Load Balancing like a pro: configurazioni avanzate di AWS Elastic Load Balancer (ELB)

12 Agosto 2022 - 8 min. read

Advanced Networking | *Application Load Balancer (ALB)* | *AWS Elastic Load Balancer (ELB)* | *Network Load Balancer (NLB)*

Una vecchia pubblicità recitava: "Two is megl che one".

Soprattutto quando, come diciamo sempre, "Tutto può fallire, in qualsiasi momento". Per rendere il business resiliente ai fallimenti, è necessario fare in modo che applicazioni ed infrastrutture siano fault tolerant, ad esempio utilizzando più istanze applicative.

Il **Load balancing** è un componente fondamentale che può permetterci di avere migliori uptime e rendere le applicazioni sempre disponibili: ridistribuire il traffico su istanze differenti in auto scaling (e controllandone il funzionamento corretto) non è così semplice come sembra.

Nel mio passato da system administrator ho sempre avuto difficoltà nel trovare una soluzione ridondante, resiliente e scalabile. Solo dopo tanto lavoro ed automazioni sono riuscito a passare notti di sonno tranquille !

Usando i servizi gestiti, come sempre, ci aiuta a ridurre la quantità di lavoro necessaria per raggiungere i nostri scopi.

Quando si parla di load balaincing, AWS mette a disposizione servizi molto flessibili. Sotto al cappello "Elastic Load Balancing (ELB)" ci sono molte opzioni disponibili.

In questo articolo vedremo in breve i concetti fondamentali legati ai load balancer e andremo ad analizzare alcuni casi d'uso non molto comuni.

Concetti Base

AWS mette a disposizione tre differenti tipologie di load balancer: **Application, Network** e **Gateway.**

In questo articolo ci focalizzeremo su **Application Load Balancer** e **Network Load Balancer**. Non descriveremo invece il Classic Load balancer perché si tratta di un mix di application e network con qualche feature mancante (potete trovare qui la tabella comparativa). Al Gateway load balancer, invece, dedicheremo prestoun articolo dedicato.

Ogni tipo di ELB può estendersi in più Availability Zones e le sue componenti fondamentali sono tre: **listener**, **target group** ed **health checks**.

Gli ELB possono essere pubblici (internet facing) o privati (accessibilisolo dalle risorse interne private).

Un **listener** è una piccola porzione della configurazione dell'ELB che definisce il punto di ingresso del traffico. Se la nostra applicazione deve essere disponibile usando il protocollo HTTP sulla porta 8080 allora la configurazione del listener sarà sulla stessa porta e protocollo.

Un **target group** è l'insieme delle risorse computazionali che possono ricevere il traffico distribuito dal load balancer.

Può essere composto da istanze EC2, container ECS, indirizzi IP, funzioni lambda ed anche un altro application load balancer.

N.B: non tutti i tipi di ELB possono usare tutti i target group. Ad esempio, solo un Application Load Balancer può avere funzioni lambda come target.
Un **health check** è il test eseguito dal target group per determinare lo stato di salute dell'istanza. Quelle non funzionanti sono automaticamente escluse e non ricevono traffico Ad esempio se la nostra applicazione riceve traffico TCP sulla porta 31337, l'health check verifica che il servizio sia disponibile su quella porta.

Application Load Balancer

L'application Load Balancer (ALB) opera al livello 7 ISO/OSI, da qui il nome.

È il tipo di load balancer più utilizzato perché è in grado di bilanciare traffico HTTP ed HTTPS, ed è in grado di fare **redirezioni**, **autenticazione** e **offloading SSL** utilizzati certificati rilasciati da Amazon Certificate Manager (ACM).

Network Load Balancer

Il Network Load Balancer (NLB) opera al livello 4 dello stack ISO/OSI (network). Può gestire tutto il traffico basato su TCP o UDP senza una specializzazione specifica per alcun protocollo. Supporta anche l'offload TLS e offre IP statici (sia per load balancer interni che esterni).

Gateway Load Balancer

Il Gateway Load Balancer è l'ultimo aggiunto alla famiglia di servizi ELB. È in grado di gestire il traffico a livello 3 usando il protocollo GENEVE per incapsulare il i pacchetti. Questo rende possibile l'utilizzo di appliance per la sicurezza ed il networking custom o fornite da terze parti.

