

Riferimento Rapido Alpine Linux

Gestione pacchetti, servizi, rete, immagine base Docker

Gestione Pacchetti

Basi di apk

```
apk update # aggiorna l'indice dei pacchetti
apk upgrade # aggiorna tutti i pacchetti
apk add curl git vim # installa pacchetti
apk del curl # rimuove un pacchetto
apk search nginx # cerca pacchetti
```

Informazioni sui Pacchetti

```
apk info # lista pacchetti installati
apk info -a nginx # info dettagliate pacchetto
apk info -L nginx # elenca file nel pacchetto
apk policy nginx # mostra versioni disponibili
```

Pacchetti Virtuali

```
# Installa dipendenze di build come gruppo, rimuovibile dopo
apk add --virtual .build-deps gcc musl-dev
make && make install
apk del .build-deps
```

Repository

```
# /etc/apk/repositories
https://dl-cdn.alpinelinux.org/alpine/v3.20/main
https://dl-cdn.alpinelinux.org/alpine/v3.20/community
@edge https://dl-cdn.alpinelinux.org/alpine/edge/testing
```

Servizi

Gestione Servizi OpenRC

```
rc-service nginx start # avvia servizio
rc-service nginx stop # ferma servizio
rc-service nginx restart # riavvia servizio
rc-service nginx status # controlla stato
```

Gestione Runlevel

```
rc-update add nginx default # abilita all'avvio
rc-update del nginx default # disabilita all'avvio
rc-update show # elenca tutti i servizi
rc-status # mostra servizi attivi
```

Runlevel

```
sysinit Inizializzazione sistema (filesystem, orologio)
boot Servizi base (rete, syslog)
default Servizi normali (web server, daemon)
shutdown Task di spegnimento
```

Configurazione

File di Configurazione Principali

```
/etc/apk/repositories URL dei repository pacchetti
/etc/hostname Nome host del sistema
/etc/network/interfaces Configurazione interfacce di rete
/etc/conf.d/ Configurazione specifica per servizio
/etc/motd Messaggio del giorno
```

Configurazione di Sistema

```
setup-alpine # configurazione guidata completa
setup-timezone # imposta fuso orario
setup-keymap # configura layout tastiera
setup-hostname myhost # imposta nome host
```

Fuso Orario

```
apk add tzdata
cp /usr/share/zoneinfo/US/Eastern /etc/localtime
echo "US/Eastern" > /etc/timezone
apk del tzdata # opzionale: rimuovi per risparmiare spazio
```

Rete

Configurazione Interfaccia

```
# /etc/network/interfaces
auto eth0
iface eth0 inet dhcp
# --- statico ---
iface eth0 inet static
address 192.168.1.10/24
gateway 192.168.1.1
```

Comandi di Rete

```
ip addr show # mostra indirizzi IP
ip route show # mostra tabella di routing
ip link set eth0 up # attiva interfaccia
setup-interfaces # configurazione rete guidata
```

DNS e Firewall

```
# DNS: /etc/resolv.conf
nameserver 1.1.1.1
nameserver 8.8.8.8
# Firewall
apk add iptables
iptables -L -n # elenca regole
```

Utenti

Gestione Utenti

```
adduser alice # crea utente (interattivo)
adduser -D -s /bin/sh bob # non interattivo, imposta shell
deluser alice # elimina utente
passwd alice # imposta/cambia password
```

Gruppi e Sudo

```
addgroup devs # crea gruppo
addgroup alice devs # aggiunge utente al gruppo
apk add doas # alternativa leggera a sudo
# /etc/doas.conf
permit persist alice as root
```

Utenti di Sistema

```
adduser -S -D -H -s /sbin/nologin myapp
# -S utente sistema -D nessuna password
# -H nessuna home -s nessuna shell
```

Disco e Storage

Comandi Filesystem

```
df -h # riepilogo utilizzo disco
du -sh /var/log # dimensione directory
lsblk # elenca dispositivi a blocchi
mount /dev/sda1 /mnt # monta dispositivo
umount /mnt # smonta
```

LBU (Alpine Local Backup)

```
# Per modalità diskless/data - persistere i cambiamenti tra riavvii
lbu status # mostra modifiche non salvate
lbu commit # salva modifiche sul supporto di avvio
lbu list # elenca file di backup
lbu include /etc/myconf # aggiunge percorso al backup
```

Configurazione Disco

```
setup-disk # installazione disco guidata
setup-disk /dev/sda # installa su disco specifico
# Modalità: sys (tradizionale), data, diskless
```

Immagine Base Docker

Perché Alpine per Docker

```
~5 MB base image vs ~80 MB per Debian slim
musl libc Più piccolo di glibc (alcuni problemi di compatibilità)
apk package manager Veloce, nessuna cache predefinita
Superficie di attacco minima Meno pacchetti = meno CVE
```

Dockerfile Minimale

```
FROM alpine:3.20
RUN apk add --no-cache python3 py3-pip
COPY app.py /app/
CMD ["python3", "/app/app.py"]
```

Build Multi-Stage

```
FROM golang:1.22-alpine AS builder
WORKDIR /src
COPY . .
RUN go build -o /app
FROM alpine:3.20
COPY --from=builder /app /app
CMD ["/app"]
```

Problemi Comuni

```
--no-cache Usare sempre per mantenere l'immagine piccola
musl vs glibc Alcuni binari richiedono il pacchetto gcompat
Nessuna bash predefinita Usare /bin/sh o installare bash con apk
Fuso orario mancante Installare tzdata se necessario
```

Pattern Comuni

Installare Strumenti di Build

```
apk add --no-cache build-base # gcc, make, ecc.
apk add --no-cache python3-dev # header Python
apk add --no-cache linux-headers # header kernel
```

Cron Job

```
# Aggiunge un cron job
echo "*/5 * * * * /usr/local/bin/task.sh" \
| crontab -
rc-service crond start
rc-update add crond default
```

Abilitare SSH

```
apk add openssh
rc-service sshd start
rc-update add sshd default
# Config: /etc/ssh/sshd_config
```

Aggiornare Versione Alpine

```
# Modifica /etc/apk/repositories: cambia v3.19 -> v3.20
apk update
apk upgrade --available
sync && reboot
```