PRUEBAS DE VULNERABILIDAD EN EL SISTEMA EMBEBIDO DE UN ROBOT MÓVIL IORT

VULNERABILITY TESTS IN THE EMBEDDED SYSTEM OF A MOBILE ROBOT IORT

Luis Alberto Flores Montaño
Instituto Politécnico Nacional, México
lfloresm1703@alumno.ipn.mx

Juan Carlos Herrera Lozada
Instituto Politécnico Nacional, México
jlozada@ipn.mx

Jacobo Sandoval Gutiérrez
Universidad Autónoma Metropolitana, México
j.sandoval@correo.ler.uam.mx

Recepción: 29/noviembre/2022 **Aceptación:** 14/abril/2023

Resumen

Dentro de la arquitectura del Internet de las Cosas Robóticas IoRT se tienen problemas de vulnerabilidad en los sistemas embebidos debido a la falta de ciberseguridad. Los estudios similares protegen los sistemas del Internet de las Cosas (IoT), pero no a la arquitectura del IoRT. Por ello, el objetivo del trabajo es demostrar tres pruebas de vulnerabilidad de un sistema embebido tipo Raspberry Pi 4.0 dentro de un robot móvil que opera en el IoRT. La metodología consiste en distinguir un punto de vulnerabilidad con la plataforma del robot, diseñar un ataque al sistema embebido con la herramienta "*Rubber Ducky*" a través de un código malicioso, y analizar las implicaciones indirectas sobre el robot. Estos experimentos consisten en atacar el cliente que opera el robot y después dos ataques al servidor. Finalmente, se prueba la facilidad de recibir ataques si no se tienen protocolos ciberseguros.

Palabras Clave: Ciberataque, IoRT, Raspberry, Rubber Ducky, vulnerabilidad.

Abstract

The architecture of the Internet of Robotic Things IoRT there are vulnerability problems in embedded systems due to lack of security. Similar studies protect Internet of Things (IoT) systems, but not the IoRT architecture. Therefore, the objective of the work is to demonstrate three vulnerability tests of a Raspberry Pi 4.0 type embedded system within a mobile robot that operates in the IoRT. The methodology consists of distinguishing a point of vulnerability with the robot platform, designing an attack on the embedded system with the "Rubber Ducky" tool through malicious code, and analyzing the indirect implications on the robot. These experiments consist of attacking the client operating the robot and after two attacks on the server.

***Keywords*:** *Cyberattack, IoRT, Raspberry, Rubber Ducky, vulnerability.*

1. Introducción

Actualmente existen diversos dispositivos computacionales que son capaces de procesar, transmitir y mandar información de diversas formas; tal es el caso de las laptops, computadoras de escritorio y dispositivos móviles (teléfonos inteligentes y tabletas). Por otro lado, también existen los sistemas embebidos, los cuales pueden tener sistemas de computación basados en microprocesadores o microcontroladores, diseñados para realizar una tarea en específico, o en su defecto diversas de ellas en tiempo real [Stankovic,1996]; un ejemplo de estas son las computadoras de placa única de bajo costo, como es el caso de tarjetas Raspberry Pi (teniendo diversas versiones) que han funcionado como base de diversas plataformas digitales y robóticas. Tomando en cuenta lo anterior, estos dispositivos computacionales tienen algo en común, la entrada de información a través de dispositivos periféricos; dicho de otra forma, pueden recibir cualquier tipo de dato ya sea de un micrófono, cámara, ratón, o en su defecto el teclado convencional como un dispositivo muy frecuente. Mencionado lo anterior, se tienen teclados estándar que son reconocidos y clasificados como de interfaz humana (HID), y estos suelen ser aceptados sin restricción por una amplia gama de dispositivos, así como los sistemas operativos conocidos como "*plug and play*" [Li, 2018].

No obstante; los teclados suelen conectarse por lo general en interfaces de unidades USB (en inglés Universal Serial Bus). Estas conexiones suelen ser potencialmente peligrosas, por lo que pudieran provocar amenazas y vulnerabilidades en los dispositivos computacionales donde son utilizados. En específico, se considera que existen pendrives maliciosos con las mismas propiedades que tiene cualquier HID, estos pendrives tendrían la misma jerarquía y acceso de un teclado convencional, por tanto, se clasifica para estos pendrives como una herramienta de piratería.

Acorde con los investigadores [Walter, 2012] y [Falliere, 2011]; el uso de almacenamiento mediante puertos USB, se han podido utilizar como un programa "malware" o también conocido como programa malicioso. Es decir, hay evidencia de la detección de ciberataques implementados sobre los puertos, que se conocen generalmente como ataques físicos o ataques de inyección, o en este caso, siendo específicos los ataques "BadUSB" en [Al-Zarouni, 2006] y [Bhakte, 2016]. Este último, consiste en emularse como un dispositivo normal y realizar de forma encubierta acciones en la computadora (máquina huésped) [Griscioli, 2016] y [Tyutyunnik, 2021]. Las unidades flash USB que pueden registrarse como dispositivo o teclado cuentan con la capacidad de inyectar scripts maliciosos. Una forma de lograr la función previamente descrita es mediante la herramienta desarrollada por la empresa Hak5, llamada "USB Rubber Ducky", o "Patito de hule". La razón para no ser detectado como un dispositivo ajeno o potencialmente riesgoso se obtiene por el tipo de firmware con que la herramienta del dispositivo se presenta en los puertos USB y, por tanto, no puede ser detectado por la máquina huésped o por los antivirus actuales, esto imposibilita la protección ante estos ataques [Cannoles, 2017].

