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 obiettivi

Contesto e finalità

- Evoluzione del Cloud Computing e modelli (IaaS, PaaS, SaaS)

Introduzione e obiettivi

Contesto e finalità

- Evoluzione del Cloud Computing e modelli (IaaS, PaaS, SaaS)
- AWS come contesto di riferimento

Introduzione e obiettivi

Contesto e finalità

- Evoluzione del Cloud Computing e modelli (IaaS, PaaS, SaaS)
- AWS come contesto di riferimento
- La compliance come sfida attuale

Introduzione e obiettivi

Contesto e finalità

- Evoluzione del Cloud Computing e modelli (IaaS, PaaS, SaaS)
- AWS come contesto di riferimento
- La compliance come sfida attuale
- Sviluppo di sonde per verificare la compliance in ambienti Cloud

Introduzione e obiettivi

Contesto e finalità

- Evoluzione del Cloud Computing e modelli (IaaS, PaaS, SaaS)
- AWS come contesto di riferimento
- La compliance come sfida attuale
- Sviluppo di sonde per verificare la compliance in ambienti Cloud
- Integrazione nella piattaforma Moon Cloud

Compliance e standard nel cloud

Significato, sfide e riferimenti normativi

- **Compliance:** rispetto di requisiti normativi e di sicurezza
- Importanza crescente nel contesto cloud
- Sfide legate alla gestione di risorse distribuite

Compliance e standard nel cloud

Significato, sfide e riferimenti normativi

- **Compliance:** rispetto di requisiti normativi e di sicurezza
- Importanza crescente nel contesto cloud
- Sfide legate alla gestione di risorse distribuite
- **Standard** di riferimento:
  - Center for Internet Security (*CIS*) - CIS AWS Foundations Benchmark
  - National Institute of Standards and Technology (*NIST*) - NIST SP 800-53

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()
```

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

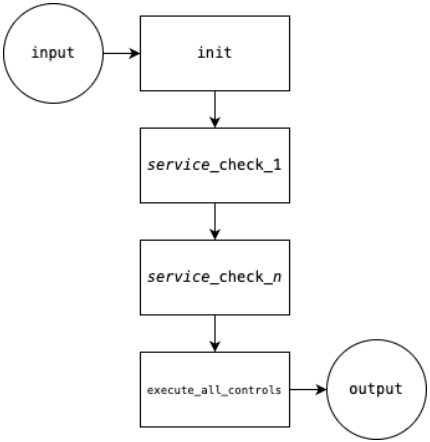
Componenti per la creazione

- 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*



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

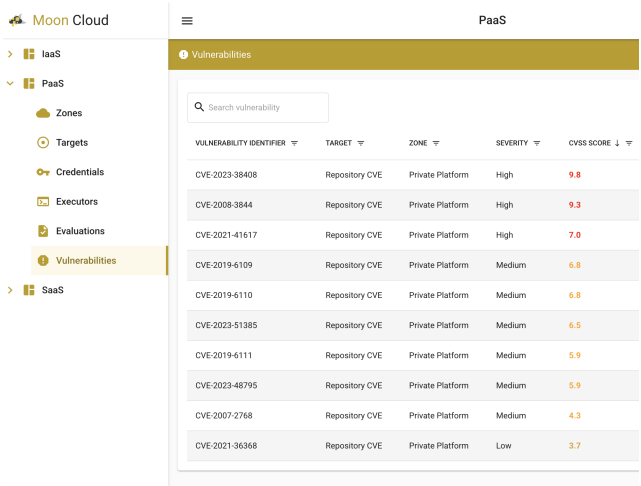
Sonda per la gestione CVE

- 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

- Controlli strutturati in blocchi
- Risultato numerico e descrittivo
- Sommario con percentuale di conformità
- Log dettagliato con eccezioni gestite

Competenze acquisite e sviluppi futuri

Riflessioni e prospettive

Competenze acquisite

- Python, AWS, Docker, GitLab CI/CD
- Analisi di documentazione tecnica
- Sviluppo modulare su un framework esistente
- Esperienza su progetto reale (Moon Cloud)

Competenze acquisite e sviluppi futuri

Riflessioni e prospettive

Competenze acquisite

- Python, AWS, Docker, GitLab CI/CD
- Analisi di documentazione tecnica
- Sviluppo modulare su un framework esistente
- Esperienza su progetto reale (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!