Projet ISE - BadUSB

Partie 2 - Préparation du raspberry pico 2

Introduction

Ce projet a pour objectif d'explorer les concepts derrière les attaques par "*rubber ducky*" (aussi connues sur le nom d'attaques *BadUSB*) et en apprenant à utiliser un Raspberry Pico 2 pour concevoir une clé USB malveillante. Les *rubber duckies* sont des dispositifs qui émulent un clavier USB pour injecter des commandes rapidement sur une machine cible. Leur popularité provient de leur simplicité et de leur efficacité pour réaliser des attaques telles que l'exfiltration de données, l'installation de logiciels malveillants, ou la création de portes dérobées.

> ❗ Ce projet a été rédigé à des fins pédagogiques. Il n'a pas pour vocation à être utilisé "sur le terrain". Pour rappel l'article L.323-1 du code pénal prévoit que "*Le fait d'accéder ou de se maintenir, frauduleusement, dans tout ou partie d'un système de traitement automatisé de données est puni de trois ans d'emprisonnement et de 100 000 € d'amende.*"

Le projet est divisé en 3 parties. Dans cette seconde partie, vous allez prendre en main le raspberry pico 2, vous familiariser avec ses capacités, et le transformer en *BadUSB*.

Présentation du Raspberry Pico 2

Le Raspberry Pico 2 est un microcontrôleur **programmable** qui ne contient par défaut aucun programme actif. Pour pouvoir lui faire exécuter des tâches, il faut d'abord y transférer ("flasher") un firmware compatible.

Le **flashage** consiste à copier un fichier spécifique (UF2) sur le Pico en le branchant en mode spécial appelé `BOOTSEL` (en appuyant sur le seul bouton de la carte pendant que vous le branchez). Ce fichier contient un système d'exploitation minimal qui définit le comportement de la carte.

Par exemple, CircuitPython permet ensuite au Pico d'exécuter des scripts Python directement. Par exemple, une fois flashé avec CircuitPython, le Pico fonctionne en exécutant automatiquement un script nommé `code.py`, situé à la racine de son stockage interne.

Vous remarquerez aussi les nombreuses tiges métalliques sur les côtés. Ces *pins*, appelées **GPIO** (General Purpose Input/Output), permettent d'ajouter des modules ou de configurer le comportement de la carte. Par exemple, on s'en servira plus tard pour dire à la carte "si la PIN 1 (GP0) est branchée à la PIN 3 (GND), on est en mode configuration".

Réinitialisation

Si vous rencontrez un problème **logiciel** suite à une mauvaise manipulation, vous pouvez toujours réinitialiser la carte, pour la remettre à sa configuration initiale. Vous perdrez les données stockées dessus, mais ça permet de repartir sur des bases saines. Pour cela, suivez les instructions suivantes :

1. Téléchargez le firmware flash_nuke.uf2. Il s'agit d'un firmware conçu pour effacer la configuration actuelle.
2. Maintenez le bouton `BOOTSEL` de la carte et branchez là. Elle devrait être reconnue comme un périphérique nommé "RP2350". Vous pouvez alors lâcher le bouton.
3. Copier le fichier `flash_nuke.uf2` sur le périphérique et patientez quelques secondes.

La carte est réinitialisée, c'est aussi simple que ça !

1 Préparation du Raspberry Pico 2 avec CircuitPython

Précédemment, on a mentionné CircuitPython. Ce firmware est très utile pour paramétrer facilement la carte. On va donc flasher le Pico avec ce firmware, ce qui facilitera grandement la suite du TP.

▶ **Question 1.** Configurez votre Pico :

- Téléchargez CircuitPython depuis : https://circuitpython.org/board/raspberry_pi_pico2/.
- Branchez votre Pico en mode `BOOTSEL` et copiez le fichier UF2 téléchargé.

Comment savez-vous que votre Pico est bien flashé ?

▶ **Question 2.** Ouvrez le fichier `code.py` situé à la racine du périphérique. Rédigez un script de quelques lignes faisant clignoter la LED intégrée chaque seconde.
Voici le début du fichier :

```
import board
import digitalio
import time

# Récupère l'objet permettant de manipuler la LED.
led = digitalio.DigitalInOut(board.LED)
led.direction = digitalio.Direction.OUTPUT

# TODO complétez pour faire clignoter la LED
```

Indice : `led.value` est un booléen qui permet de choisir si la LED est allumée.

▶ **Question 3.** La fonction `getGPIOStatus` ci-dessous permet de détecter si le pin passé en paramètre (par exemple `GP0`) est relié à la terre. Ajouter cette fonction à votre code, et faite en sorte de pouvoir désactiver l'exécution du code (celui qui fait clignoter la LED) si le GPIO GP0 est relié à un pin GND.

```
def getGPIOStatus(pin=board.GP0):
    progStatusPin = digitalio.DigitalInOut(pin)
    progStatusPin.switch_to_input(pull=digitalio.Pull.UP)
    progStatus = not progStatusPin.value
    return(progStatus)
```

Testez et validez que votre code fonctionne comme prévu.

