IAM Policy e Service Control Policies (SCPs): come gestire in modo sicuro accessi e permessi all'interno di una Landing Zone su AWS

31 Marzo 2023 - 11 min. read

AWS Identity and Access Management (IAM) | AWS Organizations | Landing Zone | multi-account strategy | Noovolari Leapp | Service Control Policies (SCPs)

Bentornati su Proud2beCloud!

Oggi vi presentiamo un'uscita speciale: direttamente dal blog di Noovolari Leapp dedicato a IAM, *Nicolò Marchesi - Nico per gli amici! :) - ci guiderà alla scoperta di uno degli aspetti-chiave per una strategia multi-account di successo su AWS.*

Vi abbiamo già parlato in questa serie dell'importanza di impostare fin da subito un ambiente AWS sicuro, dinamico, scalabile e "a prova di futuro" e di come, in questo contesto, una la Landing Zone progettata a regola d'arte giochi un ruolo fondamentale.

Nella sua implementazione, tuttavia, occorre prestare particolare attenzione ad alcuni aspetti. Uno di questi è la gestione sicura degli accessi a risorse, ambienti e account. Abbiamo chiesto a chi di Cloud Access Management se ne intende per davvero di raccontarci tutto ciò che sa sull'argomento. In particolare, oggi ci parlerà di come utilizzare le IAM Policy e le Service Control Policies (SCPs) in modo efficace per permettere alle aziende a soddisfare vincoli di security e governance.

Parola a Nico!

Ciao compagni di cloud!

Oggi approfondiremo la serie sulla **cloud landing zone** per esplorare un contesto affascinante ma spesso ignorato... le Service Control Policies (SCP)! Le SCP sono potenti, ma ci sono alcune precauzioni e trucchi per farle funzionare efficacemente con ogni diverso tipo di **policy IAM**. In questo post del blog, vedremo come interagisce l'intero ecosistema IAM e come possiamo sfruttare efficacemente gli strumenti per implementare una strategia IAM nella nostra landing zone.

C'è molto da dire, quindi iniziamo subito!

Non voglio tediarti con tutti i dettagli su AWS Organizations in modo da poter mantenere il nostro focus sul flusso IAM. Se ti serve, consulta prima uno dei migliaia di articoli sul web o controlla QUESTO che ho scritto.

Policy

Quindi, andiamo a coprire le basi (per i più pigri di voi, saltate pure a dopo il grafico, c'è un pratico riassunto in punti). Prima di passare direttamente alle interazioni, dobbiamo capire che strumenti abbiamo a disposizione nella nostra strategia IAM:

1. Le **Identity-based policies** sono il tipo di policy più comune. Sono legate agli utenti, ai gruppi e ai ruoli IAM e specificano le azioni che tali entità possono eseguire su determinati servizi AWS.
2. D'altra parte, le **Resource-based policies** sono policy direttamente associate a una risorsa AWS, come un bucket S3 o un'istanza EC2. Queste policies specificano chi ha accesso alla risorsa e quali azioni possono eseguire. Non tutti i servizi AWS le supportano (per una lista completa, controllare la pagina **AWS services that work with IAM**), e di solito vengono utilizzate per scenari molto specifici.
3. Le **Permission Boundaries (PBs)** sono policy che definiscono le autorizzazioni massime che un'entità IAM può avere. Queste politiche limitano le autorizzazioni di un'entità, garantendo che non possa eseguire azioni che superano il loro ambito autorizzato. I limiti di autorizzazione sono scritti anche nel linguaggio delle politiche AWS e possono essere allegati alle entità IAM.
4. Le **Service Control Policies (SCPs)** impongono restrizioni sugli account AWS all'interno di un'organizzazione AWS. Le SCP sono policy gerarchiche applicate all'intera Organization o a Organizational Unit (OU) specifiche. Le SCP possono essere utilizzate per limitare le azioni che un account AWS può eseguire, impedendo loro di eseguire attività al di fuori del loro perimetro autorizzato.

