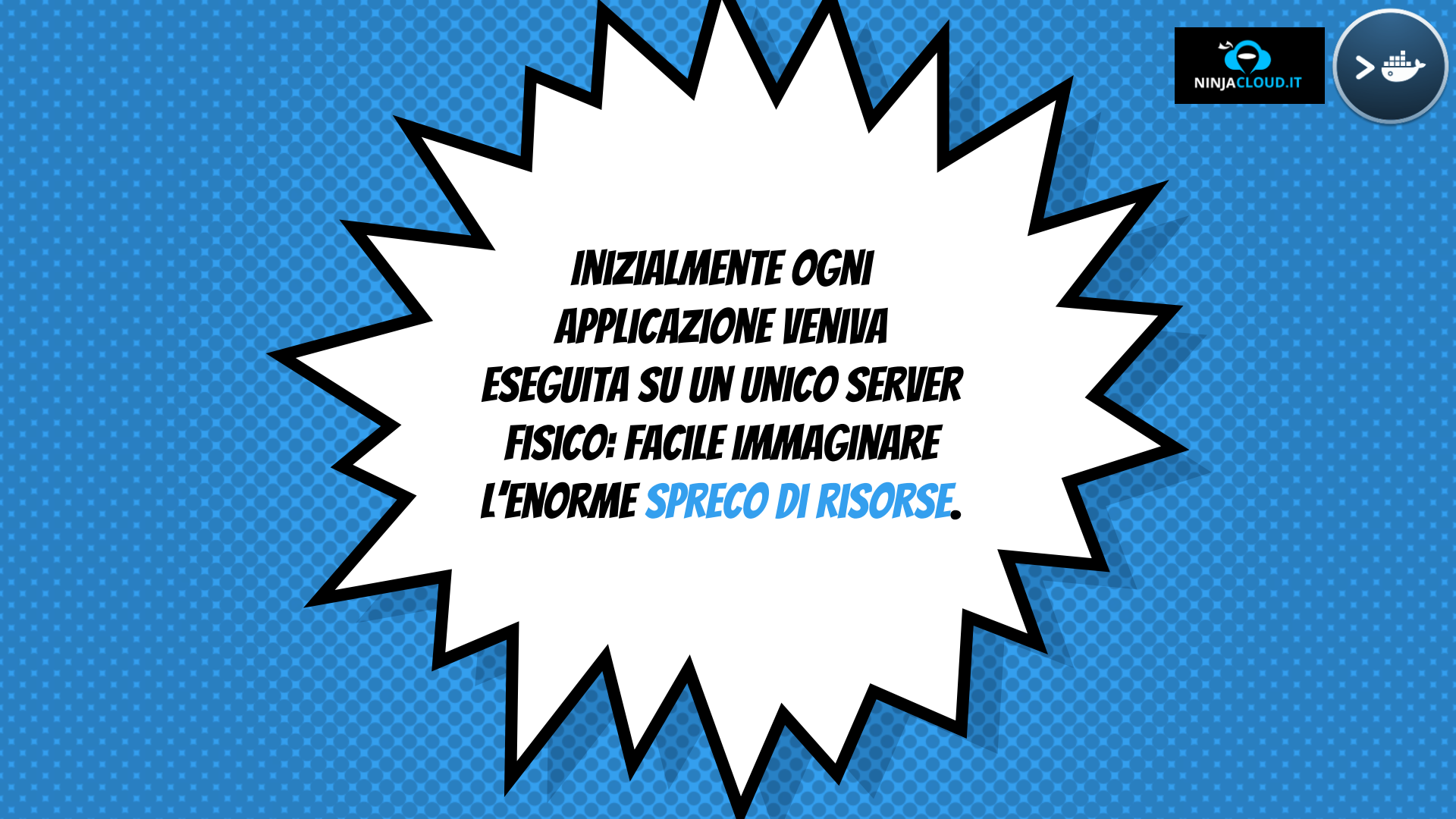

NINJACLOUD.IT
INTRODUZIONE
A DOCKER!

NINJACLOUD.IT

CONTAINER


NINJACLOUD.IT
INIZIALMENTE OGNI APPLICAZIONE VENIVA ESEGUITA SU UN UNICO SERVER FISICO: FACILE IMMAGINARE L'ENORME SPRECO DI RISORSE.

NINJACLOUD.IT

SUCCESSIVAMENTE CI SIAMO ORIENTATI VERSO LA VIRTUALIZZAZIONE CHE HA RESO LE COSE SICURAMENTE MIGLIORI...MA C'È UN PROBLEMA...

Il problema è che ogni macchina virtuale necessita del proprio sistema operativo e come ben sappiamo ogni sistema operativo consuma risorse in termini di CPU, RAM e spazio disco.
Oltre ad eventuali costi di licenze e di aggiornamento.

NINJACLOUD.IT

E QUINDI?


NINJACLOUD.IT
E QUINDI INTRODUCIAMO IL
CONCETTO DI
CONTAINER

I CONTAINER CI OFFRONO LA POSSIBILITÀ DI "PACCHETTIZZARE" UN'APPLICAZIONE E RENDERLA TRASPORTABILE E QUINDI DISPONIBILE VERSO ALTRI SISTEMI.
UNA DELLE DIFFERENZE RISPETTO ALLA VIRTUALIZZAZIONE È CHE I CONTAINER ALL'INTERNO DELLO STESSO HOST CONDIVIDONO LO STESSO SISTEMA OPERATIVO.

NINJACLOUD.IT

UN CONTAINER È QUINDI UN AMBIENTE ISOLATO ED È PROPRIO L'ISOLAMENTO E LA SICUREZZA CHE NE DERIVANO CHE CI PERMETTONO DI ESEGUIRE PIÙ CONTAINER SIMULTANEAMENTE ALL'INTERNO DI UN HOST!

NINJACLOUD.IT

I CONTAINER SONO "LEGGERI" IN QUANTO NON NECESSITANO DELLA PRESENZA DI UN HYPERVISOR COME NEL CASO DELLE MACCHINE VIRTUALI. SI ESEGUONO INFATTI DIRETTAMENTE ALL'INTERNO DEL KERNEL DELLA MACCHINA HOST.

I CONTAINER

Cerchiamo di rendere il concetto chiaro e immediato.

Possiamo immaginare i container come delle piccole **entità al cui interno viene eseguito un'applicazione** in maniera autonoma: ovvero quell'applicazione contiene tutti i componenti essenziali affinché possa eseguirsi correttamente.

