크로드 팀에 의존하여 각자 전체 네트워크 스택, 보호 및 자동화를 정의합니다. 워크로드 팀이 사용할 수 있도록 네트워크 스택 및 보호의 표준 측면을 중앙 집중식으로 게시하지 않습니다.
- 중앙 네트워크 팀에 의존하여 네트워크, 보호 및 자동화의 모든 측면을 정의합니다. 네트워크 스택 및 보호의 워크로드별 측면을 해당 워크로드 팀에 맡기지 않습니다.
- 네트워크 팀과 워크로드 팀 사이에 중앙 집중화와 위임 간의 적절한 균형을 유지하고 있지만, IaC 템플릿 및 CI/CD 파이프라인 전체에서 일관된 테스트 및 배포 표준을 적용하지 않습니다. 템플릿의 준수 여부를 검사하는 도구에서 필수 구성을 캡처하지 않습니다.

이 모범 사례 확립의 이점: 템플릿을 사용하여 네트워크 보호를 정의하면 버전 제어 시스템을 통해 시간 경과에 따른 변경 사항을 추적하고 비교할 수 있습니다. 자동화를 사용하여 변경 사항을 테스트하고 배포하면 표준화와 예측 가능성을 높여 배포가 성공할 확률이 커지고 반복적인 수동 구성 작업을 줄일 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

SEC05-BP02 네트워크 계층 내 트래픽 흐름 제어 및 SEC05-BP03 검사 기반 보호 구현에 설명된 여러 네트워크 보호 제어 기능에는 최신 위협 인텔리전스를 기반으로 자동 업데이트할 수 있는 관리형 규칙 시스템이 함께 제공됩니다. 웹 엔드포인트 보호의 예로는 AWS WAF 관리형 규칙 및 AWS Shield Advanced Shield Advanced 자동 애플리케이션 계층 DDoS 완화가 있습니다. AWS Network Firewall 관리형 규칙 그룹을 사용하여 평판이 낮은 도메인 목록과 위협 서명도 최신 상태로 유지합니다.

관리형 규칙 외에도 DevOps 관행을 적용하여 네트워크 리소스, 보호 및 지정한 규칙의 배포를 자동화하는 것이 좋습니다. 이러한 정의를 AWS CloudFormation 또는 다른 원하는 코드형 인프라(IaC) 도구로 캡처하고 버전 제어 시스템에 커밋하며 CI/CD 파이프라인을 사용하여 배포할 수 있습니다. 이 접근 방식을 사용하면 네트워크 제어 관리 시 DevOps의 전통적인 이점을 누릴 수 있습니다. 예를 들어 릴리스 예측 가능성을 높이고, AWS CloudFormation Guard와 같은 도구를 사용하여 테스트를 자동화하며, 배포된 환경과 원하는 구성 간의 차이를 탐지할 수 있습니다.

SEC05-BP01 네트워크 계층 생성의 일부로 내린 결정에 따라 중앙 관리 접근 방식을 통해 수신, 송신 및 검사 흐름 전용 VPC를 생성할 수 있습니다. AWS Security Reference Architecture(AWS SRA)에서 설명한 대로, 전용 네트워크 인프라 계정에서 이러한 VPC를 정의할 수 있습니다. 유사한 기술을 사용하여 다른 계정의 워크로드, 보안 그룹, AWS Network Firewall 배포, Route 53 Resolver 규칙, DNS 방화벽 구성, 기타 네트워크 리소스에서 사용하는 VPC를 중앙에서 정의할 수 있습니다. AWS Resource Access Manager을 통해 다른 계정과 이러한 리소스를 공유할 수 있습니다. 이 접근 방식을 사용하면 관리할 대상이 하나뿐이므로, 네트워크 제어 기능은 네트워크 계정에 자동으로 테스트하고 배포하는 작업을 간소화할 수 있습니다. 하이브리드 모델에서 이 작업을 수행할 수 있습니다. 하이브리드 모델에서는 특정 제어 기능을 중앙에서 배포 및 공유하고 다른 제어 기능은 개별 워크로드 팀과 해당 계정에 위임할 수 있습니다.

구현 단계

1. 네트워크와 보호의 어떤 부분을 중앙에서 정의하고 워크로드 팀은 어떤 부분을 유지 관리할지에 대한 소유권을 설정합니다.
2. 네트워크 및 보호에 대한 변경 사항을 테스트하고 배포할 수 있는 환경을 만듭니다. 예를 들어, 네트워크 테스트 계정과 네트워크 프로덕션 계정을 사용하세요.
3. 버전 관리 시스템에서 템플릿을 저장하고 유지 관리하는 방법을 결정합니다. 중앙 템플릿은 워크로드 리포지토리와는 다른 리포지토리에 저장하고, 워크로드 템플릿은 해당 워크로드와 관련된 리포지토리에 저장할 수 있습니다.
4. CI/CD 파이프라인을 생성하여 템플릿을 테스트하고 배포합니다. 잘못된 구성이 있는지 점검하고 템플릿이 기업 표준을 준수하는지 확인하는 테스트를 정의합니다.

리소스

관련 모범 사례:

- SEC01-BP06 표준 보안 제어의 배포 자동화

관련 문서:

- AWS Security Reference Architecture - Network account

관련 예제:

- AWS Deployment Pipeline Reference Architecture
- NetDevSecOps to modernize AWS networking deployments
- Integrating AWS CloudFormation security tests with AWS Security Hub and AWS CodeBuild reports

관련 도구:

- AWS CloudFormation
- AWS CloudFormation Guard
- cfn_nag

컴퓨팅 보호

컴퓨팅 리소스에는 EC2 인스턴스, 컨테이너, AWS Lambda 함수, 데이터베이스 서비스, IoT 디바이스 등이 포함됩니다. 이러한 각 컴퓨팅 리소스 유형을 보호하려면 서로 다른 접근 방식이 필요합니다. 그러나 심층 방어, 취약성 관리, 공격 표면 감소, 구성 및 운영 자동화, 원거리 작업 수행 등 고려해야 할 공통 전략은 서로 공유합니다. 이 섹션에서는 주요 서비스의 컴퓨팅 리소스를 보호하기 위한 일반적인 지침을 제공합니다. 사용되는 각 AWS 서비스에 대해 서비스 설명서의 구체적인 보안 권장 사항을 확인해야 합니다.

모범 사례

- SEC06-BP01 취약성 관리 수행
- SEC06-BP02 강화된 이미지로부터 컴퓨팅 프로비저닝
- SEC06-BP03 수동 관리 및 대화형 액세스 감소
- SEC06-BP04 소프트웨어 무결성 검증
- SEC06-BP05 컴퓨팅 보호 자동화

SEC06-BP01 취약성 관리 수행

코드, 종속성 및 인프라에 취약성이 있는지 자주 스캔하고 패치를 적용하여 새로운 위협으로부터 보호합니다.

원하는 성과: 워크로드에서 소프트웨어 취약성, 잠재적 결함 및 의도하지 않은 네트워크 노출을 지속적으로 검사하는 솔루션이 있습니다. 위험 평가 기준에 따라 이러한 취약성을 식별하고, 우선순위를 지정하고, 해결하는 프로세스와 절차를 수립했습니다. 또한 컴퓨팅 인스턴스에 자동 패치 관리를 구현했습니다. 취약성 관리 프로그램은 CI/CD 파이프라인 중에 소스 코드를 스캔하는 솔루션과 함께 소프트웨어 개발 수명 주기에 통합됩니다.

일반적인 안티 패턴:

- 취약성 관리 프로그램이 없습니다.
- 심각도나 위험 회피를 고려하지 않고 시스템 패치 적용을 수행합니다.
- 공급업체에서 제공한 수명 종료(EOL) 날짜가 지난 소프트웨어를 사용합니다.
- 보안 문제를 분석하기 전에 코드를 프로덕션 환경에 배포합니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

취약성 관리는 안전하고 견고한 클라우드 환경을 유지하는 데 있어 중요한 요소입니다. 여기에는 보안 스캔, 문제의 식별 및 우선순위 지정, 식별된 취약성을 해결하기 위한 패치 작업을 포함하는 포괄적인 프로세스가 포함됩니다. 자동화는 잠재적인 문제 및 의도하지 않은 네트워크 노출과 문제 해결 노력이 있는지 워크로드를 지속적으로 스캔할 수 있도록 하기 때문에 이 프로세스에서 중추적 역할을 합니다.

AWS 공동 책임 모델은 취약성 관리를 뒷받침하는 기본 개념입니다. 이 모델에 따르면 AWS는 AWS 서비스를 실행하는 하드웨어, 소프트웨어, 네트워킹 및 시설을 포함한 기본 인프라를 보호할 책임이 있습니다. 반대로 사용자는 서비스와 관련된 Amazon EC2 인스턴스 및 Amazon S3 객체 등 사용자의 데이터, 보안 구성 및 관리 작업을 보호할 책임이 있습니다.

AWS는 취약성 관리 프로그램을 지원하는 다양한 서비스를 제공합니다. Amazon Inspector는 지속적으로 AWS 워크로드에 소프트웨어 취약성과 의도하지 않은 네트워크 액세스가 있는지 스캔하는 한편, AWS Systems Manager Patch Manager는 Amazon EC2 인스턴스 전반의 패치 관리를 지원합니다. 이러한 서비스는 AWS 보안 검사를 자동화하고, 보안 알림을 중앙 집중화하고, 조직의 보안 태세에 대한 포괄적인 보기를 제공하는 클라우드 보안 태세 관리 서비스인 AWS Security Hub와 통합할 수 있습니다. 또한 Amazon CodeGuru Security는 정적 코드 분석을 사용하여 개발 단계에서 Java 및 Python 애플리케이션의 잠재적 문제를 식별합니다.

취약성 관리 사례를 소프트웨어 개발 수명 주기에 통합하면 취약성이 프로덕션 환경에 도입되기 전에 사전 예방적으로 해결할 수 있으므로 보안 이벤트의 위험을 줄이고 취약성의 잠재적 영향을 최소화할 수 있습니다.

구현 단계

1. 공동 책임 모델 이해: AWS 공동 책임 모델을 검토하여 클라우드에서 워크로드와 데이터를 보호하는 책임을 이해합니다. AWS는 기본 클라우드 인프라를 보호하는 역할을 담당하고, 사용자는 애플리케이션, 데이터 및 사용하는 서비스를 보호할 책임이 있습니다.
2. 취약성 스캔 구현: Amazon Inspector와 같은 취약성 스캔 서비스를 구성하여 컴퓨팅 인스턴스(예: 가상 머신, 컨테이너 또는 서버리스 함수)에 소프트웨어 취약성, 잠재적 결함 및 의도하지 않은 네트워크 노출이 있는지 자동으로 검사합니다.
3. 취약성 관리 프로세스 수립: 취약성을 식별하고, 우선순위를 지정하고, 해결하기 위한 프로세스 및 절차를 정의합니다. 여기에는 정기적인 취약성 검사 일정 설정, 위험 평가 기준 설정, 취약성 심각도에 따른 개선 일정 정의가 포함될 수 있습니다.
4. 패치 관리 설정: 패치 관리 서비스를 사용하여 운영 체제 및 애플리케이션에 대한 컴퓨팅 인스턴스 패치 프로세스를 자동화합니다. 누락된 패치가 있는지 인스턴스를 스캔하고 일정에 따라 자동으로 설치하도록 서비스를 구성할 수 있습니다. 이 기능을 제공하려면 AWS Systems Manager Patch Manager를 고려해 보세요.
5. 맬웨어 방지 구성: 환경에서 악성 소프트웨어를 감지하는 메커니즘을 구현합니다. 예를 들어 Amazon GuardDuty와 같은 도구를 사용하여 EC2 및 EBS 볼륨에서 맬웨어를 분석 및 탐지하고 맬웨어에 관해 알림을 보낼 수 있습니다. 또한 GuardDuty는 Amazon S3에 새로 업로드된 객체를 스캔하여 잠재적 맬웨어 또는 바이러스가 있는지 확인하고 다운스트림 프로세스에 수집하기 전에 격리 조치를 취할 수 있습니다.
6. CI/CD 파이프라인에 취약성 스캔 통합: 애플리케이션 배포에 CI/CD 파이프라인을 사용하는 경우 취약성 스캔 도구를 파이프라인에 통합합니다. Amazon CodeGuru Security 및 오픈 소스 옵션과 같은 도구는 소스 코드, 종속성 및 아티팩트에서 잠재적 보안 문제를 스캔할 수 있습니다.
7. 보안 모니터링 서비스 구성: AWS Security Hub와 같은 보안 모니터링 서비스를 설정하여 여러 클라우드 서비스에서 보안 태세를 포괄적으로 확인할 수 있습니다. 이 서비스는 다양한 소스에서 보안 조사 결과를 수집하여 표준화된 형식으로 제시함으로써 우선순위 지정 및 수정을 용이하게 해야 합니다.
8. 웹 애플리케이션 침투 테스트 구현: 애플리케이션이 웹 애플리케이션이고 조직이 필요한 스킬을 갖추고 있거나 외부 지원할 사람을 고용할 수 있는 경우, 애플리케이션의 잠재적 취약성을 식별하기 위해 웹 애플리케이션 침투 테스트를 구현하는 것이 좋습니다.

9. 코드형 인프라로 자동화: AWS CloudFormation과 같은 코드형 인프라(IaC) 도구를 사용하여 앞서 언급한 보안 서비스를 포함하여 리소스의 배포 및 구성을 자동화합니다. 이 방법을 사용하면 여러 계정 및 환경에서 보다 일관되고 표준화된 리소스 아키텍처를 생성할 수 있습니다.
10. 모니터링 및 지속적인 개선: 취약성 관리 프로그램의 효과를 지속적으로 모니터링하고 필요에 따라 개선합니다. 보안 조사 결과를 검토하고, 개선 노력의 효과를 평가하고, 그에 따라 프로세스와 도구를 조정합니다.

리소스

관련 문서:

- AWS Systems Manager
- Security Overview of AWS Lambda
- Amazon CodeGuru
- Improved, Automated Vulnerability Management for Cloud Workloads with a New Amazon Inspector
- Automate vulnerability management and remediation in AWS using Amazon Inspector and AWS Systems Manager – Part 1

관련 비디오:

- Securing Serverless and Container Services
- Security best practices for the Amazon EC2 instance metadata service

SEC06-BP02 강화된 이미지로부터 컴퓨팅 프로비저닝

강화된 이미지에서 배포하여 런타임 환경에 의도치 않게 액세스하는 상황을 줄이세요. 신뢰할 수 있는 레지스트리에서 컨테이너 이미지 및 애플리케이션 라이브러리와 같은 런타임 종속성만 획득하고 해당 서명을 확인합니다. 자체 프라이빗 레지스트리를 생성하여 빌드 및 배포 프로세스에 사용할 신뢰할 수 있는 이미지와 라이브러리를 저장하세요.

원하는 성과: 컴퓨팅 리소스가 강화된 기준 이미지에서 프로비저닝됩니다. 신뢰할 수 있는 레지스트리에서만 컨테이너 이미지 및 애플리케이션 라이브러리와 같은 외부 종속성을 검색하고 해당 서명을 확인합니다. 이러한 정보는 빌드 및 배포 프로세스에서 참조할 수 있도록 프라이빗 레지스트리에 저장됩니다. 이미지와 종속성을 정기적으로 스캔하고 업데이트하여 새로 발견된 취약성으로부터 보호합니다.

일반적인 안티 패턴:

- 신뢰할 수 있는 레지스트리에서 이미지와 라이브러리를 가져오지만, 사용하기 전에 서명을 확인하거나 취약성 스캔을 수행하지는 않습니다.
- 이미지를 강화하지만, 정기적으로 새로운 취약성을 테스트하거나 최신 버전으로 업데이트하지는 않습니다.
- 이미지의 예상 수명 주기 동안 필요하지 않은 소프트웨어 패키지를 설치하거나 제거하지 않습니다.
- 프로덕션 컴퓨팅 리소스를 최신 상태로 유지하는 데 패치 작업에만 의존합니다. 패치만 적용하면 시간이 지나면서 컴퓨팅 리소스가 강화된 표준에서 벗어날 수 있습니다. 또한, 패치를 적용해도 보안 이벤트 중에 위협 행위자가 설치했을 수 있는 맬웨어를 제거하지 못할 수 있습니다.

이 모범 사례 확립의 이점: 이미지를 강화하면 런타임 환경에서 승인되지 않은 사용자나 서비스에 의도하지 않은 액세스를 허용할 수 있는 경로의 수를 줄일 수 있습니다. 또한, 의도하지 않은 액세스가 발생할 경우 영향을 받는 범위를 줄일 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

시스템을 강화하려면 최신 버전의 운영 체제, 컨테이너 이미지, 애플리케이션 라이브러리부터 시작하세요. 알려진 문제에 패치를 적용합니다. 불필요한 애플리케이션, 서비스, 디바이스 드라이버, 기본 사용자 및 기타 자격 증명을 제거하여 시스템을 최소화합니다. 워크로드에 필요한 리소스와 기능만 갖춘 환경을 만들기 위해 포트를 비활성화하는 등 필요한 기타 조치를 수행합니다. 이 기준에서 워크로드 모니터링 또는 취약성 관리와 같은 목적에 필요한 소프트웨어, 에이전트 또는 기타 프로세스를 설치할 수 있습니다.

CIS(Center for Internet Security) 및 DISA(Defense Information Systems Agency) STIG(Security Technical Implementation Guide) 등 신뢰할 수 있는 소스에서 제공하는 지침을 활용하여 시스템 강화에 대한 부담을 덜 수 있습니다. 먼저 AWS 또는 APN 파트너가 게시한 Amazon Machine Image(AMI)로 시작하여 CIS 및 STIG 제어의 적절한 조합에 따라 AWS EC2 Image Builder를 사용하여 구성을 자동화하는 것이 좋습니다.

CIS 또는 DISA STIG 권장 사항을 적용하는 강화된 이미지와 EC2 Image Builder 레시피가 제공되지만, 이러한 구성으로 인해 소프트웨어가 제대로 실행되지 않을 수 있습니다. 이 경우 강화되지 않은 기본 이미지에서 시작하여 소프트웨어를 설치한 다음, CIS 제어 기능을 점진적으로 적용하여 영향을 테스트하면 됩니다. 소프트웨어 실행을 방해하는 CIS 제어 기능의 경우, 대신 DISA에서 세분화된 강화 권장 사항을 구현할 수 있는지 테스트하세요. 성공적으로 적용할 수 있는 다양한 CIS 제어 기능 및

DISA STIG 구성을 추적합니다. 이를 사용하여 EC2 Image Builder의 이미지 강화 레시피를 적절하게 정의합니다.

컨테이너식 워크로드의 경우 Docker의 강화된 이미지는 Amazon Elastic Container Registry(ECR) 퍼블릭 리포지토리에서 사용할 수 있습니다. EC2 Image Builder를 사용하여 AMI와 함께 컨테이너 이미지를 강화할 수 있습니다.

운영 체제 및 컨테이너 이미지와 마찬가지로 pip, npm, Maven 및 NuGet과 같은 도구를 통해 퍼블릭 리포지토리에서 코드 패키지(또는 라이브러리)를 가져올 수 있습니다. AWS CodeArtifact 내부와 같은 프라이빗 리포지토리를 신뢰할 수 있는 퍼블릭 리포지토리와 통합하여 코드 패키지를 관리하는 것이 좋습니다. 이 통합을 통해 패키지를 검색 및 저장하고 최신 상태로 유지할 수 있습니다. 그러면 애플리케이션 빌드 프로세스에서 소프트웨어 구성 분석(SCA), 정적 애플리케이션 보안 테스트(SAST), 동적 애플리케이션 보안 테스트(DAST)와 같은 기술을 사용하여 애플리케이션과 함께 이러한 패키지의 최신 버전을 구해 테스트할 수 있습니다.

AWS Lambda를 사용하는 서버리스 워크로드의 경우 Lambda 계층을 사용하여 패키지 종속성 관리를 간소화합니다. Lambda 계층을 사용하여 여러 기능에서 공유되는 표준 종속성 집합을 독립형 아카이브로 구성합니다. 자체 빌드 프로세스를 통해 함수를 중앙에서 최신 상태로 유지하는 방법을 제공하며 계층을 생성하고 유지 관리할 수 있습니다.

구현 단계

- 운영 체제를 강화합니다. 신뢰할 수 있는 소스의 기본 이미지를 기반으로 강화된 AMI를 구축합니다. EC2 Image Builder를 사용하면 이미지에 설치된 소프트웨어를 사용자 지정하는 데 도움이 됩니다.
- 컨테이너식 리소스를 강화합니다. 보안 모범 사례에 맞춰 컨테이너식 리소스를 구성합니다. 컨테이너를 사용할 때는 빌드 파이프라인에서 이미지 리포지토리에 대해 정기적으로 ECR Image Scanning을 구현하여 컨테이너에서 CVE를 찾습니다.
- AWS Lambda를 통해 서버리스 구현을 사용하는 경우 Lambda 계층을 사용하여 애플리케이션 함수 코드와 공유 종속 라이브러리를 분리합니다. 신뢰할 수 있는 코드만 Lambda 함수에서 실행되도록 Lambda에 대한 코드 서명을 구성합니다.

리소스

관련 모범 사례:

- OPS05-BP05 패치 관리 수행

관련 비디오:

- Deep dive into AWS Lambda security

관련 예제:

- Quickly build STIG-compliant AMI using EC2 Image Builder
- Building better container images
- Using Lambda layers to simplify your development process
- Develop & Deploy AWS Lambda Layers using Serverless Framework
- Building end-to-end AWS DevSecOps CI/CD pipeline with open source SCA, SAST and DAST tools

SEC06-BP03 수동 관리 및 대화형 액세스 감소

자동화를 사용하여 가능한 모든 곳에서 배포, 구성, 유지 관리 및 조사 작업을 수행합니다. 긴급 절차나 안전한(샌드박스) 환경에서 자동화를 사용할 수 없는 경우에는 컴퓨팅 리소스에 수동으로 액세스하는 것이 좋습니다.

원하는 성과: 프로그래밍 스크립트와 자동화 문서(런북)가 컴퓨팅 리소스에서 승인된 작업을 캡처합니다. 이러한 런북은 변경 탐지 시스템을 통해 자동으로 시작될 수도, 사람의 판단이 필요할 때는 수동으로 시작될 수도 있습니다. 컴퓨팅 리소스에 대한 직접 액세스는 자동화를 사용할 수 없는 긴급 상황에서만 지원됩니다. 모든 수동 활동이 로깅되고 검토 프로세스에 통합되어 자동화 기능을 지속적으로 개선합니다.

일반적인 안티 패턴:

- SSH 또는 RDP와 같은 프로토콜을 사용하여 Amazon EC2 인스턴스에 대화식으로 액세스합니다.
- `/etc/passwd` 또는 Windows 로컬 사용자와 같은 개별 사용자 로그인을 유지 관리합니다.
- 여러 사용자 간에 인스턴스에 액세스하기 위한 암호 또는 프라이빗 키를 공유합니다.
- 수동으로 소프트웨어를 설치하고 구성 파일을 만들거나 업데이트합니다.
- 수동으로 소프트웨어를 업데이트하거나 패치를 적용합니다.
- 문제 해결을 위해 인스턴스에 로그인합니다.

이 모범 사례 확립의 이점: 자동화를 통해 작업을 수행하면 의도하지 않은 변경 및 잘못된 구성으로 인한 운영 위험을 줄일 수 있습니다. 대화형 액세스에 Secure Shell(SSH) 및 Remote Desktop Protocol(RDP) 사용을 제거하면 컴퓨팅 리소스에 대한 액세스 범위가 줄어듭니다. 이렇게 하면 무단

행위가 발생하는 일반적인 경로가 사라집니다. 자동화 문서 및 프로그래밍 스크립트에 컴퓨팅 리소스 관리 작업을 캡처하면 승인된 활동의 전체 범위를 세밀하게 정의하고 감사할 수 있는 메커니즘이 제공됩니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

인스턴스에 로그인하는 것은 전형적인 시스템 관리 접근 방식입니다. 사용자는 일반적으로 서버 운영 체제를 설치한 후 수동으로 로그인하여 시스템을 구성하고 원하는 소프트웨어를 설치합니다. 서버 수명 기간에 사용자는 로그인하여 소프트웨어 업데이트를 수행하고, 패치를 적용하며, 구성을 변경하고, 문제를 해결할 수 있습니다.

그러나 수동으로 액세스하면 여러 위험이 따릅니다. 무단 액세스에 대한 잠재적 경로를 제공할 수 있는 SSH 또는 RDP 서비스와 같은 요청을 수신하는 서버를 필요로 합니다. 또한, 수동 단계 수행과 관련되어 인적 오류가 발생할 위험도 증가합니다. 이로 인해 워크로드 인시던트, 데이터 손상 또는 폐기를 비롯하여 기타 보안 문제가 발생할 수 있습니다. 나아가 사람이 액세스하려면 자격 증명 공유에 대한 보호가 필요하므로, 관리 오버헤드가 가중됩니다.

이러한 위험을 완화하기 위해 AWS Systems Manager와 같은 에이전트 기반 원격 액세스 솔루션을 구현할 수 있습니다. AWS Systems Manager 에이전트(SSM 에이전트)는 암호화된 채널을 시작하므로, 외부에서 시작된 요청을 수신 대기하는 데 의존하지 않습니다. VPC 엔드포인트를 통해 이 채널을 설정하도록 SSM 에이전트를 구성하는 방법을 고려하세요.

Systems Manager를 사용하면 관리형 인스턴스와 상호 작용하는 방법을 세밀하게 제어할 수 있습니다. 실행할 자동화, 실행할 수 있는 사용자, 실행할 수 있는 시기를 정의합니다. Systems Manager는 인스턴스에 대한 대화형 액세스 없이 패치를 적용하고, 소프트웨어를 설치하며, 구성을 변경할 수 있습니다. 또한 Systems Manager는 원격 쉘에 액세스하여 세션 중에 간접 호출된 모든 명령과 해당 출력을 로그 및 Amazon S3에 로깅할 수 있습니다. AWS CloudTrail은 검사 목적으로 Systems Manager API 간접 호출을 기록합니다.

구현 단계

1. Amazon EC2 인스턴스에 AWS Systems Manager 에이전트(SSM 에이전트)를 설치합니다. SSM Agent가 기본 AMI 구성의 일부로 포함되고 자동으로 시작되는지 확인합니다.
2. EC2 인스턴스 프로파일과 연결된 IAM 역할에 `AmazonSSMManagedInstanceCore` 관리형 IAM 정책이 포함되어 있는지 확인합니다.

3. 인스턴스에서 실행 중인 SSH, RDP 및 기타 원격 액세스 서비스를 비활성화합니다. 시작 템플릿의 사용자 데이터 섹션에 구성된 스크립트를 실행하거나 EC2 Image Builder와 같은 도구를 사용하여 사용자 지정 AMI를 구축하면 됩니다.
4. EC2 인스턴스에 적용되는 보안 그룹 수신 규칙이 포트 22/tcp(SSH) 또는 포트 3389/tcp(RDP)에서의 액세스를 허용하지 않는지 확인합니다. AWS Config 등의 서비스를 사용하여 잘못 구성된 보안 그룹에 대한 탐지 및 알림을 구현합니다.
5. Systems Manager에서 적절한 자동화, 런북을 정의하고 명령을 실행합니다. IAM 정책을 사용하여 이러한 작업을 수행할 수 있는 사람과 작업이 허용되는 조건을 정의합니다. 프로덕션 환경이 아닌 환경에서 자동화를 철저하게 테스트하세요. 필요한 경우 대화형 방식으로 인스턴스에 액세스하지 않고 자동화를 간접 호출할 수 있습니다.
6. 필요한 경우 AWS Systems Manager Session Manager를 사용하여 인스턴스에 대한 대화형 액세스를 제공합니다. 세션 활동 로깅을 활성화하여 Amazon CloudWatch Logs 또는 Amazon S3에서 감사 기록을 유지 관리합니다.

리소스

관련 모범 사례:

- REL08-BP04 변경 불가능한 인프라를 사용하여 배포

관련 예제:

- Replacing SSH access to reduce management and security overhead with AWS Systems Manager

관련 도구:

- AWS Systems Manager

관련 비디오:

- Controlling User Session Access to Instances in AWS Systems Manager Session Manager

SEC06-BP04 소프트웨어 무결성 검증

암호화 검증을 사용하여 워크로드에서 사용하는 소프트웨어 아티팩트(이미지 포함)의 무결성을 검증합니다. 컴퓨팅 환경 내에서 실행되는 무단 변경을 방지하기 위해 소프트웨어를 암호화 방식으로 서명합니다.

원하는 성과: 모든 아티팩트를 신뢰할 수 있는 소스에서 가져옵니다. 공급업체 웹 사이트 인증서가 검증되었습니다. 다운로드한 아티팩트는 서명을 통해 암호화 방식으로 확인됩니다. 자체 소프트웨어는 컴퓨팅 환경에서 암호화 방식으로 서명되고 확인됩니다.

일반적인 안티 패턴:

- 믿을 수 있는 공급업체 웹 사이트를 신뢰하여 소프트웨어 아티팩트를 가져오지만, 인증서 만료 통지는 무시합니다. 인증서가 유효한지 확인하지 않고 다운로드를 진행합니다.
- 공급업체 웹 사이트 인증서를 검증하지만, 해당 웹 사이트에서 다운로드한 아티팩트를 암호화 방식으로 확인하지는 않습니다.
- 요약 또는 해시에만 의존하여 소프트웨어 무결성을 검증합니다. 해시는 아티팩트가 원본 버전에서 수정되지 않았음을 확인하지만, 소스를 검증하지는 않습니다.
- 자체 배포에서만 사용하는 경우에도 자체 소프트웨어, 코드 또는 라이브러리에 서명하지 않습니다.

이 모범 사례 확립의 이점: 워크로드가 의존하는 아티팩트의 무결성을 검증하면 맬웨어가 컴퓨팅 환경에 침입하는 것을 방지할 수 있습니다. 소프트웨어에 서명하면 컴퓨팅 환경에서 무단 실행되지 않도록 보호하는 데 도움이 됩니다. 코드 서명 및 확인을 통해 소프트웨어 공급망을 보호하세요.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

운영 체제 이미지, 컨테이너 이미지 및 코드 아티팩트는 요약이나 해시와 같은 무결성 검사가 가능한 상태로 배포되는 경우가 많습니다. 이를 통해 클라이언트는 페이로드의 자체 해시를 계산하고 게시된 것과 동일한지 검증하여 무결성을 확인할 수 있습니다. 이러한 검사는 페이로드가 변조되지 않았음을 확인하는 데 도움이 되지만, 페이로드가 원본 소스(출처)에서 왔는지 검증하지는 않습니다. 출처를 확인하려면 신뢰할 수 있는 기관에서 아티팩트에 디지털 서명하기 위해 발급한 인증서가 필요합니다.

워크로드에서 다운로드한 소프트웨어 또는 아티팩트를 사용하는 경우 제공업체가 디지털 서명 확인을 위한 퍼블릭 키를 제공하는지 확인하세요. AWS에서 당사가 게시하는 소프트웨어에 대한 퍼블릭 키 및 확인 지침을 제공하는 방법의 몇 가지 예는 다음과 같습니다.

- EC2 Image Builder: Verify the signature of the AWSTOE installation download

- AWS Systems Manager: SSM 에이전트의 서명 확인
- Amazon CloudWatch: Verifying the signature of the CloudWatch agent package

SEC06-BP02 강화된 이미지로부터 컴퓨팅 프로비저닝에서 설명한 대로 이미지 획득 및 강화에 사용하는 프로세스에 디지털 서명 검증을 통합합니다.

AWS Signer를 사용하여 서명 검증은 물론, 자체 소프트웨어 및 아티팩트에 대한 자체 코드 서명 수명 주기를 관리하는 데 도움이 될 수 있습니다. AWS Lambda 및 Amazon Elastic Container Registry 모두 코드 및 이미지의 서명 확인을 위해 Signer와의 통합을 제공합니다. 리소스 섹션의 예제를 바탕으로 지속적 통합 및 지속적 전달(CI/CD) 파이프라인에 Signer를 통합하여 서명 검증과 자체 코드 및 이미지 서명을 자동화할 수 있습니다.

