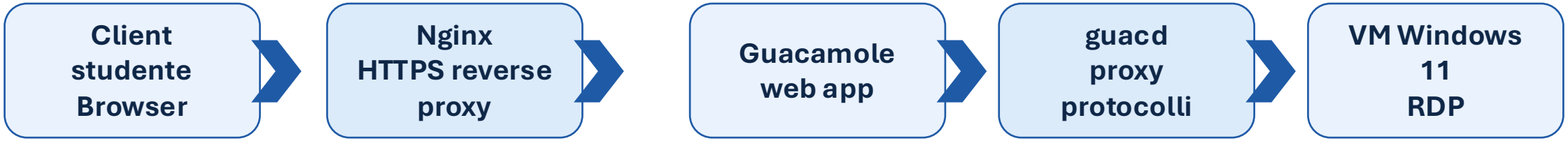
Virtual Desktop Infrastructure persistente con accesso via browser

- Obiettivo: ogni studente accede alla propria VM Windows 11 attraverso Guacamole.
- Le VM sono già create: in aula si evita il provisioning on-demand e si riducono i tempi morti.
- Fedora ospita KVM/libvirt e Docker Compose; Guacamole usa MariaDB per utenti, connessioni e permessi.
- Nel pacchetto trovi PPTX, docker-compose.yml, file .env.example, SQL bootstrap, script di inizializzazione, script VM e configurazione Nginx/HTTPS.

Architettura logica



Chi fa cosa

KVM/libvirt crea e avvia le VM. Guacamole non genera macchine: pubblica web sessioni RDP/SSH/VNC già esistenti.

Perché MariaDB

Il DB conserva utenti, gruppi, connessioni, permessi e configurazioni. Senza schema DB, l'interfaccia parte ma non ha dati da usare.

Contenuto del pacchetto

- 01_pptx/kit_docente_super_parlante_sql.pptx → presentazione con comandi e spiegazione dei comandi.
- 02_compose/docker-compose.yml + .env.example → stack Guacamole, guacd, MariaDB, Nginx.
- 03_sql/guacamole-bootstrap.sql → crea database, utente dedicato e grant minimi.
- 03_sql/init-guacamole-schema.sh → estrae lo schema reale di Guacamole con initdb.sh e lo importa nel DB.
- 04_scripts_vm/create_vms.sh, start_vms.sh, stop_vms.sh, reset_vm.sh → gestione del pool di VM studenti.
- 05_nginx/nginx_guacamole.conf → reverse proxy con terminazione HTTPS.
- README.md → sequenza consigliata di esecuzione in laboratorio.

Prerequisiti sul server Fedora

Aggiornamento sistema

```
sudo dnf update -y
sudo reboot
```

Spiegazione

dnf update aggiorna pacchetti e dipendenze. Il riavvio è consigliato prima di iniziare se il kernel cambia.

Virtualizzazione

```
sudo dnf install -y @virtualization
sudo systemctl enable --now libvirtd
lsmod | grep kvm
```

Spiegazione

@virtualization installa KVM, QEMU, libvirt e strumenti base. systemctl enable --now avvia subito il servizio e lo abilita al boot. lsmod verifica il supporto KVM nel kernel.

Gestione web VM

```
sudo dnf install -y cockpit cockpit-machines
sudo systemctl enable --now cockpit
sudo firewall-cmd --add-service=cockpit --
permanent
sudo firewall-cmd --reload
```

Spiegazione

Cockpit offre una GUI web per creare e osservare VM. In aula è utile per far vedere agli studenti cosa avviene lato host.

Creazione del template Windows 11

- 1. Crea una VM “template-win11” in Cockpit o virt-manager.
- 2. Installa Windows 11, i driver VirtIO, gli aggiornamenti essenziali e abilita Desktop remoto.
- 3. Crea l’utente locale o la configurazione desiderata per il laboratorio.
- 4. Esegui Sysprep solo se vuoi generalizzare il template; in alternativa usa una base già configurata per il corso.
- 5. Spegni il template e non usarlo direttamente: da qui in poi si clonano nuove VM studenti.

Comando utile lato host

```
virsh list --all
virsh dominfo template-win11
```

Perché è importante

Il template è la sorgente unica. Se è pulito e aggiornato, tutte le VM degli studenti saranno coerenti.

Creazione del pool di VM studenti

Clone persistenti

```
for i in {1..10}; do
  virt-clone \
    --original template-win11 \
    --name vm-studente$i \
    --file /var/lib/libvirt/images/vm-
studente$i.qcow2
done
```

Spiegazione comando

virt-clone registra una nuova VM in libvirt, copia/aggancia il disco indicato e crea una macchina pronta a partire. Qui realizza 10 desktop persistenti, uno per studente.

