Creación e implementación de un evento de ciberseguridad

Noel Carrasco-García

Resumen– El objetivo de este proyecto ha sido crear un evento orientado a retos y centrado en la ciberseguridad. El enfoque planteado ha ayudado a evaluar los conocimientos sobre seguridad informática de los participantes, a partir de las veces que se ha resuelto cada reto. El número de retos asciende a diez en total, y se han repartido entre diversas áreas de conocimiento: criptografía, ingeniería inversa, forense, explotación binaria y web. Con el fin de ofrecer un entorno de pruebas lo más real posible, cinco de los diez retos se han implementado en contenedores, dando una mayor sensación de sistema real. Una vez finalizados los retos, se puso en marcha el evento donde los participantes compitieron por ver quien conseguía una mayor puntuación. Acabada la competición y analizados los resultados, se ha observado que los estudiantes resolvieron los retos web con facilidad pero encontraron dificultades en los retos sobre criptografía.

Palabras clave– Seguridad informática, ciberseguridad, criptografía, web, contenedores, CTF.

Abstract– The objective of this project was to create a challenge-oriented event focused on cybersecurity. The approach taken helped to evaluate the participants' knowledge on computer security, based on the number of times each challenge was solved. The number of challenges amounted to ten in total, and they were divided between different areas of knowledge: cryptography, reverse engineering, forensics, binary exploitation and web. In order to provide the most realistic testing environment possible, five of the ten challenges have been implemented in containers, giving a greater sense of a real system. Once the challenges were completed, the event was launched where participants competed to see who could achieve the highest score. Once the competition was over and the results were analyzed, it was observed that the students solved the web challenges with ease but encountered difficulties in the cryptography challenges.

Keywords– Computer security, cybersecurity, cryptography, web, containers, CTF.

1 Introducción

En la actualidad, la mayoría de empresas, organizaciones y personas tienen gran parte o la totalidad de su información digitalizada. Esto incluye: datos personales, dinero, conversaciones, imágenes... En el mundo real, se utilizarían cajas fuertes, candados y otros utensilios para proteger toda esta valiosa información, pero en el mundo digital es algo más complicado.

Debido al gran número de vectores de ataque y a la complejidad de algunos, se crea un nuevo perfil profesional, un perfil con un gran conocimiento técnico sobre la seguridad de los sistemas y de las aplicaciones. Este nuevo perfil se encargará, no solo de fortificar los sistemas frente a ataques de ciberdelincuentes, sino también de identificar posibles puntos débiles mediante la simulación de ataques cibernéticos.

En plena era de transformación digital, la necesidad de este nuevo perfil profesional no para de crecer. Pero, ¿cómo se prepara a este nuevo perfil profesional sin incurrir en posibles delitos informáticos? Cuando nació Internet no existía regulación jurídica para el mundo cibernético. Los piratas informáticos y los hackers pululaban por las redes, accediendo a todos los sistemas que tenían a su alcance; en pocas palabras: su plataforma de entrenamiento era Internet en toda su plenitud. Por suerte, hoy en día no es así. Se ha hecho y se sigue haciendo un gran esfuerzo legislativo para regular Internet, marcando donde está el límite de

• E-mail de contacto: noel.carrascoit@gmail.com
• Mención realizada: Tecnologies de la Informació
• Trabajo tutorizado por: Sergio Castillo-Pérez (DEIC)
• Curso 2021/22

lo que se puede hacer y lo que no. La seguridad de los sistemas tampoco es la era hace treinta años, se han puesto muchos más recursos y esfuerzos en proteger a todos los usuarios de Internet, haciendo de la ciberseguridad un tema mucho más serio que antaño. Así pues, con una legislación mucho más restrictiva vetando a los mal denominados «hackers» vagar por sistemas de terceros a placer, es donde nace la iniciativa de los eventos «Capture The Flag» (o CTF para abreviar), unas competiciones realizadas en entornos controlados para que los entusiastas de la ciberseguridad puedan aprender sin cometer ninguna irregularidad.