Acorde con los investigadores [Tian, 2015] y [Harianto, 2019], los ataques no solo se limitan a las memorias o teclados que hacen uso de USB, este tipo de problemas o ataques se pueden realizar con cualquier dispositivo que utilice un puerto USB, por lo que, cualquier equipo puede ser vulnerable e inhabilitado. Por otra parte, los dispositivos que utilizan puertos USB son demasiados sencillos o simples para funcionar de manera confiable, por lo que asegurar o autenticar estos dispositivos

suele ser anormal en distintos casos, por lo tanto, defenderse de este tipo de ataques deja grandes expectativas a resolver en cuanto a la seguridad de estos [Arora, 2021].

Adicionalmente, se pueden encontrar otros métodos capaces de infiltrarse a una máquina como el caso de un probador de penetración, también conocido como "Pentesting", en donde son explotadas las vulnerabilidades del sistema, otro método muy utilizado para obtener información sensible y facilitar el acceso o infiltración puede ser lograda mediante la ingeniería social.

No obstante, en [Papp, 2015] se hace una recopilación de diversas investigaciones donde se tienen presentes las vulnerabilidades de los sistemas embebidos, como es el caso de la carencia de encriptación y autentificación. Dando como resultado que, los dispositivos encargados de mandar mensajes pueden ser husmeados, falsificados o reproducidos. También, el autor menciona algunos requerimientos para que un atacante o pirata informático pueda vulnerar con facilidad a estos dispositivos, dichos requerimientos se encuentran orientados al internet, que tengan un acceso local o remoto al dispositivo, así como un acceso físico, y una proximidad física.

Algunos de los dispositivos con sistemas embebidos que han sido vulnerados son inalámbricos, SCADA con protocolos ModBus, RuggedCom, para domótica inteligente como es el caso del termostato Nest, así como enrutadores de banda dual Wireless-N.

Teniendo en cuenta lo anterior, la estrategia de usar un dispositivo "BadUSB", y ejecutar un código malicioso sin el conocimiento de la víctima, ha llegado a mostrar diversas ventajas, para los atacantes o piratas informáticos, como puede ser el tiempo de ejecución, la obtención de datos y la alteración del funcionamiento correcto del equipo o dispositivo [Griscioli, 2016]. Un escenario de esto podría ser que la víctima tiene un descuido y deja su equipo de trabajo por unos momentos, el pirata informático o "hacker", podría conectar USB, y ejecutar códigos maliciosos o "scripts" en la máquina huésped [Cannoles, 2017]. En [Vouteva, 2015], se menciona que pueden realizarse ataques similares a "BadUSB" o RubberDucky reemplazando con otras herramientas complementarias como un Micro Arduino.

También se tienen otras investigaciones similares, en [Hariato, 2017] y [Cannoles, 2017], se desarrollan una secuencia de comandos en la USB Rubber Ducky, esto con el propósito de obtener una identificación de inicio de sesión en texto claro, y las contraseñas de una máquina con Windows; adicionalmente, los autores hacen uso de otras herramientas exploits como es el caso de Powershell y Mimikatz; las vulnerabilidades operan desde versiones de Windows 7 y posteriores.

El fabricante específico de la USB Rubber Ducky es Hak5. El dispositivo incluye un microcontrolador con herramientas programables. Una característica es la imitación de un teclado con la apariencia de cualquier unidad flash USB; esta característica facilita estar oculta en el puerto de un equipo de cómputo, también se mantiene oculta en el administrador de tareas, por lo que llega ser indetectable para el sistema operativo. Además, cuenta con una amplia gama de cobertura para los diversos distros de Linux, así como en Windows o MacOS.

Es importante mencionar que para realizar un ataque con “Rubber Ducky” se necesita tener un acceso físico a la máquina víctima, así como un “script” programado conocido como “Duckyscript”, el cual contiene el malware y es ejecutado en el dispositivo o equipo huésped.

Como ya se ha mencionado los equipos de cómputo confían plenamente en los dispositivos que forman parte del HID (como puede ser un teclado), ya que estos logran una interacción y comunicación máquina-humano. Mencionado lo anterior, el USB Rubber Ducky funciona como un emulador de teclado “disfrazado” de un dispositivo muy parecido al de una memoria flash USB. La capacidad de operación de la herramienta es suficiente con un microcontrolador programable de 60 MHz, así como una ranura SD. La herramienta ha sido utilizada por diversos profesionales de tecnologías de la información, probadores de penetración (en inglés pentesters), y por piratas informáticos desde hace más de una década, por lo que, se ha convertido en una plataforma comercial muy utilizada, para los ataques físicos y de inyección, haciendo uso de pulsaciones de teclas en el sistema o máquina huésped [Zhao, 2019].

Como ya se ha mencionado el insumo para los ataques son los scripts o si se utiliza Rubber Ducky, es a través de Duckyscripts, los cuales contienen una combinación

de lenguaje de secuencias de comandos, conocidas como cargas útiles (payloads); estas pueden ser escritas y desarrolladas como el autor del ataque lo deseé. La figura 1 muestra la herramienta de USB Rubber Ducky de la empresa Hak5.

Figura 1 USB Rubber Ducky.

