# Rubber Ducky

Jakub Chuchla, Olaf Sujata, Łukasz Czerwiec, Małgorzata Andrasz

May 2023

## Spis treści



# 1 Słownik

- HID - human input device lub human interface device. Urządzenia służące do wsprowadzania danych przez człowieka. Należą do nich między innymi klawiatura, mysz, czy dżojstik.
- USB HID - technologia umożliwiająca podłączenie urządzenia do komputera bez konieczności wgrywania sterownika.
- HID usage table -

## 2 Wstęp

USB Rubber Ducky jest urządzeniem przypominającym pendrive'a, lecz wykrywanym przez komputer, jako urządzenie HID. Należy do kategorii "bad USB", czyli narzędzi, które mają wyrządzić szkodę po podłoczeniu do portu USB komputera. Jego twórcą jest Darren Kitchen, który stworzył je, by ukazać słabość modelu zaufania komputera, który po wykryciu USB Rubber Ducky, jako klawiatury, ufa, że to użytkownik komunikuje się z nim poprzez klikanie i pisanie.

Projekt Rubber Ducky ma na celu poznanie sposobu komunikacji z urządzeniami peryferyjnymi. Przygotowaliśmy urządzenie z oprogramowaniem emulującym wciśnięcia klawiszy klawiatury po podłaczeniu do komputera przez USB. Zasada działania jest analogiczna do urządzenia Rubber Ducky.

Rubber to z pozoru przypominające pendrive urządzenie zaliczane jest do kategorii tzw. bad USB, czyli narzędzi, które są nam w stanie wyrządzić jakąś szkodę po podłączeniu do portu USB komputera

## 3 Opis działania

W celu wykorzystania urządzenia USB Rubber Ducky należy je najpierw zaprogramować. Programowanie odbywa się poprzez wgranie napisanego i przekompilowanego wcześniej na postać binarną skryptu na umieszczoną w urządzeniu kartę pamięci. Po wgraniu skryptu na USB Rubber Ducky, aby nośnik zadziałał trzeba go podłączyć do portu USB komputera. Po wpięciu do portu USB Rubber Ducky jest wykrywane przez komputer jako klawiatura i zaczyna wysyłać napisane wcześniej ciągi znaków, poleceń i skrótów klawiszowych.

Rubber Ducky, chociaż wygląda jak pamięć flash, w rzeczywistości jest wykrywany przez system operacyjny jako urządzenie HID (Human Interface Device), a konkretnie klawiatura. Po podłączeniu do komputera zaczyna wysyłać zaprogramowane wcześniej ciągi znaków, poleceń i skrótów klawiaturowych. Efekt jest zatem taki, jakby osoba, która podłącza urządzenie do swojego komputera pozwoliła intruzowi skorzystać z własnej klawiatury.

## 4 Zagrożenia

Rubber Ducky niesie ze sobą wiele różnych zagrożeń:

- Pozwala stworzyć fałszywe okno logowania do Windowsa wykradając w ten sposób dane logowania lub przesłać wszystkie hasła przeglądarki Chrome na serwer hackera.
- Umożliwiają sprawdzenie do jakiego komputera został podłączony hakerski pendrive - PC czy Mac. Na podstawie otrzymanych informacji, wykonane zostaną polecenia, które będą kompatybilne z danym sprzętem. Ponadto Rubber Ducky może teraz kodować dane w formacie binarnym i przesyłać je wykorzystując sygnały zapalające na klawiaturze lampkę od caps locka.

- Kradzież danych logowania - Rubber Ducky może skopiować lub przechwycić wprowadzane dane logowania, takie jak nazwy użytkowników i hasła, które następnie mogą być wykorzystane przez atakującego do uzyskania nieautoryzowanego dostępu do kont i systemów.

- Wstrzykiwanie złośliwego kodu: Atakujący może użyć Rubber Ducky do wstrzykiwania złośliwego kodu lub skryptów na zainfekowanym komputerze. Może to prowadzić do instalacji szkodliwego oprogramowania, takiego jak wirusy, trojany czy keyloggery, które mogą kontrolować system lub kraść dane.

- Wykonanie zautomatyzowanych działań: Rubber Ducky może wykorzystać skrypty do wykonywania automatycznych działań, takich jak uruchamianie szkodliwego oprogramowania, przechwytywanie obrazów ekranu, zmiana ustawień systemowych lub przeglądarki, co może prowadzić do utraty prywatności, naruszenia bezpieczeństwa lub poważnych szkód dla użytkownika.

- Przejęcie sesji lub zdalne sterowanie: Atakujący może wykorzystać Rubber Ducky do przejęcia sesji użytkownika lub zdalnego sterowania komputerem, co pozwala im na monitorowanie działań, kradzież danych lub kontrolę nad systemem.

