Technology Know-How: NGINX

Sistemi Distribuiti
Elena Morelli
A.A. 2022/2023

Che cos’è NGINX ?

Nginx, è un web server leggero ad alte prestazioni, disponibile per vari sistemi Unix, Unix-like e Windows. Rilasciato nel 2004 come software open source, NGINX è stato concepito con lo scopo esplicito di essere più performante del web server Apache.

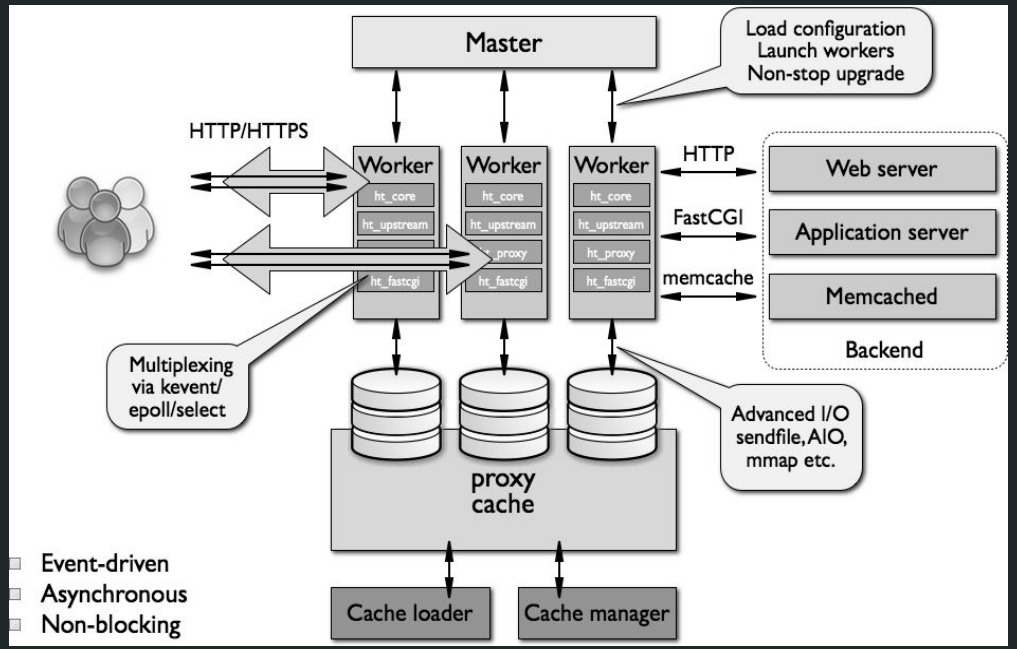
Non è soltanto un server web, può agire da reverse proxy, da proxy TCP/UDP, da e-mail proxy e da load balancer.

Architettura 1/2

Nginx presenta un’architettura modulare, event-based, asincrona, single threaded, non bloccante di tipo master-slave.

Le componenti fondamentali sono:

- Master
- Worker
- Cache Loader
- Cache Manager



Architettura 2/2

Master: È il processo principale che si occupa di creare e orchestrare tutti gli altri, ed è responsabile della lettura e validazione della configurazione, creazione, binding e chiusura dei socket, gestione dei processi Worker, riconfigurazione senza interruzioni del servizio, e altro.

Workers: Sono i processi che contengono il core e moduli principali per il funzionamento di NGINX, sono single-thread e indipendenti tra di loro. Vengono creati all'avvio dal Master e si mettono in ascolto per eventuali eventi, scatenati da connessioni in ingresso, su una determinata lista di socket.

Cache Loader: È il processo che gestisce il caricamento in memoria della cache disk-based, popolando il database in memoria con metadati e la preparazione dell'istanza di NGINX per lavorare con file storicizzati sul disco in una determinata struttura di directory.

Cache Manager: Processo che viene eseguito periodicamente, e si occupa di pulire la cache affinché resti all'interno dei limiti di grandezza specificata in configurazione.

Struttura

Event context: Viene utilizzato per impostare opzioni globali che influenzano il modo in cui Nginx gestisce le connessioni a livello generale.

HTTP context: contiene tutte le direttive e i contesti necessari per definire il modo in cui il programma gestirà le connessioni HTTP o HTTPS.