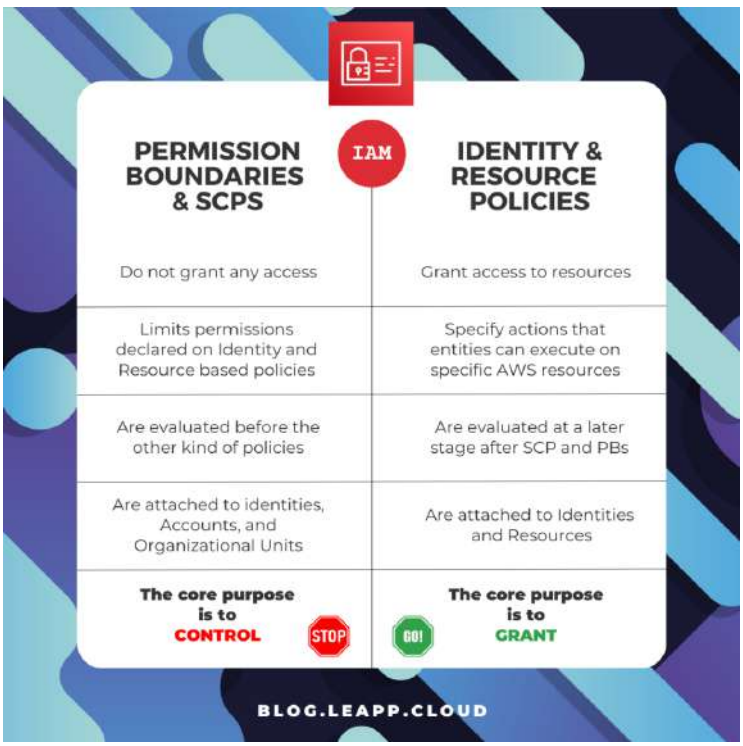
5. Le **Session Policies** sono applicate alle credenziali temporanee create dai ruoli IAM. Le Session Policies limitano le autorizzazioni di un set di credenziali temporanee, garantendo che non possano eseguire azioni che superano il loro perimetro autorizzato. Tuttavia, funzionano principalmente come PB e SCP: non concedono autorizzazioni e vengono applicate solo per la durata del token. Per semplicità, le introdurremo solo alla fine, quindi per ora, non lo considereremo.

Avvertenze

Come potreste già intuire, esiste una differenza fondamentale tra queste policy, ed è illustrata nel diagramma dall'azione che collega la policy alle effettive autorizzazioni.

NOTA: Permission Boundaries e Service Control Policies NON conferiscono ALCUNA autorizzazione.

Un'identità può accedere a una risorsa solo attraverso le policy basate sull'identità e sulle risorse. Le Permission Boundaries e le Service Control Policies possono solo limitare le autorizzazioni menzionate in precedenza. Ciò significa che abbiamo bisogno di una policy basata sull'identità o sulla risorsa che consente esplicitamente l'autorizzazione per permettere all'elaboratore di valutare positivamente la policy.



In breve, le Identity-based e le Resource-based policies definiscono chi può accedere alle risorse e quali azioni possono eseguire. Le Permission boundaries e le Service Control Policies limitano la portata di tali autorizzazioni, garantendo che le entità non possano eseguire azioni non autorizzate.

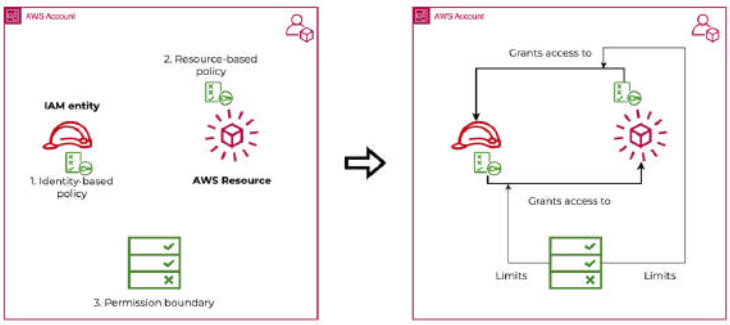
Rappresentazione visiva dell'interazione tra le policy

Vediamo ora una rappresentazione visiva delle nostre policy: iniziamo vedendo come funziona in un singolo account, e poi vediamo come le cose cambiano aggiungendo AWS Organizations all'equazione.

Sì, so che le icone differiscono dal framework di AWS, ma per favore abbiate pazienza; voglio evidenziare le differenze e il fatto che stiamo lavorando con policy fondamentalmente diverse.