A grandes rasgos, un CTF es una competición de ciberseguridad creada por y para profesionales o entusiastas del sector, con la finalidad de aprender de manera divertida y competitiva. Estas competiciones son jugadas habitualmente por equipos y consisten en resolver una serie de retos y encontrar una bandera escondida dentro de un fichero o sistema. Cada reto tiene su propia bandera por la que se otorgarán más puntos a mayor dificultad del reto. Para encontrar dicha bandera se deberán superar una serie de retos técnicos que desafían al ingenio y la creatividad de los participantes.

Así pues, este proyecto consiste en el diseño e implementación de un evento CTF enfocado a estudiantes y entusiastas de la ciberseguridad con un nivel de conocimientos bajo o medio. Los retos de cada categoría serán progresivamente más difíciles y estarán alineados con los conocimientos y aptitudes esperados de los estudiantes del grado de Ingeniería Informática.

Durante este articulo conoceremos el punto en el que se encuentran las competiciones de ciberseguridad a día de hoy, la planificación del proyecto y las diferentes modalidades que puede presentar un evento CTF, así como otros tipos de eventos interesantes relacionados con la ciberseguridad. Acto seguido se hablará más concretamente sobre el proyecto, presentando el tipo de modalidad, las categorías involucradas, el número de retos y que se espera de cada sección. Una vez finalizada esta parte, se dará comienzo a la explicación de cada reto de manera más concreta, poniendo en perspectiva la implementación y la resolución del reto. Para acabar, se explicará la puesta en marcha del evento, se presenta el «feedback» recibido por los participantes y finalizaremos con una reflexión sobre qué se ha conseguido y hacia donde marcha esta iniciativa.

2 Objetivos

A continuación se describen los objetivos generales y específicos de este proyecto.

Objetivo General

- Crear una competición en materia ciberseguridad que ayude a fomentar el aprendizaje mediante una serie de retos alineados con las competencias adquiridas durante las diversas asignaturas del grado.

Objetivos específicos

- Diseñar 10 retos divididos en las categorías de: criptografía, ingeniería inversa, explotación binaria, web y forense.
- Crear imágenes de los contenedores de los retos que necesitan un entorno específico para llevarse a cabo.
- Promocionar, organizar y ofrecer soporte a la competición.

Objetivos de aprendizaje

- Conocer y comprender el proceso explotación de vulnerabilidades desde un punto de vista ofensivo.
- Conocer y aprender a identificar distintos fallos de seguridad en aplicaciones y sistemas informáticos.
- Conocer y comprender las medidas mitigatorias más importantes en los sistemas informáticos.
- Desarrollar un pensamiento lateral. Pensar diferente, de manera poco convencional o desde una nueva perspectiva.
- Gestionar el tiempo y los recursos disponibles. Trabajar de forma organizada.

3 Metodología

Para conseguir todos los objetivos y cumplir con los plazos estipulados, ha sido necesario definir una serie de pasos a seguir que se detallan a lo largo de este apartado. La metodología que se ha utilizado se conoce como salto de agua, y en este caso esta formada por 5 fases bien diferenciadas. Para ver las fechas claves del proyecto, vea el diagrama de Gantt en la sección A del apéndice.

3.1. Fase1: Búsqueda de información

Antes de empezar con el diseño de los retos, fue necesario buscar ideas en otras competiciones. En mi caso, mis dos mayores fuentes de inspiración han sido los «writeups» de otros eventos CTF, y el evento evento PicoCTF[1].

El objetivo principal de esta fase ha sido extraer 10 retos con la intención de modificarlos y adaptarlos al nivel que se crea conveniente. Los retos se repartieron de forma más o menos equitativa entre las 5 grandes categorías (criptografía, ingeniería inversa, explotación binaria, web y forense).

3.2. Fase 2: Definición

Una vez recogidas todas las ideas necesarias, se procedió a clasificar los retos dentro de las 5 categorías. Una vez categorizadas, se estipuló el material necesario (necesidad de un contenedor, una imagen, un binario, etc.), un esquema sobre como resolverlo y el plan de implementación para cada uno de ellos.

Los retos que no fueron viables por la falta de infraestructura, por su dificultad de implementación o por la dificultad del reto per se, se simplificaron. Aquellos retos que no fue posible simplificarlos, se eliminaron y se buscó un reto de la misma categoría y de una dificultad similar para reemplazarlo.