Esta herramienta para ser funcional requiere solo que los usuarios dejen desatendidos sus equipos de cómputo durante unos segundos, es decir, solo el tiempo necesario para tener la oportunidad de colocar la herramienta directamente en el puerto sin otro requerimiento adicional.

En este caso la investigación se ha orientado hacia una plataforma robótica donde hay dos sistemas el cliente y el servidor (robot). Dicha plataforma robótica está basada en la arquitectura del Internet de las Cosas Robóticas (IoRT), la cual es una tecnología que busca monitorear, operar y mantener las tareas de los múltiples robots a través de la nube [Flores-Montaño, 2021]. Acorde con la literatura en [Eswaran, 2020], [Mourtzis, 2019], [Andrea, 2015], [Basheer, 2019], [Varga, 2017] y [Jamai, 2020] se muestran las diversas vulnerabilidades en sistemas IoRT, como pueden ser las credenciales por defecto, ausencia de mecanismo de bloqueo, puertos abiertos no deseados, ausencia de factores de autenticación; con dichas vulnerabilidades es posible efectuar diversos ataques como es el caso del de lógica falsa, tráfico de análisis, interferencia, sumidero, de hombre en el medio, DDoS, entre otros. No obstante, existe una vulnerabilidad que es el enfoque de esta investigación, la cual los puertos externos no inhabilitados, por lo que en dicha vulnerabilidad se puede realizar los ataques físicos y de inyección de código malicioso.

Dentro de la plataforma robótica IoRT se evalúan en los ataques mencionados (físico y de inyección de código malicioso). Para dicha plataforma se implementa sobre un equipo de cómputo (laptop) con un sistema operativo Windows 11, y el otro en un sistema embebido (Raspberry Pi 4) con un sistema operativo Raspbian OS, la cual está conectado al modelo de robot móvil “4WD SmartCar” de la empresa Freenove, dicho robot se muestra en la figura 2.

Figura 2 Robot móvil, modelo: 4WD Car Freenove.

2. Métodos

Como se mencionó en la parte de la introducción, algunos de las vulnerabilidades en dispositivos que usan sistemas embebidos, como es el caso de dispositivos SCADA, RuggedCom, Nest, entre otros; dichos dispositivos suelen fallar como es en el caso de la encriptación y autentificación dando como resultado que sean falsificados o husmeados; otro aspecto importante que también se mencionó fue acerca de que los atacantes, muchas veces necesitan acceso físico o una proximidad física para realizar ataques. Por lo que acorde con estas vulnerabilidades y los métodos que el atacante suele realizar se procede a ejecutar este tipo de pruebas de penetración para vulnerar el sistema embebido que en este caso es una Raspberry Pi 4. Mencionado lo anterior, el método dentro de esta investigación se desarrolla un ataque de tipo inyección y a la vez físico, por lo que es necesario hacer uso de distintas herramientas, siendo estas de tipo hardware y software, por lo que durante esta sección se analizan diversas herramientas y tecnologías que se utilizan para el ataque.

Para la realización de estos ataques, es necesario tener un enfoque de la máquina objetivo o huésped a la cual se le realiza un ataque, como un análisis más adelante, esto es importante debido a que las cargas útiles son programadas de distinta manera, dependiendo el sistema operativo y el lenguaje del teclado del cual es utilizada por la máquina huésped.

Teniendo en cuenta lo anterior, se utilizan dos tipos de máquinas huésped una es una computadora Huawei D15 con Windows 11, 64 bits, la cual, utiliza un antivirus Bitdefender, para su protección; por otro lado, el otro dispositivo huésped es la microcomputadora Raspberry pi 4, esta última funcionando como el cerebro del robot móvil 4WD Car Freenove. Para la realización del ataque se hace uso de la herramienta USB Rubber Ducky, la cual es conectada a las máquinas huéspedes o máquinas objetivos.

Para realizar el ataque, se utiliza un lenguaje de comando para escribir las cargas útiles de programa malicioso, también conocidas como *"malware payload"*, las cuales son secuencias de comandos utilizadas por Rubber Ducky. Este tipo de escritura puede hacerse desde cualquier editor de textos, como puede ser el caso de un Wordpad o un Bloc de notas. La herramienta tiene algunas restricciones de sintaxis como son escribir cada comando en mayúsculas y en una nueva línea, adicionalmente se pueden invocar pulsaciones de teclas, combinaciones de estas o cadenas de texto para ofrecer retrasos o pausas.

Algunos de los comandos más utilizados son DELAY y STRING. DELAY es seguido de un número que representa milisegundos. Un ejemplo podría ser "DELAY 4000", en donde 4000 milisegundos, representa 4 segundos, significa que en la secuencia de comandos tendrá una pausa de ejecución de este tiempo, antes de pasar al siguiente comando. Lo anterior es necesario para asegurar que la secuencia de comandos funcione de manera adecuada y sin ningún tipo de problemas, también funciona para llevar la secuencia de comandos del Rubber Ducky a la computadora objetivo o huésped, ya que pudieran estar desactualizadas o ser más lentas.

Por otro lado, el comando STRING, indica el procesamiento del texto escrito, por lo que podrían escribirse uno o más caracteres haciendo uso de este comando. También se puede utilizar el comando WINDOWS (o GUI), el cual, sirve para emular

el botón de Windows; cabe mencionar, que para el caso de los sistemas operativos Linux, se utiliza la combinación de teclas ALT -t. En la figura 3, se muestra un ejemplo sencillo de la serie de comandos para el Duckyscript, el cual se encarga de mostrar un mensaje en un Bloc de notas y escribir la frase “Hola estoy en tu máquina”, esta secuencia de comando se realiza en cuestión de milisegundos, posteriormente de conectar el Rubber Ducky.