Per implementare una soluzione IDS custom in alta affidabilità (quindi usando diverse Availability Zones) è necessario usare un Gateway Load Balancer, che si occuperà proprio di distribuire il traffico IP, senza dover specificare porta o protocollo

Dopo questo breve cappello introduttivo non ci resta che tuffarci nell'argomento analizzando qualche configurazione non proprio comune per il setup avanzato dei load balancer su AWS.

Usare un ALB per mettere in sicurezza WordPress

Portare le applicazioni in Cloud facendone il refactoring o adattandole non è sempre una strada percorribile quando il tempo stringe.

WordPress è un esempio comune di applicazione che non è semplice migrare (se il tempo non è un problema tenete presente la possibilità di usare container.

Per migrare un sito WordPress è necessario mantenere lo stesso nome DNS e la connessione in HTTPS. Una soluzione “quick and dirty” potrebbe essere usare una istanza Amazon EC2 pubblica; esporre però una installazione WordPress senza usare un WAF può essere un rischio per la sicurezza.

Mettere un Application Load Balancer di fronte all'istanza Wordpress aiuta ad aumentare la sicurezza e ridurre il lavoro necessario alla manutenzione, anche se non si tratta di un bilanciamento di traffico vero e proprio. Infatti:

- L'istanza EC2 non sarà accessibile pubblicamente
- È possibile usare WAF e impostare il ruleset specializzato per WordPress
- Amazon Certificate Manager può fornire i certificati e rinnovarli automaticamente

In questo caso d'uso l'ALB si occuperà di redirigere il traffico HTTP al listener HTTPS, il bot LetsEncrypt si occuperà invece di gestire i certificati SSL sull'istanza per cifrare la comunicazione in transito fra istanza ed ALB.

Se associare un Web Application Firewall al load balancer è relativamente facile, vediamo invece come configurare l'ALB per permettere il **rinnovo dei certificati con LetsEncrypt**.

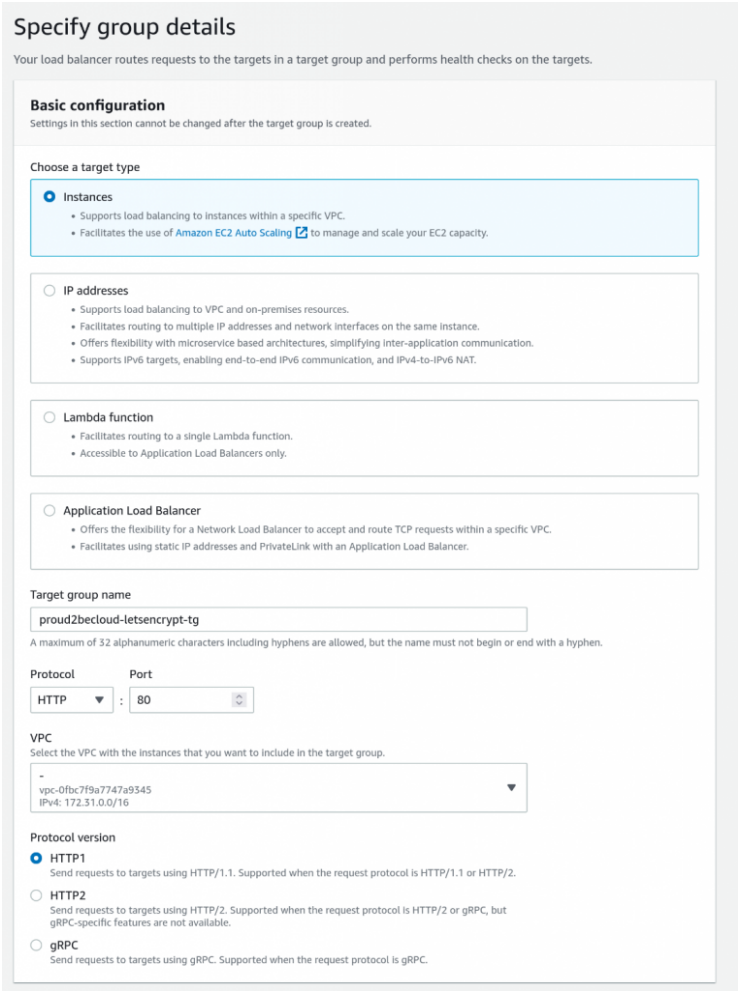
Dopo aver installato CertBot, è possibile ottenere certificati per il sito, ma il processo di verifica richiede l'accesso ad un path sul web server

```
(/.well-known-acme-challenge/).
```

