UNIVERSIDAD POLITECNICA DE VALENCIA

ESCUELA POLITECNICA SUPERIOR DE GANDIA

Grado en Ing. Sist. de Telecom., Sonido e Imagen

UNIVERSIDAD
POLITECNICA
DE VALENCIA

ESCUELA POLITECNICA
SUPERIOR DE GANDIA

"Detección y prevención de ataques basados en vulnerabilidades de los ficheros DICOM contra sistemas de gestión médicos."

TRABAJO FINAL DE GRADO

Autor/a:
Arturo Navarro Quijada

Tutor/a:
Jordi Bataller Mascarell

GANDIA, 2020

RESUMEN

Los ciberataques han aumentado en los últimos años, especialmente los ataques contra las infraestructuras críticas, incluyendo los entornos clínicos. Estos ataques son llevados a cabo por actores hostiles como cibercriminales o grupos APT, con diferentes objetivos.

Se sabe que estos ataques se están produciendo. Nuestro objetivo es averiguar cómo, y qué medidas podemos tomar para protegernos.

Construiremos una herramienta para infectar radiografías con software malicioso real, empleando una vulnerabilidad única y característica de estos entornos: una vulnerabilidad en la estructura de los ficheros de imágenes médicas (DICOM) que permite introducir *malware* en imágenes médicas e infectar así toda la infraestructura.

Con estas muestras atacaremos un entorno virtualizado, en el que estudiaremos cómo se puede detectar empleando las técnicas actuales de detección y monitorización de equipos, y diseñaremos una solución para desinfectar las radiografías maliciosas.

ABSTRACT

Cyber-attacks have increased in recent years, especially attacks on critical infrastructures, including hospitals. These attacks are made by hostile actors such as cybercriminals or APT groups with different objectives.

We know we are under attack. Our goal is to find out how, and how can we protect ourselves.

We will build a tool to infect radiographs with real malicious software, using a unique vulnerability characteristic of these environments: a vulnerability in the structure of medical image files (DICOM) that allows malware to be injected into medical images and thus infect the entire infrastructure.

With these samples we will attack a virtualized environment, were we will study how it can be detected using current tools, detection and monitoring techniques, and we will design a solution to disinfect malicious radiographys.

CONTENIDO

1 INTRODUCCIÓN

En este apartado comentaremos la organización de esta memoria.

En la sección 1.1 realizaremos una presentación general, declarando los objetivos que trataremos de alcanzar en este proyecto.

En la sección 1.2 realizaremos una breve introducción al resto de capítulos y apartados, dando una visión de conjunto de todo el contenido de la memoria.

1.1 PRESENTACIÓN GENERAL Y OBJETIVOS

Los ciberataques han aumentado en los últimos años. Tanto es así que algunas compañías aseguradoras han pasado a considerarlos uno de los mayores riesgos que puede sufrir una empresa.

Merecen especial consideración los ataques contra las infraestructuras críticas, por el impacto directo que pueden tener sobre la vida de las personas. Dentro de esta categoría se encuentran las instalaciones clínicas, cuyo sector ha sido víctima de ataques muy importantes en los últimos años. Estos ataques son llevados a cabo por actores hostiles como cibercriminales o (grupos APT) con diferentes objetivos.

Estas infraestructuras clínicas poseen unas condiciones que las hacen tentadoras para los atacantes, entre las que destacamos la convergencia entre los entornos IT y OT. En este proyecto se estudiará cómo detectar ataques a entornos clínicos empleando una vulnerabilidad única y característica de estos entornos.: una vulnerabilidad en la estructura de los ficheros de imágenes médicas (DICOM) que permite introducir *malware* en imágenes médicas e infectar así toda la infraestructura. Dicha vulnerabilidad se hizo pública en abril de 2019, junto a dos pruebas de concepto que mostraban cómo podían construirse ficheros políglotas PE-DICOM.

Basándonos en esta problemática, el principal objetivo de este proyecto es la detección y prevención de este tipo de amenazas. Construiremos una herramienta para infectar radiografías con software malicioso real, que atacarán un entorno virtualizado, en el que estudiaremos el rastro forense que deja este tipo de ataques y cómo se puede detectar empleando las técnicas actuales de detección y monitorización de equipos. Explicaremos la citada vulnerabilidad en detalle y las ampliaciones que hemos desarrollado para sofisticar estos ataques y reducir la detección por parte de los sistemas de *anti-virus*; proponiendo medidas de mitigación en base a los resultados obtenidos y diseñaremos una solución para desinfectar las radiografías maliciosas.

Para alcanzar estos objetivos, haremos un repaso de los conceptos técnicos y la terminología relacionada, así como un estudio detallado del funcionamiento de la vulnerabilidad y de la herramienta desarrollada, y las distintas técnicas de detección. Además, se contemplarán una serie de ataques avanzados para planificar una hipotética detección en un entorno virtualizado, divididos por etapas para facilitar su estudio. En cada etapa se probarán medidas de detección utilizado herramientas de código abierto, y ofreciendo configuraciones y reglas adecuadas para detectar estos ataques.

1.2 CONTENIDO DE LA MEMORIA

Esta memoria está estructurada de la siguiente forma. En el capítulo 2, *Estado del arte*, se expone el marco contextual en el que se ubica este trabajo, y se introducen los conceptos clave que serán necesarios para comprender este proyecto. También se describen los artículos de investigación relacionados con ciberataques a entornos sanitarios, comentando brevemente su influencia en esta investigación.

El capítulo 3, *Análisis de requerimientos*, describe las necesidades del proyecto: qué vamos a aportar, qué limitaciones esperamos encontrar y qué herramientas emplearemos para nuestros objetivos. Específicamente, nos centraremos en los requisitos para construir la herramienta, bautizada *Bad-Dicom*, que genere muestras maliciosas para poder estudiarlas, así como en las herramientas que emplearemos para detectar dichas muestras.

El capítulo 4, *Diseño de ataques a infraestructuras clínicas*, detalla el diseño de la herramienta *Bad-Dicom*, sometido los requerimientos del apartado anterior, describiendo qué debe hacer y cómo. Se describen los ataques que los supuestos atacantes llevarían a cabo, y cómo se debe emplear la herramienta. También se diseña el entorno de pruebas.

En el capítulo 5, *Evaluación de resultados*, dispondremos escenarios para las muestras generadas con la herramienta, mostrando una prueba de concepto para cada variedad antes descrita.

El capítulo 6, *Contramedidas*, propone una serie de recomendaciones para contrarrestar los ataques, repartidas en tres secciones. La sección 6.1: *Prevención*, propone medidas para evitar ser atacados, empleando un script para desinfectar una radiografía posiblemente infectada. La sección 6.1: *Detección*, describe cómo detectar que estamos siendo atacados, empleando herramientas de código abierto, y finalmente la sección 6.3: *Mitigación*, propone medidas para evitar daño una vez seamos atacados.

Finalmente, el capítulo 7 expone las *conclusiones*, realizando un breve recorrido por todo el proyecto y recopilando los hitos del mismo. También se proponen aplicaciones del proyecto y mejoras de cara a posibles ampliaciones en el futuro.

2 ESTADO DEL ARTE

En esta sección se introducen conceptos que serán necesarios para entender el trabajo, y se exponen las investigaciones publicadas relacionadas con la temática, explicando qué influencia han tenido en este proyecto.

2.1 CONTEXTO DEL PROYECTO

Este apartado describe la situación actual en los entornos clínicos desde una perspectiva técnica, incidiendo en las características únicas de este ecosistema.

2.1.1 Los ficheros de imágenes DICOM en el sector sanitario

En Medicina, se hace uso de imágenes médicas con el objetivo de examinar, analizar, observar y diagnosticar el interior de un paciente. Estas imágenes siguen el protocolo DICOM (ISO 12052:2017 s.f.).

> *DICOM® (*Digital Imaging and Communications in Medicine*) define los formatos para imágenes médicas que pueden intercambiarse con los datos y la calidad necesarios para el uso clínico (NEMA (National Electrical Manufacturers Association) s.f.).*

Estas imágenes se generan mediante escáneres, siendo los dos tipos más comunes las *Imágenes por Resonancia Magnética* (IRM/MRI, Magnetic Resonance *Imaging*) y la *Tomografía Axial Computarizada* (TAC/CT, *Computed Tomography*). Así, resultan en una imagen tridimensional del paciente mediante la agrupación de imágenes bidimensionales del cuerpo sobre el plano axial (de arriba abajo exceptuando el cráneo), sagital (lateral a la línea media) o coronal (de atrás a adelante) a lo largo del cuerpo. Las MRI emplean campos magnéticos, mientras que la tecnología TAC emplea rayos-X; ambas se emplean solas o en combinación para el diagnóstico y seguimiento de diversas patologías, tanto de tejidos rígidos como hueso como de tejidos blandos.

Estas imágenes se centralizan a través del PACS (*Picture Archiving and Communication System*), un sistema de red basado en Ethernet y concentrado en un servidor central, que recibe los escaneos de los diferentes dispositivos, almacenándolos en una base de datos y permitiendo a los radiólogos su almacenamiento.

2.1.2 Vulnerabilidades de los entornos clínicos

El entorno clínico es complejo. Se trata de un ecosistema tradicionalmente OT (*Operational Technologies*), en el que la funcionalidad y la accesibilidad de los datos prima sobre la integridad de los mismos, y que lentamente se ha incorporado al mundo IT (*Information Technologies*), lo que dificulta la implantación de los estándares modernos de Seguridad de la Información.

Esto se refleja no sólo en las peculiaridades técnicas del entorno sino en la filosofía de las empresas del sector que, centrándose en las prioridades OT, han descuidado la Seguridad de sus sistemas. Encontramos así un sistema centralizado y poco o nada segmentado (Seals 2019), pensado para la interconexión de un gran número de máquinas con uno o varios servidores. Incluso los PACS que no estén conectados directamente a internet pueden ser vulnerables al estar indirectamente conectados a través de la red interna de la organización (H.K 2019).

Por ejemplo, haciendo una rápida búsqueda en *Shodan* descubrimos 2673 servicios DICOM (Shodan, resultados de "DICOM" s.f.) y 618 servidores PACS (Shodan, resultados de "PACS" s.f.) expuestos a internet, entre ellos varios RDP mal bastionados. Muchos de estos servicios carecen de los últimos parches de seguridad y son vulnerables a varios CVE (*Common Vulnerabilities and Exposures*). En 2018, Investigadores de McAfee demostraron cómo se puede explotar estos servicios expuestos y acceder al almacén de imágenes, llegando incluso a poder alterarlas (Beek 2018). Otras amenazas a las que también son vulnerables comprenden ataques de ingeniería social (Efe 2020), intrusiones físicas o *insiders*.

Los ficheros de imágenes DICOM también son vulnerables. En abril de 2019, *Cylera Labs* publicó un *whitepaper* donde demostraba cómo podían crearse ficheros DICOM políglotas PE-DICOM, lo que resultaba en un DICOM infectado que es al mismo tiempo un archivo ejecutable y un fichero DICOM funcional (Picado, Attacking Digital Imaging and Communication in Medicine (DICOM) file format standard 2019). Esta vulnerabilidad se recoge en el CVE-2019-11687 (CveDetails 2019).

Al formar parte del mundo OT, los entornos clínicos pueden sufrir ataques del ámbito ICS (*Industrial Control System*), dirigidos contra los propios dispositivos médicos (Mirsky 2019).

Además, las clínicas y hospitales tratan con datos de carácter personal y sensible, lo que les hace especialmente vulnerables a las brechas de datos.

Por todo lo anterior, sin comentar los grandes presupuestos que manejan, son un objetivo muy jugoso para los atacantes (George 2018), lo que ha producido un aumento en los ciberataques contra este sector.

2.1.3 Amenazas al sector Salud

Los ciberataques contra estructuras clínicas están a la orden del día. Así lo advierte la INTERPOL (INTERPOL 2020) y el equipo de Inteligencia de Amenazas de Microsoft (Microsoft Threat Protection Intelligence Team. 2020), acentuado todavía más si cabe por la pandemia del COVID-19 y el papel fundamental que estos organismos tienen en la misma. Entre otras amenazas, advierten contra ataques de *ransomware* y brechas de datos (Healthcare IT News 2019).

Los actores detrás de estos ataques pueden tener diferentes motivaciones: provocar el caos, debilitar a un Estado (STOLTON 2020), ciberterrorismo o beneficio económico. De estos actores cabe destacar las bandas del crimen organizado y los grupos APT (*Advanced Persistent Threat).*

Las bandas de crimen organizado buscan normalmente obtener dinero. Esto suele hacerse de dos formas: mediante una exfiltración de información y posterior chantaje, o desplegando un *ransomware* en toda la organización que inutilice los sistemas informáticos, paralizando la actividad de la organización hasta que paguen un rescate, normalmente en criptomonedas. Algunas variantes de *ransomware* ofrecen descuentos si se paga pronto, e incluso amenazan con publicar los datos robados si no se paga el

rescate (Abrams, New Nefilim ransomware threatens to release victims data 2020). En general, no buscan un objetivo concreto siempre que puedan obtener beneficio.

Ejemplos de estas bandas criminales son *Sodinokibi*, *Dopplepaymer*, *Dridex*, *Ryuk* (EnigmaSoftware s.f.) (NCSC (GCHQ). 2019), *Maze*, *Netwalker* y *TA505/CLOP* (Abrams, TA505 hackers behind Maastrich University ransomware attack 2020).