Avvio del pool

```
for i in {1..10}; do
  virsh start vm-studente$i
done
virsh list --all
```

Messaggio da dire in classe

Le macchine non nascono quando lo studente entra: sono già state preparate dal docente o dallo script. Così la lezione è più affidabile.

Installazione Docker e plugin Compose

Installazione

```
sudo dnf install -y docker docker-compose-
plugin
sudo systemctl enable --now docker
sudo usermod -aG docker $USER
```

Spiegazione

docker installa il motore container. docker-compose-plugin abilita il comando moderno “docker compose”. usermod aggiunge il tuo utente al gruppo docker per evitare sudo in ogni comando.

Verifica

```
docker --version
docker compose version
docker ps
```

Risultato atteso

Se docker ps non restituisce errori di permesso, l’ambiente è pronto. Se compare “permission denied”, esegui logout/login o newgrp docker.

docker-compose.yml: struttura seria

Estratto del compose

```
services:
  guacd:
    image: guacamole/guacd
    restart: unless-stopped

  db:
    image: mariadb:11
    env_file: .env
    volumes:
      - db_data:/var/lib/mysql

  guacamole:
    image: guacamole/guacamole
    env_file: .env
    depends_on: [guacd, db]
    ports: ["8080:8080"]
```

Come leggerlo

Ogni servizio ha un ruolo distinto. guacd parla i protocolli remoti. db conserva i dati. guacamole espone la web app. Il volume db_data fa sopravvivere il database ai riavvii dei container.

File .env: credenziali separate dal compose

.env.example

```
MYSQL_ROOT_PASSWORD=CambiareSubito
MYSQL_DATABASE=guacamole_db
MYSQL_USER=guacuser
MYSQL_PASSWORD=PasswordForteQui
TZ=Europe/Rome
```

Perché usare .env

Separi configurazione e segreti dal file compose. In laboratorio è più facile sostituire password e orario senza toccare la struttura dei servizi.

Buona pratica

Nel pacchetto trovi .env.example. Prima dell'avvio lo copi in .env e cambi le password. Non lasciare credenziali deboli in un server esposto in rete.

Il file SQL del DB: cosa contiene davvero

guacamole-bootstrap.sql

```
CREATE DATABASE IF NOT EXISTS guacamole_db
  CHARACTER SET utf8mb4
  COLLATE utf8mb4_unicode_ci;

CREATE USER IF NOT EXISTS 'guacuser'@'%'
  IDENTIFIED BY 'cambiare_subito';

GRANT SELECT,INSERT,UPDATE,DELETE
ON guacamole_db.* TO 'guacuser'@'%';
FLUSH PRIVILEGES;
```

Spiegazione corretta

Questo file NON è l'intero schema di Guacamole. Serve a creare database e utente dedicato. Lo schema vero delle tabelle di Guacamole si genera dal container con initdb.sh.

Import bootstrap

```
docker compose exec -T db \
  mariadb -u root -p"$MYSQL_ROOT_PASSWORD" \
  < 03_sql/guacamole-bootstrap.sql
```

Lo schema reale di Guacamole: initdb.sh

Estrazione schema

```
docker compose exec guacamole \
  /opt/guacamole/bin/initdb.sh --mysql > guacamole-
schema.sql
```

Cosa fa il comando

Esegue lo script ufficiale incluso nell'immagine guacamole/guacamole e scrive su file SQL tutte le tabelle e i dati iniziali necessari all'autenticazione JDBC MySQL/MariaDB.

Import schema nel DB

```
docker compose exec -T db \
  mariadb -u root -p"$MYSQL_ROOT_PASSWORD"
guacamole_db \
  < guacamole-schema.sql
```

Da dire agli studenti

Il bootstrap crea contenitore e utente. initdb.sh crea la struttura logica dell'applicazione: utenti, connessioni, permessi, cronologia e impostazioni.

Avvio ordinato dello stack

Sequenza consigliata

```
cp .env.example .env
docker compose pull
docker compose up -d
docker compose ps
```

Spiegazione

pull scarica le immagini più recenti disponibili. up -d crea e avvia i container in background. ps mostra stato e porte esposte.

Log utili

```
docker compose logs -f guacamole
docker compose logs -f db
```

Quando usarli

Se la pagina non si apre o il login fallisce, i log mostrano quasi sempre il punto esatto: credenziali DB errate, schema assente, container non raggiungibile.

