Università degli Studi di Camerino

Scuola di Scienze e Tecnologie

Corso di Laurea in Informatica (Classe L-31)

FARAWAY
Appliance automatizzato per l'accesso allo smart working

Laureando
Jacopo Rizzo

Matricola 105132

Relatore
Fausto Marcantoni

Correlatore
Samuele Cacchiarelli

A.A. 2020/2021

Indice

Elenco delle figure

1. Introduzione

FARAWAY è un appliance, inteso come macchina hardware atta a fornire un'unica risorsa computazionale, che si propone con lo scopo di semplificare ed automatizzare il processo di transizione dal lavoro in ufficio allo smart-working.

In questo capitolo illustreremo le finalità che hanno guidato la realizzazione del progetto e la sua struttura dal punto di vista hardware e software. Nel resto della tesi verranno poi approfonditi i singoli argomenti in modo individuale.

1.1 Finalità del progetto

Come i recenti tempi di pandemia hanno evidenziato, in molti casi quando si presenta l'esigenza di introdurre lo smart-working in azienda in modo tempestivo, spesso le infrastrutture e le conoscenze tecniche del personale non sono sufficientemente adeguate a garantire il funzionamento del lavoro da casa in modo veloce, accessibile e soprattutto sicuro.

Da qui l'idea di un applicativo che consenta di superare tutte queste difficoltà come la configurazione di un'adeguata infrastruttura al livello di rete (VPN) o la gestione degli accessi da parte degli amministratori aziendali.

1.2 Schema di collegamento

Uno dei vantaggi principali nell'utilizzo di FARAWAY sta nella sua caratterizzazione "Plug&Play", ossia una soluzione pronta che riduce al minimo l'entità degli interventi di tipo sistemistico e di configurazione da effettuare per garantirne il corretto funzionamento.

Come visibile dalla figura infatti, l'hardware che contiene l'applicativo si può collegare alla rete locale dell'azienda come un normale PC, e mette subito a disposizione l'interfaccia di gestione per gli amministratori, dalla quale è possibile inizializzare la VPN, registrare i PC aziendali, gli utenti che li utilizzeranno in Desktop Remoto e altre funzionalità che verranno descritte in seguito.
L'unico intervento necessario che comporta la modifica dell'infrastruttura di rete dell'azienda consiste nell' apertura di una porta nel firewall del gateway, consentendo così l'accesso al server VPN dall'esterno della rete locale.

Il sistema prevede poi la possibilità per gli utenti di controllare l'accensione e lo spegnimento dei pc a cui hanno accesso, sfruttando la tecnologia Wake-On-Lan quando possibile (non sempre viene supportata dall'hardware di rete), o in alternativa attraverso degli attuatori remoti (o "smart plug") collegati alla rete wi-fi dell'azienda. Que-

sti attuatori comunicano con il broker MQTT presente nell'appliance, e sono quindi controllabili tramite l'interfaccia web.

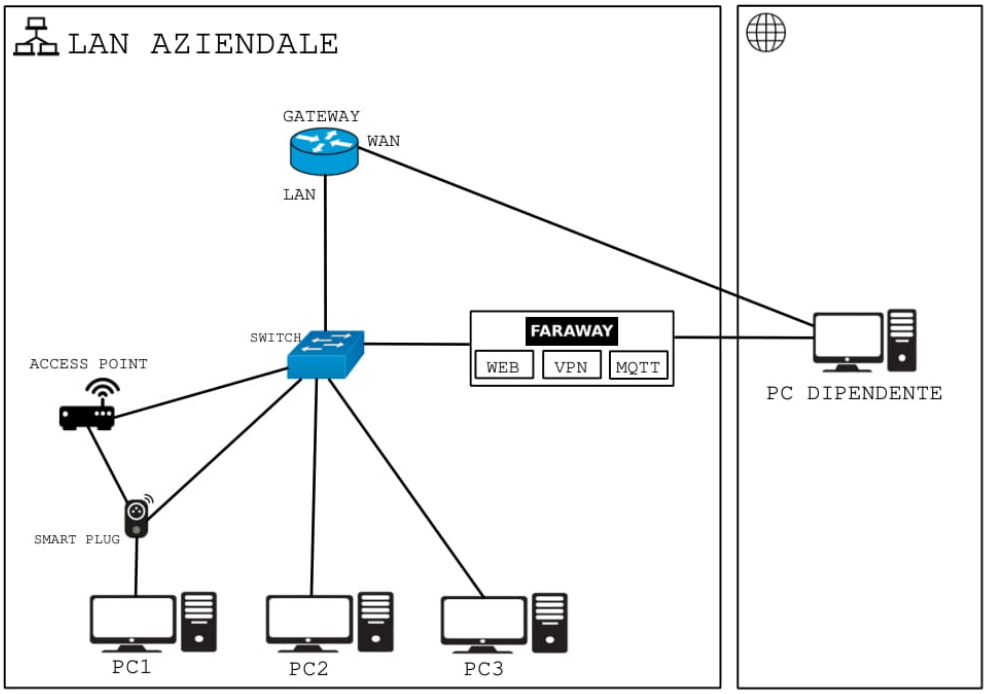


Figura 1.1: Schema di collegamento

1.3 Componenti

Il termine "appliance" indica un'apparecchio che nasce dalla combinazione di hardware, sistema operativo e software applicativo pre-installati e pre-configurati in modo da rendere minimo il lavoro di integrazione da parte degli utenti.
Di seguito sono quindi elencate e categorizzate le componenti del sistema.

1.3.1 Hardware

La macchina su cui viene eseguito l'applicativo non necessità di una potenza computazionale elevata, basti pensare che in fase di test è stato utilizzato un RaspberryPi4 e non ci sono stati problemi di funzionamento. I requisiti minimi necessari sono quindi il supporto del sistema operativo Ubuntu e la presenza di una sola interfaccia di rete fisica.

Per quanto riguarda gli attuatori remoti è sufficiente scegliere dei dispositivi che supportino la configurazione manuale del protocollo MQTT, poiché molte case produttrici ne supportano solo l'utilizzo attraverso cloud e app proprietaria. Nell'implementazione sono state utilizzate delle prese smart Shelly Plug S [1].

1.3.2 Software

Nel progetto vengono impiegati diversi componenti software sia di alto che basso livello, tra cui i principali:

- *Ubuntu 20.04 LTS*: come sistema operativo
- *Django web framework [2]*: framework per il linguaggio Python con cui è stato scritto l'applicativo web che gestisce le interazioni con le varie funzionalità
- *G*unicorn [3]: web server che gestisce le richieste http da passare all'applicativo.
- *N*ginx [4]: web server utilizzato per il servizio dei file statici.
- *O*penVPN [5]: soluzione open source per implementare il server VPN
- *Eclipse Mosquitto* [6]: broker mqtt che si occupa della comunicazione con gli attuatori remoti
- *I*ptables [7], firewall Linux con cui vengono configurate le rotte raggiungibili da ogni utente
- *N*etplan [8], utlity per la configurazione di rete in ambiente Linux.

1.4 Struttura della tesi

La tesi si articola nei seguenti capitoli:

- *Capitolo 2*: fornisce una descrizione delle funzionalità messe a disposizione degli utenti, divisi tra amministratori e utenti ordinari.
- *Capitolo 3*: approfondisce il funzionamento del sistema al livello di rete
- *Capitolo 4*: approfondisce tecnologie e protocolli utilizzati nell'implementazione
- *Capitolo 5*: descrive le problematiche affrontate e ulteriori sviluppi futuri

2. Descrizione e funzionalità

FARAWAY espone all'utente finale diverse funzionalità, come la registrazione di utenti e dispositivi, la gestione del server VPN e delle relative chiavi di accesso e la comunicazione con gli attuatori remoti. Queste e altre funzionalità sono messe a disposizione tramite l'interfaccia web, che si propone come interfaccia unificata per la gestione dei vari servizi.
L'interfaccia si divide in due sottosezioni, "manager" e "client", che consentono rispettivamente di accedere alle funzioni amministratore e alle funzioni utente, descritte dettagliatamente di seguito.

2.1 Funzioni amministratore

All'utente amministratore (inteso come figura amministrativa dell'azienda) è consentito modificare ogni aspetto del sistema, dagli utenti e i dispositivi registrati all'impostazione dei parametri per il funzionamento dei servizi di rete.

2.1.1 Gestione utenti

Entrando nell'interfaccia con le credenziali di amministratore, è possibile registrare nuovi utenti del sistema, specificandone nome, indirizzo e-mail e password. E' necessario inoltre specificare se l'utente che si sta creando è un amministratore o un utente ordinario, in modo da restringerne adeguatamente i privilegi.
Una volta creato il nuovo utente, l'amministratore potrà visualizzare e modificarne i dettagli (come i PC a cui l'utente ha accesso e le chiavi VPN che ha associate) accedendo alla lista utenti e scegliendo l'utente specifico da gestire.

2.1.2 Gestione dispositivi

Attraverso la sezione "computer" dell'interfaccia amministratore è possibile aggiungere nuovi PC e visualizzare la lista di quelli già registrati. Alla registrazione di un nuovo PC, è necessario specificare alcuni parametri quali:

- Un nome identificativo
- Indirizzo MAC
- Indirizzo IP
- Possibilità di accendere il pc tramite Wake-on-LAN

E' possibile anche specificare, al momento della registrazione, gli utenti che devono avere accesso a quel dispositivo, scegliendoli tra quelli già registrati. Questa operazione potrà comunque essere svolta in seguito in qualsiasi momento.

Figura 2.1: Schermata di gestione dei PC

2.1.3 Configurazione dei dispositivi raggiungibili

Accedendo, tramite la lista dei PC, alla pagina di dettaglio di uno specifico dispositivo è possibile modificare la lista degli utenti che vi hanno accesso. L'operazione di aggiunta di un utente alla lista di accesso si traduce, all'interno dell'applicativo, nella creazione di due regole di routing.
La prima riguarda il firewall di Linux ip-tables, nel quale viene impostato il NAT verso l'ip del PC da raggiungere.
La seconda invece consiste nel mettere a conoscenza il client VPN che l'ip locale del PC sarà raggiungibile utilizzando il server VPN come gateway. Per fare ciò vengono modificati i file contenuti nella cartella "ccd" (client-configuration-directory) del server VPN associati alle chiavi possedute dall'utente.

In modo speculare, è anche possibile accedere alla pagina di dettaglio di un utente e specificare la lista dei PC a cui deve avere accesso.

2.1.4 Configurazione di rete

La configurazione di rete predefinita dell'appliance prevede l'utilizzo del DHCP, ma è possibile modificare questo comportamento accedendo alla sezione "Rete" dell'interfaccia. Questo consentirà di disattivare il DHCP e impostare manualmente i parametri di configurazione come indirizzo IP statico, mascheratura di rete, indirizzo del server DNS da utilizzare e indirizzo del gateway.
La modifica della configurazione di rete viene poi gestita dall'applicativo, che provvede a generare un file di configurazione in formato .yaml contenente i parametri specificati. La configurazione generata viene applicata sfruttando l'interazione dell'applicazione con l'utility Netplan.