Algunas bandas criminales muestra cierto grado de ética. Los *CLOP* y los *Dopplepaymer* declaran que evitan atacar hospitales y ONG, aunque sí al sector farmacéutico, y que en caso de tener evidencias de que ese laboratorio está trabajando en la vacuna le entregarían la clave de descifrado gratuitamente. Por otro lado, los *Maze* manifestaron que dejarían de atacar al sector clínico mientras durase la pandemia (Abrams, Ransomware Gangs to Stop Attacking Health Orgs During Pandemic 2020), y aunque a finales de marzo atacaran de nuevo, alegaron que la infección se llevó a cabo antes del manifiesto (Goodwin 2020). *Nefilim* y *Netwalker* también los evitan, aunque los últimos se negarían a descifrar gratuitamente la información.

Otros, como los *Ryuk*, no detienen su actividad (Abrams, Ryuk ransomware keeps targeting hospitals during the pandemic 2020). Los *Ryuk* tienen un notable historial de ataques en España (Ayuntamiento de Jerez, Cadena SER, Prosegur y Hospital de Torrejón (S. González 2020) (Raya 2020)).

Cabe destacar que estos ataques pueden producirse tanto si el PACS se aloja localmente como si se almacena en la nube (Miliard 2019).

Otro tipo de actores detrás de ataques al sector sanitario son los grupos APT (*Advanced Persistent Threat*) (Kaspersky s.f.). Se trata de grupos pertenecientes a un Estado o financiados por el mismo, generalmente forman parte de un ejército o cuerpo de inteligencia. Al contrario que los cibercriminales, estos grupos no necesariamente buscan beneficio económico, sino que suelen perseguir objetivos de alto nivel, como espionaje (político, industrial), estudio de las capacidades enemigas y ciberguerra. Estos grupos se caracterizan por el uso de técnicas avanzadas (llegando incluso a emplear uno o varios *zero days*), campañas muy largas y la capacidad de pasar desapercibidos durante mucho tiempo (incluso años) dentro de las organizaciones objetivo. Debido a la sofisticación de sus técnicas, resulta muy difícil descubrir sus actividades.

> *An adversary that possesses sophisticated levels of expertise and significant resources which allow it to create opportunities to achieve its objectives using multiple attack vectors (NIST SP800-61)*
>
> *Scope Note: The APT:*
>
> - *Pursues its objectives repeatedly over an extended period of time*
> - *Adapts to defenders efforts to resist it*
> - *Is determined to maintain the level of interaction needed to execute its objectives (ISACA. 2014).*

Uno de los grupos APT presuntamente vinculado a *Elfin* (Irán) (Rincón 2019) que atacan al sector de la Salud es OrangeWorm (Symantec 2018). Activo desde 2015, fue

descubierto por Symantec en 2018, y se sabe que ataca a proveedores de equipamiento y material, farmaceúticas y hospitales de Estados Unidos, Europa y Asia para espionaje corporativo. Así, para alcanzar un objetivo es capaz de atacar a la cadena de suministro del mismo. Symantec afirma que los objetivos son cuidadosamente escogidos.

OrangeWorm emplea el troyano *Kwampirs*, un *backdoor* que se instala en las máquinas de rayos-x y MRI. Orangeworm también infecta los equipos dedicados a atender a los pacientes para rellenar consentimientos para procedimientos clínicos. Kwampirs.exe desencripta y extrae una copia de su *payload* (un RAT (Lab52 s.f.) en forma de DLL) contenida en su sección *resources*, y antes de escribir en disco inserta una cadena aleatoria en medio del payload para evadir detecciones basadas en hash. Para persistir, crea un servicio, "*WmiApSrvEx*" que invoca a rundll32.exe para llamar nuevamente a su DLL maliciosa cuando el equipo se reinicie. *Kwampirs* recopila información básica sobre la víctima, incluyendo información de red, sistema e idioma. Una peculiaridad de este *malware* es que no intenta ser sigiloso, propagándose de manera agresiva a través de los recursos compartidos de red.

Una de las peculiaridades del ciberespacio es que la atribución de los ataques es compleja, con una alta probabilidad de salir indemne, por lo que estos ataques son cada vez más frecuentes, desde criminales a Estados (Stolton 2020).

2.2 CONCEPTOS CLAVE

Vamos a explicar en profundidad algunos conceptos que serán necesarios para entender los siguientes apartados. Se profundizará en los tres tipos de ficheros que intervendrán en la composición de los *Bad-DICOM*: PE, DICOM y scripts.

2.2.1 Estructura de los ficheros PE

Los ficheros PE (*Portable Executable*) son ficheros ejecutables del sistema operativo Windows. Se trata de una estructura de datos compuesta por secciones, que encapsula la información necesaria para que el *loader* de Windows interprete el código ejecutable contenido en ella (Microsoft s.f.). Algunos tipos de PE son EXE, DLL y SYS.

PE está basado en el formato de los COFF (*Common Object File Format*) ejecutables del sistema operativo MS-DOS, con el que mantiene retrocompatibilidad conservando la cabecera de los antiguos ficheros ejecutables MZ (Kowalczyk s.f.).

La estructura de los PE se divide en una cabecera y un conjunto de secciones.

La cabecera está compuesta a su vez de varias partes:

Cabecera MS-DOS: La cabecera MS-DOS está formada por un MZ-HEADER (64 bytes) y un MZ-STUB (longitud variable). Esto permite mostrar el mensaje "This program cannot be run in DOS mode" cuando el PE se ejecuta en modo MS-DOS. La cabecera MS-DOS comienza por la firma MZ (4D 5A), y contiene además, en el byte 0x3c llamado *e_lfanew*,

la dirección de memoria donde se encuentra la firma PE (0x50450000), codificada en *little endian*.

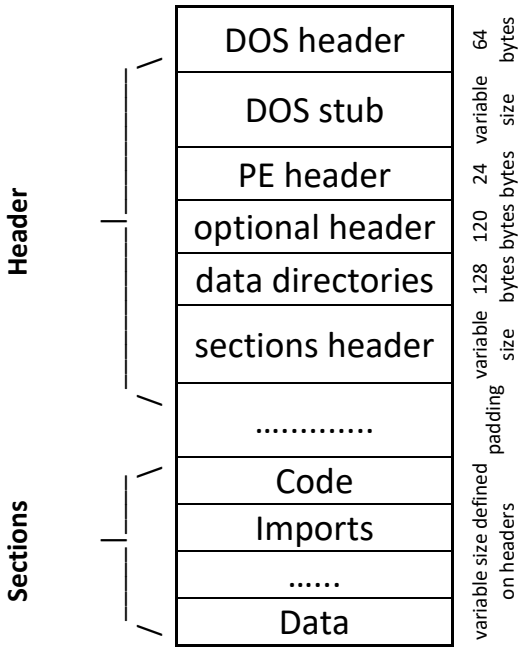

Gráfico 1: Formato PE

El DOS-STUB es completamente opcional y puede ser eliminado (Corkami s.f.), perdiendo retrocompatibilidad con MS-DOS. En Windows, el sistema leerá hasta la dirección 0x3c, interpretando la dirección de la firma PE e ignorando el DOS-STUB.

Firma PE: La firma PE consiste en cuatro bytes (50 45 00 00) que corresponden con PE\0\0 (P + E + cero + cero). Esta firma identifica al binario como un PE.

Cabecera COFF: Nada de interés para este proyecto.

Cabecera opcional: contiene los campos *SizeOfHeaders*, *SectionAlignment* y el *FileAlignment*. SizeOfHeaders determina cuánto mide la cabecera, y por tanto dónde comenzarán las secciones. *FileAlignment* se utiliza para alinear las secciones, determinando el tamaño del *padding* o relleno, y puede medir entre 512 y 64k. Si el valor de *SectionAlignment* es menor que el tamaño de la página del mapeado en memoria para esa arquitectura, FileAlignment debe ser igual al valor de SectionAlignment.

Directorio de Datos: Nada de interés para este proyecto.

Cabecera de las secciones: Las cabeceras de las secciones están agrupadas en la *Section Table*. Cada cabecera de sección contiene el campo *PointerToRawData*, que es un puntero a la primera página de la sección dentro del archivo COFF. Debe ser un múltiplo del *FileAlignment* de la cabecera opcional, y alinearse en un límite de 4 bytes para un mejor rendimiento en memoria.

2.2.2 Estructura de los ficheros DICOM

DICOM es un estándar tanto de un protocolo de red como de una estructura de ficheros. En esta sección se describe la estructura de los ficheros.

Los ficheros DICOM (NEMA (National Electrical Manufacturers Association) s.f.) contienen imágenes tridimensionales de un paciente, y son un estándar de 1993, diseñado por la NEMA y reconocido como estándar de imágenes médicas por la ISO 12052:2017. La estructura de los DICOM se divide en dos partes: Una cabecera de 132 bytes y un *Dataset*.

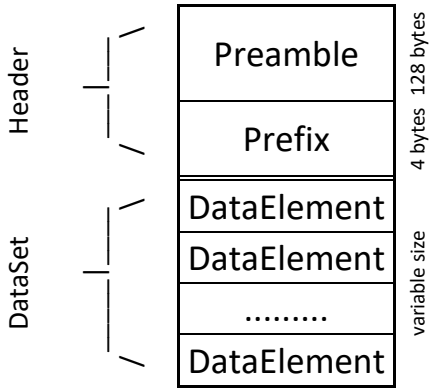

Gráfico 2: Formato DICOM

La cabecera se compone de un *preamble* de los primeros 128 bytes, seguido un *Prefix* que contiene la firma DICM (44 49 43 4D) en la posición 0x80. El *preamble* es empleado por algunos programas para almacenar metadatos, mientras que la firma DICM identifica al fichero como un archivo DICOM.

El *Dataset* está compuesto por varios *DataElement*, que a su vez se componen de varios campos. Los *DataElements* pueden ser de tres tipos:

Type 1	TAG	VR	Value len		Value
Type 2	TAG	VR	-	Value len	Value
Type 3	TAG	Value len		Value	

Gráfico 3: Tipos de DataElement en los ficheros DICOM

Tag: identifica el atributo del *DataElement*, representado de la forma (XXXX, XXXX) en hexadecimal.

VR: *Value Representation*, describe el tipo de datos y formato del atributo valor.

Value len: longitud del campo *Value*.

Value: datos relacionados con el DataElement, de longitud *ValueLen*.

Los primeros *DataElements* de un DICOM están relacionados con Metadatos, empleando el Tag (0002, XXXX). Hay un tipo de Tag que es ignorado por la mayoría de *parseadores* e intérpretes de DICOM, llamados *private tags*, y expresados de la forma (0009, XXXX).

2.2.3 Shell scripting: Powershell, batch, bash

Powershell, batch y bash son tres procesadores de comandos de los sistemas operativos Windows y Unix. Generalmente se trata de una ventana de texto donde el usuario escribe comandos, que se interpretarán como acciones. Los procesadores de comandos pueden interpretar ficheros de script, que son texto sin formato que contienen secuencias de comandos, guardados con una extensión característica.

Bash (*Bourne Agains SHell*) es el procesador de comandos nativo en los sistemas *Unix* (FreeSoftware Foundation s.f.), que comprenden *Linux* (*Ubuntu*, *Debian*, *Mint*, *Arch*) y *MacOS*, así como las últimas versiones de Windows. *Bash* es capaz de interpretar los ficheros de scripting .sh y está basado en *Bourne shell* (sh). Otros intérpretes son *Almquist shell* (ash) y *Z shell* (predeterminada en *MacOS* desde *Catalina* (G. González s.f.)).

Batch es el lenguaje de scripting para la consola *COMMAND.COM* (Microsoft 2018) de *DOS*, *Windows 95*, *Windows 98* y *Windows ME*. Se trata de un intérprete que se mantiene en todas las versiones de Windows por compatibilidad, a pesar de datar de 1980 y funcionar bajo arquitectura x86. Es capaz de interpretar los ficheros por lotes .bat

PowerShell (Microsoft s.f.) es un intérprete de comandos diseñado para administradores de sistemas en máquinas Windows. Lanzado en 2006, está programado en C# y basado en objetos .NET. Funciona para arquitecturas x86 y x64, y se incluye en el sistema por defecto desde Windows 7, aunque es compatible con todas las versiones desde Windows XP. *Powershell* permite acceso completo a COM (*Component Object Model*) y WMI (*Windows Management Instrumentation*). PowerShell interpreta ficheros de script con extensión .ps1 y .psm1, y está disponible en las últimas versiones de Linux y MacOS.

Powershell ha sido muy utilizado por los atacantes durante los últimos años, y es especialmente popular para la fase de Post-Explotación, habiendo adaptado herramientas (como *Mimikatz* (Gentilkiwi 2016), que realiza un volcado de memoria del proceso *lsass.exe* para obtener credenciales almacenadas en memoria) y diseñado *frameworks* como *Empire* (BC-Security 2020) o *iBombShell* (ElevenPaths 2020) para comprometer organizaciones con relativa facilidad.

2.3 Publicaciones actuales sobre ataques mediante DICOM (Related work)

Respecto a ataques a entornos médicos empleando DICOM destacamos tres estudios. Uno trata de alterar las imágenes de los DICOM (Mirsky 2019), atacando a la integridad de los mismos, para provocar errores en el diagnóstico de enfermedades, otro explica el CVE-2019-11687 (Picado, Attacking Digital Imaging and Communication in Medicine (DICOM) file format standard 2019) y un tercero advierte que los DICOM pueden ser troyanizados (Zaw 2017). Éste último estudio ha motivado el desarrollo de este proyecto.