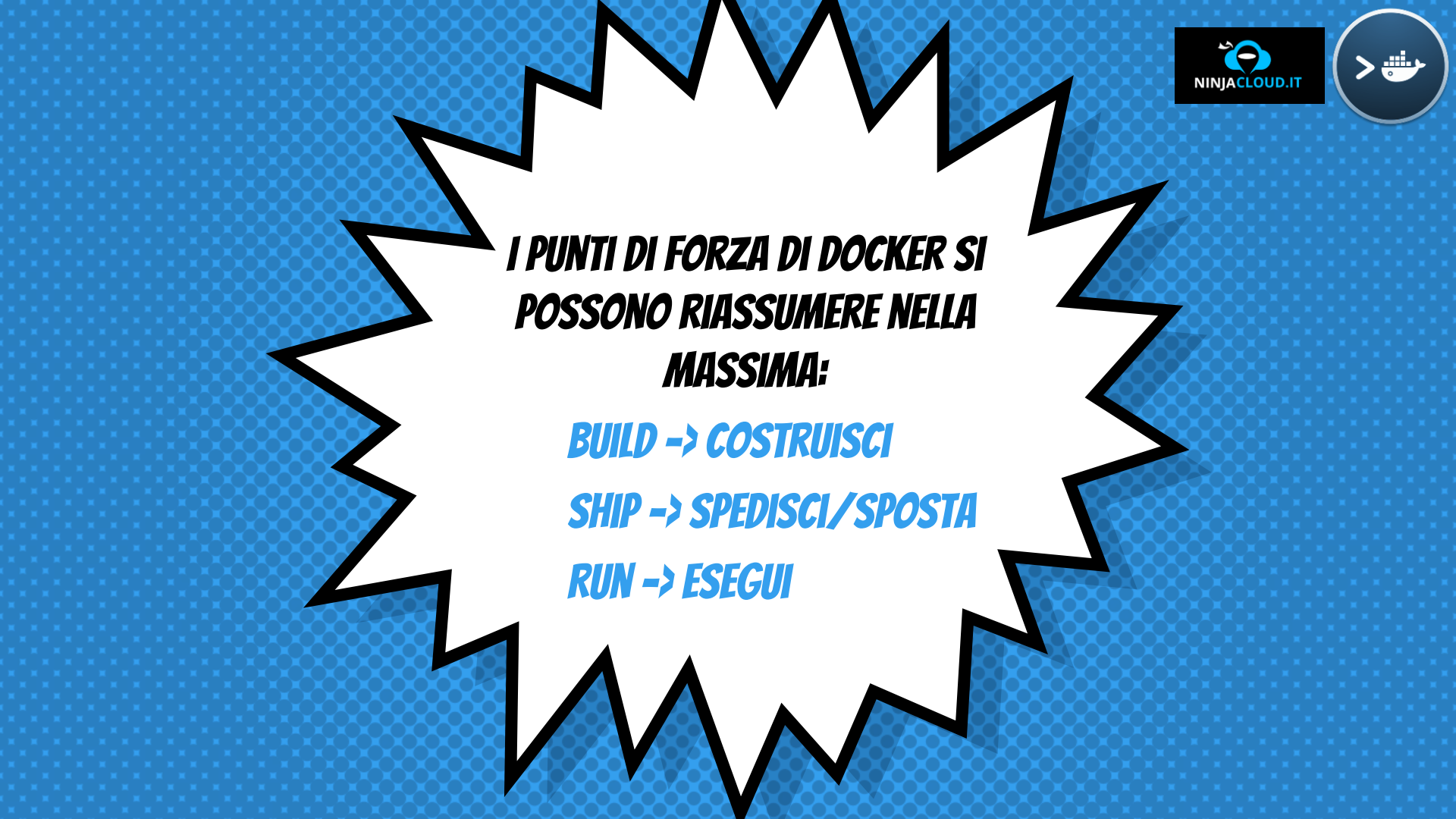
Cioè rende i container trasportabili che è uno dei maggiori punti di forza di questa tecnologia.

NINJACLOUD.IT

PRIMA DI DOCKER LA GESTIONE E L'UTILIZZO DEI CONTAINER ERANO TEMI MOLTO COMPLESSI. DOCKER HA RESO I CONTAINER ACCESSIBILI SU LARGA SCALA.

NINJACLOUD.IT

DOCKER RISOLVE UN PROBLEMA FONDAMENTALE OVVERO LA NECESSITÀ DI AVERE DEGLI AMBIENTI DI SVILUPPO IDENTICI PER OGNI SINGOLO COMPONENTE APPLICATIVO.

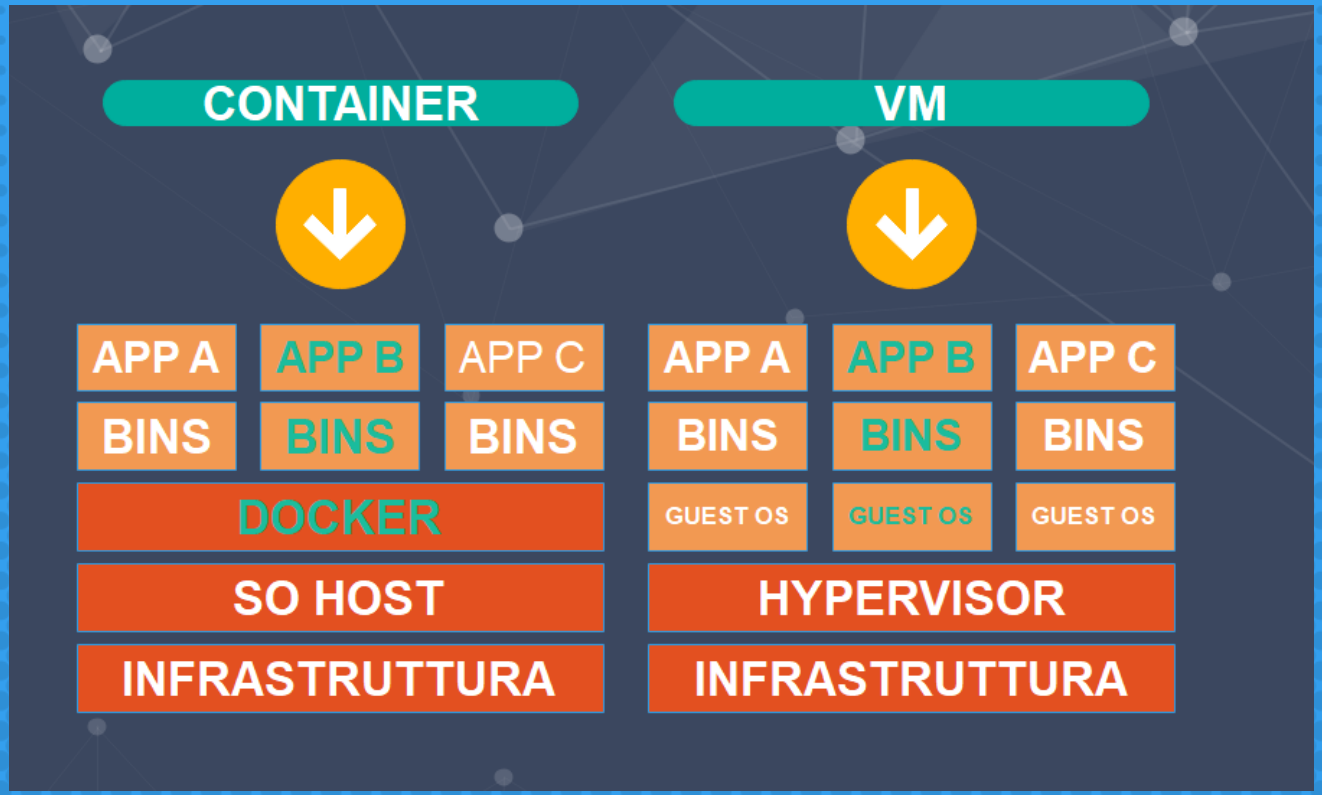

NINJACLOUD.IT
I PUNTI DI FORZA DI DOCKER SI POSSONO RIASSUMERE NELLA MASSIMA:
BUILD -> COSTRUISCI
SHIP -> SPEDISCI/SPOSTA
RUN -> ESEGUI