2 Transformation en Rubber Ducky

Maintenant que vous avez vu le fonctionnement de base, passons aux choses sérieuses. Cette étape consiste à paramétrer le Pico pour qu'il agisse comme une Rubber Ducky, c'est-à-dire qu'il se fasse passer pour un clavier lorsqu'on le branche, et qu'il "tape" les touches clavier fournies dans un fichier lorsqu'on le branche.

Vous avez déjà fait la première étape en flashant la carte avec CircuitPython ! Pour le reste, vous allez reprendre un projet adapté pour les besoins de ce TP, disponible sur moddle.

> ❗ Lisez et suivez attentivement les instructions, elles vous permettront d'utiliser correctement la rubber ducky une fois celle-ci configurée.

Pour cela suivez les instructions suivantes :

1. **Télécharger le projet :** Téléchargez le projet sur Moodle.
2. **Lancez le formatage de la carte :** Ne branchez pas encore le Raspberry. Un script d'installation automatique est disponible à la racine du dossier (`installer.sh`). Lancez-le et suivez les instructions.

3. Si le script termine sans soucis, votre Rubber Ducky est prête à être utilisée ! Sinon, vous trouverez une procédure d'installation pas à pas dans le fichier README. Les liens de téléchargement des diverses dépendances sont listés dans le script d'installation.

4. **Utilisation des GPIO :** Les GPIO (*General Purpose Input/Output*) du Raspberry Pico 2 permettent d'interagir avec divers composants électroniques. Ils permettent également de paramétrer certaines actions du Pico, en reliant certains *pins* à la terre (`GND`). Leur utilisation est documentée dans le dépôt git.

> **Attention** À partir de maintenant, si vous ne branchez pas les GPIO adéquats pour être en mode configuration, la carte tentera d'exécuter les DuckScripts dès que vous la branchez à un PC !

Les questions suivantes ont pour but de vérifier que vous avez bien pris connaissance des éléments clés de l'utilisation de la carte que vous venez de configurer. Vous trouverez les réponses dans le fichier README.md.

▶ **Question 1.** Comment passer la carte en mode configuration, et qu'est-ce que ça implique ?

▶ **Question 2.** Comment désactiver le mode stockage ?

▶ **Question 3.** Où stocker le script à faire exécuter par la rubber ducky ?

▶ **Question 4.** Résumer en quelques ligne la démarche à effectuer pour configurer le lancement d'un nouveau script par la rubber ducky. Précisez bien quand brancher la carte au PC, et quel pin connecter. Il serait judicieux de faire valider ce protocole par un ou une chargé.e de projet avant de poursuivre.

3 Premier DuckyScript

Pour savoir quelles touches clavier émuler au branchement, la carte va utiliser un "script" au format DuckScript*. Le principe est de créer un fichier texte classique contenant une suite de touche clavier que la rubber ducky devra effectuer lorsqu'elle est branché à une cible.
Le format est donc assez lisible, bien que verbeux. Pour des utilisations plus avancées, il est possible de créer des fonctions, de boucles, ... mais vous ne devriez pas avoir besoin d'avoir recours à ces fonctionnalités pour ce TP.

Certains mot-clés permettent d'effectuer certaines actions, tandis que d'autre vont directement émuler des frappe clavier. Par exemple :

- `STRING abcd` va taper la chaine de caractère "abcd"
- `DELAY 100` va attendre 100ms avant de traiter la prochaine commande (ça peut être utile s'il faut patienter pour l'ouverture d'un terminal par exemple)
- `REM`: toute ligne commençant par `REM` est un commentaire et ne sera pas traité.
- Voici une liste de touches particulièrement utiles pour les raccourcis clavier : `ALT` `CTRL` et `ENTER` ont des noms suffisament clairs. La touche "windows" correspond à `GUI` ou `WINDOWS`.
- Si plusieurs touches sont listées sur la même ligne, elles seront pressées en même temps, et relachées à la fin de la ligne. Par exemple une ligne `CTRL C` fera une copie du texte sélectionné.

▶ **Question 1.** Identifiez deux moyens, sur votre ordinateur, d'accéder à un terminal en utilisant uniquement le clavier.

*https://docs.hak5.org/hak5-usb-rubber-ducky/ducky-script-basics/hello-world

▶ **Question 2.** Créez un script DuckyScript simple pour afficher le message ”Hello World” dans un éditeur de texte., et sauvegardez-le dans le fichier `payloads/payload.dd` sur votre rubber ducky. Testez votre script en le branchant sur l'ordinateur cible. Le script sera à joindre au rendu du projet.

> **Note : :** Un petit bug dans le code fait que certaisn raccourcis sont mal interpretés (en cours de correction). Si le raccourci clavier actuel pour ouvrir le terminal ne fonctionne pas, vous pouvez le redéfinir pour ALT+T.

▶ **Question 3.** Modifiez le script pour qu'il ouvre un terminal et affiche des informations sur le CPU (via la commande `uname -a` sur Linux).