2.3.1 Primer estudio: alteración maliciosa de DICOM

El primer estudio, "*CT-GAN: Malicious Tampering of 3D Medical Imagery using Deep Learning*", fue realizado por Yisroel Mirsky, Yuval Elovici y Tom Mahler, de la Universidad de Ben-Gurion (Israel), y por Ilan Shelef, del Centro Médico de la Universidad de Soroka, Israel.

El estudio pone de manifiesto las vulnerabilidades del sector clínico, explica la topología de la red desde la perspectiva de un atacante y muestra cómo los DICOM son susceptibles de sufrir modificaciones, incluso siendo interceptados en el camino desde la máquina de MRI/CT hasta el PACS o el equipo del sanitario. Demostraron cómo podían, mediante algoritmos GAN, diseñar una IA capaz de eliminar o generar tumores en una radiografía.

2.3.2 Segundo estudio: Ficheros híbridos PE-DICOM

El segundo estudio, "*Attacking Digital Imaging and Communication in Medicine (DICOM) file format standard*", fue presentado por Markel Picado, y explica cómo pueden crearse ficheros políglotas PE-DICOM explotando el CVE-2019-11687.

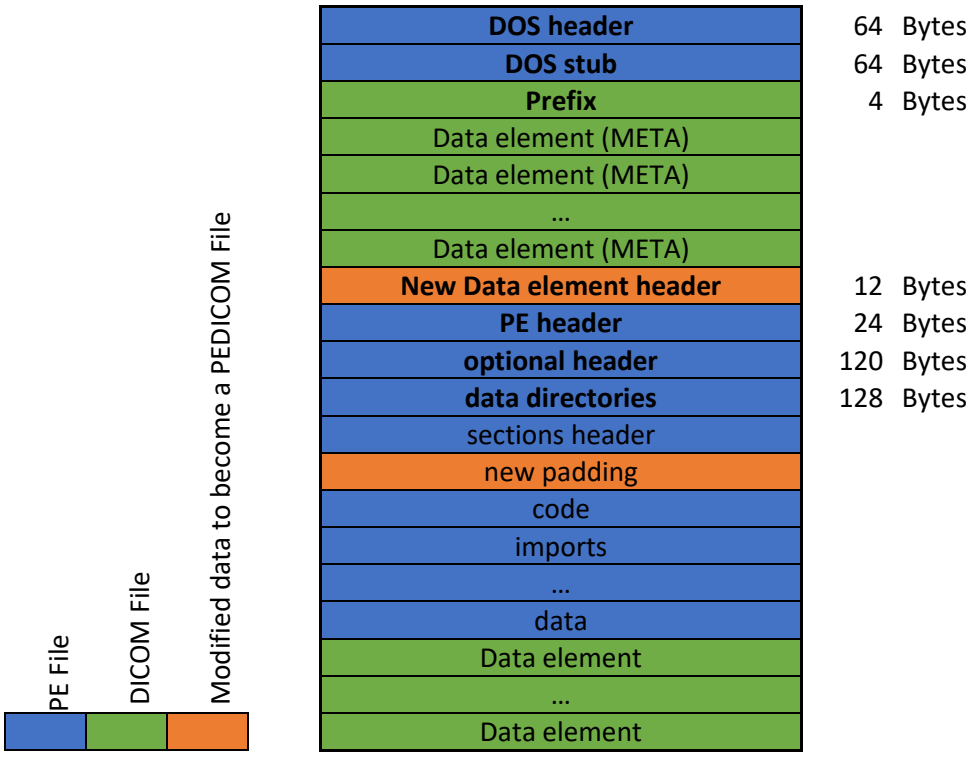

Gráfico 4: Formato PE-DICOM

Esto se logra fragmentando el PE en dos piezas: la cabecera DOS y el resto del binario. La cabecera DOS se mete en los 128 bytes del *preamble* del DICOM, teniendo que introducir padding en el DOS-STUB si el DOS-HEADER fuera menor de 128 bytes. Como el campo *e_lfanew* apunta a la firma PE, si se recalculan los campos SizeOfHeaders y PointerToRawData, actualizando a su vez el *padding* para respetar el FileAlignment, el PE seguirá siendo funcional. La segunda parte del PE se introduciría en un nuevo DataElement en el DataSet del DICOM, creando una nueva cabecera de 12 bytes, que deberá ser computada en las modificaciones del PE. Así, el *Preamble* contendrá la

cabecera DOS, mientras que una nueva sección privada del DICOM almacenará el resto del PE.

Desde el punto de vista del intérprete de PE (el loader de Windows) se ha aumentado el DOS-STUB, y al llegar a la dirección 0x3c continúa la ejecución en la firma PE. Desde el punto de vista del intérprete de DICOM, se ha añadido una sección nueva con *tag* privado que ignorará.

2.3.3 Tercer estudio: Ataques de ingeniería social alterando DICOM

El tercero advierte que DICOM es un formato antiguo, que no está diseñado para ser seguro, y que puede ser un vector de ataque contra una infraestructura clínica. Expone que el sector tiene una dependencia muy fuerte de los DICOM para diagnosticar enfermedades complejas, y es altamente vulnerable a intrusiones físicas con DICOM contaminados, por ejemplo con *ransomware*. Propone un escenario de ataque en que un paciente entrega al médico una *memoria USB* o un DVD con un DICOM maligno, que sería insertado y se desplazaría hasta el PACS, contaminando toda la organización. Señala también que las medidas de comprobación de integridad tradicionales afectan a la capacidad operativa de los hospitales (recordemos que estamos en un entorno OT), y que es complicado determinar cuándo un fichero está limpio o no, debido a la sofisticación de las técnicas de esteganografía. Propone como solución el empleo de tecnologías CDR/CDNR (*Content Disarm and Reconstruction*/*Content Deconstruction Neutralization and Reconstruction*) de grado militar, que consiste en reducir, sanear y reconstruir el fichero.

Otros ensayos proponen medidas para comprobar la integridad de los DICOM, como la adición de una marca de agua (VVAA, Security Protection of DICOM Medical Images Using Dual-Layer Reversible Watermarking with Tamper Detection Capability 2011), impidiendo a un atacante modificarlos en el trayecto de la máquina MRI/CT al puesto de sanitario (primer estudio), pero no serviría para la técnica del segundo estudio ni para el escenario del tercer estudio.

2.3.4 Contribución

Cuando en abril de 2019 Markel Picado publica un ensayo de ocho páginas explicando el CVE-2019-11687 (Picado, Attacking Digital Imaging and Communication in Medicine (DICOM) file format standard 2019), demuestra cómo los DICOM pueden emplearse para comprometer instalaciones clínicas.

Este proyecto pretende determinar el alcance y capacidades de los ataques empleando DICOM y reglas de detección de los mismos. A pesar de la necesaria parte ofensiva, se trata de un proyecto de orientación defensiva, que permita a los defensores conocer hasta dónde pueden llegar los atacantes y cómo detectarlos. En el momento en que se redacta esta memoria no hay constancia de que esta técnica se haya empleado en un ataque real, pero es una técnica más que podría utilizarse para comprometer una infraestructura sanitaria, como se demostrará.

Además, se diseña un *framework* para explotar el CVE de varias formas, dotando a los DICOM de capacidades adicionales empleando técnicas avanzadas y extendiendo el término *Bad-DICOM*. Estas capacidades incluyen escalada de privilegios, ejecución de comandos, identidad políglota o incluso herramientas completas de terceros. Tanto al *framework* como a su resultado lo denominaremos *Bad-DICOM*.

3 ANÁLISIS DE REQUERIMIENTOS

En esta breve sección comentaremos los retos con los que esperamos encontrarnos para desarrollar y emplear nuestra herramienta y probarla en un entorno simulado. Hablaremos de las tecnologías y bibliotecas específicas que emplearemos para alcanzar nuestros objetivos.

3.1 REQUISITOS PARA LA HERRAMIENTA *BAD-DICOM*

Se va a construir una herramienta que permita generar muestras maliciosas para su posterior estudio, tomando como entrada un fichero DICOM y una muestra de malware y combinándolas para obtener una radiografía infectada.

Para ello, necesitaremos radiografías DICOM, diferentes muestras de malware y un entorno de pruebas para simular el hospital que será atacado por nuestro *Bad-DICOM*.

Para las radiografías emplearemos la web *DICOM library* (Softneta s.f.), que ofrece gran variedad de DICOM para investigadores.

Las muestras de *malware* las obtendremos de *Any.run* (Any.Run s.f.), un servicio de *sandboxing* público.

La herramienta será desarrollada en Python3, aprovechando las bibliotecas *PeFile* (PyPi.org s.f.) y *Pydicom* (ResearchApps s.f.). La primera permite modificar con comodidad diferentes secciones de los ficheros PE, mientras que la segunda biblioteca contiene multitud de funciones para interactuar con los DICOM.

Utilizaremos también un lector de DICOM de código abierto. Hemos elegido *Dicompyler*, desarrollado en Python.

Respecto al entorno de pruebas, emplearemos una máquina virtual con Windows 7 de 64 bits, ya que permite ejecutar muestras compiladas para arquitectura x86 y x64. Esta máquina emulará a un hospital, conteniendo usuarios, documentos y software propio de estas instalaciones, así como credenciales, historial de navegación y *anti-virus*.

4 DISEÑO DE ATAQUES A INFRAESTRUCTURAS CLÍNICAS

Este capítulo se compone de tres partes: modelado de ataques contra infraestructuras clínicas, ejemplos de ataques siguiendo este modelo y funcionamiento interno de la herramienta *Bad-DICOM*.

En la primera sección se define un modelo por fases que permita estudiar ataques contra el entorno clínico, permitiendo diseñar contramedidas apropiadas para cada etapa éstos. Las contramedidas se expondrán en el punto 6, después de analizar el comportamiento de los *Bad-DICOM* en el entorno de pruebas.

En la segunda sección se ofrecen ejemplos de estos ataques, siguiendo el modelo por etapas definido.

En la tercera sección se expone el funcionamiento de la herramienta para llevar a cabo las funciones descritas anteriormente.

4.1 MODELADO DE ATAQUES CONTRA INFRAESTRUCTURAS CLÍNICAS

Se propone un modelo basado en etapas que describa posibles ataques que realizaría un cibercriminal o un grupo APT. De esta manera, en el apartado de contramedidas se propondrán soluciones para contrarrestarlos en cada fase del mismo.

Para facilitar el estudio y la respuesta ante el ataque, dividiremos los ataques en tres etapas genéricas, que aterrizaremos con ejemplos en los siguientes apartados. Estas fases serán:

1. Etapa de Intrusión: agrupará los dos primeros pasos del apartado anterior: Toda la fase de recopilación de información sobre el objetivo, si la hubiera, la selección de *payloads*, el preparado del DICOM malicioso y demás preparativos para llegar al equipo de la víctima.

2. Etapa de Explotación: esta etapa la lleva a cabo la víctima, conectando a su equipo un dispositivo infectado, ejecutando el *payload* o abriendo un correo malicioso e infectándose. Existen múltiples maneras de realizar esta etapa, como se describirá posteriormente.

3. Etapa de Post-Explotación: una vez ejecutado el *payload*, el atacante tiene acceso a la red interna y control sobre el equipo. Ahora se podría contactar contra el *Command&Control*, descargar nuevas piezas de *malware* o desplazarse lateralmente. Si el atacante se desplaza hasta el PACS, podría infectar al resto de DICOM y distribuirse por toda la organización.

Existen muchos modelos para describir ataques avanzados, como la CyberKill-Chain (Lockheed Martin s.f.), la Matriz de Mitre ATT&CK (MITRE s.f.) y la OSSTMM (*Open Source Security Testing Methodology Manual*) (ISECOM s.f.), pero por simplicidad los resumiremos en estas tres fases.

4.1.1 Requisitos para llevar a cabo el ataque

Imaginemos el siguiente escenario.

Un paciente acude a una clínica privada para realizarse una operación, porque en la sanidad pública no le dan cita. Otro paciente acude a operarse a un hospital público para no pagar una operación. Una clínica recibe un email con un paciente solicitando una segunda opinión, enviando una radiografía por no poder acudir presencialmente (telemedicina, vivir en un país diferente, cuarentena por coronavirus…).

En todos estos escenarios, el paciente porta un DVD o una *memoria USB* con un DICOM, que el médico abrirá con un lector para ver qué contiene y realizar una valoración. El DICOM contiene un *payload*, que se ejecuta de alguna forma y compromete el puesto del médico.

Desde la perspectiva técnica del atacante, supone realizar una intrusión física o enviar un *spear phishing* con un DICOM infectado, logrando de algún modo que la víctima lo ejecute y se infecte. Para esto serán necesarios varios pasos:

1. Se debe preparar un DICOM malicioso.
2. El DICOM debe llegar al equipo de la víctima
3. La víctima debe ejecutar el DICOM de la manera adecuada para hacer saltar el *payload*.

Asumimos que una vez se ejecute el *payload* el equipo será comprometido, cayendo bajo el control del atacante.