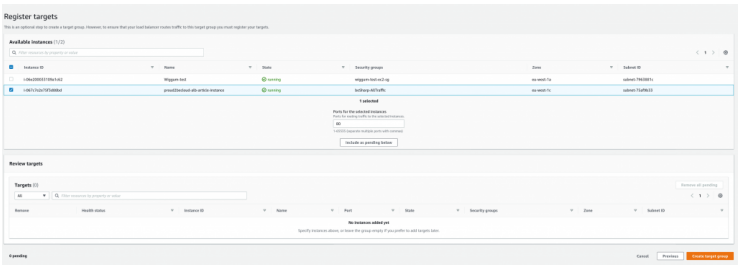
utilizzando il protocollo HTTP.

Per permettere questo, dovremo definire un target group che usa la porta 80 e poi inserire una regola che lo utilizzerà solamente per il path specifico.

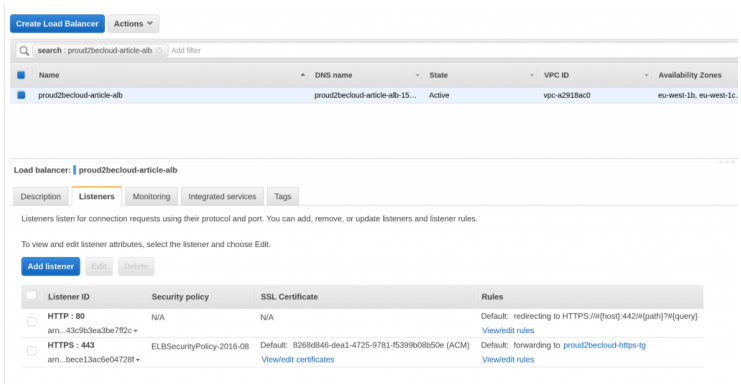
Per prima cosa, sulla console di gestione, occorre creare un nuovo target group:

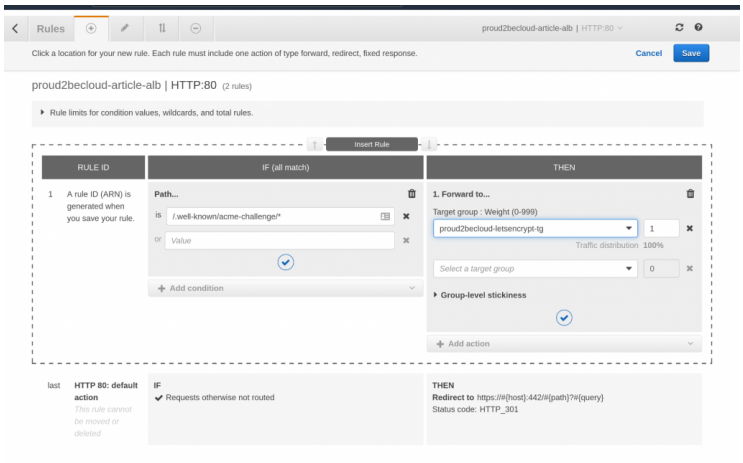


Selezioniamo l'istanza e clicchiamo su "Include as pending below"



Dopo aver creato il target group, aggiungiamo una nuova regola cliccando su "**View/edit rules**"

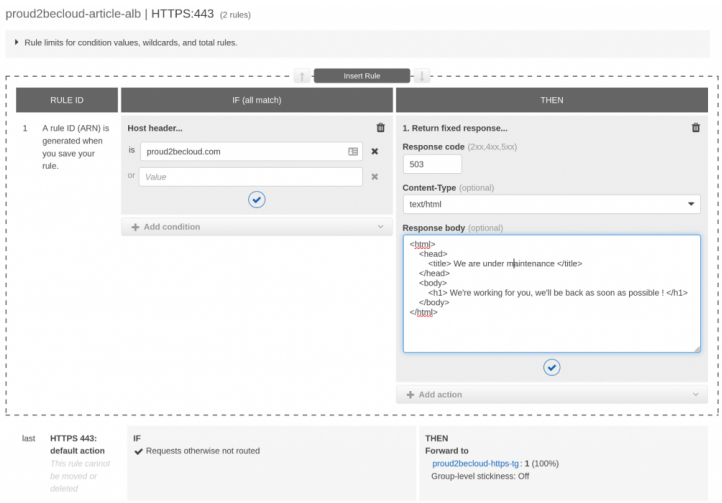




La regola inserita deve avere una priorità più alta rispetto all'azione di default che fa la redirezione da HTTP ad HTTPS

