Università degli studi di Napoli Parthenope
Laboratorio di Reti di Calcolatori
Anno 2022/2023
Progetto 2

Green Pass

Studenti	Matricola
Fabrizio Lanzuise	0124002449
Davide Perrotta	0124002389
Riccardo Maione	0124002391

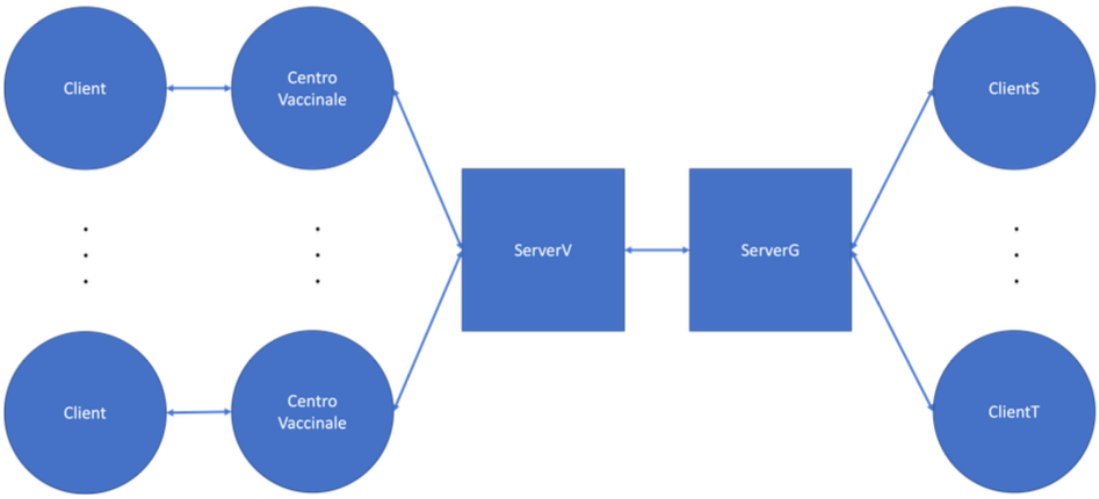
Professore Alessio Ferone

Descrizione

Lo scopo del progetto è quello di realizzare un servizio di gestione dei green pass. Il progetto ha le seguenti specifiche: Un utente, una volta effettuata la vaccinazione, tramite un client si collega ad un centro vaccinale e comunica il codice della propria tessera sanitaria. Il centro vaccinale comunica al ServerV il codice ricevuto dal client ed il periodo di validit`a del green pass. Un ClientS, per verificare se un green pass `e valido, invia il codice di una tessera sanitaria al ServerG il quale richiede al ServerV il controllo della validit`a. Un ClientT, inoltre, pu`o invalidare o ripristinare la validit`a di un green pass comunicando al ServerG il contagio o la guarigione di una persona attraverso il codice della tessera sanitaria.

Descrizione e schemi dell'architettura

<



Schema del protocollo applicazione

Il ClientA si connette al CentroVaccinale e gli invia la tessera sanitaria per il calcolo della scadenza del Green Pass. Successivamente quando il CentroVaccinale riceve la tessera sanitaria allora calcola la scadenza del Green Pass e lo inoltra al ServerV.
Il ServerV successivamente memorizza i Green Pass. Il ClientS e il ClientT si connettono al serverG e inviano la tessera sanitaria. Il ClientS richiede il green Pass associato a quella tessera mentre il ClientT annulla o valida il green pass. Di conseguenza il ServerG riceve le richieste da ClientS e ClientT e le invia al serverV, quest'ultimo riceve la richiesta e risponde. Dopo che il ServerG riceve le risposte dal ServerV le invia al Client che le ha richieste.

Logica del pacchetto applicazione

- Stringa di 20 caratteri: la tessera sanitaria. (TS)
- Campo scadenza: Green Pass. (Scad)
- option: indica il tipo di servizio richiesto a ServerV:
    - `Case 1: Registrazione su file del nuovo/aggiornato greenPass`
    - `Case 2: Risposta al client se il green pass è valido o meno`
    - `Case 3: Invalidamento greenPass da contagio`
    - `Case 4: Ripristino greenPass da guarigione`

Quando *ServerV* riceve una richiesta di registrare o aggiornare il green pass si sposta con la funzione fseek all'inizio del file e inizia a cercare il Green Pass richiesto. Se lo trova memorizza/ aggiorna il green pass. Se non ha trovato il green pass da aggiornare nel file green pass lo crea e lo mette in fondo al file.

```
case 1:    // CentroVaccinale operazione-> registrazione sul file del nuovo/aggiornato green pass

    fseek(file,0,SEEK_SET);
    printf("CASO 1: \n");
    int gp_trovato = 0;

    while(fread(&temp_gp,sizeof(struct green_pass),1,file) == 1){

        if(strcmp(temp_gp.TS, in_gp.TS) == 0){

            fseek(file,-(sizeof(struct green_pass)),SEEK_CUR);
            gp_trovato=1;
            sem_wait(sem_phore); /* DECREMENTA IL VALORE DI access DA 1 A 0 (IN QUESTO MODO AUMENTA LA PRIORITA DEL THREAD) */

            gp_trovato=1;
            in_gp.validita = 1;
            fwrite(&in_gp,sizeof(struct green_pass),1,file);
            printf(" P2 Ts %s Scad %.24s  validita %d \r\n",in_gp.TS,ctime(&in_gp.Scad),in_gp.validita);
            sem_post(sem_phore); /* AUMENTA IL VALORE DI access DA 0 A 1 (IN QUESTO MODO DIMINUSCE LA PRIORITA DEL THREAD) */
            break;
        }
    }


    if(gp_trovato==0){
        fseek(file,0,SEEK_END);
        sem_wait(sem_phore); // DECREMENTA IL VALORE DI access DA 1 A 0 (IN QUESTO MODO AUMENTA LA PRIORITA DEL THREAD)
        in_gp.validita = 1;
        fwrite(&in_gp,sizeof(struct green_pass),1,file);
        printf(" P1 Ts %s Scad %.24s validita %d\r\n",in_gp.TS,ctime(&in_gp.Scad),in_gp.validita);
        sem_post(sem_phore);
    }

break;
```