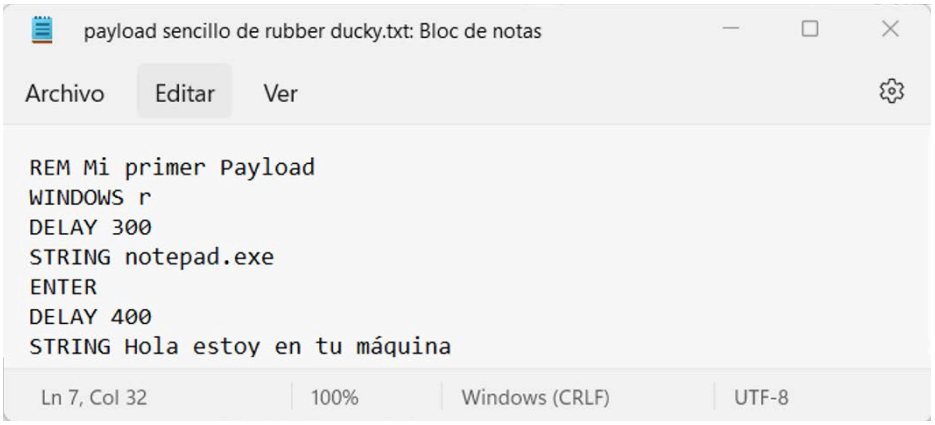

Figura 3 Ejemplo de una secuencia de comandos para Rubber Ducky.

Adicionalmente, se pueden realizar diversos payload de Duckyscript para distintas tareas, siempre cuando se puedan realizar mediante instrucciones provenientes de un teclado o HID; por lo que es posible realizar distintas tareas, como puede ser el caso de apagar la máquina, modificar los diversos ajustes del equipo, o en su defecto modificar documentos de cualquier tipo, adicionalmente también es posible extraer información del equipo y enviarla a través de la red.

En la tabla 1 se da una breve descripción de los principales comandos a utilizar dentro de la carga útil, payload, Duckyscript dentro de Rubber Ducky de acuerdo con [Wolfang, 2022]. Una vez que se tiene los comandos se procede a generar la carga útil o el Ducky Script, para lograr esto, es necesario utilizar una herramienta en línea conocida como Duck Toolkit NG, dicha herramienta es una plataforma de código abierto, que se puede encontrar en línea en la página oficial [Ducktoolkit, 2022]; cabe mencionar que se puede utilizar para generar diversos scripts ya sea en Windows, Mac OSX o Linux. En resumen, la herramienta ayuda a pre-construir

cargas útiles o payloads, y decodificar algunas existentes; en este caso como ya se tienen los comandos únicamente se procede a decodificar. Cabe resaltar que este kit tiene, acceso a internet y powershell, con el propósito de colocar los comandos dentro de esta; adicionalmente se elige el idioma que usa la máquina.

Tabla 1 Comandos en Rubber Ducky.

Sintaxis	Descripción
REM	Comentarios, código que no será procesado.
DEFAULT_DELAY / DEFAULTDELAY	Se debe definir un valor entre 0 y 10000 (milisegundos), el cual establecerá un delay a todas las secuencias de comandos.
DELAY	Define un valor entre 0 y 10000 (milisegundos), el cual establecerá un delay entre secuencias de comandos.
STRING	Permite inyectar cadenas de caracteres: a-z A-Z 0-9! -) `~+=_- “‘;:<,>.?[{]}/|!@#$%^&*().
WINDOWS / GUI	Equivale a la tecla windows o la tecla cmd en macos.
APP / MENU	Equivale a la combinación de teclas SHIFT F10 de windows (clic derecho).
SHIFT	Simula la tecla SHIFT, y esta se puede combinar con: DELETE, HOME, INSERT, PAGEUP, PAGEDOWN, WINDOWS, GUI, UPARROW, DOWNARROW, LEFTARROW, RIGHTARROW, TAB.
ALT	Simula la tecla ALT, y esta se puede combinar con: END, ESC, ESCAPE, F1-F12, caracteres simples (ejemplo: f, s), SPACE, TAB.
CONTROL / CTRL	Simula la tecla CTRL, y esta se puede combinar con: BREAK, PAUSE, F1-F12, ESCAPE, ESC, caracteres simples.
DOWNARROW / DOWN	Simulan las teclas de dirección.
LEFTARROW / LEFT	Simulan las teclas de dirección.
RIGHTARROW / RIGHT	Simulan las teclas de dirección.
UPARROW / UP	Simulan las teclas de dirección.
BREAK / PAUSE	Equivale a la combinación CTRL BREAK.
CAPSLOCK	Es la tecla que permite escribir en mayúsculas o minúsculas.
DELETE	Simula la tecla suprimir.
END	Tecla que permite ir al final de algo (página web, documento, etc.).
ESC / ESCAPE	Tecla de escape.
HOME	En español equivale a la tecla inicio.
INSERT	Simula la tecla insertar.
NUMLOCK	Tecla para bloquear o desbloquear los numerales en los teclados.
PAGEUP	Simula la tecla Page Up.
PAGEDOWN	Simula la tecla Page Down.
PRINTSCREEN	Tecla para tomar screenshots.
SCROLLLOCK	Simula la tecla Scroll Lock.
SPACE	Tecla espacio.
TAB	Tecla de tabulación.
REPEAT	Repite la cantidad de veces que le indiquemos lo ya ejecutado.