4.1.2 Vectores de ataque

Antes de diseñar los ataques vamos a explorar los distintos vectores de ataque a los que podría ser vulnerable un hospital. Posteriormente contemplaremos algunos de estos vectores, aterrizándolos en nuestro escenario.

Ingeniería social: Se trata de una disciplina englobada en la seguridad de la información que emplea manipulación psicológica para engañar a la gente, logrando que revelen información o cometan alguna acción (Anderson 2008). También puede definirse como:

> *Cualquier acto que influencia a una persona para tomar una acción que puede ir o no en contra de sus intereses (Social Engineering Defined - Security Through Education s.f.). Realmente no es un vector de entrada, sino una disciplina de la que hacen uso varias técnicas.*

Spear phishing: se trata de una técnica que hace uso de la ingeniería social para engañar a un individuo para que cometa una acción, como descargar malware o introducir credenciales. A diferencia del *phishing*, que es genérico, el *spear phishing* se trata de un ataque dirigido con un estudio previo de la víctima. Para ello, emplearía un tono imperativo o atractivo para incitar a descargar un adjunto malicioso con un nombre tentador, o un enlace a una página casi idéntica a una real, relacionada con las áreas del interés del objetivo.

> *A targeted phishing attempt that seems more credible to its victims and thus has a higher probability of success. For example, a spear phishing e-mail may spoof an organization or individual that the recipient actually knows (Brand 2014)..*

USB drop: Técnica de ingería social que consiste en dejar caer una memoria USB con un *payload* malicioso, normalmente un *script*, un documento malicioso o incluso un ejecutable. Una víctima poco concienciada, motivada por la curiosidad, conectará la memoria a su equipo, infectándose. En organizaciones con empleados concienciados la mayoría de memorias USB terminan entregándose al área de Seguridad, si la hubiera, pero si se suelta una cantidad suficiente de unidades aumenta la probabilidad de que un usuario se infecte. Es una variante física de *phishing*.

intrusiones físicas: Consiste en entrar físicamente a las instalaciones de la organización, desde la que se podría tener acceso físico a los equipos, pudiendo manipularlos, o atacar las redes. Si la organización no está bien bastionada, un atacante podría conectarse a la red interna enchufándose a un puerto Ethernet.

Intrusión remota: Consiste en acceder a la red interna a través de un servicio expuesto. Esto puede realizarse empleando contraseñas filtradas en la *dark web*, obteniéndolas mediante *phishing* o explotando una vulnerabilidad de un servicio expuesto a Internet.

Intrusión indirecta: Similar a la anterior, consiste en comprometer un servicio de la misma red que la máquina objetivo y realizar posteriormente un desplazamiento lateral.

4.2 EJEMPLOS DE ATAQUES DIVIDIDOS POR FASES

Vamos a exponer ejemplos de posibles ataques en cada etapa por separado, pudiendo combinarse cualquiera de estos ejemplos entre sí intercambiando las distintas etapas.

4.2.1 Etapa de Intrusión

En esta etapa el objetivo es generar un *Bad-DICOM* con el *framework* (que se explicará en detalle en el siguiente punto) y hacerlo llegar al equipo de nuestra víctima. Para esto pueden explotarse los vectores de entrada descritos antes. Pongamos unos casos de ejemplo:

Spear phishing: Un médico recibe un correo de un compañero pidiéndole una valoración de un paciente. Alguien de la Administración solicita urgente un diagnóstico de un conocido, adjuntando el DICOM. Un paciente extranjero quiere viajar a la clínica a operarse, pero quiere saber primero si lo que tiene es grave.

USB drop: en la consulta, alguien deja una memoria USB olvidada, y el médico lo conecta a su equipo para ver de quién es. Una memoria sin dueño aparece en el suelo, o sobre una mesa, en la sala de descanso de los sanitarios, a la vista. Dentro, contiene la radiografía con cáncer de un conocido político.

Intrusiones físicas: Un paciente acude a una clínica privada para realizarse una operación, porque en la sanidad pública no le dan cita. Otro paciente acude a operarse a un hospital público para no pagar una operación. Una clínica recibe un email con un paciente solicitando una segunda opinión, enviando una radiografía por no poder acudir presencialmente (telemedicina, vivir en un país diferente, cuarentena por coronavirus…). En todos estos escenarios, el paciente porta un DVD o una *memoria USB* con un DICOM, que el médico abrirá con un lector para ver qué contiene y realizar una valoración. El DICOM contiene un *payload*, que se ejecuta de alguna forma y compromete el puesto del médico.

Intrusión remota: un servidor PACS está expuesto a internet. Con unas credenciales débiles, el atacante logra acceder y depositar un *bad-DICOM*. Quizás incluso es capaz de infectar todos los DICOM del PACS.

Intrusión indirecta: Un hospital ha implantado el teletrabajo con mucha prisa. Un atacante descubre servicios RDP mal bastionados, y logra acceder al equipo del médico, pero no tiene privilegios. Para escalar, deja un *bad-DICOM* en el escritorio del médico.

Es necesario decir que, en la mayoría de estos escenarios, la memoria USB o el DVD contienen un lector portable que permite al médico abrir el DICOM. Esto es una práctica

habitual en estas situaciones (VVAA, Considerations for Exchanging and Sharing Medical Images for Improved Collaboration and Patient Care: HIMSS-SIIM Collaborative White Paper 2016) (Varma 2012). Al explotar cualquiera de estos vectores, el *bad-DICOM* estaría en el equipo de la máquina, a la espera de la acción de la víctima.

4.2.2 Etapa de Explotación

En esta etapa, la víctima tiene la iniciativa. Para activar el *payload* malicioso, empleará la segunda pieza del f*ramework Bad-DICOM*, que puede tomar varias formas. Estas piezas se describirán en el siguiente punto, pero todas tienen en común que la víctima cree estar realizando una acción legítima, cuando en realidad ha sido engañada para activar nuestro malware.

Cabe destacar la existencia de un dispositivo denominado *bad-usb* (Serna 2018), que asegura la explotación del objetivo al mismo conectarlo. Se trata de un dispositivo con el aspecto físico de una memoria USB, que bajo la carcasa tiene un microprocesador, generalmente un *ATtiny85* (Isaac s.f.) o un *atmega32u4* (ATmega32u4 s.f.), programado para ser reconocido por el equipo como un teclado. Cuando es conectado, carga los drivers necesarios y comienza a ejecutar los comandos que le fueran programados. A nivel de equipo es un teclado, y si fuera conectado mientras el equipo está bloqueado no funcionaría. Estos dispositivos pueden sofisticarse mucho, e incluso contener una memoria SD. En cuanto la víctima conectase el *bad-USB*, éste empezaría a ejecutar los comandos predefinidos, pudiendo ejecutar nuestro *payload* y comprometer el equipo.

4.2.3 Etapa de Post-Explotación

Aquí comienza la fiesta. Una vez se ha activado el *payload*, tenemos el control de la máquina. Según los *payloads* que hayamos metido en el *Bad-DICOM*, podríamos ejecutar una rutina de comandos que escalara privilegios, escaneara vulnerabilidades y contactara contra un *Command&Control*, o almacenase el resultado en una nueva sección del *Bad-DICOM* (si el escenario nos permite recuperarlo). Puede extraerse un binario oculto y cifrado, volcarlo en una localización oculta del sistema o una clave de regitro y crear una tarea programada que lo invoque. Podrían emplearse herramientas de Nirsoft para volcar credenciales. Los *Bad-DICOM* pueden contener como payload Frameworks completos de post-explotación de terceros. Podría lograrse persistencia de nuestro *malware* o directamente activarlo. Se puede desactivar el antivirus, o moverse lateralmente a otros equipos de la red, o sencillamente realizar un escaneo de red para comprobar la visibilidad del equipo de cara a un ataque más agresivo en el futuro. Podrían robarse datos del equipo para venderlos en la *dark web*. El PE podría ejecutarse en memoria, sin dejar artefactos forenses en el disco, dificultando así la investigación del analista (TheWover 2020).

Esta etapa depende de las intenciones del atacante, pero abre un abanico enorme de posibilidades.

4.3 THE BAD-DICOM FRAMEWORK

The Bad-DICOM framework será una herramienta desarrollada en Python 3 que constará de dos partes separadas.

La primera pieza genera un *Bad-DICOM*. Recibe como *input* un DICOM (que puede estar infectado o no estarlo) y un *payload* (generalmente un *EXE*, una *DLL* o un *script* en *bash* o *powershell*). Esto puede hacerse recursivamente, añadiendo un *payload* diferente en cada ocasión.

Hay que tener en cuenta que, en una intrusión normal, el *malware* trata de ser lo más pequeño posible para pasar desapercibido, oculto en una imagen o un documento malicioso, pero en este escenario tenemos mucho espacio disponible, porque los DICOM son imágenes muy grandes. Por ello, podemos considerar que tenemos espacio ilimitado y no necesitamos descargar nuevos fragmentos de *malware* ni realizar conexiones de red que un IDS podría detectar. Es posible almacenar las siguientes partes cifradas dentro de nuestro *Bad-DICOM*, e incluso emplearlo para exfiltrar información creando una nueva sección.

La segunda pieza interactuará con el *Bad-DICOM* durante la etapa de explotación, detonando el *payload*.

La herramienta se ha programado en python3 para poder emplear las librerías *pefile* y *pydicom*. Se definen tres tipos de funcionamiento, para crear tres tipos de *Bad-DICOM* diferentes y combinables: *modo pe-dicom*, *modo stego* y *modo powershell*.

4.3.1 Modo PE-DICOM

Como define Markel Picado en su ensayo (Picado, Attacking Digital Imaging and Communication in Medicine (DICOM) file format standard 2019), este modo genera un PE-DICOM como *output*. Necesita recibir un PE y un DICOM. En primer lugar, la herramienta analiza el DICOM hasta encontrar el final del último *DataElement* con *Tag* de metadatos (0002, XXXX). Este valor se guarda.

A continuación, el programa analiza el tamaño del DOS-STUB del PE, rellenando hasta 64 bytes en caso de ser menor de ese valor, para poder rellenar la cabecera. Posteriormente, fragmenta el DOS-HEADER y el resto del PE.

Se toma el valor original del campo *e_lfanew* (en la posición 0x3c) y se le añade todos los bytes del DICOM desde la firma DICM hasta el valor almacenado anteriormente: el final de la última sección de metadatos. Posteriormente le añadimos 12 bytes para contemplar la nueva cabecera. Esta modificación requiere actualizar los campos antes citados: *SizeOfHeaders* y los *PointerToRawData* de cada sección. Además, se debe añadir un *padding* entre la cabecera y el resto del cuerpo, que en el caso de la herramienta dibuja en *ASCII*, correctamente alineado, un castillo y una firma.

$$NewSizeOfHeaders = SizeOfHeaders + Prefix + MetadataDataElements + NewPadding + 12 \text{ bytes}$$

$$NewPointerToRawData = PointerToRawData + (NewSizeOfHeaders - SizeOfHeaders)$$

En este punto ya podemos calcular el tamaño del binario que esconderemos en la nueva sección.

Posteriormente, inyecta una nueva sección formando una cabecera de 12 bytes completamente nueva, en la que graba un *private Tag* (0009, XXXX), el tipo de información que contiene la sección (OB, 4F 42 00 00) y el tamaño de la misma, que se corresponderá al tamaño del PE modificado desde la firma PE hasta el final.

Nótese que al modificar el binario PE original también cambiaría su hash, lo que permitiría evadir detecciones estáticas basadas en firmas como el hash, pero no el *imphash*.

Una prueba de concepto de un PE-DICOM resultado de esta herramienta puede encontrase públicamente en GitHub (Kosmokato 2019).

Para seleccionar este modo de funcionamiento lanzaremos la herramienta como:

C:>python3 bad-dicom.py --pedicom <DICOM> <PE>

La herramienta devolverá el fichero PEDICOM.dcm

4.3.2 Modo Stego

El modo *stego* toma su nombre de la esteganografía:

> *The use of encoding techniques for hiding content within other content. For example, there are computer-based steganographic techniques and tools for embedding the contents of a computer file containing engineering diagrams and text into an image file (e.g., a JPG document) such that the existence of the engineering data in the image file is difficult for the observer to detect (NATO 2013).*

Así, este modo de funcionamiento toma un *payload* (un binario, que suele ser un PE pero podría ser un ELF para Linux o un Mach-O para MacOS) y lo esconde íntegramente en una nueva sección del DICOM. De este modo, las firmas de PE que analizan el *Preamble*. Para evadir completamente las firmas de detección, por ejemplo las reglas Yara, existe la posibilidad de codificar o cifrar esta sección, por ejemplo aplicando un sencillo base64 a todo el contenido. Además, con esto logramos cambiar el hash del binario mientras está en el DICOM, lo que ayudaría a pasar desapercibido. Recordemos que este *Bad-DICOM* no tiene infectado el *preamble*, por lo que es indistinguible de otro DICOM legítimo.

Para seleccionar este modo de funcionamiento lanzaremos la herramienta como:

C:>python3 bad-dicom.py --stego <DICOM> <PE>

La herramienta devolverá el fichero STEGO.dcm

4.3.3 Modo Powershell

Este modo es una variante del anterior, distinción necesaria a la hora de asignar un intérprete para cada tipo de *payload*. En este caso, el *framework* tomará un DICOM, infectado o no, y añadirá una sección adicional en la que ocultará un fichero de scripting, normalmente un *.ps1* de *Powershell*. Esto tiene múltiples funcionalidades y permite añadir toda la potencia de los lenguajes de scripting a nuestro ataque, así como cargar módulos adicionales a la consola destino.