Singolo account (senza Organizzazioni)

Come potete vedere, i tre elementi che possiamo utilizzare sono le Identity-based e le Resource-based policies, e le Permission boundaries:

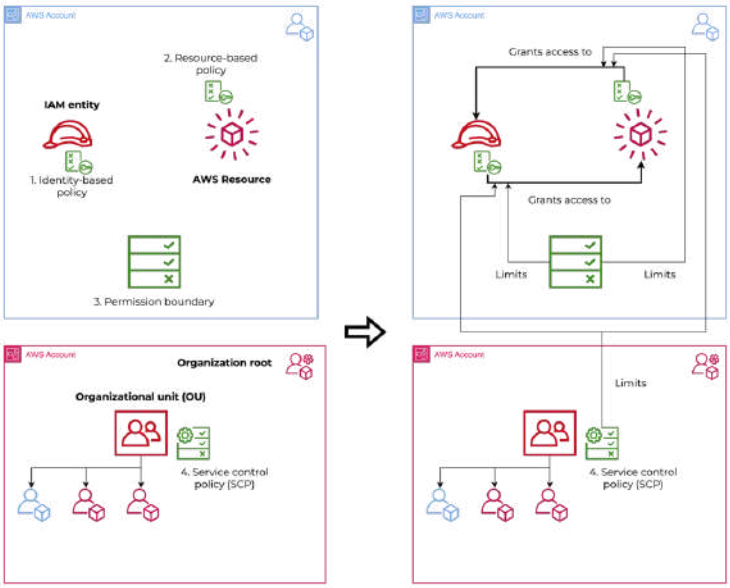


Come possiamo vedere, mentre le policy che entrano e escono dall'Identità e dalle Risorse concedono l'accesso, le Permission Boundaries limitano invece quel set di autorizzazioni.

È di fatto impossibile concedere autorizzazioni aggiuntive tramite l'uso delle Permission Boundaries.

Struttura dell'organizzazione

Quando integriamo il servizio AWS Organizations, acquisiamo la capacità di utilizzare le Service Control Policies per avere un maggiore controllo sul nostro ambiente:



Il comportamento è praticamente lo stesso delle Permission Boundaries, ma vedremo presto che ha una leggera differenza

Quindi, per riassumere brevemente:

1. **AWS Identity-based policies:**
   - Il tipo di policy più comune in AWS.
   - Collegato alle entità IAM (utenti, gruppi e ruoli)
   - Specifica le azioni che le entità possono eseguire su risorse AWS specifiche
   - *Va dalla Identity alla Resource*

2. **AWS Resource-based policies:**
   - Collegato a una risorsa AWS (ad esempio, un bucket S3 o un'istanza EC2)
   - Determina quali utenti hanno accesso alla risorsa
   - Definisce le azioni che gli utenti autorizzati possono eseguire sulla risorsa
   - Non tutti i servizi AWS li supportano
   - Usati in **scenari molto specifici**
   - *Va dalla Resource alla Identity*

3. **Permission boundaries:**
   - Collegato alle entità IAM
   - Definisce le autorizzazioni massime per un'entità IAM
   - **Limita** le autorizzazioni dell'entità al campo autorizzato

4. **Service control policies (SCPs):**
    - Policy gerarchiche per gli account AWS in un'Organizzazione
    - Usati per limitare le azioni degli account AWS
    - **Limitano** le attività al di fuori del campo delle autorizzazioni.

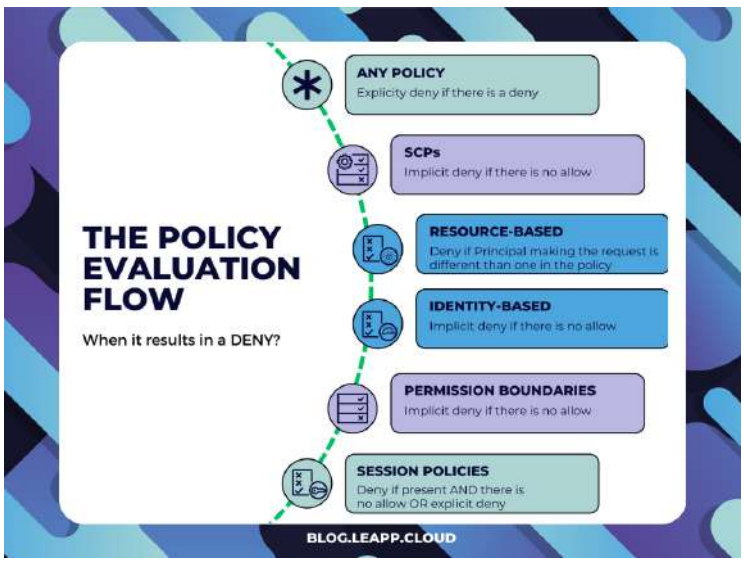
5. **Session policies:**
    - Usate con le Identity-based policy e le Permission Boundaries
    - Applicate alle credenziali temporanee dei ruoli IAM
    - Limitano le autorizzazioni delle credenziali temporanee
    - Assicura che il perimetro autorizzato non sia superato