리소스

관련 문서:

- Cryptographic Signing for Containers
- Best Practices to help secure your container image build pipeline by using AWS Signer
- Announcing Container Image Signing with AWS Signer and Amazon EKS
- AWS Lambda에 대한 코드 서명 구성
- Best practices and advanced patterns for Lambda code signing
- Code signing using AWS Certificate Manager Private CA and AWS Key Management Service asymmetric keys

관련 예제:

- Automate Lambda code signing with Amazon CodeCatalyst and AWS Signer
- Signing and Validating OCI Artifacts with AWS Signer

관련 도구:

- AWS Lambda
- AWS Signer
- AWS Certificate Manager
- AWS Key Management Service

- AWS CodeArtifact

SEC06-BP05 컴퓨팅 보호 자동화

컴퓨팅 보호 운영을 자동화하여 사람이 개입할 필요성을 줄입니다. 자동 스캔을 사용하여 컴퓨팅 리소스 내의 잠재적 문제를 탐지하고 자동화된 프로그래밍 방식 응답 또는 플릿 관리 작업을 통해 문제를 해결하세요. CI/CD 프로세스에 자동화를 통합하여 최신 종속성을 갖춘 신뢰할 수 있는 워크로드를 배포하세요.

원하는 성과: 자동화된 시스템이 컴퓨팅 리소스의 모든 스캔 및 패치를 수행합니다. 자동 검증을 사용하여 소프트웨어 이미지와 종속성이 신뢰할 수 있는 소스에서 비롯되었으며 변조되지 않았는지 확인합니다. 워크로드는 자동으로 최신 종속성을 확인하고 AWS 컴퓨팅 환경에서 신뢰성을 확보하도록 서명됩니다. 규정을 준수하지 않는 리소스가 탐지되면 자동 수정이 시작됩니다.

일반적인 안티 패턴:

- 변경 불가능한 인프라의 관행을 따르고 있지만, 프로덕션 시스템의 긴급 패치 적용 또는 교체를 위한 솔루션이 없습니다.
- 자동화를 사용하여 잘못 구성된 리소스를 수정하지만, 수동 재정의 메커니즘은 없습니다. 요구 사항을 조정해야 하는 상황이 발생할 수 있으며, 이러한 변경을 수행할 때까지 자동화를 일시 중단해야 할 수 있습니다.

이 모범 사례 확립의 이점: 자동화를 통해 컴퓨팅 리소스의 무단 액세스 및 사용 위험을 줄일 수 있습니다. 이를 통해 프로덕션 환경에 잘못된 구성이 유입되는 것을 방지하고 잘못된 구성이 발생할 경우 이를 탐지하여 수정할 수 있습니다. 자동화는 또한 무단 액세스 및 컴퓨팅 리소스 사용을 탐지하여 대응 시간을 줄이는 데 도움이 됩니다. 결과적으로 문제로 인한 전체 영향 범위를 줄일 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

보안 원칙 사례에 설명된 자동화를 적용하여 컴퓨팅 리소스를 보호할 수 있습니다. SEC06-BP01 취약성 관리 수행에서는 CI/CD 파이프라인에서 그리고 런타임 환경에서 알려진 일반적인 취약성 및 노출(CVE)을 지속적으로 스캔하는 경우에 Amazon Inspector를 사용하는 방법을 설명합니다. AWS Systems Manager를 사용하면 패치를 적용하거나 자동화된 런북을 통해 최신 이미지에서 재배포하여 컴퓨팅 플릿을 최신 소프트웨어 및 라이브러리로 업데이트할 수 있습니다. 이러한 기술을 사용하면 수동 프로세스는 물론 컴퓨팅 리소스에 대한 대화형 액세스의 필요성을 줄일 수 있습니다. 자세한 내용은 SEC06-BP03 수동 관리 및 대화형 액세스 감소를 참조하세요.

또한 자동화는 신뢰할 수 있는 워크로드를 배포하는 데도 중요한 역할을 합니다(SEC06-BP02 강화된 이미지로부터 컴퓨팅 프로비저닝 및 SEC06-BP04 소프트웨어 무결성 검증 참조). EC2 Image Builder, AWS Signer, AWS CodeArtifact, Amazon Elastic Container Registry(ECR)와 같은 서비스를 사용하여 강화 및 승인된 이미지와 코드 종속성을 다운로드, 확인, 구성 및 저장할 수 있습니다. Inspector와 함께 이들 각각은 CI/CD 프로세스에서 역할을 수행할 수 있으므로, 종속성이 신뢰할 수 있는 출처에서 비롯된 최신 상태인 점이 확인된 경우에만 워크로드가 프로덕션에 전달됩니다. 또한 AWS Lambda 및 Amazon Elastic Kubernetes Service(EKS)와 같은 AWS 컴퓨팅 환경에서 워크로드가 실행되기 전에 변조되지 않았는지 확인할 수 있도록 워크로드가 서명됩니다.

이러한 예방적 제어 외에도 컴퓨팅 리소스에 대한 탐지 제어에서도 자동화를 사용할 수 있습니다. 한 가지 예로, AWS Security Hub에서는 NIST 800-53 Rev. 5 표준을 제공합니다. 이 표준은 [EC2.8] EC2 인스턴스가 인스턴스 메타데이터 서비스 버전 2(IMDSv2)를 사용해야 한다는 등의 검사를 포함합니다. IMDSv2는 세션 인증 기술을 사용하여 X-Forwarded-For HTTP 헤더를 포함하는 요청을 차단하고 네트워크 TTL 1을 사용하여 외부 소스에서 발생하는 트래픽을 중지하고 EC2 인스턴스에 대한 정보를 검색합니다. Security Hub의 검사는 EC2 인스턴스가 IMDSv1을 사용하는 시기를 탐지하고 자동 수정을 시작할 수 있습니다. SEC04-BP04 규정 미준수 리소스 관련 문제 해결 시작에서 자동 탐지 및 수정에 대해 자세히 알아보세요.

구현 단계

1. EC2 Image Builder를 사용하여 안전하고 규정을 준수하며 강화된 AMI 생성을 자동화합니다. 기본 AWS 및 APN 파트너 이미지의 Center for Internet Security(CIS) Benchmarks 또는 Security Technical Implementation Guide(STIG) 표준의 제어를 통합하여 이미지를 생성할 수 있습니다.
2. 구성 관리를 자동화합니다. 구성 관리 서비스 또는 도구를 사용하여 컴퓨팅 리소스의 보안 구성을 자동으로 적용하고 검증합니다.
    a. AWS Config를 사용한 자동화된 구성 관리
    b. AWS Security Hub를 사용한 자동화된 보안 및 규정 준수 상태 관리
3. Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 패치 또는 교체를 자동화합니다. AWS Systems Manager Patch Manager는 보안 관련 업데이트와 기타 유형의 업데이트를 모두 사용하여 관리형 인스턴스를 패치하는 프로세스를 자동화합니다. 패치 관리자를 사용하면 운영 체제와 애플리케이션 모두에 패치를 적용할 수 있습니다.
    a. AWS Systems Manager Patch Manager
4. 일반적인 취약성 및 노출(CVE)에 대한 컴퓨팅 리소스 스캔을 자동화하고 빌드 파이프라인에 보안 스캔 솔루션을 포함시킵니다.
    a. Amazon Inspector
    b. ECR 이미지 스캔

5. 컴퓨팅 리소스를 보호하기 위한 자동 맬웨어 및 위협 탐지 기능에 대해 Amazon GuardDuty를 고려하세요. 또한 GuardDuty는 AWS 환경에서 AWS Lambda 함수가 간접 호출될 때 잠재적인 문제를 식별할 수 있습니다.
   a. Amazon GuardDuty
6. AWS 파트너 솔루션을 고려해 보세요. AWS 파트너는 사용자 온프레미스 환경의 기존 제어 솔루션과 동등한 수준이거나, 동일하거나, 통합된 업계 최고 수준의 제품을 제공합니다. 이러한 제품은 기존 AWS 서비스를 보완하여 클라우드 및 온프레미스 환경에 포괄적인 보안 아키텍처와 보다 원활한 환경을 배포할 수 있도록 해줍니다.
   a. 인프라 보안

리소스

관련 모범 사례:

- SEC01-BP06 표준 보안 제어의 배포 자동화

관련 문서:

- Get the full benefits of IMDSv2 and disable IMDSv1 across your AWS infrastructure

관련 비디오:

- Security best practices for the Amazon EC2 instance metadata service

데이터 보호

워크로드를 설계하려면 먼저 보안과 관련된 기본적인 관행부터 마련해야 합니다. 예를 들어 데이터 분류는 민감도에 따라 데이터를 구분하는 하나의 방법이고 암호화는 무단 액세스 사용자가 데이터를 해석하지 못하게 만들어 데이터를 보호하는 방법입니다. 이러한 방법이 중요한 이유는, 조작 부주의 방지 또는 규제 의무 사항 준수와 같은 목표의 달성을 지원하기 때문입니다.

AWS에서는 데이터 보호를 수행할 때 사용할 수 있는 여러 가지 방식이 있습니다. 다음 섹션에서는 이러한 접근 방식을 사용하는 방법을 설명합니다.

주제

- 데이터 분류
- 저장 데이터 보호
- 전송 중 데이터 보호

데이터 분류

데이터 분류는 적절한 보호 및 보존 제어를 결정하는 데 도움이 되도록 중요도를 기준으로 조직 데이터를 분류하는 방법을 제공합니다.

모범 사례

- SEC07-BP01 데이터 분류 체계 이해
- SEC07-BP02 데이터 민감도에 따라 데이터 보호 제어 적용
- SEC07-BP03 식별 및 분류 자동화
- SEC07-BP04 확장 가능한 데이터 수명 주기 관리 정의

SEC07-BP01 데이터 분류 체계 이해

워크로드에서 처리 중인 데이터의 분류, 처리 요구 사항, 관련 비즈니스 프로세스, 데이터가 저장되는 위치, 데이터 소유자가 누구인지 이해하세요. 데이터 분류 및 처리 체계는 워크로드의 적용 가능한 법률 및 규정 준수 요구 사항과 필요한 데이터 제어 기능을 고려해야 합니다. 데이터를 이해해야 데이터 분류 여정을 시작할 수 있습니다.

원하는 성과: 워크로드에 있는 데이터 유형이 잘 이해되고 문서화되어 있습니다. 분류에 따라 민감한 데이터를 보호하기 위한 적절한 제어 조치가 마련되어 있습니다. 이러한 제어 기능은 누가 어떤 목적

으로 데이터에 액세스할 수 있는지는 물론 데이터가 저장되는 위치, 해당 데이터에 대한 암호화 정책 및 암호화 키가 관리되는 방식, 데이터의 수명 주기 및 보존 요구 사항, 적절한 폐기 프로세스, 마련된 백업 및 복구 프로세스, 액세스 감사와 같은 고려 사항을 관리합니다.

일반적인 안티 패턴:

- 데이터 민감도 수준과 처리 요구 사항을 정의하는 공식적인 데이터 분류 정책이 마련되어 있지 않습니다.
- 워크로드 내 데이터의 민감도 수준을 제대로 이해하지 못하고 아키텍처 및 운영 문서에 해당 정보를 포함하지 않습니다.
- 데이터 분류 및 처리 정책에 명시된 대로 데이터의 민감도 및 요구 사항을 기반으로 데이터에 적절한 제어 기능을 적용하지 않습니다.
- 정책 소유자에게 데이터 분류 및 처리 요구 사항에 대한 피드백을 제공하지 않습니다.

이 모범 사례 확립의 이점: 이 방법을 사용하면 워크로드 내에서 데이터의 적절한 처리와 관련된 모호성이 사라집니다. 조직 내 데이터의 민감도 수준과 필수 보호 조치를 정의하는 공식 정책을 적용하면 법규와 기타 사이버 보안 증명 및 인증을 준수하는 데 도움이 될 수 있습니다. 워크로드 소유자는 민감한 데이터가 어디에 저장되고 어떤 보호 제어 기능이 적용되는지 알고 안심할 수 있습니다. 이러한 내용을 문서화하면 신입 팀원의 이해도를 높이고 업무를 배정받은 초반에도 제어 수준을 유지하도록 도울 수 있습니다. 그리고 각 데이터 유형에 대한 제어 규모를 적절하게 조정하여 비용을 절감하는 데도 도움이 될 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

워크로드를 설계할 때 민감한 데이터를 직관적으로 보호하는 방법을 고려할 수 있습니다. 예를 들어, 다중 테넌트 애플리케이션에서는 직관적으로 각 테넌트의 데이터를 민감한 것으로 생각하고 한 테넌트가 다른 테넌트의 데이터에 액세스할 수 없도록 보호 기능을 적용합니다. 마찬가지로 관리자만 데이터를 수정할 수 있고 다른 사용자는 읽기 수준의 액세스 권한만 가지거나 전혀 액세스할 수 없도록 액세스 제어를 직관적으로 설계할 수 있습니다.

이러한 데이터 민감도 수준을 데이터 보호 요구 사항과 함께 정의하고 정책에 포함하면 워크로드에 있는 데이터를 공식적으로 식별할 수 있습니다. 그런 다음 올바른 제어 기능이 있는지, 이를 감사할 수 있는지, 데이터가 잘못 처리되는 경우 어떻게 대응해야 적절한지 결정할 수 있습니다.

워크로드 내에서 민감한 데이터가 있는 위치를 식별하는 데 도움이 되도록 데이터 카탈로그를 사용하는 것이 좋습니다. 데이터 카탈로그는 조직의 데이터, 위치, 민감도 수준 및 해당 데이터를 보호하기 위

해 마련된 제어를 매핑하는 데이터베이스입니다. 또한 가능한 경우 리소스 태그를 사용하는 것도 고려해 보세요. 예를 들어, 보호 의료 정보(PHI)에서 `Classification`의 태그 키와 `PHI`의 태그 값이 있는 태그 그리고 `Sensitivity`의 태그 키와 `High`의 태그 값이 있는 다른 태그를 적용할 수 있습니다. 그런 다음 AWS Config와 같은 서비스를 사용하여 리소스에서 변경 여부를 모니터링하고 보호 요구 사항을 준수하지 못하게 하는 방식으로 수정된 경우(예: 암호화 설정 변경) 알림을 보낼 수 있습니다. AWS Organizations의 기능인 태그 정책을 사용하여 태그 키의 표준 정의와 허용 가능한 값을 캡처할 수 있습니다. 태그 키 또는 값에 개인 데이터나 민감한 데이터는 포함하지 않는 것이 좋습니다.

구현 단계

1. 조직의 데이터 분류 체계 및 보호 요구 사항을 이해합니다.
2. 워크로드에서 처리하는 민감한 데이터의 유형을 식별합니다.
3. 조직에서 데이터가 있는 위치와 해당 데이터의 민감도 수준에 대한 단일 보기를 제공하는 데이터 카탈로그에서 데이터를 캡처합니다.
4. 가능한 경우 리소스 및 데이터 수준 태그 지정을 사용하여 민감도 수준과 모니터링 및 인시던트 대응에 도움이 될 수 있는 기타 운영 메타데이터로 데이터에 태그를 지정하는 것을 고려해 보세요.
   a. AWS Organizations 태그 정책은 태그 지정 표준을 적용하는 데 사용할 수 있습니다.

리소스

관련 모범 사례:

- SUS04-BP01 데이터 분류 정책 구현

관련 문서:

- Data Classification 백서
- Best Practices for Tagging AWS Resources

관련 예제:

- AWS Organizations Tag Policy Syntax and Examples

관련 도구

- AWS Tag Editor

SEC07-BP02 데이터 민감도에 따라 데이터 보호 제어 적용

분류 정책에 정의된 각 데이터 클래스에 대해 적절한 수준의 제어를 제공하는 데이터 보호 제어 기능을 적용합니다. 이렇게 하면 데이터의 가용성과 사용을 유지하면서 민감한 데이터를 무단 액세스 및 사용으로부터 보호할 수 있습니다.

원하는 성과: 조직의 데이터에 대한 다양한 민감도 수준을 정의하는 분류 정책이 있습니다. 이러한 민감도 수준 각각에 대해 승인된 보관 및 취급 서비스, 위치 및 필수 구성과 관련하여 명확한 지침이 게시되어 있습니다. 필요한 보호 수준 및 관련 비용에 따라 각 수준에 대해 제어 기능을 구현합니다. 데이터가 승인되지 않은 위치에 존재하거나, 무허가 환경에서 처리되거나, 무단 행위자가 액세스하거나, 관련 서비스의 구성이 규정을 준수하지 않는 경우 등을 탐지할 수 있는 모니터링 및 알림 기능을 갖추고 있습니다.

일반적인 안티 패턴:

- 모든 데이터에 동일한 수준의 보호 제어 기능을 적용합니다. 이로 인해 민감도가 낮은 데이터에 대한 보안 제어가 과도하게 프로비저닝되거나 매우 민감한 데이터에 대한 보호가 불충분해질 수 있습니다.
- 데이터 보호 제어를 정의할 때 보안, 규정 준수 및 비즈니스 팀의 관련 이해관계자를 참여시키지 않습니다.
- 데이터 보호 제어의 구현 및 유지 관리와 관련된 운영 오버헤드 및 비용을 간과합니다.
- 분류 정책을 준수하기 위해 정기적인 데이터 보호 제어 검토를 수행하지 않습니다.
- 저장 상태일 때와 전송 중에 데이터의 위치에 대한 완전한 인벤토리가 없습니다.

이 모범 사례 확립의 이점: 데이터 분류 수준에 맞게 제어를 조정하면 조직은 필요한 경우 더 높은 수준의 제어에 투자할 수 있습니다. 여기에는 보안, 모니터링, 측정, 문제 해결 및 보고에 대한 리소스 확충이 포함될 수 있습니다. 적절한 제어 수단이 적을수록 직원, 고객 또는 구성원을 위한 데이터의 접근성과 완전성을 개선할 수 있습니다. 이 접근 방식을 통해 조직은 데이터 보호 요구 사항을 준수하면서 데이터를 최대한 유연하게 사용할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

데이터 민감도 수준을 기반으로 데이터 보호 제어 기능을 구현하려면 몇 가지 주요 단계를 거쳐야 합니다. 먼저 워크로드 아키텍처 내의 다양한 데이터 민감도 수준(예: 공개, 내부, 기밀, 제한)을 식별하고 이 데이터를 저장하고 처리하는 위치를 평가합니다. 다음으로 민감도 수준에 따라 데이터 주변의 격리 경계를 정의합니다. 서비스 제어 정책(SCP)을 통해 데이터 민감도 수준별로 허용되는 서비스 및 작업을

제한하여 데이터를 서로 다른 AWS 계정 계정으로 분리하는 것이 좋습니다. 이렇게 하면 강력한 격리 경계를 만들고 최소 권한 원칙을 적용할 수 있습니다.

격리 경계를 정의한 후 데이터 민감도 수준에 따라 적절한 보호 제어 기능을 구현합니다. 암호화, 액세스 제어 및 감사와 같은 관련 제어 기능을 구현하려면 저장 데이터 보호 및 전송 중 데이터 보호에 대한 모범 사례를 참조하세요. 토큰화나 익명화와 같은 기법을 고려하여 데이터의 민감도를 낮춥니다. 토큰화 및 탈토큰화를 위한 중앙 집중식 시스템을 통해 기업 전체에 일관된 데이터 정책을 간편하게 적용할 수 있습니다.

구현된 제어 기능의 효과를 지속적으로 모니터링하고 테스트합니다. 진화하는 조직 데이터 환경과 위협에 발맞춰 데이터 분류 체계, 위험 평가 및 보호 제어 기능을 정기적으로 검토하고 업데이트하세요. 구현된 데이터 보호 제어 기능을 관련 산업 규제, 표준 및 법적 요구 사항에 맞게 조정합니다. 또한, 직원이 데이터 분류 체계와 민감한 데이터의 취급 및 보호에 대한 책임을 이해할 수 있도록 보안 인식 및 교육을 제공합니다.

구현 단계

1. 워크로드 내 데이터의 분류 및 민감도 수준을 식별합니다.
2. 각 수준에 대한 격리 경계를 정의하고 시행 전략을 결정합니다.
3. 액세스, 암호화, 감사, 보존 및 데이터 분류 정책에 필요한 기타 사항을 관리하기 위해 정의한 제어 기능을 평가합니다.
4. 적절한 경우 토큰화 또는 익명화 사용 등 데이터의 민감도 수준을 낮추는 옵션을 평가합니다.
5. 구성된 리소스의 자동화된 테스트 및 모니터링을 사용하여 제어 기능을 확인합니다.

리소스

관련 모범 사례:

- PERF03-BP01 데이터 액세스 및 스토리지 요구 사항을 가장 잘 지원하는 목적별 데이터 스토어 사용
- COST04-BP05 데이터 보존 정책 적용

관련 문서:

- Data Classification 백서
- 보안, 자격 증명 및 규정 준수를 위한 모범 사례
- AWS KMS 모범 사례

- Encryption best practices and features for AWS services

관련 예제:

- Building a serverless tokenization solution to mask sensitive data
- How to use tokenization to improve data security and reduce audit scope

관련 도구:

- AWS Key Management Service (AWS KMS)
- AWS CloudHSM
- AWS Organizations

SEC07-BP03 식별 및 분류 자동화

데이터 식별 및 분류를 자동화하면 올바른 제어를 구현하는 데 도움이 될 수 있습니다. 자동화를 사용하여 수동 결정을 보강하면 인적 오류 및 노출의 위험을 줄일 수 있습니다.

원하는 성과: 분류 및 취급 정책에 따라 적절한 통제가 마련되어 있는지 확인할 수 있습니다. 자동화된 도구 및 서비스는 데이터의 민감도 수준을 식별하고 분류하는 데 도움이 됩니다. 나아가 자동화를 통해 환경을 지속적으로 모니터링하여 데이터가 무단으로 저장되거나 처리되는 경우 이를 감지하고 알림을 보내 시정 조치를 신속하게 취할 수 있습니다.

일반적인 안티 패턴:

- 데이터 식별 및 분류를 위한 수동 프로세스에만 의존하므로, 오류가 발생하기 쉽고 시간이 많이 걸릴 수 있습니다. 이로 인해 특히 데이터 볼륨이 증가하면 데이터 분류의 효율성과 일관성이 떨어질 수 있습니다.
- 조직 전체에 걸쳐 데이터 자산을 추적하고 관리하는 메커니즘이 없습니다.
- 조직 내에서 이동하고 변화하는 데이터를 지속적으로 모니터링하고 분류해야 할 필요성을 간과합니다.

이 모범 사례 확립의 이점: 데이터 식별 및 분류를 자동화하면 데이터 보호 제어를 보다 일관되고 정확하게 적용하여 인적 오류의 위험을 줄일 수 있습니다. 자동화는 또한 민감한 데이터 액세스 및 이동에 대한 가시성을 제공하여 무단 처리를 탐지하고 시정 조치를 취하는 데 도움이 됩니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

워크로드의 초기 설계 단계에서 데이터를 분류할 때는 사람이 판단하는 경우가 많지만, 예방 차원에서 테스트 데이터의 식별 및 분류를 자동화하는 시스템을 마련하는 것을 고려해 보세요. 예를 들어, 개발자에게 대표 데이터를 스캔하여 민감도를 결정하는 도구나 서비스를 제공할 수 있습니다. AWS에서 Amazon S3에 데이터세트를 업로드하고 Amazon Macie, Amazon Comprehend 또는 Amazon Comprehend Medical을 사용하여 스캔할 수 있습니다. 마찬가지로 단위 및 통합 테스트의 일환으로 데이터를 스캔하여 민감한 데이터가 예상되지 않는 부분을 찾아내는 것도 고려해 보세요. 이 단계에서 민감한 데이터에 대한 알림을 통해 프로덕션에 배포하기 전에 보호의 허점을 강조할 수 있습니다. AWS Glue, Amazon SNS, Amazon CloudWatch에서 민감한 데이터 탐지와 같은 기타 기능을 사용하여 PII를 탐지하고 완화 조치를 수행할 수도 있습니다. 자동화된 도구 또는 서비스의 경우, 민감한 데이터를 정의하는 방법을 이해하고 필요에 따라 다른 사람이나 자동화된 솔루션으로 이를 보강하여 허점을 메우세요.

탐지 제어 기능으로 환경을 지속적으로 모니터링하여 민감한 데이터가 규정을 준수하지 않는 방식으로 저장되고 있는지 탐지하세요. 이를 통해 민감한 데이터가 로그 파일로 내보내지거나, 적절히 식별되지 않거나 수정되지 않고 데이터 분석 환경으로 복사되는 등의 상황을 탐지할 수 있습니다. Amazon S3에 저장된 데이터는 Amazon Macie를 사용하여 민감한 데이터가 있는지 지속적으로 모니터링할 수 있습니다.

구현 단계

1. SEC07-BP01에 설명된 조직 내 데이터 분류 체계를 검토합니다.
   a. 조직의 데이터 분류 체계를 이해하면 회사 정책에 맞는 자동 식별 및 분류를 위한 정확한 프로세스를 수립할 수 있습니다.
2. 자동 식별 및 분류를 위해 환경에 대한 초기 스캔을 수행합니다.
   a. 초기에 데이터를 전체적으로 스캔하면 민감한 데이터가 사용자 환경의 어디에 있는지 포괄적으로 이해하는 데 도움이 될 수 있습니다. 처음에 전체 스캔이 필요하지 않거나 비용 때문에 미리 완료할 수 없다면 데이터 샘플링 기술이 성과를 달성하는 데 적합한지 평가하세요. 예를 들어, Amazon Macie를 사용하여 S3 버킷에서 광범위하고 자동화된 민감한 데이터 검색 작업을 수행하도록 구성할 수 있습니다. 이 기능은 샘플링 기술을 사용하여 민감한 데이터가 있는 위치에 대한 예비 분석을 비용 효율적으로 수행합니다. 그런 다음 민감한 데이터 검색 작업을 사용하여 S3 버킷에 대한 심층 분석을 수행할 수 있습니다. 다른 데이터 스토어를 S3로 내보내 Macie에서 스캔할 수도 있습니다.
   b. 스캔으로 식별된 데이터 스토리지 리소스에 대해 SEC07-BP02에 정의된 액세스 제어를 설정합니다.
3. 환경에 대한 지속적인 스캔을 구성합니다.

   a. Macie의 자동화된 중요 데이터 검색 기능을 사용하여 환경을 지속적으로 스캔할 수 있습니다. 민감한 데이터를 저장할 권한이 있는 알려진 S3 버킷은 Macie의 허용 목록을 사용하여 제외할 수 있습니다.

4. 식별 및 분류를 구축 및 테스트 프로세스에 통합합니다.

   a. 워크로드가 개발 중인 동안 개발자가 데이터의 민감도를 스캔하는 데 사용할 수 있는 도구를 식별합니다. 이러한 도구를 통합 테스트의 일부로 사용하여 민감한 데이터가 예상되지 않는 경우 이를 알리고 추가 배포를 방지할 수 있습니다.

5. 승인되지 않은 위치에서 민감한 데이터가 발견될 경우 조치를 취할 수 있는 시스템 또는 런북을 구현합니다.

   a. 자동 수정을 사용하여 데이터에 대한 액세스를 제한합니다. 예를 들어, 속성 기반 액세스 제어(ABAC)를 사용하는 경우 이 데이터를 액세스가 제한된 S3 버킷으로 이동하거나 객체에 태그를 지정할 수 있습니다. 또한 데이터가 감지될 때 마스킹하는 것도 고려해 보세요.

   b. 데이터 보호 및 인시던트 대응 팀에 알려 인시던트의 근본 원인을 조사합니다. 이들이 찾아내는 모든 교훈은 향후 인시던트를 방지하는 데 도움이 될 수 있습니다.

리소스

관련 문서:

- AWS Glue: Detect and process sensitive data
- Using managed data identifiers in Amazon SNS
- Amazon CloudWatch Logs: Help protect sensitive log data with masking

관련 예제:

- Enabling data classification for Amazon RDS database with Macie
- Detecting sensitive data in DynamoDB with Macie

관련 도구:

- Amazon Macie
- Amazon Comprehend
- Amazon Comprehend Medical
- AWS Glue

SEC07-BP04 확장 가능한 데이터 수명 주기 관리 정의

다양한 수준의 데이터 분류 및 처리와 관련된 데이터 수명 주기 요구 사항을 파악하세요. 여기에는 데이터가 환경에 처음 유입되었을 때 데이터를 처리하는 방법, 데이터가 변환되는 방식, 데이터 폐기 규칙이 포함될 수 있습니다. 보존 기간, 액세스, 감사, 출처 추적과 같은 요소를 고려하세요.

원하는 성과: 데이터를 수집 시점 및 시간에 최대한 가깝게 분류합니다. 데이터 분류에 마스킹, 토큰화 또는 민감도 수준을 낮추는 기타 프로세스가 필요한 경우 수집 시점과 시간에 최대한 가깝게 작업을 수행하세요.

더 이상 보관하기에 적절하지 않은 경우 정책에 따라 분류를 기반으로 데이터를 삭제합니다.

일반적인 안티 패턴:

- 다양한 민감도 수준과 액세스 요구 사항을 고려하지 않고 데이터 수명 주기 관리에 대한 획일적인 접근 방식을 구현합니다.
- 사용 가능한 데이터 또는 백업된 데이터의 관점에서만 수명 주기 관리를 고려하고, 모두 고려하지 않습니다.
- 가치나 출처를 확인하지 않고 워크로드에 입력된 데이터가 유효하다고 가정합니다.
- 데이터 백업 및 보호 대신 데이터 내구성에 의존합니다.
- 유용성 및 필수 보존 기간을 초과하여 데이터를 보존합니다.

이 모범 사례 확립의 이점: 잘 정의되고 확장 가능한 데이터 수명 주기 관리 전략은 적절한 제어를 유지하면서 규제를 계속해서 준수하고, 데이터 보안을 개선하며, 스토리지 비용을 최적화하는 데 도움을 줄 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

워크로드 내의 데이터는 동적인 경우가 많습니다. 워크로드 환경에 유입될 때 취하는 형태는 비즈니스 로직, 보고, 분석 또는 기계 학습에 저장되거나 사용될 때와 다를 수 있습니다. 또한, 데이터의 가치는 시간이 지남에 따라 변할 수 있습니다. 일부 데이터는 본질적으로 일시적이며, 오래될수록 가치를 잃습니다. 데이터에 대한 이러한 변경이 데이터 분류 체계 및 관련 제어에 따른 평가에 어떤 영향을 미치는지 생각해 보세요. 가능한 경우 Amazon S3 수명 주기 정책 및 Amazon Data Lifecycle Manager와 같은 자동화된 수명 주기 메커니즘을 사용하여 데이터 보존, 아카이빙 및 만료 프로세스를 구성합니다. DynamoDB에 저장된 데이터의 경우 Time To Live(TTL) 기능을 사용하여 항목별 만료 타임스탬프를 정의할 수 있습니다.

사용할 수 있는 데이터와 백업으로 저장된 데이터를 구분합니다. AWS 서비스 전반의 데이터 백업을 자동화하기 위해 AWS Backup 사용을 고려하세요. Amazon EBS 스냅샷은 수명 주기, 데이터 보호, 보호 메커니즘에 대한 액세스를 비롯한 S3 기능을 사용하여 EBS 볼륨을 복사하고 저장하는 방법을 제공합니다. 이 두 가지 메커니즘은 S3 객체 잠금 및 AWS Backup 볼트 잠금으로, 이를 통해 백업에 대한 추가적인 보안 및 제어를 제공할 수 있습니다. 백업에 대한 업무와 액세스 권한을 명확하게 구분하여 관리합니다. 계정 수준에서 백업을 격리하여 이벤트 발생 시 영향을 받는 환경으로부터 분리할 수 있습니다.