Figura 2.2: Schermata di configurazione della rete

2.1.5 Gestione della VPN

Uno dei punti cruciali del funzionamento del sistema è la gestione del servizio di VPN. Al momento dell'installazione della macchina, il servizio non sarà inizializzato. Infatti, se si tenta di accedere a una delle funzionalità associate alla VPN, l'interfaccia comunicherà all'utente la necessità di inizializzare il server, chiedendo di specificare l'IP pubblico della rete e un numero di porta per il server (la porta predefinita è la 1194). La procedura automatizzata di inizializzazione del server si articola nei seguenti passi:

1. Creazione della directory "pki", all'interno del path "/etc/openvpn", la quale contiene tutte le sottocartelle dove verranno memorizzati tutti i certificati, chiavi e file di configurazione generati tramite FARAWAY.
2. Generazione di due coppie di chiavi RSA, rispettivamente per la Certification Authority e per il server stesso.
3. Generazione dei certificati della Certification Authority e del server, firmati con le corrispondenti chiavi private.
4. Creazione del file "server.conf", contente la configurazione del server che verrà letta ad ogni avvio.
5. Generazione del file contenente i parametri di Diffie-Hellman, necessario al server per consentire una comunicazione sicura.
6. Inizializzazione di un file CRL (Certificate Revocation List), che contiene i numeri seriali di tutti i certificati revocati. Questo file inizialmente è vuoto (non contiene alcun numero seriale) e viene modificato con le operazioni di revoca delle chiavi.

Una volta terminata questa procedura, sarà possibile avviare il server VPN dal pannello di configurazione, visualizzarne lo stato, modificare ulteriormente i parametri specificati in fase di inizializzazione, e generare o revocare nuove chiavi di accesso. E' inoltre possibile avviare la procedura di eliminazione del server, che consiste nella cancellazione di tutte le chiavi e i certificati, al termine della quale il sistema permetterà una nuova inizializzazione.

2.1.6 Creazione/revoca delle chiavi

Quando il server VPN è inizializzato, si rendono disponibili le operazioni di creazione o revoca delle chiavi, accedendo alla sezione "VPN" dell'interfaccia.
Per creare una nuova chiave di accesso, è necessario specificare l'utente a cui la chiave è associata e una password (questa dovrà essere inserita dal client al momento della connessione). La procedura automatizzata di creazione della chiave si articola nei seguenti passi:

1. Generazione di una nuova coppia di chiavi RSA per il client
2. Generazione del certificato del client, firmato con la chiave privata della Certification Authority
3. Creazione del file di configurazione in formato .ovpn, questo formato viene utilizzato prevalentemente dai client Windows, e consiste nella concatenazione inline dei file necessari alla connessione, ovvero: i parametri di configurazione, il certificato della Certification Authority, il certificato del server e la chiave privata del client.

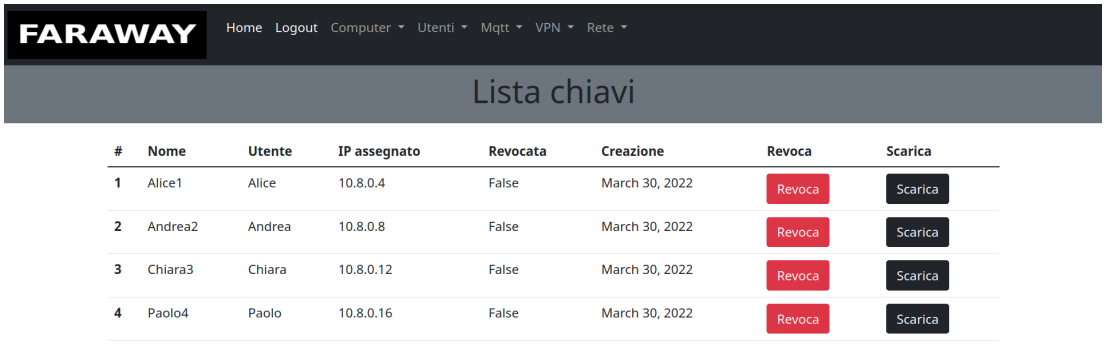


Figura 2.3: Schermata di gestione delle chiavi attive

Le chiavi generate possono essere revocate in ogni momento da un amministratore accedendo alla lista delle chiavi attive e scegliendo l'opzione "Revoca". Quando una chiave viene revocata, il numero seriale del certificato corrispondente viene aggiunto alla Certificate Revocation List del server e non sarà quindi più accettato da quest'ultimo al momento della connessione.

```
client
dev tun
proto udp
remote 192.168.1.43 1194
resolv-retry infinite
nobind
persist-key
persist-tun
ca [inline]
cert [inline]
key [inline]
cipher AES-256-CBC
verb 3
<ca>
-----BEGIN CERTIFICATE-----
MIIDRjCCAi6gAwIBAgIUAwmQhDExvzMujZK+pfho8cGDAsswDQYJKoZIhvcNAQEL
...
pAkuu+DM6/rbgSMpCxuxsd++VrYv9AZkBAMFtbEQEwLDF99k7VyeoxsX+ZkWttP5
ZF/pFt4D1qgxDWXfDJpQKO/wtX5ub+3qCgE=
-----END CERTIFICATE-----
</ca>
<cert>
-----BEGIN CERTIFICATE-----
MIIDMzCCAhugAwIBAgIULMuV9yszVqEke9yQqHjJP3k0pykwDQYJKoZIhvcNAQEL
...
2bclIkfs50lqWQFrI3J0WrQoFSzeUYJo6SFtGnPgBYDwQzzOaJjBMneo7ln1eoB9
gkEa2atzXw==
-----END CERTIFICATE-----
</cert>
<key>
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-256-CBC,4FC2D4ED4C5800E74CC2A46FFF2ACB9D

kpwu1kdPuxXQdIjyAtn3J0QN+W8hRpKhfQOlvW2nRcP2E06lYMN9W2qv7O1QIUIy
...
hJYNb7x9LtM0QFpd81sRJrSXKJk8kCmYm9b2/7oqQLSEqexsAy25+yLXWl5kY4Op
-----END RSA PRIVATE KEY----
</key>
```

Figura 2.4: File .ovpn generato dall'applicativo

2.1.7 Associazione dei dispositivi MQTT

Per sfruttare le funzioni di accensione e spegnimento dei PC tramite attuatore remoto, è innanzitutto necessario registrare questi dispositivi specificandone un nome identificativo e l'ip del broker MQTT che devono raggiungere (di default l'ip del broker è lo stesso dell'appliance, ma questa configurazione è stata lasciata aperta per consentire sviluppi futuri). Il nome identificativo serve a definire il nome del topic a cui l'applicativo invierà i messaggi MQTT per interagire con l'attuatore.
Una volta registrato il dispositivo, è possibile associarlo ad un PC per consentirne l'accensione remota.

2.2 Funzioni utente

Le funzioni disponibili agli utenti ordinari sono molto limitate rispetto a quelle degli amministratori e mirano a consentire e semplificare la connessione di un dipendente ai pc dell'azienda dall'ambiente domestico. Di seguito sono descritte le principali operazioni che un utente ordinario può svolgere.

2.2.1 Controllo stato PC

Ogni utente può consultare la lista dei PC a cui gli è consentito accedere. Scegliendo un particolare PC, è possibile visualizzarne l'indirizzo IP che permette la connessione tramite Desktop Remoto una volta connessi alla VPN. E' inoltre possibile all'utente controllare lo stato di accensione del proprio PC (questo viene determinato a seconda della risposta al ping) e, quando previsto, accenderlo e spegnerlo sfruttando il Wake-on-LAN o gli attuatori remoti.

2.2.2 Accesso alle chiavi VPN

Alla sezione "VPN" dell'interfaccia utente viene messa a disposizione una lista delle chiavi VPN associate al dipendente, con la possibilità di effettuare il download del file di configurazione in formato .ovpn che consente la connessione.
Sarà quindi sufficiente scaricare il file e importarlo nel client di OpenVPN per essere connessi alla VPN aziendale e raggiungere i PC utilizzando gli stessi indirizzi della LAN.

2.3 Esempio di funzionamento

Supponendo che un amministratore (per comodità verrà chiamato *Admin*) abbia la necessità di consentire all'utente *Alice* l'accesso al pc *Amministrazione*, una tipica storia utente sull'utilizzo del sistema può essere descritta come segue:

1. *Admin* registra l'utente *Alice* all'interno dell'applicativo.
2. *Admin* registra (se non è già registrato) il pc *Amministrazione* specificandone l'ip.
3. *Admin* aggiunge *Alice* alla lista degli utenti che hanno accesso al PC *Amministrazione*.
4. *Admin* crea una chiave VPN per l'utente *Alice*, specificando anche una password per l'utilizzo da comunicare in modo sicuro.
5. *Alice* accede all'interfaccia web con le sue credenziali e scarica la chiave VPN che troverà tra le chiavi disponibili.
6. *Alice* si connette alla VPN utilizzando la chiave appena scaricata e il client OpenVPN.
7. *Alice* può ora avviare il software Desktop Remoto e accedere al pc *Amministrazione* specificandone l'ip che viene mostrato dall'interfaccia di FARAWAY.

3. Networking

Per consentire un corretto funzionamento del sistema, è stato necessario applicare diverse metodologie di networking durante il progetto. Le principali riguardano il routing, il firewall e la traduzione degli indirizzi di rete. In questo capitolo verranno descritte le soluzioni adottate al livello di rete per rendere la comunicazione funzionale e sicura.

3.1 Linux come router

I sistemi basati su kernel linux forniscono software nativi per la configurazione e la gestione del routing. Per questo, viste le funzionalità che svolge, possiamo attribuire all'appliance progettato anche i ruoli di router e firewall, poiché si occupa di stabilire le rotte specifiche per i client e di gestire l'instradamento dei pacchetti dalla VPN verso la rete locale.
Per svolgere queste operazioni, sono stati coinvolti nel sistema il firewall di Linux iptables, lo strumento di configurazione di rete Netplan e altre metodologie descritte di seguito.

3.1.1 Iptables

Iptables [7] è un firewall da riga di comando installato in modo predefinito sui sistemi Ubuntu. Consente di configurare le regole di filtraggio dei pacchetti e interagisce in modo diretto con il firewall al livello kernel di Linux.
Le regole di filtraggio sono organizzate in diverse tabelle (*tables*), ognuna delle quali contiene una serie di catene (*chains*) di regole da applicare ai pacchetti in transito. Ogni tabella è associata ad una diversa procedura di filtraggio. I pacchetti in entrata vengono processati in modo sequenziale attraverso le regole concatenate in una tabella. Ogni pacchetto che viene ricevuto o inoltrato dalla macchina router attraversa almeno una catena.
Nel progetto vengono coinvolte principalmente due tabelle di iptables: "nat" e "filter". La prima si occupa delle operazioni di mascheramento dei pacchetti e contiene le seguenti catene:

- INPUT, catena che gestisce i pacchetti in entrata che andranno consegnati localmente.
- OUTPUT, pacchetti inviati dalla macchina stessa verso l'esterno.
- POSTROUTING, i pacchetti attraversano questa catena quando le scelte di routing sono già state prese e il pacchetto sta per essere consegnato ai layer sottostanti.

Per quanto riguarda la tabella "filter" invece, questa si occupa propriamente delle operazioni di filtraggio dei pacchetti, e oltre alle catene INPUT e OUTPUT, utilizza anche la catena FORWARD, che viene attraversata dai pacchetti ricevuti dalla macchina che devono essere instradati verso l'esterno.

3.1.2 Netplan

Netplan [8] è un'utility impiegata nei sistemi Linux per la configurazione agile della rete. Durante il funzionamento, legge i diversi file in formato .yaml localizzati nella sua directory e genera dei file di configurazione da passare ai backend sottostanti che interagiscono col kernel (solitamente NetworkManager o networkd).

Netplan viene utilizzato nel progetto durante la modifica della configurazione di rete dell'appliance. Quando si modificano i parametri, l'applicativo provvede a generare un nuovo file di configurazione temporaneo e lancia il comando di sistema "netplan apply" per assicurarsi che la configurazione non contenga errori.
Se non vengono generate eccezioni, il file temporaneo viene scritto nella cartella di configurazione di netplan sostituendo il precedente, in modo da rendere persistenti le modifiche effettuate.