Il flusso di valutazione

Il flusso di valutazione delle policy prevede l'uso di un motore che analizza tutte le policy viste in precedenza e determina se l'azione specifica deve essere consentita o negata.

Qui ho sintetizzato la logica per comprendere meglio il flusso decisionale:

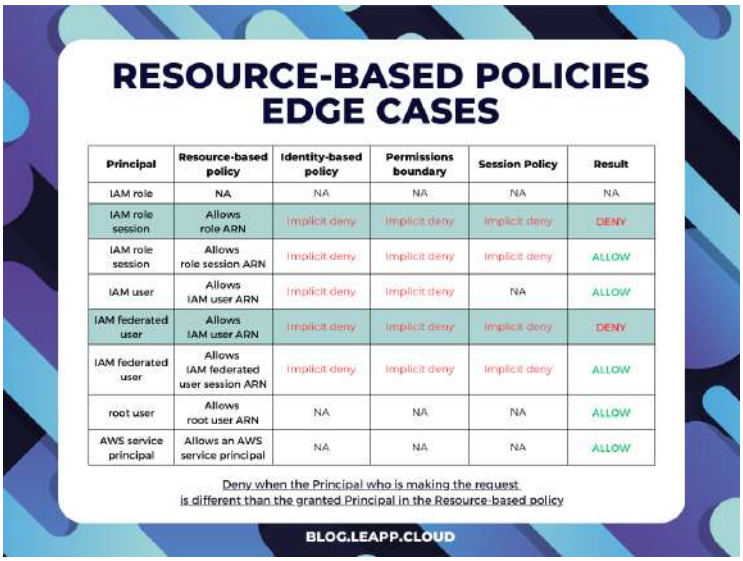
È interessante notare che le autorizzazioni basate su risorse e identità sono valutate al centro del flusso di valutazione. Attorno ad esse, vengono valutati SCP e PB, che abbiamo raggruppato in policy di controllo.



Al di la delle Resource-based policy e delle Session policies, si scopre che il flusso è piuttosto semplice; il trucco è concentrarsi sui casi limite.

Con le politiche basate sulle risorse

Le politiche basate sulle risorse comportano un rifiuto quando il Principal che effettua la richiesta è diverso dal Principal a cui concediamo l'accesso attraverso la policy basata sulle risorse. Ho evidenziato i casi interessati nello schema proposto da AWS a questo **link**.



Con le policy basate su risorse

Le Resource-based policy portano a un deny quando il Principal che effettua la richiesta è diverso dal Principal cui concediamo l'accesso attraverso la Resource-based policy.

Ho evidenziato il caso interessato nello schema proposto da AWS a questo **link**.

Con le Session policies

Anche in questo caso, è più semplice di quanto sembri. Le Session policies entrano in gioco solo quando sono presenti.

Se non le utilizzi nella tua richiesta, non dovresti preoccuparti, ma quando arriva il momento in cui le usi, devi ricordare quanto segue:

- non c'è allow → deny implicito
- non c'è deny esplicito → deny implicito

Nota a margine sull'accesso multi account

Fino ad ora, abbiamo considerato solo l'accesso nello stesso account, ma cosa succederebbe se dovessimo valutare anche l'accesso tra account diversi? È più semplice di quanto sembri: la richiesta viene valutata sulle policy e le autorizzazioni dal punto di vista di entrambi gli account e consentita solo se entrambi sono valutati positivamente!