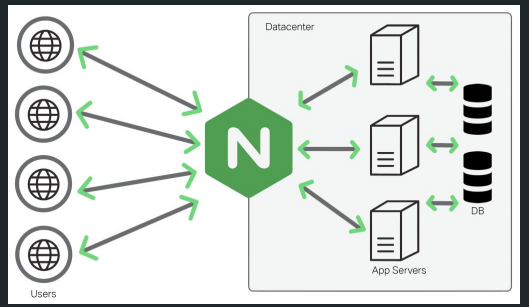
Server context: ogni sua istanza definisce un server virtuale specifico per gestire le richieste client.

Location Context: utilizzata per gestire un determinato tipo di richiesta selezionata in virtù della corrispondenza tra la definizione della location, con la richiesta client, tramite un algoritmo di selezione.

```
events {

}

http {

      server {

            listen 80;
            server_ name localhost;

            location  /  {

                  proxy_pass http://backend.example.com;
            }
      }
}
```

Configurazioni principali

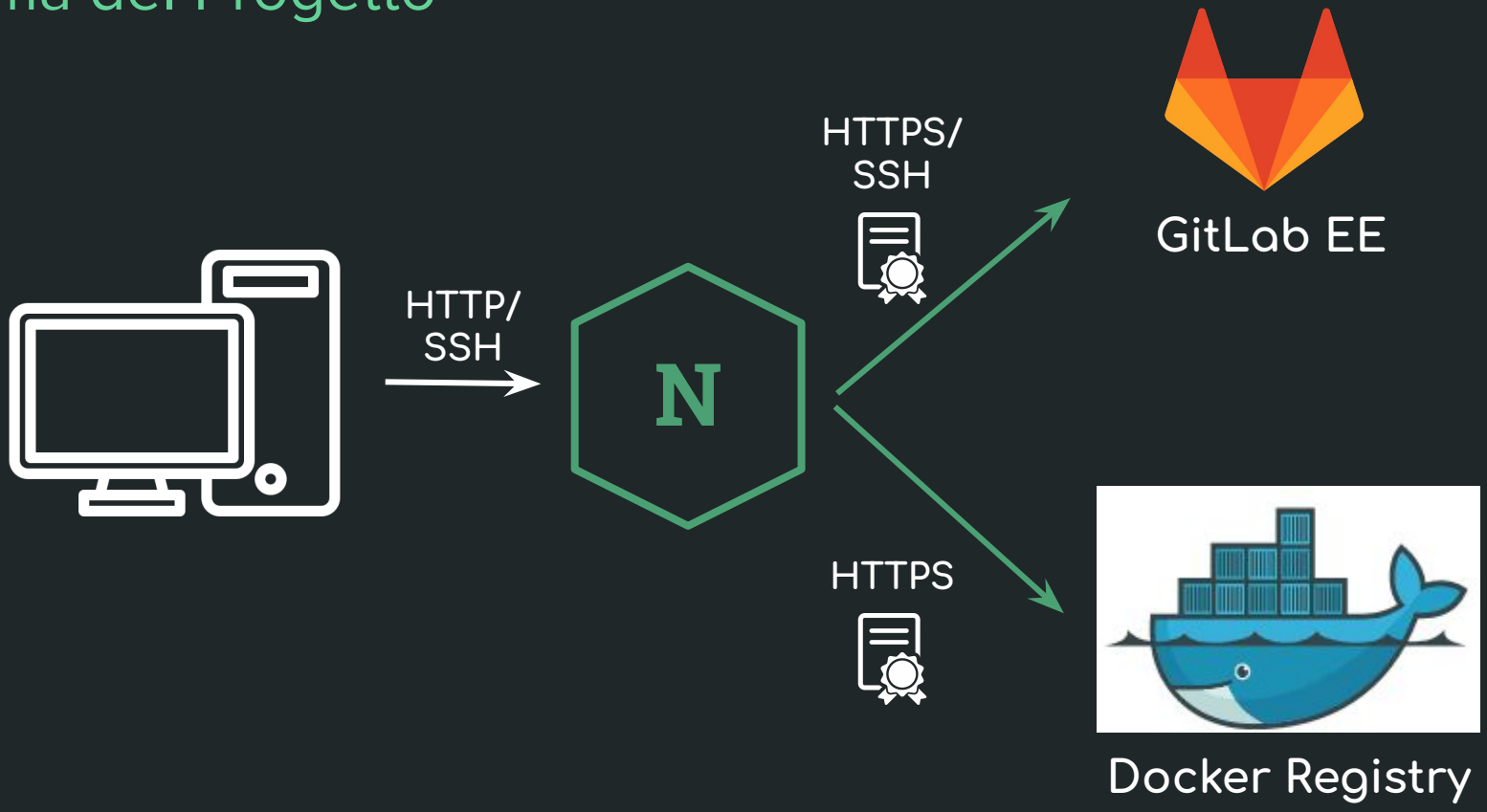
- **Load Balancer**: Il bilanciamento del carico su più istanze è una tecnica comunemente utilizzata per ottimizzare l'impiego delle risorse, massimizzare il throughput, ridurre la latenza di risposta e garantire una certa resilienza in caso di guasti. Consiste nel far pervenire tutte le richieste ad un unico server, il quale provvede a distribuirle ai vari application server. Nel load balancer TCP/UDP, detto di livello 4, il server è ignaro del contenuto dei messaggi scambiati tra client e server e del loro significato, limitandosi semplicemente a reindirizzare i flussi opportunamente.
- **Reverse Proxy**: server web che si frappone tra i client HTTP(s) ed il/i server utilizzato/i. I client interrogano il server proxy inviando le proprie richieste HTTP, il quale a sua volta interroga il server interno che contiene la risorsa richiesta, elabora la risposta e la ritrasmette al client.
- **Mail Proxy Server**: impiegato come proxy server per i protocolli IMAP, POP3 e SMTP. L'impiego di un unico endpoint pubblico per l'accesso ai servizi di posta permette di distribuire il carico su più server di backend e di implementare politiche avanzate di selezione del server