NINJACLOUD.IT

DOCKER ENGINE

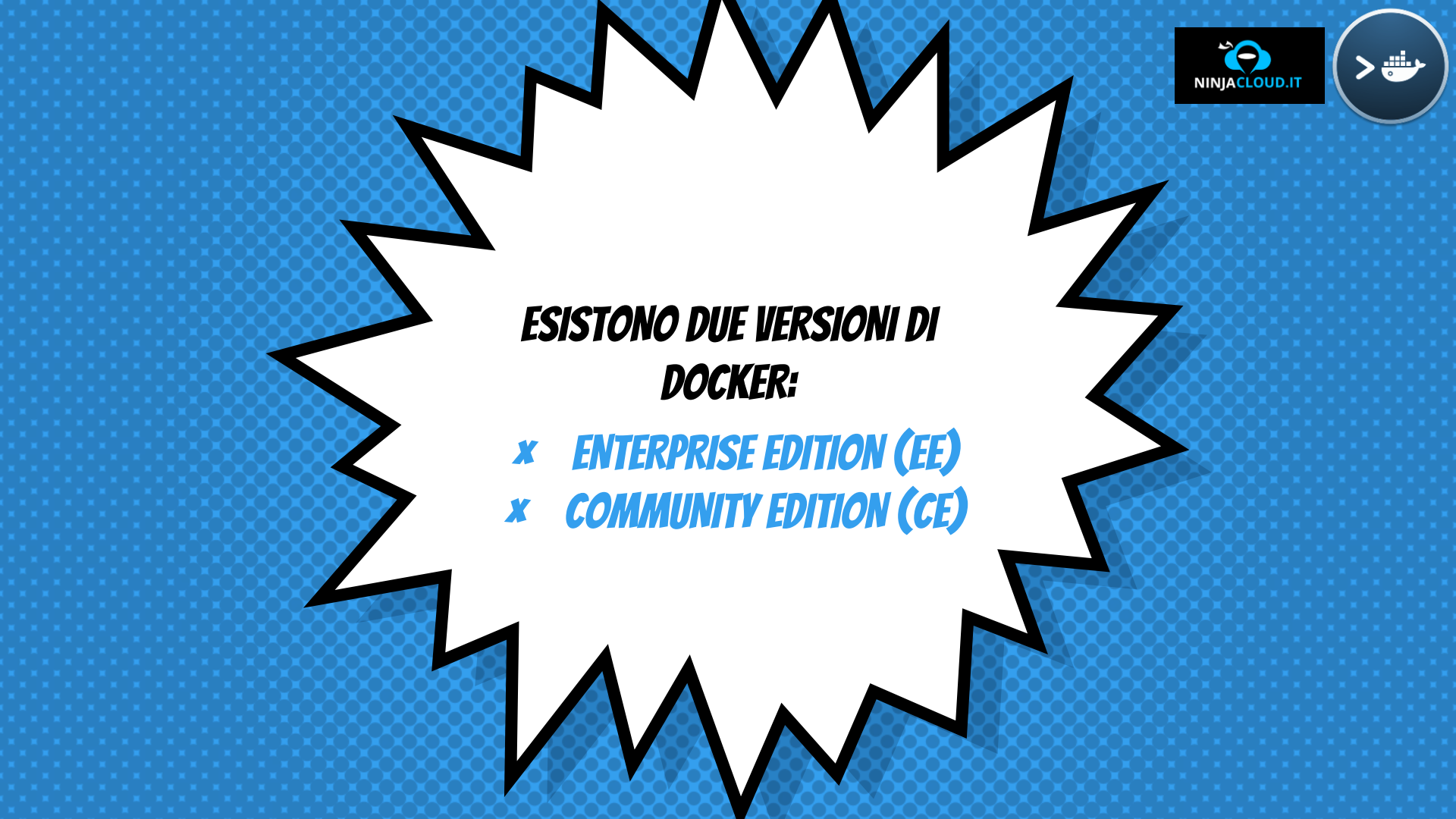
Come in VMWARE abbiamo ESXi che è l'hypervisor che ci permette di eseguire le macchine, in Docker abbiamo il **DOCKER ENGINE che ci permette di eseguire i CONTAINER.**

