Verifiche di compliance in ambienti Cloud

Presentazione dell'elaborato finale

Laurea Triennale in Sicurezza dei Sistemi e delle Reti Informatiche

Niccolò Volontè (20642A)

16 luglio 2025

UNIVERSITÀ DEGLI STUDI DI MILANO

Introduzione e contesto normativo

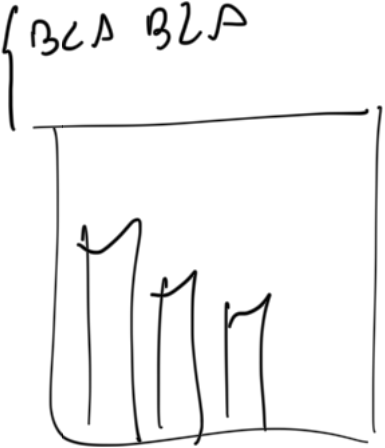
Cloud, compliance e obiettivi

- Crescita del Cloud Computing e della sua adozione

Introduzione e contesto normativo

Cloud, compliance e obiettivi

- Crescita del Cloud Computing e della sua adozione
- AWS: il cloud provider più diffuso su scala globale



Introduzione e contesto normativo

Cloud, compliance e obiettivi

- Crescita del Cloud Computing e della sua adozione
- *AWS*: il cloud provider più diffuso su scala globale
- La *compliance* nel cloud: sfide legate alla gestione di risorse distribuite

Introduzione e contesto normativo

Cloud, compliance e obiettivi

- Crescita del Cloud Computing e della sua adozione
- *AWS*: il cloud provider più diffuso su scala globale
- La *compliance* nel cloud: sfide legate alla gestione di risorse distribuite
- **Standard di riferimento**:
  - CIS AWS Foundations Benchmark
  - NIST SP 800-53

Introduzione e contesto normativo

Cloud, compliance e obiettivi

- Crescita del Cloud Computing e della sua adozione
- *AWS*: il cloud provider più diffuso su scala globale
- La *compliance* nel cloud: sfide legate alla gestione di risorse distribuite
- **Standard di riferimento**:
  - CIS AWS Foundations Benchmark
  - NIST SP 800-53
- **Obiettivo:** sviluppare sonde automatizzabili per verifiche di compliance su AWS, integrabili nella piattaforma Moon Cloud

CIS Center for Internet Security®

NIST NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY U.S. DEPARTMENT OF COMMERCE

Standard di riferimento

Esempio di controllo per `aws_account`

1.4 Ensure MFA is enabled for the 'root' user account (Automated)

- **Profile Applicability:** Level 1
- **Description:** The 'root' user account is the most privileged user in an AWS account. Multi-factor Authentication (MFA) adds an extra layer of protection on top of a username and password. With MFA enabled, when a user signs in to an AWS website, they will be prompted for their username and password as well as for an authentication code from their AWS MFA device.
- **Rationale:** Enabling MFA provides increased security for console access as it requires the authenticating principal to possess a device that emits a time-sensitive key and have knowledge of a credential.

Tecnologie utilizzate

Linguaggi e strumenti

- *Python* come linguaggio di programmazione
- `Boto3` per l'interazione con AWS
- *Moon Cloud* come piattaforma di integrazione

Tecnologie utilizzate

Linguaggi e strumenti

- *Python* come linguaggio di programmazione
- `Boto3` per l'interazione con AWS
- *Moon Cloud* come piattaforma di integrazione

Esempio di utilizzo di Boto3

```
import boto3
client = boto3.client(
 'sqs',
 region_name='eu-central-1',
 aws_access_key_id='YOUR_ACCESS_KEY',
 aws_secret_access_key='YOUR_SECRET_KEY'
)
response = client.list_queues()
```

Analisi della documentazione

AWS Security Hub, Boto3 Account

aws

Get started Service guides Developer tools AI resources

[Account.1] Security contact information should be provided for an AWS account

Related requirements: NIST.800-53.r5 CM-2, NIST.800-53.r5 CM-2(2)

Category: Identify > Resource Configuration

Severity: Medium

Resource type: `AWS::::Account`

AWS Config rule: `security-account-information-provided`

Schedule type: Periodic

Parameters: None

This control checks if an Amazon Web Services (AWS) account has security contact information. The control fails if security contact information is not provided for the account.