Impostare una pagina di manutenzione in modo semplice e veloce

Può sembrare ovvio, ma a volte ce ne dimentichiamo. Impostare una regola con una fixed response ed uno status code di 503 permette di ottenere una pagina di manutenzione in modo rapido per un sito web:



Implementare una logica custom per gli health check in ECS bilanciati da un NLB

In questo esempio, useremo un serven Minecraft, anche se non si tratta di un workload tradizionale. Minecraft utilizza la porta 25565 con un protocollo custom.

Possiamo eseguire l'immagine del server in un cluster ECS configurando il servizio per mantenere il server in esecuzione in modalità "auto-healing" impostando il parametro desired count ad 1.

Con un Network Load Balancer, avremo indirizzi IP statici a cui connetterci, per cui il container funzionante riceverà sempre traffico e non dovremo preoccuparci di un

eventuale cambio di indirizzo IP.

Possiamo configurare il target group per controllare che la porta 25665 sia raggiungibile. Vorremmo anche essere sicuri che il server sia disponibile e che sia in grado di rispondere ad un comando.

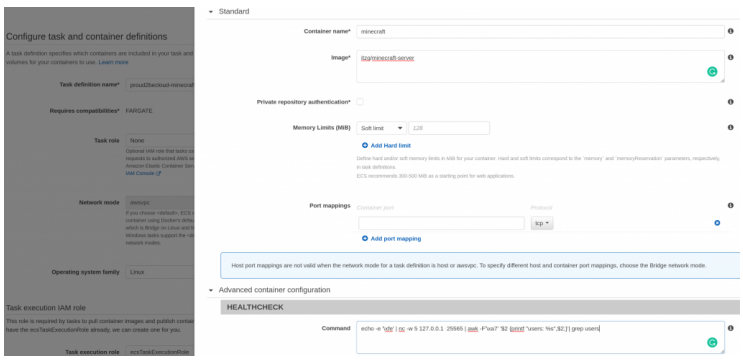
Questo script bash controlla il numero di utenti connessi. Sfortunatamente, però, il target group non può essere configurato per eseguirlo.

```
echo -e '\xfe' | nc -w 5 127.0.0.1  25565 | awk -F'\xa7' '$2 {printf
 "users: %s",$2;}'| grep users
```

Fortunatamente, ECS Fargate permette di definire un health check personalizzato che viene eseguito all'interno della task instance.

Basta semplicemente aggiungere il comando alla container definition. In questo modo, nel caso di fallimento, il container sarà terminato ed una nuova istanza prenderà il suo posto.

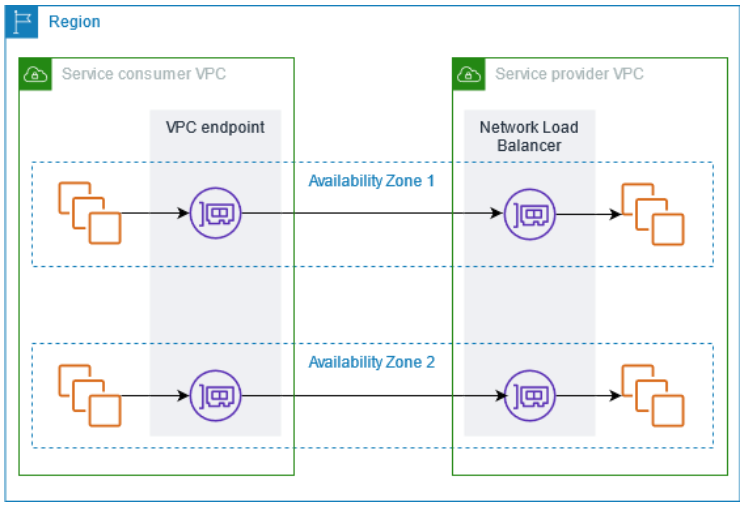
Nota: per essere considerato OK, lo script di health check deve uscire con uno status code impostato a 0.