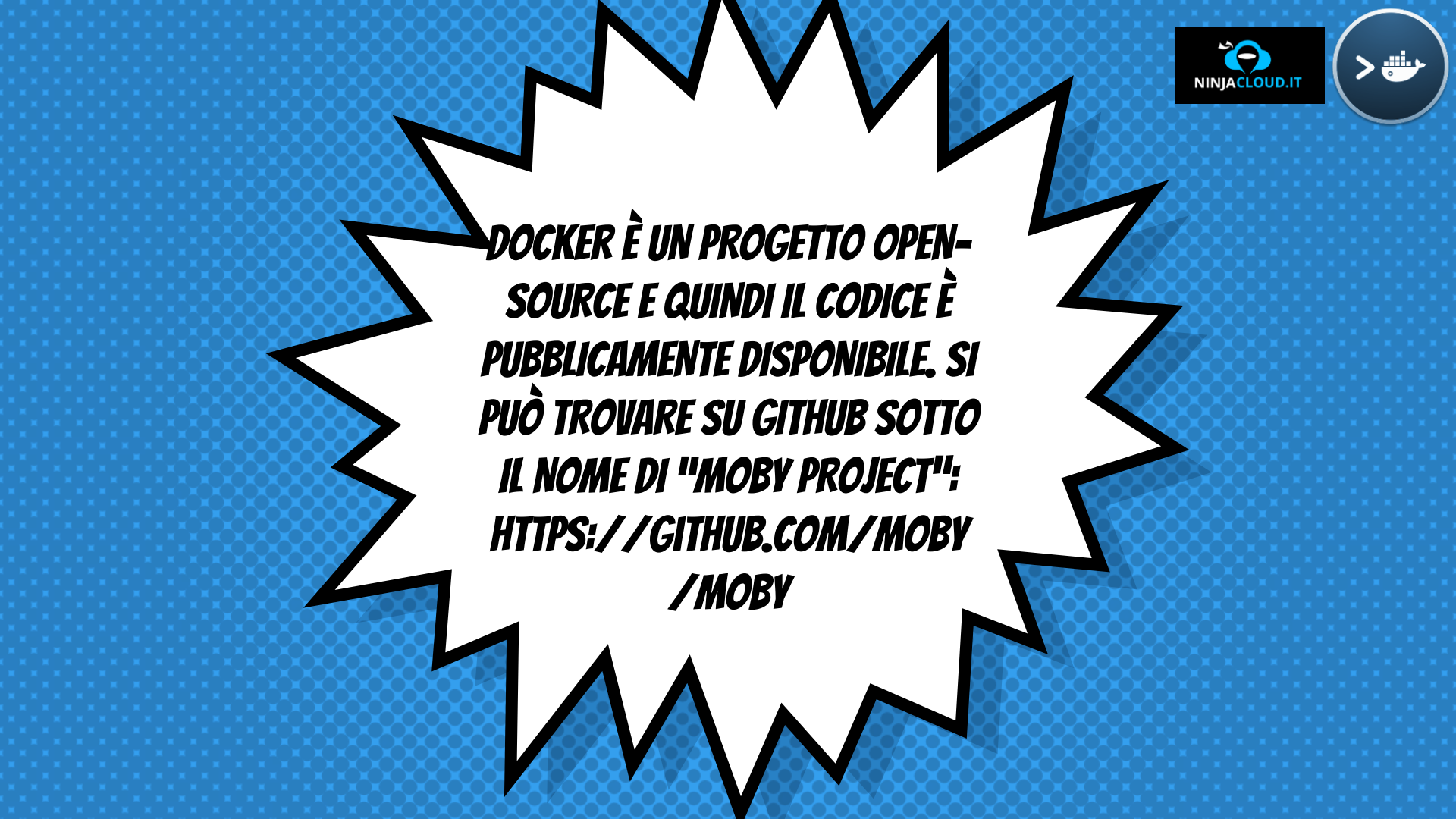
Ogni altro prodotto facente parte dell'ecosistema di Docker è direttamente collegato e dipende dal DOCKER ENGINE.

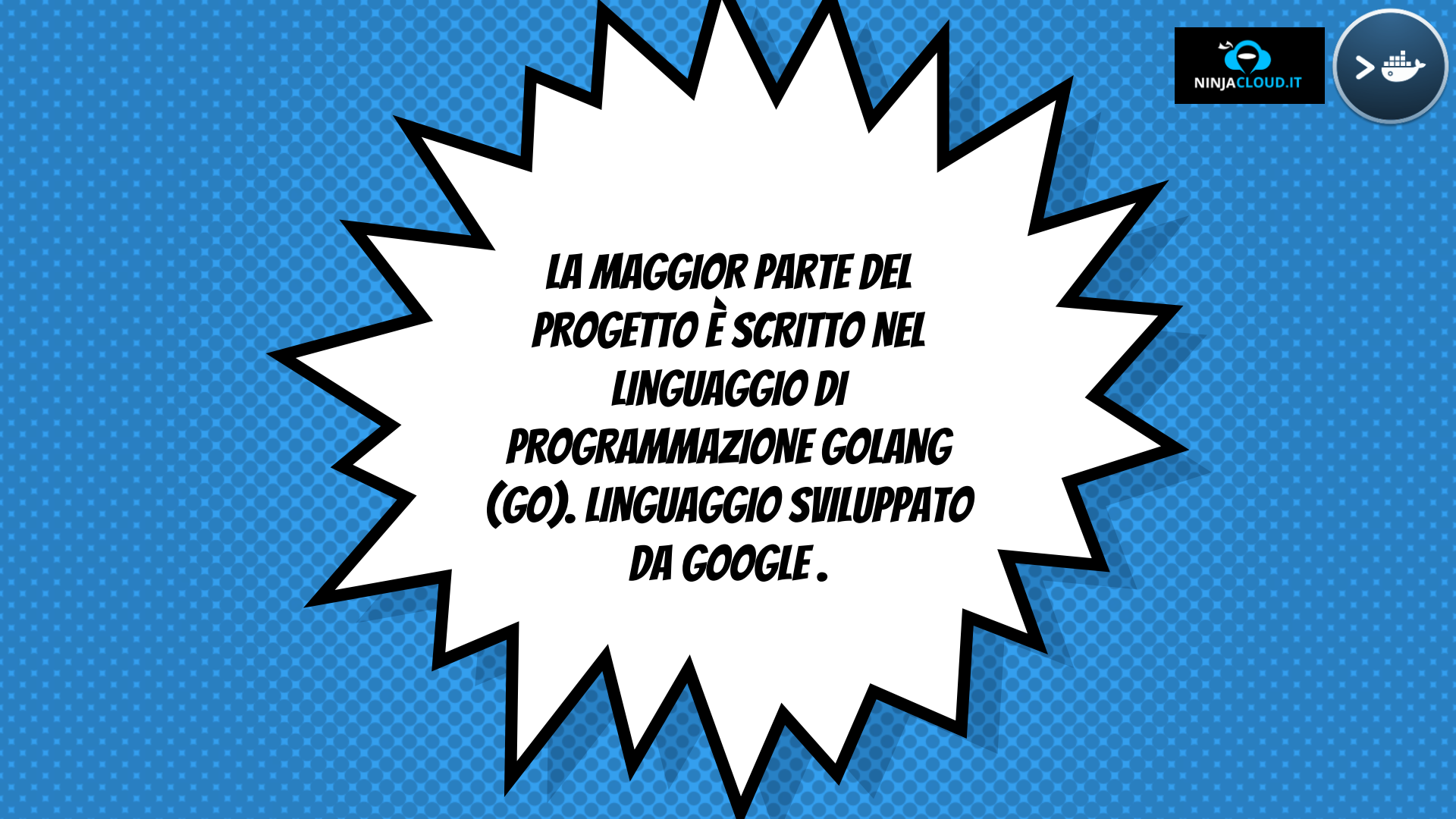

CONTAINER
VM
APP A
APP B
APP C
BINS
BINS
BINS
DOCKER
SO HOST
INFRASTRUTTURA
APP A
APP B
APP C
BINS
BINS
BINS
GUEST OS
GUEST OS
GUEST OS
HYPERVISOR
INFRASTRUTTURA

NINJACLOUD.IT

IL DOCKER ENGINE PUÒ ESSERE SCARICATO DIRETTAMENTE DAL SITO UFFICIALE DOCKER ED È DISPONIBILE SIA PER LINUX CHE PER WINDOWS.