Il ServerV controlla la validità del green pass tramite il codice della tessera sanitaria inviata dal serverG. Quindi per svolgere questa operazione legge il file e con la strcmp va trovare il green pass. Se lo trova riporta tutte le informazioni e la validità. Se non trova il green pass ritorna solo il numero della tessera sanitaria.

```
case 2:    // ClientS output-> risposta al client se il green pass è valido o meno
    printf("CASO 2: ");
    fseek(file,0,SEEK_SET);
    gp_trovato = 0;

    while(fread(&temp_gp,sizeof(struct green_pass),1,file) == 1){

        if(strcmp(temp_gp.TS, in_gp.TS) == 0){

            sem_wait(sem_phore); /* DECREMENTA IL VALORE DI access DA 1 A 0 (IN QUESTO MODO AUMENTA LA PRIORITA DEL THREAD) */
            FullWrite(conn_fd,&temp_gp,sizeof(temp_gp));
            printf(" Ts %s Scad %.24s validita %d\r\n",temp_gp.TS,ctime(&temp_gp.Scad),temp_gp.validita);
            sem_post(sem_phore); /* AUMENTA IL VALORE DI access DA 0 A 1 (IN QUESTO MODO DIMINUSCE LA PRIORITA DEL THREAD) */
            break;
        }
    }

    if(gp_trovato==0){
        sem_wait(sem_phore); /* DECREMENTA IL VALORE DI access DA 1 A 0 (IN QUESTO MODO AUMENTA LA PRIORITA DEL THREAD) */
        in_gp.validita=0;
        FullWrite(conn_fd,&in_gp,sizeof(in_gp));
        printf("Ts %s ",in_gp.TS);
        sem_post(sem_phore);
    }


break;
```

Se un utente è contagiato, il Green Pass non deve essere valido. Quindi con fseek si pone all'inizio del file e lo va a leggere con la fread. Se trova il green pass allora imposta la validità a 0, se non trova la TS restituisce : " TS non presente"

```
case 3:    // ClientT operazione-> Invalidamento greenPass da contagio
    printf("CASO 3: \n");
    fseek(file,0,SEEK_SET);
    gp_trovato=0;

    while(fread(&temp_gp,sizeof(struct green_pass),1,file) == 1){

        if(strcmp(temp_gp.TS, in_gp.TS) == 0){

            fseek(file,-(sizeof(struct green_pass)),SEEK_CUR);
            gp_trovato=1;
            sem_wait(sem_phore); /* DECREMENTA IL VALORE DI access DA 1 A 0 (IN QUESTO MODO AUMENTA LA PRIORITA DEL THREAD) */
            in_gp.validita = 0;
            fwrite(&in_gp,sizeof(struct green_pass),1,file);
            printf("Ts %s validita %d INVALIDATO \r\n",in_gp.TS,in_gp.validita);
            sem_post(sem_phore); /* AUMENTA IL VALORE DI access DA 0 A 1 (IN QUESTO MODO DIMINUSCE LA PRIORITA DEL THREAD) */

            break;
        }
    }

    if(gp_trovato==0){
        printf("TS NON PRESENTE (invalidamento)");
    }

break;
```

Se un utente è guarito, il green Pass deve essere ripristinato. Con fseek si pone a inizio file e comincia la lettura, se trova la TS allora ripristina la validità a 6 mesi. Altrimenti restituisce: "TS NON PRESENTE (guarigione)"

```
case 4:    // ClientT operazione-> Ripristino greenPass da guarigione
    printf("CASO 4: \n");
    fseek(file,0,SEEK_SET);
    gp_trovato=0;

    while(fread(&temp_gp,sizeof(struct green_pass),1,file) == 1){

        if(strcmp(temp_gp.TS, in_gp.TS) == 0){

            fseek(file,-(sizeof(struct green_pass)),SEEK_CUR);
            gp_trovato=1;
            sem_wait(sem_phore); /* DECREMENTA IL VALORE DI access DA 1 A 0 (IN QUESTO MODO AUMENTA LA PRIORITA DEL THREAD) */
            in_gp.validita = 1;
            in_gp.Scad = time(NULL) + SCADENZASEIMESI;
            fwrite(&in_gp,sizeof(struct green_pass),1,file);
            printf("Ts %s validita %d GUARITO\r\n",in_gp.TS,in_gp.validita);
            sem_post(sem_phore); /* AUMENTA IL VALORE DI access DA 0 A 1 (IN QUESTO MODO DIMINUSCE LA PRIORITA DEL THREAD) */

            break;
        }
    }
    if(gp_trovato==0){
        printf("TS NON PRESENTE (guarigione)");
    }

break;
```

Manuale

Istruzioni per la compilazione

```
gcc -pthread -o serverV serverV.c
gcc -o serverG serverG.c
gcc -o clientA clientA.c
gcc -o clientS clientS.c
gcc -o clientT clientT.c
gcc -o cv centrovaccinale.c
gcc -o p prova.c
```

Istruzioni per la compilazione

```
./serverV
./serverG 127.0.0.1
./cv 127.0.0.1
./clientA 127.0.0.1 <TS>
./clientS 127.0.0.1 <TS>
./clientT 127.0.0.1 <TS> <V/I>
```