ELHACHIMI Aniss
SABIRI Abderrahman

TP : Systèmes de télécommunications sans-fils et applications

Carte sans-contact MIFARE Classic

Encadré par : Vincent THIVENT

Sommaire

1- Développement de l'interface graphique

1.1- Fonctionnalités de l'interface graphique

Dans ce TP on va développer l'interface graphique pour la gestion de la carte sans-contact MIFARE Classic. L'interface graphique doit pouvoir

•Enroller une carte MIFARE Classic d'origine ;

• Formater une carte MIFARE Classic pour qu'elle reprenne son état d'origine ;

• Incrémenter et décrémenter des unités dans la carte (avec la gestion d'un backup) ;

• Écrire et lire l'identité de la personne dans la carte (nom et prénom) ; Une information visuelle et/ou sonore du coupleur doit être faite à chaque transaction.

1.2- Architecture de la carte

1.2.1- Identification

Afin que les cartes soient lues par toutes les applications. Nous définirons une architecture commune à chaque carte. Pour l'identification, la carte aura cette configuration : 2 blocks de données + 1 block contenant le nom de l'application « Identité ».

Sector	Block	Data
2	11	KeyA+AccessBit+KeyB
	10	Nom
	9	Prenom
	8	« Indentite »

L'authentification se fera avec les clés suivantes :

- Key A : A0 A1 A2 A3 A4 A5
- Key B : B0 B1 B2 B3 B4 B5

1.2.2- Compteur

Pour le compteur, la carte aura cette configuration : 2 blocks de compteur+ 1 block contenant le nom de l'application « Porte-monnaie».

Sector	Block	Data
3	15	KeyA+AccessBit+KeyB
	14	Compteur
	13	Backup Compteur
	12	« Porte Monnaie »

L'authentification se fera avec les clés suivantes :

- Key A : C0 C1 C2 C3 C4 C5
- Key B : D0 D1 D2 D3 D4 D5

1.3- Fonctionnalités techniques et logiciel

Bouton connecte : permet de se connecter à la carte.

```
void Fenetre::on_Connect_clicked()
{
    int16_t status = 0;
    uint8_t atq[2];
    uint8_t sak[1];
    uint8_t uid[12];
    uint16_t uid_len = 12;
    uint8_t data[16];


    MonLecteur.Type = ReaderCDC;
    MonLecteur.device = 0;
    status = OpenCOM1(&MonLecteur);
    qDebug() << "OpenCOM1" << status;


    status = Mf_Classic_LoadKey(&MonLecteur, Auth_KeyA, key1, 0);
    qDebug() << "Mf_Classic_LoadKey KEY 1" << status;
    status = Mf_Classic_LoadKey(&MonLecteur, Auth_KeyB, key2, 0);
    qDebug() << "Mf_Classic_LoadKey KEY 2" << status;
    status = Mf_Classic_LoadKey(&MonLecteur, Auth_KeyA, key3, 0);
    qDebug() << "Mf_Classic_LoadKey KEY 3" << status;
    status = Mf_Classic_LoadKey(&MonLecteur, Auth_KeyB, key4, 0);
    qDebug() << "Mf_Classic_LoadKey KEY 4" << status;

    char version[30];
    uint8_t serial[4];
    char stackReader[20];
    status = Version(&MonLecteur, version, serial, stackReader);
    qDebug() << "Version" << status;
    ui->Affichage->setText(version);
    ui->Affichage->update();

    // RF field ON
    RF_Power_Control(&MonLecteur, TRUE, 0);

    status = ISO14443_3_A_PollCard(&MonLecteur, atq, sak, uid, &uid_len);
    qDebug() << "ISO14443_3_A_PollCard" << status;

}
```

Bouton Read : permet de lire le contenu du block 9 et 10 du secteur 2 et d'afficher son contenu.

```
void Fenetre::on_saisie_clicked(){


    int16_t status = 0;
    uint8_t atq[2];
    uint8_t sak[1];
    uint8_t uid[12];
    uint16_t uid_len = 12;
    uint8_t dataName[16];
    uint8_t dataNickname[16];

    MonLecteur.Type = ReaderCDC;
    MonLecteur.device = 0;

    status = Mf_Classic_Read_Block(&MonLecteur, TRUE, ((2*4)+1), dataNickname, Auth_KeyA, 2);
    qDebug() << "Mf_Classic_Read_Sector_Nickname" << status;

    status = Mf_Classic_Read_Block(&MonLecteur, TRUE, ((2*4)+2), dataName, Auth_KeyA, 2);
    qDebug() << "Mf_Classic_Read_Sector_Name" << status;

    ui->fenetre_saisi->setText((char*)dataName);
    ui->fenetre_saisi->update();

    ui->affiche_prenom->setText((char*)dataNickname);
    ui->affiche_prenom->update();

}
```