En la figura 4 se muestra la herramienta Duck Toolkit NG, con los comandos necesarios para realizar el ataque, así como el idioma elegido.

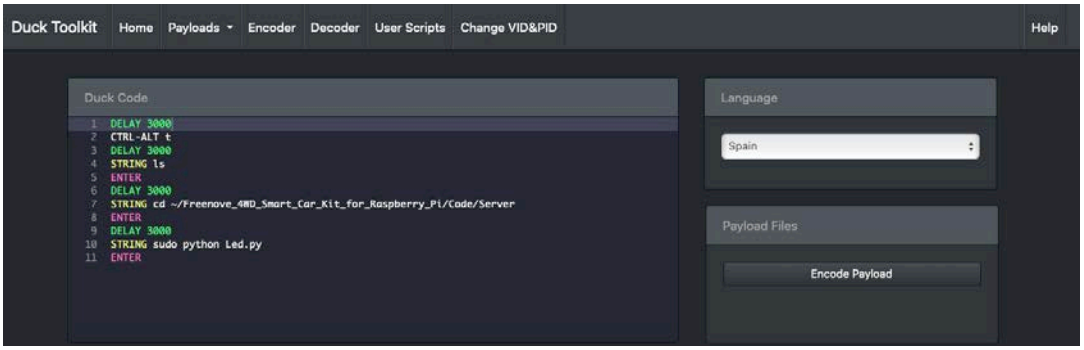

Figura 4 Plataforma Duck Toolkit NG.

La secuencia de comandos que utiliza el script de Rubber Ducky es un formato simple, legible por humanos, por lo tanto, es sencillo de compartir y modificar fácilmente; sin embargo, esto no puede ser procesado por el USB Rubber Ducky, por lo que se debe de crear un archivo “inject.bin” haciendo uso de la herramienta Duck Tookit NG, con el propósito de poder decodificarlo. Alternativamente, existen diversos decodificadores de código abierto para la secuencia de comandos; la empresa creadora de este dispositivo Hak5, recomienda hacer el uso del decodificador oficial llamado JavaScript Ducky, el cual puede decodificar en línea desde cualquier navegador o en su defecto descargando su software.

Las configuraciones adicionales de la herramienta pueden ser descargadas en el enlace: www.downloads.hak5.org/ducky; la secuencia es abrir el archivo en jseconder.html utilizando un navegador compatible con Javascript; posteriormente se codifica el archivo que contiene los comandos en el Ducky Script sobre el área de un texto principal, se hace clic en *“Generate Payload”* y por último se hace clic en *“Download Payload”*. En la figura 5 se muestra un ejemplo sencillo del codificador *“Ducky Encoder”*.

En cuanto a la realización del ataque es necesario conocer los aspectos esenciales en la comunicación entre los dispositivos. Para poder establecer una comunicación del lado del servidor que se encuentra en la Raspberry Pi 4; se requiere una serie de pasos para activar la comunicación como servidor; esto incluye abrir el archivo

ejecutable programado en Python llamado “main.py”, el cual se encuentra en la librería de la documentación descargable y proporcionada por la empresa Freenove para el 4WD Car; una vez que se hace la descarga, el archivo a ejecutar se aloja en la dirección: /Freenove_4WD_Smart_Car_Kit_for_Raspberry_Pi/Code/Server.

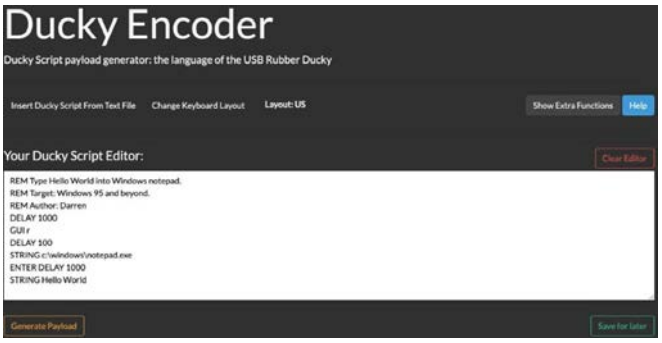

Figura 5 Interfaz del decodificador Ducky Encoder.

Posteriormente ya que se encuentra el archivo “main.py”, este se ejecuta con el siguiente comando “python main.py”.

Al hacer lo anterior, emergerá una ventana que indica que el servidor está encendido (Server on), mostrando un botón de “on” u “off”, esto con la finalidad de apagar o prender el servidor, adicionalmente también se hace visible la dirección IP del servidor, la cual es 172.16.42.115, como se muestra en la figura 6.

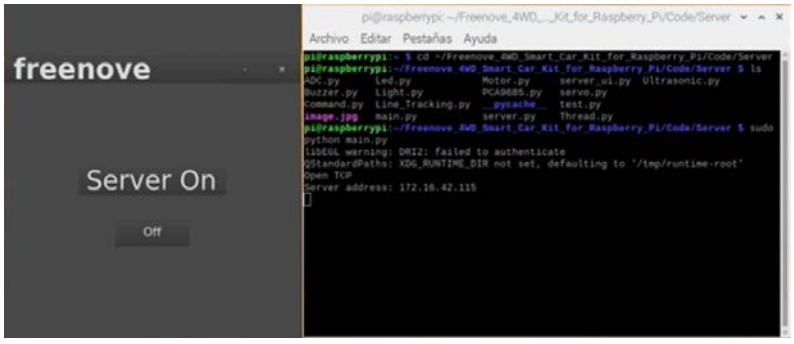

Figura 6 Interfaz del servidor Freenove.