Alternate security contacts allow AWS to contact another person about issues with your account in case you're unavailable. Notifications can be from Support, or other AWS service teams about security-related topics associated with your AWS account usage.

Remediation

To add an alternate contact as a security contact to your AWS account, see Update the alternate contacts for your AWS account in the *AWS Account Management Reference Guide*.

(15 pagde servizi, 51 c)

aws

Boto3 1.39.4 documentation

Search

Feedback

Do you have a suggestion to improve this website or boto3? Give us feedback.

Quickstart

A Sample Tutorial

Code Examples

Developer Guide

Security

Available Services

AccessAnalyzer

Account

ACM

ACMPCA

AIOps

PrometheusService

Account / Client / get_alternate_contact

get_alternate_contact

`Account.Client.get_alternate_contact(**kwargs)`

Retrieves the specified alternate contact attached to an Amazon Web Services account.

For complete details about how to use the alternate contact operations, see Access or updating the alternate contacts.

Note

Before you can update the alternate contact information for an Amazon Web Services account that is managed by Organizations, you must first enable integration between Amazon Web Services Account Management and Organizations. For more information, see Enabling trusted access for Amazon Web Services Account Management.

See also: AWS API Documentation

Request Syntax

```
response = client.get_alternate_contact(
    AccountId='string',
    AlternateContactType='BILLING'|'OPERATIONS'|'SECURITY'
)
```

PARAMETERS:

- **AccountId** (*string*) –

  Specifies the 12 digit account ID number of the Amazon Web Services account that you want to access or modify with this operation.

  If you do not specify this parameter, it defaults to the Amazon Web Services account of the identity used to call the operation.

  To use this parameter, the caller must be an identity in the organization's management account or a delegated administrator account, and the specified account ID must be a

Moon Cloud

Piattaforma, funzionalità e architettura

- Esegue sonde di assurance su infrastrutture ICT

Moon Cloud

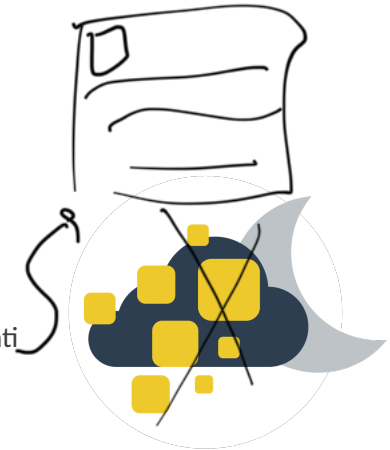
Piattaforma, funzionalità e architettura

- Esegue sonde di assurance su infrastrutture ICT
- Architettura basata su immagini Docker e CI/CD

Moon Cloud

Piattaforma, funzionalità e architettura

- Esegue sonde di assurance su infrastrutture ICT
- Architettura basata su immagini Docker e CI/CD
- Dashboard per la gestione dei target, credenziali e risultati



Moon Cloud

Piattaforma, funzionalità e architettura

- Esegue sonde di assurance su infrastrutture ICT
- Architettura basata su immagini Docker e CI/CD
- Dashboard per la gestione dei target, credenziali e risultati
- Modello a stati finiti: *forward*, *rollback*

