

Costi di Data Transfer su AWS: facciamo i conti! Dalle “pillole” a 3 scenari reali

14 Gennaio 2026 - 9 min. read

FinOps

Nell'articolo precedente: "[Guida ai costi di Data Transfer su AWS in pillole](#)" abbiamo esplorato in dettaglio i costi di network su AWS: data transfer in ingresso e uscita, traffico tra Region e Availability Zone, VPC peering, NAT Gateway e VPC Endpoints. La teoria è chiara.

Ma come si traduce nella pratica?

È il momento di passare dall'astrazione alla realtà. In questo articolo analizzeremo **tre scenari reali di complessità crescente**, dimostrando come una corretta analisi dei costi di network possa portare a risparmi significativi e decisioni architetturali più consapevoli.

Preparate i vostri Jupyter Notebook: è tempo di fare i conti.

Caso 1: MongoDB Atlas - Public Endpoint vs VPC Peering

Il contesto

Un'applicazione ospitata su AWS utilizza MongoDB Atlas come database. Attualmente l'accesso avviene tramite endpoint pubblici, ma ci viene chiesto di valutare se convenga investire tempo per configurare un VPC Peering verso la VPC messa a disposizione da MongoDB.

La differenza sostanziale tra le due soluzioni risiede nel percorso di rete:

- **Public Endpoint:** il traffico esce dalla VPC attraverso un NAT Gateway, raggiunge MongoDB Atlas su internet, e incorre nei relativi costi
- **VPC Peering:** il traffico rimane privato, viaggia sul peering connection di AWS, ed evita i costi di NAT Gateway

Analisi dei costi

Consideriamo un'istanza MongoDB M10 (57\$/mese) e analizziamo i costi di network per entrambe le soluzioni.

Soluzione A - Public Endpoint

I costi includono:

- Cluster M10: \$57/mese
- NAT Gateway: \$0.048/h
- NAT Gateway data processing: \$0.048/GB
- Data transfer OUT: \$0.09/GB

La formula del costo totale è:

$$\text{Cost}_{\text{SaaS-M10}} = \text{Cluster}_{\text{M10}} + \text{NAT}_{\text{hourly}} \times 730 + V \times (\text{NAT}_{\text{processing}} + \text{Internet}_{\text{out}})$$

dove V rappresenta il volume di traffico mensile (V assumerà lo stesso significato anche per le successive formule).

Soluzione B - VPC Peering

MongoDB Atlas non applica costi per il VPC peering. I costi sono limitati al traffico AWS:

- Cluster M10: \$57/mese
- VPC Peering same-AZ: gratuito
- VPC Peering cross-AZ: \$0.01/GB (in entrambe le direzioni)

Qui sorge una complicazione: le Availability Zone non sono allineate tra account AWS diversi. L'identificatore *eu-west-1a* nel nostro account non corrisponde necessariamente a *eu-west-1a* nell'account MongoDB. Senza poter garantire il posizionamento same-AZ, dobbiamo assumere traffico cross-AZ.

La formula diventa:

$$\text{Cost}_{\text{Peering-M10}} = \text{Cluster}_{\text{M10}} + V \times (\text{CrossAZ}_{\text{in}} + \text{CrossAZ}_{\text{out}})$$

Nota: il traffico cross-AZ costa \$0.01/GB in entrambe le direzioni (ingresso e uscita dall'AZ), per un totale di \$0.02/GB sul peering.

Confronto e Risultati

Calcoliamo la differenza di costo:

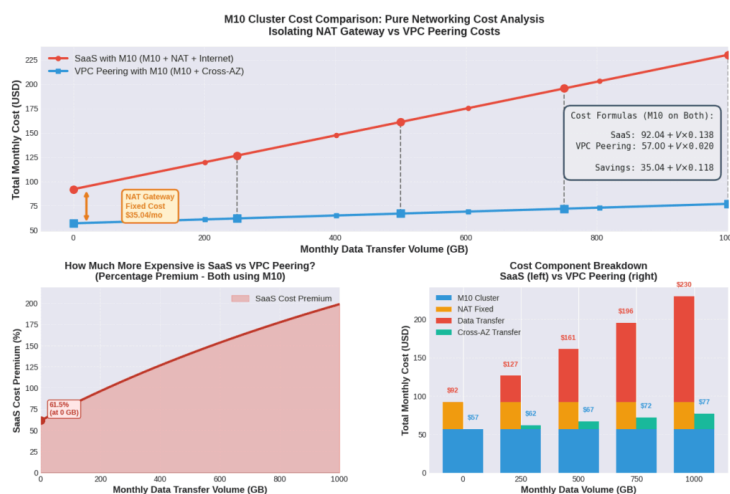
$$\Delta \text{Cost}_{\text{M10}}(V) = \text{Cost}_{\text{SaaS-M10}} - \text{Cost}_{\text{Peering-M10}}$$