Para los ataques que se realizaron en la Raspberry Pi (servidor), fueron 2 series de comandos o scripts en un archivos con extensión .bin; uno de estos scripts se encarga de apagar el servidor que en este caso es la tarjeta Raspberry Pi, todo esto

a través de una serie de comandos que simulan una HID simulado; mientras que el otro se encarga de alterar el código del programa Main.py, que como ya se mencionó se utiliza para encender el servidor de Freenove, en consecuencia a esto, habrá una modificación en el archivo de ejecución, así como una interrupción o desconexión en el robot móvil, hasta que el archivo vuelva a ser modificado como en su forma inicial; adicionalmente también se puede modificar el código del archivo Led.py, encargado de prender los leds; cabe mencionar que para que esto sea posible, primero es necesario hacer la decodificación de comandos en el programa en línea de Ducky Encoder.

Para establecer la comunicación del lado del cliente, se debe instalar el software de Freenove en una computadora convencional (en este caso laptop Huawei), dicho software puede ser descargado desde la página oficial de Freenove, cabe mencionar que este también puede ser instalado en un dispositivo móvil, ya sea teléfono inteligente o tableta, haciendo uso de una aplicación de Freenove, la cual puede ser descargada en Play Store. Posteriormente, al instalar el software se puede abrir una interfaz como se muestra en la figura 7, en donde se pueden apreciar los distintos controles que permiten manipular el robot móvil (Smart car 4WD), como es el caso de mover las ruedas y la cámara, activar diversos sensores, y prender los leds; no obstante, solo se necesita la IP del servidor que en este caso es 172.16.42.115 (Raspberry), para establecer una comunicación entre estos.

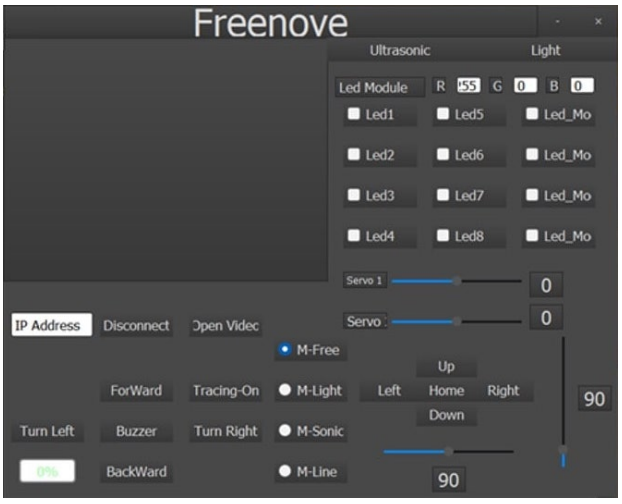

Figura 7 Interfaz de la comunicación cliente de Freenove.

Por lo tanto, para realizar un ataque en la laptop Huawei (cliente); se realiza un ataque de inyección utilizando USB Rubber Ducky, la cual permite cargar una serie de comandos en un archivo .bin, y se realiza el mismo procedimiento de decodificación como fue en el caso de la Raspberry Pi 4.

3. Resultados

Los resultados en los tres escenarios propuestos fueron satisfactorios, en el primer caso se logró apagar la computadora laptop con sistema operativo Windows 11, el script propuesto no tuvo conflicto con la configuración del teclado y la ejecución fue oculta para el usuario. De esta manera, el cliente perdió el enlace de comunicación con el servidor, es decir un ataque indirecto sobre el robot (una negación de servicio).

El segundo caso también fue exitoso en la ejecución del ataque, enfocándose en un script o duckyspcript para apagar el sistema del servidor (Raspberry Pi 4), el script operó de la manera esperada, además la ejecución también se logró de forma oculta para el usuario. En este caso, el robot sufrió un ataque directo al apagar la operación de servicio por completo, sin importar la tarea ejecutada en ese momento.

En el último caso el script o duckyscript logró ejecutar un ataque directo en la operación del robot, es decir, se cambió la configuración interna del robot, tal y como se especificó en el método. El archivo “Main.py” de operación tenía los registros, las variables y las direcciones de memoria asociadas a las instrucciones que manda el cliente y el archivo Led.py que enciende los leds del robot móvil. Por lo que, en las pruebas se comprobó que el archivo Main.py y Led.py fueron alterados, la consecuencia fue un arranque del sistema diferente a lo esperado y que las luces led del robot fueran intercambiadas por colores diferentes.

4. Discusión

La tendencia de utilizar los sistemas embebidos en los robots da la ventaja de optimizar y tener una mayor versatilidad en la operación de estos. Algunas de ellas son la capacidad de no tener cables, programar en lenguajes de alto nivel y crear repositorios de códigos para los robots. Con lo anterior, se tiene la posibilidad de

descargar una rutina o biblioteca para los robots, simplificando así las tareas para estos. La problemática de utilizar herramientas compatibles u homologadas también podría comprometer el sistema a nivel de software. Y con la llegada del internet a los robots se incrementan las vulnerabilidades debido a la escasez de las herramientas de ciberseguridad.