NINJACLOUD.IT
ESISTONO DUE VERSIONI DI DOCKER:
x ENTERPRISE EDITION (EE)
x COMMUNITY EDITION (CE)


NINJACLOUD.IT
DOCKER È UN PROGETTO OPEN-SOURCE E QUINDI IL CODICE È PUBBLICAMENTE DISPONIBILE. SI PUÒ TROVARE SU GITHUB SOTTO IL NOME DI "MOBY PROJECT": HTTPS://GITHUB.COM/MOBY/MOBY


NINJACLOUD.IT
LA MAGGIOR PARTE DEL PROGETTO È SCRITTO NEL LINGUAGGIO DI PROGRAMMAZIONE GOLANG (GO). LINGUAGGIO SVILUPPATO DA GOOGLE .

L'OPEN CONTAINER INITIATIVE (OCI)

Questa organizzazione ha il compito di standardizzare l'infrastruttura DOCKER. Al momento sono stati pubblicati due standard:

- **image -spec.**
- **Runtime-spec.**

Approfondiremo questi concetti nel seguito del corso.

SI PUÒ OPERARE CON I CONTAINER DA DIFFERENTI PUNTI DI VISTA, DA PROSPETTIVE ORIENTATE ALLA **GESTIONE SISTEMISTICA** OPPURE ALL'A**MBITO DELLO SVILUPPO**.

NINJACLOUD.IT

Se facciamo riferimento all'ambito sistemistico ci preoccuperemo soprattutto della **gestione dei container**: creazione, start, stop, eliminazione, comunicazione di rete, ecc.

Se facciamo riferimento all'ambito dello sviluppo ci soffermeremo maggiormente sull'**applicazione**, su come trasformarla in un container e delle funzionalità che dovrà avere.

NINJACLOUD.IT

OVVIO CHE PER POTER OPERARE PROFITTEVOLMENTE È NECESSARIO CONOSCERE ENTRAMBI GLI ASPETTI, SPECIALIZZANDOSI POI SU QUELLO DI PROPRIO INTERESSE.

I CONCETTI BASE

Giusto qualche concetto per iniziare a prendere confidenza con la terminologia. Ci ritorneremo in seguito più volte.

NINJACLOUD.IT

1.

COMPONENTI PRINCIPALI

Ad installazione completata saranno presenti i seguenti componenti:

- Docker Client
- Docker Daemon

NINJACLOUD.IT

2.

IMMAGINI

Possiamo visualizzare le immagini Docker come delle **entità contenenti il filesystem del SO ed una certa applicazione**. Per fare un paragone, pensiamo ad un template di una VM oppure ad una classe Java.

NINJACLOUD.IT

3.

CONTAINER

Un container è un'istanza in esecuzione di una certa immagine Docker. Facendo riferimento al mondo della virtualizzazione è una VM che è stata avviata a partire da un template. A partire da una singola immagine possiamo avviare più container..

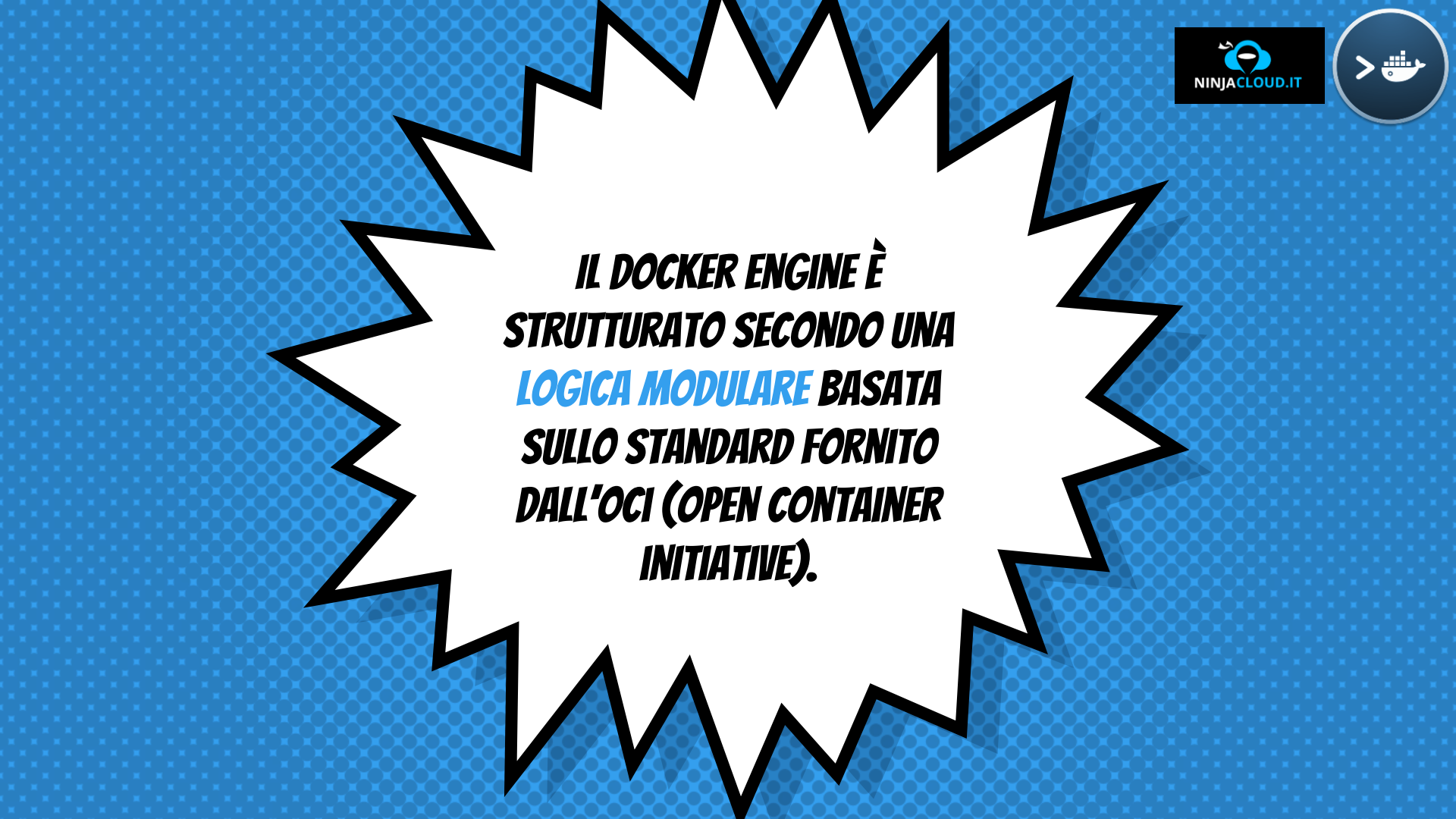
NINJACLOUD.IT

4.

DOCKERFILE

E' il **file che descrive l'applicazione** e fornisce a Docker le istruzioni su come l'applicazione deve essere definita all'interno dell'immagine.