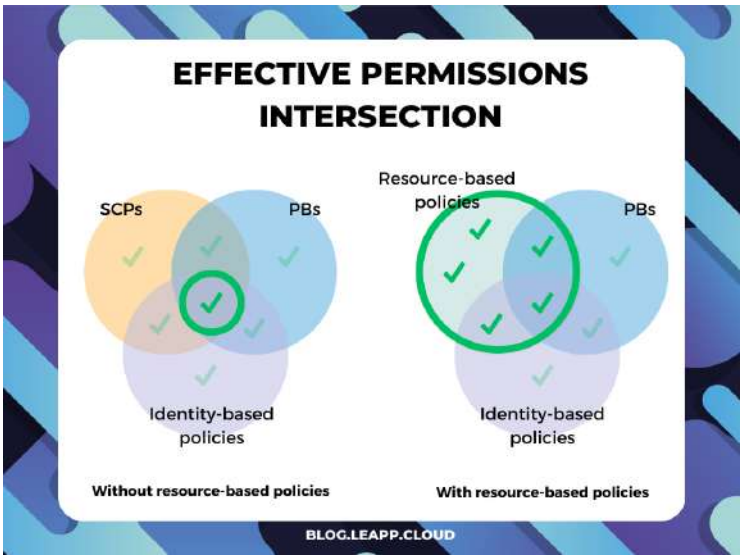
Se ci pensi, è più restrittivo dell'accesso singolo. Poiché le autorizzazioni AWS partono da un deny implicito, è necessario impostare esplicitamente le autorizzazioni su entrambi gli

account prima di valutare la richiesta come un allow.

Intersezioni di autorizzazioni

Dopo aver capito quali sono i fattori coinvolti nella decisione di permettere o negare una richiesta, possiamo passare a come interagiscono.

Ciò ha portato a due scenari pratici, uno con e uno senza policy basate su risorse. Il punto principale qui è capire che **l'unico caso in cui le autorizzazioni effettive possono superare quelle dell'intersezione complessiva è quando sono coinvolte le Resource-based policy**.



Quando sono coinvolte le Resource-based policies, se valutate come allow, si prende la decisione finale prima che vengano valutate le Identity-based policies, le Permission Boundaries e le Session policies nel flusso di valutazione della policy. Ciò comporta il rilascio di autorizzazioni che possono superare quelle esplicitamente consentite da tali politiche.

Riguardo all'ereditarietà delle SCP

L'ultima cosa da dire riguarda le Service Control Policies e il fatto che possono essere ereditate.

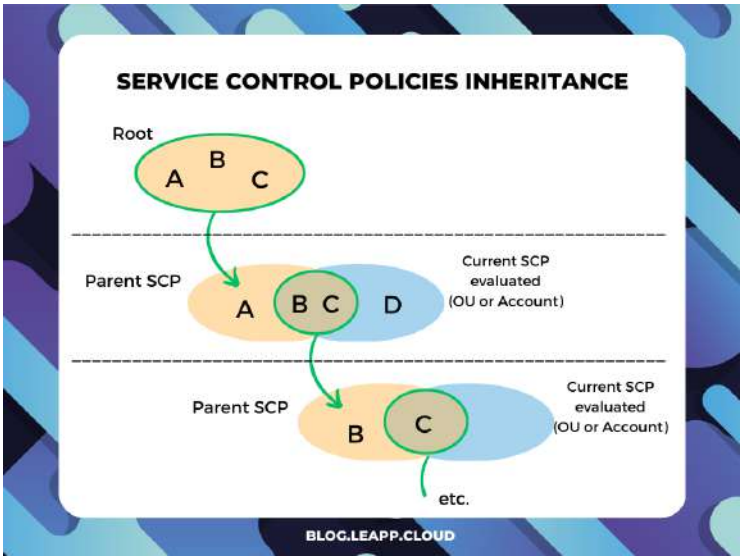
Stranamente, questo non funziona come ci si aspetterebbe (ma è meglio così, fidati, tra poco capirai il perchè).

Quando si pensa all'ereditarietà, di solito in programmazione, ma anche in altri campi, si pensa alla configurazione del genitore che viene trasmessa al figlio. Se applichiamo questo concetto alle SCP, alle unità organizzative e agli account, si potrebbe definire le SCP a livello di root e poi ereditare tutto. Beh, NON funziona così.

Poiché le dichiarazioni DENY sono valutate per prime, in pratica, **solo le dichiarazioni DENY sono ereditate**.