Los ataques físicos y de inyección realizados en los dispositivos objetivos fueron exitosas, para la inhabilitación de los dispositivos y modificación de archivos dentro de estos. Sin embargo, la modificación del programa de ejecución puede ser enfocado a cualquier otro programa dentro de la carpeta de Freenove, ya que está contiene otros programas ejecutables, como es el caso de mover el auto, o prender la cámara y sensores; por ejemplo, en el programa encargado de prender los leds de color rojo, se podría modificar para encender los leds de otro color, así como secuencia de movimiento del robot móvil, entre otras alteraciones del programa.

Adicionalmente, se puede realizar diversas modificaciones en general, ya sea para Linux, MacOS, y Windows; sin embargo, en este último se pueden encontrar más “payloads”, como es el caso de la obtención de credenciales, cambiar fondo de pantallas, o deshabilitar el antivirus de los sistemas operativos.

Para finalizar se menciona que existen diversos repositorios donde se pueden encontrar más *“payloads”*; no obstante, estos últimos utilizan otras herramientas junto con Rubber Ducky, como es en el caso de la investigación de [Cannols, 2017], donde utilizan también las herramientas de Powershell, Mimikatz, un servidor web o PHP; lo cual permitió explotar más las vulnerabilidades de sistemas operativos del sistema operativo Windows y robar la información como el id y la contraseña. Por otro lado, los investigadores en [Harianto, 2019] realizaron ataque usando Rubber Ducky; sin embargo, su tasa de éxito fue de 94,28%, y debido a la alta tasa de reutilización de contraseñas que alcanza el 81%, por lo que al final se tiene una alta probabilidad de éxito; adicionalmente realizaron un ataque de fuerza bruta, en conjunto con el Rubber Ducky; por otro lado, también modificaron el firmware de dicha herramienta, guardando archivos localmente dentro de su memoria interna.

A pesar de que se implementaron acciones distintas utilizando Rubber Ducky, así como diversas herramientas en conjunto, o métodos, se tiene en cuenta que solo

se necesitan unos segundos para robar o alterar de manera discreta y confidencial la información de un equipo, ya sea en distintos sistemas operativos siendo el caso de las demás investigaciones diversas versiones de Windows, y en esta investigación la implementación con Raspian OS. Una comparativa en la evaluación de los puertos se muestra en la tabla 2.

Tabla 2 Evaluación de riegos de los puertos.

Evaluación/ Vulnerabilidad	**Riesgo**	**Método de reducción de riesgo**
#1 Puertos externos no inhabilitados	Alterar el suministro de energía la computadora de lado del cliente en el sistema operativo Windows	Habilitar el bloqueo de pantalla ante cualquier introducción de un dispositivo USB. esto con la herramienta Pentrack Disguised-keyboard Detector.
#2 Puertos externos no inhabilitados	Alterar el suministro de energía de la tarjeta Raspberry Pi 4, del sistema operativo Raspbian OS	Resguardo del dispositivo Raspberry, inhabilitación de los cuatro puertos.
#3 Puertos externos no inhabilitados	Modificar los scripts de ejecución en la Raspberry Pi	Editar las listas (blancas y negras) de los dispositivos autorizados.

5. Conclusiones

La comunicación entre distintos dispositivos cada vez es mayor, en este caso se puede interactuar con un robot móvil a distancia, esto toma como referencia el internet de las cosas robóticas, ya que se encarga de obtener información a través de sensores y la comunicación a distancia con estos dispositivos robóticos. Sin embargo, en estas tecnologías surgen dificultades tales como las vulnerabilidades de ciberseguridad; es decir, los ataques que se realizan para extraer información, o modificar el comportamiento de estos dispositivos.

En esta investigación justamente se muestra un ataque cibernético, sencillo, pero a la vez con un potencial enorme, haciendo uso de la herramienta USB Rubber Ducky, y comandos de secuencias, simulando un teclado HID; por otro lado, se conoció la vulnerabilidad que puede tener Windows o en su defecto cualquier otro sistema operativo, con la imposibilidad de ser detectado como una amenaza.

Por último, se menciona que los ataques físicos y de inyección, suelen ser muy comunes; sin embargo, proteger estos dispositivos que pudieran ser vulnerados,

puede ser más sencillo, si se tiene la atención adecuada; uno de estos es la vigilancia humana a dichos dispositivos, o en su defecto pueden ser resguardados de manera segura en un lugar físico; por otro lado, también puede restringir los accesos a los puertos USB, o configurar los dispositivos únicamente de confianza. Con estas atenciones a los dispositivos se podría evitar un ataque directo y físico; tomando en cuenta que dichos dispositivos controlan a distintos tipos de robots que tienen interacción en el mundo real, puede repercutir de manera grave en distintos escenarios de la sociedad; provocando desastres o incluso perdidas ya sea materiales o humanas. De este punto la importancia de tener una cultura en la ciberseguridad en especial de IoRT.

En estos casos, se demuestra que la arquitectura en la comunicación entre los dispositivos se puede verificar a nivel software, incluso se puede utilizar una VPN, para asegurar la comunicación entre cliente-servidor; sin embargo, la seguridad en la comunicación del lado del hardware muchas veces no puede ser verificada, por lo tanto, esto se considera un riesgo importante en la integridad del robot.