Apache vs NGINX

Apache	Nginx
Apache viene progettato per essere un web server	Nginx è un web server oltre ad essere un proxy server
Un singolo thread può processare una sola connessione	Un singolo thread può processare connessioni multiple
Segue un approccio multi-thread per processare le richieste del client	Utilizza un approccio event-driven per servire le richieste del client
Gestisce il contenuto dinamico all'interno del web server stesso	Non è in grado di elaborare contenuti dinamici in modo nativo
Non riesce a processare richieste multiple contemporaneamente quando il traffico in rete risulta essere intenso	Può elaborare più richieste client contemporaneamente ed efficientemente con risorse hardware limitate
I moduli vengono caricati o scaricati dinamicamente, rendendolo più flessibile	I moduli non possono essere caricati dinamicamente. Devono essere compilati all'interno del software di base
Esegue su tutti i sistemi Unix come Linux, BSD, ecc. oltre a supportare completamente Windows	Gira su sistemi moderni Unix like; tuttavia ha un supporto limitato per Windows

Schema del Progetto



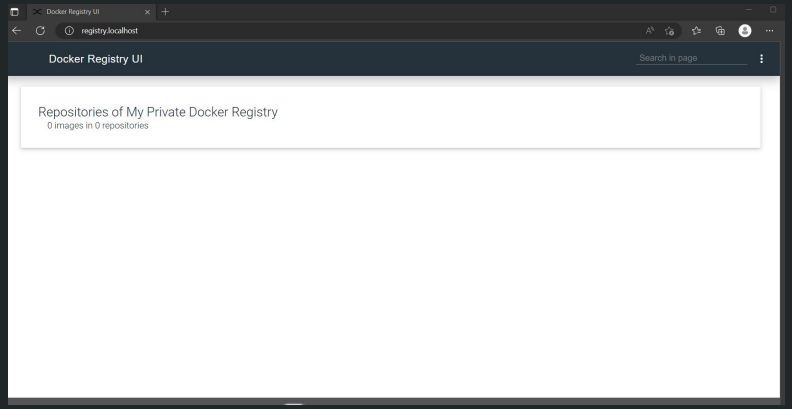
NGINX file di configurazione

```
stream {

    upstream dockergitlabssh {
       server gitlab:22;
    }

    server {
       listen 22;
       proxy_pass dockergitlabssh;
       ssl_preread on;
    }

}

http {
    client_max_body_size 3000M;

    upstream dockergitlab {
        server gitlab:443;
    }

    upstream dockerregistry {
         server ui:80;
    }
```