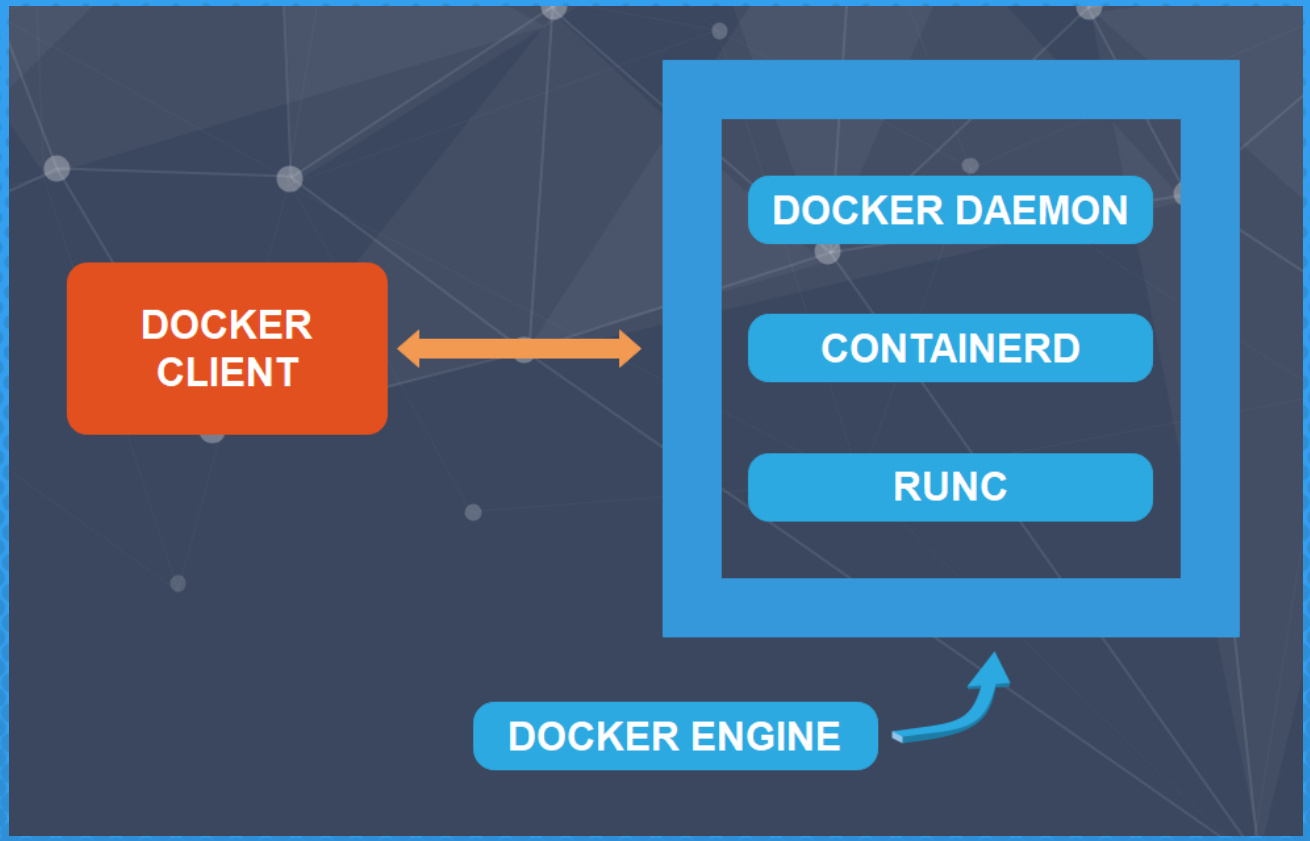
L'abbiamo già accennato, adesso cerchiamo di comprenderlo meglio!


NINJACLOUD.IT
IL DOCKER ENGINE È STRUTTURATO SECONDO UNA LOGICA MODULARE BASATA SULLO STANDARD FORNITO DALL'OCI (OPEN CONTAINER INITIATIVE).

NINJACLOUD.IT

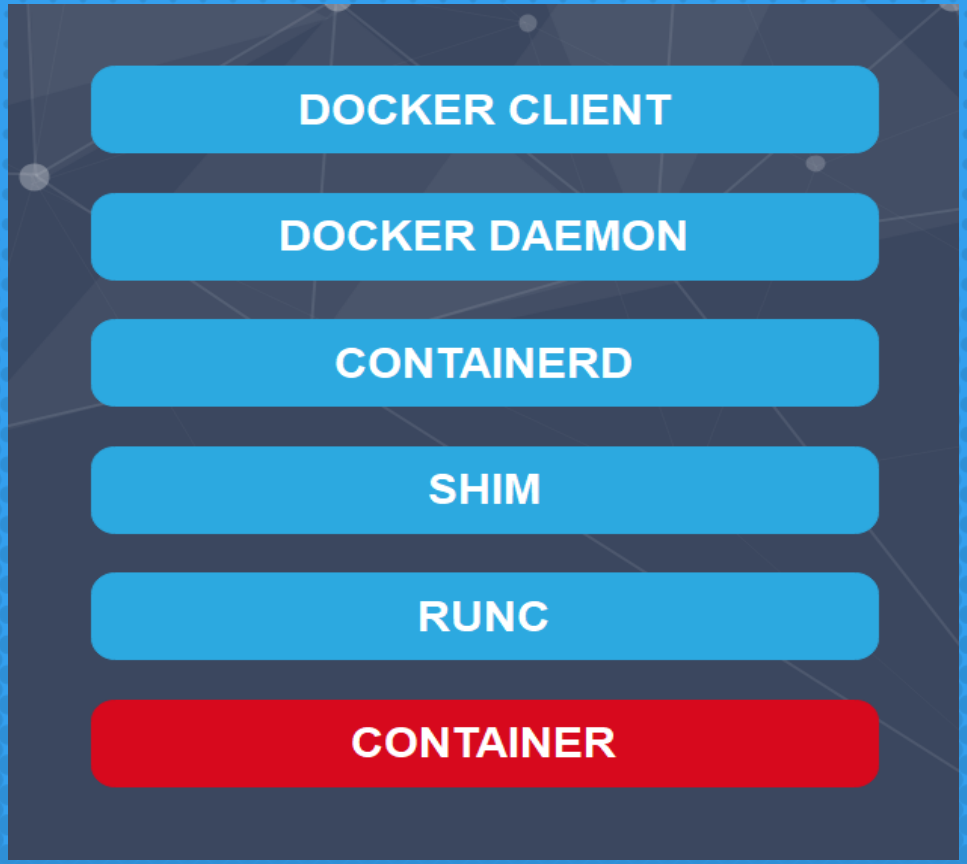
I componenenti principali del Docker Engine sono:

- Docker client.
- Docker daemon.
- containerd.
- runc.