수명 주기 관리의 또 다른 측면은 워크로드가 진행되는 동안 데이터 내역을 기록하는 것입니다. 이를 데이터 출처 추적이라고 합니다. 따라서 데이터의 출처, 수행한 변환, 변경한 소유자 또는 프로세스, 변경 시기를 확실하게 알 수 있습니다. 이 기록이 있으면 잠재적 보안 이벤트 발생 시 문제 해결 및 조사에 도움이 됩니다. 예를 들어 Amazon DynamoDB 테이블에 변환에 대한 메타데이터를 로깅할 수 있습니다. 데이터 레이크 내에서 각 데이터 파이프라인 단계의 서로 다른 S3 버킷에 변환된 데이터의 복사본을 보관할 수 있습니다. AWS Glue Data Catalog에 스키마와 타임스탬프 정보를 저장합니다. 그리고 솔루션에 관계없이 최종 사용자의 요구 사항을 고려하여 데이터 출처를 보고하는 데 필요한 적절한 도구를 결정하세요. 이렇게 하면 출처를 가장 잘 추적하는 방법을 결정하는 데 도움이 됩니다.

구현 단계

1. 워크로드의 데이터 유형, 민감도 수준 및 액세스 요구 사항을 분석하여 데이터를 분류하고 적절한 수명 주기 관리 전략을 정의합니다.
2. 법률, 규제 및 조직 요구 사항에 맞는 데이터 보존 정책 및 자동 폐기 프로세스를 설계하고 구현합니다.
3. 변화하는 워크로드 요구 사항 및 규제 변화에 맞춰 데이터 수명 주기 관리 전략, 제어, 정책을 지속적으로 모니터링, 감사 및 조정하기 위한 프로세스와 자동화를 수립합니다.
   a. AWS Config를 사용하여 자동 수명 주기 관리가 활성화되지 않은 리소스 감지

리소스

관련 모범 사례:

- COST04-BP05 데이터 보존 정책 적용
- SUS04-BP03 정책을 사용하여 데이터세트의 수명 주기 관리

관련 문서:

- Data Classification 백서

- AWS Blueprint for Ransomware Defense
- DevOps Guidance: Improve traceability with data provenance tracking

관련 예제:

- How to protect sensitive data for its entire lifecycle in AWS
- Build data lineage for data lakes using AWS Glue, Amazon Neptune, and Spline

관련 도구:

- AWS Backup
- Amazon Data Lifecycle Manager
- AWS Identity and Access Management Access Analyzer

저장 데이터 보호

저장 데이터란 워크로드의 어느 기간에서든지 비휘발성 스토리지에 지속되는 모든 데이터를 의미합니다. 여기에는 블록 스토리지, 객체 스토리지, 데이터베이스, 아카이브, IoT 디바이스 그리고 데이터가 지속되는 모든 기타 스토리지 미디어가 포함됩니다. 저장 데이터를 보호하여 암호화 및 적절한 액세스 제어가 구현될 경우 무단 액세스 위험이 감소합니다.

암호화와 토큰화는 둘 모두 중요한 데이터 보호 체계이나 별개의 개념입니다.

토큰화는 사용자 신용 카드 정보와 같이 중요한 정보를 나타내는 토큰을 정의할 수 있는 프로세스입니다. 토큰 자체는 아무 의미가 없어야 하며, 토큰화하는 데이터에서 파생되어서는 안 됩니다. 따라서 암호화 다이제스트를 토큰으로 사용할 수 없습니다. 토큰화 방식을 신중하게 계획하면 콘텐츠를 추가로 보호할 수 있으며 규정 준수 요구 사항을 충족할 수 있습니다. 예를 들어 신용 카드 번호 대신 토큰을 활용하는 경우 신용 카드 처리 시스템의 규정 준수 범위를 줄일 수 있습니다.

암호화는 콘텐츠를 다시 일반 텍스트로 해독하는 데 필요한 비밀 키가 없으면 읽을 수 없는 방식으로 콘텐츠를 변환하는 방식입니다. 토큰화 및 암호화를 사용하면 정보를 효과적으로 보호할 수 있습니다. 또한 마스킹은 중요하지 않은 것으로 간주되는 데이터만 남을 때까지 데이터의 일부를 수정할 수 있는 기술입니다. 예를 들어 PCI-DSS를 사용하면 카드 번호의 마지막 네 자리가 인덱싱을 위한 규정 준수 범위 경계를 벗어나도록 할 수 있습니다.

암호화 키 사용 감사: 암호화 키 사용을 이해하고 감사하여 키에 대한 액세스 제어 메커니즘이 적절하게 구현되었는지 검증합니다. 예를 들어 AWS KMS 키를 사용하는 AWS 서비스는 AWS CloudTrail에

서 모든 사용 사례를 로깅합니다. Amazon CloudWatch 로그 인사이트와 같은 도구를 사용하여 AWS CloudTrail을 쿼리함으로써 모든 키 사용이 유효한지 확인할 수 있습니다.

모범 사례

- SEC08-BP01 보안 키 관리 구현
- SEC08-BP02 저장 시 암호화 적용
- SEC08-BP03 저장 데이터 보호 자동화
- SEC08-BP04 액세스 제어 적용

SEC08-BP01 보안 키 관리 구현

보안 키 관리에는 워크로드의 저장 데이터를 보호하는 데 필요한 키 구성 요소의 저장, 순환, 액세스 제어 및 모니터링이 포함됩니다.

원하는 성과: 확장 가능하고 반복 가능하며 자동화된 키 관리 메커니즘이 있습니다. 메커니즘이 키 구성 요소에 대한 액세스 권한을 최소한으로 제한하고 키 가용성, 기밀성 및 무결성 간에 적절한 균형을 유지합니다. 키에 대한 액세스를 모니터링하고 키 구성 요소의 교체가 필요한 경우 자동화된 프로세스를 사용하여 키를 교체합니다. 인간 작업자가 키 구성 요소에 액세스하도록 허용하지 않습니다.

일반적인 안티 패턴:

- 암호화되지 않은 키 구성 요소에 대한 인적 접근이 가능합니다.
- 사용자 지정 암호화 알고리즘을 생성합니다.
- 키 구성 요소에 액세스할 수 있는 권한이 지나치게 광범위합니다.

이 모범 사례 확립의 이점: 워크로드에 대한 보안 키 관리 메커니즘을 구축하면 무단 액세스로부터 콘텐츠를 보호하는 데 도움이 될 수 있습니다. 또한 데이터를 암호화하기 위한 규제 요구 사항이 적용될 수 있습니다. 효과적인 키 관리 솔루션은 키 구성 요소를 보호하기 위해 해당 규정에 맞는 기술적 메커니즘을 제공할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

저장 데이터의 암호화는 기본적인 보안 제어입니다. 이러한 제어를 구현하려면 워크로드에 저장 데이터를 암호화하는 데 사용되는 키 구성 요소를 안전하게 저장하고 관리하는 메커니즘이 필요합니다.

AWS는 AWS Key Management Service(AWS KMS)를 제공하여 내구성이 뛰어나고 안전하며 중복된 AWS KMS 키 스토리지를 제공합니다. 많은 AWS 서비스가 AWS KMS에 통합되어 데이터 암호화를 지원합니다. AWS KMS에서는 FIPS 140-2 레벨 3 검증 하드웨어 보안 모듈을 사용하여 키를 보호합니다. AWS KMS 키를 일반 텍스트로 내보내는 메커니즘은 없습니다.

다중 계정 전략을 사용하여 워크로드를 배포할 때 AWS KMS 키를 사용하는 워크로드와 동일한 계정에 키를 유지해야 합니다. 이 분산 모델에서는 AWS KMS 키 관리에 대한 책임이 사용자의 팀에 있습니다. 다른 사용 사례에서는 조직에서 AWS KMS 키를 중앙 집중식 계정에 저장하도록 선택할 수 있습니다. 이 중앙 집중식 구조에는 워크로드 계정이 중앙 집중식 계정에 저장된 키에 액세스하는 데 필요한 크로스 계정 액세스를 가능하게 하는 추가 정책이 필요하지만 단일 키를 여러 AWS 계정에서 공유하는 사용 사례에서는 더 적합할 수 있습니다.

키 구성 요소의 보관 위치와 관계없이 키 정책 및 IAM 정책을 사용하여 키에 대한 액세스를 엄격하게 제어해야 합니다. 키 정책은 AWS KMS 키에 대한 액세스를 제어하는 기본 방법입니다. 또한 AWS KMS 키 부여는 사용자를 대신하여 데이터를 암호화 및 복호화하는 AWS 서비스에 대한 액세스를 제공할 수 있습니다. AWS KMS 키에 대한 액세스 제어 지침을 검토합니다.

암호화 키 사용을 모니터링하여 비정상적인 액세스 패턴을 탐지해야 합니다. AWS KMS에 저장된 고객 관리형 키와 AWS 관리형 키를 사용하여 수행한 작업은 AWS CloudTrail에 로그인할 수 있으며 정기적으로 검토해야 합니다. 키 폐기 이벤트를 모니터링하는 데 특히 주의를 기울이세요. 키 구성 요소의 우발적 또는 악의적 폐기를 방지하기 위해 키 폐기 이벤트는 키 구성 요소를 즉시 삭제하지 않습니다. AWS KMS에서 키 삭제 시도에는 대기 시간이 적용됩니다. 기본값은 30일, 최솟값은 7일이므로 관리자는 이러한 작업을 검토하고 필요한 경우 요청을 롤백할 시간을 확보할 수 있습니다.

대부분의 AWS 서비스는 사용자에게 투명한 방식으로 AWS KMS를 사용합니다. AWS 관리형 키를 사용할지 아니면 고객 관리형 키를 사용할지 결정하는 것만이 유일한 요구 사항입니다. 워크로드에서 데이터를 암호화하거나 복호화하는 데 AWS KMS를 직접 사용해야 하는 경우 봉투 암호화를 사용하여 데이터를 보호해야 합니다. AWS Encryption SDK에서는 애플리케이션에 클라이언트측 암호화 기본 요소를 제공하여 봉투 암호화를 구현하고 AWS KMS와 통합할 수 있습니다.

구현 단계

1. 키에 적합한 키 관리 옵션(AWS 관리형 또는 고객 관리형)을 결정합니다.
    a. 사용 편의성을 위해 AWS에서는 대부분의 서비스에 대해 AWS 소유 및 AWS 관리형 키를 제공하며, 키 구성 요소나 키 정책을 관리할 필요 없이 저장 시 암호화 기능을 제공합니다.
    b. 고객 관리형 키를 사용할 때는 민첩성, 보안, 데이터 주권, 가용성 사이에서 최상의 균형을 유지할 수 있도록 기본 키 스토어를 고려하세요. 다른 사용 사례에서는 AWS CloudHSM 또는 외부 키 저장소와 함께 사용자 지정 키 저장소를 사용해야 할 수도 있습니다.

2. 워크로드에 사용 중인 서비스 목록을 검토하여 서비스와의 AWS KMS 통합 방식을 파악하세요. 예를 들어, EC2 인스턴스는 암호화된 EBS 볼륨을 사용하여 해당 볼륨에서 생성된 Amazon EBS 스냅샷도 고객 관리형 키를 사용하여 암호화되는지 확인하고 암호화되지 않은 스냅샷 데이터가 우발적으로 공개되는 것을 방지할 수 있습니다.
   a. How AWS services use AWS KMS
   b. AWS 서비스에서 제공하는 암호화 옵션에 대한 자세한 내용은 해당 서비스 사용 설명서 또는 개발자 안내서의 저장된 데이터 암호화 주제를 참조하세요.
3. AWS KMS 구현: AWS KMS는 다양한 AWS 서비스와 애플리케이션에서 간단하게 키를 생성 및 관리하고 암호화 사용을 제어할 수 있습니다.
   a. Getting started: AWS Key Management Service (AWS KMS)
   b. AWS KMS 키에 대한 액세스 제어 모범 사례를 검토합니다.
4. AWS Encryption SDK 고려: 애플리케이션이 클라이언트 측 데이터를 암호화해야 하는 경우 AWS Encryption SDK를 AWS KMS와 통합하세요.
   a. AWS Encryption SDK
5. AWS KMS 키 정책이 지나치게 광범위한지 자동으로 검토하고 알리도록 IAM Access Analyzer를 활성화합니다.
   a. 사용자 지정 정책 확인을 사용하여 리소스 정책 업데이트가 KMS 키에 대한 퍼블릭 액세스 권한을 부여하지 않는지 확인하는 것이 좋습니다.
6. Security Hub를 활성화하여 잘못 구성된 키 정책, 삭제 예정 키 또는 자동 순환이 활성화되지 않은 키가 있는 경우 알림을 받습니다.
7. AWS KMS 키에 적합한 로깅 수준을 결정하세요. 읽기 전용 이벤트를 포함하여 AWS KMS에 대한 직접 호출이 로깅되므로 AWS KMS에 연결된 CloudTrail 로그의 양이 많아질 수 있습니다.
   a. 일부 조직에서는 AWS KMS 로깅 활동을 별도의 트레일로 분리하는 것을 선호합니다. 자세한 내용은 AWS KMS 개발자 안내서의 Logging AWS KMS API calls with CloudTrail 섹션을 참조하세요.

리소스

관련 문서:

- AWS Key Management Service
- AWS cryptographic services and tools
- 암호화로 Amazon S3 데이터 보호
- 봉투 암호화

- Digital sovereignty pledge
- Demystifying AWS KMS key operations, bring your own key, custom key store, and ciphertext portability
- AWS Key Management Service cryptographic details

관련 비디오:

- How Encryption Works in AWS
- Securing Your Block Storage on AWS
- AWS data protection: Using locks, keys, signatures, and certificates

관련 예제:

- Implement advanced access control mechanisms using AWS KMS

SEC08-BP02 저장 시 암호화 적용

저장 상태의 프라이빗 데이터를 암호화하여 기밀성을 유지하고 의도하지 않은 데이터 공개 또는 유출에 대한 추가 보호 계층을 제공합니다. 암호화는 먼저 복호화되지 않고는 데이터를 읽거나 액세스할 수 없도록 하여 데이터를 보호합니다. 암호화되지 않은 데이터의 인벤토리를 만들고 제어하여 데이터 노출과 관련된 위험을 완화합니다.

원하는 성과: 저장 시 기본적으로 프라이빗 데이터를 암호화하는 메커니즘이 있습니다. 이러한 메커니즘은 데이터의 기밀성을 유지하는 데 도움이 되며 의도치 않은 데이터 공개 또는 유출에 대한 추가 보호 계층을 제공합니다. 암호화되지 않은 데이터의 인벤토리를 유지하고 이를 보호하기 위해 마련된 제어를 이해합니다.

일반적인 안티 패턴:

- 기본적으로 암호화 구성을 사용하지 않습니다.
- 복호화 키에 지나치게 관대한 액세스를 제공합니다.
- 암호화 및 복호화 키의 사용을 모니터링하지 않습니다.
- 데이터를 암호화되지 않은 상태로 저장합니다.
- 데이터 용도, 유형 및 분류에 관계없이 모든 데이터에 동일한 암호화 키를 사용합니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 지침

워크로드 내의 데이터 분류에 암호화 키를 매핑합니다. 이 접근 방식은 데이터에 단일 또는 매우 적은 수의 암호화 키를 사용할 때 지나치게 허용적인 액세스 권한을 방지하는 데 도움이 됩니다(SEC07-BP01 데이터 분류 체계 이해 참조).

AWS Key Management Service(AWS KMS)는 많은 AWS 서비스와 통합되어 저장 데이터를 보다 쉽게 암호화할 수 있습니다. 예를 들어 Amazon Elastic Compute Cloud(Amazon EC2)에서 새 EBS 볼륨이 자동으로 암호화되도록 계정에 기본 암호화를 설정할 수 있습니다. AWS KMS를 사용할 때 데이터를 얼마나 엄격하게 제한해야 하는지 고려합니다. 기본 및 서비스 제어 AWS KMS 키는 사용자를 대신하여 AWS에서 관리하고 사용합니다. 기본 암호화 키에 대한 세분화된 액세스 권한이 필요한 민감한 데이터의 경우 고객 관리형 키(CMK)를 고려합니다. 키 정책을 사용하여 교체 및 액세스 관리를 포함하여 CMK를 완전히 제어할 수 있습니다.

또한 Amazon Simple Storage Service(Amazon S3)와 같은 서비스는 이제 기본적으로 모든 새 객체를 암호화합니다. 이 구현은 성능에 영향을 주지 않으면서 향상된 보안을 제공합니다.

Amazon Elastic Compute Cloud(Amazon EC2) 또는 Amazon Elastic File System(Amazon EFS)과 같은 기타 서비스는 기본 암호화 설정을 지원합니다. AWS Config 규칙을 사용하여 Amazon Elastic Block Store(Amazon EBS) 볼륨, Amazon Relational Database Service(Amazon RDS) 인스턴스 및 Amazon S3 버킷에 암호화를 사용하고 있는지 자동으로 확인할 수 있습니다.

AWS는 또한 클라이언트측 암호화 옵션을 제공하므로 데이터를 클라우드에 업로드하기 전에 암호화할 수 있습니다. AWS Encryption SDK에서는 봉투 암호화를 사용하여 데이터를 암호화하는 방법을 제공합니다. 래핑 키를 제공하면 AWS Encryption SDK가 암호화하는 각 데이터 객체에 대해 고유한 데이터 키를 생성합니다. 관리형 단일 테넌트 하드웨어 보안 모듈(HSM)이 필요한 경우 AWS CloudHSM을 고려합니다. AWS CloudHSM을 사용하면 FIPS 140-2 레벨 3 검증 HSM에서 암호화 키를 생성, 가져오기 및 관리할 수 있습니다. AWS CloudHSM의 일부 사용 사례에는 인증 기관(CA) 발급을 위한 프라이빗 키 보호와 Oracle 데이터베이스용 투명한 데이터 암호화(TDE) 활성화가 포함됩니다. AWS CloudHSM 클라이언트 SDK는 데이터를 AWS에 업로드하기 전에 AWS CloudHSM에 저장된 키를 사용하여 데이터 클라이언트측을 암호화할 수 있는 소프트웨어를 제공합니다. Amazon DynamoDB Encryption Client를 사용하면 DynamoDB 테이블에 업로드하기 전에 항목을 암호화하고 서명할 수도 있습니다.

구현 단계

- 새로운 Amazon EBS 볼륨에 대해 기본 암호화 구성: AWS에서 제공하는 기본 키 또는 사용자가 생성한 키를 사용하는 옵션을 통해, 새로 생성되는 모든 Amazon EBS 볼륨이 암호화된 형식으로 생성되도록 지정합니다.

- 암호화된 Amazon Machine Image(AMI) 구성: 암호화가 구성된 상태에서 기존 AMI를 복사하면 루트 볼륨 및 스냅샷을 자동으로 암호화합니다.
- Amazon RDS 암호화 구성: 암호화 옵션을 사용하여 Amazon RDS 데이터베이스 클러스터 및 저장된 스냅샷에 대한 암호화를 구성합니다.
- 각 데이터 분류를 위해 적절한 보안 주체에 대한 액세스를 제한하는 정책으로 AWS KMS 키 생성 및 구성: 예를 들어 프로덕션 데이터 암호화를 위해 하나의 AWS KMS 키를 생성하고 개발 또는 테스트 데이터 암호화를 위해 다른 키를 생성합니다. 다른 AWS 계정에 키 액세스를 제공할 수도 있습니다. 개발 및 프로덕션 환경에 대해 서로 다른 계정을 사용하는 것이 좋습니다. 프로덕션 환경에서 개발 계정의 아티팩트를 복호화해야 하는 경우 개발 아티팩트를 암호화하는 데 사용되는 CMK 정책을 편집하여 프로덕션 계정에 해당 아티팩트를 복호화할 수 있는 기능을 제공할 수 있습니다. 그러면 프로덕션 환경에서 프로덕션에 사용하기 위해 복호화된 데이터를 수집할 수 있습니다.
- 추가 AWS 서비스에서 암호화 구성: 사용하는 다른 AWS 서비스의 경우 해당 서비스의 보안 설명서를 검토하여 서비스의 암호화 옵션을 결정합니다.

리소스

관련 문서:

- AWS 암호화 도구
- AWS Encryption SDK
- AWS KMS Cryptographic Details 백서
- AWS Key Management Service
- AWS cryptographic services and tools
- Amazon EBS Encryption
- Default encryption for Amazon EBS volumes
- Amazon RDS 리소스 암호화
- How do I enable default encryption for an Amazon S3 bucket?
- 암호화로 Amazon S3 데이터 보호

관련 비디오:

- How Encryption Works in AWS
- Securing Your Block Storage on AWS

SEC08-BP03 저장 데이터 보호 자동화

자동화를 사용하여 저장된 데이터 제어를 검증하고 적용하세요. 자동 스캐닝을 사용하여 데이터 스토리지 솔루션의 잘못된 구성을 탐지하고, 가능하다면 자동화된 프로그래밍 방식 응답을 통해 수정을 수행합니다. CI/CD 프로세스에 자동화를 통합하여 프로덕션 환경에 배포하기 전에 데이터 스토리지 구성 오류를 감지할 수 있습니다.

원하는 성과: 자동화된 시스템이 데이터 스토리지 위치를 스캔하고 모니터링하여 제어 기능의 잘못된 구성, 무단 액세스 및 예상치 못한 사용이 있는지 확인합니다. 잘못 구성된 스토리지 위치를 탐지하면 자동 수정이 시작됩니다. 자동화된 프로세스는 데이터 백업을 생성하고 원본 환경 외부에 변경 불가능한 사본을 저장합니다.

일반적인 안티 패턴:

- 지원되는 경우에 기본 설정으로 암호화를 활성화하는 옵션을 고려하고 있지 않습니다.
- 자동 백업 및 복구 전략을 수립할 때 운영 이벤트 외에 보안 이벤트는 고려하지 않습니다.
- 스토리지 서비스에 대한 공개 액세스 설정을 적용하지 않습니다.
- 저장 데이터를 보호하기 위한 제어 기능을 모니터링하고 감사하지 않습니다.

이 모범 사례 확립의 이점: 자동화는 데이터 스토리지 위치를 잘못 구성할 위험을 방지하는 데 도움이 됩니다. 프로덕션 환경에 잘못된 구성이 유입되는 것을 방지하는 데 도움이 됩니다. 이 모범 사례는 잘못된 구성이 발생할 경우 이를 탐지하고 수정하는 데도 도움이 됩니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

자동화는 저장 데이터를 보호하기 위한 관행 전반에 적용되는 주제입니다. SEC01-BP06 표준 보안 제어의 배포 자동화에서는 AWS CloudFormation과 같이 코드형 인프라(IaC) 템플릿을 사용하여 리소스 구성을 캡처하는 방법을 설명합니다. 이러한 템플릿은 버전 제어 시스템에 적용되며 CI/CD 파이프라인을 통해 AWS에 리소스를 배포하는 데 사용됩니다. 이러한 기술은 Amazon S3 버킷의 암호화 설정과 같은 데이터 스토리지 솔루션의 구성을 자동화하는 데도 동일하게 적용됩니다.

AWS CloudFormation Guard의 규칙을 사용하여 IaC 템플릿에서 정의한 설정에 CI/CD 파이프라인의 잘못된 구성이 있는지 확인할 수 있습니다. AWS Config를 통해 CloudFormation 또는 기타 IaC 도구에서 아직 사용할 수 없는 설정에 잘못된 구성이 있는지 모니터링할 수 있습니다. Config가 잘못된 구성에 대해 생성하는 알림은 SEC04-BP04 비준수 리소스에 대한 문제 해결 시작에서 설명한 대로 자동으로 수정할 수 있습니다.

권한 관리 전략의 일부로 자동화를 사용하는 것도 자동화된 데이터 보호의 필수 구성 요소입니다. SEC03-BP02 최소 권한 액세스 부여 및 SEC03-BP04 지속적으로 권한 축소에서는 최소 권한 액세스 정책을 구성하는 방법을 설명합니다. AWS Identity and Access Management Access Analyzer에서는 이 정책을 지속적으로 모니터링하여 권한을 축소할 수 있는 경우 조사 결과를 생성합니다. 권한 모니터링에 대한 자동화 외에도 EBS 볼륨(EC2 인스턴스를 통해), S3 버킷 및 지원되는 Amazon Relational Database Service 데이터베이스에 대한 비정상적인 데이터 액세스 동작을 감시하도록 Amazon GuardDuty를 구성할 수 있습니다.

자동화는 민감한 데이터가 승인되지 않은 위치에 저장되는 시점을 탐지하는 역할도 합니다. SEC07-BP03 식별 및 분류 자동화에서는 Amazon Macie가 S3 버킷에서 예기치 못한 민감한 데이터를 모니터링하고 자동화된 응답을 시작할 수 있는 알림을 생성하는 방법에 대해 설명합니다.

REL09 데이터 백업의 관행에 따라 자동화된 데이터 백업 및 복구 전략을 개발합니다. 데이터 백업 및 복구는 운영 이벤트와 마찬가지로 보안 이벤트 복구에도 중요합니다.

구현 단계

1. IaC 템플릿에서 데이터 스토리지 구성을 캡처합니다. CI/CD 파이프라인의 자동 검사를 사용하여 구성 오류를 감지합니다.
   a. AWS CloudFormation을 IaC 템플릿에 사용할 수 있으며, AWS CloudFormation Guard를 사용하여 템플릿에 잘못된 구성이 있는지 확인할 수 있습니다.
   b. AWS Config를 사용하여 사전 평가 모드에서 규칙을 실행합니다. 이 설정을 사용하면 리소스를 생성하기 전에 CI/CD 파이프라인의 한 단계로 리소스의 규정 준수를 확인할 수 있습니다.
2. 리소스에서 데이터 스토리지 구성 오류를 모니터링합니다.
   a. 데이터 스토리지 리소스에서 제어 구성의 변경 사항을 모니터링하고 잘못된 구성이 감지되면 수정 조치를 간접 호출하는 알림을 생성하도록 AWS Config를 설정합니다.
   b. 자동 수정에 대한 자세한 지침은 SEC04-BP04 비준수 리소스에 대한 수정 시작을 참조하세요.
3. 자동화를 통해 데이터 액세스 권한을 지속적으로 모니터링하고 줄입니다.
   a. IAM Access Analyzer를 지속적으로 실행하여 권한이 축소될 가능성이 있는 경우 알림을 생성할 수 있습니다.
4. 비정상적인 데이터 액세스 동작을 모니터링하고 알림을 보냅니다.
   a. GuardDuty는 EBS 볼륨, S3 버킷, RDS 데이터베이스와 같은 데이터 스토리지 리소스에 대한 기본 액세스 동작과의 편차와 알려진 위협 서명을 모두 감시합니다.
5. 예상치 못한 위치에 저장되는 민감한 데이터를 모니터링하고 알림을 보냅니다.
   a. Amazon Macie를 사용하여 S3 버킷에서 민감한 데이터를 지속적으로 스캔합니다.
6. 안전하고 암호화된 데이터 백업을 자동화합니다.

a. AWS Backup은 AWS에서 다양한 데이터 소스의 암호화되고 안전한 백업을 생성하는 관리형 서비스입니다. Elastic Disaster Recovery를 사용하면 전체 서버 워크로드를 복사하고 초 단위로 측정된 Recovery Point Objective(RPO)로 지속적인 데이터 보호를 유지 관리할 수 있습니다. 두 서비스가 함께 작동하도록 구성하여 데이터 백업 생성 및 장애 조치 위치에 복사하는 작업을 자동화할 수 있습니다. 이렇게 하면 운영 또는 보안 이벤트의 영향을 받는 경우에도 데이터를 계속 사용할 수 있습니다.

리소스

관련 모범 사례:

- SEC01-BP06 표준 보안 제어의 배포 자동화
- SEC03-BP02 최소 권한 액세스 부여
- SEC03-BP04 지속적으로 권한 축소
- SEC04-BP04 규정 미준수 리소스 관련 문제 해결 시작
- SEC07-BP03 식별 및 분류 자동화
- REL09-BP02 백업 보안 및 암호화
- REL09-BP03 자동으로 데이터 백업 수행

관련 문서:

- AWS Prescriptive Guidance: Automatically encrypt existing and new Amazon EBS volumes
- Ransomware Risk Management on AWS Using the NIST Cyber Security Framework (CSF)

관련 예제:

- How to use AWS Config proactive rules and AWS CloudFormation Hooks to prevent creation of noncompliant cloud resources
- Automate and centrally manage data protection for Amazon S3 with AWS Backup
- AWS re:Invent 2023 - Implement proactive data protection using Amazon EBS snapshots
- AWS re:Invent 2022 - Build and automate for resilience with modern data protection

관련 도구:

- AWS CloudFormation Guard

- AWS CloudFormation Guard Rules Registry
- IAM 액세스 분석기
- Amazon Macie
- AWS Backup
- Elastic Disaster Recovery

SEC08-BP04 액세스 제어 적용

저장 데이터를 보호하려면 격리 및 버전 관리와 같은 메커니즘을 사용하여 액세스 제어를 적용합니다. 최소 권한 및 조건부 액세스 제어를 적용합니다. 데이터에 대한 퍼블릭 액세스 권한 부여를 방지합니다.

원하는 성과: 인증된 사용자만 알아야 할 데이터에 액세스할 수 있는지 확인합니다. 정기적인 백업 및 버전 관리를 통해 데이터를 보호하여 의도적이거나 우발적인 데이터 수정 또는 삭제를 방지합니다. 중요한 데이터를 다른 데이터와 분리하여 기밀성과 데이터 무결성을 보호합니다.

일반적인 안티 패턴:

- 민감도 요구 사항이 다르거나 분류가 다른 데이터를 함께 저장합니다.
- 복호화 키에 지나치게 관대한 권한을 사용합니다.
- 데이터를 잘못 분류합니다.
- 중요한 데이터의 자세한 백업을 유지하지 않습니다.
- 프로덕션 데이터에 대한 지속적인 액세스를 제공합니다.
- 데이터 액세스를 감사하거나 정기적으로 권한을 검토하지 않습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 지침

저장 데이터를 보호하는 것은 데이터 무결성, 기밀성 및 규제 요구 사항 준수를 유지하는 데 중요합니다. 액세스 제어, 격리, 조건부 액세스 및 버전 관리를 포함하여 이를 달성하는 데 도움이 되는 여러 제어를 구현할 수 있습니다.

최소 권한 원칙에 따라 액세스 제어를 적용할 수 있습니다. 이 원칙은 사용자와 서비스가 작업을 수행하는 데 필요한 권한만 제공합니다. 여기에는 암호화 키에 대한 액세스가 포함됩니다. AWS Key Management Service(AWS KMS) 정책을 검토하여 부여하는 액세스 수준이 적절하고 관련 조건이 적용되는지 확인합니다.

각 수준에 대해 고유한 AWS 계정을 사용하여 다양한 분류 수준에 따라 데이터를 분리하고 AWS Organizations을 사용하여 이러한 계정을 관리할 수 있습니다. 이러한 격리는 무단 액세스를 방지하고 데이터 노출 위험을 최소화하는 데 도움이 될 수 있습니다.

Amazon S3 버킷 정책에 부여된 액세스 수준을 정기적으로 검토합니다. 절대적으로 필요한 경우가 아니면 공개적으로 읽을 수 있거나 쓸 수 있는 버킷을 사용하지 마세요. 또한 AWS Config를 사용하여 공개적으로 사용 가능한 버킷을 감지하고 Amazon CloudFront를 사용하여 Amazon S3에서 콘텐츠를 제공하는 것이 좋습니다. 퍼블릭 액세스를 허용하면 안 되는 버킷은 퍼블릭 액세스가 되지 않도록 적절히 구성되어 있는지 확인합니다.

