Struttura server

Partendo da un fissato root path, al esempio `/livenet`, vediamo la struttura del server. E' una descrizione minimale, che verrà approfondita nel seguito.

```
drwxr-xr-x  9 root root 4096 Nov 18 14:51 .
drwxr-xr-x 23 root root 4096 Oct 24 08:24 ..
drwxr-xr-x  7 root root 4096 Oct  9 09:00 aux
drwxr-xr-x  8 root root 4096 Nov 13 12:38 .git
drwxr-xr-x 10 root root 4096 Oct 31 13:25 images
drwxr-xr-x  4 root root 4096 Nov 13 11:57 overlay
drwxr-xr-x 11 root root 4096 Oct 31 08:16 releases
drwxr-xr-x  2 root root 4096 Nov 13 12:06 scripts
drwxrwxr-x 29 root root 4096 Oct  2 15:20 vms
-rw-r--r--  1 root root  408 Sep 23 09:12 .env
-rw-r--r--  1 root root  405 Aug 27 08:48 env.sample
-rw-r--r--  1 root root  449 Sep 24 15:07 .gitignore
-rw-r--r--  1 root root    0 Sep 23 09:12 .init_ok
-rw-r--r--  1 root root 2156 Aug 27 08:48 README.md
-rw-r--r--  1 root root 5167 Sep 18 16:59 Taskfile.yml
```

- `aux` contiene tutto ciò che non è direttamente collegato ad una immagine livenet. Ad esempio, il collettore dei dati, chè è una utility, ma la cui assenza non pregiudica il funzionamento del sistema.
- `images` contiene le definizioni dei vari dipartimenti, quindi files, scripts, etc...
- `overlay` è la directory che contiene la definizione minimale di livenet, ovvero tutto ciò che è comune ai vari dipartimenti e che altrimenti dovrebbe essere replicato in ciascuno
- `releases` contiene le releases dei vari dipartimenti
- `scripts` sono gli script di gestione del sistema che vengono richiamati dai task
- `vms` mantiene tutte le macchine virtuali, essenzialmente windows, che possono essere avviate nei client.
- `.env` è il file di configurazione generale dei vari dipartimenti
- `Taskfile.yml` contiene tutte le descrizioni dei task possibili

aux

Contiene tutto ciò che non è essenziale al funzionamento di livenet, principalmente utilità di sistema.

- Container collettore dati
- Container syslog
- Script `post-merge` per sistemare i permessi critici ad ogni `git pull`

images

Ogni subdirectory, della quale riporto solo gli elementi di interesse, ha di base la seguente struttura:

```
drwxr-xr-x 1 kreen kreen  74 27 set 08.03 conf
drwxr-xr-x 1 kreen kreen  60 11 set 13.19 overlay.all
```

```
drwxr-xr-x 1 kreen kreen  12 12 set 17.47 overlay.local
drwxr-xr-x 1 kreen kreen  24  7 ott 16.09 runtime
drwxr-xr-x 1 kreen kreen  68 27 set 08.03 scripts
drwxr-xr-x 1 root  root    0  5 ott 12.08 setup
-rw-r--r-- 1 kreen kreen 489 24 ott 09.49 docker-compose.yml
-rw-r--r-- 1 kreen kreen 767 27 set 08.03 Dockerfile
-rw-r--r-- 1 kreen kreen 394  8 ott 09.21 .env
```

- `conf` contiene file di configurazione, ad esempio la lista dei pacchetti specifici dell'immagine
- `overlay.all` è la directory che contiene la configurazione comune a tutte le istanze
- `overlay.local` è la directory con contiene la configurazione specifica per l'istanza, per esempio i profili, e ha la precedenza su `overlay.all`
- `runtime` è usata in sviluppo, non ha impiego in produzione
- `scripts` sono gli script di gestione per la costruzione dell'immagine
- `setup` contiene file e pacchetti che vengono installati nell'immagine (per esempio *.deb)
- `docker-compose.yml` dell'immagine
- `Dockerfile` dell'immagine
- `.env` variabili ambiente specifiche dell'immagine

Oerlay

Questa directory contiene tutto ciò che va inserito nell'immagine durante la costruzione e i commit. Viene fatta una copia 1:1 nella root dell'immagine docker. Evidenzio cheil contenuto è comune a tutte le immagini, viene fatto un *bind* in `/etc/fstab` con `./images/$NOMEIMMAGINE/overlay.all` per aggirare i limiti di docker, che non consente di accedere alle directory superiori.

releases

Ogni sotto directory contiene gli archivi tgz delle immagini da installare e degli upgrade. Viene fatto un bind di volume nel `docker-compose.yaml` dell'immagine `live` ed esportato in sola lettura dal server NFS della stessa.

scripts

Contiene gli script:

* `build.sh` che costruisce una full image di un dipartimento * `commit.sh` crea una *diff* ovvero un archivio incrementale e relativa immagine docker, partendo da un container in esecuzione e agendo sulle sole differenze rispetto all'immagine da cui è stato creato il container.

.env

* Unordered List ItemVariabili ambiente generali

- `ROOT_PATH=/xyz` la radice del progetto, usata come directory basename
- `IMG_PATH="${ROOT_PATH}/images"` è la radice che contiene le definizioni dei dipartimenti

- `REL_PATH="${ROOT_PATH}/releases"` punta alla radice delle releases
- `VMS_PATH="${ROOT_PATH}/vms"` la radice del repository delle macchine virtuali
- `LIVENET_SERVER=192.168.1.10` è l'ip dell'host dove è in esecuzione il server livenet
- `LIVENET_PORT=2049` la porta utilizzata dal server NFS. Tipicamente la 2049, ma può essere modificata.
- `SYSLOG_SERVER=192.168.1.10` autoesplicativo
- `SYSLOG_PORT=513`
- `_docker="docker"` container runtime, che può essere anche `podman`
- `_compose="docker compose"`
- `DEVELOP=0` usato in sviluppo, se messo a `1` nell'immagine non viene copiato il software non necessario (ad esempio matlab). usato nel `Dockerfile`