Struttura di una sonda

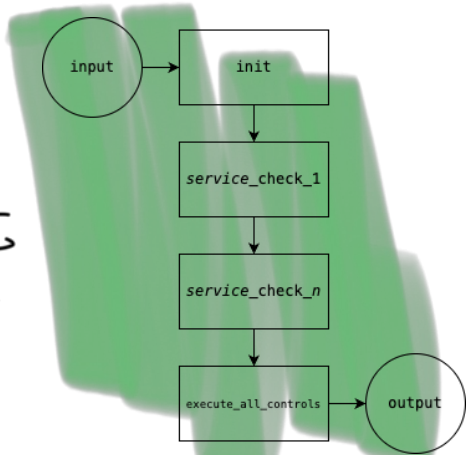
Componenti per la creazione

- Codice Python all'interno del file `probe.py`
- `schema.json` e `test.json` per input validato
- `Dockerfile`, `.gitlab-ci.yml` per la pipeline

Struttura di una sonda

Componenti per la creazione

- Codice Python all'interno del file `probe.py`
- `schema.json` e `test.json` per input validato
- `Dockerfile`, `.gitlab-ci.yml` per la pipeline
- Struttura con `atoms` eseguiti in sequenza
- Output *strutturato*



Suite di sonde

Elenco delle sonde sviluppate

13 sonde sviluppate con **51** controlli

- `aws_sqs`
- `aws_inspector`
- `aws_iam`
- `aws_ec2`
- `aws_s3`
- `aws_account`

Suite di sonde

Elenco delle sonde sviluppate

13 sonde sviluppate con **51** controlli

- `aws_sqs`
- `aws_inspector`
- `aws_iam`
- `aws_ec2`
- `aws_s3`
- `aws_account`
- `aws_config`
- `aws_cloudtrail`
- `aws_efs`
- `aws_kms`
- `aws_rds`
- `aws_eks`

Suite di sonde

Elenco delle sonde sviluppate

13 sonde sviluppate con **51** controlli

- `aws_sqs`
- `aws_inspector`
- `aws_iam`
- `aws_ec2`
- `aws_s3`
- `aws_account`
- `aws_config`
- `aws_cloudtrail`
- `aws_efs`
- `aws_kms`
- `aws_rds`
- `aws_eks`
- `aws_vulnerability`

Esempio: `aws_sqs`

Controllo cifratura, tag e accesso pubblico

- Controlli relativi a crittografia, tagging, e policy pubbliche

Snippet: gestione client multiregione e controllo SQS.1

```
specific_regions = re.split(r'[,;\s]+', raw_regions) if raw_regions else []
for idx, region in enumerate(specific_regions[:6], start=1):
    self.clients[f'client_{idx}'] = boto3.client(...)
```

```
attr_response = client.get_queue_attributes(
    AttributeNames=['SqsManagedSseEnabled'],
    QueueUrl=queue_url
)
if attr_response.get('Attributes', {}).get('SqsManagedSseEnabled') == 'true':
    encrypted_queues.append(...)
else unencrypted_queues.append(...)
```

Esempio: `aws_sqs`

Controllo cifratura, tag e accesso pubblico

- Controlli relativi a crittografia, tagging, e policy pubbliche
- Integra scansione multiregione

Snippet: gestione client multiregione e controllo SQS.1

```
specific_regions = re.split(r'[,;\s]+', raw_regions) if raw_regions else []
for idx, region in enumerate(specific_regions[:6], start=1):
    self.clients[f'client_{idx}'] = boto3.client(...)
```

```
attr_response = client.get_queue_attributes(
    AttributeNames=['SqsManagedSseEnabled'],
    QueueUrl=queue_url
)
if attr_response.get('Attributes', {}).get('SqsManagedSseEnabled') == 'true':
    encrypted_queues.append(...)
else unencrypted_queues.append(...)
```

Esempio: `aws_sqs`

Controllo cifratura, tag e accesso pubblico

- Controlli relativi a crittografia, tagging, e policy pubbliche
- Integra scansione multiregione
- Ogni controllo è una funzione separata