Prima configurazione di Guacamole

- Accedi a http://IP_HOST:8080/guacamole o, meglio, all'URL HTTPS dietro Nginx.
- Login iniziale: guacadmin / guacadmin. Cambia subito la password dell'amministratore.
- Crea un gruppo o utenti separati per gli studenti.
- Per ogni utente configura una connessione RDP verso la sua VM: IP o hostname, porta 3389, username e password Windows.
- Associa ciascuna connessione soltanto allo studente corretto: 1 utente = 1 VM.

Mappatura consigliata utenti ↔ VM

Utente Guacamole	VM	IP/hostname	Note
studente1	vm-studente1	192.168.122.101	Desktop persistente
studente2	vm-studente2	192.168.122.102	Desktop persistente
studente3	vm-studente3	192.168.122.103	Desktop persistente
…	…	…	…

Suggerimento pratico

Usa IP statici o prenotazioni DHCP. In aula riduci il rischio di errori se prepari prima una tabella stampata con studente, VM e indirizzo.

Sicurezza: Nginx davanti a Guacamole

Proxy base

```
server {
  listen 443 ssl http2;
  server_name guac.scuola.local;

  ssl_certificate
/etc/letsencrypt/live/.../fullchain.pem;
  ssl_certificate_key
/etc/letsencrypt/live/.../privkey.pem;

  location / {
    proxy_pass http://127.0.0.1:8080/guacamole/;
    proxy_set_header Host $host;
    proxy_set_header X-Forwarded-Proto https;
    proxy_set_header X-Forwarded-For
$proxy_add_x_forwarded_for;
  }
```

Perché conviene

Nginx termina TLS/HTTPS, semplifica l’esposizione verso la rete scolastica e ti consente in futuro di aggiungere limiti IP, header di sicurezza e certificati veri.

Sicurezza: firewall e superficie esposta

Cockpit e web app

```
sudo firewall-cmd --add-service=cockpit --
permanent
sudo firewall-cmd --add-service=https --
permanent
sudo firewall-cmd --reload
```

Scelta consigliata

Non esporre direttamente MariaDB né la porta di guacd. Guacamole e Nginx bastano. Se il DB è nel compose, resta sulla rete interna Docker.

Verifica porte aperte

```
sudo ss -tulpn
docker compose ps
```

Cosa devi vedere

HTTPS verso Nginx, eventualmente Cockpit su 9090 per amministrazione, e nessuna esposizione pubblica inutile per db o guacd.

Persistenza, snapshot e backup

Snapshot VM

```
virsh snapshot-create-as vm-studente1 pre-
esercitazione
virsh snapshot-list vm-studente1
```

Uso didattico

Se uno studente rompe la configurazione, uno snapshot ti permette di tornare indietro senza ricreare tutta la VM.

Backup database

```
docker compose exec db \
  mariadb-dump -u root -
p"$MYSQL_ROOT_PASSWORD" guacamole_db \
  > backup-guacamole.sql
```

Perché farlo

Il DB contiene utenti e mapping delle connessioni. Se lo perdi, Guacamole si avvia ma la configurazione amministrativa va ricostruita.

Troubleshooting: cosa controllare per primo

- La pagina Guacamole non si apre → controlla docker compose ps, porte e Nginx.
- Login fallisce o schermata vuota → verifica schema DB importato e variabili MYSQL_*.
- La connessione RDP non parte → testa la VM con ping, verifica 3389 e credenziali Windows.
- La VM non esiste o è spenta → virsh list --all, virsh start vm-studenteX.
- Permessi Docker negati → logout/login o uso temporaneo di sudo.

Comandi rapidi

```
docker compose logs -f guacamole
virsh list --all
virsh domifaddr vm-studente1
ping 192.168.122.101
```

Scaletta consigliata per 3 ore

Tempo	Attività	Focus
0:00–0:25	Architettura VDI e ruoli	Chi crea VM, chi pubblica accesso, ruolo DB
0:25–1:05	Fedora host: KVM, Cockpit, pool VM	Host e desktop persistenti
1:05–1:45	Docker Compose, MariaDB, bootstrap e schema	Parte applicativa e DB
1:45–2:20	Configurazione Guacamole utenti/connessioni	Associazione 1:1 studente ↔ VM
2:20–3:00	Test, troubleshooting, verifica	Consolidamento pratico

Messaggio finale da passare agli studenti

“Una VDI non è solo una VM remota: è la combinazione di virtualizzazione, database, rete, sicurezza e pubblicazione del servizio.”

- KVM/libvirt gestisce il desktop vero e proprio.
- Guacamole lo rende accessibile via browser.
- MariaDB conserva configurazione e permessi.
- Nginx/HTTPS protegge l’accesso in rete.
- Le VM persistenti semplificano il laboratorio e valorizzano il lavoro degli studenti.