$$\Delta \text{Cost}_{\text{M10}}(V) = [\text{Cluster}_{\text{M10}} + \text{NAT}_{\text{hourly}} \times 730 + V \times (\text{NAT}_{\text{processing}} + \text{Internet}_{\text{out}})] - [\text{Cluster}_{\text{M10}} + V \times (\text{CrossAZ}_{\text{in}} + \text{CrossAZ}_{\text{out}})]$$

$$\Delta \text{Cost}_{\text{M10}}(V) = \text{NAT}_{\text{hourly}} \times 730 + V \times [(\text{NAT}_{\text{processing}} + \text{Internet}_{\text{out}}) - (\text{CrossAZ}_{\text{in}} + \text{CrossAZ}_{\text{out}})]$$

$$\Delta \text{Cost}_{\text{M10}}(V) = \$35.04 + V \times 0.118$$

Il risultato è sempre positivo: il VPC peering genera un risparmio in ogni scenario di traffico.



Il grafico evidenzia due aspetti fondamentali:

1. **Risparmio base fisso:** il solo costo del NAT Gateway (\$32.85/mese) rappresenta un risparmio immediato, indipendentemente dal volume di traffico
2. **Risparmio marginale:** ogni GB trasferito costa \$0.118 in meno con il peering

Anche con volumi di traffico molto elevati (oltre 10TB/mese), il costo del data transfer cross-AZ sul peering non raggiunge nemmeno il costo del solo NAT Gateway della soluzione pubblica.

Conclusioni

Il VPC peering verso MongoDB Atlas è conveniente dal punto di vista economico in qualsiasi scenario di traffico. Ai benefici economici si aggiungono:

- **Sicurezza:** il traffico rimane nella rete privata AWS
- **Performance:** latenza inferiore e throughput più stabile rispetto alla rete pubblica
- **Semplicità:** eliminazione di un NAT Gateway e relativa gestione

Vale la pena notare che, pur assumendo traffico cross-AZ (scenario peggiore), esiste una probabilità statistica di ottenere un matching same-AZ, che renderebbe il risparmio ancora maggiore grazie all'assenza di costi sul peering.

Caso 2: AWS S3 Backup - Private vs Public Connectivity

Il contesto

Un cliente ci commissiona un'architettura per effettuare backup di server on-premise su S3. I requisiti forniti sono:

1. "Mantenere il traffico privato passando dal Direct Connect"
2. "Contenere i costi"

Il cliente dispone già di una connessione Direct Connect verso AWS con un Transit Gateway configurato.

Mettere in discussione i requisiti

Come sempre in architettura, è fondamentale validare i requisiti prima di accettarli acriticamente.

Comprendiamo il ragionamento: avendo già investito in un Direct Connect, sembra logico utilizzarlo per ogni caso d'uso. Tuttavia, per il traffico verso S3, AWS fornisce connettività pubblica ottimizzata e, soprattutto, **il data transfer IN verso AWS è gratuito**.

La domanda diventa: quanto costa realmente questa "privatezza" del traffico?

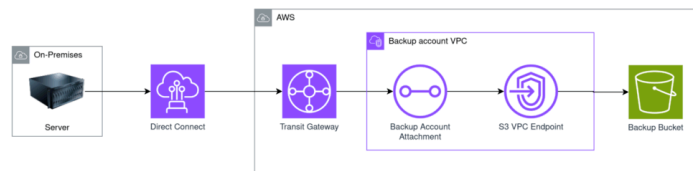
Analisi dei costi

In questo caso la nostra variabile sarà il volume in GB di cui effettuare mensilmente il backup su S3 Standard.