DOCKER CLIENT
DOCKER DAEMON
CONTAINERD
RUNC
DOCKER ENGINE

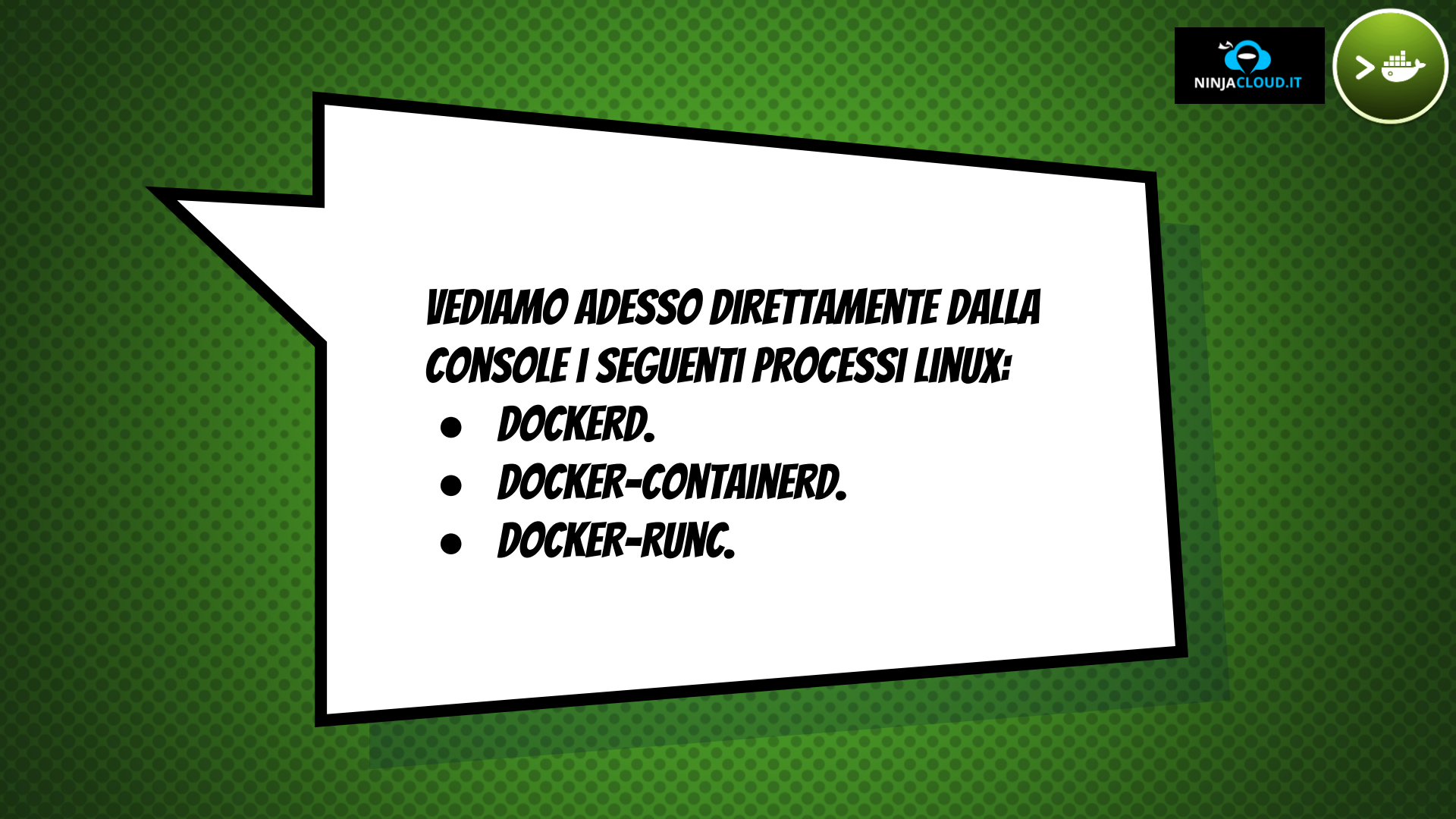
COME INTERAGISCONO QUESTI COMPONENTI?

1. Dalla Docker CLI sia esegue il comando di start del container.
2. Il Docker Client converte le istruzioni in un formato adeguato il per Docker Daemon.
3. Quando il Docker Daemon riceve i comandi di creazione del container, viene a sua volta chiamato il componenente “containerd”.
4. Containerd non può creare direttamente il container e utilizza il componente “runc”.
5. Runc si interfaccia con il kernel del SO e riunisce tutto ciò che è necessario per la creazione del container (namespace,cgroups,ecc).


DOCKER CLIENT
DOCKER DAEMON
CONTAINERD
SHIM
RUNC
CONTAINER

NINJACLOUD.IT

Il comando "docker version" ci permette di verificare se docker è in esecuzione o meno. Il comando "service docker status" ci permette di verificare lo stato del servizio docker.


NINJACLOUD.IT
VEDIAMO ADESSO DIRETTAMENTE DALLA CONSOLE I SEGUENTI PROCESSI LINUX:
DOCKERD.
DOCKER-CONTAINERD.
DOCKER-RUNC.

SIMULATORI WEB

NINJACLOUD.IT

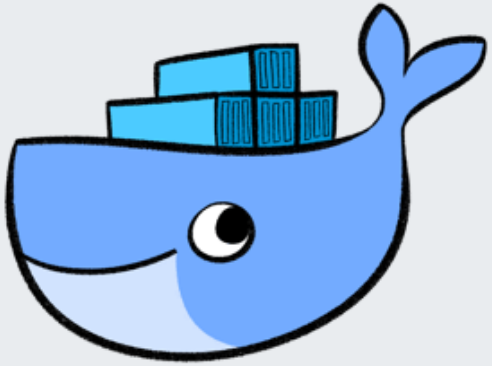
NELLA PROSSIMA SEZIONE VEDREMO COME INSTALLARE DOCKER. TUTTAVIA È POSSIBILE UTILIZZARE DEI SIMULATORI DIRETTAMENTE DAL WEB.

NINJACLOUD.IT

1.