Se si nega un servizio in una SCP, non c'è modo di concederlo in una OU o un account di livello inferiore. Quindi, quando viene valutata una SCP, ci sono effettivamente solo due SCP che concorrono all'esito:

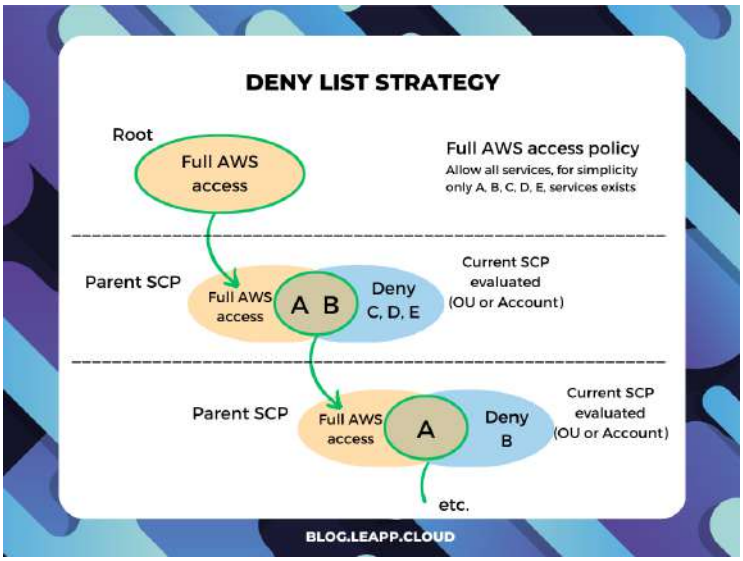
1. La SCP genitore — è la SCP valutata della SCP di livello superiore successiva; per questo è necessario navigare fino alla root dell'Organizzazione, quindi basta aggiungere il livello successivo, passo dopo passo.
2. La SCP corrente — la SCP attualmente in fase di valutazione.



Da questa prospettiva, si può notare che le dichiarazioni ALLOW non vengono ereditate ma devono essere esplicitamente impostate nella SCP dell'Account o dell'Organizational Unit. Questo accade per motivi di sicurezza, in quanto vogliamo esplicitamente definire i confini degli account e delle Organization Unit per evitare autorizzazioni implicite troppo lasche.

Strategia di deny

Con una strategia di deny, le azioni sono consentite per impostazione predefinita attraverso una SCP di FullAccess gestita direttamente da AWS. È necessario attaccare quella SCP a tutti gli account e alle OU e specificare quali servizi e azioni sono vietati:



Questo è il modo in cui AWS Organizations è configurato per impostazione predefinita in modo che gli amministratori degli account possano delegare tutti i servizi e le azioni fino a quando non si crea e si attacca una SCP che nega un servizio specifico.

Il vantaggio di questo approccio è che **gli statement deny richiedono meno manutenzione** perché non è necessario aggiornarli quando AWS aggiunge nuovi servizi, ed è supportato out-of-the-box. Inoltre, è possibile limitare l'accesso a risorse specifiche o definire le condizioni per le quali le SCP sono in vigore.

Questo è un ottimo modo per iniziare per le organizzazioni più piccole che devono agire rapidamente e non hanno requisiti rigorosi per la governance e la sicurezza.

Strategia Allow

Con la strategia Allow, devi rimuovere la SCP FullAWSAccess gestita da AWS. Ora tutte le azioni per tutti i servizi sono implicitamente negate e devi creare una SCP che esplicitamente permetta solo i servizi e le azioni che vuoi consentire. Questo caso è lo stesso schema della valutazione dell'ereditarietà.

Il vantaggio di questo approccio è che **hai più controllo su ciò che è consentito**. Poiché tutto viene implicitamente negato, questo modo è più facile da ridurre lo scope dei confini effettivi di un account. Poiché i permessi Deny sono ereditati, non è necessario specificarli ovunque.

Tuttavia, richiede più manutenzione poiché devi consentire manualmente ogni servizio (e questo include i nuovi servizi). E ci sono alcune limitazioni sugli statement Allow: gli elementi delle risorse possono avere solo un'entrata "*" e non possono avere una Condition.

Il valore di questo approccio si può notare quando una grande parte dell'organizzazione potrebbe non avere bisogno di tutti i servizi ma potrebbe avere alcuni servizi richiesti da pochi per casi d'uso specifici. In questo scenario, è più semplice aumentare i permessi e soddisfare i requisiti di sicurezza e governance consentendo allo stesso tempo flessibilità ed eccezioni.