Soluzione A - Traffico Privato via Direct Connect

Il percorso del traffico è:

On-premise → Direct Connect → Transit Gateway → VPC → VPC Endpoint (S3) → S3



I costi includono:

- Direct Connect: \$0.30/h (già esistente)
- Transit Gateway attachment: \$0.05/h
- Transit Gateway data processing: \$0.02/GB
- VPC Endpoint per S3: gratuito
- S3 storage: \$0.023/GB-mese
- S3 API calls: costo trascurabile ai fini del confronto in quanto identico in entrambe le soluzioni

Di conseguenza il costo totale dell'opzione A è:

$$\text{CostPrivate} = \text{DCPortHours} + \text{TGAttachmenthourly} \times 730 + (\text{V} \times \text{TGprocessing}) + \text{S3Storage} + \text{S3API}$$

$$\text{Cost}_{\text{Private}} = \$219 + \$36.50 + (V \times \$0.02) + \text{S3Storage} + \text{API Costs}$$

$$\text{Cost}_{\text{Private}} = \$255.50 + (V \times \$0.02) + \text{S3Storage} + \text{API Costs}$$

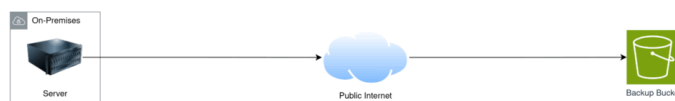
In realtà, se il Direct Connect è già presente, è possibile escluderlo dal costo totale:

$$\text{Cost}_{\text{Private}} = \$36.50 + (V \times \$0.02) + \text{S3Storage} + \text{API Costs}$$

Soluzione B - Traffico Pubblico

Il percorso è semplicemente:

On-premise → Internet → S3



I costi sono:

- Data IN verso AWS: gratuito
- S3 storage: \$0.023/GB-mese
- S3 API calls: costo trascurabile ai fini del confronto in quanto identico in entrambe le soluzioni

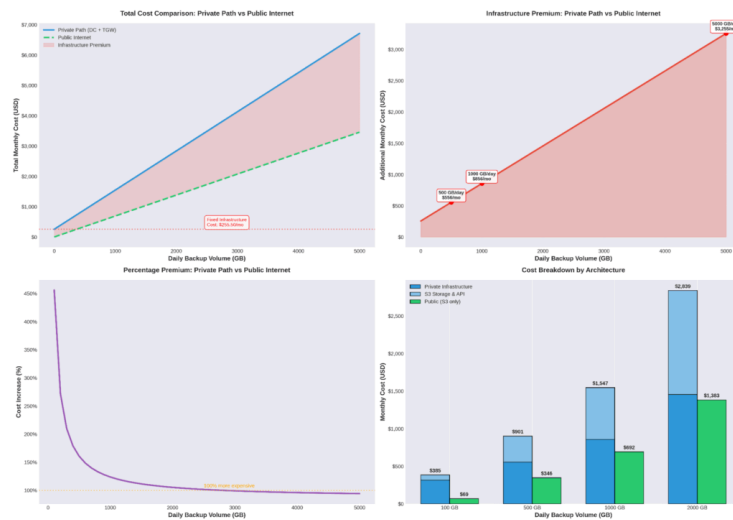
$$\text{Cost}_{\text{Public}} = \text{S3Storage} + \text{S3API}$$

Confronto e Risultati

A questo punto risulta molto semplice fare la differenza tra i due costi:

$$\Delta \text{Cost}(V) = \text{Cost}_{\text{Private}} - \text{Cost}_{\text{Public}}$$

$$\Delta \text{Cost}(V) = \$36.50 + (V \times \$0.02)$$



L'analisi grafica rivela un quadro inequivocabile:

1. **Per volumi ridotti** (centinaia di GB): la soluzione privata costa diverse volte in più rispetto alla soluzione pubblica. Il costo fisso del Transit Gateway attachment (\$35/mese) domina la spesa totale.
2. **Per volumi elevati** (decine di TB): il costo della soluzione privata si stabilizza a circa il doppio della soluzione pubblica. La componente variabile (\$0.02/GB del Transit Gateway processing) rappresenta un overhead costante.
3. **Non esiste un break-even point:** la soluzione pubblica è sempre più conveniente, indipendentemente dal volume di dati.

Considerazioni aggiuntive

- Alternative non analizzate in dettaglio:

Per completezza, menzioniamo che esiste una terza opzione: utilizzare un AWS Direct Connect Public VIF invece del Private VIF con Transit Gateway. Questa soluzione mantiene il traffico sulla connessione Direct Connect dedicata (evitando il transito su internet pubblico) ma elimina la necessità del Transit Gateway, riducendo significativamente i costi. L'analisi dettagliata di questa opzione esula dagli obiettivi di questo articolo, ma rappresenta un'alternativa interessante per contesti specifici.