PLAY WITH DOCKER

URL -> https://labs.play-with-docker.com/



Play with Docker

A simple, interactive and fun playground to learn Docker

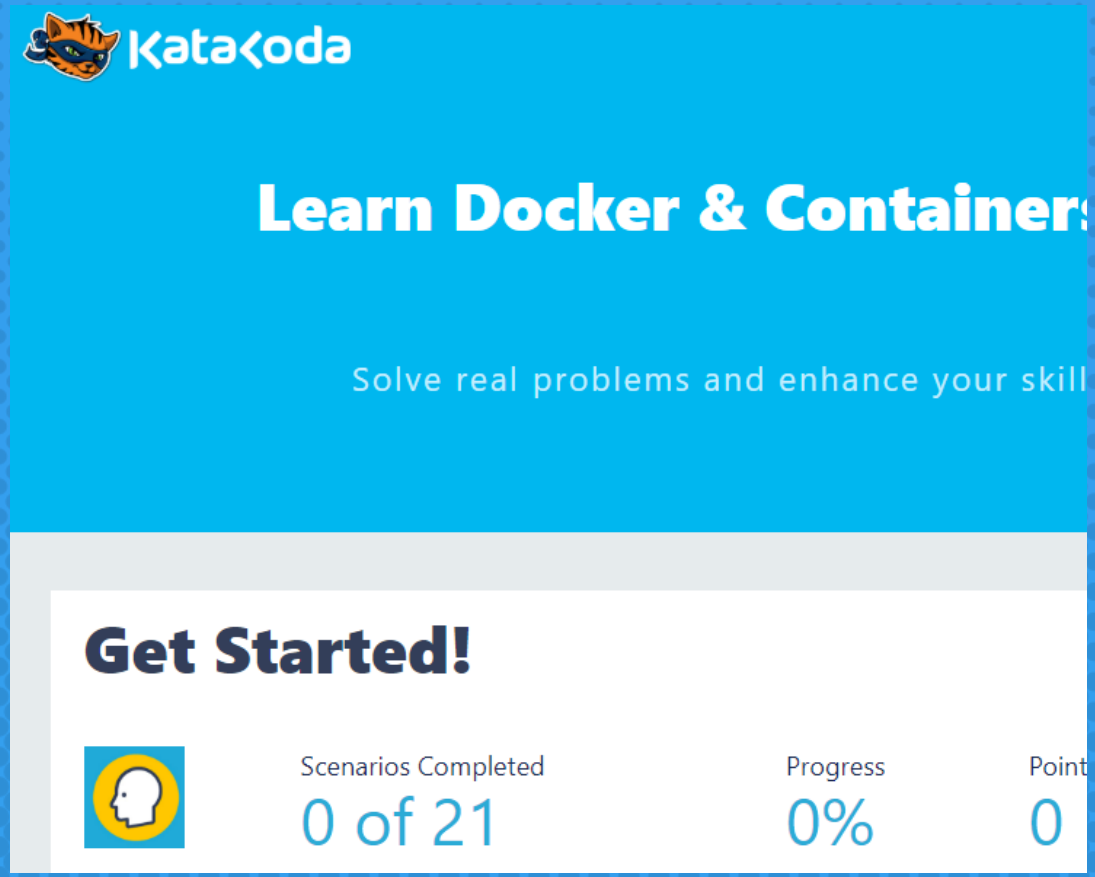
Login

NINJACLOUD.IT

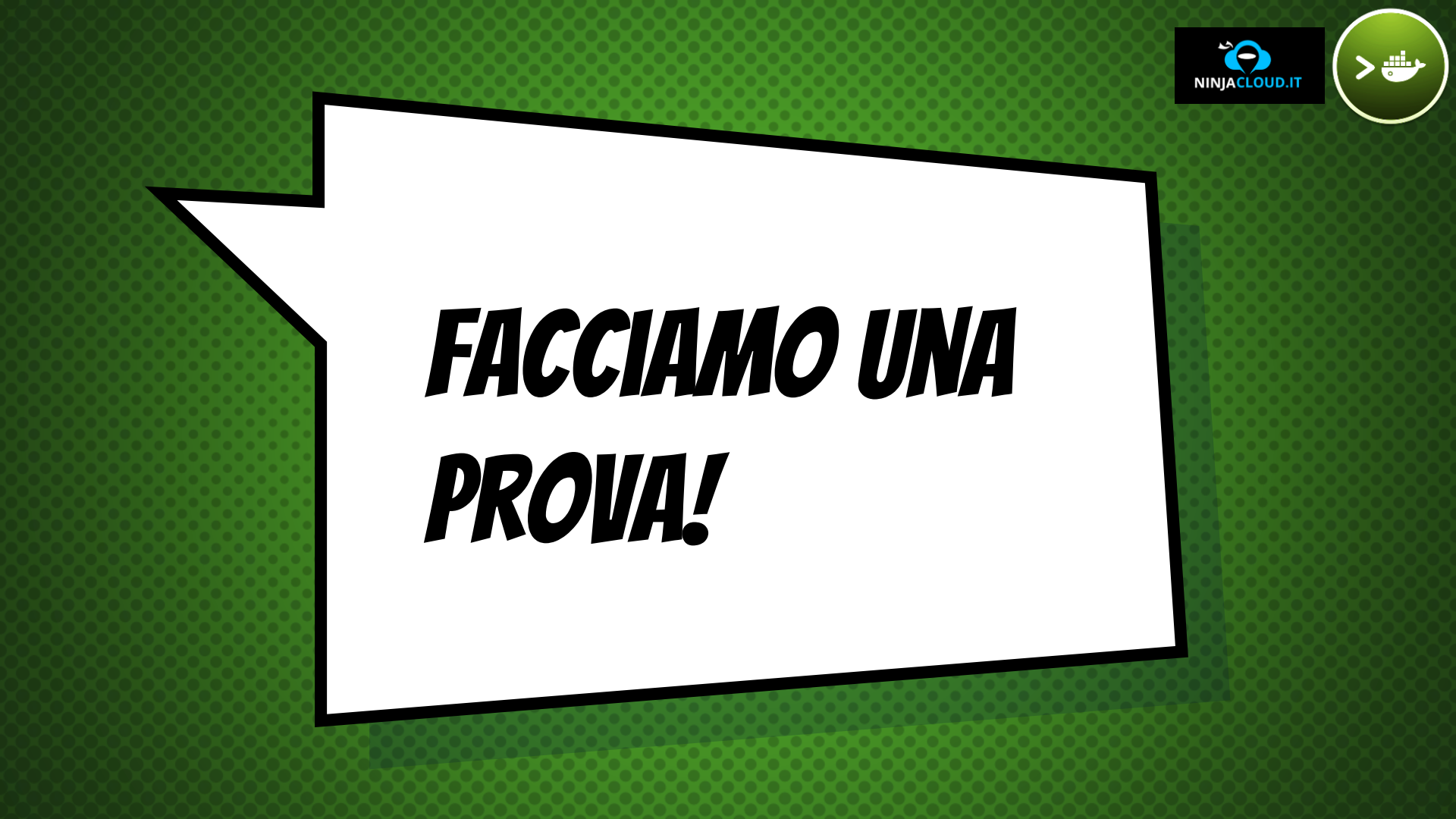
2.

KATACODA

URL –>
https://www.katacoda.com/courses/docker


Katacoda
Learn Docker & Containers
Solve real problems and enhance your skill
Get Started!
Scenarios Completed
0 of 21
Progress
0%
Point
0

SONO PRESENTI ANCHE ALTRI SIMULATORI. NOI UTILIZZEREMO QUESTI DUE NEL SEGUITO DI QUESTO CORSO OLTRE OVVIAMENTE AD UTILIZZARE DOCKER INSTALLATO LOCALMENTE.


NINJACLOUD.IT
FACCIAMO UNA PROVA!