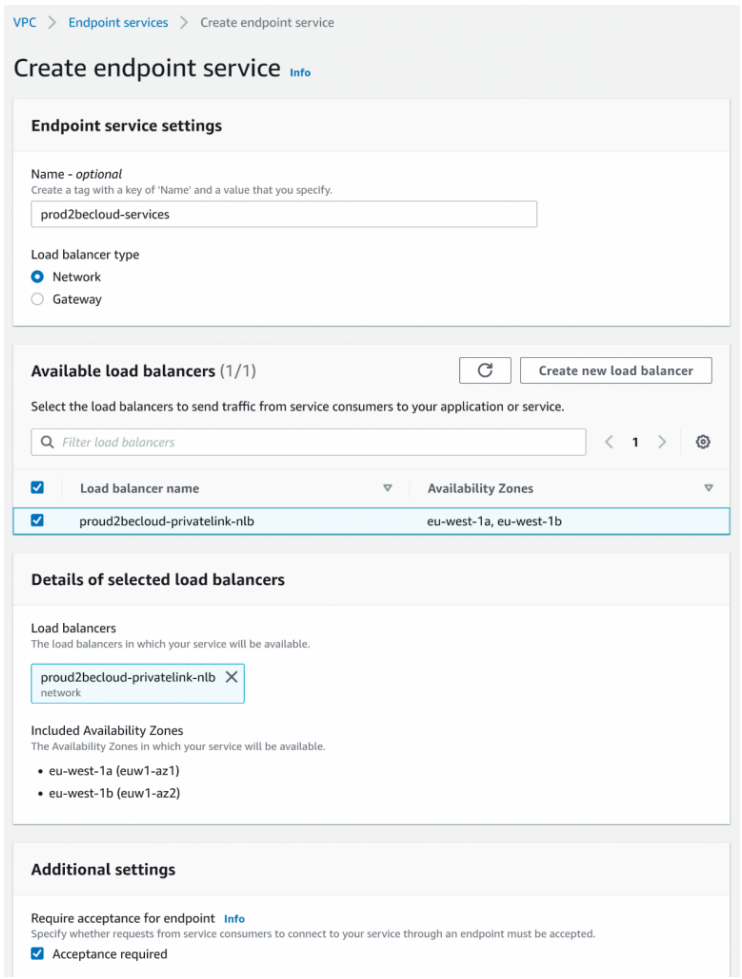
Condividere servizi con altri account AWS senza utilizzare Internet

AWS PrivateLink è una tecnologia che permette di offrire servizi a clienti e partner, traendo vantaggio dall'infrastruttura di rete AWS e senza utilizzare internet.
Chi condivide il servizio con un altro account AWS diventa il *Service Provider*, mentre l'utilizzatore è chiamato *Service Consumer*.

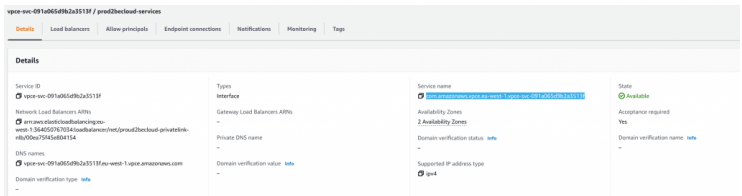
PrivateLink funziona collegando una interfaccia di rete di un Network Load balancer con altri account, creando una interfaccia locale nella VPC "consumer".