- Wprowadzanie zmian w systemie: Rubber Ducky może zmieniać ustawienia systemowe, takie jak firewall, antywirusy czy aktualizacje, aby umożliwić łatwiejsze wdrożenie złośliwego oprogramowania lub uniknąć wykrycia przez zabezpieczenia.

- Atak fizyczny na infrastrukturę: Rubber Ducky może być wykorzystany do przeprowadzenia ataków fizycznych na infrastrukturę, takie jak wstrzykiwanie złośliwego kodu w systemy sterowania przemysłowego, bankomaty lub urządzenia IoT.

# 5 Działania prewencyjne

Takie ataki mogą być bardzo groźne dla naszych urządzeń jak i dla nas - ktoś może wykraść dane do logowania, np. do banku. Oto kilka sposobów, jak możemy bronić nasz komputer przed takimi atakami:

- Świadomość i edukacja - dowiedz się więcej o atakach "Rubber Ducky". Wiedza na temat zagrożeń i podstawowych zasad bezpieczeństwa cyfrowego pomoże Ci lepiej rozpoznawać potencjalne zagrożenia i unikać ich.

- Uniemożliwienie kontaktu fizycznego z urządzeniem: nie możemy pozwolić na podłączenie pendrive'a nieznanego pochodzenia do naszego komputera. Nie zostawiaj nigdy swojego urządzenia bez opieki w publicznym miejscu.

- Zaktualizowane oprogramowanie: Regularnie aktualizuj oprogramowanie na swoim komputerze, w tym system operacyjny, przeglądarkę internetową i inne aplikacje. Aktualizacje często zawierają łatki bezpieczeństwa, które chronią przed znanymi lukami w zabezpieczeniach,
- Programy antywirusowe i antymalware: Zainstaluj i regularnie aktualizuj programy antywirusowe i antymalware na swoim komputerze. Skanuj system w poszukiwaniu potencjalnych zagrożeń. Uważaj na ostrzeżenia i blokady, które mogą wskazywać na potencjalne złośliwe oprogramowanie na twoim komputerze.
- Firewall: Włącz firewall (zaporę sieciową) na swoim komputerze, aby kontrolować ruch sieciowy i blokować podejrzane połączenia.
- Wyłączanie automatycznego uruchamiania: Zablokuj automatyczne uruchamianie skryptów lub programów z urządzeń zewnętrznych, takich jak pendrive'y. Możesz to zrobić, dostosowując ustawienia urządzenia lub oprogramowania do zarządzania urządzeniami pamięci masowej.
- Monitorowanie aktywności USB: Jeśli jesteś zaniepokojony możliwością ataku "Rubber Ducky", możesz skorzystać z narzędzi monitorujących aktywność na porcie USB, takich jak USBGuard, aby wykryć i zablokować podejrzane działania.

Trzeba pamiętać, że żadna metoda nie jest w pełni skuteczna i niezawodna. Dlatego warto stosować kombinację różnych środków ostrożności i zasad bezpieczeństwa cyfrowego, aby jak najlepiej chronić się przed różnymi rodzajami ataków, w tym atakiem "Rubber Ducky".

# 6 Cel projektu

Projekt ma na celu poznanie sposobu komunikacji z urządzeniami peryferiów. Stworzyliśmy cztery Rubber Ducky -każde z nich wykonuje inne zadanie, mianowicie:

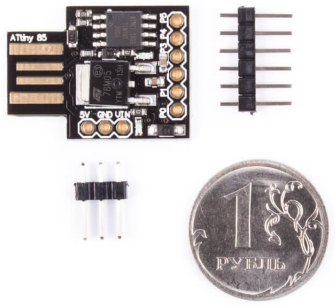
- Wyświetlenie krótkiego tekstu w notatniku
- Wykonanie prostego skryptu
- Zmiana przycisków na klawiaturze podłączonej do komputera
- DOPISAĆ CO ROBI PROGRAM 4

Dokładne opisy napisanych programów można znaleźć w sekcji 8. Napisanie czterech programów o różnych działaniach ma na celu pokazanie w praktyce jak różne zastosowanie może mieć Rubber Ducky.

# 7 Sposób implementacji

## 7.1 Geneza wyboru mikrokontrolera

Wybrany przez nas mikrokontroler to Digispark Attiny 85, pokazany na poniższym zdjęciu obok monety (źródło: https://3d-diy.ru/product/digispark-attiny-85).