Bouton Write : permet d'écrire le contenu des champs de texte fenetre_write et fenetre_prenom Dans les blocs 9 et 10.

```
void Fenetre::on_Write_clicked(){


    int16_t status = 0;
    uint8_t atq[2];
    uint8_t sak[1];
    uint8_t uid[12];
    uint16_t uid_len = 12;
    uint8_t dataNom[16];
    uint8_t dataPrenom[16];


    QString Text = ui->fenetre_write->toPlainText();
    QString Text1 = ui->fenetre_prenom->toPlainText();
    strcpy((char*)dataNom, (char*)Text.toLatin1().data());
    strcpy((char*)dataPrenom, (char*)Text1.toLatin1().data());

    status = Mf_Classic_Write_Block(&MonLecteur, TRUE, ((2*4)+1), dataPrenom, Auth_KeyB, 2);
    qDebug() << "Mf_Classic_Write_Sector_Prenom" << status;

    status = Mf_Classic_Write_Block(&MonLecteur, TRUE, ((2*4)+2), dataNom, Auth_KeyB, 2);
    qDebug() << "Mf_Classic_Write_Sector_Nom" << status;
    qDebug() << "Nom : " << Text;
    qDebug() << "Prénom : " << Text1;
```

Bouton Incrémenter : permet d'incrémenter la valeur du bloc 13 du secteur 3 et de mettre le résultat dans le bloc 14 du même secteur.

```
void Fenetre::on_button_in_clicked(){

    uint32_t value =0;


    status = Mf_Classic_Increment_Value(&MonLecteur, TRUE, ((3*4)+2), 1, ((3*4)+1), Auth_KeyB,3);
    status = Mf_Classic_Restore_Value(&MonLecteur, TRUE, ((3*4)+1), ((3*4)+2), Auth_KeyB,3);

    status = Mf_Classic_Read_Value(&MonLecteur, TRUE, ((3*4)+2), &value, Auth_KeyA, 3);
    qDebug() << "Mf_Classic_Read_Value_14" << status;

    status = Mf_Classic_Read_Value(&MonLecteur, TRUE, ((3*4)+1), &value, Auth_KeyA, 3);
    qDebug() << "Mf_Classic_Read_Value_13" << status;
```

Bouton Décrémenter : permet de décrémenter la valeur du bloc 13 du secteur 3 et de mettre le résultat dans le bloc 14 du même secteur.

```
void Fenetre::on_button_de_clicked(){
    int16_t status = 0;
    uint32_t valueCard = 0;
    uint32_t value = (uint32_t) ui->/*l-ValeurIncrement*/ ->text().toInt();

    status = Mf_Classic_Read_Value(&MonLecteur, TRUE, ((3*4)+2), &valueCard, Auth_KeyA, 3);

    if(valueCard == 0 || valueCard-value > valueCard)
    {
        // On ne peut pas décrémenter plus (unsigned --> on évite d'atteindre 2^32-1)
        // TO DO Status bar
        ui->statusBar->showMessage("Il faut plus de crédit pour pouvoir décrémenter");
    }
    else
    {

        status = Mf_Classic_Decrement_Value(&MonLecteur, TRUE, ((3*4)+2), value, ((3*4)+1), Auth_KeyA, 3);

        if(status != 0)
            qDebug() << GetErrorMessage(status);

        status = Mf_Classic_Restore_Value(&MonLecteur, TRUE, ((3*4)+1), ((3*4)+2), Auth_KeyA, 3);

        if(status != 0)
            qDebug() << GetErrorMessage(status);


        status = Mf_Classic_Read_Value(&MonLecteur, TRUE, ((3*4)+2), &valueCard, Auth_KeyA, 3);


        //ui->l_Credit->setText(QString("Crédit : %1").arg(valueCard));
```

Bouton Quitter : permet de quitter la fenêtre.

```
void Fenetre::on_Quitter_clicked()
{
    int16_t status = 0;
    RF_Power_Control(&MonLecteur, FALSE, 0);
    status = LEDBuzzer(&MonLecteur, LED_OFF);
    status = CloseCOM1(&MonLecteur);
    qApp->quit();
```

Enfin, Nous obtenons la forme de l'interface graphique qui permet de lire, écrire, incrémenter et décrémenter des valeurs de bloc de la carte sans contact de type NXP Mifare Classic.

Ce TP nous a donné une idée globale sur le fonctionnement des systèmes detélécommunications sans-fils.