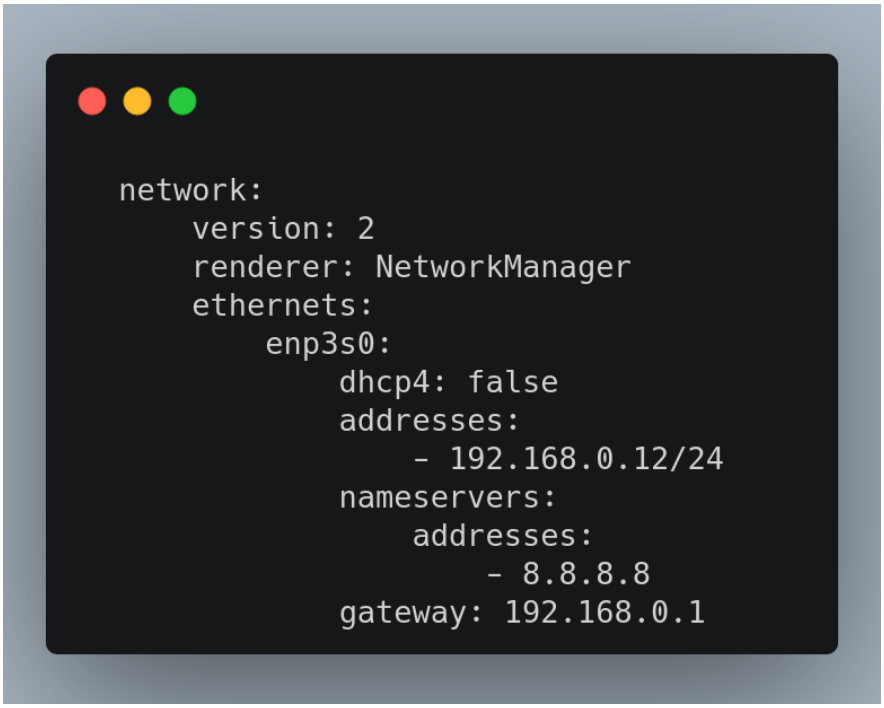


Figura 3.1: File generato dall'applicazione in caso di configurazione statica

3.1.3 N.A.T. - Network Address Translation

Il NAT (Network Address Translation) è una metodologia che permette di mappare tra loro spazi di indirizzi IP modificando l'intestazione dei pacchetti IP quando transitano attraverso il router. E' diventata uno strumento essenziale per la conservazione degli indirizzi pubblici e ne evita l'esaurimento.
Nel progetto è stato utilizzato il NAT per permettere ad un host della rete virtuale di poter comunicare con un PC aziendale, poiché questo non è possibile in modo diretto a causa delle diverse classi di IP.

Come funziona il NAT

Ogni pacchetto TCP o UDP contiene nei propri header un numero di porta sorgente e un numero di porta destinazione. Queste è altre informazioni vengono poi incapsulate in un pacchetto IP che a sua volta contiene un indirizzo ip sorgente e uno di destinazione. La tripla indirizzo/protocollo/porta definisce quella che viene chiamata una network socket.

Utilizzando il NAT, ogni volta che un pacchetto viene inviato verso una rete esterna a quella locale, il suo ip sorgente (cioè l'ip privato della LAN) viene sostituito con l'ip del dispositivo che sta effettuando il NAT prima di essere inoltrato all'esterno. Viene anche modificata la porta sorgente, assegnandone una presa da una pool di porte libere.
Il dispositivo che si occupa del NAT procede quindi a registrare nella sua tabella di traduzione la tripla costituita da ip locale, porta originale e porta tradotta. Procede quindi a inviare il pacchetto verso l'esterno.
Quando invece viene ricevuto un pacchetto proveniente da una rete esterna, il dispositivo di NAT analizza la sua tabella di traduzione per trovare un match con il numero di porta di destinazione del pacchetto. Identificata la tripla corretta, provvede a sostituire l'ip e la porta di destinazione con quelli presenti nella tabella, e il pacchetto viene inoltrato all'host locale.

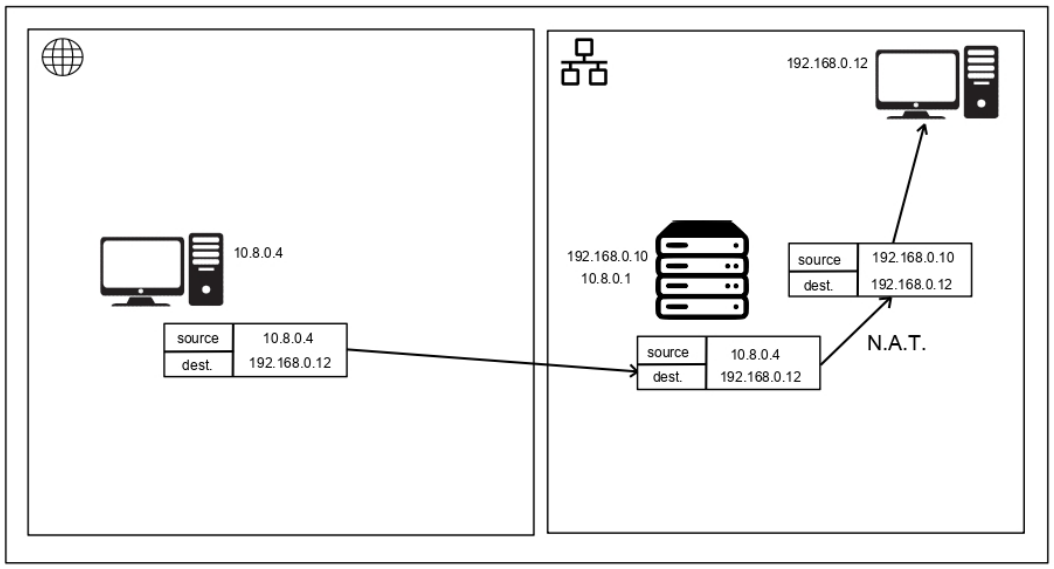


Figura 3.2: Funzionamento del NAT sull'appliance

3.2 Routing dei pacchetti

Le principali configurazioni di routing nel progetto vengono effettuate dal server OpenVPN. Queste operazioni si rendono necessarie nel momento in cui viene dato l'accesso a un utente ad un determinato PC. Oltre ad aggiungere, al livello di applicazione, il PC tra quelli visualizzabili dall'utente, è necessario configurare la rete affinché possa avvenire la comunicazione tra i due endpoint.

Supponiamo che all'utente Alice venga dato l'accesso al PC "amministrazione" con indirizzo ip 192.168.0.10 attraverso l'interfaccia di FARAWAY. Supponiamo inoltre che Alice possegga una chiave VPN generata per lei dal suo amministratore e che a tale chiave sia stato assegnato l'indirizzo 10.8.0.4 all'interno della VPN.
Le procedure eseguite dall'applicativo FARAWAY dietro le quinte sono due: una riguarda l'impostazione della rotta specifica sul client VPN di Alice e l'altra il mascheramento dei pacchetti IP provenienti dal PC di Alice e diretti verso il PC "amministrazione".

3.2.1 OpenVPN - Specifica della rotta

Per far sì che il pc di Alice raggiunga il pc "amministrazione" quando i pacchetti in uscita hanno come destinazione l'indirizzo 192.168.0.10, è necessario che il client VPN imposti una rotta specifica sul pc di Alice. In questo modo, quando quest'ultimo dovrà indirizzare un pacchetto all'indirizzo 192.168.0.10 (che è un indirizzo locale) saprà che il gateway incaricato dell'instradamento è il server VPN, e non il gateway predefinito.

Al livello pratico questa operazione si traduce nella modifica del file ccd associato alla chiave di alice all'interno del server. Come visibile in figura, nel file ccd sono indicate le rotte specifiche assegnate alla chiave VPN a cui fa riferimento.

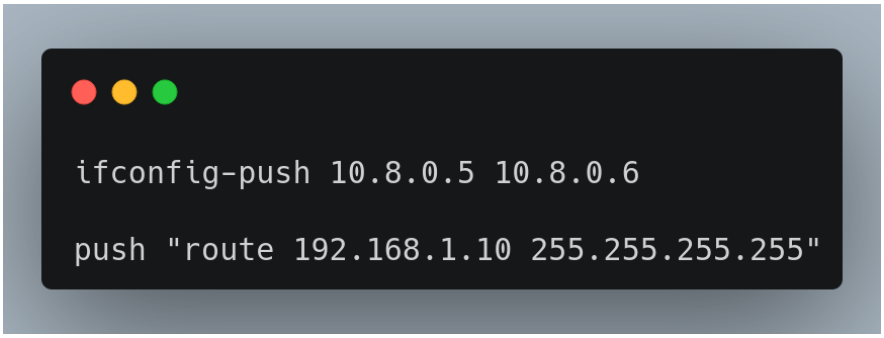


Figura 3.3: File ccd generato per una chiave specifica

In fase di connessione del client, il server VPN comunicherà al client le nuove rotte, e il PC di Alice modificherà la sua tabella di routing aggiungendo le nuove regole come visibile a titolo di esempio in figura 3.4.

3.2.2 Aggiunta delle regole su iptables

La sola operazione di modifica della rotta sul PC di Alice non è sufficiente a consentire una comunicazione funzionante. Per fare ciò bisogna implementare il mascheramento dei pacchetti da parte dell'appliance quando questi sono diretti ad un PC della rete aziendale.

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         192.168.1.1     0.0.0.0         UG    100    0        0 enp0s31f6
10.0.3.0        0.0.0.0         255.255.255.0   U     0      0        0 lxcbr0
10.8.0.1        10.8.0.6        255.255.255.255 UGH   0      0        0 tun0
10.8.0.6        0.0.0.0         255.255.255.255 UH    0      0        0 tun0
link-local      0.0.0.0         255.255.0.0     U     1000   0        0 lxcbr0
192.168.1.0     0.0.0.0         255.255.255.0   U     100    0        0 enp0s31f6
192.168.1.80    10.8.0.6        255.255.255.255 UGH   0      0        0 tun0
```

Figura 3.4: Modifica della rotta sul PC client

Quando viene generato un nuovo accesso quindi, oltre che alla modifica del file ccd, l'applicativo procede alla riscrittura del file di configurazione di iptables, aggiungendo ad esso due nuove regole come in figura 3.5.