Snippet: gestione client multiregione e controllo SQS.1

```
specific_regions = re.split(r'[,;\s]+', raw_regions) if raw_regions else []
for idx, region in enumerate(specific_regions[:6], start=1):
    self.clients[f'client_{idx}'] = boto3.client(...)
```

```
attr_response = client.get_queue_attributes(
    AttributeNames=['SqsManagedSseEnabled'],
    QueueUrl=queue_url
)
if attr_response.get('Attributes', {}).get('SqsManagedSseEnabled') == 'true':
    encrypted_queues.append(...)
else unencrypted_queues.append(...)
```

aws_vulnerability

Sonda per la gestione CVE

- Sonda custom che elenca CVE trovate da AWS Inspector

`aws_vulnerability`

Sonda per la gestione CVE

- Sonda custom che elenca CVE trovate da AWS Inspector
- Analisi di Elastic Container Registry (*ECR*), Elastic Compute Cloud (*EC2*) e *Lambda* functions

aws_vulnerability

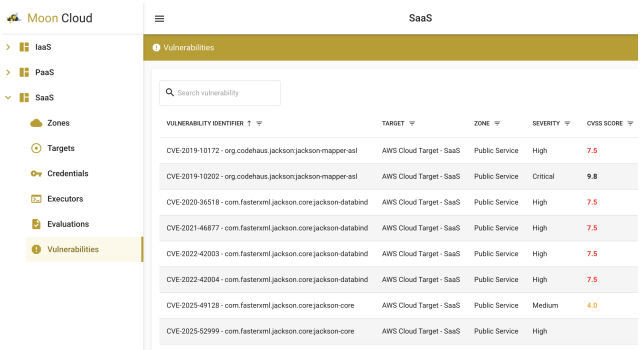
Sonda per la gestione CVE

- Sonda custom che elenca CVE trovate da AWS Inspector
- Analisi di Elastic Container Registry (*ECR*), Elastic Compute Cloud (*EC2*) e *Lambda* functions
- Consente una visione dinamica del rischio

aws_vulnerability

Sonda per la gestione CVE

- Sonda custom che elenca CVE trovate da AWS Inspector
- Analisi di Elastic Container Registry (*ECR*), Elastic Compute Cloud (*EC2*) e *Lambda* functions
- Consente una visione dinamica del rischio

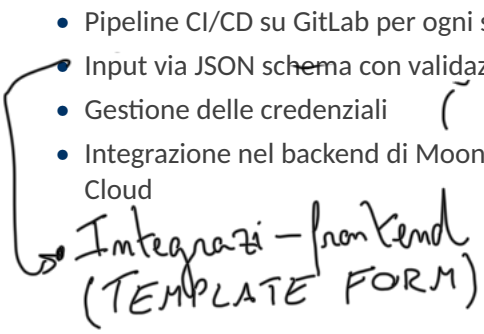


Deploy e output delle sonde

Esecuzione, integrazione e risultati

Esecuzione e integrazione

- Pipeline CI/CD su GitLab per ogni sonda
- Input via JSON schema con validazione
- Gestione delle credenziali
- Integrazione nel backend di Moon Cloud



Deploy e output delle sonde

Esecuzione, integrazione e risultati

Esecuzione e integrazione

- Pipeline CI/CD su GitLab per ogni sonda
- Input via JSON schema con validazione
- Gestione delle credenziali
- Integrazione nel backend di Moon Cloud

Risultati ottenuti

- Risultato numerico e descrittivo
- Sommario con percentuale di conformità
- Log dettagliato con eccezioni gestite
- Conformità a standard come CIS e NIST

Competenze acquisite e sviluppi futuri

Riflessioni e prospettive

Competenze acquisite

- Competenze tecniche in Python e Docker
- Analisi di documentazione tecnica
- Esperienza su progetto reale e relativo framework: *Moon Cloud*

Competenze acquisite e sviluppi futuri

Riflessioni e prospettive

~~**Competenze acquisite**~~

- Competenze tecniche in Python e Docker
- Analisi di documentazione tecnica
- Esperienza su progetto reale e relativo framework: *Moon Cloud*

Sviluppi futuri

- Estensione a nuovi benchmark e servizi AWS
- Apertura verso altri cloud provider
- Supporto multi regione

Verifiche di compliance in ambienti Cloud

Grazie per l'attenzione!