Amazon S3에 저장된 중요 데이터에 대한 버전 관리 및 객체 잠금 메커니즘을 구현합니다. Amazon S3 버전 관리는 이전 버전의 객체를 보존하여 우발적 삭제 또는 덮어쓰기를 했을 때 데이터를 복구합니다. 잠금이 만료될 때까지 루트 사용자라도 객체를 삭제하거나 덮어쓰지 못하게 하는 Amazon S3 Object Lock은 객체에 대한 필수 액세스 제어를 제공합니다. 또한 Amazon S3 Glacier Vault Lock은 Amazon S3 Glacier에 저장된 아카이브에 대해 유사한 기능을 제공합니다.

구현 단계

1. 최소 권한 원칙에 따라 액세스 제어 적용:
    - 사용자 및 서비스에 부여된 액세스 권한을 검토하고 작업을 수행하는 데 필요한 권한만 있는지 확인합니다.
    - AWS Key Management Service(AWS KMS) 정책을 확인하여 암호화 키에 대한 액세스를 검토합니다.
2. 다양한 분류 수준에 따라 데이터 분리:
    - 각 데이터 분류 수준에 대해 고유한 AWS 계정을 사용합니다.
    - AWS Organizations을 사용하여 이러한 계정을 관리합니다.
3. Amazon S3 버킷 및 객체 권한 검토:
    - Amazon S3 버킷 정책에 부여된 액세스 수준을 정기적으로 검토합니다.
    - 절대적으로 필요한 경우가 아니면 공개적으로 읽을 수 있거나 쓸 수 있는 버킷을 사용하지 마세요.
    - AWS Config를 사용하여 공개적으로 사용 가능한 버킷을 감지하는 것을 고려하세요.
    - Amazon CloudFront를 사용하여 Amazon S3의 콘텐츠를 제공합니다.
    - 퍼블릭 액세스를 허용하면 안 되는 버킷은 퍼블릭 액세스가 되지 않도록 적절히 구성되어 있는지 확인합니다.
    - SQS 또는 서드파티 데이터 스토어와 같이 IAM 인증을 사용하는 데이터베이스 및 기타 데이터 소스에 동일한 검토 프로세스를 적용할 수 있습니다.

4. AWS IAM Access Analyzer 사용:
    - Amazon S3 버킷을 분석하고 S3 정책이 외부 엔터티에 대한 액세스 권한을 부여할 때 조사 결과를 생성하도록 AWS IAM Access Analyzer를 구성할 수 있습니다.
5. 버전 관리 및 객체 잠금 메커니즘 구현:
    - Amazon S3 버전 관리를 사용하여 이전 버전의 객체를 보존하여 실수로 삭제하거나 덮어쓰는 것을 방지합니다.
    - 잠금이 만료될 때까지 루트 사용자라도 객체를 삭제하거나 덮어쓰지 못하게 하는 Amazon S3 Object Lock을 사용하여 객체에 대한 필수 액세스 제어를 제공합니다.
    - Amazon S3 Glacier에 저장된 아카이브에 Amazon S3 Glacier Vault Lock을 사용합니다.
6. Amazon S3 Inventory 사용:
    - Amazon S3 Inventory를 사용하여 S3 객체의 복제 및 암호화 상태를 감사하고 보고할 수 있습니다.
7. Amazon EBS 및 AMI 공유 권한 검토:
    - Amazon EBS 및 AMI 공유에 대한 권한을 검토하여 워크로드 외부에 존재하는 AWS 계정과 이미지 및 볼륨이 공유되지 않는다는 것을 확인합니다.
8. AWS Resource Access Manager Shares 정기적으로 검토:
    - AWS Resource Access Manager를 사용하여 Amazon VPC 내에서 AWS Network Firewall 정책, Amazon Route 53 Resolver 규칙, 서브넷과 같은 리소스를 공유할 수 있습니다.
    - 공유 리소스를 정기적으로 감사하고 더 이상 공유할 필요가 없는 리소스 공유를 중지합니다.

리소스

관련 모범 사례:

- SEC03-BP01 액세스 요구 사항 정의
- SEC03-BP02 최소 권한 액세스 부여

관련 문서:

- AWS KMS Cryptographic Details 백서
- Amazon S3 리소스에 대한 액세스 권한 관리 소개
- Overview of managing access to your AWS KMS resources
- AWS Config 규칙

- Amazon S3 + Amazon CloudFront: A Match Made in the Cloud
- 버전 관리 사용
- Amazon S3 객체 잠금을 사용하여 객체 잠금
- Sharing an Amazon EBS Snapshot
- 공유 AMI
- Hosting a single-page application on Amazon S3
- AWS 글로벌 조건 키
- Building a Data Perimeter on AWS

관련 비디오:

- Securing Your Block Storage on AWS

전송 중 데이터 보호

전송 중 데이터는 시스템 간에 전송되는 모든 데이터를 의미합니다. 여기에는 워크로드 내 리소스 간의 통신과 다른 서비스 및 최종 사용자 간의 통신이 포함됩니다. 전송 중 데이터를 적절한 수준으로 보호하면 워크로드 데이터의 기밀성과 무결성을 보호할 수 있습니다.

VPC 또는 온프레미스 위치 사이에서 데이터 보호: Amazon Virtual Private Cloud(VPC) 또는 AWS에 호스팅된 서비스에 대한 온프레미스 연결 사이에서 안전한 프라이빗 네트워크 연결을 생성하는 데 AWS PrivateLink를 사용할 수 있습니다. 프라이빗 네트워크에서와 같이 AWS 서비스, 서드파티 서비스 및 기타 AWS 계정의 서비스에 액세스할 수 있습니다. AWS PrivateLink를 사용하면 인터넷 게이트웨이 또는 NAT 없이도 IP CIDR이 중복되는 계정에서 서비스에 액세스할 수 있습니다. 또한 방화벽 규칙, 경로 정의 또는 라우팅 테이블을 구성하지 않아도 됩니다. 트래픽은 Amazon 백본에 머무르며 인터넷을 통과하지 않으므로 데이터가 보호됩니다. HIPAA 및 EU/US Privacy Shield와 같은 업계별 규정 준수 규제를 계속 준수할 수 있습니다. AWS PrivateLink는 서드파티 솔루션과 원활하게 연동하여 간소화된 글로벌 네트워크를 구축하므로 클라우드로의 마이그레이션을 가속화하고 사용 가능한 AWS 서비스를 활용할 수 있습니다.

모범 사례

- SEC09-BP01 보안 키 및 인증서 관리 구현
- SEC09-BP02 전송 중 암호화 적용
- SEC09-BP03 네트워크 통신 인증

SEC09-BP01 보안 키 및 인증서 관리 구현

전송 계층 보안(TLS) 인증서는 인터넷과 프라이빗 네트워크에서 네트워크 통신을 보호하고 웹 사이트, 리소스 및 워크로드의 ID를 설정하는 데 사용됩니다.

원하는 성과: 퍼블릭 키 인프라(PKI)에서 인증서를 프로비저닝, 배포, 저장 및 갱신할 수 있는 보안 인증서 관리 시스템. 보안 키 및 인증서 관리 메커니즘은 인증서 개인 키 구성 요소가 공개되는 것을 방지하고 정기적으로 인증서를 자동 갱신합니다. 또한 다른 서비스와 통합하여 워크로드 내부의 머신 리소스에 대한 보안 네트워크 통신 및 ID를 제공합니다. 키 구성 요소는 인적 자격 증명이 절대 접근할 수 없어야 합니다.

일반적인 안티 패턴:

- 인증서 배포 또는 갱신 프로세스 중에 수동 단계를 수행합니다.
- 프라이빗 CA를 설계할 때 인증 기관(CA) 계층 구조에 충분히 주의를 기울이지 않습니다.
- 퍼블릭 리소스에 자체 서명된 인증서를 사용합니다.

이 모범 사례 확립의 이점:

- 자동 배포 및 갱신을 통해 인증서 관리 간소화
- TLS 인증서를 사용하여 전송 중 데이터의 암호화 장려
- 인증 기관이 취한 인증서 작업의 보안 및 감사 가능성 향상
- CA 계층 구조의 여러 계층에서 관리 업무 구성

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

최신 워크로드는 TLS와 같은 PKI 프로토콜을 사용하는 암호화된 네트워크 통신을 광범위하게 사용합니다. PKI 인증서 관리는 복잡할 수 있지만 자동화된 인증서 프로비저닝, 배포 및 갱신을 통해 인증서 관리와 관련된 마찰을 줄일 수 있습니다.

AWS에서는 범용 PKI 인증서를 관리하기 위해 AWS Certificate Manager 및 AWS Private Certificate Authority(AWS Private CA)와 같은 두 가지 서비스를 제공합니다. ACM은 고객이 퍼블릭 워크로드와 프라이빗 AWS 워크로드 모두에서 사용할 인증서를 프로비저닝, 관리 및 배포하는 데 사용하는 기본 서비스입니다. ACM은 AWS Private CA를 사용하여 프라이빗 인증서를 발급하고 다른 많은 AWS관리

형 서비스와 통합하여 워크로드에 보안 TLS 인증서를 제공합니다. ACM은 Amazon Trust Services 에서 공개적으로 신뢰할 수 있는 인증서를 발급할 수도 있습니다. ACM의 퍼블릭 인증서는 퍼블릭 워크로드에 사용할 수 있습니다. 최신 브라우저와 운영 체제는 기본적으로 이러한 인증서를 신뢰하기 때문입니다.

AWS Private CA를 사용하면 자체 루트의 CA 또는 하위 CA를 설정하고 API를 통해 TLS 인증서를 발급할 수 있습니다. 이러한 종류의 인증서는 TLS 연결의 클라이언트 측에서 신뢰 체인을 제어하고 관리하는 시나리오에서 사용할 수 있습니다. TLS 사용 사례 외에도 AWS Private CA를 사용하면 사용자 지정 템플릿으로 Kubernetes 포드, Matter 디바이스 제품 증명, 코드 서명 및 기타 사용 사례에 인증서를 발급할 수 있습니다. IAM Roles Anywhere를 사용하여 프라이빗 CA에서 서명한 X.509 인증서를 발급한 온프레미스 워크로드에 임시 IAM 자격 증명을 제공할 수 있습니다.

ACM 및 AWS Private CA 외에도 AWS IoT Core는 PKI 인증서를 IoT 디바이스에 프로비저닝, 관리 및 배포하기 위한 전문 지원을 제공합니다. AWS IoT Core는 대규모 퍼블릭 키 인프라에 적용할 수 있는 IoT 디바이스 온보딩용 전문 메커니즘을 제공합니다.

Amazon API Gateway 및 Elastic Load Balancing과 같은 일부 AWS 서비스는 인증서를 사용하여 애플리케이션 연결을 보호하는 자체 기능을 제공합니다. 예를 들어 API Gateway와 Application Load Balancer(ALB)는 모두 AWS Management Console, CLI 또는 API를 사용하여 생성하고 내보내는 클라이언트 인증서를 사용하여 상호 TLS(mTLS)를 지원합니다.

프라이빗 CA 계층 구조 설정 시 고려 사항

프라이빗 CA를 설정해야 하는 경우 CA 계층 구조를 미리 적절하게 설계할 수 있도록 특별히 주의를 기울이는 것이 중요합니다. 프라이빗 CA 계층 구조를 만들 때는 CA 계층 구조의 각 수준을 별도의 AWS 계정에 배포하는 것이 좋습니다. 이 의도적인 단계는 CA 계층 구조의 각 수준에 대한 노출 영역을 줄여 CloudTrail 로그 데이터에서 이상 징후를 더 쉽게 발견하고 계정 중 하나에 대한 무단 액세스가 발생할 경우 액세스 또는 영향 범위를 줄일 수 있습니다. 루트 CA는 별도의 계정에 있어야 하며 하나 이상의 중간 CA 인증서를 발급하는 데만 사용해야 합니다.

그런 다음 루트 CA 계정과 분리된 계정에 하나 이상의 중간 CA를 생성하여 최종 사용자, 디바이스 또는 기타 워크로드에 대한 인증서를 발급합니다. 마지막으로 루트 CA에서 중간 CA로 인증서를 발급합니다. 그러면 중간 CA가 최종 사용자나 디바이스에 인증서를 발급합니다. 복원력 계획, 크로스 리전 복제, 조직 전반의 CA 공유 등을 포함하여 CA 배포 계획 및 CA 계층 설계에 대한 자세한 내용은 Planning your AWS Private CA deployment를 참조하세요.

구현 단계

1. 사용 사례에 필요한 관련 AWS 서비스 확인:

- 많은 사용 사례에서 AWS Certificate Manager를 사용하여 기존 AWS 퍼블릭 키 인프라를 활용할 수 있습니다. ACM은 웹 서버나 로드 밸런서용 또는 공개적으로 신뢰할 수 있는 인증서를 위한 기타 용도로 TLS 인증서를 배포하는 데 사용할 수 있습니다.
- 자체 프라이빗 인증 기관 계층 구조를 설정해야 하거나 내보낼 수 있는 인증서에 액세스해야 하는 경우 AWS Private CA를 고려하세요. 그런 다음 ACM에서는 AWS Private CA를 사용하여 다양한 유형의 최종 엔터티 인증서를 발급할 수 있습니다.
- 내장된 사물 인터넷(IoT) 디바이스에 대규모로 인증서를 프로비저닝해야 하는 사용 사례의 경우에는 AWS IoT Core를 고려하세요.
- Amazon API Gateway 또는 Application Load Balancer와 같은 서비스에서 네이티브 mTLS 기능을 사용하는 것을 고려하세요.

2. 가능한 경우 자동 인증서 갱신 구현:
   - 통합된 AWS 관리형 서비스와 함께 ACM에서 발급한 인증서에 대해 ACM 관리형 갱신을 사용합니다.
3. 로깅 및 감사 트레일 설정:
   - CloudTrail 로그를 활성화하여 인증 기관이 있는 계정에 대한 액세스를 추적합니다. CloudTrail에서 로그 파일 무결성 검증을 구성하여 로그 데이터의 신뢰성을 확인하는 것이 좋습니다.
   - 프라이빗 CA가 발급 또는 취소한 인증서를 나열하는 감사 보고서를 정기적으로 생성하고 검토합니다. 이러한 보고서는 S3 버킷으로 내보낼 수 있습니다.
   - 프라이빗 CA를 배포할 때는 인증서 폐기 목록(CRL)을 저장할 S3 버킷도 설정해야 합니다. 워크로드 요구 사항에 따라 이 S3 버킷을 구성하는 방법에 대한 지침은 Planning a certificate revocation list (CRL)를 참조하세요.

리소스

관련 모범 사례:

- SEC02-BP02 임시 자격 증명 사용
- SEC08-BP01 보안 키 관리 구현
- SEC09-BP03 네트워크 통신 인증

관련 문서:

- How to host and manage an entire private certificate infrastructure in AWS
- How to secure an enterprise scale ACM Private CA hierarchy for automotive and manufacturing

- Private CA best practices
- How to use AWS RAM to share your ACM Private CA cross-account

관련 비디오:

- Activating AWS Certificate Manager Private CA(워크숍)

관련 예제:

- Private CA 워크숍
- IOT Device Management 워크숍(디바이스 프로비저닝 포함)

관련 도구:

- Plugin to Kubernetes cert-manager to use AWS Private CA

SEC09-BP02 전송 중 암호화 적용

조직, 법률 및 규정 준수 요구 사항을 충족할 수 있도록 조직의 정책, 규제 의무 및 표준에 따라 정의된 암호화 요구 사항을 적용합니다. 민감한 데이터를 Virtual Private Cloud(VPC) 외부로 전송할 때 암호화된 프로토콜만 사용합니다. 암호화는 데이터가 신뢰할 수 없는 네트워크로 전송되는 경우에도 데이터 기밀성을 유지하는 데 도움이 됩니다.

원하는 성과: 데이터에 대한 무단 액세스를 완화하기 위해 리소스와 인터넷 간의 네트워크 트래픽을 암호화합니다. 보안 요구 사항에 따라 내부 AWS 환경 내에서 네트워크 트래픽을 암호화합니다. 보안 TLS 프로토콜 및 암호 세트를 사용하여 전송 중 데이터를 암호화합니다.

일반적인 안티 패턴:

- 사용 중단된 버전의 SSL, TLS 및 암호 그룹 구성 요소(예: SSL v3.0, 1024비트 RSA 키 및 RC4 암호)를 사용합니다.
- 퍼블릭 리소스에서 암호화되지 않은(HTTP) 트래픽을 허용합니다.
- 만료되기 전에 X.509 인증서를 모니터링하고 교체하지 않습니다.
- TLS에 자체 서명된 X.509 인증서를 사용합니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

AWS 서비스는 통신에 TLS를 사용하는 HTTPS 엔드포인트를 제공하여 AWS API와 통신할 때 전송 중 암호화 기능을 제공합니다. 안전하지 않은 HTTP 프로토콜은 보안 그룹을 사용하여 가상 프라이빗 클라우드(VPC)에서 감사 및 차단할 수 있습니다. HTTP 요청은 Amazon CloudFront의 HTTPS로나 Application Load Balancer에서 자동으로 리디렉션될 수도 있습니다. Amazon Simple Storage Service(Amazon S3) 버킷 정책을 사용하여 HTTP를 통해 객체를 업로드하는 기능을 제한하여 객체를 버킷에 업로드하는 데 HTTPS 사용을 효과적으로 적용할 수 있습니다. 컴퓨팅 리소스를 완전하게 제어하여 서비스 간에 전송 중 암호화를 구현할 수 있습니다. 외부 네트워크 또는 AWS Direct Connect로부터 특정 VPC로의 VPN 연결을 사용하여 트래픽을 쉽게 암호화할 수도 있습니다. AWS가 2024년 2월부터 이전 버전의 TLS 사용을 중단했으므로 클라이언트가 최소 TLS 1.2를 사용하여 AWS API를 직접 호출하는지 확인합니다. TLS 1.3을 사용할 것을 권장합니다. 전송 중 암호화에 대한 특별한 요구 사항이 있는 경우 AWS Marketplace에서 서드파티 솔루션을 찾을 수 있습니다.

구현 단계

- 전송 중 암호화 적용: 정의된 암호화 요구 사항은 최신 표준 및 모범 사례를 토대로 하고 보안 프로토콜만 허용해야 합니다. 예를 들어 Application Load Balancer 또는 Amazon EC2 인스턴스로의 HTTPS 프로토콜을 허용하는 보안 그룹만 구성합니다.
- 엣지 서비스에서 보안 프로토콜 구성: Amazon CloudFront로 HTTPS를 구성하고 보안 태세 및 사용 사례에 적합한 보안 프로파일을 사용합니다.
- 외부 연결을 위해 VPN 사용: 데이터 프라이버시와 무결성을 모두 지원할 수 있도록 지점 간 또는 네트워크 간 연결에 IPsec VPN 사용을 고려합니다.
- 로드 밸런서에서 보안 프로토콜 구성: 리스너에 연결할 클라이언트가 지원하는 가장 강력한 암호 그룹을 제공하는 보안 정책을 선택합니다. Application Load Balancer에 대한 HTTPS 리스너를 생성합니다.
- Amazon Redshift에서 보안 프로토콜 구성: 클러스터가 보안 소켓 계층(SSL) 또는 전송 계층 보안(TLS) 연결을 요구하도록 구성합니다.
- 보안 프로토콜 구성: AWS 서비스 설명서를 검토하여 전송 중 암호화 기능을 확인합니다.
- Amazon S3 버킷에 업로드할 때 보안 액세스 구성: Amazon S3 버킷 정책 제어를 사용하여 데이터에 대한 보안 액세스를 적용합니다.
- AWS Certificate Manager 사용 고려: ACM에서는 AWS 서비스에서 사용하도록 퍼블릭 TLS 인증서를 프로비저닝, 관리 및 배포할 수 있습니다.
- 프라이빗 PKI 요구 사항에 AWS Private Certificate Authority 사용 고려: AWS Private CA를 사용하면 프라이빗 인증 기관(CA) 계층 구조를 생성하여 암호화된 TLS 채널을 생성하는 데 사용할 수 있는 최종 엔터티 X.509 인증서를 발급할 수 있습니다.

리소스

관련 문서:

- CloudFront에서 HTTPS 사용
- AWS Virtual Private Network를 사용하여 VPC를 원격 네트워크에 연결
- Create an HTTPS listener for your Application Load Balancer
- Tutorial: Configure SSL/TLS on Amazon Linux 2
- SSL/TLS를 사용하여 DB 인스턴스 연결 암호화
- 연결을 위한 보안 옵션 구성

SEC09-BP03 네트워크 통신 인증

전송 계층 보안(TLS) 또는 IPsec과 같은 인증을 지원하는 프로토콜을 사용하여 통신의 자격 증명을 확인합니다.

서비스, 애플리케이션 또는 사용자 간에 통신할 때마다 안전하고 인증된 네트워크 프로토콜을 사용하도록 워크로드를 설계합니다. 인증 및 권한 부여를 지원하는 네트워크 프로토콜을 사용하면 네트워크 흐름을 더 강력하게 제어할 수 있고 무단 액세스의 영향을 줄일 수 있습니다.

원하는 성과: 서비스 간 트래픽 흐름이 잘 정의된 데이터 영역 및 컨트롤 플레인 트래픽 흐름을 보유한 워크로드. 트래픽 흐름은 기술적으로 가능한 경우 인증되고 암호화된 네트워크 프로토콜을 사용합니다.

일반적인 안티 패턴:

- 암호화되지 않았거나 인증되지 않은 트래픽이 워크로드 내에 흐릅니다.
- 여러 사용자 또는 엔터티가 인증 자격 증명을 재사용합니다.
- 액세스 제어 메커니즘으로 네트워크 제어에만 의존합니다.
- 업계 표준 인증 메커니즘에 의존하지 않고 사용자 지정 인증 메커니즘을 구축합니다.
- 서비스 구성 요소 또는 VPC의 다른 리소스 간에 지나치게 허용적인 트래픽이 흐릅니다.

이 모범 사례 확립의 이점:

- 무단 액세스의 영향 범위를 워크로드의 한 부분으로 제한합니다.

- 작업이 인증된 엔터티에 의해서만 수행되도록 더 높은 수준의 보장을 제공합니다.
- 의도된 데이터 전송 인터페이스를 명확하게 정의하고 적용함으로써 서비스의 분리를 개선합니다.
- 요청 어트리뷰션 및 잘 정의된 통신 인터페이스를 통해 모니터링, 로깅 및 인시던트 대응을 개선합니다.
- 네트워크 제어와 인증 및 권한 제어를 결합하여 워크로드에 대한 심층 방어를 제공합니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 낮음

구현 가이드

워크로드의 네트워크 트래픽 패턴은 두 가지 범주로 분류할 수 있습니다.

- 동서 트래픽은 워크로드를 구성하는 서비스 간의 트래픽 흐름을 나타냅니다.
- 남북 트래픽은 워크로드와 소비자 간의 트래픽 흐름을 나타냅니다.

남북 트래픽을 암호화하는 것이 일반적이지만 인증된 프로토콜을 사용하여 동서 트래픽을 보호하는 것은 흔하지 않습니다. 최신 보안 관행에서는 네트워크 설계만으로는 두 엔터티 간에 신뢰할 수 있는 관계를 부여하지 않을 것을 권장합니다. 두 서비스가 공통 네트워크 경계 내에 있을 수 있는 경우에도 이러한 서비스 간의 통신을 암호화 및 인증하고 권한을 부여하는 것이 가장 좋습니다.

예를 들어, AWS 서비스 API는 요청이 시작된 네트워크와 상관없이 AWS Signature Version 4(SigV4) 서명 프로토콜을 사용하여 호출자를 인증합니다. 이 인증을 통해 AWS API는 작업을 요청한 자격 증명을 확인할 수 있으며, 그런 다음 해당 자격 증명을 정책과 결합하여 권한 부여 결정을 내려 작업의 허용 여부를 결정할 수 있습니다.

Amazon VPC Lattice 및 Amazon API Gateway와 같은 서비스를 사용하면 동일한 SigV4 서명 프로토콜을 사용하여 자체 워크로드의 동서 트래픽에 인증 및 권한 부여를 추가할 수 있습니다. AWS 환경 외부의 리소스가 SigV4 기반 인증 및 권한 부여가 필요한 서비스와 통신해야 하는 경우 AWS 이외 리소스에서 AWS Identity and Access Management(IAM) Roles Anywhere를 사용하여 임시 AWS 자격 증명을 얻을 수 있습니다. 이러한 자격 증명을 사용하여 액세스 권한 부여에 SigV4를 사용하는 서비스에 대한 요청에 서명할 수 있습니다.

동서 트래픽을 인증하는 또 다른 일반적인 메커니즘은 TLS 상호 인증(mTLS)입니다. 많은 사물 인터넷(IoT), B2B 애플리케이션 및 마이크로서비스는 mTLS를 사용하여 클라이언트 및 서버 측 X.509 인증서를 모두 사용하여 TLS 통신 양측의 자격 증명을 확인합니다. 이러한 인증서는 AWS Private Certificate Authority(AWS Private CA)에서 발급할 수 있습니다. Amazon API Gateway와 같은 서비스를 사용하여 워크로드 간 또는 워크로드 내부 통신을 위한 mTLS 인증을 제공할 수 있습니다. Application Load

Balancer는 내부 또는 외부 대상 워크로드에 대한 mTLS도 지원합니다. mTLS는 TLS 통신 양쪽에 대한 인증 정보를 제공하지만 권한 부여 메커니즘을 제공하지는 않습니다.

마지막으로 OAuth 2.0 및 OIDC(OpenID Connect)는 일반적으로 사용자의 서비스 액세스를 제어하는 데 사용되는 프로토콜이지만 이제는 서비스 간 트래픽에서도 널리 사용되고 있습니다. API Gateway는 JSON 웹 토큰(JWT) 권한 부여자를 제공하여 워크로드가 OIDC 또는 OAuth 2.0 ID 제공업체에서 발급한 JWT를 사용하여 API 경로에 대한 액세스를 제한할 수 있도록 합니다. OAuth2 범위는 기본 권한 부여 결정을 위한 소스로 사용될 수 있지만, 애플리케이션 계층에서 여전히 권한 부여 검사를 구현해야 하며, OAuth2 범위만으로는 더 복잡한 권한 부여 요구 사항을 지원할 수 없습니다.

구현 단계

- 워크로드 네트워크 흐름 정의 및 문서화: 심층 방어 전략을 구현하기 위한 첫 번째 단계는 워크로드의 트래픽 흐름을 정의하는 것입니다.
  - 워크로드를 구성하는 여러 서비스 간에 데이터가 전송되는 방식을 명확하게 정의하는 데이터 흐름도를 만듭니다. 이 다이어그램은 인증된 네트워크 채널을 통해 이러한 흐름을 적용하는 첫 번째 단계입니다.
  - 개발 및 테스트 단계에서 워크로드를 계측하여 데이터 흐름도가 런타임 시 워크로드의 동작을 정확하게 반영하는지 확인합니다.
  - 데이터 흐름도는 SEC01-BP07 위협 모델을 사용하여 위협 식별 및 완화 조치의 우선순위 지정에서 설명한 대로 위협 모델링 연습을 수행할 때도 유용할 수 있습니다.
- 네트워크 제어 설정: 데이터 흐름에 맞게 네트워크 제어를 설정할 수 있는 AWS 기능을 고려합니다. 네트워크 경계가 유일한 보안 제어가 되어서는 안 되지만, 네트워크 경계는 워크로드를 보호하기 위한 심층 방어 전략의 한 계층을 제공합니다.
  - 보안 그룹을 사용하여 리소스 간 데이터 흐름을 설정, 정의 및 제한합니다.
  - AWS 및 AWS PrivateLink를 지원하는 서드파티 서비스 모두와 통신하기 위해 AWS PrivateLink 사용을 고려하세요. AWS PrivateLink 인터페이스 엔드포인트를 통해 전송된 데이터는 AWS 네트워크 백본 내에 머물며 퍼블릭 인터넷을 통과하지 않습니다.
- 워크로드의 서비스 전반에 인증 및 권한 부여 구현: 워크로드에서 인증되고 암호화된 트래픽 흐름을 제공하는 데 가장 적합한 AWS 서비스 세트를 선택합니다.
  - 서비스 간 통신을 보호하려면 Amazon VPC Lattice를 고려하세요. VPC Lattice는 인증 정책과 결합된 SigV4 인증을 사용하여 서비스 간 액세스를 제어할 수 있습니다.
  - mTLS를 사용한 서비스 간 통신의 경우 API Gateway, Application Load Balancer를 고려해 보세요. AWS Private CA는 mTLS와 함께 사용할 인증서를 발급할 수 있는 프라이빗 CA 계층 구조를 설정하는 데 사용할 수 있습니다.

  - OAuth 2.0 또는 OIDC를 사용하여 서비스와 통합할 때는 JWT 권한 부여자를 사용하여 API Gateway를 사용하는 것을 고려하세요.
  - 워크로드와 IoT 디바이스 간 통신의 경우 네트워크 트래픽 암호화 및 인증을 위한 여러 옵션을 제공하는 AWS IoT Core를 고려해 보세요.
- 무단 액세스 모니터링: 의도하지 않은 통신 채널, 보호된 리소스에 대한 무단 액세스 시도 및 기타 부적절한 액세스 패턴을 지속적으로 모니터링합니다.
  - VPC Lattice를 사용하여 서비스에 대한 액세스를 관리하는 경우 VPC Lattice 액세스 로그를 활성화하고 모니터링하는 방법을 고려해 보세요. 이러한 액세스 로그에는 요청 엔티티에 대한 정보, 소스 및 대상 VPC를 비롯한 네트워크 정보, 요청 메타데이터가 포함됩니다.
  - VPC 흐름 로그를 사용하여 네트워크 흐름의 메타데이터를 캡처하고 주기적으로 이상 징후를 검토하는 방법을 고려해 보세요.
  - 보안 인시던트 계획, 시뮬레이션 및 대응에 대한 자세한 지침은 AWS Security Incident Response Guide 및 AWS Well-Architected Framework 보안 원칙의 인시던트 대응 섹션을 참조하세요.

리소스

관련 모범 사례:

- SEC03-BP07 퍼블릭 및 크로스 계정 액세스 분석
- SEC02-BP02 임시 자격 증명 사용
- SEC01-BP07 위협 모델을 사용하여 위협 식별 및 완화 조치의 우선순위 지정

관련 문서:

- Evaluating access control methods to secure Amazon API Gateway APIs
- REST API에 대한 상호 TLS 인증 구성
- How to secure API Gateway HTTP endpoints with JWT authorizer
- Authorizing direct calls to AWS services using AWS IoT Core credential provider
- AWS Security Incident Response Guide

관련 비디오:

- AWS re:invent 2022: Introducing VPC Lattice
- AWS re:invent 2020: Serverless API authentication for HTTP APIs on AWS

관련 예제:

- Amazon VPC Lattice 워크숍
- 제로 트러스트 에피소드 1 - The Phantom Service Perimeter 워크숍

인시던트 대응

예방 및 탐지 제어를 사용하더라도 조직은 잠재적 보안 인시던트에 대응하고 그 영향을 완화하기 위한 메커니즘을 구현해야 합니다. 이러한 준비는 인시던트 발생 시 보안팀이 효과적으로 문제를 격리 및 억제하고 문제에 대한 포렌식을 수행하고, 운영을 알려진 정상 상태로 복구하는 능력에 지대한 영향을 미칩니다. 보안 인시던트보다 앞서 도구 및 액세스를 마련하고 게임 데이를 통해 인시던트 대응을 정기적으로 연습한다면 비즈니스 중단을 최소화하면서 복구할 수 있습니다.