3.3. Fase 3: Implementación

Una vez todos los retos se encontraron bien definidos se comenzaron a implementar uno a uno, creando los archivos y/o las imágenes de los contenedores que fueron necesarias.

El orden de implementación fue por categorías, empezando por criptografía, luego ingeniería inversa, forense, «exploiting», y por último web. Una vez acabados los retos, se hicieron pequeños esbozos con las resoluciones para cada reto, detallando paso por paso como se debe resolver. Adicionalmente se ha hecho un pequeño test para comprobar que todo funciona según lo esperado.

3.4. Fase 4: Puesta en marcha

Con todo el material necesario para la ejecución de los retos y para la puesta en marcha, se procedió a subir los retos a la plataforma Google Drive y se creó un enlace de acceso para los participantes. Acto seguido se asignaron los puntos de cada reto.

Llegado el veintisiete de mayo, se abrió la página web al público y se otorgaron 48 horas para la resolución de los retos.

3.5. Fase 5: Análisis y resolución

Cerrada la competición, se analizaron las veces que se resolvió cada reto, así como el perfil de los participantes.

4 Estado del arte

Durante la última década, el número de eventos CTF no ha parado de crecer. Se organizan cientos de eventos cada año, donde empresas y entes gubernamentales buscan reclutar miembros para reforzar sus filas. No obstante, muchos de estos eventos son organizados por la comunidad. A menudo, los mismos equipos que participan en eventos CTFs se juntan para crear el suyo propio, con el único fin de pasar un buen rato aprendiendo. Esto ha incrementado en gran medida el número de CTFs que tienen lugar a lo largo el año, dando más variedad y más oportunidades para participar.

En paralelo, han aparecido los programas de «Bug Bounty». Los «Bug Bounty» son programas ofrecidas por las compañías que tienen como propósito premiar a aquellas personas que logren encontrar fallos y vulnerabilidades en alguno de sus servicios: páginas web, programas, hardware, etc. Actualmente están en pleno auge y cada vez aparecen más plataformas que dan soporte a este tipo de programas.

Los programas de «Bug Bounty» surgieron a raíz de los eventos CTF, y no es de extrañar que guarden cierto parecido con ellos: las compañías ponen a disposición de la comunidad el producto o productos que quieren auditar junto con una serie de normas que hay que seguir (los retos). El participante debe encontrar una vulnerabilidad dentro del sistema que se le ofrece, con el fin de comprometer el sistema y/o los datos sensibles de los usuarios (la bandera). Una vez se haya encontrado un fallo, se debe redactar un informe para que la empresa recree el ataque y aplique las medidas mitigatorias necesarias (el «writeup»). Por último, se recompensa al participante en base a la criticidad de la vulnerabilidad que haya encontrado (puntos y premios). Como se puede apreciar, ambos eventos son en esencia lo mismo pero aplicado a entornos distintos.

Grandes empresas como Microsoft [2], Google [3] y Amazon [4] ya se han sumado a esta iniciativa, y la previsión es que este mercado no pare de crecer. Se cuentan por miles las personas alrededor del mundo que se dedican exclusivamente a participar en este tipo de programas, ya sea como su principal fuente de ingresos o como un sobresueldo ocasional. Así pues, por su similitud con este tipo de programas, los eventos CTF se han consolidado como un muy buen punto de entrada a este mercado.

5 Modalidades de un CTF

A la hora de presentarse o organizar un evento de este tipo es importante fijarse en el nivel que se espera de la competición, si es más para principiantes o para gente con más conocimientos, la infraestructura que se va a utilizar, si el evento es presencial o en línea, etc. Dependiendo de todas estas características nos perfilaremos a organizar o participar en CTF de modalidad «attack and defense» o «jeopardy».

5.1. Attack and defense

La modalidad «Attack and defense» es un formato donde las fases clasificatorias se juegan utilizando un túnel VPN pero la fase final suele ser presencial. En este tipo de eventos se entrega una máquina a cada equipo, quienes serán los encargados de defenderla. Todas las máquinas son iguales, por lo que encontrar una fallo de seguridad en una significa que va a estar presente en todas, incluyendo la propia. Así pues, los equipos deberán atacarse entre ellos sin ser detectados a la vez que monitorizan y arreglan los fallos de seguridad que están presentes en su máquina. Depende del evento se otorga entre treinta minutos y un día para que los equipos se familiaricen con el entorno y sus servicios.