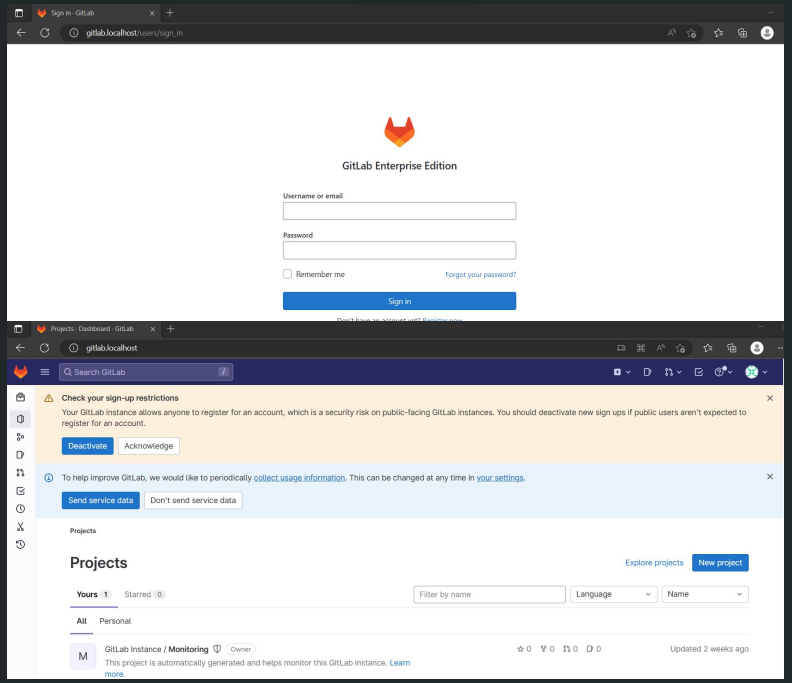
```
    server {
        listen 80 default_server;
        listen 443 ssl;

        ssl_certificate /etc/nginx/ssl/certificate.crt;
        ssl_certificate_key /etc/nginx/ssl/privateKey.key;

        server_name gitlab.localhost;
        server_tokens off;

        location / {
            proxy_pass https://dockergitlab;
        }
    }

    server {
         #...
         location / {
             proxy_pass http://dockerregistry;
         }
    }
}
```

Schermate dei Servizi

http://registry.localhost



http://gitlab.localhost



ssh://localhost

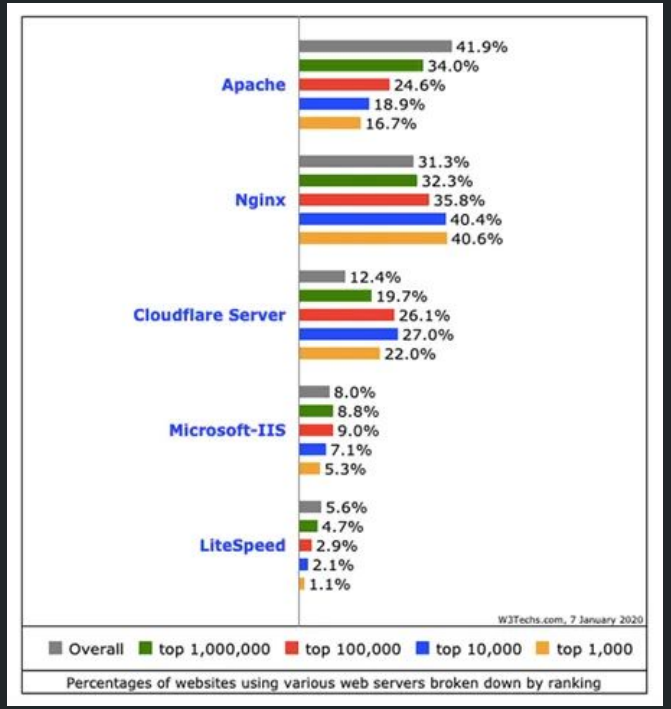
```
PS C:\Users\elena> ssh -T git@localhost
Welcome to GitLab, @root!
```

Conclusioni

Nginx è stato progettato per gestire un elevato volume di richieste contemporaneamente. Pertanto, offre tempi di caricamento più rapidi e prestazioni migliori rispetto alla maggior parte degli altri web server.

Utilizza meno risorse e hardware rispetto ai competitor, rendendolo una soluzione più economica.

Nonostante Apache continui ad essere, in termini assoluti, il web server più utilizzato, si può notare come Nginx si sia creato una notevole fetta di mercato tra i siti di fascia più alta, diventando a tutti gli effetti uno standard nel mercato.



Grazie

Sistemi Distribuiti
Elena Morelli
A.A. 2022/2023