Podríamos así ocultar un *framework* de post-explotación de terceros como *Metasploit* (Rapid7 2019), *Empire* (BC-Security 2020) o *iBombShell* (ElevenPaths 2020), cargándolo directamente a memoria en la fase de Explotación, y no dejar rastro de nuestra intrusión en la máquina. Otras capacidades que podría explotar es el bypass de medidas de seguridad como la UAC (*User Account Control*) (P. H. González 2018) o el AMSI (*Antimalware Scan Interface*) (Rasta-Mouse, Amsi Scan Buffer Bypass 2019) de Windows, o el uso de *LOLBINs* (Spehn 2018) para escalar privilegios en la máquina (SwissKyRepo 2020) (Oddvar s.f.) o descubrir vulnerabilidades (Rasta-Mouse, Watson 2020), preparando el terreno a otro *payload* oculto en el *Bad-DICOM*.

Este *Bad-DICOM* tampoco tiene infectado el *Preamble*, y si se codifica o cifra correctamente podría evadir todas las medidas de detección estáticas, al ser indistinguible de un DICOM legítimo.

Para seleccionar este modo de funcionamiento lanzaremos la herramienta como:

C:>python3 bad-dicom.py --powershell <DICOM> <.ps1>

La herramienta devolverá el fichero POWERSHELL.dcm

4.3.4 Loader

La segunda pieza ejecuta el *payload* oculto en el *Bad-DICOM*. Se trata de la parte de código que ejecutará la víctima, por lo que hemos diseñado varios aspectos y formatos para distintos escenarios, pero todos tienen en común la función de *parsear* el *Bad-DICOM*, identificar el *payload* y su tipo, e interpterarlo.

En función del modo de funcionamiento reconocido (expuesto en los puntos anteriores) el Loader realizará una acción diferente:

Modo PE-DICOM: El PE-DICOM tiene una peculiaridad muy interesante. Si se ejecuta con un lector DICOM, se interpreta como un DICOM. Si se ejecuta con un intérprete de *powershell*, se abre con el lector DICOM por defecto. Pero si se ejecuta desde la COMMAND.COM de Windows, se ejecutará como PE, incluso si su extensión es .dcm.

Así, nuestro Loader invocará al intérprete de *COMMAND.COM*, *cmd.exe* para ejecutar el *Bad-DICOM* como PE.

Modo Stego: Si el loader reconoce un *payload* en modo *stego*, lo extraerá y lo almacenará en una ruta predefinida, por defecto en %TEMP%. Desde ahí, lo ejecutará. Otras opciones

podrían ser la creación de una tarea programada que invocase al binario en un momento deseado, variación de la ruta o cualquier otro tipo de método de persistencia en el sistema. Esto podría combinarse con el siguiente modo, para escalar privilegios y asegurar el entorno antes de que el binario se descrifre y se ejecute. El tamaño del binario a extraer está definido en la cabecera del *DataElement* privado.

Modo Powershell: En este modo, el Loader invoca al intérprete de powershell, powershell.exe, pasándole el contenido de todo el *DataElement* privado. En la cabecera se incluye el tamaño del script, por lo que igual que en el caso anterior, sólo tendría que leer el tamaño para definir el rango a interpretar.

Para engañar a la víctima de activarlo, Loader puede tener varias formas:

Open-source DICOM viewer modificado: Consiste en un lector *open-source* que ha sido modificado, inyectándole una subrutina antes de ejecutarse que activa el payload del *Bad-DICOM*. Después de ejecutarse, reanuda su actividad legítima, logrando que la víctima achaque el retardo a una tardanza en la carga del DICOM. En nuestras pruebas hemos escogido *Dicompyler* (Bastula 2020) por estar hecho en python y emplear librerías ya utilizadas en nuestra herramienta, pero podría usarse cualquiera. Posteriormente este programa se compilará para la versión del equipo de la víctima, habitualmente Windows x86 o Windows x64, incluso ambas.

Malicious .LNK + AlternateDataStream: El Loader está contenido dentro de un .LNK, un acceso directo, que ejecuta comandos primero y abre el DICOM después. Estos comandos podrían estar directamente en el .LNK o dentro de un fichero oculto en un AlternateDataStream (olmedo 2015), ligado a sí mismo o al DICOM, lo que ocultaría a la vista el fichero.

Rubber-Ducky/Bad-USB: En este caso, el lector será un programa para el microprocesador elegido, que ejecutará comandos, probablemente ejecutando un PE-DICOM, aunque podría contener su propia secuencia de comandos, empleando el DICOM como almacén para exfiltrar información, creando un nuevo *DataElement* donde almacenar el resultado del reconocimiento del equipo.

Cabe destacar que el *Loader* ha de diseñarse para trabajar en la arquitectura y sistema operativo de la máquina objetivo. Si por ejemplo un *Bad-DICOM* contiene scripts en *bash* o en *zsh*, ha de llamar al intérprete adecuado o actualizar las rutas en el sistema si detecta el empleo de esteganografía. Una manera sencilla de identificar todos estos matices podría ser emplear el segundo par de bytes del *Tag* del *DataElement* para codificar condiciones que *Loader* debe interpretar, aunque en este proyecto no se ha implementado para mantener la sencillez.

Las peculiaridades de este tipo de explotación tienen consecuencias positivas y negativas:

Ventajas: al necesitar la segunda pieza para ejecutarse, no se lanzará por accidente en una *sandbox*. En los dos últimos modos de funcionamiento ni siquiera podrá ser detectada la carga maliciosa, por lo que permitirá al *Bad-DICOM* evadir casi todos los controles de seguridad y obligará a un análisis manual de la muestra.

Inconvenientes: aunque el *Bad-DICOM* tenga una carga muy potente no seremos capaces de ejecutarla si no la activamos con la segunda pieza.

4.4 IMPLEMENTACIÓN DE ATAQUES EN UN ENTORNO SIMULADO

Para llevar a cabo los ataques se configura una máquina virtual con Windows 7 con los últimos parches de seguridad, con y sin antivirus, en el que se recrea un equipo de un hospital, "*Hospytal*". Se llena de documentos relacionados, se crean cuentas asociadas, se almacenan contraseñas.

Mediante tres pruebas de concepto de tres ataques diferentes se demostrará las capacidades de estos ataques y se diseñarán medidas de detección. Cada prueba empleará una forma de explotación del CVE-2019-11687: PE-DICOM, ocultación por esteganografía y ejecución de scripts.

Se diseñará *malware* a medida para algunos ataques, y se empleará *malware* de terceros en otros, para probar el escenario contra un *malware* real.

5 EVALUACIÓN DE RESULTADOS

En este apartado se describen tres pruebas de concepto, una para cada tipo de *Bad-DICOM*, y los efectos que tiene en la máquina objetivo.

5.1 PoC 1: Ransomware.

Empleando el modo PE-DICOM, creamos un fichero políglota que es al tiempo un DICOM legítimo y un *ransomware*. Para el *malware* tomamos una muestra de Ryuk (Any.Run 2020), que ha atacado hospitales en todo el mundo.

Cuando la víctima ejecuta el *Loader*, creyendo usar un lector legítimo, el *payload* es invocado y comienza a cifrar todos los archivos. En unos minutos, el equipo entero queda inservible, con una característica nota de rescate en cada directorio con ficheros.

EXE original: d083ecc1195602c45d9cb75a08c395ad7d2b0bf73d7e70e2fc76101c780dd38f
PEDICOM: d1ea58740b6e5ad5159d23f65283096d385285756166946bbdc2587c0c93cfdc

5.2 PoC 2: Esteganografía y droppers.

La segunda prueba de concepto expone un *Bad-DICOM* creado de la forma *stego*. Al ejecutar el Loader, reconoce el *payload* oculto y lo vuelca en la carpeta designada. Este binario resulta ser un instalador de Atom (AtomEditor s.f.) (VirusTotal s.f.).

sha256: 738beee28c2966fc73770b133a7839e2700e05dcf0c180c0e0bfc821cf07ffcd

5.3 PoC 3: Data exfiltration.

En esta prueba, el *Bad-DICOM* se crea con un DICOM legítimo y un script *.ps1*. Al ejecutar Loader, invoca al intérprete de comandos *C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe*, y ejecuta el siguiente script:

```
(netsh wlan show profiles) | Select-String "\:(.+)$" | %{$name=$_.Matches.Groups[1].Value.Trim(); $_} | %{(netsh wlan show profile name="$name" key=clear)} | Select-String "Contenido de la clave\W+\:(.+)$" | %{$pass=$_.Matches.Groups[1].Value.Trim(); $_} | %{[PSCustomObject]@{ PROFILE_NAME=$name;PASSWORD=$pass }} | Format-Table -AutoSize
```

Posteriormente, elimina el contenido del *DataElement* y graba el resultado del comando, exfiltrando las contraseñas de *Wi-Fi* almacenadas en el equipo.

6 CONTRAMEDIDAS

Estos ataques pueden mitigarse, e incluso evitarse, mediante una serie de medidas. A continuación, expondremos medidas de detección (6.1), prevención (6.2) y mitigación (6.3) para ciberataques a infraestructuras clínicas.

6.1 PREVENCIÓN

Para prevenir ataques empleando el CVE-2019-11687 proponemos dos medidas. La primera limpiará los PE-DICOM, mientras que la segunda monitorizará cualquier posible uso de Powershell por parte del *Bad-DICOM.*

6.1.1 Purificador de DICOM

Una manera efectiva de prevenir la infección de PEDICOM es sobrescribir el *preamble* de los DICOM, eliminando la posible cabecera del fichero políglota.

Una posible implementación en *Powershell* podría ser:

```
# Abrimos el fichero sospechoso
$bytes  = [System.IO.File]::ReadAllBytes("D:\DICOM.dcm")

# Sobreescribimos los 128 primeros bytes (preamble)
For ($i=0; $i -le 128; $i++) { $bytes[$i] = 0x00 }

# Guardamos los cambios en el fichero
[System.IO.File]::WriteAllBytes("D:\DICOM.dcm", $bytes)
```

Esta técnica no es efectiva si se emplea otro tipo de *Bad-DICOM*.

Otras opciones son la aplicación de actualizaciones y parches de seguridad, que dificultará y encarecerá el uso de *exploits*, y una política segura de contraseñas, como impedir que los administradores reutilicen credenciales en la red o limitar las capacidades de un usuario sin privilegios.

6.1.2 Monitorización y registro de la actividad de Powershell en el sistema

Desde el punto de vista de la seguridad defensiva, Powershell es una herramienta muy potente y peligrosa, que no guarda registro de las líneas de comandos introducidas si no se habilita explícitamente la opción *PowerShell script block logging*, habilitando una clave de registro:

> HKLM:\SOFTWARE\Wow6432Node\Policies\Microsoft\Windows\PowerShell\ScriptBlockLogging *(Microsoft s.f.)*.

6.2 DETECCIÓN

Existen varias estrategias y técnicas para detectar ciberataques, que funcionan de manera complementaria. Las clasificaremos en tres: *detección pasiva*, *detección activa* y *detección proactiva*. Definiremos cada clase y nombraremos algunas herramientas y sus características, centrándonos en la *detección proactiva* en *endpoint*, ya que el CVE-2019-11687 afecta a los ficheros DICOM. Así, definiremos reglas para detectar los ataques antes de que se produzcan, ya que, si la explotación es exitosa, lo que deberá detectarse dependerá del *payload* escogido.

Detección pasiva: Consideramos detección pasiva todas aquellas reglas y técnicas que, de manera automatizada, bloquean o eliminan acciones o ficheros identificados como maliciosos. Esto se consigue aplicando reglas de detección estática y heurística. Las reglas de detección estática buscan patrones conocidos, generalmente obtenidos de *feeds* de inteligencia propios. La heurística analiza el comportamiento de los ficheros, estableciendo una puntuación interna a cada uno y bloqueándolos cuando superan un umbral establecido. Estas configuraciones no son completamente configurables por un analista, sino que son gestionadas por el propio producto.

En esta categoría englobamos a los *antivirus* y los *firewall*. Es necesario destacar que estas reglas son creadas en base a actividad maliciosa conocida, por lo que sólo resultan efectivos contra *malware* conocido. Muestras de *malware* nuevas y atacantes avanzados podrían *bypassear* estos controles, modificando los payloads para evadir firmas, o generando una heurística errónea. En el caso de los *antivirus* (Koret 2015), una práctica muy común entre los atacantes es la modificación del código, variando un *string*, de manera que el código compilado sea diferente y desconocido para las firmas del *antivirus*. Además, el *malware polimórfico* posee capacidad de mutar su propio código, lo que dificulta todavía más su detección. En el caso de los *firewall*, pueden ser evadidos mediante *tunneling* (Barwise 2018).

Sin embargo, estas medidas de detección sirven de apoyo al analista generando logs con anomalías, y pueden bloquear *malware* comercial y genérico. Respecto a los *Bad-DICOM*, la detección pasiva podría detectar y para ataques siempre que el DICOM reutilice un *malware* conocido o sea reconocido por su comportamiento en el sistema.