```
nat
:PREROUTING ACCEPT [0:0]
:INPUT ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]
-A POSTROUTING -s 10.8.0.4/32 -d 192.168.1.80/32 -j MASQUERADE
COMMIT
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A FORWARD -s 10.8.0.4/32 -d 192.168.1.80/32 -j ACCEPT
COMMIT
```

Figura 3.5: File di configurazione generato con le nuove regole

La prima regola visibile in figura, fa sì che ad ogni pacchetto proveniente dall'indirizzo 10.8.0.4/32 venga modificato l'indirizzo ip sorgente, sostituendolo con quello dell'appliance che appartiene alla stessa rete locale del PC da raggiungere. La seconda regola invece permette il passaggio dei pacchetti provenienti dall'ip VPN di alice e diretti verso il pc "amministrazione".

4. Tecnologie e protocolli utilizzati

La realizzazione del progetto ha richiesto l'approfondimento dal punto di vista teorico di alcuni argomenti, in particolare riguardo il networking, la crittografia e la comunicazione tra dispositivi IoT.
Nel presente capitolo ho scelto di approfondire quelli che sono ritenuti più rilevanti e attuali, come la gestione del server VPN, il protocollo TLS e le metodologie di controllo dello stato di accensione di una macchina.

4.1 Virtual Private Network

Le VPN sono un elemento chiave del progetto svolto, poiché consentono agli utenti di collegarsi alla LAN aziendale dall'esterno in modo sicuro. L'implementazione di un modello in grado di gestire in modo automatizzato la Virtual Private Network ha coperto buona parte del tempo di sviluppo impiegato per l'appliance, a causa dell'alto numero di parametri da gestire e del non banale sistema di criptografia a chiave pubblica utilizzato. Seguono dettagli teorici sull'argomento e sulla sua caratterizzazione nel progetto.

4.1.1 Come funziona e perché utilizzarla

Per dare una definizione generale, le Virtual Private Network (VPN) rappresentano un modo di estendere una rete privata per consentire agli utenti di accedere e scambiare dati attraverso reti pubbliche esterne come se fossero direttamente connessi alla LAN. Le risorse interne dell'azienda risulteranno quindi accessibili da remoto, rendendo la criptografia un elemento chiave nell'infrastruttura.

Le VPN utilizzano meccanismi di tunneling (movimento di dati attraverso le reti sfruttando l'incapsulamento) e sistemi crittografici per garantire l'autenticazione, l'integrità e la confidenzialità dei dati trasportati.

I tipi di implementazione di una VPN, descritti in figura, sono principalmente due:

- SSL VPN, utilizzata tipicamente (come nel nostro caso) quando gli impiegati di una compagnia hanno bisogno di accedere ai propri PC in ufficio collegati alla LAN aziendale. E' utilizzata quindi per connettere un singolo client ad una nuova rete privata dall'esterno.
- Site-to-Site VPN, utilizzata per raggruppare tra loro reti private distanti geograficamente, come ad esempio le diverse sedi di un azienda.

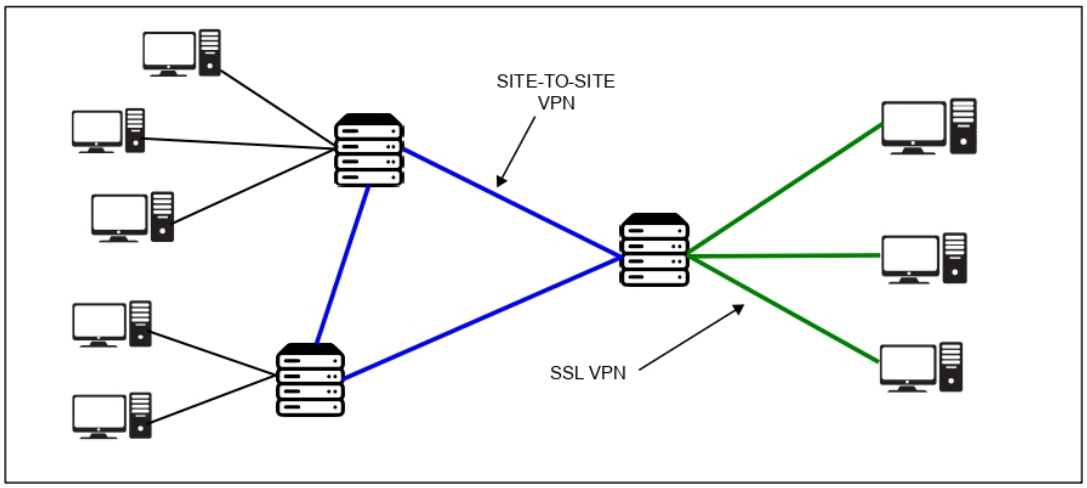


Figura 4.1: SSL VPN e Site-to-Site VPN

Il funzionamento viene gestito da un server VPN facente parte della rete private da raggiungere. Nel caso specifico è stato utilizzato OpenVPN server.
Quando i dati sono trasferiti attraverso la rete privata virtuale, vengono sempre criptati utilizzando una combinazione tra un meccanismo a chiave pubblica (nel nostro caso il protocollo TLS) e un meccanismo di crittografia a chiave simmetrica pre-condivisa, che impedisce la lettura dei dati a soggetti esterni anche nel caso in cui vengano intercettati.

4.1.2 OpenVPN

OpenVPN [5] è un applicativo open source per l'implementazione di server e client VPN originariamente sviluppata da James Yonan, oggi co-founder e CTO della OpenVPN Technologies Inc.
Il software si vuole proporre come soluzione universale e flessibile, fornisce infatti il supporto ai protocolli SSL/TLS per la sicurezza, tunneling attraverso TCP o UDP utilizzando proxy o NAT, scalabilità e supporto alla configurazione dinamica degli host. E' inoltre un software con una forte portabilità, disponibile per tutti i maggiori sistemi operativi, anche su mobile.

OpenVPN è strettamente legata alla libreria OpenSSL [9], che rappresenta un'implementazione dei protocolli di sicurezza SSL e TLS, e viene utilizzata per le funzioni crittografiche. Sono supportati due principali metodi di crittografia convenzionali:

- *Crittografia a chiave segreta pre-condivisa*, che prevede l'impiego di una sola chiave scambiata inizialmente tra due utenti su canale sicuro. Viene quindi utilizzato un meccanismo di crittografia simmetrica per permettere di codificare e decodificare i messaggi con la stessa chiave.
- *Crittografia a chiave pubblica*, che invece prevede l'impiego di una coppia di chiavi per ogni utente. Le chiavi private di ogni utente devono rimanere segrete, mentre quelle pubbliche possono essere condivise. Questo sistema consente, oltre alla protezione dei messaggi, anche un forte sistema di autenticazione, applicabile firmando un messaggio con la propria chiave privata.

Al livello di rete, OpenVPN utilizza dispositivi di rete virtuali TUN/TAP presenti sulla maggior parte delle piattaforme. Questi sono interfacce di rete supportate interamente da software, senza una corrispondente interfaccia fisica. I dispositivi TUN e TAP sono entrambi dispositivi progettati per gestire il tunneling, ma se ne può combinare l'utilizzo poiché operano a livelli diversi dello stack di rete:

- *TUN (network tunnel)*, simula un dispositivo del livello di rete (layer 3) per trasportare pacchetti IP. Viene utilizzato per gestire gli aspetti di routing.
- *TAP (o network TAP)*, simula invece un dispositivo del livello di collegamento (layer 2), trasportando frame ethernet.

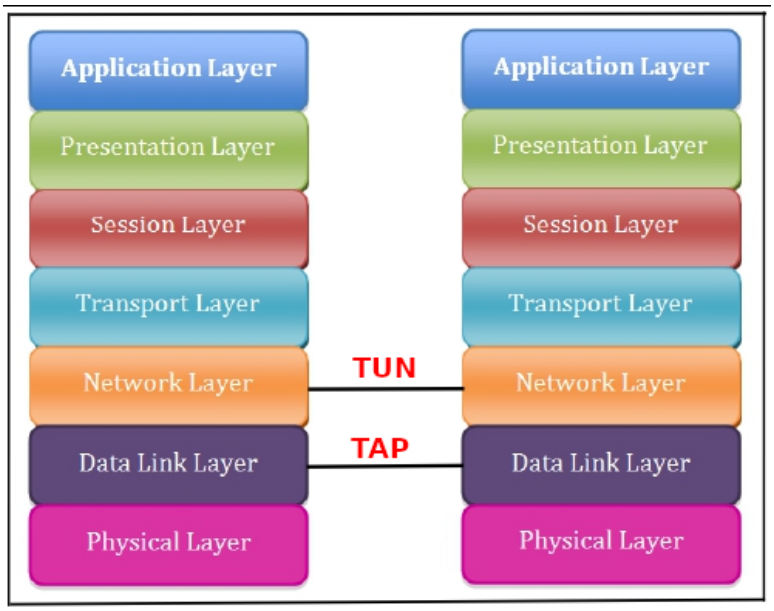


Figura 4.2: TUN e TAP nello stack di rete

OpenVPN consente di configurare i vari parametri del server sia da riga di comando all'avvio che attraverso la scrittura di un file di configurazione, necessario per l'avvio in background come servizio di sistema.

La configurazione del server deve contenere diverse informazioni essenziali, tra le quali ricordiamo le principali:

- Numero di porta da utilizzare (la porta di default è la 1194)
- Specifica del protocollo (TCP o UDP)
- Nome dell'interfaccia virtuale (*tun0* in figura)
- I path dei file necessari al funzionamento (chiave privata del server, certificato, parametri di Diffie-Hellman e lista dei certificati revocati).
- Spazio degli indirizzi ip da utilizzare per la configurazione dei client
- Path della directory contente le rotte specifiche per ogni client

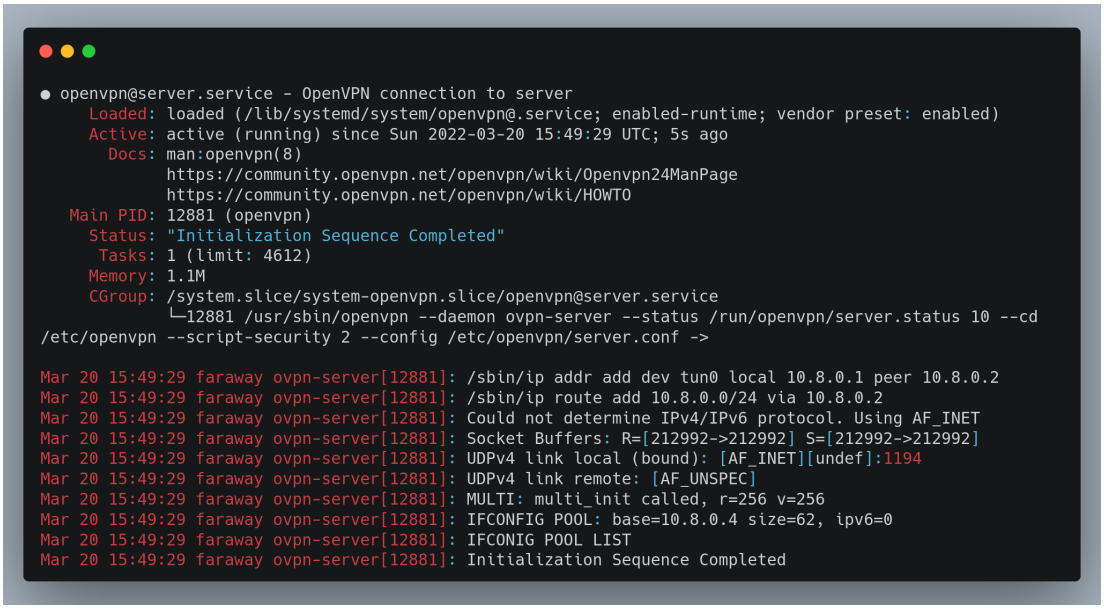


Figura 4.3: Stato del servizio OpenVPN

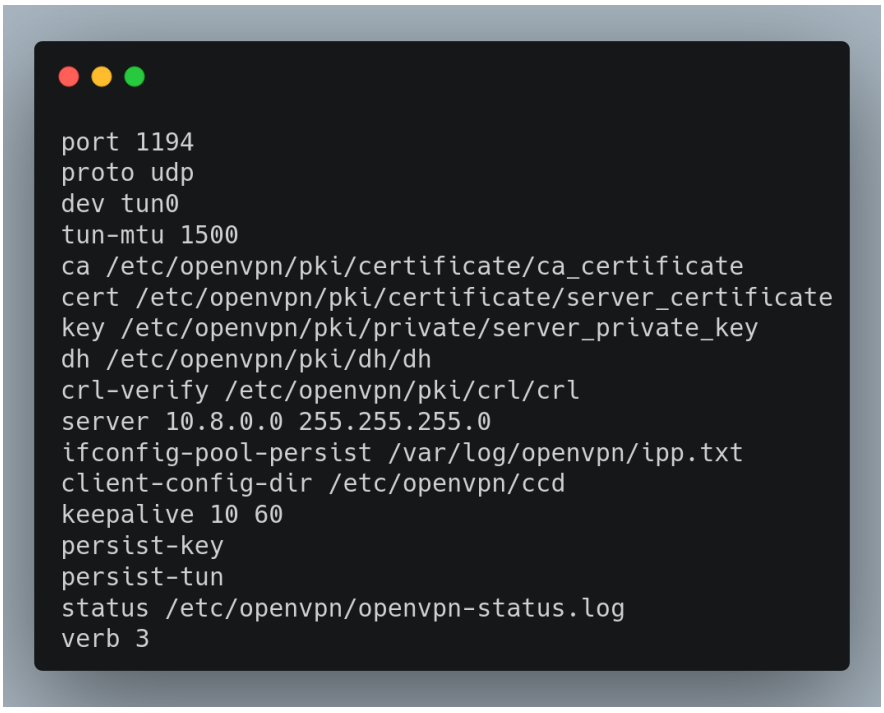


Figura 4.4: File di configurazione del server

4.2 SSL/TLS

TLS [10] (Transport Layer Security) rappresenta la principale tecnologia di crittografia utilizzata da OpenVPN. E' il successore dell'ormai deprecato protocollo SSL (Secure Sockets Layer) ed è un protocollo crittografico che mira a fornire comunicazioni sicure su una rete di computer. E' ampiamente utilizzato in app di messaggistica e telefonia, ma il suo campo di utilizzo più diffuso rimane quello del protocollo HTTPS.

Lo standard è stato proposto dall'Internet Engineering Taskforce (IEETF) nel 1999 basandosi sulle precedenti specifiche dello standard SSL.

Il protocollo TLS opera al livello di applicazione (layer 7) e raggiunge i suoi scopi di privacy, integrità e autenticazione attraverso l'utilizzo di particolari certificati che vengono scambiati tra i comunicatori e firmati da un terzo soggetto fidato, la Certification Authority.

4.2.1 Certification Authority e certificati digitali

I certificati digitali utilizzati dal protocollo TLS sono documenti informatici che certificano l'appartenenza di una chiave pubblica al soggetto specificato nel documento stesso. Questo consente al possessore del certificato di far risultare affidabili le firme da lui poste con la chiave privata corrispondente a quella pubblica contenuta nel certificato.

E' necessaria quindi la presenza di un terzo organismo che abbia la fiducia di tutte le parti coinvolte e che possa certificare la legittimità delle chiavi. Questo ruolo è ricoperto dalle Certification Authorities (CA), che si occupano di generare e firmare i certificati oltre che a gestirne la validità nel tempo.

Quando il possessore di una chiave necessita della certificazione da parte della CA, provvede ad emettere una Certificate Signing Request (CSR), ossia un documento che contiene:

- Chiave pubblica del client richiedente la certificazione.
- Dati del richiedente (nome organizzazione, dominio, nazione, common name e altri).
- Firma digitale applicata con la chiave privata del richiedente, per preservare l'integrità e l'autenticità della richiesta.

La richiesta viene poi inviata alla Certification Authority, che può verificarne l'autenticità, estrarre la chiave pubblica del richiedente ed utilizzarla per emettere un certificato che ne assicura la legittimità, firmato con la propria chiave privata.