주제

- AWS 인시던트 대응 측면
- 클라우드 응답의 설계 목표
- 준비
- 운영
- 인시던트 사후 활동

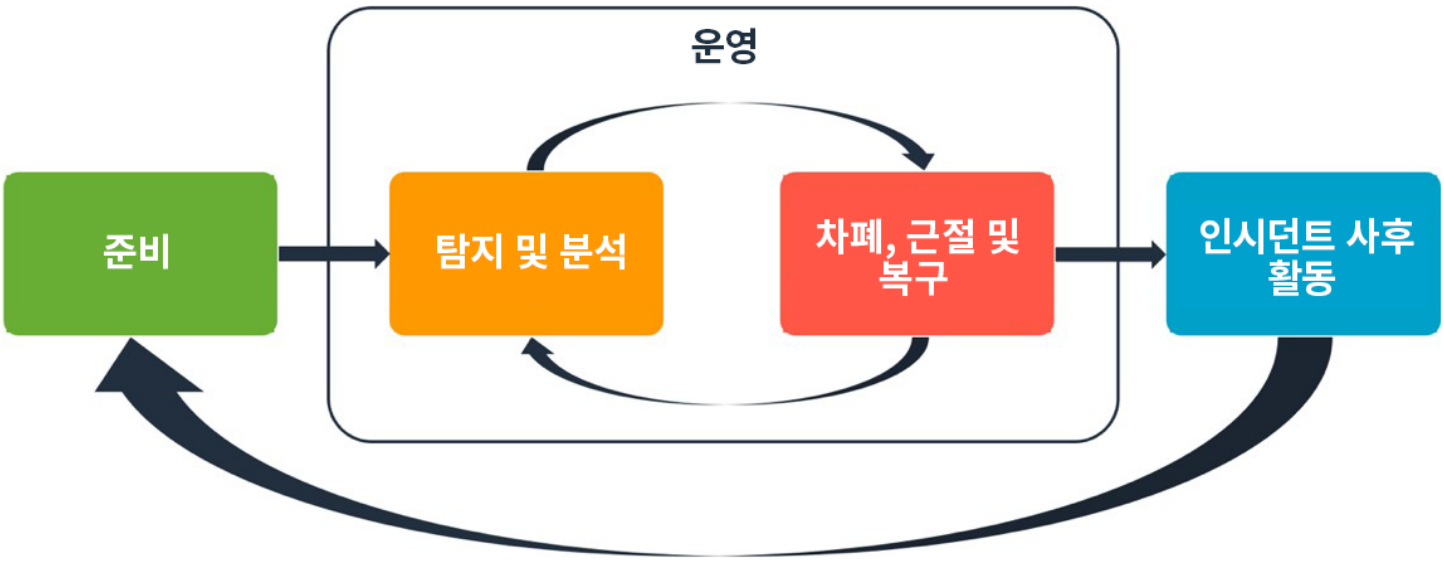
AWS 인시던트 대응 측면

조직 내 모든 AWS 사용자는 보안 인시던트 대응 프로세스에 대한 기본적인 이해가 있어야 하며 보안 직원은 보안 문제에 대응하는 방법을 이해해야 합니다. 교육, 훈련 및 경험은 성공적인 클라우드 인시던트 대응 프로그램에 필수적이며 발생 가능한 보안 인시던트를 처리하기 전에 미리 구현하는 것이 이상적입니다. 클라우드에서의 성공적인 인시던트 대응 프로그램은 준비, 운영, 인시던트 사후 활동에 기반합니다.

이러한 각 측면을 이해하려면 다음 설명을 고려하세요.

- 준비: 탐지 제어를 활성화하고 필요한 도구 및 클라우드 서비스에 대한 적절한 액세스를 확인하여 인시던트 대응 팀이 AWS 내부 인시던트를 탐지하고 이에 대응할 수 있도록 준비합니다. 또한 신뢰할 수 있는 일관된 응답을 보장하는 데 필요한 수동 및 자동 런북을 준비합니다.
- 운영: 탐지, 분석, 격리, 근절 및 복구와 같은 NIST의 사고 대응 단계에 따라 보안 이벤트 및 잠재적 인시던트를 해결합니다.
- 인시던트 사후 활동: 보안 이벤트 및 시뮬레이션의 결과를 반복하여 대응의 효율성을 높이고, 대응 및 조사를 통해 도출된 가치를 높이며, 위험을 더욱 줄입니다. 인시던트를 통해 배우고 개선 활동에 대한 강한 주인의식을 가져야 합니다.

다음 다이어그램에서는 이러한 측면의 흐름을 보여줍니다. 흐름은 앞서 언급한 NIST 인시던트 대응 수명 주기와 일치하지만 탐지 및 분석, 격리, 근절 및 복구를 포함하는 작업을 수행합니다.



AWS 인시던트 대응 측면

클라우드 응답의 설계 목표

NIST SP 800-61 Computer Security Incident Handling Guide에 정의된 일반적인 인시던트 대응 프로세스 및 메커니즘도 여전히 유효하지만, 클라우드 환경에서 보안 인시던트에 대응하는 것과 관련된 구체적인 설계 목표를 평가해 보는 것이 좋습니다.

- 대응 목표 수립: 이해관계자, 법률 자문, 조직 리더십과 협력하여 인시던트 대응 목표를 결정합니다. 몇 가지 공통 목표로, 문제 억제 및 완화, 영향을 받는 리소스 복구, 포렌식을 위한 데이터 보존, 알려진 안전한 운영 환경으로 복구, 궁극적으로 인시던트를 통한 학습 등이 포함됩니다.
- 클라우드를 사용하여 대응: 이벤트와 데이터가 존재하는 곳에 응답 패턴을 구현합니다.
- 무엇을 가지고 있고 무엇이 필요한지 파악: 로그, 리소스, 스냅샷 및 기타 증거를 응답 전용 중앙 집중식 클라우드 계정에 복사 및 저장하여 보존합니다. 태그, 메타데이터, 보존 정책을 적용하는 메커니즘을 사용합니다. 어떤 서비스를 사용하고 있는지 파악한 다음 해당 서비스를 조사하기 위한 요구 사항을 파악해야 합니다. 환경을 이해하는 데 도움이 되도록 태그 지정을 사용할 수도 있습니다.
- 재배포 메커니즘 사용: 잘못된 구성으로 인해 보안 이상이 발생한 경우 올바른 구성으로 리소스를 재배포하여 변형을 제거하는 것만으로 간단하게 문제를 해결할 수 있습니다. 손상 가능성이 확인되면 재배포에 성공적이고 검증된 근본 원인 완화 조치가 포함되어 있는지 확인하세요.
- 가능한 경우 자동화: 문제가 발생하거나 인시던트가 반복되면 프로그래밍 방식으로 분류하고 일반적인 이벤트에 대응하는 메커니즘을 구축하세요. 자동화가 불가능한 고유하거나 복잡하거나 민감한 인시던트에는 사람의 대응을 활용하세요.

- 확장 가능한 솔루션 선택: 클라우드 컴퓨팅에 대한 조직의 접근 방식의 확장성과 일치하도록 노력하세요. 환경 전반으로 확장되는 탐지 및 대응 메커니즘을 구현하여 탐지와 대응 사이의 시간을 효과적으로 줄이세요.
- 프로세스 교육 및 개선: 프로세스, 도구 또는 인력의 격차를 사전에 파악하고 이를 해결하기 위한 계획을 실행하세요. 시뮬레이션은 격차를 찾고 프로세스를 개선할 수 있는 안전한 방법입니다.

이러한 설계 목표는 인시던트 대응과 위협 탐지를 모두 수행할 수 있는지 아키텍처 구현을 검토하도록 상기시켜줍니다. 클라우드 구현을 계획할 때는 포렌식에 기반하여 타당한 대응 방법론을 사용하여 인시던트에 대응하는 방안을 생각해 보세요. 경우에 따라 이러한 대응 작업을 위해 특별히 설정된 조직, 계정 및 도구가 여러 개 있을 수 있습니다. 이러한 도구와 기능은 배포 파이프라인을 통해 인시던트 대응 담당자가 사용할 수 있도록 해야 합니다. 더 큰 위험을 초래할 수 있으므로 정적이어서는 안 됩니다.

준비

인시던트 대비는 시기적절하고 효과적인 인시던트 대응을 위해 매우 중요합니다. 준비는 세 가지 영역에서 이루어집니다.

- 사람: 보안 인시던트에 대비하려면 인시던트 대응을 위한 관련 이해관계자를 식별하고 인시던트 대응 및 클라우드 기술에 대해 교육해야 합니다.
- 프로세스: 보안 인시던트에 대비하여 프로세스를 준비하려면 아키텍처 문서화, 철저한 인시던트 대응 계획 개발, 보안 이벤트에 대한 일관된 대응을 위한 플레이북 작성이 포함됩니다.
- 기술: 보안 인시던트에 대비한 기술 준비에는 액세스 설정, 필요한 로그 집계 및 모니터링, 효과적인 경고 메커니즘 구현, 대응 및 조사 기능 개발이 포함됩니다.

이러한 각 영역은 효과적인 인시던트 대응을 위해 동일하게 중요합니다. 이 세 가지가 모두 없으면 인시던트 대응 프로그램이 완전하거나 효과적이지 않습니다. 인시던트에 대비하려면 긴밀한 통합을 통해 직원, 프로세스 및 기술을 준비해야 합니다.

모범 사례

- SEC10-BP01 주요 직원과 외부 리소스 파악
- SEC10-BP02 인시던트 관리 계획 개발
- SEC10-BP03 포렌식 역량 확보
- SEC10-BP04 보안 인시던트 대응 플레이북 개발 및 테스트
- SEC10-BP05 액세스 권한 사전 프로비저닝

- SEC10-BP06 도구 사전 배포
- SEC10-BP07 시뮬레이션 실행

SEC10-BP01 주요 직원과 외부 리소스 파악

조직이 인시던트에 대응하는 데 도움이 될 수 있는 내부 및 외부 직원, 리소스, 법적 의무를 파악합니다.

원하는 성과: 주요 담당자, 연락처 정보, 보안 이벤트 대응 시 수행하는 역할 목록이 있습니다. 이 정보를 정기적으로 검토하고 내부 및 외부 도구 관점에서 직원 변경 사항을 반영하도록 업데이트합니다. 이러한 정보를 문서화할 때는 보안 파트너, 클라우드 제공업체, 서비스형 소프트웨어(SaaS) 애플리케이션을 비롯한 모든 서드파티 서비스 제공업체 및 공급업체를 고려합니다. 보안 이벤트 중에는 적절한 수준의 책임을 맡고 상황 정보를 알고 있으며 액세스 권한을 가진 직원이 대응하고 복구할 수 있습니다.

일반적인 안티 패턴:

- 보안 이벤트 대응 시 연락처 정보, 역할, 담당 업무가 나와 있는 주요 인력의 최신 목록을 유지 관리하지 않습니다.
- 이벤트에 대응하고 이벤트에서 복구할 때 모두가 인력, 종속성, 인프라, 솔루션을 이해하고 있다고 가정합니다.
- 주요 인프라 또는 애플리케이션 설계를 나타내는 문서 또는 정보 리포지토리가 없습니다.
- 신입 직원이 보안 이벤트 대응에 효과적으로 기여할 수 있도록 돕는 적절한 온보딩 프로세스(예: 이벤트 시뮬레이션 수행)가 없습니다.
- 보안 이벤트 중에 주요 인력이 일시적으로 부재하거나 대응에 실패할 경우에 대비하여 에스컬레이션 경로가 마련되어 있지 않습니다.

이 모범 사례 확립의 이점: 이 방법을 사용하면 이벤트 중에 적합한 담당자와 역할을 식별하는 데 걸리는 분류 및 대응 시간이 단축됩니다. 주요 담당자 및 역할 목록을 업데이트하여 이벤트가 발생한 동안 낭비되는 시간을 최소화함으로써 적절한 인력 배치로 이벤트를 분류하고 복구할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

조직의 주요 담당자 파악: 참여해야 하는 조직 내 담당자의 연락처 목록을 유지 관리합니다. 운영상 변화, 승진, 팀 변경 등 인사이동이 발생하는 경우 관련 정보를 정기적으로 검토하고 업데이트하세요. 이는 인시던트 관리자, 인시던트 대응 담당자, 커뮤니케이션 책임자와 같은 주요 역할에 특히 중요합니다.

- 인시던트 관리자: 인시던트 관리자는 이벤트 대응 과정에서 전반적인 권한을 보유합니다.
- 인시던트 담당자: 인시던트 담당자는 조사 및 수정 활동을 담당합니다. 이러한 담당자는 이벤트 유형에 따라 다를 수 있지만. 보통 영향을 받는 애플리케이션을 다루는 개발자와 운영 팀이 역할을 맡습니다.
- 커뮤니케이션 책임자: 커뮤니케이션 책임자는 내부 및 외부 커뮤니케이션, 특히 공공 기관, 규제 기관 및 고객과의 커뮤니케이션을 담당합니다.
- 온보딩 프로세스: 인시던트 대응 노력에 효과적으로 기여하는 데 필요한 스킬과 지식을 갖추도록 신입 직원을 정기적으로 훈련하고 온보딩합니다. 온보딩 프로세스의 일부로 시뮬레이션 및 실습을 포함하여 준비를 촉진합니다.
- 분야별 전문가(SME): 분산되고 자율적인 팀의 경우 미션 크리티컬 워크로드를 처리할 SME를 파악하는 것이 좋습니다. SME는 이벤트와 관련된 중요 워크로드의 운영 및 데이터 분류에 대한 인사이트를 제공합니다.

예시 테이블 형식:

```
  | Role | Name | Contact Information | Responsibilities |
1 | ——- | ——- | ——- | ——- |
2 | Incident Manager | Jane Doe| jane.doe@example.com | Overall authority during
 response |
3 | Incident Responder | John Smith | john.smith@example.com | Investigation and
 remediation |
4 | Communications Lead | Emily Johnson | emily.johnson@example.com | Internal and
 external communications |
5 | Communications Lead | Michael Brown | michael.brown@example.com | Insights on
 critical workloads |
```

AWS Systems Manager Incident Manager 기능을 사용하여 주요 담당자를 포착하고, 대응 계획을 정의하며, 당직 일정을 자동화하고, 에스컬레이션 계획을 생성하는 것을 고려해 보세요. 당직 일정에 따라 모든 직원을 자동화하고 교체하여 워크로드에 대한 책임을 여러 담당자가 분담하도록 합니다. 이를 통해 관련 지표 및 로그를 내보내고 워크로드에 중요한 경보 임곗값을 정의하는 등의 모범 사례를 활용할 수 있습니다.

외부 파트너 식별: 기업은 독립 소프트웨어 개발 판매 회사(ISV), 파트너 및 하청업체가 구축한 도구를 사용하여 고객을 위한 차별화 솔루션을 구축합니다. 인시던트에 대응하고 인시던트로부터 복구하는 데 도움을 줄 수 있는 주요 담당자를 참여시키세요. 지원 사례를 통해 AWS 분야별 전문가의 도움을 빠르게 받으려면 적절한 수준의 지원에 가입하는 것이 좋습니다. 워크로드에 대해 모든 주요 솔루션 제공업체와 유사한 계약을 체결하는 것을 고려하세요. 일부 보안 이벤트의 경우 상장 기업은 관련 공공 기

관 및 규제 기관에 이벤트와 영향을 알려야 합니다. 관련 부서 및 담당자의 연락처 정보를 유지 및 업데이트하세요.

구현 단계

1. 인시던트 관리 솔루션을 설정합니다.
   a. 보안 도구 계정에 Incident Manager를 배포하는 것을 고려해 보세요.
2. 인시던트 관리 솔루션에서 담당자를 정의합니다.
   a. 인시던트 발생 시 연락이 안 되는 일이 없도록 각 담당자에 대해 최소 2가지 유형의 연락 채널(예: SMS, 전화 또는 이메일)을 정의합니다.
3. 대응 계획을 정의합니다.
   a. 인시던트 발생 시 가장 적절한 담당자를 식별합니다. 개별 담당자가 아닌 참여 대상 직원의 역할에 맞게 에스컬레이션 계획을 정의합니다. 외부 주체가 인시던트 해결에 직접 관여하지 않았더라도 외부 기관에 알릴 책임이 있는 담당자를 포함시키는 것이 좋습니다.

리소스

관련 모범 사례:

- OPS02-BP03 운영 활동에서 성능을 담당하는 소유자 식별

관련 문서:

- AWS Security Incident Response Guide

관련 예제:

- AWS customer playbook framework
- Prepare for and respond to security incidents in your AWS environment

관련 도구:

- AWS Systems Manager Incident Manager

관련 비디오:

- Amazon's approach to security during development

SEC10-BP02 인시던트 관리 계획 개발

인시던트 대응을 위해 작성해야 할 첫 번째 문서는 인시던트 대응 계획입니다. 인시던트 대응 계획은 인시던트 대응 프로그램 및 전략의 기초가 되도록 설계되었습니다.

이 모범 사례 확립의 이점: 철저하고 명확하게 정의된 인시던트 대응 프로세스를 개발하는 것은 성공적이고 확장 가능한 인시던트 대응 프로그램의 핵심입니다. 보안 이벤트가 발생하면 명확한 단계 및 워크플로가 적시에 대응하는 데 도움이 됩니다. 기존 인시던트 대응 프로세스가 이미 있을 수 있습니다. 현재 상태에 관계없이 인시던트 대응 프로세스를 정기적으로 업데이트, 반복, 테스트하는 것이 중요합니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 가이드

인시던트 관리 계획은 보안 인시던트의 잠재적 영향에 대한 대응, 완화 및 복구에 매우 중요합니다. 인시던트 관리 계획은 보안 인시던트를 적시에 파악하고 해결 및 대응하기 위한 구조화된 프로세스입니다.

클라우드에는 온프레미스 환경에서 볼 수 있는 수많은 동일한 운영 역할과 요구 사항이 있습니다. 인시던트 관리 계획을 수립할 때는 비즈니스 성과와 규정 준수 요구 사항에 가장 잘 맞는 대응 및 복구 전략을 고려하는 것이 중요합니다. 예를 들어, 미국 내 FedRAMP 규정을 준수하는 AWS에서 워크로드를 운영하는 경우 NIST SP 800-61 Computer Security Handling Guide의 권장 사항을 따르세요. 마찬가지로 개인 식별 정보(PII)를 저장하는 워크로드를 운영할 때는 데이터 레지던시 및 사용과 관련된 문제를 보호하고 대응하는 방법을 고려합니다.

AWS에서 워크로드에 대한 인시던트 관리 계획을 구축하는 경우, 인시던트 대응에 대한 심층 방어 방식을 구축하기 위해 AWS 공동 책임 모델부터 시작합니다. 이 모델에서 AWS는 클라우드 자체의 보안을 관리하지만 클라우드 내에서 보안을 유지하는 것은 고객의 책임입니다. 즉, 고객은 구현을 선택하는 보안 제어에 대한 제어 권한을 보유하며 이에 대한 책임이 있습니다. AWS Security Incident Response Guide에서는 클라우드 중심 인시던트 관리 계획을 구축하기 위한 주요 개념과 기본적인 지침을 자세히 설명합니다.

효과적인 인시던트 관리 계획은 클라우드 운영 목표와 함께 끊임없이 반복되고 항상 최신 상태를 유지해야 합니다. 인시던트 관리 계획을 수립 및 개선할 때 아래에서 자세히 설명하는 구현 계획의 사용을 고려해 볼 수 있습니다.

구현 단계

1. 조직 내에서 보안 이벤트를 처리하기 위한 역할과 책임을 정의합니다. 여기에는 다음을 포함한 다양한 부서의 담당자가 참여해야 합니다.
   - 인적 자원(HR)
   - 경영진
   - 법무 부서
   - 애플리케이션 소유자 및 개발자(주제 전문가 또는 SME)
2. 인시던트 발생 시 업무에 대한 책임을 지는 사람, 결과에 대한 책임을 지는 사람, 상의해야 할 사람, 정보를 제공해야 할 사람(RACI)을 명확하게 설명합니다. RACI 차트를 생성하여 빠르고 직접적인 커뮤니케이션을 촉진하고 이벤트의 다양한 단계에서 리더십을 명확하게 설명합니다.
3. 인시던트 중에 애플리케이션 소유자와 개발자(SME)가 영향을 측정하는 데 도움이 되는 귀중한 정보와 컨텍스트를 제공할 수 있으므로 참여시킵니다. 이러한 SME와 관계를 구축하고 실제 인시던트가 발생하기 전에 해당 SME와 인시던트 대응 시나리오를 연습합니다.
4. 신뢰할 수 있는 파트너 또는 외부 전문가가 추가적인 전문성과 관점을 제공할 수 있으므로 조사 또는 대응 프로세스에 참여시킵니다.
5. 인시던트 관리 계획 및 역할을 조직에 적용되는 모든 현지 규정 또는 규정 준수 요구 사항에 맞게 조정합니다.
6. 인시던트 대응 계획을 정기적으로 연습 및 테스트하고 정의된 모든 역할과 책임을 포함합니다. 이렇게 하면 프로세스를 간소화하고 보안 인시던트에 대한 조정되고 효율적인 대응 조치가 있는지 확인할 수 있습니다.
7. 역할, 책임 및 RACI 차트를 정기적으로 또는 조직 구조 또는 요구 사항이 변경될 때 검토 및 업데이트합니다.

AWS 대응 팀 및 지원 이해

- AWS Support
  - 지원는 다양한 플랜을 제공합니다. 이러한 플랜을 통해 AWS 솔루션의 성공과 운영 상태를 지원하는 도구 및 전문 지식에 액세스할 수 있습니다. AWS 환경을 계획, 배포, 최적화하는 데 도움이 되는 기술 지원 및 추가 리소스가 필요한 경우 AWS 사용 사례에 가장 적합한 지원 플랜을 선택할 수 있습니다.
  - AWS Management Console의 지원 센터(로그인이 필요함)를 AWS 리소스에 영향을 미치는 문제에 대한 지원을 받을 수 있는 중앙 연락 창구로 고려하세요. 지원에 대한 액세스는 AWS Identity

and Access Management로 제어됩니다. 지원 기능에 액세스하는 방법에 대한 자세한 내용은 Getting started with 지원를 참조하세요.

- AWS 고객 인시던트 대응 팀(CIRT)
  - AWS 고객 인시던트 대응 팀(CIRT)은 24/7로 운영되는 전문 글로벌 AWS 팀으로, AWS 공동 책임 모델의 고객 측에서 보안 이벤트가 진행되는 동안 고객을 지원합니다.
  - 고객을 지원할 때 AWS CIRT는 AWS에서의 활성 보안 이벤트 분류 및 복구를 지원합니다. 팀은 AWS 서비스 로그를 사용하여 근본 원인 분석을 지원하고 복구를 위한 권장 사항을 제공할 수 있습니다. 또한 향후 보안 이벤트를 방지하는 데 도움이 되는 보안 권장 사항 및 모범 사례를 제공할 수 있습니다.
  - AWS 고객은 지원 사례를 통해 AWS CIRT의 지원을 요청할 수 있습니다.
- DDoS 대응 지원
  - AWS에서는 AWS Shield를 제공합니다. 이를 통해 AWS에서 실행 중인 웹 애플리케이션을 보호하는 관리형 분산 서비스 거부(DDoS) 보호 서비스를 제공합니다. Shield는 애플리케이션 가동 중지 시간과 대기 시간을 최소화하는 상시 탐지 및 자동 인라인 완화를 제공하므로 지원를 이용하지 않고도 DDoS 보호를 활용할 수 있습니다. Shield에는 AWS Shield Standard 및 AWS Shield Advanced와 같은 두 가지 티어가 있습니다. 이 두 티어의 차이점에 대해 알아보려면 Shield 기능 설명서를 참조하세요.
- AWS Managed Services(AMS)
  - AWS Managed Services(AMS)에서는 AWS 인프라를 지속적으로 관리하므로 사용자는 애플리케이션에 집중할 수 있습니다. 인프라를 유지 관리하기 위한 모범 사례를 구현함으로써 AMS는 운영 오버헤드와 위험을 줄이도록 지원합니다. AMS는 변경 요청, 모니터링, 패치 관리, 보안, 백업 서비스 등과 같은 일반적인 활동을 자동화하고 인프라를 프로비저닝, 운영 및 지원하기 위한 전체 수명 주기 서비스를 제공합니다.
  - AMS는 일련의 보안 탐지 제어를 배포하고 경고에 대한 일차 대응을 연중무휴로 제공합니다. 경고가 시작되면 AMS는 일련의 표준 자동 및 수동 플레이북에 따라 일관된 응답을 확인합니다. 이러한 플레이북은 온보딩 중에 AMS 고객과 공유되므로 고객이 AMS를 통해 대응 방안을 개발하고 조정할 수 있습니다.

인시던트 대응 계획 개발

인시던트 대응 계획은 인시던트 대응 프로그램 및 전략의 기초가 되도록 설계되었습니다. 인시던트 대응 계획은 공식 문서에 포함되어야 합니다. 인시던트 대응 계획에는 일반적으로 다음 섹션이 포함됩니다.

- 인시던트 대응 팀 개요: 인시던트 대응 팀의 목표와 기능을 간략하게 설명합니다.

- 역할 및 책임: 인시던트 대응 이해관계자를 나열하고 인시던트 발생 시 해당 이해관계자의 역할을 자세히 설명합니다.
- 커뮤니케이션 계획: 연락처 정보 및 인시던트 발생 시 커뮤니케이션 방법을 자세히 설명합니다.
- 커뮤니케이션 방법 백업: 인시던트 커뮤니케이션의 백업으로 대역 외 통신을 사용하는 것이 모범 사례입니다. 안전한 대역 외 통신 채널을 제공하는 애플리케이션의 예는 AWS Wickr입니다.
- 인시던트 대응 단계 및 수행할 조치: 인시던트 대응의 단계(예: 탐지, 분석, 제거, 억제, 복구)를 열거합니다. 여기에는 해당 단계 내에서 취해야 할 상위 수준 조치가 포함됩니다.
- 인시던트 심각도 및 우선순위 정의: 인시던트의 심각도를 분류하는 방법, 인시던트의 우선순위를 지정하는 방법, 심각도 정의가 에스컬레이션 절차에 미치는 영향을 자세히 설명합니다.

이러한 섹션은 규모 및 업종이 다른 회사 간에 공통적으로 사용되지만 각 조직의 인시던트 대응 계획은 고유합니다. 조직에 가장 적합한 인시던트 대응 계획을 수립해야 합니다.

리소스

관련 모범 사례:

- SEC04 Detection

관련 문서:

- AWS Security Incident Response Guide
- NIST: Computer Security Incident Handling Guide

SEC10-BP03 포렌식 역량 확보

보안 인시던트에 앞서 보안 이벤트 조사를 지원하기 위한 포렌식 기능을 개발하는 것이 좋습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

기존 온프레미스 포렌식의 개념이 AWS에 적용됩니다. AWS 클라우드에서 포렌식 역량 구축을 시작하는 데 필요한 주요 정보는 Forensic investigation environment strategies in the AWS 클라우드를 참조하세요.

포렌식을 위한 환경과 AWS 계정 계정 구조를 설정한 후에는 네 단계에 걸쳐 포렌식 방법론을 효과적으로 수행하는 데 필요한 기술을 정의하세요.

- 수집: AWS CloudTrail, AWS Config, VPC 흐름 로그, 호스트 수준 로그 등 관련 AWS 로그를 수집합니다. 가능한 경우 영향을 받은 AWS 리소스의 스냅샷, 백업, 메모리 덤프를 수집합니다.
- 검사: 관련 정보를 추출하고 평가하여 수집한 데이터를 검사합니다.
- 분석: 수집된 데이터를 분석하여 인시던트를 이해하고 결론을 도출합니다.
- 보고: 분석 단계의 결과 정보를 제시합니다.

구현 단계

포렌식 환경 준비

AWS Organizations는 AWS 리소스의 성장과 규모 조정에 따라 AWS 환경을 중앙에서 관리하고 제어할 수 있도록 도와줍니다. AWS 조직은 AWS 계정을 통합하여 단일 단위로 관리할 수 있도록 합니다. 조직 단위(OU)를 사용하면 계정을 그룹으로 만들어 단일 유닛으로 관리할 수 있습니다.

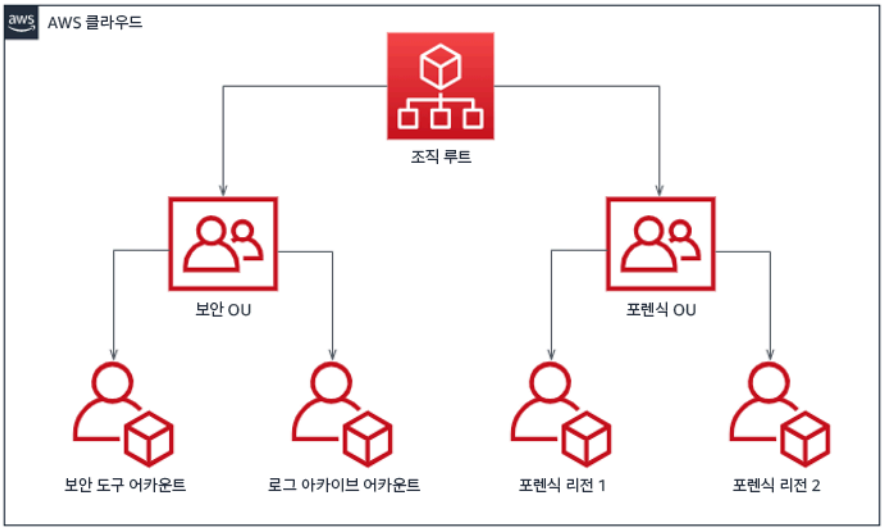
인시던트 대응에는 보안 OU 및 포렌식 OU를 포함하는 인시던트 대응 기능을 지원하는 AWS 계정 구조를 갖는 것이 도움이 됩니다. 보안 OU 내에는 다음에 대한 계정이 있어야 합니다.

- 로그 보관: 제한된 권한으로 로그 아카이브 AWS 계정에서 로그를 집계합니다.
- 보안 도구: 보안 도구 AWS 계정에서 보안 서비스를 중앙 집중화합니다. 이 계정은 보안 서비스에 대한 위임된 관리자 역할을 합니다.

포렌식 OU 내에서, 비즈니스와 운영 모델에 가장 적합한 것에 따라 운영하는 각 리전에 대해 단일 포렌식 계정 또는 여러 개의 계정을 구현할 수 있는 옵션이 있습니다. 리전별로 포렌식 계정을 생성하면 해당 리전 외부에서 AWS 리소스를 생성하는 것을 차단하고 리소스가 의도하지 않은 리전으로 복사되는 위험을 줄일 수 있습니다. 예를 들어, 미국 동부(버지니아 북부) 리전(`us-east-1`) 및 미국 서부(오레곤)(`us-west-2`)에서만 비즈니스를 운영하는 경우 포렌식 OU에는 두 개의 계정(`us-east-1`에 대한 계정과 `us-west-2`에 대한 계정)이 있습니다.

여러 리전에 대한 포렌식 AWS 계정 계정을 만들 수 있습니다. 데이터 주권 요구 사항을 준수하고 있는지 확인하기 위해 AWS 리소스를 해당 계정으로 복사할 때는 주의해야 합니다. 새 계정을 프로비저닝하는 데는 시간이 걸리므로, 사고 발생 훨씬 전에 포렌식 계정을 생성하고 계측하여 대응 담당자가 대응에 효과적으로 사용할 수 있도록 준비하는 것이 필수적입니다.

다음 다이어그램은 리전별 포렌식 계정이 있는 포렌식 OU를 포함한 샘플 계정 구조를 보여줍니다.



인시던트 대응을 위한 리전별 계정 구조

백업 및 스냅샷 캡처

주요 시스템과 데이터베이스의 백업을 설정하는 것은 보안 인시던트 복구 및 포렌식 용도로 매우 중요합니다. 백업을 설정하면 시스템을 이전의 안전한 상태로 복원할 수 있습니다. AWS에서는 다양한 리소스의 스냅샷을 생성할 수 있습니다. 스냅샷은 해당 리소스의 특정 시점 백업을 제공합니다. 백업 및 복구를 지원할 수 있는 많은 AWS 서비스가 있습니다. 백업 및 복구를 위한 이러한 서비스와 접근 방식에 대한 자세한 내용은 Backup and Recovery Prescriptive Guidance 및 Use backups to recover from security incidents를 참조하세요.