- Quando la soluzione privata ha senso:

L'analisi economica non è l'unico fattore decisionale. La soluzione privata può essere giustificata in presenza di:

- Requisiti di compliance che impongono traffico privato
- Politiche di sicurezza aziendali stringenti

- Limitazioni della banda internet disponibile on-premise
- Necessità di QoS garantite

Conclusioni

Dal punto di vista puramente economico, la soluzione pubblica è superiore in ogni scenario. Il risparmio varia da un fattore 5-10x per piccoli volumi a un fattore 2x per grandi volumi.

Caso 3: NAT Gateway - Centralizzati o Distribuiti?

Il contesto

Un'organizzazione AWS con decine di account si trova a gestire un elevato numero di NAT Gateway distribuiti. Ogni account ha una VPC con tre NAT Gateway (uno per Availability Zone), generando costi mensili significativi anche in assenza di traffico.

La domanda sorge spontanea: ha senso mantenere NAT Gateway in ogni account, o conviene centralizzare l'egress internet in un account dedicato, raggiungibile dagli altri account tramite Transit Gateway?

Premessa importante

Questa decisione architetturale ha implicazioni che vanno oltre il puro aspetto economico:

- **Traffic inspection:** centralizzare permette di implementare controlli di sicurezza centralizzati
- **IP addressing:** consente di utilizzare IP pubblici di proprietà aziendale per tutto il traffico in uscita
- **Governance:** facilita audit e compliance centralizzando il punto di egress
- **Blast radius:** un problema al NAT Gateway centralizzato impatta tutti gli account

In questo articolo ci concentreremo esclusivamente sull'analisi economica, ma è fondamentale considerare questi aspetti in una valutazione reale.

Analisi dei costi

Consideriamo un'organizzazione con N account AWS, ognuno con una VPC e tre NAT Gateway (uno per AZ).

Soluzione A - NAT Gateway Distribuiti

Ogni account ha:

- 3 NAT Gateway: $3 \times \$0.048/\text{h}$
- NAT Gateway data processing: $\$0.048/\text{GB}$
- Data transfer OUT: $\$0.09/\text{GB}$

$$\text{CostDistributed} = N \times (\text{NAT}_{\text{per_AZ}} \times \text{AZCount} \times \text{NAT}_{\text{hourly}} \times 730 + \text{V}_{\text{per_account}} \times \text{NAT}_{\text{processing}}) + V \times \text{Internet}_{\text{out}}$$

Sostituendo alcune variabili arriviamo a:

$$\text{CostDistributed} = N \times (3 \times \$0.048 \times 730 + \text{V}_{\text{per_account}} \times \$0.048) + V \times \text{Internet}_{\text{out}}$$
$$\text{CostDistributed} = N \times (\$105.12 + \text{V}_{\text{per_account}} \times \$0.048) + \text{InternetEgressCost}$$

Lasciamo il costo del traffico in uscita espresso come variabile, in quanto sarà presente anche nel secondo caso, di conseguenza non è necessario calcolarlo ai fini del confronto.

Soluzione B - NAT Gateway Centralizzati

Architettura:

VPC spoke → Transit Gateway → VPC egress → NAT Gateway → Internet

Costi coinvolti:

- Transit Gateway attachments: $\$0.05/\text{h} \times (N \text{ spoke} + 1 \text{ egress}) \text{ account}$
- NAT Gateway centralizzati: $3 \times \$0.048/\text{h}$
- Transit Gateway data processing: $\$0.02/\text{GB}$
- NAT Gateway data processing: $\$0.048/\text{GB}$
- Data transfer OUT: $\$0.09/\text{GB}$

$$\text{CostCentralized} = (N + 1) \times \text{TGAttachment}_{\text{hourly}} \times 730 + \text{NAT}_{\text{count}} \times$$

$$\text{NAT}_{\text{hourly}} \times 730 + V \times (\text{TG}_{\text{processing}} + \text{NAT}_{\text{processing}} + \text{Internet}_{\text{out}})$$