6. Bibliografía y Referencias

[1] Al-Zarouni, M. The reality of risks from consented use of USB devices. 4th Australian Information Security Management Conference, Edith Cowan University, Perth, Western Australia, 5th December, 2006.

[2] Andrea, I, C Chrysostomou and G Hadjichristofi. Internet of Things: Security vulnerabilities and challenges. In: 2015 IEEE Symposium on Computers and Communication (ISCC) [online]. 180–187. 2015. Available at: doi: https://doi.org/10.1109/ISCC.2015.7405513.

[3] Arora, L., Thakur, N., & Yadav, S. K. (2021, February). USB Rubber Ducky Detection by using Heuristic Rules. In 2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS). IEEE. 156-160, 2021.

[4] Basheer, M. and A. Varol. An Overview of Robot Operating System Forensics. In: 1st International Informatics and Software Engineering Conference: Innovative Technologies for Digital Transformation, IISEC 2019 - Proceedings

[online]. 2019. ISBN 9781728139920. Available at: doi: https://doi.org/10.1109/UBMYK48245.2019.8965649.

[5] Bhakte, R., Zavarsky, P., & Butakov, S. (2016, June). Security Controls for Monitored Use of USB Devices Based on the NIST Risk Management Framework. In 2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC). IEEE. (2), 461-466, 2016.

[6] Cannoles, B., & Ghafarian, A. Hacking experiment by using usb rubber ducky scripting. Journal of Systemics, 15(2), 6671, 2017.

[7] Ducktoolkit. https://www.ducktoolkit.com/, (Consultada el 24 de noviembre de 2022).

[8] Eswaran, K., M. S. S. Kumar, D. Thangavelusamy and V. Murugadoss. Smart Servomotor for Robotics and its Cyber security. In: Proceedings- 2020 Advanced Computing and Communication Technologies for High Performance Applications, ACCTHPA 2020 [online]. 231–236, 2020. ISBN 9781728164533. Available at: doi: https://doi.org/10.1109/ACCTHPA49271.2020.9213226.

[9] Flores-Montaño, L. A., Herrera-Lozada, J. C., Sandoval-Gutiérrez, J., Vázquez-López, R., & Martínez-Vázquez, D. L. Ciberseguridad del internet de las cosas robóticas: plataforma experimental. DYNA-Ingeniería e Industria, 96(5), 2021.

[10] Griscioli, F., Pizzonia, M., & Sacchetti, M. USBCheckIn: Preventing BadUSB attacks by forcing human-device interaction. In 2016 14th Annual Conference on Privacy, Security and Trust (PST). IEEE. 493-496, 2016.

[11] Falliere, N., Murchu, L. O., & Chien, E. W32. stuxnet dossier. White paper, symantec corp., security response, 5(6), 29, 2011.

[12] Harianto, H. E., & Gunawan, D. Wi-Fi password stealing program using USB rubber ducky. TELKOMNIKA (Telecommunication Computing Electronics and Control), 17(2), 745-752, 2019.

[13] Mourtzis, D., K. Angelopoulos and V. Zogopoulos. Mapping vulnerabilities in the industrial internet of things landscape. In: Procedia CIRP [online]. 265–270, 2019. Available at: doi: https://doi.org/10.1016/j.procir.2019.04.201.

[14] Jamai, I., L. Ben Azzouz and L.A. Saidane. Security issues in Industry 4.0. In: 2020 International Wireless Communications and Mobile Computing, IWCMC 2020 [online]. 481–488, 2020. ISBN 9781728131290. Available at: doi: https://doi.org/10.1109/IWCMC48107.2020.9148447.

[15] Li, X., You, J., Zhou, X., Chen, R., & Tang, C. (2018, May). HID Architecture and Design in Embedded USB Host System. In 2018 International Conference on Network, Communication, Computer Engineering (NCCE 2018). Atlantis Press. 465-469, 2018.

[16] Papp, D., Ma, Z., & Buttyan, L. Embedded systems security: Threats, vulnerabilities, and attack taxonomy. In 2015 13th Annual Conference on Privacy, Security and Trust (PST), IEEE.145-152, 2015.

[17] Stankovic, J. A. Real-time and embedded systems. ACM Computing Surveys (CSUR). 28(1), 205-208, 1996.

[18] Tian, D. J., Bates, A., & Butler, K. Defending against malicious USB firmware with GoodUSB. In Proceedings of the 31st Annual Computer Security Applications Conference. 261-270, 2015.

[19] Tyutyunnik, A., & Lazarev, A. Intelligent System for Preventing Rubber Ducky Attacks Using Deep Learning Neural Networks. In 2021 International Russian Automation Conference RusAutoCon, IEEE. 497-502, 2021.

[20] Varga, P., S. Plosz, G. Soos and C. Hegedus. Security threats and issues in automation IoT. In: IEEE International Workshop on Factory Communication Systems - Proceedings, WFCS [online]. 2017. ISBN 9781509057887. Available at: doi: https://doi.org/10.1109/WFCS.2017.7991968.

[21] Vouteva, S., Verbij, R., & Roos, J. Feasibility and Deployment of Bad USB. University of Amsterdam, System and Network Engineering Master Research Project. 2015.

[22] Wolfang. https://www.w0lff4ng.org/usb-rubber-ducky-101/, (Consultada el 24 de noviembre de 2022).

[23] Zhao, S., & Wang, X. A. A Survey of Malicious HID Devices. In International Conference on Broadband and Wireless Computing, Communication and Applications, Springer, Cham. 777-786, 2019.