특히 랜섬웨어와 같은 상황에서는 백업의 보안을 잘 유지하는 것이 중요합니다. 백업 보안을 위한 지침은 Top 10 security best practices for securing backups in AWS를 참조하세요. 백업의 보안을 유지하는 것 외에도 정기적으로 백업 및 복원 프로세스를 테스트하여 보유한 기술과 프로세스가 정상적으로 작동하는지 확인해야 합니다.

포렌식 자동화

보안 이벤트가 발생하는 동안 인시던트 대응팀은 이벤트와 관련된 기간에 정확성을 유지하면서 증거를 신속하게 수집하고 분석할 수 있어야 합니다(예: 특정 이벤트 또는 리소스와 관련된 로그 캡처 또는 Amazon EC2 인스턴스의 메모리 덤프 수집). 특히 많은 인스턴스와 계정에서 관련 증거를 수동으로 수집하는 작업은 인시던트 대응팀에게 어렵고 시간이 많이 소요되는 업무입니다. 또한 수작업으로 수집하는 경우 인적 오류가 발생하기 쉽습니다. 이러한 이유로 포렌식을 위한 자동화를 최대한 개발하고 구현해야 합니다.

AWS는 포렌식을 위한 여러 가지 자동화 리소스를 제공하며 이는 아래 리소스 섹션에 열거되어 있습니다. 다음은 저희가 개발하고 고객이 구현한 포렌식 패턴의 리소스 예제입니다. 시작하기에 유용한 참조

아키텍처가 될 수 있지만, 환경, 요구 사항, 도구, 포렌식 프로세스에 따라 이를 수정하거나 새로운 포렌식 자동화 패턴을 생성하는 것을 고려하세요.

리소스

관련 문서:

- AWS Security Incident Response Guide - Develop Forensics Capabilities
- AWS Security Incident Response Guide - Forensics Resources
- Forensic investigation environment strategies in the AWS 클라우드
- How to automate forensic disk collection in AWS
- AWS Prescriptive Guidance - Automate incident response and forensics

관련 비디오:

- Automating Incident Response and Forensics

관련 예제:

- Automated Incident Response and Forensics Framework
- Automated Forensics Orchestrator for Amazon EC2

SEC10-BP04 보안 인시던트 대응 플레이북 개발 및 테스트

인시던트 대응 프로세스를 준비하는 데 있어 가장 중요한 부분은 플레이북을 개발하는 것입니다. 인시던트 대응 플레이북은 보안 이벤트가 발생했을 때 따라야 할 권장 가이드와 단계를 제공합니다. 명확한 구조와 단계를 갖추면 대응 프로세스가 간소화되고 인적 오류의 가능성이 줄어듭니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

다음과 같은 인시던트 시나리오에 대한 플레이북을 만들어야 합니다.

- 예상되는 인시던트: 예상되는 인시던트에 대한 플레이북을 만들어야 합니다. 여기에는 서비스 거부(DoS), 랜섬웨어, 자격 증명 유출과 같은 위협이 포함됩니다.
- 알려진 보안 조사 결과 또는 알림: 알려진 보안 조사 결과 및 알림(예: Amazon GuardDuty 조사 결과 및 알림)을 해결하기 위한 플레이북을 만들어야 합니다. GuardDuty 조사 결과를 받으면 플레이북은

알림을 잘못 처리하거나 무시하지 않도록 명확한 단계를 제공해야 합니다. 자세한 문제 해결 세부 정보 및 지침은 감지된 GuardDuty 보안 결과 해결을 참조하세요.

플레이북에는 보안 분석가가 잠재적인 보안 사고를 적절히 조사하고 대응하기 위해 완료해야 할 기술 단계가 포함되어야 합니다.

구현 단계

플레이북에 포함할 항목은 다음과 같습니다.

- 플레이북 개요: 이 플레이북은 어떤 위험 또는 인시던트 시나리오를 다루고 있나요? 플레이북의 목표는 무엇인가요?
- 사전 조건: 이 인시던트 시나리오에 어떤 로그, 탐지 메커니즘 및 자동화된 도구가 필요한가요? 예상되는 알림은 무엇인가요?
- 커뮤니케이션 및 에스컬레이션 정보: 누가 관여하며 담당자의 연락처 정보는 무엇인가요? 관련된 각 이해관계자의 책임은 무엇인가요?
- 대응 단계: 인시던트 대응 단계 전반에서 어떤 전술적 단계를 수행해야 하나요? 분석가는 어떤 쿼리를 실행해야 하나요? 원하는 성과를 얻으려면 어떤 코드를 실행해야 하나요?
  - 감지: 어떻게 인시던트를 감지하나요?
  - 분석: 어떻게 영향 범위를 결정하나요?
  - 포함: 범위를 제한하기 위해 어떻게 인시던트를 격리하나요?
  - 근절: 환경에서 위협을 어떻게 제거하나요?
  - 복구: 영향을 받은 시스템이나 리소스를 어떻게 프로덕션 환경으로 복구하나요?
- 예상 결과: 쿼리와 코드가 실행된 후 플레이북의 예상 결과는 무엇인가요?

리소스

관련 Well-Architected 모범 사례:

- SEC10-BP02 인시던트 관리 계획 개발

관련 문서:

- Framework for Incident Response Playbooks
- Develop your own Incident Response Playbooks
- Incident Response Playbook Samples

- Building an AWS incident response runbook using Jupyter playbooks and CloudTrail Lake

SEC10-BP05 액세스 권한 사전 프로비저닝

인시던트 응답자에게 AWS에 사전 프로비저닝된 올바른 액세스 권한이 있는지 확인하여 조사 및 복구 시간을 단축할 수 있도록 합니다.

일반적인 안티 패턴:

- 인시던트 대응을 위해 루트 계정을 사용합니다.
- 기존 계정을 변경합니다.
- 적시 권한 승격을 제공할 때 IAM 권한을 직접 조작합니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 지침

AWS는 가능한 경우 장기 자격 증명에 대한 의존도를 줄이거나 제거할 것을 권장합니다. 그 대신, 임시 자격 증명 및 적시 권한 에스컬레이션 메커니즘을 사용합니다. 장기 자격 증명은 보안 위험에 노출되기 쉽고, 운영 오버헤드가 증가합니다. 대부분의 관리 작업과 인시던트 대응 작업의 경우 관리 액세스를 위한 임시 에스컬레이션과 함께 ID 페더레이션을 구현하는 것이 좋습니다. 이 모델에서는 사용자가 더 높은 수준의 권한(인시던트 대응 역할 등)을 요청하고, 사용자가 권한 승격에 적합한 경우 요청이 승인자에게 전송됩니다. 요청이 승인되면 사용자는 작업을 완료하는 데 사용할 수 있는 임시 AWS 자격 증명 세트를 받습니다. 이러한 자격 증명이 만료되면 사용자는 새로운 승격 요청을 제출해야 합니다.

대부분의 인시던트 대응 시나리오에서는 임시 권한 승격을 사용하는 것이 좋습니다. 이를 위한 올바른 방법은 AWS Security Token Service 및 세션 정책을 사용하여 액세스의 범위를 지정하는 것입니다.

페더레이션형 ID를 사용할 수 없는 시나리오는 다음과 같습니다.

- ID 제공업체(idP)의 침해로 인한 중단.
- 잘못된 구성 또는 인적 오류로 인한 페더레이션 액세스 관리 시스템의 손상.
- 분산 서비스 거부(DDoS) 이벤트 배포 또는 시스템을 사용할 수 없도록 렌더링하는 등의 악의적인 활동.

위의 사례에서는 긴급 break glass 액세스를 구성하여 인시던트에 대한 조사 및 적시 수정이 이루어지도록 해야 합니다. 적절한 권한이 있는 사용자, 그룹 또는 역할을 사용하여 작업을 수행하고 AWS 리소

스에 액세스하는 것이 좋습니다. 루트 사용자 자격 증명이 필요한 작업에서만 루트 사용자를 사용합니다. 인시던트 응답자가 AWS 및 기타 관련 시스템에 대한 올바른 수준의 액세스 권한이 있는지 확인할 수 있도록, 전용 계정을 사전 프로비저닝하는 것이 좋습니다. 계정에는 권한이 있는 액세스가 필요하며, 엄격하게 제어 및 모니터링해야 합니다. 계정은 필요한 작업을 수행하기 위한 가장 최소한의 권한만으로 구축해야 하며, 액세스의 수준은 인시던트 관리 계획의 일부로 생성된 플레이북을 기준으로 해야 합니다.

모범 사례는 목적별 전용 사용자 및 역할을 사용하는 것입니다. IAM 정책 추가를 통해 사용자나 역할 액세스 권한을 임시 승격할 경우 인시던트가 발생하는 동안 사용자의 액세스 대상이 불명확해질 뿐만 아니라 승격된 권한이 취소되지 않는 위험이 발생합니다.

최대한 많은 수의 실패 시나리오에서 액세스를 얻을 수 있는지 확인할 수 있도록 가능한 많은 종속성을 제거하는 것이 중요합니다. 이를 지원하기 위해, 인시던트 대응 담당자가 전용 보안 계정에서 사용자로 생성되었는지 그리고 기존 페더레이션 또는 Single Sign-On(SSO) 솔루션을 통해 관리되고 있지 않은지 확인할 수 있는 플레이북을 생성합니다. 각 개별 대응 담당자는 자신만의 명명된 계정을 가지고 있어야 합니다. 계정 구성은 강력한 암호 정책 및 다중 인증(MFA)를 적용해야 합니다. 인시던트 대응 플레이북에서 AWS Management Console에 대한 액세스 권한만 요구할 경우, 사용자는 구성된 액세스 키를 가지고 있지 않아야 하며 액세스 키 생성이 명시적으로 허용되지 않아야 합니다. 이것은 IAM 정책 또는 서비스 제어 정책(SCP)을 통해 구성할 수 있습니다(AWS 보안 보범 사례의 AWS Organizations SCP 참조). 사용자는 다른 계정에서 인시던트 대응 역할을 수임할 수 있는 기능 외에 다른 권한이 없어야 합니다.

인시던트 과정에서 조사, 개선 조치 또는 복구 활동을 지원하기 위해 기타 내부 또는 외부 인력에게 액세스 권한을 부여해야 할 수 있습니다. 이 경우, 이전에 언급한 플레이북 메커니즘을 사용해야 하며, 인시던트가 완료된 후 모든 추가 액세스 권한이 즉시 취소되었는지 확인할 수 있는 프로세스가 반드시 있어야 합니다.

인시던트 대응 역할의 사용이 적절히 모니터링 및 감사되고 있는지 확인할 수 있도록, 이 목적을 위해 생성된 IAM 사용자 계정이 개인 간에 공유되지 않도록 하고, 특정 작업에 필요하지 않는 한 AWS 계정 루트 사용자가 사용되지 않도록 합니다. 루트 사용자가 필요한 경우(예를 들어, 특정 계정에 대한 IAM 액세스를 사용할 수 없는 경우), 사용 가능한 플레이북을 통해 별도의 프로세스를 사용하여 루트 사용자 자격 증명 및 MFA 토큰의 가용성을 확인해야 합니다.

인시던트 대응 역할에 대한 IAM 정책을 구성하려면 IAM Access Analyzer를 사용하여 AWS CloudTrail 로그를 기반으로 정책을 생성하는 방법을 고려하세요. 이를 위해서는 비프로덕션 계정에서 인시던트 대응 역할에 대한 관리자 액세스 권한을 부여하고 플레이북에 따라 실행해야 합니다. 완료되면 수행된 작업만 허용하는 정책을 생성할 수 있습니다. 그 후 이 정책은 모든 계정의 모든 인시던트 대응 역할에 적용할 수 있습니다. 더욱 쉬운 관리 및 감사를 허용할 수 있도록 각 플레이북에 대한 별도의 IAM 정책

을 생성할 수 있습니다. 플레이북에 포함할 수 있는 예로는 랜섬웨어, 데이터 침해, 프로덕션 액세스의 손실 및 기타 시나리오에 대한 대응 계획이 있을 수 있습니다.

인시던트 대응 계정을 사용하여 다른 AWS 계정에서 전용 인시던트 대응 IAM 역할을 수임합니다. 이러한 역할은 보안 계정의 사용자만 수임할 수 있도록 구성되어야 하며, 신뢰 관계에서는 직접 호출하는 보안 주체가 MFA를 사용하여 인증해야 합니다. 역할은 범위가 좁은 IAM 정책을 사용하여 액세스를 제어해야 합니다. 이러한 역할에 대한 모든 `AssumeRole` 요청은 CloudTrail에 로깅되고 알림이 생성되며, 이러한 역할을 사용하여 수행된 모든 작업이 로깅되도록 합니다.

IAM 계정과 IAM 역할의 이름을 모두 명확하게 지정하여 CloudTrail 로그에서 쉽게 찾을 수 있도록 하는 것이 좋습니다. 그 예로는 IAM 계정은 *<USER_ID>*-BREAK-GLASS, IAM 역할은 BREAK-GLASS-ROLE로 이름을 지정합니다.

CloudTrail은 AWS 계정의 API 활동을 로깅하는 데 사용되며 인시던트 대응 역할의 사용에 대한 알림을 구성하는 데 사용해야 합니다. 루트 키 사용 시 알림 구성에 대한 블로그 게시물을 참조하세요. 인시던트 대응 IAM 역할과 관련된 `AssumeRole` 이벤트를 필터링하기 위해 Amazon CloudWatch 지표 필터를 구성하도록 지침을 수정할 수 있습니다.

```
{ $.eventName = "AssumeRole" && $.requestParameters.roleArn =
 "<INCIDENT_RESPONSE_ROLE_ARN>" && $.userIdentity.invokedBy NOT EXISTS && $.eventType !
= "AwsServiceEvent" }
```

인시던트 대응 역할은 높은 수준의 액세스 권한을 가질 가능성이 높기 때문에, 이러한 알림은 광범위한 그룹에 전달되고 신속하게 조치되는 것이 중요합니다.

인시던트 발생 시 응답자는 IAM에 의해 직접 보호되지 않는 시스템에 대한 액세스가 필요할 수 있습니다. 여기에는 Amazon Elastic Compute Cloud 인스턴스, Amazon Relational Database Service 데이터베이스 또는 서비스형 소프트웨어(SaaS) 플랫폼이 포함될 수 있습니다. SSH 또는 RDP와 같은 기본 프로토콜을 사용하는 대신 Amazon EC2 인스턴스에 대한 모든 관리 액세스에 AWS Systems Manager Session Manager를 사용하는 것이 좋습니다. 이 액세스는 보안 및 감사 기능이 있는 IAM을 사용하여 제어할 수 있습니다. AWS Systems Manager Run Command 문서를 사용하여 플레이북의 일부를 자동화함으로써 사용자 오류를 줄이고 복구 시간을 단축할 수도 있습니다. 데이터베이스 및 서드파티 도구에 대한 액세스를 위해, AWS Secrets Manager에 액세스 자격 증명을 저장하고 인시던트 응답자 역할에 액세스 권한을 부여하는 것이 좋습니다.

마지막으로, 인시던트 대응 IAM 계정의 관리를 입사, 전근 및 퇴사 프로세스에 추가하고 주기적으로 검토 및 테스트하여 의도한 액세스만 허용되는지 확인해야 합니다.

리소스

관련 문서:

- Managing temporary elevated access to your AWS environment
- AWS Security Incident Response Guide
- AWS Elastic Disaster Recovery
- AWS Systems Manager Incident Manager
- IAM 사용자의 계정 암호 정책 설정
- AWS의 다중 인증(MFA) 사용
- Configuring Cross-Account Access with MFA
- Using IAM Access Analyzer to generate IAM policies
- Best Practices for AWS Organizations Service Control Policies in a Multi-Account Environment
- How to Receive Notifications When Your AWS Account's Root Access Keys Are Used
- Create fine-grained session permissions using IAM managed policies
- Break glass access

관련 비디오:

- Automating Incident Response and Forensics in AWS
- DIY guide to runbooks, incident reports, and incident response
- Prepare for and respond to security incidents in your AWS environment

SEC10-BP06 도구 사전 배포

보안 담당자가 조사부터 복구까지 소요되는 시간을 단축할 수 있는 올바른 도구를 미리 배포했는지 확인합니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 지침

보안 대응 및 운영 기능을 자동화하기 위해 AWS의 포괄적인 API 및 도구 세트를 사용할 수 있습니다. 자격 증명 관리, 네트워크 보안, 데이터 보호, 모니터링 기능을 완전히 자동화하고 이미 사용하고 있는 대중적인 소프트웨어 개발 방법을 사용하여 제공할 수 있습니다. 보안 자동화를 구축하면 직원이 보안

상태를 모니터링하면서 수동으로 이벤트에 대응하는 것이 아니라 시스템이 모니터링 및 검토하고 대응을 시작할 수 있습니다.

인시던트 대응팀은 같은 방식으로 계속 알림에 대응할 경우 알림에 대한 피로감을 느낄 위험이 있습니다. 시간이 지남에 따라 팀이 알림에 무감각한 상태가 되어 일상적인 상황을 처리하는 데 실수하거나 비정상적인 알림을 놓칠 수 있습니다. 자동화는 반복적이고 일상적인 알림을 처리하는 기능을 사용함으로써 알림에 대한 피로감을 방지하며, 중요하고 특별한 인시던트만 사람이 직접 처리하도록 합니다. Amazon GuardDuty, AWS CloudTrail Insights, Amazon CloudWatch Anomaly Detection과 같은 이상 탐지 시스템을 통합하면 일반적인 임곗값 기반 알림의 부담을 줄일 수 있습니다.

프로세스의 단계를 프로그래밍 방식으로 자동화하여 수동 프로세스를 개선할 수 있습니다. 이벤트에 대한 수정 패턴을 정의한 후 해당 패턴을 실행 가능한 로직으로 분해하고 코드를 작성하여 해당 로직을 수행할 수 있습니다. 그런 다음, 응답자가 해당 코드를 실행하여 문제를 해결할 수 있습니다. 시간이 지남에 따라 점점 더 많은 단계를 자동화할 수 있으며, 궁극적으로 일반적인 인시던트의 전체 클래스를 자동으로 처리할 수 있습니다.

보안 조사 중에 관련 로그를 검토하여 인시던트의 전체 범위와 타임라인을 기록하고 이해할 수 있어야 합니다. 관심 있는 특정 작업이 발생했음을 나타내는 알림 생성에도 로그가 필요합니다. 쿼리 및 검색 메커니즘을 선택, 활성화, 저장 및 설정하고 경보를 설정하는 것이 중요합니다. 또한 로그 데이터를 검색할 수 있는 도구를 제공하는 효과적인 방법은 Amazon Detective입니다.

AWS는 200개 이상의 클라우드 서비스와 수천 개의 기능을 제공합니다. 인시던트 대응 전략을 지원하고 간소화할 수 있는 서비스를 검토하는 것이 좋습니다.

로깅 외에도 태그 지정 전략을 개발하고 구현해야 합니다. 태그 지정은 AWS 리소스의 목적에 대한 컨텍스트를 제공하는 데 도움이 될 수 있습니다. 태그 지정은 자동화를 위해서도 사용할 수 있습니다.

구현 단계

분석 및 알림을 위한 로그 선택 및 설정

인시던트 대응을 위한 로깅 구성에 대한 다음 설명서를 참조하세요.

- 보안 인시던트 분석을 위한 로깅 전략
- SEC04-BP01 서비스 및 애플리케이션 로깅 구성

보안 서비스를 사용하여 탐지 및 대응 지원

AWS는 기본 탐지, 예방 및 대응 기능을 제공하며, 기타 서비스를 사용하여 맞춤형 보안 솔루션을 설계할 수 있습니다. 보안 인시던트 대응과 가장 관련성이 높은 서비스 목록은 클라우드 기능 정의를 참조하세요.

태그 지정 전략 개발 및 구현

AWS 리소스를 사용하는 비즈니스 사용 사례와 관련 내부 이해관계자에 대한 컨텍스트 정보를 얻는 것은 어려울 수 있습니다. 한 가지 방법은 AWS 리소스에 메타데이터를 할당하고 사용자 정의 키와 값으로 구성되는 태그의 형태를 사용하는 것입니다. 태그를 생성하여 리소스를 목적, 소유자, 환경, 처리되는 데이터 유형 및 기타 원하는 기준에 따라 분류할 수 있습니다.

일관된 태그 전략을 사용하면 AWS 리소스에 대한 컨텍스트 정보를 신속하게 식별할 수 있으므로 응답 시간을 단축하고 조직 컨텍스트에 소요되는 시간을 최소화할 수 있습니다. 태그는 응답 자동화를 시작하는 메커니즘으로도 사용할 수 있습니다. 태그 지정 대상에 대한 자세한 내용은 Tagging your AWS resources를 참조하세요. 먼저 조직 전체에 구현할 태그를 정의하는 것이 좋습니다. 그런 다음 이 태그 지정 전략을 구현하고 적용합니다. 구현 및 시행에 대한 자세한 내용은 Implement AWS resource tagging strategy using AWS Tag Policies and Service Control Policies (SCPs)를 참조하세요.

리소스

관련 Well-Architected 모범 사례:

- SEC04-BP01 서비스 및 애플리케이션 로깅 구성
- SEC04-BP02 표준화된 위치에서 로그, 조사 결과 및 지표 캡처

관련 문서:

- 보안 인시던트 분석을 위한 로깅 전략
- 인시던트 대응 클라우드 기능 정의

관련 예제:

- Threat Detection and Response with Amazon GuardDuty and Amazon Detective
- Security Hub 워크숍
- Vulnerability Management with Amazon Inspector

SEC10-BP07 시뮬레이션 실행

시간이 지나면서 조직이 성장하고 발전함에 따라 위협 환경도 변화하므로 인시던트 대응 능력을 지속적으로 검토하는 것이 중요합니다. 시뮬레이션(게임 데이라고도 함)을 실행하는 것도 이 평가를 수행하는 데 사용할 수 있는 방법 중 하나입니다. 시뮬레이션은 위협 행위자의 전술, 기술 및 절차(TTP)를 모방하도록 설계된 실제 보안 이벤트 시나리오를 사용하며, 이를 통해 조직은 이러한 모의 사이버 이벤트에 실제 상황과 같이 대응하여 인시던트 대응 능력을 발휘하고 평가할 수 있습니다.

이 모범 사례 확립의 이점: 시뮬레이션은 다음과 같은 다양한 이점을 제공합니다.

- 사이버 대비 상태를 검증하고 인시던트 대응자의 자신감을 높입니다.
- 도구 및 워크플로의 정확성과 효율성을 테스트합니다.
- 인시던트 대응 계획에 맞춰 커뮤니케이션 및 에스컬레이션 방법을 개선합니다.
- 덜 일반적인 벡터에 대응할 수 있는 기회를 제공합니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

시뮬레이션에는 다음과 같은 세 가지 주요 유형이 있습니다.

- 탁상 연습: 시뮬레이션에 대한 탁상 접근 방식은 다양한 인시던트 대응 이해관계자가 참여하여 책임진 역할을 연습하고 확립된 커뮤니케이션 도구과 플레이북을 사용하는 토론 기반 세션입니다. 연습은 일반적으로 가상 장소, 실제 장소 또는 이들 장소의 조합에서 하루 종일 수행할 수 있어 언제든 촉진시킬 수 있습니다. 토론을 기반으로 하기 때문에 탁상 연습은 프로세스, 사람, 협업에 중점을 둡니다. 기술은 토론의 핵심 부분이지만 인시던트 대응 도구 또는 스크립트의 실제 사용은 일반적으로 탁상 연습의 일부가 아닙니다.
- 퍼플 팀 연습: 퍼플 팀 연습은 인시던트 대응 담당자(블루 팀)와 시뮬레이션된 위협 행위자(레드 팀) 간의 협업 수준을 높입니다. 블루 팀은 보안 운영 센터(SOC)의 직원으로 구성되지만 실제 사이버 이벤트 중에 관여하게 될 다른 이해관계자들도 포함될 수 있습니다. 레드 팀은 보안 공격 교육을 받은 침투 테스트 팀 또는 주요 이해관계자로 구성됩니다. 레드 팀은 시나리오를 설계할 때 연습 진행자와 협력하여 시나리오가 정확하고 실현 가능한지 확인합니다. 퍼플 팀 연습에서는 인시던트 대응 작업을 지원하는 탐지 메커니즘, 도구 및 표준 운영 절차(SOP)에 주로 초점을 맞춥니다.
- 레드 팀 연습: 레드 팀 연습 중에 공격 팀(레드 팀)은 미리 정해진 범위에서 특정 목표 또는 일련의 목표를 달성하기 위해 시뮬레이션을 수행합니다. 방어 팀(블루 팀)은 훈련의 범위와 기간을 꼭 알 필요가 없습니다. 이를 모르면 실제 인시던트에 어떻게 대응하는지에 대한 더 현실적인 평가를 받을 수

있습니다. 레드 팀 연습은 침습적 테스트일 수 있으므로 주의가 필요하고 해당 연습이 환경에 실제로 해를 끼치지 않는지 확인하기 위한 관리 조치를 취해야 합니다.

정기적으로 사이버 시뮬레이션을 진행하는 것이 좋습니다. 각 연습 유형에는 참가자와 조직 전체에 대한 고유한 이점이 있으므로 덜 복잡한 시뮬레이션 유형(예: 탁상 연습)에서 시작하여 더 복잡한 시뮬레이션 유형(레드 팀 연습)으로 진행할 수 있습니다. 보안 성숙도, 리소스, 원하는 성과에 따라 시뮬레이션 유형을 선택해야 합니다. 일부 고객은 복잡성과 비용 때문에 레드 팀 연습을 선택하지 않을 수 있습니다.

구현 단계

선택한 유형에 관계없이 시뮬레이션은 일반적으로 다음 구현 단계를 따릅니다.

1. 핵심 연습 요소 정의: 시뮬레이션의 시나리오와 목표를 정의합니다. 이 두 가지 모두 리더의 승인을 받아야 합니다.
2. 주요 이해관계자 식별: 연습에는 최소한 연습 진행자와 참가자가 필요합니다. 시나리오에 따라 법무, 커뮤니케이션 또는 경영진과 같은 추가 이해관계자가 참여할 수 있습니다.
3. 시니라오 구축 및 테스트: 특정 요소가 실현 가능하지 않은 경우 구축 중인 시나리오를 재정의해야 할 수 있습니다. 이 단계의 결과로 최종 시나리오가 도출될 것으로 예상됩니다.
4. 시뮬레이션 촉진: 시뮬레이션 유형에 따라 어떤 방법으로 촉진시킬지 결정됩니다(종이를 사용한 시나리오 또는 고도로 기술적인 시뮬레이션 시나리오). 진행자는 연습 목표에 맞게 촉진 전략을 조정해야 하며 가능한 한 모든 연습 참가자를 참여시켜 최대한의 이점을 확보해야 합니다.
5. 사후 조치 보고서(AAR) 개발: 잘 운영된 영역, 개선이 필요한 영역, 잠재적인 격차를 식별합니다. AAR은 시뮬레이션의 효과와 시뮬레이션된 이벤트에 대한 팀의 반응을 측정하여 향후 시뮬레이션을 통해 시간의 흐름에 따른 진행 상황을 추적할 수 있도록 해야 합니다.

리소스

관련 문서:

- AWS 인시던트 대응 가이드

관련 비디오:

- AWS GameDay - Security Edition
- Running effective security incident response simulations

운영

운영은 인시던트 대응 수행의 핵심입니다. 여기서 보안 인시던트 대응 및 해결 조치가 이루어집니다. 운영에는 탐지, 분석, 격리, 근절, 복구와 같은 다섯 가지 단계가 포함됩니다. 이러한 단계 및 목표에 대한 설명은 다음 표에 나와 있습니다.

Phase(단계)	목표
탐지	잠재적 보안 이벤트를 파악합니다.
분석	보안 이벤트가 인시던트인지 판단하고 인시던트 범위를 평가하세요.
격리	보안 이벤트의 범위를 최소화하고 제한합니다.
근절	보안 이벤트와 관련된 승인되지 않은 리소스 또는 아티팩트를 제거합니다. 보안 인시던트를 일으킨 완화 조치를 구현합니다.
복구	시스템을 알려진 안전 상태로 복원하고 이러한 시스템을 모니터링하여 위협이 다시 발생하지 않는지 확인합니다.

이 단계는 효과적이고 강력한 방식으로 대응하기 위해 보안 인시던트에 대응하고 이를 운영할 때 지침으로 활용해야 합니다. 실제로 취하는 조치는 인시던트에 따라 달라집니다. 예를 들어 랜섬웨어와 관련된 인시던트는 퍼블릭 Amazon S3 버킷과 관련된 인시던트와는 다른 대응 단계를 따라야 합니다. 또한 이러한 단계가 반드시 순차적으로 발생하는 것은 아닙니다. 격리 및 근절 후에는 분석 작업으로 돌아가 자신의 행동이 효과적이었는지 파악해야 할 수도 있습니다.

직원, 프로세스 및 기술 전반의 철저한 준비가 효과적인 운영을 위한 핵심입니다. 따라서 준비 섹션의 모범 사례를 따르면 활성 보안 이벤트에 효과적으로 대응할 수 있습니다.

자세한 내용은 AWS Security Incident Response Guide의 Operations 섹션을 참조하세요.

인시던트 사후 활동

위협 환경은 끊임없이 변화하므로 환경을 효과적으로 보호할 수 있는 조직의 역량도 그에 못지않게 역동적으로 대처하는 것이 중요합니다. 지속적인 개선의 핵심은 발생 가능한 보안 인시던트를 효과적으

로 탐지, 대응 및 조사할 수 있는 능력을 향상시키고, 발생 가능한 취약성을 줄이며, 대응 시간을 단축하고, 안전한 운영으로 돌아갈 수 있도록 인시던트 및 시뮬레이션의 결과를 반복해서 검토하는 것입니다. 다음 메커니즘은 조직이 상황에 관계없이 효과적으로 대응할 수 있는 최신 역량과 지식을 갖추고 있는지 확인하는 데 도움이 될 수 있습니다.

모범 사례

- SEC10-BP08 인시던트로부터 학습하기 위한 프레임워크 구축

SEC10-BP08 인시던트로부터 학습하기 위한 프레임워크 구축

학습한 교훈 프레임워크와 근본 원인 분석 기능을 구현하면 인시던트 대응 능력을 개선하는 데 도움이 될 뿐만 아니라 인시던트 재발을 방지하는 데도 도움이 됩니다. 각 인시던트에서 교훈을 얻음으로써 동일한 실수, 노출 또는 잘못된 구성을 반복하지 않도록 하여 보안 태세를 개선할 뿐만 아니라 사전에 방지 가능한 상황으로 인한 시간 손실을 최소화할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 가이드

다음 사항을 높은 수준에서 설정하고 달성하는 학습한 교훈 프레임워크를 구현하는 것이 중요합니다.

- 학습한 교훈은 언제 적용하게 되나요?
- 학습한 교훈 과정에는 무엇이 포함되나요?
- 학습한 교훈은 어떻게 수행되나요?
- 누가 어떻게 이 과정에 참여하나요?
- 개선이 필요한 부분은 어떻게 확인할 수 있나요?
- 개선 사항을 효과적으로 추적하고 구현할 수 있도록 어떻게 해야 할까요?

프레임워크는 개인에게 초점을 맞추거나 개인을 비난하는 것이 아니라 도구와 프로세스를 개선하는 데 초점을 맞춰야 합니다.

구현 단계

앞에서 설명한 개략적인 결과 외에도, 프로세스에서 최대한의 가치(실행 가능한 개선으로 이어지는 정보)를 이끌어낼 수 있도록 올바른 질문을 하는 것이 중요합니다. 다음 질문을 고려하면 학습한 교훈 토론을 시작하는 데 도움이 됩니다.

- 어떤 인시던트였나요?