```python
def generate_server_certificate():

    with open(f"{PRIVATE_KEY_PATH}/server_private_key", "rb") as key_file:
        server_private_key = serialization.load_pem_private_key(key_file.read(), password=None)
        server_public_key = server_private_key.public_key()
    with open(f"{PRIVATE_KEY_PATH}/ca_private_key", "rb") as key_file:
        ca_private_key = serialization.load_pem_private_key(key_file.read(), password=None)
    subject = x509.Name({
        x509.NameAttribute(NameOID.COUNTRY_NAME, "IT"),
        x509.NameAttribute(NameOID.STATE_OR_PROVINCE_NAME, "IT"),
        x509.NameAttribute(NameOID.LOCALITY_NAME, "IT"),
        x509.NameAttribute(NameOID.ORGANIZATION_NAME, "serverbusiness"),
        x509.NameAttribute(NameOID.COMMON_NAME, "server"),})
    issuer = x509.Name({
        x509.NameAttribute(NameOID.COUNTRY_NAME, "IT"),
        x509.NameAttribute(NameOID.STATE_OR_PROVINCE_NAME, "IT"),
        x509.NameAttribute(NameOID.LOCALITY_NAME, "IT"),
        x509.NameAttribute(NameOID.ORGANIZATION_NAME, "organizzazione"),
        x509.NameAttribute(NameOID.COMMON_NAME, "authority"),})
    cert = x509.CertificateBuilder().subject_name(subject
    ).issuer_name(issuer
    ).serial_number(x509.random_serial_number()
    ).not_valid_before(datetime.datetime.now() - datetime.timedelta(days=10)
    ).not_valid_after(datetime.datetime.now() + datetime.timedelta(days=10)
    ).public_key(server_public_key
    ).sign(ca_private_key, hashes.SHA256())
    with open(f"{CERTIFICATE_PATH}/server_certificate", "wb") as file:
        file.write(cert.public_bytes(serialization.Encoding.PEM))
```

Figura 4.5: Codice Python per la generazione del certificato del server

4.2.2 Funzionamento

Il protocollo TLS viene utilizzato dalle applicazioni client-server per assicurare la privacy dei dati e la loro integrità durante la comunicazione. Dato che le applicazioni possono usare o meno questa tecnologia, è necessario che il client richieda, al momento della connessione, l'utilizzo del protocollo TLS. Questo viene effettuato di solito riservando delle porte specifiche alla connessione protetta, ad esempio la porta 80 viene comunemente utilizzata per il traffico HTTP in chiaro mentre la 443 viene usata per il traffico criptato su HTTPS.
Quando client e server hanno entrambi aderito all'utilizzo di TLS, inizia la procedura di handshake:

1. Il client presenta al server una lista di cifrari da lui supportati.
2. Il server sceglie un cifrario supportato da entrambi e una funzione hash e informa il client della decisione.
3. Il server fornisce il suo certificato digitale al client, contenente la sua chiave pubblica e la firma della Certification Authority fidata.
4. Il client verifica la validità del certificato.
5. Il server utilizza l'algoritmo Diffie-Hellman per la generazione di una chiave random associata alla sessione.

La chiave di sessione generata è di tipo simmetrico, viene quindi utilizzata da entrambe le parti sia per criptare che per decriptare i dati, ma ha la caratteristica di avere

una segretezza pro-attiva: se la chiave venisse resa nota in futuro, non potrebbe essere utilizzata per decriptare i dati di una nuova sessione diversa da quella per la quale la chiave stessa è stata generata.

Quando una connessione tra client e server viene resa sicura attraverso TLS, possiamo dire che:

- La connessione è *privata*, perché coinvolge l'utilizzo di un algoritmo di crittografia a chiave simmetrica e viene generata una nuova chiave per ogni connessione.
- La connessione è *autenticata*, attraverso il sistema di crittografia a chiave pubblica
- La connessione è *affidabile*, perché ogni messaggio contiene un controllo di integrità per prevenire la perdita o l'alterazione dei dati.

4.2.3 X.509

X.509 [11] è uno standard definito dall'International Telecommunication Union (ITU) che stabilisce le tipologie e il formato dei certificati digitali su chiave pubblica, utilizzati anche dal protocollo TLS e dai sistemi di firma digitale.
Un certificato X.509 associa tra loro un'identità e una chiave pubblica. Chi è in possesso del certificato può utilizzare quindi la chiave pubblica per stabilire connessioni sicure o per validare i documenti emessi firmandoli con la chiave privata corrispondente a quella pubblica.
I certificati X.509 si dividono in certificati della CA e certificati delle entità finali ("end-entity" certificate). I certificati della CA sono autorizzati ad emettere e firmare altri certificati, cosa che non è possibile con l'altra tipologia. Il certificato generato dalla Certification Authority e da lei stessa firmato, è quello di più alto livello e viene chiamato "root certificate". Partendo dal certificato radice vengono poi emessi gli altri.

Nella definizione dello standard viene utilizzato un linguaggio formale, Abstract Syntax Notation One (ASN.1), per la definizione della struttura che i certificati devono assumere:

- Numero di versione
- Numero seriale del certificato
- ID dell'algoritmo di firma
- Nome dell'emittente
- Periodo validità (inizio e fine)
- Nome del subject (proprietario della chiave pubblica)
- Dati della chiave pubblica (algoritmo e chiave)
- (Opzionale) Estensioni
- Algoritmo di firma
- Firma del certificato

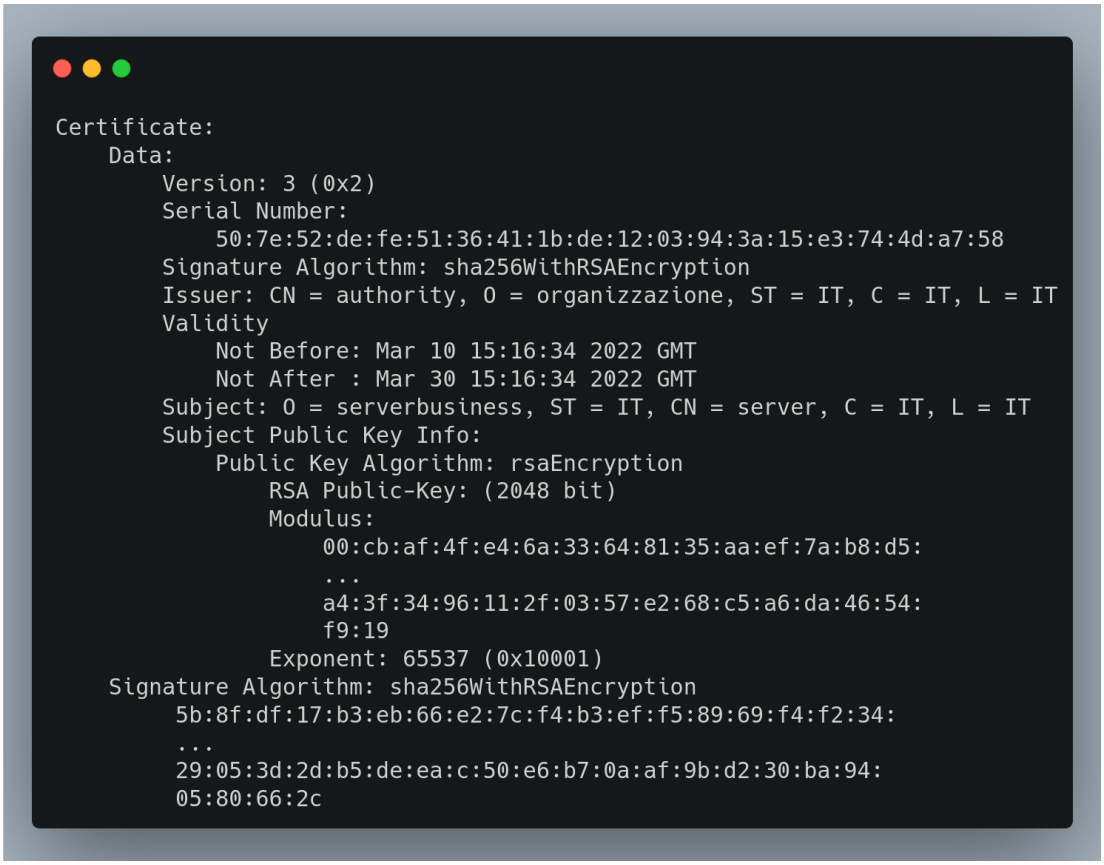
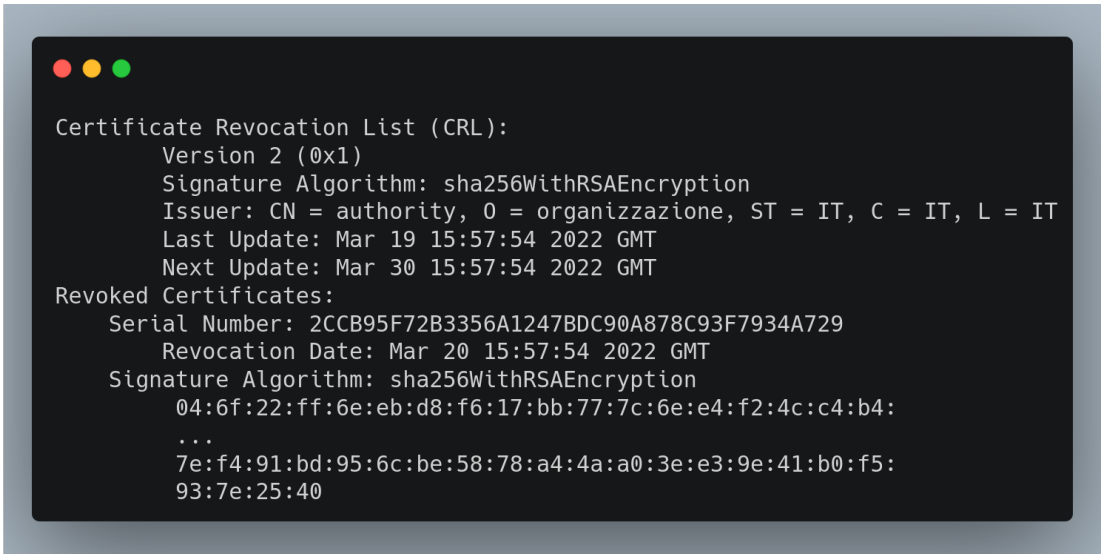


Figura 4.6: Certificato del server firmato dalla CA

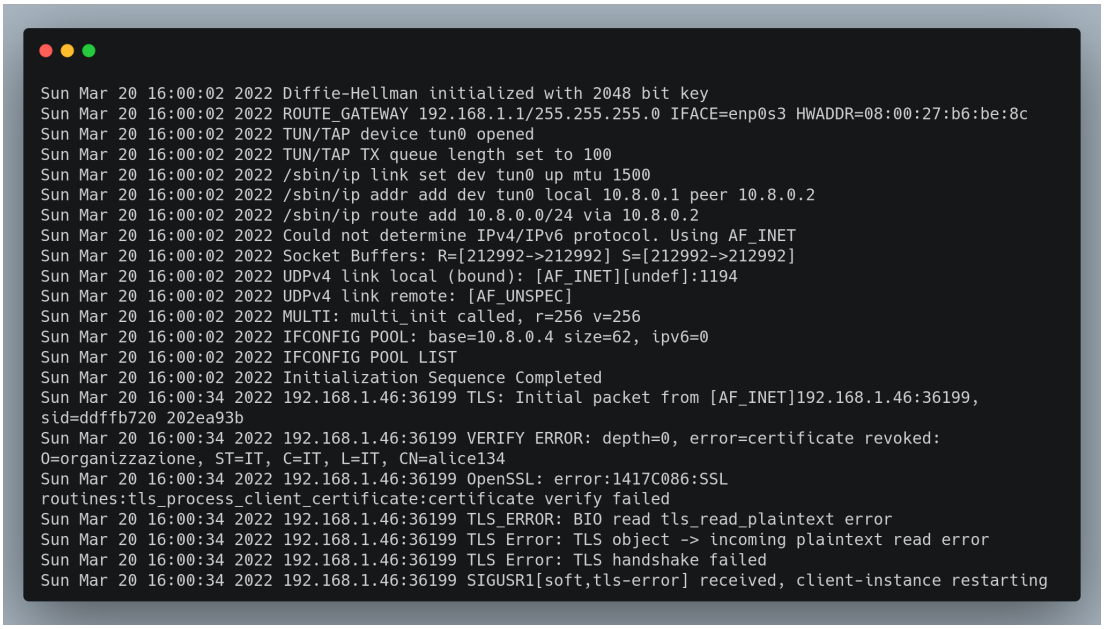
CRL - Certificate Revocation List

X.509 definisce anche gli standard da applicare per la gestione delle Certificate Revocation List (CRL), che rappresentano liste di certificati che non devono essere più considerati attendibili nonostante non sia ancora giunta la loro data di fine validità.
Il CRL viene mantenuto in memoria dal server, e contiene i numeri seriali dei certificati revocati. Ad ogni nuova connessione, il server controlla che il numero seriale del certificato presentato dal client non sia contenuto nella lista di quelli revocati prima di autorizzare la connessione.
Ogni volta che viene revocato un certificato attivo, viene generato un nuovo CRL, contente tutti i seriali dei certificati precedentemente invalidati più il nuovo seriale del certificato revocato.