Detección activa: Consideraremos detección activa a la labor que realiza un analista, aplicando diferentes técnicas y herramientas, y apoyándose en las herramientas de detección pasiva (logs de proxy, de antivirus, de firewall y de cualquier dispositivo de red con capacidades de análisis). En la detección activa, los analistas se apoyan en herramientas IDS (*Intrusion Detection System*), que pueden ser NIDS (*Network Intrusion Detection System*) y HIPS (*Host Intrusion Detection System*). Estas herramientas analizan actividad en la red y los equipos, *esnifando* el tráfico de la organización mediante un *port-mirroring*/SPAN (*Switched Port ANalyzer*) o un *TAP* (*Test Access Point*), analizándolo y buscando coincidencias con reglas de detección. Estos IDS suelen estar en equipos denominados *sondas*, que habitualmente reportan a un *correlador*. El correlador aplica una configuración, agrupa las alertas y las envía a una consola de monitorización de eventos.

En el caso del *Centro Criptológico Nacional*, el SIEM que emplean se denomina *GLORIA* (CCN-CERT s.f.), y se compone de *Argos* (sondas), *Tritón* (correlador) y *Emas* (consola de monitorización de eventos). SIEM comerciales podrían ser *Splunk* (Splunk s.f.), *QRadar* (IBM s.f.) y FortiSIEM (Fortinet s.f.).

En cuanto a SIEM *open-source* destacamos *Security Onion* (INCIBE-CERT 2017), que es una distribución de Linux que engloba herramientas *open-source* como HIDS (*Wazuh*), NIDS (*SNORT* (Cisco s.f.)*/Suricata* (OISF s.f.)), *Zeek/BroIDS* (Zeek s.f.) y *Wireshark* (The Wireshark team s.f.) entre otras. *Security Onion* utiliza la pila ELK (Elastic s.f.)

(*ElasticSearch-Logstash-Kibana*). Finalmente, otra alternativa popular es *AlienVault OSSIM* (AT&T s.f.).

Respecto a los *Bad-DICOM*, la presencia de un SIEM puede detectar el uso de software de post-explotación o incluso de exploits y conexiones inusuales de red, lo que permite reconocer y frustrar el ataque.

Detección proactiva: Los ataques más sofisticados son difíciles de detectar, siendo a menudo sigilosos y con largos intervalos de inactividad. Detrás de estos ataques se encuentran los grupos APT, que ya nombramos antes, y requieren de análisis e investigación proactiva para su detección. Esta disciplina de búsqueda de amenazas se denomina *Threat Hunting*, y sigue distintas metodologías (VVAA, TaHiTi, Threat Hunting Methodology 2018) y aproximaciones para investigar anomalías que puedan revelar la presencia de un actor avanzado oculto en la organización (Ilascu 2019).

Si en la *detección activa* los analistas buscaban coincidencias con reglas, en *Threat Hunting* se busca lo opuesto: acciones inusuales que sugieran un comportamiento anómalo o inadecuado. Para ello, se emplean diversas herramientas, que en su mayoría trabajan sobre una base de datos *ElasticSearch* realizando búsquedas definidas por el analista. En ocasiones, estas búsquedas se automatizan y programan, pasándose a llamar *analizadores*. También se emplean técnicas con anomalías estadísticas, por ejemplo para detectar dominios DGA (*Domain generation algorithm*) o buscar *User-Agent* poco comunes o incluso inexistentes. En los últimos años, también se controla la ejecución de LOLBINs, ya que al ser binarios firmados por Microsoft normalmente, son ignorados por los antivirus. El CCN emplea la herramienta CARMEN (CCN-CERT s.f.). Otras soluciones populares son Darktrace Antigena (Darktrace s.f.) y Splunk. Como alternativa *open-source* existe Zeek/BroIDS, un diseccionador de tráfico que genera logs clasificados por tipo de tráfico (SMTP, DNS, HTTP, RDP). Si se junta con el *framework* RITA (G. 2018) (ActiveCM 2020) almacena los logs en una base de datos MongoDB y permite realizar búsquedas mediante una interfaz web.

Ejemplos de analizadores podría ser una imagen *POWERSHELL.EXE* que tiene como *ParentImage* a *WINWORD.EXE*, o en nuestro caso, un *CMD.EXE* que ejecuta un fichero .DCM

Estas herramientas beben de una base de datos alimentada por *logs* de diversas fuentes: antivirus, firewall, NIDS, HIDS, scripts personalizados, *feeds* de inteligencia como REYES (CCN-CERT s.f.), MISP (MISP s.f.) o VirusTotal (VirusTotal s.f.)…

6.2.1 Sysmon

Uno de los HIDS más populares es Sysmon (*System Monitor* (Russinovich 2019)). Sysmon es un driver incluido en las Windows SysInternals, y mediante una configuración .xml permite recoger distintas acciones realizadas en un equipo y generar un Evento del Sistema. Los eventos de Sysmon han de especificarse en el fichero de configuración, y son los siguientes:

- Event ID 1: Process creation
- Event ID 2: A process changed a file creation time
- Event ID 3: Network connection
- Event ID 4: Sysmon service state changed
- Event ID 5: Process terminated
- Event ID 6: Driver loaded
- Event ID 7: Image loaded
- Event ID 8: CreateRemoteThread
- Event ID 9: RawAccessRead
- Event ID 10: ProcessAccess
- Event ID 11: FileCreate
- Event ID 12: RegistryEvent (Object create and delete)
- Event ID 13: RegistryEvent (Value Set)
- Event ID 14: RegistryEvent (Key and Value Rename)
- Event ID 15: FileCreateStreamHash
- Event ID 17: PipeEvent (Pipe Created)
- Event ID 18: PipeEvent (Pipe Connected)
- Event ID 19: WmiEvent (WmiEventFilter activity detected)
- Event ID 20: WmiEvent (WmiEventConsumer activity detected)
- Event ID 21: WmiEvent (WmiEventConsumerToFilter activity detected)
- Event ID 22: DNSEvent (DNS query)

Para los ataques diseñados previamente, Sysmon puede detectar:

- Creación de un nuevo proceso, *CMD.EXE*, de padre **.DCM* (EventId:1),
- Carga de driver (si el payload del *Bad-DICOM* es una DLL) (EventID:6),
- Carga de una imagen *POWERSHELL.EXE* (EventID:7),
- Intento de inyectarse en otro proceso (Hasherezade s.f.) (StephenFewer 2013) (Hasherezade s.f.) (EventID:8)
- Creación de una carpeta en una ruta con permisos de escritura, como %TEMP% (EventID:11)
- Intento de bypass de AMSI (EventID:12-14)
- Intento de movimiento lateral empleando WMI (EventID:19-21)

Una configuración muy completa y con un amplio ratio de detección, con falsos positivos excepcionados, es la de *SwiftOnSecurity* (SwiftOnSecurity s.f.).

6.2.2 Reglas YARA

YARA (Yet Another Recursive Acronym) es una herramienta de búsqueda de patrones escrita en ruby y basada en reglas. Diseñada originalmente para clasificar muestras de *malware*, busca coincidencias de cadenas de texto y hexadecimales en un binario. Las firmas YARA se lanzan sobre toda clase de ficheros, y alertan en caso de encontrar coincidencias.

Los repositorios de reglas YARA públicas más conocidos son *YaraRules* (M. Alvarez s.f.) y *AwesomeYara* (InQuest 2020).

Actualmente existen reglas YARA para detectar PE-DICOM (Horn s.f.) que, aunque resultan una buena aproximación, son imprecisas. Al estar disponibles públicamente,

puede asumirse que esto es el *state of art* de la detección actual de los *Bad-DICOM*. Estas reglas presentan falsos positivos, porque alertan si la el *Preamble* es distinto de 0, o si los primeros bytes se corresponden con la firma MZ. Esto podría ocurrir por accidente y no implica actividad sospechosa. También buscan indicos de un DICOM políglota con un ELF, un JAR o un Mach-O, pero sin comprobar más allá del *MagicNumber*. En estos ataques avanzados, la segunda pieza del *framework*, *Loader*, podría reescribir el *Preamble*, o directamente la firma MZ, evadiendo todas las firmas anteriores.

La firma correcta, y que no presenta *falsos positivos* para los PE-DICOM, es la siguiente:

```
rule cve_2019_11687_pedicom : pe-dicom dicom tfg
{
  meta:
    //Esta regla busca un PE-DICOM CVE-2019-11687
    autor = "Arturo Navarro"
    fuente = "https://www.cvedetails.com/cve/CVE-2019-11687/"
    criticidad = "alta"

  condition:
    uint16(0)                 == 0x5a4d and  //MZ signature
    uint16(uint32(0x3C))      == 0x4550 and  //PE signature
    uint16(uint32(0x3C) - 12) == 0x0009 and  //Private DataSet
    int32(0x80)               == 0x4d434944  //DICM signature
}
```

Esta firma nos asegura que el binario tiene cabecera *MZ*, que *e_lfanew* apunta a una dirección de memoria donde se encuentra la firma *PE* y donde, 12 bytes antes, se encuentra un *private tag* de un *DataElement*, y que además contiene la firma *DICM* en la posición 0x80.

Sin embargo, en el caso de los otros *Bad-DICOM*, no existe una manera sencilla de encontrar patrones debido a la posibilidad de cifrar y encriptar el contenido, y a la necesidad de revisar el documento entero en busca de estos patrones (la regla anterior sólo comprueba ciertos bytes, es muy rápida). La aproximación que proponemos emplea el módulo matemático de YARA (VirusTotal s.f.), que permite realizar un análisis entrópico en busca de anomalías. Buscando *DataElements* privados y definiendo como *offset* las cabeceras con *private tags* + 12 bytes, se podría analizar sólo el contenido de la sección. Otra aproximación podría buscar, de manera complementaria, secciones de gran tamaño.

YARA puede combinarse con otras herramientas de código abierto para cazar anomalías buscando patrones. Un modelo sencillo (Bernal 2019) consiste en conectar *Zeek*/*BroIDS* a un SPAN del tráfico. Cuando *Zeek* extrae binarios, un script en *python* lanza reglas YARA periódicamente contra estos adjuntos, clasificándolos en función de los resultados. El script en *python* debería entonces alertar con el resultado de YARA, generando un log que puede lanzarse contra un SIEM, una base de datos o incluso un Email.

YARA es una herramienta muy sencilla y potente que es empleada por la mayoría de empresas de antivirus, y puede arrojarse tanto contra los adjuntos de correo electrónico como a los binarios en un equipo, lo que la convierte en la herramienta idónea para detectar los *Bad-DICOM*.

6.3 MITIGACIÓN

La mitigación sucede cuando el ataque ya ha tenido lugar, y trata de evitar el impacto en la organización en la medida de lo posible. En esta sección se explica la topología de red más común en un entorno clínico y las soluciones que se pueden implementar.

6.3.1 Topología de red

Las redes clínicas suelen ser horizontales y generalmente constan de los siguientes elementos (Mirsky 2019):

Servidor PACS: Almacena, organiza y ofrece los DICOM. Suele tener asociada una base de datos SQL o MongoDB, y puede estar alojado localmente o en la nube (Siwicki 2018). Es el centro del sistema PACS.

Servidor RIS: (*Radiology Information System*), gestiona las imágenes médicas y los datos asociados. Su función principal es rastrear pedidos de imágenes de radiología y los informes de los radiólogos. Los médicos de la red interna del hospital pueden interactuar con el RIS para solicitar escaneos, recibir los informes resultantes y obtener los escaneos DICOM.

Workstation estándar: Equipo de trabajo empleado para controlar un escáner TC o MRI. Durante una cita, el técnico configura el escáner y captura las imágenes empleando esta máquina, que después enviará todas las imágenes en formato DICOM al PACS para su almacenamiento.

Workstation de radiología: Permite al radiólogo ver exploraciones almacenadas en el PACS desde varias ubicaciones. Este equipo suele estar en un despacho, pero podría estar en un equipo personal o incluso en un dispositivo móvil (a través de internet o red local).

Servidor Web: Servicio que permite a los radiólogos consultar los DICOM del PACS a través de internet, mediante el uso de un navegador (MedDream (MedDream s.f.), Orthanc (Orthanc s.f.)), una app en un dispositivo móvil (Mobility Synapse de FujiFilm (FujiFilm s.f.)) o acceder a través de una API web (Dicoogle (VVAA. 2010)).

Equipos de Asistente de Administración: Estos equipos cuentan con salida a internet (por ejemplo para correos electrónicos) y acceso a la red del PACS. El acceso al PACS estaría habilitado para que el asistente pueda gestionar las citas a los dispositivos: cuando un paciente llega a la sala del equiopo MRI/CT, por razones de seguridad, el técnico confirma la identidad del paciente con los detalles enviados a la estación de trabajo (ingresados por el asistente). Esto asegura que los escaneos no se mezclen accidentalmente entre pacientes.

Red del Hospital: Varios departamentos tienen acceso al PACS: oncología, cardiología, patología, quirófano/cirugía. En estos casos, varios equipos por todo el hospital pueden cargar archivos DICOM desde el servidor con las credenciales correctas. Además, es común que un hospital implemente puntos de acceso Wi-Fi, que están conectados a la red interna, para el acceso de los empleados.

Algunos hospitales presentan una topología más sencilla, donde todo está conectado a Internet (Beek 2018).