- 인시던트가 언제 처음 확인되었나요?
- 어떻게 식별되었나요?
- 어떤 시스템에서 해당 활동에 대해 경고했나요?
- 어떤 시스템, 서비스 및 데이터가 관련되어 있나요?
- 구체적으로 어떤 일이 발생했나요?
- 어떤 점이 잘 작동했나요?
- 어떤 점이 잘 작동하지 않았나요?
- 인시던트에 대응하기 위해 어떤 프로세스 또는 절차가 실패했거나 조정되지 못했나요?
- 다음 영역에서 개선할 수 있는 사항:
  - 사람
    - 연락이 필요한 직원이 실제로 연락이 가능했고 연락처 목록이 최신 상태였나요?
    - 인시던트에 효과적으로 대응하고 조사하는 데 필요한 교육이나 역량을 갖춘 직원이 없었나요?
    - 적절한 리소스가 준비되어 있고 이용 가능했나요?
  - 프로세스
    - 프로세스와 절차를 준수했나요?
    - 이 (유형의) 인시던트에 대한 프로세스와 절차가 문서화되어 있고 사용 가능했나요?
    - 필요한 프로세스 및 절차가 누락되지는 않았나요?
    - 대응 담당자가 문제를 대응하는 데 필요한 정보에 적시에 액세스할 수 있었나요?
  - 기술
    - 기존 경고 시스템이 활동을 효과적으로 식별하고 경고했나요?
    - 어떻게 하면 탐지 시간을 50%까지 줄일 수 있을까요?
    - 기존 경고 시스템을 개선해야 하나요? 아니면 이 (인시던트 유형) 사고에 대해 새로운 경고 시스템을 구축해야 하나요?
    - 기존 도구로 인시던트를 효과적으로 조사(검색 및 분석)할 수 있었나요?
    - 이 (유형의) 인시던트를 더 빨리 식별하려면 어떻게 해야 할까요?
    - 이 (유형의) 인시던트가 재발하는 것을 방지하려면 어떻게 해야 할까요?
    - 개선 계획의 담당자는 누구이며 개선 계획이 실행되었는지 어떻게 테스트할 예정인가요?
    - 추가 모니터링 또는 예방적 통제 및 프로세스를 구현하고 테스트할 일정은 어떻게 되나요?

이 목록은 모든 것을 포함하지는 않지만, 조직 및 비즈니스 요구 사항이 무엇인지 식별하고 인시던트로부터 가장 효과적으로 학습하고 보안 태세를 지속적으로 개선하기 위해 이를 분석할 수 있는 방법을 식

별하기 위한 출발점이 될 수 있습니다. 가장 중요한 것은 인시던트 대응 프로세스, 문서화 및 이해관계자 전반의 기대치에서 학습한 교훈을 표준으로 삼아 통합하는 것부터 시작하는 것입니다.

리소스

관련 문서:

- AWS Security Incident Response Guide - Establish a framework for learning from incidents
- NCSC CAF guidance - Lessons learned

애플리케이션 보안

애플리케이션 보안(AppSec)은 개발하는 워크로드의 보안 속성을 설계, 구축 및 테스트하는 전반적인 프로세스를 설명합니다. 조직에 적절한 교육을 받은 직원이 있어야 하며, 빌드 및 릴리스 인프라의 보안 속성을 이해하고, 자동화를 사용하여 보안 문제를 식별해야 합니다.

소프트웨어 개발 수명 주기(SDLC) 및 릴리스 후 프로세스에 정기적으로 애플리케이션 보안 테스트를 수행하면 프로덕션 환경에 유입되는 애플리케이션 보안 문제를 식별, 수정 및 방지할 수 있는 구조화된 메커니즘을 갖출 수 있습니다.

애플리케이션 개발 방법에는 워크로드를 설계, 구축, 배포 및 운영할 때의 보안 제어 기능이 포함되어야 합니다. 이와 동시에 지속적으로 결함을 줄이고 기술 부채를 최소화하도록 프로세스를 조정하세요. 예를 들어 설계 단계에서 위협 모델링을 사용하면 설계 결함을 조기에 발견할 수 있으므로 기다렸다가 나중에 문제를 완화하는 것보다 수정이 더 쉽고 비용이 적게 듭니다.

SDLC에서 결함은 보통 일찍 해결해야 비용과 복잡성이 줄어듭니다. 문제를 해결하는 가장 쉬운 방법은 애초에 문제가 발생하지 않게 하는 것입니다. 따라서 위협 모델로 시작하면 설계 단계부터 올바른 결과에 집중하는 데 도움이 됩니다. AppSec 프로그램이 발전을 거듭하면서 자동화를 사용하여 수행되는 테스트의 양을 늘리고, 빌더에 대한 피드백의 충실도를 개선하며, 보안 검토에 필요한 시간을 줄일 수 있습니다. 이러한 모든 작업은 구축하는 소프트웨어의 품질을 개선하고, 프로덕션에 기능을 도입하는 속도를 높입니다.

이 구현 지침은 조직 및 문화, 파이프라인 자체 보안, 파이프라인 내부 보안, 종속성 관리라는 네 가지 영역에 중점을 둡니다. 각 영역은 구현할 수 있는 일련의 원칙을 제공하며, 워크로드를 설계, 개발, 구축, 배포 및 운영하는 방법을 아우르는 전체적인 관점을 제공합니다.

AWS에는 애플리케이션 보안 프로그램을 다룰 때 사용할 수 있는 여러 방법이 있습니다. 이러한 접근 방식 중 일부는 기술에 의존하며, 일부는 애플리케이션 보안 프로그램의 인력 및 조직 측면에 중점을 두고 있습니다.

모범 사례

- SEC11-BP01 애플리케이션 보안 교육
- SEC11-BP02 개발 및 릴리스 수명 주기를 통한 테스트 자동화
- SEC11-BP03 정기적인 침투 테스트 시행
- SEC11-BP04 코드 검토 수행
- SEC11-BP05 패키지 및 종속성 서비스의 중앙 집중화

- SEC11-BP06 프로그래밍 방식으로 소프트웨어 배포
- SEC11-BP07 정기적으로 파이프라인의 보안 속성 평가
- SEC11-BP08 보안 소유권이 워크로드 팀에 귀속되도록 프로그램 구축

SEC11-BP01 애플리케이션 보안 교육

팀에 안전한 개발 및 운영 방식에 대한 교육을 제공하세요. 안전한 고품질 소프트웨어를 구축하는 데 도움이 됩니다. 이 방식을 통해 팀은 개발 수명 주기 초기에 보안 문제를 예방, 탐지 및 해결할 수 있습니다. 위협 모델링, 안전한 코딩 방식, 안전한 구성 및 운영을 위한 서비스 사용을 다루는 교육을 고려해 보세요. 셀프 서비스 리소스를 통해 팀이 교육에 액세스할 수 있도록 하고 지속적인 개선을 위해 정기적으로 피드백을 수집하세요.

원하는 성과: 처음부터 보안을 염두에 두고 소프트웨어를 설계하고 구축하는 데 필요한 지식과 기술을 팀에 제공합니다. 위협 모델링 및 안전한 개발 방식에 대한 교육을 통해 팀은 잠재적 보안 위험과 소프트웨어 개발 수명 주기(SDLC) 동안 이를 완화하는 방법을 깊이 이해할 수 있습니다. 이러한 선제적 보안 접근 방식은 팀 문화의 일부이며, 잠재적인 보안 문제를 조기에 식별하고 해결할 수 있습니다. 따라서 팀은 고품질의 안전한 소프트웨어와 기능을 더 효율적으로 제공하여 전체 제공 일정을 단축합니다. 조직 내에 보안 소유권이 모든 빌더에게 공유되는 협업적이고 포괄적인 보안 문화가 있습니다.

일반적인 안티 패턴:

- 보안 검토가 끝날 때까지 기다린 다음 시스템의 보안 속성을 고려합니다.
- 중앙의 보안 팀에 모든 보안 결정을 맡깁니다.
- SDLC에서 내린 결정이 조직의 전반적인 보안 기대치 또는 정책과 어떤 관련이 있는지에 대해 소통하지 않습니다.
- 보안 검토 프로세스를 너무 늦게 수행합니다.

이 모범 사례 확립의 이점:

- 개발 주기 초기에 조직의 보안 요구 사항을 보다 효과적으로 이해합니다.
- 잠재적인 보안 문제를 보다 신속하게 식별하고 해결하여 기능을 발 빠르게 제공할 수 있습니다.
- 소프트웨어 및 시스템의 품질이 향상됩니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 지침

안전한 고품질 소프트웨어를 구축하기 위해 안전한 애플리케이션 개발 및 운영을 위한 일반적인 방식에 대해 팀에 교육을 제공합니다. 이 방식은 팀이 개발 수명 주기 초기에 보안 문제를 예방, 탐지 및 해결하는 데 도움이 될 수 있으며, 결과적으로 제공 일정이 단축될 수 있습니다.

이 방식이 자리 잡히도록 위협 모델링 워크숍과 같은 AWS 리소스를 사용하여 위협 모델링에 대해 팀을 교육하는 것을 고려합니다. 위협 모델링은 팀이 잠재적인 보안 위험을 이해하고 처음부터 보안을 염두에 두고 시스템을 설계하는 데 도움이 될 수 있습니다. 또한 안전한 개발 방식에 대한 AWS 교육 and Certification, 산업 또는 AWS 파트너 교육에 대한 액세스를 제공할 수 있습니다. 대규모 설계, 개발, 보안 및 효율적인 운영에 대한 포괄적인 접근 방식에 대한 자세한 내용은 AWS DevOps Guidance를 참조하세요.

조직의 보안 검토 프로세스를 명확하게 정의하고 소통하며 팀, 보안 팀 및 기타 이해관계자의 책임을 개괄적으로 설명합니다. 보안 요구 사항을 충족하는 방법을 다루는 자습형 지침, 코드 예시, 템플릿을 게시합니다. AWS CloudFormation, AWS 클라우드 개발 키트 (AWS CDK)(AWS CDK) Constructs 및 Service Catalog와 같은 AWS 서비스를 사용하여 사전 승인된 보안 구성을 제공하고 사용자 지정 설정의 필요성을 줄일 수 있습니다.

팀의 보안 검토 프로세스 및 교육 경험에 대해 정기적으로 피드백을 수집하고, 피드백을 바탕으로 지속적으로 개선합니다. 게임 데이 또는 버그 퇴치 캠페인을 수행하여 보안 문제를 식별하고 해결하는 동시에 팀의 스킬을 향상시킵니다.

구현 단계

1. 교육 요구 사항 식별: 설문조사, 코드 검토 또는 팀원과의 논의를 통해 안전한 개발 방식과 관련하여 팀 내 현재 스킬 수준 및 지식 격차를 평가합니다.
2. 교육 계획: 식별된 요구 사항에 따라 위협 모델링, 안전한 코딩 방식, 보안 테스트 및 안전한 배포 방식과 같은 관련 주제를 다루는 교육 계획을 수립합니다. 위협 모델링 워크숍, AWS 교육 and Certification, 산업 또는 AWS 파트너 교육 프로그램과 같은 리소스를 사용합니다.
3. 교육 일정 수립 및 제공: 팀을 위한 정기적인 교육 세션 또는 워크숍 일정을 잡습니다. 팀의 선호도와 시간적 여유에 따라 강사 주도형 또는 자기 주도형일 수 있습니다. 학습을 강화하기 위해 실습을 장려하고 실제 사례를 활용하도록 합니다.
4. 보안 검토 프로세스 정의: 보안 팀 및 기타 이해관계자와 협력하여 애플리케이션의 보안 검토 프로세스를 명확하게 정의합니다. 개발 팀, 보안 팀 및 기타 관련 이해관계자를 포함하여 프로세스에 관련된 각 팀 또는 개인의 책임을 문서화합니다.
5. 셀프 서비스 리소스 생성: 조직의 보안 요구 사항을 충족하는 방법을 보여주는 셀프 서비스 지침, 코드 예시 및 템플릿을 개발합니다. 사전 승인된 보안 구성을 제공하고 사용자 지정 설정의 필요성을

줄이기 위해 CloudFormation AWS CDK Constructs 및 Service Catalog와 같은 AWS 서비스를 고려합니다.

6. 소통 및 교류: 보안 검토 프로세스와 사용 가능한 셀프 서비스 리소스를 팀에 효과적으로 알립니다. 교육 세션 또는 워크숍을 수행하여 팀이 이러한 리소스를 숙지하도록 하고 리소스 사용 방법을 이해하고 있는지 확인합니다.
7. 피드백 수집 및 개선: 팀의 보안 검토 프로세스 및 교육 경험에 대해 정기적으로 피드백을 수집합니다. 이 피드백을 사용하여 개선이 필요한 영역을 식별하고 교육 자료, 셀프 서비스 리소스 및 보안 검토 프로세스를 지속적으로 개선합니다.
8. 보안 연습 수행: 게임 데이 또는 버그 퇴치 캠페인을 주최하여 애플리케이션 내의 보안 문제를 식별하고 해결합니다. 이러한 연습은 잠재적 취약성을 발견하는 데 도움이 될 뿐만 아니라, 팀이 보안 개발 및 운영에 대한 스킬을 강화하는 실용적인 학습 기회 역할을 합니다.
9. 지속적인 학습 및 개선: 팀이 최신 보안 개발 방식, 도구 및 기술을 파악하도록 장려합니다. 진화하는 보안 환경과 모범 사례를 반영하도록 교육 자료와 리소스를 정기적으로 검토하고 업데이트합니다.

리소스

관련 모범 사례:

- SEC11-BP08 보안 소유권이 워크로드 팀에 귀속되도록 프로그램 구축

관련 문서:

- AWS 교육 및 인증
- How to think about cloud security governance
- How to approach threat modeling
- Accelerating training – The AWS Skills Guild
- AWS DevOps Sagas

관련 비디오:

- Proactive security: Considerations and approaches

관련 예제:

- Workshop on threat modeling

- Industry awareness for developers

관련 서비스:

- AWS CloudFormation
- AWS 클라우드 개발 키트 (AWS CDK)(AWS CDK) Constructs
- 서비스 카탈로그

SEC11-BP02 개발 및 릴리스 수명 주기를 통한 테스트 자동화

개발 및 릴리스 수명 주기 전반에 걸쳐 보안 속성 테스트를 자동화하세요. 자동화를 구현하면 릴리스에 앞서 소프트웨어의 잠재적인 문제를 일관되고 반복적으로 손쉽게 식별할 수 있어 소프트웨어 제공 중에 보안 문제가 발생할 위험이 줄어듭니다.

원하는 성과: 자동화된 테스트의 목표는 개발 수명 주기 전반에 걸쳐 잠재적인 문제를 조기에 자주 탐지할 수 있는 프로그래밍 방식을 제공하는 것입니다. 회귀 테스트를 자동화하면 기능 테스트 및 비기능 테스트를 다시 실행하여 이전에 테스트한 소프트웨어가 변경 후에도 예상대로 작동하는지 확인할 수 있습니다. 보안 디바이스 테스트를 정의하여 인증 정보 손상 또는 누락과 같은 일반적인 구성 오류를 확인하면 이러한 문제를 개발 프로세스 초기에 식별하고 해결할 수 있게 됩니다.

테스트 자동화는 애플리케이션의 요구 사항과 원하는 기능을 기반으로 애플리케이션 검증을 위해 특별히 제작된 테스트 사례를 사용합니다. 자동화된 테스트 결과는 생성된 테스트 출력을 각각의 예상 출력과 비교하여 전체 테스트 수명 주기를 가속화합니다. 회귀 테스트 및 디바이스 테스트 세트와 같은 테스트 방법론이 자동화에 가장 적합합니다. 보안 속성 테스트를 자동화하면 빌더가 보안 검토를 기다리지 않고도 자동으로 피드백을 받을 수 있습니다. 정적 또는 동적 코드 분석의 형태로 자동화된 테스트는 코드 품질을 개선하고 개발 수명 주기 초기에 잠재적인 소프트웨어 문제를 탐지하는 데 도움이 됩니다.

일반적인 안티 패턴:

- 자동화된 테스트의 테스트 사례 및 테스트 결과를 전달하지 않습니다.
- 릴리스 직전에만 자동화된 테스트를 수행합니다.
- 요구 사항이 자주 변경되는 테스트 사례를 자동화합니다.
- 보안 테스트 결과를 처리하는 방법에 대한 지침을 제공하지 못합니다.

이 모범 사례 확립의 이점:

- 시스템의 보안 속성을 평가하는 사용자에 대한 의존도를 낮춥니다.
- 여러 작업 흐름에서 일정한 결과를 도출하여 일관성이 향상됩니다.
- 프로덕션 소프트웨어에 보안 문제가 발생할 가능성이 줄어듭니다.
- 소프트웨어 문제를 조기에 발견하여 탐지부터 해결에 걸리는 시간이 단축됩니다.
- 여러 작업 흐름에 걸쳐 체계적이거나 반복적인 동작에 대한 가시성이 향상되어 조직 전체의 개선을 추진하는 데 유용합니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 지침

소프트웨어를 구축할 때 다양한 소프트웨어 테스트 메커니즘을 채택하여 애플리케이션의 비즈니스 논리에 바탕을 둔 기능적 요구 사항과 애플리케이션 신뢰성, 성능, 보안에 중점을 둔 비기능적 요구 사항을 기반으로 애플리케이션을 테스트합니다.

정적 애플리케이션 보안 테스트(SAST)는 비정상적인 보안 패턴에 대한 소스 코드를 분석하고 결함이 발생하기 쉬운 코드를 표시해 줍니다. SAST는 문서(요구 사항 사양, 설계 설명서, 설계 사양)와 같은 정적 입력 및 애플리케이션 소스 코드를 전적으로 사용하여 알려진 여러 보안 문제를 테스트합니다. 정적 코드 분석기는 대량의 코드를 신속하게 분석하는 데 도움이 됩니다. NIST Quality Group에서는 소스 코드 보안 분석기에 대한 비교 정보를 제공합니다. 여기에는 Byte Code Scanners 및 Binary Code Scanners에 대한 오픈 소스 도구도 포함됩니다.

실행 중인 애플리케이션을 테스트하여 잠재적으로 예상치 못한 동작을 식별하는 동적 분석 보안 테스트(DAST) 방법론으로 정적 테스트를 보완합니다. 동적 테스트를 사용하면 정적 분석을 통해 탐지할 수 없는 잠재적 문제를 감지할 수 있습니다. 코드 리포지토리, 구축 및 파이프라인 단계에서 테스트하면 코드 입력 시 발생 가능한 여러 유형의 잠재적 문제를 확인할 수 있습니다. Amazon Q Developers는 빌더의 IDE에서 보안 스캔을 포함한 코드 권장 사항을 제공합니다. Amazon CodeGuru Security는 애플리케이션 개발 중에 중요한 문제, 보안 문제 및 발견하기 어려운 버그를 식별할 수 있으며 코드 품질을 개선하기 위한 권장 사항을 제공합니다. 또한 소프트웨어 재료 사양서(SBOM)를 추출하면 소프트웨어 구축에 사용되는 다양한 구성 요소의 세부 정보 및 관계가 포함된 공식 레코드를 추출할 수 있습니다. 이것을 취약성 관리에 반영하고 소프트웨어 또는 구성 요소 종속성과 공급망 위험을 빠르게 식별하는 데 사용할 수 있습니다.

Security for Developers 워크숍에서는 SAST 및 DAST 테스트 방법론이 포함된 릴리스 파이프라인 자동화를 위해 AWS CodeBuild, AWS CodeCommit, AWS CodePipeline과 같은 AWS 개발자 도구를 사용합니다.

SDLC를 진행하면서 보안 팀과 함께 정기적인 애플리케이션 검토를 포함하는 반복 프로세스를 수립하세요. 이러한 보안 검토에서 수집된 피드백은 릴리스 준비 상태 검토 과정에서 해결하고 검증해야 합니다. 이러한 검토를 통해 강력한 애플리케이션 보안 태세를 확립하고, 빌더에게 잠재적인 문제를 해결하는 데 도움이 되는 실용적인 피드백을 제공할 수 있습니다.

구현 단계

- 보안 테스트가 포함된 IDE, 코드 검토 및 CI/CD 도구를 일관성 있게 구현합니다.
- 문제를 해결해야 한다고 빌더에게 통보하는 대신 SDLC의 어느 지점에서 파이프라인을 차단하는 것이 적절한지 고려해 보세요.
- Automated Security Helper(ASH)는 오픈 소스 코드 보안 스캔 도구의 예입니다.
- 개발자 IDE와 통합된 Amazon Q Developer, 커밋 시 코드 스캔을 위한 Amazon CodeGuru Security와 같은 자동화된 도구를 사용하여 테스트 또는 코드 분석을 수행하면 빌더가 적시에 피드백을 받을 수 있습니다.
- AWS Lambda를 사용하여 구축할 때 Amazon Inspector를 사용하여 함수의 애플리케이션 코드를 스캔할 수 있습니다.
- CI/CD 파이프라인에 자동화된 테스트가 포함된 경우 티켓팅 시스템을 사용하여 소프트웨어 문제의 알림 및 해결 방법을 추적해야 합니다.
- 결과를 생성할 수 있는 보안 테스트의 경우 수정 지침에 연결하면 빌더가 코드 품질을 개선하는 데 도움이 됩니다.
- 자동화된 도구의 결과를 정기적으로 분석하여 다음 자동화, 빌더 교육 또는 인식 캠페인의 우선순위를 지정합니다.
- CI/CD 파이프라인의 일부로 SBOM을 추출하려면 Amazon Inspector SBOM 생성기를 사용하여 아카이브, 컨테이너 이미지, 디렉터리, 로컬 시스템 및 컴파일된 Go 및 Rust 바이너리에 대한 SBOM을 CycloneDX SBOM 형식으로 생성합니다.

리소스

관련 모범 사례:

- DevOps 지침: DL.CR.3 Establish clear completion criteria for code tasks

관련 문서:

- 지속적 전송 및 지속적 배포

- AWS DevOps 컴피던시 파트너
- 애플리케이션 보안을 위한 AWS 보안 컴피던시 파트너
- Choosing a Well-Architected CI/CD approach
- Secrets detection in Amazon CodeGuru Security
- Amazon CodeGuru Security Detection Library
- Accelerate deployments on AWS with effective governance
- AWS에서 안전하고 간편한 배포를 자동화하는 접근 방법
- How Amazon CodeGuru Security helps you effectively balance security and velocity

관련 비디오:

- Hands-off: Automating continuous delivery pipelines at Amazon
- Automating cross-account CI/CD pipelines
- The Software Development Prcess at Amazon
- Testing software and systems at Amazon

관련 예제:

- Industry awareness for developers
- Automated Security Helper (ASH)
- AWS CodePipeline Governance - Github

SEC11-BP03 정기적인 침투 테스트 시행

정기적으로 소프트웨어 침투 테스트를 시행하세요. 이러한 메커니즘은 자동화된 테스트나 수동 코드 검토로 감지할 수 없는 잠재적인 소프트웨어 문제를 식별하는 데 도움이 됩니다. 또한 탐지 컨트롤의 효율성을 이해하는 데 도움이 될 수 있습니다. 침투 테스트를 통해 보호해야 하는 데이터를 노출하거나 예상보다 더 광범위한 권한을 부여하는 등 소프트웨어가 예기치 않은 방식으로 작동할 가능성이 있는지 확인해야 합니다.

원하는 성과: 침투 테스트는 애플리케이션의 보안 속성을 탐지, 수정 및 검증하는 데 사용됩니다. 소프트웨어 개발 수명 주기(SDLC)의 일부로 일정을 정해 정기적인 침투 테스트를 수행해야 합니다. 침투

테스트로 인해 발견한 결과는 소프트웨어 출시 전에 해결해야 합니다. 침투 테스트의 결과를 분석하여 자동화를 통해 찾을 수 있는 문제가 있는지 확인해야 합니다. 능동적 피드백 메커니즘을 포함하는 정기적이고 반복 가능한 침투 테스트 프로세스를 통해 빌더에게 지침을 제공하고 소프트웨어 품질을 개선할 수 있습니다.

일반적인 안티 패턴:

- 알려진 보안 문제 또는 일반적인 보안 문제에 대한 침투 테스트만 수행합니다.
- 종속된 서드파티 도구 및 라이브러리가 없는 애플리케이션에 대한 침투 테스트만 수행합니다.
- 패키지 보안 문제에 대한 침투 테스트만 수행하고 구현된 비즈니스 논리는 평가하지 않습니다.

이 모범 사례 확립의 이점:

- 릴리스 전에 소프트웨어의 보안 속성에 대한 신뢰도가 높아집니다.
- 선호하는 애플리케이션 패턴을 식별할 수 있는 기회를 제공하여 소프트웨어 품질을 개선합니다.
- 피드백 루프를 통해 자동화 또는 추가 교육으로 소프트웨어의 보안 속성을 개선할 여지가 있는 개발 주기의 초기 단계를 식별할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 지침

침투 테스트는 계획된 보안 위반 시나리오를 실행하여 보안 제어 기능을 탐지, 문제 해결 및 검증하는 체계적인 보안 테스트 활동입니다. 침투 테스트는 정찰부터 시작되는데, 이때 애플리케이션의 현재 설계와 종속성을 기반으로 데이터가 수집됩니다. 보안 관련 테스트 시나리오의 선별 목록도 작성되고 실행됩니다. 침투 테스트의 주요 목적은 환경이나 데이터에 무단으로 액세스할 권한을 얻는 데 악용될 수 있는 애플리케이션의 보안 문제를 파악하는 것입니다. 새 기능을 출시할 때 또는 애플리케이션 기능이나 기술 구현이 크게 변경될 때마다 침투 테스트를 수행해야 합니다.

침투 테스트를 수행하려면 개발 수명 주기에서 가장 적합한 단계를 식별해야 합니다. 시스템의 기능이 원하는 릴리스 상태에 근접했을 정도로 늦은 시점에 수행하되, 이때 문제를 해결할 시간이 충분히 남아 있어야 합니다.

구현 단계

- 침투 테스트의 범위를 파악하는 체계적인 프로세스를 마련합니다. 이 프로세스를 위협 모델에 기반하면 바람직한 방식으로 컨텍스트를 유지 관리할 수 있습니다.

- 침투 테스트를 수행하기 위한 개발 주기의 적절한 시점을 파악합니다. 애플리케이션에 예상되는 변경이 최소한만 남아 있는 동시에 문제를 해결하기에 시간이 충분한 시점이어야 합니다.
- 빌더에게 침투 테스트 결과에서 기대할 수 있는 정보를 알려주고 문제 해결 정보를 얻는 방법을 교육합니다.
- 도구를 통해 공통 또는 반복 가능한 테스트를 자동화하여 침투 테스트 프로세스를 가속화합니다.
- 침투 테스트 결과를 분석하여 시스템 보안 문제를 식별하고, 이 데이터를 바탕으로 자동화된 테스트를 추가로 수행하고 빌더를 꾸준히 교육합니다.

리소스

관련 모범 사례:

- SEC11-BP01 애플리케이션 보안 교육
- SEC11-BP02 개발 및 릴리스 수명 주기를 통한 테스트 자동화

관련 문서:

- AWS 침투 테스트에서는 AWS에서의 침투 테스트에 대한 자세한 지침을 제공합니다.
- Accelerate deployments on AWS with effective governance
- AWS 보안 컴피턴시 파트너
- Modernize your penetration testing architecture on AWS Fargate
- AWS Fault Injection Simulator

관련 예제:

- Automate API testing with AWS CodePipeline(GitHub)
- Automated security helper(GitHub)

SEC11-BP04 코드 검토 수행

코드 검토를 구현하여 개발 중인 소프트웨어의 품질과 보안을 확인합니다. 코드 검토에는 원래 코드 작성자 이외의 팀원이 코드에서 잠재적 문제, 취약성, 코딩 표준 및 모범 사례 준수 여부를 검토하도록 하는 것이 포함됩니다. 이 프로세스는 원래 개발자가 간과했을 수 있는 오류, 불일치 및 보안 결함을 포착하는 데 도움이 됩니다. 자동 도구를 사용하여 코드 검토를 지원합니다.

원하는 성과: 작성 중인 소프트웨어의 품질을 높이기 위해 개발 중에 코드 검토를 포함합니다. 코드 검토 중에 식별한 내용을 통해 경험이 부족한 팀원의 스킬을 향상합니다. 자동화 기회를 식별하고 자동화된 도구 및 테스트를 사용하여 코드 검토 프로세스를 지원합니다.

일반적인 안티 패턴:

- 배포 전에 코드 검토를 수행하지 않습니다.
- 같은 사람이 코드를 작성하고 검토합니다.
- 코드 검토를 지원하거나 오케스트레이션하는 데 자동화와 도구를 사용하지 않습니다.
- 빌더가 코드를 검토하기 전에 빌더에게 애플리케이션 보안 교육을 하지 않습니다.

이 모범 사례 확립의 이점:

- 코드 품질이 향상됩니다.
- 공통된 접근 방식을 재사용할 수 있어 코드 개발의 일관성이 높아집니다.
- 침투 테스트 및 이후 단계에서 발견되는 문제의 수가 줄어듭니다.
- 팀 내 지식 전달이 개선됩니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 지침

코드 검토는 개발 중에 소프트웨어의 품질과 보안을 확인하는 데 도움이 됩니다. 수동 검토에는 원래 코드 작성자 이외의 팀원이 코드에서 잠재적 문제, 취약성, 코딩 표준 및 모범 사례 준수 여부를 검토하도록 하는 것이 포함됩니다. 이 프로세스는 원래 개발자가 간과했을 수 있는 오류, 불일치 및 보안 결함을 포착하는 데 도움이 됩니다.

자동 코드 검토를 수행하기 위해 Amazon CodeGuru Security를 고려합니다. CodeGuru Security는 기계 학습 및 자동 추론을 사용하여 코드를 분석하고 잠재적 보안 취약성 및 코딩 문제를 식별합니다. 자동화된 코드 검토를 기존 코드 리포지토리 및 지속적 통합/지속적 배포(CI/CD) 파이프라인과 통합합니다.

구현 단계

1. 코드 검토 프로세스 수립:
    - 코드를 주 브랜치에 병합하기 전 또는 프로덕션에 배포하기 전과 같이 코드 검토를 해야 하는 시기를 정의합니다.

- 팀원, 선임 개발자, 보안 전문가와 같이 코드 검토 프로세스에 참여해야 하는 사람을 결정합니다.
- 사용할 프로세스 및 도구를 포함하여 코드 검토 방법론을 결정합니다.

2. 코드 검토 도구 설정:
   - GitHub Pull 요청 또는 CodeGuru Security와 같이 팀의 요구 사항에 맞는 코드 검토 도구를 평가하고 선택합니다.
   - 선택한 도구를 기존 코드 리포지토리 및 CI/CD 파이프라인과 통합합니다.
   - 최소 검토자 수 및 승인 규칙과 같은 코드 검토 요구 사항을 적용하도록 도구를 구성합니다.
3. 코드 검토 체크리스트 및 지침 정의:
   - 검토해야 할 사항을 설명하는 코드 검토 체크리스트 또는 지침을 생성합니다. 코드 품질, 보안 취약성, 코딩 표준 준수 및 성능과 같은 요소를 고려합니다.
   - 체크리스트 또는 지침을 개발 팀과 공유하고 모든 사람이 기대 사항을 이해하고 있는지 확인합니다.
4. 코드 검토 모범 사례에 대해 개발자 교육:
   - 효과적인 코드 검토를 수행하는 방법에 대해 팀에 교육을 제공합니다.
   - 검토 중에 찾아야 할 애플리케이션 보안 원칙과 일반적인 취약성에 대해 팀을 교육합니다.
   - 지식 공유를 장려하고 프로그래밍 세션을 병행하여 경험이 부족한 팀원의 스킬을 향상합니다.
5. 코드 검토 프로세스 구현:
   - Pull 요청 생성 및 검토자 할당과 같은 코드 검토 단계를 개발 워크플로에 통합합니다.
   - 코드 변경을 병합 또는 배포하기 전에 코드 검토를 거치도록 합니다.
   - 검토 프로세스 중에 열린 커뮤니케이션과 건설적인 피드백을 장려합니다.
6. 모니터링 및 개선:
   - 코드 검토 프로세스의 효과를 정기적으로 검토하고 팀으로부터 피드백을 수집합니다.
   - 코드 검토 프로세스를 간소화하기 위한 자동화 또는 도구 개선 기회를 식별합니다.
   - 알게 된 내용 및 업계 모범 사례를 기반으로 코드 검토 체크리스트 또는 지침을 지속적으로 업데이트하고 개선합니다.
7. 코드 검토 문화 조성:
   - 코드 품질과 보안을 유지하기 위해 코드 검토의 중요성을 강조합니다.
   - 코드 검토 프로세스에서 얻은 성공과 배운 내용을 축하하고 기념합니다.
   - 개발자가 편안하게 피드백을 주고받을 수 있는 협력적이고 지지하는 분위기를 장려합니다.