```
Certificate Revocation List (CRL):
        Version 2 (0x1)
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: CN = authority, O = organizzazione, ST = IT, C = IT, L = IT
        Last Update: Mar 19 15:57:54 2022 GMT
        Next Update: Mar 30 15:57:54 2022 GMT
Revoked Certificates:
    Serial Number: 2CCB95F72B3356A1247BDC90A878C93F7934A729
        Revocation Date: Mar 20 15:57:54 2022 GMT
    Signature Algorithm: sha256WithRSAEncryption
         04:6f:22:ff:6e:eb:d8:f6:17:bb:77:7c:6e:e4:f2:4c:c4:b4:
         ...
         7e:f4:91:bd:95:6c:be:58:78:a4:4a:a0:3e:e3:9e:41:b0:f5:
         93:7e:25:40
```

Figura 4.7: Esempio di CRL contenente il seriale di un certificato revocato



```
Sun Mar 20 16:00:02 2022 Diffie-Hellman initialized with 2048 bit key
Sun Mar 20 16:00:02 2022 ROUTE_GATEWAY 192.168.1.1/255.255.255.0 IFACE=enp0s3 HWADDR=08:00:27:b6:be:8c
Sun Mar 20 16:00:02 2022 TUN/TAP device tun0 opened
Sun Mar 20 16:00:02 2022 TUN/TAP TX queue length set to 100
Sun Mar 20 16:00:02 2022 /sbin/ip link set dev tun0 up mtu 1500
Sun Mar 20 16:00:02 2022 /sbin/ip addr add dev tun0 local 10.8.0.1 peer 10.8.0.2
Sun Mar 20 16:00:02 2022 /sbin/ip route add 10.8.0.0/24 via 10.8.0.2
Sun Mar 20 16:00:02 2022 Could not determine IPv4/IPv6 protocol. Using AF_INET
Sun Mar 20 16:00:02 2022 Socket Buffers: R=[212992->212992] S=[212992->212992]
Sun Mar 20 16:00:02 2022 UDPv4 link local (bound): [AF_INET][undef]:1194
Sun Mar 20 16:00:02 2022 UDPv4 link remote: [AF_UNSPEC]
Sun Mar 20 16:00:02 2022 MULTI: multi_init called, r=256 v=256
Sun Mar 20 16:00:02 2022 IFCONFIG POOL: base=10.8.0.4 size=62, ipv6=0
Sun Mar 20 16:00:02 2022 IFCONFIG POOL LIST
Sun Mar 20 16:00:02 2022 Initialization Sequence Completed
Sun Mar 20 16:00:34 2022 192.168.1.46:36199 TLS: Initial packet from [AF_INET]192.168.1.46:36199,
sid=ddffb720 202ea93b
Sun Mar 20 16:00:34 2022 192.168.1.46:36199 VERIFY ERROR: depth=0, error=certificate revoked:
O=organizzazione, ST=IT, C=IT, L=IT, CN=alice134
Sun Mar 20 16:00:34 2022 192.168.1.46:36199 OpenSSL: error:1417C086:SSL
routines:tls_process_client_certificate:certificate verify failed
Sun Mar 20 16:00:34 2022 192.168.1.46:36199 TLS_ERROR: BIO read tls_read_plaintext error
Sun Mar 20 16:00:34 2022 192.168.1.46:36199 TLS Error: TLS object -> incoming plaintext read error
Sun Mar 20 16:00:34 2022 192.168.1.46:36199 TLS Error: TLS handshake failed
Sun Mar 20 16:00:34 2022 192.168.1.46:36199 SIGUSR1[soft,tls-error] received, client-instance restarting
```

Figura 4.8: (Lato server) Fallimento dell'handshake per certificato revocato

4.3 Interfaccia WEB

L'interfaccia web di FARAWAY costituisce il punto di contatto tra l'appliance e gli utenti che ne utilizzano i servizi. Nella sua realizzazione sono state compiute alcune scelte per definire degli aspetti tecnici, come il linguaggio di programmazione utilizzato, il framework di implementazione e le strategie di deployment dell'applicativo. Di seguito vengono approfondite le caratteristiche e le motivazioni che hanno guidato le scelte compiute nella realizzazione.

4.3.1 Perché Python?

Python è un linguaggio di programmazione e scripting di alto livello con efficienti strutture dati, che consente un approccio semplice alla programmazione orientata agli oggetti. E' caratterizzato da una sintassi lineare e da un meccanismo di typing dinamico che rendono il codice prodotto particolarmente leggibile.
La sua portabilità inoltre consente di eseguire le applicazioni realizzate su qualsiasi macchina che supporti l'interprete Python. L'applicativo di FARAWAY infatti non necessita di una specifica distribuzione Linux per essere eseguito.

Oltre a ciò, il linguaggio presenta un comodo meccanismo per la gestione delle librerie (o *moduli*), grazie al gestore di pacchetti *pip* che consente di installare e gestire le dipendenze necessarie con pochi comandi e in modo veloce.

Queste ragioni, unite alla vasta disponibilità di framework per ogni tipo di esigenza, hanno guidato verso Python la scelta del linguaggio di programmazione da utilizzare per lo sviluppo.

4.3.2 Framework di implementazione

Esistono numerosi framework per il linguaggio Python, che ne semplificano l'applicazione in settori specifici dello sviluppo come web, data science, intelligenza artificiale, grafica e altro.
Viste le esigenze di realizzazione, come framework di implementazione è stato scelto Python Django [2], un framework per lo sviluppo web di carattere open source che mira a velocizzare lo sviluppo e a curare aspetti chiave come la sicurezza e la scalabilità

Gli aspetti chiave che hanno guidato questa scelta sono:

- Portabilità, il codice prodotto con il framework può essere utilizzato su tutte le principali piattaforme.
- Open Source, la presenza di più di duemila contributori nella repository ufficiale di Django garantiscono un forte supporto da parte della community.
- Sicurezza, le funzionalità riguardanti la sicurezza sono implementate di default nel framework, come la protezione da SQL Injection e cross-site scripting.
- ORM (Object-relational mapper), Django offre un layer ORM pronto e funzionale, capace di interagire con i più comuni tipi di database in circolazione e mappare i dati in tipi nativi del linguaggio Python.

4.3.3 Deployment con Gunicorn e Nginx

Per quanto riguarda il deployment dell'applicazione, sono state analizzate varie alternative, e la migliore è risultata essere quella di utilizzare *Green Unicorn* [3] (o *Gunicorn*) come server web per interagire con l'applicativo Django e *Nginx* [4] come proxy per servire i file statici (immagini, css e html).

Gunicorn è un server web a basso utilizzo di risorse che supporta WSGI (Web Server Gateway Interface), un protocollo di trasmissione che regola le interazioni tra server e applicazioni web scritte in linguaggio Python. Nella struttura utilizzata per il deployment, Gunicorn si occupa di accettare le richieste e gestire la logica di dominio e le connessioni HTTP, ma non è in grado di servire i file statici necessari alla corretta visualizzazione delle pagine web.

A tal proposito è stato utilizzato Nginx, un altro server web che però è in grado di fare da proxy e servire i file statici senza coinvolgere l'applicativo Django. In questo modo, solo le richieste che lo richiedono vengono indirizzate all'applicazione Django, mentre i file statici vengono letti e serviti da una directory preconfigurata.

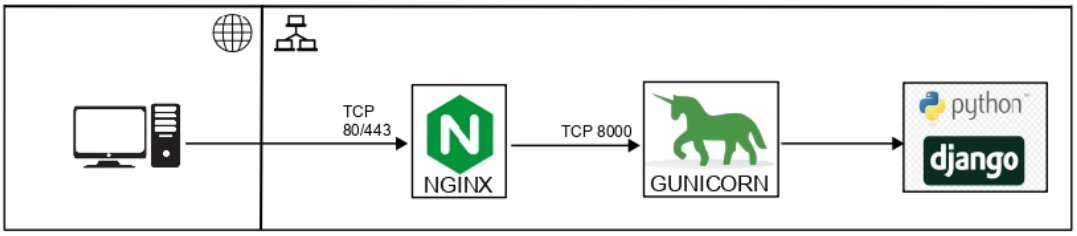


Figura 4.9: Schema di deployment

4.4 Accesso al PC tramite Desktop Remoto

FARAWAY è concepito per essere utilizzato in combinazione con Windows Remote Desktop per il controllo dei PC. RDP (Remote Desktop Protocol) è un protocollo di rete proprietario di casa Microsoft i cui client sono disponibili per la maggior parte dei sistemi operativi in circolazione. Consente una connessione remota diretta da un pc ad un altro mantenendo la privacy al livello di sessione.