Rysunek 1: Digispark Attiny 85

Jak widać na powyższym zdjęciu, nasz mikrokontroler jest bardzo mały. Jest to jednym z powodów, przez które został przez nas wybrany. Przez jego wielkość chcieliśmy pokazać, że nawet takie małe urządzenie może być bardzo niebezpieczne. Drobna wielkość wcale nie oznacza, że nasze Rubber Ducky ma mniejsze możliwości - jedynym ograniczeniem była nasza wyobraźnia i wielkość pamięci mikrokontrolera. Kolejnym powodem wyboru tego mikrokontrolera jest fakt, że ma on wbudowane złącze USB, dzięki czemu możemy użyć go jako Rubber Ducky od razu po napisaniu kodu, bez konieczności modyfikowania hardware'u. Wybraliśmy ten mikrokontroler również dlatego, że można o nim łatwo znaleźć dużo informacji oraz jest kompatybilny ze środowiskiem programistycznym Arduino. Na wybór tego mikrokontrolera wpłynęła również jego niska cena.

## 7.2 Sposób napisania programów

W poprzednim podrozdziale wspomnieliśmy, że jednym z powodów wyboru mikrokontrolera była kompatybilność ze środowiskiem programistycznym Arduino. Chcieliśmy napisać nasze programy w tym IDE, ponieważ oprócz tego, że jest popularne, wspierano ono pisanie kodu w języku C++. Wybraliśmy ten język, ponieważ jesteśmy z nim dobrze zaznajomieni i używamy go również na innych kursach znajdujących się w programie naszych studiów. Kod pisaliśmy używając biblioteki DigiKeyboard.h. Bibilioteka ta umożliwiła nam pracę, ponieważ

zawarte w niej funkcje pozwalają emulować działanie klawiatury. W bibliotece zdefiniowano wartości znaków odpowiadającym klawiszom klawiatury oraz przyciskom modyfikującym takim jak CTRL, ALT bądź GUI, dzięki czemu nie musieliśmy sami szukać ich w HID usage table. Ważne jest też to, że funkcje w tej bibliotece pozwalają nam na emulowanie kilku klawiszy jednocześnie, ponieważ ze względu na to, że mamy jedynie klawiaturę, musimy używać skrótów klawiszowych systemu Windows, żeby móc zapisać i włączyć napisany przez nas skrypt na komputerze do którego podłączymy nasze Rubber Ducky. W napisanych przez nas programach używamy funkcji delay, która każe skryptowi zaczekać przed wykonaniem dalszych instrukcji. Jest to ważne, by Rubber Ducky działało niezawodnie. Niektóre komputery mogą potrzebować trochę więcej czasu na przetworzenie danych wysyłanych przez klawiaturę. Dzięki zaczekaniu chwili przed "wciśnięciem" kolejnych klawiszy mamy pewność, że poprzednia akcja zostanie obsłużona przed próbą wykonania kolejnej - daje nam to pewność poprawnego wykonania się wszystkich instrukcji. System może mieć również problem z przepustowością przetwarzania danych z klawiatury, przez co przy zbyt szybkim wysłaniu zbyt wielu poleceń, któreś może się nie wykonać.

## 7.3 Napotkane problemy

- Pierwszym problemem jaki napotkaliśmy była instalacja sterowników do naszego mikrokontrolera, ale po krótkiej "zabawie" z nimi, mogliśmy zacząć pisać kod.
- Kolejny problem pojawił się przy próbie wypisania podanego tekstu na ekran przy użyciu funkcji println. Po przeanalizowaniu napisanego kodu oraz tabeli ASCII okazało się, że mikrokontroler wyświetla znaki cofnięte o wartość 8 w kodzie ASCII, więc przy wyświetlaniu tekstów musimy dodać tę wartość do każdego ze znaków. Błąd naprawiliśmy używając pętli for. Mylne działanie instrukcji wynika prawdopodobnie z tego, że instrukcja println nie jest zdefiniowana w bibliotece DigiKeyboard.h, lecz w bibliotece Print, która jest dziedziczona przez DigiKeyboard.h. Pomimo jej mylnego działania umożliwia nam ona przesyłanie długich ciągów znaków (string) co bardzo ułatwia pisanie kodu.
- Nie wszystkie przyciski były zdefiniowane w bibliotece DigiKeyboard.h, a ich użycie było niezbędne do działania skryptu. By poradzić sobie z problemem znaleźliśmy odpowiadające wartości takowych przycisków w HID Usage Table. Przykładowo "strzałce w dół" odpowiadają wartości 81/0x51. Przez wysłanie "stroke'a" takiej wartości (dziesiętnej/szesnastkowej) otrzymywaliśmy zamierzony efekt.
-

# 8 Opis działania programów

# 9 Wnioski

# 10 Podsumowanie

# Literatura


[1] https://opensecurity.pl/arsenal-ethical-hackera-rubber-ducky/

[2] https://www.komputerswiat.pl/aktualnosci/bezpieczenstwo/falszywy-hakerski-pendrive-stal-sie-jeszcze-grozniejszy-jest-jeden-sposob-zeby-sie/7l4v3re

[3] https://chat.openai.com/

[4] https://github.com/digistump/DigisparkArduinoIntegration/blob/master/libraries/DigisparkKeyboard/D