6.3.2 Propuesta: Segmentación de la red

Una segmentación de red limita el movimiento de un atacante, dificultando su movimiento lateral por la organización una vez comprometido el equipo víctima inicial. Por ejemplo, la división de diferentes subredes para distintas funcionalidades, existencia de firewalls internos y aislamiento de Internet, impidiendo la conexión directa de las redes con servicios críticos a Internet. Finalmente, si todo el tráfico saliente a Internet pasa por un proxy corporativo será más sencillo monitorizar y registrar accesos no autorizados a la misma.

7 CONCLUSIONES

En esta sección realizamos un recorrido por toda la memoria, resaltando los puntos clave del proyecto, evaluando los resultados y exponiendo propuestas para el futuro.

Las tecnologías de la información crecen a un ritmo vertiginoso. Es necesario implementar la seguridad desde el diseño, especialmente en infraestructuras críticas, y concienciar a los empleados en Seguridad. Los ataques contra hospitales no dejan de aumentar en frecuencia, intensidad y complejidad, especialmente durante la pandemia del Covid-19, y es preciso tomar medidas de prevención, detección y mitigación, tanto activa y pasivamente como proactivamente. La dependencia tecnológica en el sector sanitario cada vez es mayor, lo que los hace más vulnerables. El entorno OT también es inseguro, y muy difícil de reparar y mantener actualizado.

Como hemos mostrado en esta memoria, los ficheros de imágenes médicas DICOM no son seguros: pueden contener *malware* y es preciso implementar medidas de seguridad y análisis de los mismos. Estos ficheros médicos deben ser considerados como ficheros ejecutables en potencia, y deben pasar los mismos controles y las mismas medidas de detección que el resto de ejecutables.

Basándonos en publicaciones y estudios existentes, hemos realizado un proceso de ingeniería inversa de los mismos para poder replicar dichas investigaciones y llevarlas más allá, a un estadio donde la detección resulta bastante más compleja.

Hemos creado una herramienta funcional, llamada *Bad-DICOM*, que permite generar muestras maliciosas con las que se ha demostrado el peligro de estos ataques, así como sus limitaciones y capacidades. La herramienta *Bad-DICOM* puede ser empleada por consultores de seguridad para realizar intrusiones contra sistemas clínicos y comprobar la eficacia de sus medidas de seguridad, mientras que permite a analistas crear variaciones más complejas para estudiar su comportamiento.

Hemos diseñado posibles ataques mediante un modelo dividido en tres fases, que permiten una estrategia defensiva para cada etapa.

Los patrones de detección que hemos diseñado permitirán a los equipos de Seguridad detectar este tipo de amenazas, mientras que las medidas de mitigación que proponemos lograrán contrarrestar en gran medida el impacto de este tipo de ataques en la organización.

Finalmente, queremos reseñar que, gracias a este proyecto, hemos aprendido y mejorado multitud de conocimientos; entre ellos tecnologías empleadas en los entornos clínicos actuales, lenguajes modernos de programación como *python* y el entorno de programación *powershell*.

Como trabajo futuro de este proyecto, proponemos añadir módulos y opciones a la herramienta *Bad-DICOM*. Por ejemplo: *escalada de privilegios*, *evasión de anti-virus* o *persistencia en el sistema*, de manera similar a la que ofrecen otras herramientas profesionales.

8 REFERENCIAS

s.f.

Abrams, L. «New Nefilim ransomware threatens to release victims data.» *Bleeping Computer.* 2020. https://www.bleepingcomputer.com/news/security/new-nefilim-ransomware-threatens-to-release-victims-data/ (último acceso: 9 de abril de 2020).

—. «Ransomware Gangs to Stop Attacking Health Orgs During Pandemic.» *Bleeping Computer.* 2020. https://www.bleepingcomputer.com/news/security/ransomware-gangs-to-stop-attacking-health-orgs-during-pandemic/ (último acceso: 9 de abril de 2020).

—. «Ryuk ransomware keeps targeting hospitals during the pandemic.» *Bleeping Computer.* 2020. https://www.bleepingcomputer.com/news/security/ryuk-ransomware-keeps-targeting-hospitals-during-the-pandemic/ (último acceso: 9 de abril de 2020).

—. «TA505 hackers behind Maastrich University ransomware attack.» *Bleeping Computer.* 2020. https://www.bleepingcomputer.com/news/security/ta505-hackers-behind-maastricht-university-ransomware-attack/ (último acceso: 9 de abril de 2020).

ActiveCM. «RITA: Real Intelligence Threat Analytics.» *GitHub.* 2020. https://github.com/activecm/rita (último acceso: 11 de abril de 2020).

Anderson, Ross J. *Security engineering: a guide to building dependable distributed systems.* Indianapolis: Wiley, 2008.

Any.Run. «12597de0e709e44442418e89721b9140.» *Any.Run.* 2020. https://app.any.run/tasks/ccc779cd-8d29-4da2-af53-3b49efefb206/ (último acceso: 10 de abril de 2020).

—. *Any.run, interactive malware analysis.* s.f. https://app.any.run/ (último acceso: 10 de abril de 2020).

AT&T. «AlienVault OSSIM, The world's most widely used open source SIEM.» *AT&T Cybersecurity.* s.f. https://cybersecurity.att.com/products/ossim (último acceso: 11 de abril de 2020).

«ATmega32u4.» *Microchip.com.* s.f. https://www.microchip.com/wwwproducts/en/ATmega32u4 (último acceso: 9 de abril de 2020).

AtomEditor. *Atom.io.* s.f. https://atom.io/ (último acceso: 10 de abril de 2020).

Barwise, I. «Covert Channel Data Exfiltration Using DNS Tunneling.» *medium.com.* 2018. https://medium.com/@z3roTrust/evasion-obfuscation-techniques-87c33429cee2 (último acceso: 10 de abril de 2020).

Bastula. «Dicompyler.» *GitHub.* 2020. https://github.com/bastula/dicompyler (último acceso: 13 de junio de 2020).

BC-Security. «Empire (forked from EmpireProject/Empire).» *GitHub.* 2020. https://github.com/BC-security/empire/ (último acceso: 10 de abril de 2020).

Beek, C. «McAfee Researchers Find Poor Security Exposes Medical Data to Cybercriminals.» *Blog de McAfee.* 2018. https://www.mcafee.com/blogs/other-blogs/mcafee-labs/mcafee-researchers-find-poor-security-exposes-medical-data-to-cybercriminals/ (último acceso: 9 de abril de 2020).

Bernal, D. «Detecting Malicious Files with YARA Rules as They Traverse the Network.» *BlackHat USA.* 2019. https://i.blackhat.com/USA-19/Wednesday/us-19-Bernal-Detecting-Malicious-Files-With-YARA-Rules-As-They-Traverse-the-Network-wp.pdf (último acceso: 11 de abril de 2020).

Brand, W. «Cybersecurity for Dummies.» *Palo Alto Networks Edition.* 2014. http://uru.ac.in/uruonlinelibrary/Cyber_Security/Cybersecurity-for-dummies.pdf.

CCN-CERT. «CARMEN.» s.f. https://www.ccn-cert.cni.es/soluciones-seguridad/carmen.html (último acceso: 11 de abril de 2020).

—. «GLORIA.» s.f. https://www.ccn-cert.cni.es/soluciones-seguridad/gloria.html (último acceso: 10 de abril de 2020).

—. «REYES.» s.f. https://www.ccn-cert.cni.es/soluciones-seguridad/reyes.html (último acceso: 11 de abril de 2020).

Cisco. «Snort.» s.f. https://www.snort.org/ (último acceso: 10 de abril de 2020).

Corkami. «Polyglot Files and PE file format.» s.f. https://github.com/corkami/docs/blob/master/PE/PE.md (último acceso: 9 de abril de 2020).

CveDetails. *CVE-2019-11687.* 2019. https://www.cvedetails.com/cve/CVE-2019-11687 (último acceso: 9 de abril de 2020).

Darktrace. «Darktrace Antigena.» s.f. https://www.darktrace.com/es/productos/antigena/ (último acceso: 11 de abril de 2020).

Efe. *La Policía alerta de un virus que ataca el sistema informático de los hospitales.* 2020. https://www.efe.com/efe/espana/portada/la-policia-alerta-de-un-virus-que-ataca-el-sistema-informatico-los-hospitales/10010-4202411 (último acceso: 9 de abril de 2020).

Elastic. «ElasticSearch, Open Source Search.» s.f. https://www.elastic.co/es/ (último acceso: 10 de abril de 2020).

—. «Kibana, tu ventana al Elastic Stack.» s.f. https://www.elastic.co/es/kibana (último acceso: 10 de abril de 2020).

—. «Logstash, centraliza, transforma y almacena tus datos.» s.f. https://www.elastic.co/es/logstash (último acceso: 10 de abril de 2020).

ElevenPaths. «iBombShell - Dynamic Remote Shell.» *GitHub.* 2020. https://github.com/ElevenPaths/ibombshell (último acceso: 10 de abril de 2020).

EnigmaSoftware. «The Lowdown on Ontario Hospitals Ryuk Ransomware Attacks.» *EnigmaSoftware.* s.f. https://www.enigmasoftware.com/lowdown-ontario-hospitals-ryuk-ransomware-attacks/ (último acceso: 9 de abrl de 2020).

Fortinet. «FortiSIEM.» s.f. https://www.fortinet.com/lat/products/siem/fortisiem.html (último acceso: 10 de abril de 2020).

FreeSoftware Foundation. «GNU Bash.» *GNU.org.* s.f. https://www.gnu.org/software/bash/ (último acceso: 9 de abril de 2020).

FujiFilm. «SYNAPSE Mobility. SYNAPSE PACS on iPad, iPhone or Android.» s.f. https://www.fujifilm.eu/eu/products/medical-systems/solutions/synapse/synapse-mobility (último acceso: 11 de abril de 2020).

G., BRUNEAU. «Using RITA for Threat Analysis.» *isc.SANS.edu.* 2018. https://isc.sans.edu/forums/diary/Using+RITA+for+Threat+Analysis/23926 (último acceso: 11 de abril de 2020).

Gentilkiwi, PowerShellMafia. «Mimikatz.» *GitHub.* 2016. https://github.com/PowerShellMafia/PowerSploit/blob/master/Exfiltration/Invoke-Mimikatz.ps1 (último acceso: 10 de abril de 2020).

George, T. «Feeling the pulse of cyber security in healthcare.» *securityweek.com.* 2018. https://www.securityweek.com/feeling-pulse-cyber-security-healthcare (último acceso: 9 de abril de 2020).

González, G. «macOS Catalina utilizará zsh en lugar de bash como la shell por defecto.» *GenBeta.* s.f. https://www.genbeta.com/desarrollo/macos-catalina-utilizara-zsh-lugar-bash-como-shell-defecto (último acceso: 10 de abril de 2020).

González, P, Hernández, S. «UAC Bypass e investigación con la herramienta UAC-A-Mola.» *ElevenPaths.* 2018. https://www.elevenpaths.com/wp-content/uploads/2018/02/whitepaper-uac-a-mola.pdf (último acceso: 10 de abril de 2020).

González, S. «Torrejón, primer hospital español 'secuestrado' por un virus informático.» *El Mundo.* 2020. https://www.elmundo.es/ciencia-y-salud/salud/2020/01/21/5e274be1fdddffcf088b462d.html (último acceso: 9 de abril de 2020).

Goodwin, B. «Cyber gangsters hit UK medical firm poised for work on coronavirus with Maze ransomware attack.» *Computer Weekly.* 2020. https://www.computerweekly.com/news/252480425/Cyber-gangsters-hit-UK-medical-research-lorganisation-poised-for-work-on-Coronavirus (último acceso: 9 de abril de 2020).

H.K, Huang. *PACS-Based Multimedia Imaging Informatics: Basic Principles and Applications.* Wiley, 2019.

Hasherezade. «pe_to_shellcode.» *GitHub.* s.f. https://github.com/hasherezade/pe_to_shellcode (último acceso: 11 de abril de 2020).

Healthcare IT News. «The biggest healthcare data breaches of 2018 (so far).» *Healthcare IT News.* 2019. https://www.healthcareitnews.com/projects/biggest-healthcare-data-breaches-2018-so-far (último acceso: 9 de abril de 2020).

Horn, R. «DICOM-YARA-rules/DICOM-CVE-2019-11687.yar.» s.f. https://github.com/rjhorniii/DICOM-YARA-rules/blob/master/DICOM-CVE-2019-11687.yar (último acceso: 11 de abril de 2020).

IBM. «IBM QRadar SIEM.» s.f. https://www.ibm.com/es-es/marketplace/ibm-qradar-siem/details (último acceso: 10 de abril de 2020).

Ilascu, I. «DarkUniverse APT Stayed Hidden for 8 Years, Updated Regularly.» *Bleeping Computer.* 2019. https://www.bleepingcomputer.com/news/security/darkuniverse-apt-stayed-hidden-for-8-years-updated-regularly/ (último acceso: 10 de abril de 2020).

INCIBE-CERT. *Diseño y Configuración de IPS, IDS y SIEM en Sistemas de Control Industrial, Anexo 2.3. "Security Onion".* 2017.

InQuest. «Awesome YARA.» *GitHub.* 2020. https://github.com/InQuest/awesome-yara (último acceso: 11 de abril de 2020).

INTERPOL, Cybercrime Threat Response Team. «Cybercriminals targeting critical healthcare institutions with ransomware.» *interpol.int.* 2020. https://www.interpol.int/News-and-Events/News/2020/Cybercriminals-targeting-critical-healthcare-institutions-with-ransomware (último acceso: 9 de abril de 2020).