Sostituendo alcune variabili arriviamo a:

$$\text{Cost}_{\text{Centralized}} = (N + 1) \times \$0.05 \times 730 + 3 \times \$0.048 \times 730 + V \times (\$0.02 + \$0.048) + \text{InternetEgressCost}$$

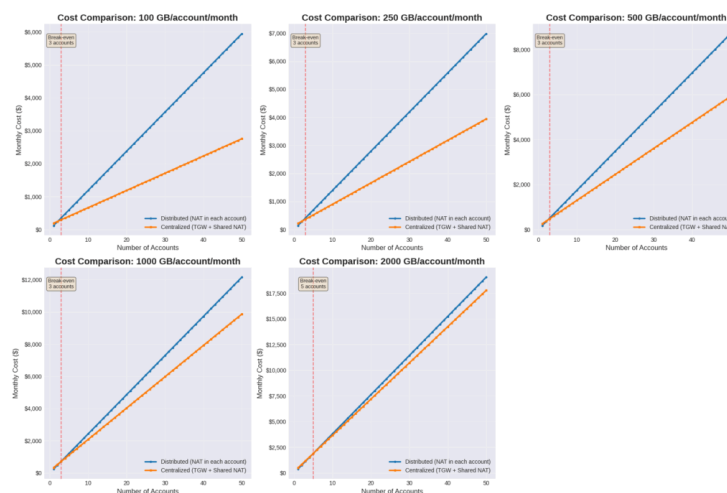
$$\text{Cost}_{\text{Centralized}} = (N + 1) \times \$36.50 + \$105.12 + V \times \$0.068 + \text{InternetEgressCost}$$

$$\text{Cost}_{\text{Centralized}} = (\$36.50 + \$105.12) + N \times \$36.50 + V \times \$0.068 + \text{InternetEgressCost}$$

$$\text{Cost}_{\text{Centralized}} = \$141.62 + N \times \$36.50 + V \times \$0.068 + \text{InternetEgressCost}$$

Analisi Bidimensionale

Ci troviamo di fronte a una funzione con due variabili: traffico e numero di account. Dobbiamo analizzare entrambe.

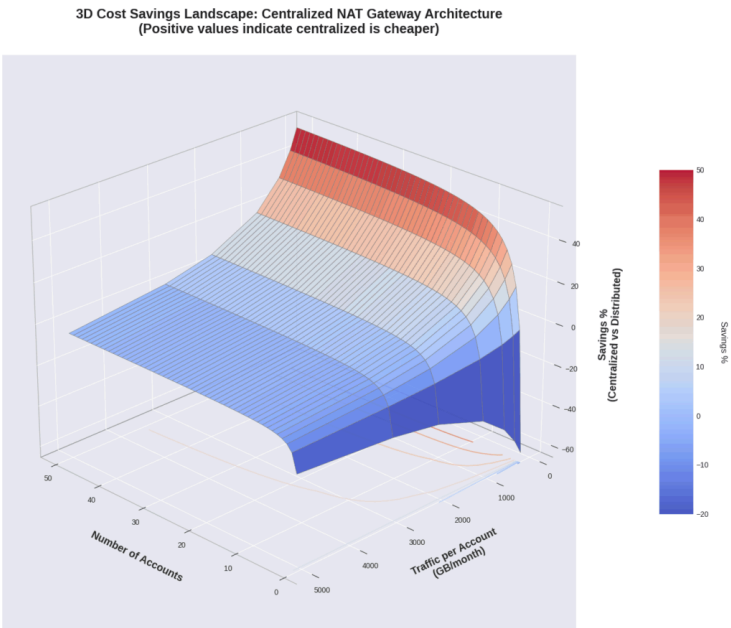


Fissando il volume di traffico a diversi livelli (1TB, 5TB, 10TB, 50TB, 100TB), osserviamo che:

1. **Per bassi volumi di traffico:** il break-even si raggiunge con pochissimi account. Da quel punto in poi, la centralizzazione genera risparmi crescenti.
2. **Per volumi elevati:** il break-even si sposta leggermente più avanti a causa del costo aggiuntivo di \$0.02/GB del Transit Gateway processing.
3. **Risparmio massimo:** si ottiene con molti account e poco traffico per account.

Grafico 2 - Visualizzazione Tridimensionale

Se vogliamo invece evitare di fissare la variabile del traffico, un grafico in due dimensioni non basta più, dobbiamo quindi passare da una retta ad un piano. Il grafico tridimensionale offre una visione completa dell’andamento dei costi al variare di entrambe le variabili simultaneamente.