리소스

관련 모범 사례:

- SEC11-BP02 개발 및 릴리스 수명 주기를 통한 테스트 자동화

관련 문서:

- DevOps 지침: DL.CR.2 Perform peer review for code changes
- About pull requests in GitHub

관련 예제:

- Automate code reviews with Amazon CodeGuru Security
- Automating detection of security vulnerabilities and bugs in CI/CD pipelines using Amazon CodeGuru Security CLI

관련 비디오:

- Continuous improvement of code quality with Amazon CodeGuru Security

SEC11-BP05 패키지 및 종속성 서비스의 중앙 집중화

팀이 소프트웨어 패키지 및 기타 종속성을 확보할 수 있도록 서비스를 중앙 집중화하세요. 서비스를 중앙 집중화하면 작성하는 소프트웨어에 포함하기 전에 패키지를 검증할 수 있습니다. 또한 조직에서 사용 중인 소프트웨어 분석에 쓰일 데이터를 중앙 집중화된 소스에서 얻을 수 있습니다.

원하는 성과: 작성하는 코드 외에도 외부 소프트웨어 패키지에서 워크로드를 구축합니다. 이렇게 하면 JSON 구문 분석기 또는 암호화 라이브러리와 같이 반복적으로 사용되는 기능을 더 간단하게 구현할 수 있습니다. 이러한 패키지 및 종속성에 대한 소스를 중앙 집중화하여 보안 팀이 사용하기 전에 검증할 수 있도록 합니다. 수동 및 자동 테스트 흐름과 함께 이 전략을 사용하여 개발하는 소프트웨어의 품질에 대한 신뢰도를 높입니다.

일반적인 안티 패턴:

- 인터넷의 임의 리포지토리에서 패키지를 가져옵니다.
- 빌더에게 새 패키지를 제공하기 전에 테스트하지 않습니다.

이 모범 사례 확립의 이점:

- 구축 중인 소프트웨어에서 어떤 패키지가 사용되고 있는지 효과적으로 이해할 수 있습니다.
- 누가 무엇을 사용하고 있는지 파악한 후에 패키지를 업데이트해야 할 때 워크로드 팀에 알릴 수 있습니다.
- 소프트웨어에 문제가 포함된 패키지의 위험을 줄입니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 중간

구현 지침

패키지 및 종속성에 대한 중앙 집중식 서비스를 빌더가 쉽게 사용할 수 있는 방식으로 제공합니다. 중앙 집중식 서비스는 단일 시스템으로 구현되기보다는 논리적으로 중앙에 배치될 수 있습니다. 이러한 접근 방식을 통해 빌더의 요구를 충족하는 방향으로 서비스를 제공할 수 있습니다. 업데이트가 발생하거나 새로운 요구 사항이 나타날 때 패키지를 리포지토리에 추가하는 효율적인 방법을 구현해야 합니다. AWS CodeArtifact와 같은 AWS 서비스 또는 이와 유사한 AWS 파트너 솔루션은 이러한 기능을 제공하는 방법을 안내합니다.

구현 단계

- 소프트웨어가 개발되는 모든 환경에서 사용할 수 있는 논리적으로 중앙 집중화된 리포지토리 서비스를 구현합니다.
- 리포지토리에 대한 액세스를 AWS 계정 벤딩 프로세스의 일부로 포함합니다.
- 패키지를 리포지토리에 게시하기 전에 테스트하는 자동화를 구축합니다.
- 가장 일반적으로 사용되는 패키지, 언어 및 변경 사항이 제일 많은 팀의 지표를 유지 관리합니다.
- 빌더 팀이 새 패키지를 요청하고 피드백을 줄 수 있도록 자동화된 메커니즘을 제공합니다.
- 리포지토리의 패키지를 정기적으로 스캔하여 새로 발견된 문제의 잠재적 영향을 식별합니다.

리소스

관련 모범 사례:

- SEC11-BP02 개발 및 릴리스 수명 주기를 통한 테스트 자동화

관련 문서:

- DevOps 지침: DL.CS.2 Sign code artifacts after each build
- Supply chain Levels for Software Artifacts (SLSA)

관련 예제:

- Accelerate deployments on AWS with effective governance
- Tighten your package security with CodeArtifact Package Origin Control toolkit
- Multi Region Package Publishing Pipeline(GitHub)
- Publishing Node.js Modules on AWS CodeArtifact using AWS CodePipeline(GitHub)
- AWS CDK Java CodeArtifact Pipeline Sample(GitHub)
- Distribute private .NET NuGet packages with AWS CodeArtifact(GitHub)

관련 비디오:

- Proactive security: Considerations and approaches
- The AWS Philosophy of Security (re:Invent 2017)
- When security, safety, and urgency all matter: Handling Log4Shell

SEC11-BP06 프로그래밍 방식으로 소프트웨어 배포

가능한 한 프로그래밍 방식으로 소프트웨어를 배포하세요. 이 접근 방식을 통해 인적 오류로 배포 실패나 예기치 않은 문제가 발생할 가능성을 줄일 수 있습니다.

원하는 성과: 테스트하는 워크로드의 버전은 배포하는 버전이며 배포는 매번 일관되게 수행됩니다. 워크로드 구성을 외부화하여 변경 없이 다양한 환경에 배포할 수 있습니다. 암호화된 서명 소프트웨어 패키지를 사용하여 환경 간에 변경된 내용이 없음을 확인합니다.

일반적인 안티 패턴:

- 프로덕션에 소프트웨어를 수동으로 배포합니다.
- 다양한 환경에 맞게 소프트웨어를 수동으로 변경합니다.

이 모범 사례 확립의 이점:

- 소프트웨어 릴리스 프로세스에 대한 신뢰도가 높아집니다.

- 비즈니스 기능에 영향을 미치는 변경 실패 위험을 줄여줍니다.
- 변경 위험 감소로 인해 릴리스 주기가 길어집니다.
- 배포 중 예기치 않은 이벤트에 대한 자동 롤백 기능이 지원됩니다.
- 테스트된 소프트웨어가 배포된 소프트웨어임을 암호화하여 증명하는 기능이 제공됩니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 지침

강력하고 신뢰할 수 있는 애플리케이션 인프라를 유지하기 위해 안전하고 자동화된 배포 방식을 구현합니다. 이 방식에는 프로덕션 환경에서 지속적인 인적 액세스 제거, 배포를 위해 CI/CD 도구 사용, 환경별 구성 데이터 외부화가 포함됩니다. 이 접근 방식을 따르면 보안을 강화하고 인적 오류의 위험을 줄이며 배포 프로세스를 간소화할 수 있습니다.

AWS 계정 구조를 구축하여 프로덕션 환경에서 지속적인 인적 액세스를 제거할 수 있습니다. 이 방식은 무단 변경 또는 우발적 수정의 위험을 최소화하여 프로덕션 시스템의 무결성을 개선합니다. 사람의 직접적인 액세스 대신 AWS CodeBuild 및 AWS CodePipeline과 같은 CI/CD 도구를 사용하여 배포를 수행할 수 있습니다. 이러한 서비스를 사용하여 구축, 테스트 및 배포 프로세스를 자동화하여 수동 개입을 줄이고 일관성을 높일 수 있습니다.

보안 및 추적성을 더욱 강화하기 위해 애플리케이션 패키지를 테스트한 후 애플리케이션 패키지에 서명하고 배포 중에 이러한 서명을 검증할 수 있습니다. 이렇게 하려면 AWS Signer 또는 AWS Key Management Service(AWS KMS)와 같은 암호화 도구를 사용합니다. 패키지에 서명하고 확인하면 인증되고 검증된 코드만 환경에 배포할 수 있습니다.

또한 팀은 워크로드를 아키텍팅하여 AWS Systems Manager Parameter Store와 같은 외부 소스에서 환경별 구성 데이터를 얻을 수 있습니다. 이 방식은 애플리케이션 코드를 구성 데이터와 분리하므로 애플리케이션 코드 자체를 수정하지 않고도 독립적으로 구성을 관리하고 업데이트할 수 있습니다.

인프라 프로비저닝 및 관리를 간소화하려면 AWS CloudFormation 또는 AWS CDK와 같은 코드형 인프라(IaC) 도구를 사용하는 것이 좋습니다. 이러한 도구를 사용하여 인프라를 코드로 정의하여 다양한 환경에서 배포의 일관성과 반복성을 개선할 수 있습니다.

소프트웨어의 성공적인 배포를 검증하기 위해 카나리 배포를 고려합니다. 카나리 배포에는 전체 프로덕션 환경에 배포하기 전에 인스턴스 또는 사용자 하위 집합에 대한 변경 사항을 롤아웃하는 작업이 포함됩니다. 그런 다음 변경의 영향을 모니터링하고 필요한 경우 롤백하여 문제가 확산될 위험을 최소화할 수 있습니다.

Organizing Your AWS Environment Using Multiple Accounts 백서에 설명된 권장 사항을 따릅니다. 이 백서는 환경을 고유한 AWS 계정으로 분리(예: 개발, 스테이징 및 프로덕션)하는 방법에 대한 지침을 제공합니다. 이 방식은 보안 및 격리를 더욱 강화합니다.

구현 단계

1. AWS 계정 구조 설정:
   - Organizing Your AWS Environment Using Multiple Accounts 백서의 지침에 따라 다양한 환경(예: 개발, 스테이징 및 프로덕션)에 대해 별도의 AWS 계정을 생성합니다.
   - 각 계정에 대한 적절한 액세스 제어 및 권한을 구성하여 프로덕션 환경에 대한 사람의 직접적인 액세스를 제한합니다.
2. CI/CD 파이프라인 구현:
   - AWS CodeBuild 및 AWS CodePipeline과 같은 서비스를 사용하여 CI/CD 파이프라인을 설정합니다.
   - 각 환경에 애플리케이션 코드를 자동으로 구축, 테스트 및 배포하도록 파이프라인을 구성합니다.
   - 버전 관리 및 코드 관리를 위해 코드 리포지토리를 CI/CD 파이프라인과 통합합니다.
3. 애플리케이션 패키지에 서명 및 확인:
   - AWS Signer 또는 AWS Key Management Service(AWS KMS)를 사용하여 애플리케이션 패키지를 테스트하고 검증한 후 서명합니다.
   - 대상 환경에 배포하기 전에 애플리케이션 패키지의 서명을 확인하도록 배포 프로세스를 구성합니다.
4. 구성 데이터 외부화:
   - 환경별 구성 데이터를 AWS Systems Manager Parameter Store에 저장합니다.
   - 배포 또는 런타임 중에 Parameter Store에서 구성 데이터를 검색하도록 애플리케이션 코드를 수정합니다.
5. 코드형 인프라(IaC) 구현:
   - AWS CloudFormation 또는 AWS CDK와 같은 IaC 도구를 사용하여 인프라를 코드로 정의하고 관리합니다.
   - CloudFormation 템플릿 또는 CDK 스크립트를 생성하여 애플리케이션에 필요한 AWS 리소스를 프로비저닝하고 구성합니다.
   - IaC를 CI/CD 파이프라인과 통합하여 애플리케이션 코드 변경과 함께 인프라 변경 사항을 자동으로 배포합니다.
6. 카나리 배포 구현:

- 전체 프로덕션 환경에 배포하기 전에 인스턴스 또는 사용자 하위 집합에 변경 사항이 롤아웃되는 카나리 배포를 지원하도록 배포 프로세스를 구성합니다.
- AWS CodeDeploy 또는 AWS ECS와 같은 서비스를 사용하여 카나리 배포를 관리하고 변경의 영향을 모니터링합니다.
- 카나리 배포 중에 문제가 감지되면 롤백 메커니즘을 구현하여 이전의 안정적인 버전으로 되돌립니다.

7. 모니터링 및 감사:
   - 모니터링 및 로깅 메커니즘을 설정하여 배포, 애플리케이션 성능 및 인프라 변경을 추적합니다.
   - Amazon CloudWatch 및 AWS CloudTrail과 같은 서비스를 사용하여 로그 및 지표를 수집하고 분석합니다.
   - 감사 및 규정 준수 검사를 구현하여 보안 모범 사례 및 규제 요구 사항 준수 여부를 확인합니다.
8. 지속적 개선:
   - 배포 방식을 정기적으로 검토 및 업데이트하고 이전 배포에서 얻은 피드백과 배운 내용을 반영합니다.
   - 배포 프로세스를 최대한 자동화하여 수동 개입과 잠재적인 인적 오류를 줄입니다.
   - 여러 분야의 팀원으로 구성된 팀(예: 운영 또는 보안)과 협력하여 배포 방식을 조정하고 지속적으로 개선합니다.

이러한 단계를 따르면 AWS 환경에서 안전하고 자동화된 배포 방식을 구현하여 보안을 강화하고 인적 오류의 위험을 줄이며 배포 프로세스를 간소화할 수 있습니다.

리소스

관련 모범 사례:

- SEC11-BP02 개발 및 릴리스 수명 주기를 통한 테스트 자동화
- DL.CI.2 Trigger builds automatically upon source code modifications

관련 문서:

- Accelerate deployments on AWS with effective governance
- 안전하고 간편한 배포 자동화
- Code signing using AWS Certificate Manager Private CA and AWS Key Management Service asymmetric keys

- Code Signing, a Trust and Integrity Control for AWS Lambda

관련 비디오:

- Hands-off: Automating continuous delivery pipelines at Amazon

관련 예제:

- Blue/Green deployments with AWS Fargate

SEC11-BP07 정기적으로 파이프라인의 보안 속성 평가

특히 권한 분리에 주의를 기울여 Well-Architected 보안 원칙을 파이프라인에 적용하세요. 파이프라인 인프라의 보안 속성을 정기적으로 평가합니다. 파이프라인 자체 보안을 효율적으로 관리하면 파이프라인을 통과하는 소프트웨어의 보안을 보장할 수 있습니다.

원하는 성과: 소프트웨어를 구축하고 배포하는 데 사용하는 파이프라인이 환경의 다른 워크로드와 동일한 권장 사례를 따릅니다. 파이프라인에서 구현하는 테스트는 이를 사용하는 팀에서 편집할 수 없습니다. 파이프라인에는 임시 자격 증명을 사용하여 수행하는 배포에 필요한 권한만 부여합니다. 파이프라인이 잘못된 환경에 배포되지 않도록 보호 조치를 구현합니다. 구축 환경의 무결성을 검증할 수 있도록 상태를 내보내도록 파이프라인을 구성합니다.

일반적인 안티 패턴:

- 빌더가 보안 테스트를 우회할 수 있습니다.
- 배포 파이프라인에 대한 권한이 지나치게 광범위합니다.
- 입력을 검증하도록 파이프라인을 구성하지 않습니다.
- CI/CD 인프라와 관련된 권한을 정기적으로 검토하지 않습니다.
- 장기 또는 하드코딩된 자격 증명을 사용합니다.

이 모범 사례 확립의 이점:

- 파이프라인을 통해 구축 및 배포되는 소프트웨어의 무결성에 대한 신뢰도가 높아집니다.
- 의심스러운 활동이 있을 때 배포를 중지할 수 있습니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 높음

구현 지침

배포 파이프라인은 소프트웨어 개발 수명 주기의 중요한 구성 요소이며 환경의 다른 워크로드와 동일한 보안 원칙 및 방식을 따라야 합니다. 여기에는 적절한 액세스 제어 구현, 입력 내용 검증, CI/CD 인프라와 관련된 권한의 정기적인 검토 및 감사가 포함됩니다.

애플리케이션 구축 및 배포를 담당하는 팀이 파이프라인에 구현된 보안 테스트 및 검사를 편집하거나 우회할 수 없는지 확인합니다. 이렇게 우려되는 사항이 발생하지 않도록 분리하면 구축 및 배포 프로세스의 무결성을 유지하는 데 도움이 됩니다.

먼저 AWS 배포 파이프라인 참조 아키텍처를 사용하는 것이 좋습니다. 이 참조 아키텍처는 AWS에서 CI/CD 파이프라인을 구축하기 위한 안전하고 확장 가능한 기반을 제공합니다.

또한 AWS Identity and Access Management Access Analyzer와 같은 서비스를 사용하여 파이프라인 권한 모두에 대해 최소 권한 IAM 정책을 생성하고 파이프라인의 한 단계로 워크로드 권한을 확인할 수 있습니다. 이렇게 하면 파이프라인과 워크로드에 특정 기능에 필요한 권한만 있는지 확인할 수 있으므로 무단 액세스 또는 무단 작업의 위험이 줄어듭니다.

구현 단계

- AWS 배포 파이프라인 참조 아키텍처부터 시작합니다.
- 파이프라인에 대한 최소 권한 IAM 정책을 프로그래밍 방식으로 생성하기 위해 AWS IAM Access Analyzer 사용을 고려하세요.
- 파이프라인을 모니터링 및 알림과 통합하여 예기치 않거나 비정상적인 활동이 발생할 경우 알림을 받을 수 있습니다. AWS 관리형 서비스의 경우 Amazon EventBridge를 사용하면 AWS Lambda 또는 Amazon Simple Notification Service(SNS)와 같은 대상으로 데이터를 라우팅할 수 있습니다.

리소스

관련 문서:

- AWS Deployment Pipelines Reference Architecture
- AWS CodePipeline 모니터링
- AWS CodePipeline의 보안 모범 사례

관련 예제:

- DevOps monitoring dashboard(GitHub)

SEC11-BP08 보안 소유권이 워크로드 팀에 귀속되도록 프로그램 구축

빌더 팀이 개발하는 소프트웨어의 보안을 결정할 수 있도록 지원하는 프로그램 또는 메커니즘을 구축합니다. 보안 팀에서 검토 시 이러한 결정을 다시금 검증해야 하지만, 빌더 팀이 보안 소유권을 가지면 더욱 신속하고 안전하게 워크로드를 구축할 수 있습니다. 또한 이 메커니즘은 구축하는 시스템의 운영에 좋은 영향을 미치는 주인 의식 문화를 강화합니다.

원하는 성과: 팀에 보안 소유권과 의사 결정이 포함되어 있습니다. 보안에 대해 생각하는 방법을 팀에 교육했거나 소속된 인력 또는 관련 보안 인력으로 팀을 강화했습니다. 그 결과 팀은 개발 주기 초반에 더 양질의 보안 결정을 내립니다.

일반적인 안티 패턴:

- 보안 팀에 모든 보안 설계 결정을 맡깁니다.
- 개발 프로세스에서 보안 요구 사항을 조기에 해결하지 못합니다.
- 빌더 및 보안 담당자로부터 프로그램 운영 피드백을 받지 못합니다.

이 모범 사례 확립의 이점:

- 보안 검토를 빠르게 완료할 수 있습니다.
- 보안 검토 단계에서만 탐지되는 보안 문제가 감소합니다.
- 작성 중인 소프트웨어의 전반적인 품질이 향상됩니다.
- 체계 문제 또는 유의미한 개선 영역을 식별하고 이해할 수 있습니다.
- 보안 검토 결과로 인해 필요한 재작업의 양이 줄어듭니다.
- 보안 기능에 대한 인식이 개선됩니다.

이 모범 사례가 확립되지 않을 경우 노출되는 위험 수준: 낮음

구현 지침

SEC11-BP01 애플리케이션 보안 교육의 지침부터 시작합니다. 그런 다음 조직에 가장 적합하다고 생각되는 프로그램의 운영 모델을 파악합니다. 2가지 주요 패턴은 빌더를 교육하거나 빌더 팀에 보안 인력을 투입하는 것입니다. 초기 접근 방식을 정한 후에는 단일 또는 소규모 워크로드 팀과 함께 시범 운영하여 해당 모델이 조직에 적합한지 입증해야 합니다. 조직의 빌더 및 보안 부문에서 리더십의 지원이

있으면 프로그램의 제공과 성공에 도움이 됩니다. 이 프로그램을 개발할 때 프로그램의 가치를 보여주는 데 사용할 수 있는 지표를 선택해야 합니다. AWS가 이러한 문제에 어떻게 접근했는지 알아보면 큰 도움이 됩니다. 이 모범 사례는 조직의 변화와 문화를 집중적으로 조명합니다. 빌더와 보안 커뮤니티 간의 협업을 지원하는 도구를 사용해야 합니다.

구현 단계

- 먼저 빌더에게 애플리케이션 보안 교육을 제공합니다.
- 빌더를 교육하기 위한 커뮤니티와 온보딩 프로그램을 개발합니다.
- 프로그램 이름을 선택합니다. Guardians, Champions, Advocates가 주로 사용됩니다.
- 빌더를 교육하거나, 보안 엔지니어를 합류시키거나, 보안 담당자를 연계하는 등 사용할 모델을 결정합니다.
- 보안, 빌더 및 기타 관련 그룹의 프로젝트 후원 주체를 결정합니다.
- 프로그램에 참여한 인원 수, 검토에 소요된 시간, 빌더 및 보안 인력의 피드백 지표를 추적합니다. 이러한 지표를 바탕으로 개선합니다.

리소스

관련 모범 사례:

- SEC11-BP01 애플리케이션 보안 교육
- SEC11-BP02 개발 및 릴리스 수명 주기를 통한 테스트 자동화

관련 문서:

- How to approach threat modeling
- How to think about cloud security governance
- How AWS built the Security Guardians program, a mechanism to distribute security ownership
- How to build a Security Guardians program to distribute security ownership

관련 비디오:

- Proactive security: Considerations and approaches
- AppSec tooling and culture tips from AWS and Toyota Motor North America

결론

보안을 유지하려면 지속적인 작업을 수행해야 합니다. 인시던트가 실제로 발생한 경우, 이를 아키텍처 보안을 개선할 기회로 간주해야 합니다. 강력한 자격 증명 제어를 적용하고, 보안 인시던트에 대한 대응을 자동화하며, 여러 수준에서 인프라를 보호하고, 암호화를 통해 적절하게 분류된 데이터를 관리하면 모든 조직에서 구현해야 하는 심층 방어 기능이 제공됩니다. 이 백서에서 설명한 프로그래밍 방식 함수와 AWS 기능 및 서비스를 사용하면 이 작업을 더 쉽게 수행할 수 있습니다.

AWS는 비즈니스 가치를 제공하는 동시에 정보, 시스템 및 자산을 보호하는 아키텍처를 구축하고 운영하는 과정을 지원합니다.

기여자

다음 개인과 조직이 이 문서에 기여했습니다.

- Jay Michael, Amazon Web Services의 Principal Security Lead Solutions Architect
- Kiaan Sumeet, Amazon Web Services의 Principal Security Consultant
- Michael Haken, Amazon Web Services의 Principal Solutions Architect
- James Devine, Amazon Web Services의 Principal Solutions Architect
- Dave Walker, Amazon Web Services의 Security & Compliance, Principal Solutions Architect
- Patrick Palmer, Amazon Web Services의 Security & Compliance, Principal Solutions Architect
- Monka Vu Minh, Amazon Web Services의 Security Consultant
- Kurt Kumar, Amazon Web Services의 Security Consultant
- Fahima Khan, Amazon Web Services의 Security Solutions Architect
- Mutaz Hajeer, Amazon Web Services의 Senior Security Solutions Architect
- Luis Pastor, Amazon Web Services의 Senior Security Solutions Architect
- Colin Igbokwe, Amazon Web Services의 Senior Security Solutions Architect
- Geoff Sweet, Amazon Web Services의 Senior Security Solutions Architect
- Anthony Harvey, Amazon Web Services의 Senior Security Solutions Architect
- Sowjanya Rajavaram, Amazon Web Services의 Senior Security Solutions Architect
- Krishna Prasad, Amazon Web Services의 Senior Solutions Architect
- Faisal Farooq, Amazon Web Services의 Senior Solutions Architect
- Arun Krishnaswamy, Amazon Web Services의 Senior Solutions Architect
- Dan Girard, Amazon Web Services의 Senior Solutions Architect
- Marc Luescher, Amazon Web Services의 Senior Solutions Architect
- Kyle Nicodemus, Amazon Web Services의 Senior Technical Account Manager
- Irina Szabo, Amazon Web Services의 Senior Technical Account Manager
- Arun Sivaraman, Amazon Web Services의 Solutions Architect
- Stephen Novak, Amazon Web Services의 Technical Account Manager
- Jonathan Risbrook, Amazon Web Services의 Technical Account Manager
- Freddy Kasprzykowski, Amazon Web Services의 Global Financial Services, Practice Manager
- Pat Gaw, Amazon Web Services의 Principal Security Consultant

- Jason Garman, Amazon Web Services의 Principal Security Solutions Architect
- Mark Keating, Amazon Web Services의 Principal Security Solutions Architect
- Zach Miller, Amazon Web Services의 Principal Security Solutions Architect
- Maitreya Ranganath, Amazon Web Services의 Principal Security Solutions Architect
- Reef Dsouza, Amazon Web Services의 Principal Solutions Architect
- Brad Burnett, Amazon Web Services의 Security Solutions Architect
- Matt Saner, Amazon Web Services의 Security Solutions Architecture, Senior Manager
- Priyank Ghedia, Amazon Web Services의 Senior Security Solutions Architect
- Arthur Mnev, Amazon Web Services의 Senior Security Solutions Architect
- Kyle Dickinson, Amazon Web Services의 Senior Security Solutions Architect
- Kevin Boland, Amazon Web Services의 Senior Security Solutions Architect
- Anna McAbee, Amazon Web Services의 Senior Security Solutions Architect
- Recep Meric Degirmenci, Amazon Web Services의 Senior Security Solutions Architect
- Daniel Salzedo, Amazon Web Services의 Senior Security Technical Product Manager
- Jake Izumi, Amazon Web Services의 Senior Solutions Architect
- Bert Bullough, Amazon Web Services의 Senior Solutions Architect
- Robert McCall, Amazon Web Services의 Solutions Architect
- Angela Chao, Amazon Web Services의 AWS Enterprise Support, ESL TAM
- Pratima Singh, Senior ANZ Security Spec. Amazon Web Services의 Solutions Architect
- Darran Boyd, Amazon Web Services의 AWS Security Principal, Office of the CISO
- Byron Pogson, Amazon Web Services의 Senior Security Solutions Architect

참조 자료

자세한 내용은 다음 출처를 참조하세요.

- AWS Well-Architected Framework 백서
- AWS 아키텍처 센터

문서 수정

이 백서에 대한 업데이트 알림을 받으려면 RSS 피드를 구독하면 됩니다.

변경 사항	설명	날짜
모범 사례 지침 업데이트됨	모범 사례는 SEC 2, SEC 3, SEC 4, SEC 6, SEC 7, SEC 8, SEC 9, SEC 10, SEC 11 영역에서 새로운 지침으로 업데이트되었습니다. 원칙 전반에 걸쳐 지침이 업데이트되고 개선되었습니다.	2024년 11월 6일
모범 사례 지침 업데이트됨	원칙 전반에 걸쳐 모범 사례가 대규모로 업데이트되었습니다. 여러 모범 사례가 재정렬되고 통합되었습니다. SEC 1, 4, 5, 6, 7, 8, 9에 중요한 변경 사항이 적용되었습니다.	2024년 6월 27일
모범 사례 지침 업데이트됨	워크로드의 안전한 운영 및 전송 중 데이터 보호 영역에 대한 새로운 지침으로 모범 사례가 업데이트되었습니다.	2023년 12월 6일
모범 사례 지침 업데이트됨	인시던트 대응에 관한 지침 및 모범 사례에 주요 업데이트. 준비 단계에서 여러 모범 사례가 업데이트되었습니다. 인시던트 대응에 운영 및 인시던트 사후 활동이라는 두 가지 새로운 영역이 추가되었습니다. 새 모범 사례 SEC10-BP08 인시던트로부터 학습하기 위한 프	2023년 10월 3일

	레임워크 구축이 추가되었습니다.	
모범 사례 지침 업데이트됨	준비 및 시뮬레이션 영역에서 새로운 지침으로 모범 사례가 업데이트되었습니다.	2023년 7월 13일
새 프레임워크 관련 업데이트.	권장 가이드 및 새로운 모범 사례를 추가하여 모범 사례를 업데이트했습니다. 애플리케이션 보안(AppSec)의 새로운 모범 사례 영역이 추가되었습니다.	2023년 4월 10일
백서 업데이트	새로운 구현 가이드와 함께 모범 사례를 업데이트했습니다.	2022년 12월 15일
백서 업데이트	모범 사례를 확대하고 개선 계획을 추가했습니다.	2022년 10월 20일
마이너 업데이트	현재 모범 사례를 반영하도록 IAM 정보를 업데이트했습니다.	2022년 6월 28일
마이너 업데이트	AWS PrivateLink 정보를 추가하고 연결되지 않는 링크를 수정했습니다.	2022년 5월 19일
마이너 업데이트	AWS PrivateLink 추가.	2022년 5월 6일
마이너 업데이트	포용적이지 않은 표현을 삭제했습니다.	2022년 4월 22일
마이너 업데이트	VPC Network Access Analyzer에 대한 정보를 추가했습니다.	2022년 2월 2일
마이너 업데이트	연결되지 않는 링크를 수정했습니다.	2021년 5월 27일
마이너 업데이트	문서 전반에 편집상 변경 사항을 적용했습니다.	2021년 5월 17일

메이저 업데이트	거버넌스에 대한 섹션을 추가하고 다양한 섹션에 대한 세부 정보를 추가했으며 전반적으로 신규 기능 및 서비스를 추가했습니다.	2021년 5월 7일
마이너 업데이트	링크를 업데이트했습니다.	2021년 3월 10일
마이너 업데이트	연결되지 않는 링크를 수정했습니다.	2020년 7월 15일
새 프레임워크 관련 업데이트	계정, 자격 증명 및 권한 관리에 관한 지침을 업데이트했습니다.	2020년 7월 8일
새 프레임워크 관련 업데이트	모든 영역에서 더 많은 조언을 제공하고 새로운 모범 사례, 서비스 및 기능을 추가하기 위해 업데이트했습니다.	2020년 4월 30일
백서 업데이트	최신 AWS 서비스 및 기능과 새로워진 참조를 반영하여 업데이트했습니다.	2018년 7월 1일
백서 업데이트	새로운 AWS 서비스 및 기능을 반영하기 위해 시스템 보안 구성 및 유지 관리 섹션을 업데이트했습니다.	2017년 5월 1일
최초 게시	보안 원칙 - AWS Well-Architected Framework를 게시했습니다.	2016년 11월 1일

고지 사항

고객은 본 문서의 정보를 독립적으로 평가할 책임이 있습니다. 본 문서는 (a) 정보 제공의 목적으로만 제공되고, (b) 사전 통지 없이 변경될 수 있는 현재 AWS 제품 및 관행을 나타내고, (c) AWS 및 그 계열사, 공급업체 또는 라이선스 제공자로부터 어떠한 약속이나 보증도 하지 않습니다. AWS 제품 또는 서비스는 명시적이든 묵시적이든 어떠한 종류의 보증, 진술 또는 조건 없이 '있는 그대로' 제공됩니다. 고객에 대한 AWS의 책임 및 채무는 AWS 계약에 준거합니다. 본 문서는 AWS와 고객 간의 어떠한 계약도 구성하지 않으며 이를 변경하지도 않습니다.

© 2023 Amazon Web Services, Inc. 또는 계열사. All rights reserved.

AWS 용어집

최신 AWS 용어는 AWS 용어집 참조서의 AWS 용어집을 참조하십시오.