Solo un paio di esempi

Vediamo ora queste SCP in azione, ecco alcuni esempi per poter immaginare meglio come appare una SCP in un caso di utilizzo concreto

Pipeline only account

In questo caso, abbiamo creato una SCP per negare tutto tranne le modifiche che passano attraverso le pipeline. Il caso d'uso è quello di creare un account solo per l'automazione in cui non è consentita alcuna azione manuale, ma le modifiche vengono sempre distribuite tramite pipeline.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAllExceptPipelines",
      "Effect": "Deny",
      "NotAction": [
        "codepipeline:*",
        "codebuild:*",
        "codecommit:*",
        "codedeploy:*",
        "codestar:*",
        "cloudformation:*",
        "iam:*",
        "s3:*",
        "logs:*",
        "cloudwatch:*",
        "cloudtrail:*",
        "codestar:*",
        "codestar-notification:*",
        "codeartifact:*",
        "kms:*",
```

```
        "tag:*",
        "access-analyzer:*",
        "codestar-connections:*",
        "ssm:GetParameter*",
        "sts:*",
        "events:*"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotLike": {
          "aws:PrincipalArn": [
            "arn:aws:iam::MY-ACCOUNT-ID:role/MY-ROLE"
          ]
        }
      }
    }
  ]
}
```

Backup protection

In questo caso, abbiamo creato una SCP per proteggere tutti i backup effettuati tramite AWS Backup dalla cancellazione. Si noti che sia S3 che i Backup vault sono protetti e il servizio non può essere disattivato.

```
{
        "Version": "2012-10-17",
        "Statement": [
                {
                        "Sid": "DenyS3BackupDelete",
                        "Action": [
                                "s3:DeleteObject",
                                "s3:DeleteObjectVersion",
                                "s3:DeleteObjectTagging",
                                "s3:DeleteBucket"
                        ],
                        "Resource": [
                                "arn:aws:s3:::MY-S3-BACKUP*/*"
```

```
                        ],
                        "Effect": "Deny"
                },
                {
                        "Sid": "DenyBackupDelete",
                        "Action": [
                                "backup:DeleteBackupVault",
                                "backup:DeleteBackupVaultAccessPolicy",
                                "backup:PutBackupVaultAccessPolicy"
                        ],
                        "Resource": [
                                "arn:aws:backup:*:*:backup-vault:MY-BACKU
P-VAULT"
                        ],
                        "Effect": "Deny",
                        "Condition": {
                                "StringNotLike": {
                                        "aws:PrincipalARN": "arn:aws:ia
m::*:role/MY-EXECUTION-ROLE"
                                }
                        }
                },
                {
                        "Sid": "DenyBackupTurnoffService",
                        "Action": [
                                "backup:UpdateRegionSettings"
                        ],
                        "Resource": [
                                "*"
                        ],
                        "Effect": "Deny",
                        "Condition": {
                                "StringNotLike": {
                                        "aws:PrincipalARN": [
                                                "arn:aws:iam::*:role/MY-E
XECUTION-ROLE"
                                        ]
```

```
                                    }
                                }
                            }
                        ]
}
```

Conclusioni

Congratulazioni, sei riuscito ad arrivare alla fine del post del blog! Siamo passati dal flusso di valutazione delle policy all'effettiva implementazione degli SCP, passando attraverso la comprensione di tutti i dettagli che concorrono all'interazione tra policy e limiti.

Dai vari esempi di SCP che abbiamo visto, come per qualsiasi strumento di governance, la loro implementazione è strettamente legata alla struttura e al funzionamento di ogni azienda. Pertanto, credo che questa conoscenza debba essere ben interiorizzata e ampiamente compresa all'interno di ciascuna organizzazione.

Ora sei sulla strada per diventare un esperto; ci vediamo la prossima volta e fammi sapere come sta andando il tuo viaggio nel cloud!

Com'è andato il vostro viaggio nelle Service Control Policy? Fateci sapere nei commenti!

È il momento di ringraziare Nico e il resto del team di Noovolari Leapp.

Per saperne di più sul progetto open-source, visitate il GitHub repository (e lasciate una star se vi piace!), oppure approfondite sul sito o sul blog.

A presto su **Proud2beCloud**!

Proud2beCloud

Dal 2011 beSharp guida le aziende italiane sul Cloud.Dalla piccola impresa alla grande multinazionale, dal manifatturiero al terziario avanzato, aiutiamo le realtà più all'avanguardia a realizzare progetti innovativi in campo IT.