Per realizzare gli obiettivi di controllo remoto spesso vengono utilizzate altre applicazioni commerciali che consentono la connessione tramite un id identificativo assegnato ad ogni pc, ma è necessario far notare che questa soluzione causa il passaggio di dati su server di terze parti e non riserva una vera e propria sessione all'utente (si tratta di una semplice condivisione dello schermo e dell'input).

Abbinando poi l'utilizzo del Desktop Remoto di Windows con la tecnologia VPN il sistema risulta ancora più sicuro dal punto di vista della privacy dei dati.

4.5 Wake-on-Lan e protocollo MQTT

Quasi sempre, nella gestione dello smart-working, viene meno la possibilità di controllare lo stato di accensione dei dispositivi, rendendo poco scorrevole il lavoro nel caso in cui il PC aziendale sia spento. Nella realizzazione di FARAWAY sono state pensate due soluzioni per risolvere questo problema. La più semplice, ma non sempre applicabile, consiste nell'utilizzo del protocollo Wake-on-LAN per l'accensione delle macchine, mentre la seconda coinvolge il protocollo MQTT per il controllo dei dispositivi IoT.

4.5.1 Wake-on-Lan

Il Wake-on-Lan (WOL) è uno standard ethernet che consente ad un computer che lo supporta di essere acceso attraverso l'invio di un messaggio in rete. Questa tecnologia fu introdotta nel 1997 da IBM e Intel.

Caratteristiche

Lo standard Wake-on-LAN agisce al livello di frame ethernet. Viene infatti utilizzato un particolare formato di frame chiamato "Magic Packet" contente l'indirizzo MAC del dispositivo da svegliare. I dispositivi che supportano il protocollo ascoltano passivamente i pacchetti in arrivo, e se viene ricevuto un magic packet, la scheda di rete della macchina invia un segnale alla scheda madre per avviare la procedura di accensione. Poiché l'ip del PC di destinazione non è conosciuto, la trasmissione del pacchetto magico avviene in broadcast. Viene quindi ricevuto da tutti gli host della rete e preso in considerazione solo dal proprietario dell'indirizzo MAC specificato.

Magic packet

Un magic packet è una frame ethernet particolare, formata da un payload di 6 bytes di 255 (FF FF FF FF FF FF in esadecimale) seguiti dall'indirizzo MAC a 48-bit del pc di destinazione ripetuto 16 volte, per un totale di 102 bytes.
Al livello di trasporto è di solito inviato come datagramma UDP, dato che un protocollo di trasporto orientato alla connessione come TCP non è particolarmente adatto per il compito.

Un magic packet ha le seguenti caratteristiche:

- Richiede il supporto hardware da parte del PC di destinazione
- Non fornisce una conferma di consegna del messaggio
- Richiede la conoscenza del MAC address del PC di destinazione
- Potrebbe non funzionare al di fuori della rete locale

Limitazioni

Per il funzionamento del Wake-on-LAN, è necessario che l'adattatore di rete del dispositivo non sia completamente spento e che mantenga il collegamento attivo con la rete. Questa caratteristica utilizza una minima parte di standby power e la velocità del link è di solito ridotta al minimo possibile per non sprecare energia. Non tutti i tipi di hardware però supportano la possibilità di mantenere questo stato: in alcuni casi

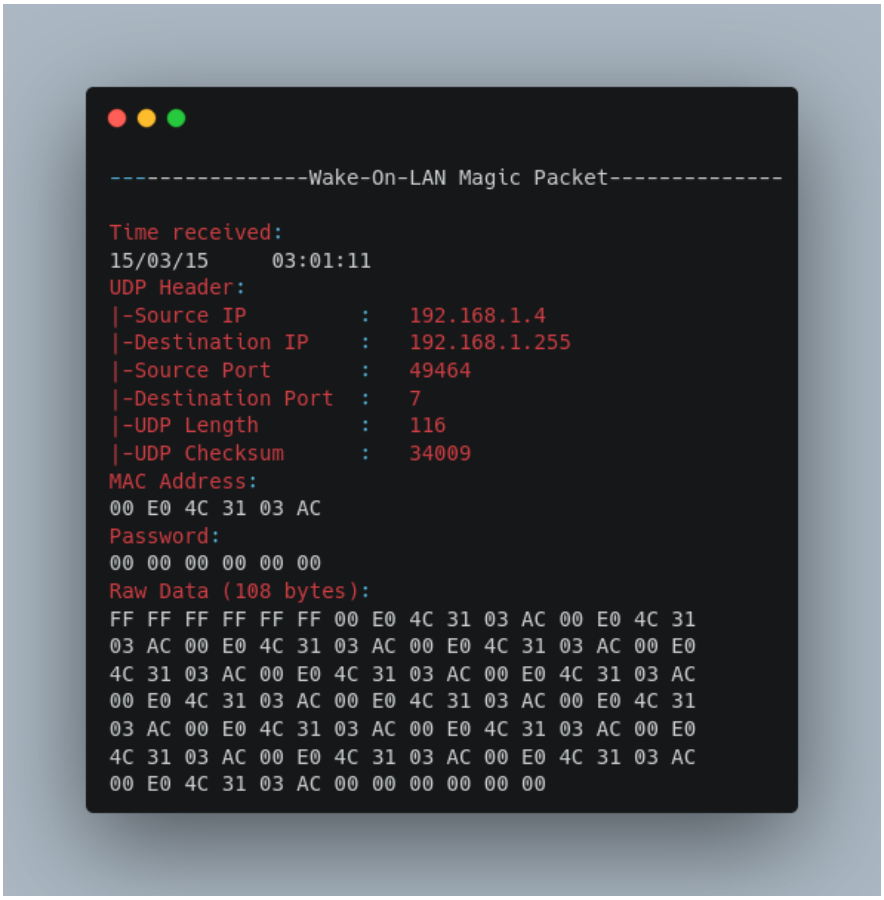


Figura 4.10: Struttura di un magic packet

infatti è stato riscontrato che è possibile svegliare i PC solo quando essi sono in stato di sospensione e non di spegnimento.

Per quanto riguarda la sicurezza, il fatto che i magic packet siano inviati al livello data-link (layer 2 del modello OSI), ne consentirebbe l'abuso da parte di qualsiasi host collegato alla LAN. Per fa fronte a ciò, alcuni NIC implementano una funzionalità chiamata "SecureOn", che permette di memorizzane all'interno dell'adattatore di rete una password esadecimale a 6 byte che dovrà poi essere inclusa nel magic packet quando inviato.

4.5.2 Protocollo MQTT

MQTT [12] (Message Queueing Telemetry Transport) è un protocollo di rete impiegato per il trasporto di messaggi prevalentemente tra dispositivi IoT. Solitamente viene utilizzato su reti TCP/IP, ma è adattabile a qualsiasi protocollo di rete che fornisca una connessione ordinata, lossless e bi-direzionale. Risulta particolarmente adatto per i casi in cui i dispositivi coinvolti abbiano risorse hardware e di rete limitate.

Broker, client e topic

Il protocollo definisce nel suo funzionamento due tipi di entità:

- *Message broker*, entità server alla quale vengono collegati tutti i client. Il broker riceve i messaggi inviati dai clienti e li instrada ai client interessati.
- *Client*, può essere considerato un client MQTT qualsiasi dispositivo che si connette ad un broker MQTT per scambiare messaggi, spaziando da un micro-controller ad un server con hardware completo.

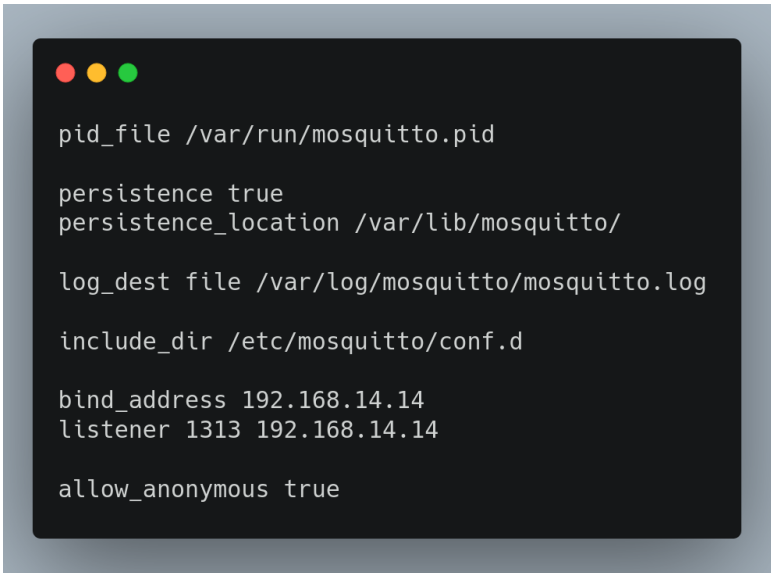
Lo scambio dei messaggi è regolato attraverso una gerarchia di topic e un meccanismo di publish-subscribe attuato su questi topic. Ogni client può decidere di sottoscrivere (subscribe) un topic e/o pubblicare (publish) messaggi su di esso.
Quando viene pubblicato un messaggio su un topic, questo viene inoltrato dal broker a tutti i client che sono sottoscritti al topic. Chi pubblica i messaggi non necessita di nessuna informazione a proposito degli eventuali ricevitori del messaggio, e viceversa i client che sottoscrivono un topic non hanno bisogno di informazioni riguardo i client che pubblicheranno messaggi.
Grazie a questa architettura, le applicazioni client e quelle server mantengono un bassissimo accoppiamento al livello software, infatti ogni client non conosce nessuna informazione al di fuori dell'indirizzo del suo broker.

Le dimensioni di un messaggio di controllo su protocollo MQTT possono variare da 2 bytes fino a 256 megabytes se necessario. Il protocollo definisce quattordici tipi diversi di messaggi che possono essere usati per connettere dispositivi, inviare dati, assicurarne la ricezione e controllare la connessione tra client e server.

Per quanto riguarda la sicurezza, il protocollo in sé prevede l'invio delle credenziali di autenticazione in chiaro. Questo problema può essere tuttavia risolto rendendo sicura la comunicazione tramite TLS.

Mosquitto MQTT broker

Il broker utilizzato nell'implementazione di FARAWAY è Eclipse Mosquitto [6], un broker MQTT open source configurabile su vari tipi di dispositivo.
La configurazione del broker viene gestita dall'applicativo attraverso la modifica del file *m*osquitto.conf, presente nella directory /etc/mosquitto. I parametri essenziali da specificare per il funzionamento (oltre a quelli impostati di default) sono l'indirizzo ip e la porta su cui il servizio deve rimanere in ascolto per eventuali messaggi.