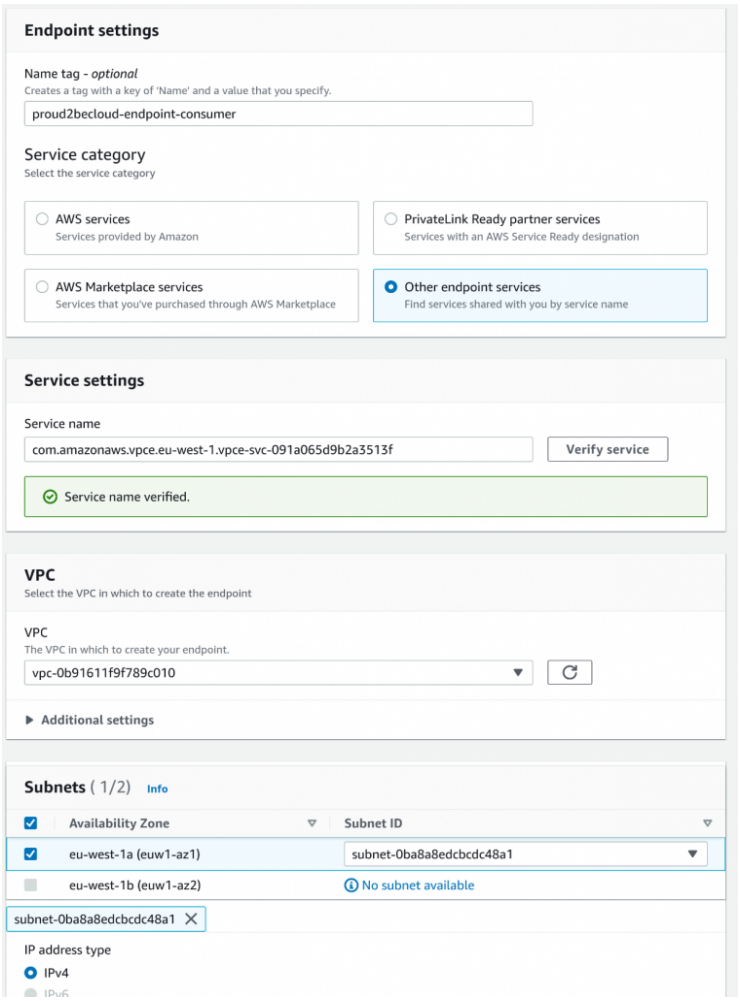
Per condividere un servizio, per prima cosa occorre creare un **Endpoint Service** andando nella sezione VPC della console, selezionando "Endpoint Services" e specificando "**Network**" come tipo di load balancer.



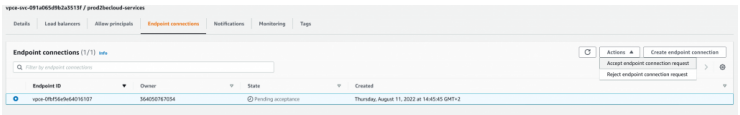
Il campo Service Name (visualizzato quando l'endpoint è disponibile), sarà utile per il consumer. Annotiamolo.



Per utilizzare il servizio come consumer, basta fare click nella sezione "Endpoints" della VPC, selezionare "Create new endpoint" e poi "**other endpoint services**", inserendo il nome del servizio preso dal passaggio precedente. Una volta fatto click su "**Verify Service**" e verificato il servizio è possibile selezionare la VPC, la subnet di destinazione e il security group per l'endpoint.



Nota: se il campo "Acceptance Required" è stato abilitato, occorre accettare la connessione nell'account "provider".



Una volta accettata la connessione, l'endpoint sarà pronto e il consumer potrà usare il servizio senza passare per Internet.

Bonus Tip

Come ultimo suggerimento è sempre utile ricordare che è possibile usare un **Application Load Balancer come target**, in modo da rendere disponibili ai consumer un servizio interno senza dovre riconfigurare l'infrastruttura.

Per concludere

I servizi AWS ELB sono molto flessibili in termini di scelta e configurazione. Usando la giusta combinazione di servizi gestiti è possibile ridurre notevolmente la complessità e i costi di gestione.

Abbiamo descritto solamente alcuni scenari, ma le combinazioni possibili sono infinite. Trovare la soluzione perfetta per le proprie esigenze è solo questione di progettazione!

E voi quali altre soluzioni di load balancing avete trovato? Fateci sapere nei commenti!

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!

Damiano Giorgi

Ex sistemista on-prem, pigro e incline all'automazione di task noiosi. Alla ricerca costante di novità tecnologiche e quindi passato al cloud per trovare nuovi stimoli.L'unico hardware a cui mi dedico ora è quello del mio basso; se non mi trovate in ufficio o in sala prove provate al pub o in qualche aeroporto!