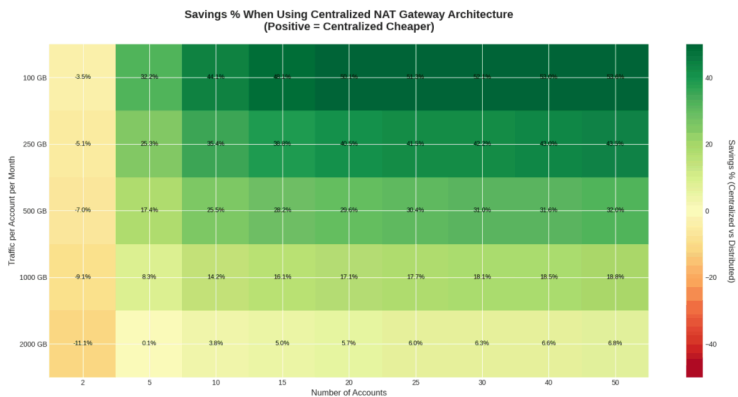


Dal grafico tridimensionale si delinea lo stesso andamento evidenziato in precedenza. Il risparmio massimo si ottiene con un elevato numero di account e basse quantità di traffico.

Questo grafico rende più evidente un'altra informazione interessante: sebbene altissime quantità di traffico possano rendere la centralizzazione dei NAT Gateway poco vantaggiosa in termini economici, a partire dalla decina di account è quasi impossibile trovarsi in questa situazione.

Grafico 3 - Semplificazione per Leggibilità

Il grafico tridimensionale, pur completo, può risultare di difficile lettura. Per semplificare la visualizzazione, abbiamo creato un grafico bidimensionale che mostra la differenza di costo (zona verde = risparmio, zona rossa = costo aggiuntivo) in funzione di account e traffico.



Conclusioni

L'analisi economica indica chiaramente che la centralizzazione dei NAT Gateway è conveniente nella maggioranza degli scenari reali, in particolare:

Centralizzare è conveniente quando:

- Si hanno 5+ account nell'organizzazione
- Il traffico per account è moderato (<5TB/account/mese)
- Si desidera ottenere risparmi crescenti con la scala

La distribuzione può essere preferibile nelle rare situazioni in cui:

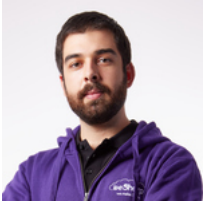
- Si hanno pochissimi account (<3-4)
- Il traffico per account è estremamente elevato (>50TB/account/mese)
- Esistono requisiti di isolation o blast radius molto stringenti

Conclusioni generali

Attraverso questi tre casi reali abbiamo mostrato che l'analisi accurata dei costi di network AWS può portare a risparmi significativi e decisioni architetturali più consapevoli. La lezione principale è chiara: mettere in discussione i requisiti iniziali e validarli con dati concreti, anche quando sembrano derivare da scelte logiche o investimenti già effettuati. I costi fissi come NAT Gateway e Transit Gateway hanno un impatto enorme per workload con traffico limitato, mentre la scala dell'organizzazione può ribaltare completamente le convenienze economiche. Il tempo investito nell'ottimizzazione iniziale viene tipicamente ripagato in pochi mesi di operatività.

About Proud2beCloud

Proud2beCloud è il blog di **beSharp**, APN Premier Consulting Partner italiano esperto nella progettazione, implementazione e gestione di infrastrutture Cloud complesse e servizi AWS avanzati. Prima di essere scrittori, siamo Solutions Architect che, dal 2007, lavorano quotidianamente con i servizi AWS. Siamo innovatori alla costante ricerca della soluzione più all'avanguardia per noi e per i nostri clienti. Su Proud2beCloud condividiamo regolarmente i nostri migliori spunti con chi come noi, per lavoro o per passione, lavora con il Cloud di AWS. Partecipa alla discussione!



Andrea Pusineri

DevOps Engineer @ beSharp. Mi diverto a risolvere problemi e sono cintura nera nel trovarli. Linux enthusiast e security guy wannabe, mi piacciono le CTF e nel tempo libero sono un avido lettore di fumetti/manga/libri. btw I use Arch

Copyright © 2011-2026 by beSharp spa - P.IVA IT02415160189