```
pid_file /var/run/mosquitto.pid

persistence true
persistence_location /var/lib/mosquitto/

log_dest file /var/log/mosquitto/mosquitto.log

include_dir /etc/mosquitto/conf.d

bind_address 192.168.14.14
listener 1313 192.168.14.14

allow_anonymous true
```

Figura 4.11: File di configurazione del broker generato dall'applicativo

4.5.3 Attuatori remoti

Gli attuatori remoti (smart plug) utilizzati nel progetto sono dispositivi prodotti dall'azienda Shelly, modello Plug-s [1].
Questi dispositivi sono pre-configurati come access-point e al primo collegamento alla rete elettrica mettono a disposizione un'interfaccia web per stabilire il loro comportamento. Per integrare gli attuatori remoti all'interno di FARAWAY, è necessario accedere all'interfaccia ed attivare la configurazione MQTT manuale, altrimenti la presa smart risulterà utilizzabile solamente attraverso applicazione proprietaria.
Una volta sbloccata la configurazione, è necessario togliere l'apparecchio dalla modalità server e attivarlo in modalità client, specificando l'SSID e la password della wi-fi a cui dovranno connettersi al nuovo avvio. L'ultimo parametro da impostare è l'indirizzo ip del broker con cui dovranno comunicare, nel nostro caso lo stesso dell'appliance che gestisce anche gli altri servizi.

Una volta effettuata questa breve configurazione del dispositivo, è possibile registrarlo dall'interfaccia di FARAWAY specificandone l'indirizzo IP e il nome identificativo (procurato dall'interfaccia di configurazione del dispositivo stesso).
Il nome identificativo del dispositivo serve a costruire la stringa che costituisce il nome del topic su cui l'applicativo di FARAWAY pubblicherà i messaggi di accensione e spegnimento.
Ad esempio, nel caso specifico del modello di attuatore remoto utilizzato, le prese sono progettate per ascoltare sul topic:

shellies/shellyplug-s-<nome_dispositivo>/relay/0/command

ed attivarsi al rilevamento dei messaggi "on" e "off".

```python
def send_on_message(device_name):

    config = MqttConfig.objects.get(name="config")
    client = Client()
    client.connect(host=config.listener_ip, port=config.server_port, keepalive=60)
    client.publish(topic=f"shellies/shellyplug-s-{device_name}/relay/0/command", payload="on")
    client.disconnect()
```

Figura 4.12: Codice Python che invia il messaggio di accensione all'attuatore

Al termine della procedura di registrazione, è possibile associare il dispositivo registrato ad un PC sprovvisto di attuatore remoto in pochi click. Una volta associato il dispositivo, è possibile aprire la pagina di dettaglio del PC per accenderlo sia tramite Wake-on-LAN che tramite attuatore remoto.

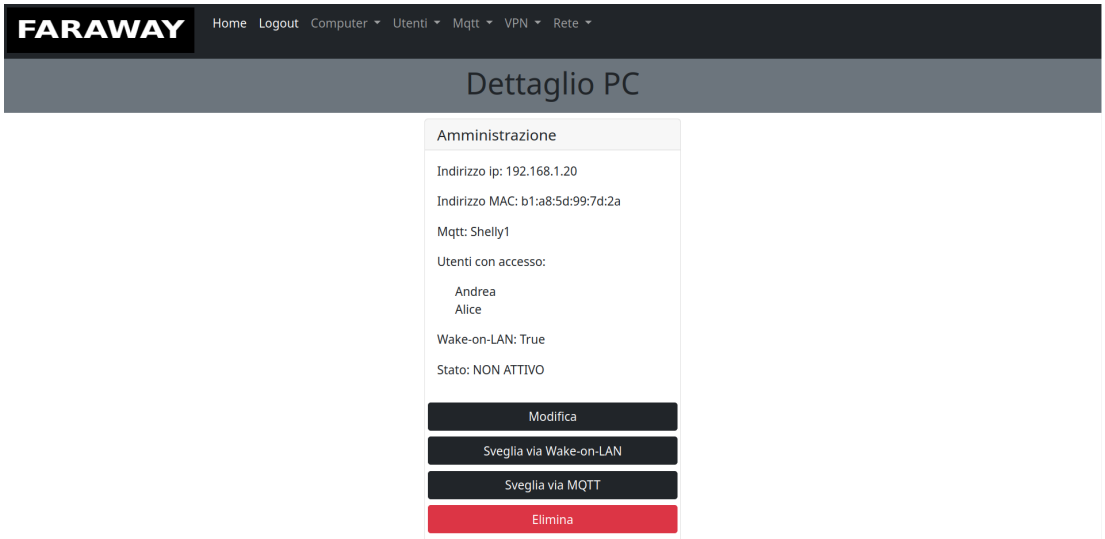


Figura 4.13: Schermata di dettaglio del PC con dispositivo MQTT associato

5. Problematiche affrontate e sviluppi futuri

In questo capitolo verranno descritte alcune delle problematiche rilevate o affrontate durante la realizzazione del progetto, le quali hanno coinvolto principalmente la gestione dei privilegi e quella degli spazi di indirizzi ip da utilizzare per i servizi di rete.

5.1 Permessi

Una delle principali problematiche affrontate durante lo sviluppo riguarda la gestione dei permessi con cui vengono eseguite le operazioni all'interno del sistema. Per lo svolgimento dei test è stato utilizzato l'utente root, ma in ambito di produzione questa scelta rappresenterebbe un grave rischio al livello di sicurezza. In caso di vulnerabilità dell'applicativo infatti, sarebbe così possibile ottenere i privilegi di amministratore sulla macchina.
L'applicazione però necessita comunque di alcuni privilegi per eseguire operazioni come la modifica dei file di configurazione, l'avvio dei servizi o la gestione delle chiavi VPN.

La problematica è stata risolta eseguendo l'applicativo tramite un utente ordinario (non amministratore), nel caso specifico "toor". A questo utente viene concesso, attraverso la modifica del file *sudoers*, di eseguire i principali comandi utilizzando il programma sudo senza che il sistema richieda di inserire una password.
Per quanto riguarda l'accesso ai file, sono state utilizzate le ACL (Access Control List) POSIX [13] per garantire all'utente specifico i privilegi necessari in lettura e scrittura. Grazie alle soluzioni applicate è stato quindi ristretto l'insieme delle operazioni effettuabili da un ipotetico attaccante sfruttando una vulnerabilità nell'applicativo.

5.2 Conflitti di IP

Una ulteriore problematica che è stata rilevata riguarda gli indirizzi IP, ed in particolare i conflitti che si possono generare tra gli indirizzi della LAN aziendale, quelli della VPN e quelli della rete domestica dell'utente.

La prima tipologia di conflitti che può emergere è quella tra gli indirizzi ip utilizzati dal servizio VPN e quelli utilizzati dalla LAN dell'azienda. Nel caso in cui il server VPN e la rete aziendale utilizzino lo stesso spazio di indirizzi IP per i propri host, questo genererebbe una situazione di conflitto causando il malfunzionamento del sistema. Allo stato attuale del progetto, la problematica non è stata ancora affrontata, ma è facilmente risolvibile rendendo lo spazio di indirizzi ip utilizzati dal servizio VPN confi-

gurabile dinamicamente dall'amministratore, in modo da poter scegliere indirizzi diversi da quelli della LAN.

Una seconda tipologia di conflitti è quella che può verificarsi quando la rete utilizzata dal client VPN per connettersi da remoto è la stessa utilizzata dalla LAN aziendale. Questa tipologia di conflitto è mitigata dal fatto che non vengono passate al client rotte verso un'intera classe ma solo verso indirizzi IP specifici, che rappresentano i dispositivi a cui l'utente ha effettivamente bisogno di accedere.
Questa problematica può comunque essere risolta con due modalità. Una consiste nel non inviare le rotte che vanno in conflitto al client ma far raggiungere il servizio tramite l'apertura di specifiche porte sull'interfaccia WAN del servizio VPN (con operazioni cosiddette di port-forwarding). Oppure decidendo di inviare al client rotte virtuali (ovvero verso ip appartenenti a una classe di rete virtuale non in conflitto con quella locale) che verranno tradotte dal server VPN in modo da reindirizzarle verso la classe di rete corretta.
Anche in questo caso le due soluzioni non vengono attualmente gestite dall'interfaccia ma possono essere implementate con estrema facilità (o in alternativa attivate manualmente all'occorrenza).

Bibliografia

[1] *Shelly Plug S*. URL: https://www.shellyitalia.com/shelly-plug-s/.

[2] *Django Web Framework*. URL: https://www.djangoproject.com/.

[3] *Green Unicorn Web Server*. URL: https://gunicorn.org/.

[4] *Nginx Web Server*. URL: https://www.nginx.com/.

[5] *OpenVPN*. URL: https://openvpn.net/.

[6] *Eclipse Mosquitto*. URL: https://mosquitto.org/.

[7] *Linux Iptables*. URL: https://wiki.ubuntu-it.org/Sicurezza/Iptables.

[8] *Linux Netplan*. URL: https://netplan.io/.

[9] *OpenSSL*. URL: https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html.

[10] *TLS RFC*. URL: https://datatracker.ietf.org/doc/html/rfc5246.

[11] *X.509 RFC*. URL: https://datatracker.ietf.org/doc/html/rfc5280.

[12] *MQTT Standard*. URL: https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html.

[13] *ACL POSIX*. URL: https://help.ubuntu.com/community/FilePermissionsACLs.

[14] Christopher Negus. *"The Linux Bible"*. 2020.

[15] Trent R. Hein Ben Whaley Dan Mackin Evi Nemeth Garth Snyder. *"Unix and Linux system administration handbook"*. 2020.

Ringraziamenti

Giunto alla fine, guardando gli anni trascorsi durante il mio percorso universitario, mi rendo conto di come questi siano stati fondamentali nella formazione della mia persona, non solo dal punto di vista didattico ma anche umano. Non sono di certo mancate occasioni di confronto e accrescimento del mio bagaglio culturale, sia con i compagni che con i docenti.

E' doveroso ringraziare tutti i miei colleghi dell'azienda Mercurio, tra cui il correlatore della mia tesi Samuele Cacchiarelli, che mi hanno assistito nel percorso e mi hanno concesso infinite opportunità di crescita dal punto di vista tecnico.

Ringrazio tutti i docenti e il personale di UNICAM in generale per aver reso l'esperienza universitaria quello che è stato: un'esperienza di arricchimento culturale supportata dalle giuste figure professionali e dalle migliori infrastrutture.

Ringrazio tutti i miei compagni di corso, in particolare Graziano, Matteo e Flavio, insieme ai quali ho acquisito nuove conoscenze nella realizzazione dei progetti universitari.

Un ulteriore ringraziamento va a mio fratello e ai miei amici più cari, oltre che a Camilla e la sua famiglia, per essermi sempre stati vicini nel momento del bisogno.

Grazie infinite a tutti voi.