Isaac. «ATtiny85: un microcontrolador que da mucho juego….» *HW Libre.* s.f. https://www.hwlibre.com/attiny85/ (último acceso: 10 de abril de 2020).

ISACA. *Cybersecurity Glossary.* 2014.

ISECOM. «OSSTMM 3, The Open Source Security Testing Methodology Manual.» s.f. https://www.isecom.org/OSSTMM.3.pdf (último acceso: 10 de abril de 2020).

ISO 12052:2017. *Health Informatics - Digital Imaging and Communication in Medicine (DICOM) including workflow and data management.* s.f. https://www.iso.org/standard/72941.html (último acceso: 9 de abril de 2020).

Kaspersky. *What Is an Advanced Persistent Threat (APT)?* s.f. https://www.kaspersky.com/resource-center/definitions/advanced-persistent-threats (último acceso: 9 de abril de 2020).

Koret, J. y Bachaalany, E. *The antivirus hacker's handbook.* Indianapolis: Wiley, 2015.

Kosmokato. «Bad-DICOM.» *GitHub.* 2019. https://github.com/kosmokato/bad-dicom/ (último acceso: 10 de abril de 2020).

Kowalczyk. «PEfile format.» *log.kowalczyk.info.* s.f. https://blog.kowalczyk.info/articles/pefileformat.html (último acceso: 9 de abril de 2020).

Lab52. «OrangeWorm Group - Kwampirs Analysis Update.» *Lab52.* s.f. https://lab52.io/blog/orangeworm-group-kwampirs-analysis-update/ (último acceso: 9 de abril de 2020).

Lockheed Martin. «The Cyber Kill Chain.» *Lockheed Matin.* s.f. https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html (último acceso: 10 de abril de 2020).

M. Alvarez, Victor. «Yara Rules.» *GitHub.* s.f. https://github.com/Yara-Rules/rules (último acceso: 11 de abril de 2020).

MedDream. «HTML5 Zero-Footprint DICOM viewer.» s.f. https://www.softneta.com/products/meddream-dicom-viewer/ (último acceso: 11 de abril de 2020).

Microsoft. «Enabling Script Block Logging.» *About Logging Windows.* s.f. https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.core/about/about_logging_windows?view=powershell-7 (último acceso: 10 de abril de 2020).

—. «MS-DOS/v2.0/source/COMMAND.ASM.» *GitHub.* 2018. https://github.com/microsoft/MS-DOS/blob/master/v2.0/source/COMMAND.ASM (último acceso: 9 de abril de 2020).

—. «PE Format.» s.f. https://docs.microsoft.com/en-us/windows/win32/debug/pe-format (último acceso: 9 de abil de 2020).

—. «PowerShell Documentation.» *Microsoft Docs.* s.f. https://docs.microsoft.com/en-us/powershell/?view=powershell-7 (último acceso: 10 de abril de 2020).

Microsoft Threat Protection Intelligence Team. «Microsoft works with healthcare organizations to protect from popular ransomware during COVID-19 crisis: Here's what to do.» *Microsoft Threat Intelligence Center (MSTIC).* 2020. https://www.microsoft.com/security/blog/2020/04/01/microsoft-works-with-healthcare-organizations-to-protect-from-popular-ransomware-during-covid-19-crisis-heres-what-to-do/ (último acceso: 9 de abril de 2020).

Miliard, M. «Ransomware attacks Cloud vendor, freezes nursing home EHR data.» *Healthcare IT News.* 2019. https://www.healthcareitnews.com/news/ransomware-attack-cloud-vendor-freezes-nursing-home-ehr-data (último acceso: 9 de abril de 2020).

Mirsky, Y., Mahler, T y Elovici, Y. «CT-GAN: Malicious Tampering of 3D Medical Imagery using Deep Learning.» *USENIX Security.* 2019. https://arxiv.org/abs/1901.03597 (último acceso: 9 de abril de 2020).

MISP. «Open Source Threat Intelligence Platform & Open Standards For Threat Information Sharing.» s.f. https://www.misp-project.org/ (último acceso: 11 de abril de 2020).

MITRE. «MITRE ATT&CK.» *MITRE.* s.f. https://attack.mitre.org/ (último acceso: 10 de abril de 2020).

NATO. *The Tallinn Manual of the International Law Applicable to CyberWarfare.* Edimburgo: Cambridge University Press, 2013.

NCSC (GCHQ). «Advisory: Ryuk ransomware targeting organisations globally.» *ncsc.gov.uk.* 2019. https://www.ncsc.gov.uk/news/ryuk-advisory (último acceso: 9 de abril de 2020).

NEMA (National Electrical Manufacturers Association). *DICOM Standard.* s.f. https://www.dicomstandard.org/about/ (último acceso: 9 de abril de 2020).

Oddvar, Moe. «How to bypass UAC in newer Windows versions.» *2018.* s.f. https://0x00-0x00.github.io/research/2018/10/31/How-to-bypass-UAC-in-newer-Windows-versions.html (último acceso: 10 de abril de 2020).

OISF. «Suricata, Open Source IDS / IPS / NSM engine.» *suricata-ids.org.* s.f. https://suricata-ids.org/ (último acceso: 10 de abril de 2020).

olmedo, Y. «Alternate Data Stream: ADS – Flujo de datos alternativos en NTFS.» *SecurityArtWork.* 2015. https://www.securityartwork.es/2015/02/23/alternate-data-stream-ads-flujo-de-datos-alternativos-en-ntfs/ (último acceso: 10 de abril de 2020).

Orthanc. «Orthanc is a Belgian, open-source, lightweight DICOM server for healthcare and medical research.» s.f. https://www.orthanc-server.com/static.php?page?about (último acceso: 11 de abril de 2020).

Picado, M. «Attacking Digital Imaging and Communication in Medicine (DICOM) file format standard.» *GitHub.* 2019. https://github.com/d00rt/pedicom/blob/master/doc/Attacking_Digital_Imaging_and_Communication_in_Medicine_(DICOM)_file_format_standard_-_Markel_Picado_Ortiz_(d00rt).pdf (último acceso: 10 de abril de 2020).

—. «HIPAA-Protected Malware? Exploiting DICOM Flaw to Embed malware in CT/MRI Imagery.» *Cylera Labs.* 2019. https://labs.cylera.com/2019/04/16/pe-dicom-medical-malware/ (último acceso: 9 de abril de 2020).

PyPi.org. «PeFile.» *PyPi.org.* s.f. https://pypi.org/project/pefile/ (último acceso: 10 de abril de 2020).

Rapid7. «The Metasploit Framework.» *GitHub.* 2019. https://github.com/rapid7/metasploit-framework/wiki/Nightly-Installers (último acceso: 10 de abril de 2020).

Rasta-Mouse. «Amsi Scan Buffer Bypass.» *GitHub.* 2019. https://github.com/rasta-mouse/AmsiScanBufferBypass (último acceso: 10 de abril de 2020).

—. «Watson.» *GitHub.* 2020. https://github.com/rasta-mouse/Watson (último acceso: 10 de abril de 2020).

Raya, A. «Así es el virus informático que ha desconectado al Hospital de Torrejón.» *El Español.* 2020. https://www.elespanol.com/omicrono/software/20200122/virus-informatico-desconectado-hospital-torrejon/461704059_0 (último acceso: 9 de abril de 2020).

ResearchApps. «PyDICOM.» *GitHub.io.* s.f. https://pydicom.github.io/.

Rincón, P. «OrangeWorm: How I met your master….» *Jornadas STIC CCN-CERT.* 2019. https://www.youtube.com/watch?v=mmMXSEExk3Y (último acceso: 9 de abril de 2020).

Russinovich, M. y Garnier, T. «Sysmon v10.42.» *Microsoft SysInternals.* 2019. https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon (último acceso: 11 de abril de 2020).

Seals, T. «Ultrasound hacked in two clicks.» *Rsa conference.* 2019. https://threatpost.com/ultrasound-hacked/142601/ (último acceso: 9 de abril de 2020).

Serna, J. «Some BadUSB attack examples.» *GitHub.* 2018. https://github.com/joelsernamoreno/badusb_examples (último acceso: 10 de abril de 2020).

Shodan, resultados de "DICOM". s.f. https://www.shodan.io/search?query=DICOM (último acceso: 09 de abril de 2020).

Shodan, resultados de "PACS". s.f. https://www.shodan.io/search?query=PACS (último acceso: 9 de abril de 2020).

Siwicki, B. «Cloud-based pacs system cuts imaging costs by half for rural hospital.» *Healthcare IT News.* 2018. https://www.healthcareitnews.com/news/cloud-based-pacs-system-cuts-imaging-costs-half-rural-hospital (último acceso: 11 de abril de 2020).

«Social Engineering Defined - Security Through Education.» *Security Through Education.* s.f. https://www.social-engineer.org/framework/general-discussion/social-engineering-defined/ (último acceso: 10 de abril de 2020).

Softneta. «DICOM library.» *DICOM library.* s.f. https://www.dicomlibrary.com/ (último acceso: 9 de abril de 2020).

Spehn, C. «Living Off The Land Binaries and Scripts (and also Libraries).» *GitHub.* 2018. https://lolbas-project.github.io/ (último acceso: 10 de abril de 2020).

Splunk. «Splunk. Investigate, monitor, alert and act on data.» s.f. https://www.splunk.com/en_us/about-splunk.html (último acceso: 10 de abril de 2020).

StephenFewer. «ReflectiveDLLInjection.» *GitHub.* 2013. https://github.com/stephenfewer/ReflectiveDLLInjection (último acceso: 11 de abril de 2020).

STOLTON, S. «Von der Leyen: Chinese cyberattacks on EU hospitals 'can't be tolerated'.» *EURACTIV.* 22 de junio de 2020. https://www.euractiv.com/section/digital/news/von-der-leyen-chinese-cyberattacks-on-eu-hospitals-cant-be-tolerated/ (último acceso: 25 de junio de 2020).

SwiftOnSecurity. «sysmon-config.» *GitHub.* s.f. https://github.com/SwiftOnSecurity/sysmon-config (último acceso: 11 de abril de 2020).

SwissKyRepo. «Windows - Privilege Escalation.» *GitHub.* 2020. https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Windows%20-%20Privilege%20Escalation.md (último acceso: 10 de abril de 2020).

Symantec. «New Orangeworm attack group targets the healthcare sector in the U.S., Europe, and Asia.» *Symantec-blogs.* 2018. https://symantec-blogs.broadcom.com/blogs/threat-intelligence/orangeworm-targets-healthcare-us-europe-asia (último acceso: 9 de abril de 2020).

The Wireshark team. «Wireshark.» s.f. https://www.wireshark.org* (último acceso: 10 de abril de 2020).

TheWover. «Donut.» *GitHub.* 2020. https://github.com/TheWover/donut (último acceso: 10 de abril de 2020).

Varma, Dandu Ravi. «Managing DICOM images: Tips and tricks for the radiologist.» 2012. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3354356/.

Verizon. *Protected health information data breach report.* 2018. https://enterprise.verizon.com/resources/reports/protected_health_information_data_breach_report.pdf (último acceso: 9 de abril de 2020).

VirusTotal. «Analyze suspicious files and URLs to detect types of malware, automatically share them with the security community.» s.f. https://www.virustotal.com/gui/home (último acceso: 11 de abril de 2020).

—. «Details of 738beee28c2966fc73770b133a7839e2700e05dcf0c180c0e0bfc821cf07ffcd.» s.f. https://www.virustotal.com/gui/file/738beee28c2966fc73770b133a7839e2700e05dcf0c180c0e0bfc821cf07ffcd/details (último acceso: 10 de abril de 2020).

—. «Math module.» *YARA.readTheDocs.* s.f. https://yara.readthedocs.io/en/v3.4.0/modules/math.html (último acceso: 11 de abril de 2020).

VVAA. «Considerations for Exchanging and Sharing Medical Images for Improved Collaboration and Patient Care: HIMSS-SIIM Collaborative White Paper.» 2016. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5023527/pdf/10278_2016_Article_9885.pdf (último acceso: 10 de abril de 2020).

—. «Security Protection of DICOM Medical Images Using Dual-Layer Reversible Watermarking with Tamper Detection Capability.» *Journal of Digital Imaging.* 2011. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3092037/pdf/10278_2010_Article_9295.pdf> <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3092037/ (último acceso: 10 de abril de 2020).

—. «TaHiTi, Threat Hunting Methodology.» 2018. https://www.betaalvereniging.nl/wp-content/uploads/DEF-TaHiTI-Threat-Hunting-Methodology.pdf (último acceso: 10 de abril de 2020).

VVAA. «Dicoogle - an open source peer-to-peer PACS.» 2010. https://www.ncbi.nlm.nih.gov/pubmed/20981467 (último acceso: 11 de abril de 2020).

Wazuh. «Wazuh.» s.f. https://wazuh.com/ (último acceso: 10 de abril de 2020).

Zaw, Nyan Tun. «DICOM: A Ticking Cybersecurity Time-Bomb In The Healthcare Industry.» *Athena Dynamics.* 2017. https://athenadynamics.com/event/dicom-unknown-vulnerability-cyber-attacks-global-healthcare-industry (último acceso: 10 de abril de 2020).

Zeek. «Zeek, An Open Source Network Security Monitoring Tool.» *zeek.org.* s.f. https://zeek.org/ (último acceso: 10 de abril de 2020).