ALPINE LINUX クイックリファレンス

パッケージ管理、サービス、ネットワーク、Docker ベースイメージ

パッケージ管理

apk 基本操作

```
apk update              # パッケージインデックスを更新
apk upgrade             # すべてのパッケージをアップグレード
apk add curl git vim    # パッケージをインストール
apk del curl            # パッケージを削除
apk search nginx        # パッケージを検索
```

パッケージ情報

```
apk info                # インストール済みパッケージを一覧表示
apk info -a nginx       # パッケージの詳細情報
apk info -L nginx       # パッケージ内のファイルを一覧表示
apk policy nginx        # 利用可能なバージョンを表示
```

仮想パッケージ

```
# ビルド依存をグループとしてインストールし、後で削除
apk add --virtual .build-deps gcc musl-dev
make && make install
apk del .build-deps
```

リポジトリ

```
# /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
```

サービス

OpenRC サービス管理

```
rc-service nginx start    # サービスを起動
rc-service nginx stop     # サービスを停止
rc-service nginx restart  # サービスを再起動
rc-service nginx status   # ステータスを確認
```

ランレベル管理

```
rc-update add nginx default   # 起動時に有効化
rc-update del nginx default   # 起動時に無効化
rc-update show                # すべてのサービスを一覧表示
rc-status                     # 実行中のサービスを表示
```

ランレベル

`sysinit`	システム初期化（ファイルシステム、クロック）
`boot`	基本システムサービス（ネットワーク、syslog）
`default`	通常サービス（Web サーバー、デーモン）
`shutdown`	シャットダウンタスク

設定

主要な設定ファイル

`/etc/apk/repositories`	パッケージリポジトリ URL
`/etc/hostname`	システムのホスト名
`/etc/network/interfaces`	ネットワークインターフェース設定
`/etc/conf.d/`	サービス固有の設定
`/etc/motd`	ログインメッセージ

システムセットアップ

```
setup-alpine              # 対話形式のフルセットアップ
setup-timezone            # タイムゾーンを設定
setup-keymap              # キーボードレイアウトを設定
setup-hostname myhost     # ホスト名を設定
```

タイムゾーン

```
apk add tzdata
cp /usr/share/zoneinfo/US/Eastern /etc/localtime
echo "US/Eastern" > /etc/timezone
apk del tzdata          # オプション: 容量節約のため削除
```

ネットワーク

インターフェース設定

```
# /etc/network/interfaces
auto eth0
iface eth0 inet dhcp
# --- 静的アドレス ---
iface eth0 inet static
  address 192.168.1.10/24
  gateway 192.168.1.1
```

ネットワークコマンド

```
ip addr show              # IPアドレスを表示
ip route show             # ルーティングテーブルを表示
ip link set eth0 up     # インターフェースを有効化
setup-interfaces          # 対話形式のネット設定
```

DNS とファイアウォール

```
# DNS: /etc/resolv.conf
nameserver 1.1.1.1
nameserver 8.8.8.8
# ファイアウォール
apk add iptables
iptables -L -n            # ルールを一覧表示
```

ユーザー

ユーザー管理

```
adduser alice             # ユーザーを作成（対話形式）
adduser -D -s /bin/sh bob # 非対話形式、シェルを指定
deluser alice             # ユーザーを削除
passwd alice              # パスワードを設定・変更
```

グループと sudo

```
addgroup devs             # グループを作成
addgroup alice devs       # ユーザーをグループに追加
apk add doas              # 軽量な sudo 代替
# /etc/doas.conf
permit persist alice as root
```

システムユーザー

```
adduser -S -D -H -s /sbin/nologin myapp
# -S システムユーザー  -D パスワードなし
# -H ホームディレクトリなし  -s シェルなし
```

ディスクとストレージ

ファイルシステムコマンド

```
df -h                     # ディスク使用量の概要
du -sh /var/log           # ディレクトリサイズ
lsblk                     # ブロックデバイスを一覧表示
mount /dev/sda1 /mnt      # デバイスをマウント
umount /mnt               # マウント解除
```

LBU（Alpine ローカルバックアップ）

```
# ディスクレス/データモード — 再起動後も変更を保持
lbu status                # コミットされていない変更を表示
lbu commit                # 起動メディアに変更を保存
lbu list                  # バックアップ済みファイルを一覧表示
lbu include /etc/myconf   # バックアップにパスを追加
```

ディスクセットアップ

```
setup-disk                # 対話形式のディスクインストール
setup-disk /dev/sda       # 特定のディスクにインストール
# モード: sys（従来型）、data、diskless
```

Docker ベースイメージ

Alpine を Docker に使う理由

`~5 MB ベースイメージ`	Debian slim の ~80 MB と比較
`musl libc`	glibc より小さい（互換性の問題あり）
`apk パッケージマネージャー`	高速、デフォルトでキャッシュなし
`最小限の攻撃面`	パッケージが少ない = CVE が少ない

最小 Dockerfile

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

マルチステージビルド

```
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"]
```

よくある注意点

`--no-cache`	イメージを小さく保つために常に使用する
`musl vs glibc`	一部のバイナリには gcompat パッケージが必要
`bash がデフォルトでない`	/bin/sh を使うか apk add bash で追加
`タイムゾーンが未設定`	必要に応じて tzdata をインストール

よくあるパターン

ビルドツールのインストール

```
apk add --no-cache build-base   # gcc、make など
apk add --no-cache python3-dev  # Python ヘッダー
apk add --no-cache linux-headers # カーネルヘッダー
```

cron ジョブ

```
# cron ジョブを追加
echo "*/5 * * * * /usr/local/bin/task.sh" \
  | crontab -
rc-service crond start
rc-update add crond default
```

SSH の有効化

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

Alpine バージョンのアップグレード

```
# /etc/apk/repositories を編集: v3.19 → v3.20 に変更
apk update
apk upgrade --available
sync && reboot
```