El sistema de puntuaciones para este tipo de modalidad es algo más complejo, y cada organizador lo implementa un poco a su gusto. Por lo general, cada servicio que corre dentro de la máquina tiene una bandera, por la que se otorgan ciertos puntos. Si el equipo defensor consigue mantener la bandera de dicho servicio, se le otorgan los puntos correspondientes, por el contrario, si el atacante roba la bandera, el equipo defensor deja de ganar esos puntos y los pasa a ganar el atacante. La posesión de la bandera se comprueba cada cierto tiempo, que se conoce como ciclo. Cada ciclo estos servicios se refrescan, devolviendo la bandera a su origen y otorgando los puntos a quien corresponda. Además de devolver la bandera, se comprueba que servicios están operativos y cuales no. El equipo que tenga uno o varios servicios caídos al final del ciclo perderá un determinado número de puntos, que serán repetidos entre los equipos que tengan dicho servicio

operativo.

Este tipo de modalidad destaca por ofrecer una situación bastante cercana a lo que sería defender y atacar una organización. Además fomenta el trabajo en equipo, la colaboración y la comunicación entre los atacantes y los defensores. Por otro lado, requiere de un alto grado de competencias ya que se necesitan conocimiento tanto de ataque como de defensa para unos servicios concretos. Algunos de los eventos más importantes que utilizan esta modalidad son: las finales de DEFCON CTF [5], el evento iCTF [6] y RuCTFE [7].

5.2. Jeopardy

La modalidad «jeopardy» es el formato más habitual en las competiciones en línea. En un evento de tipo «jeopardy», los participantes pueden presentarse por equipos o de manera individual, participando en ambos casos sobre los mismos retos y las mismas recompensas.

Por lo general, los retos que se presentan en un CTF de este tipo se dividen en cinco categorías: web, ingeniería inversa, explotación binaria, criptografía y forense. Cada reto otorga al participante una serie de puntos que dependen de la dificultad del reto y del número de veces que se haya resuelto, recompensando con puntos extra a los equipos que consigan resolver el reto rápidamente. Esta puntuación va disminuyendo hasta alcanzar un umbral.

Este tipo de modalidad destaca por ofrecer un abanico muy amplio de posibilidades en cuanto a retos se refiere, abarcando retos de ramas muy distintas dentro de la ciberseguridad, además de fomentar el trabajo en equipo a través de la colaboración.

Debido a la falta de infraestructura y el nivel del público objetivo de este CTF se ha decidido organizar el evento con la modalidad «jeopardy». En la siguiente sección se describirán las diferentes categorías del evento y como se han implementado y como se resuelven cada uno de los retos. Algunos de los eventos con más nombre que utilizan este formato son: el evento CSAW CTF [8], el CTF de Google [1] y la etapa clasificatoria de DEFCON CTF [5].

6 Retos

A continuación se describirán cada uno de los retos diseñados para este CTF según cada una de las categorías: criptografía, ingeniería inversa, análisis forense, explotación binaria y web. A cada reto se le ha dado un nombre que proporciona a los participantes una pequeña pista para su resolución. Asimismo, para cada uno de ellos se proporciona una descripción sobre cómo se han construido e indicaciones para su resolución.

Tal y como se ha descrito anteriormente, estos retos incluirán una bandera a localizar por los participantes, y se empleara un formato codificado como la cadena UAB seguida de un número aleatorio de dígitos. En la figura 1 se puede observar una de éstas banderas.

Los diferentes retos se han subido al repositorio de GitHub en la URL https://github.com/Eiki-sgv/CTF-UAB, dejando estos disponibles para cualquier persona interesada.

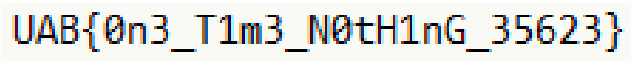

Fig. 1: Bandera del reto OTP

6.1. Criptografía

Los retos criptográficos consisten en descifrar algún fichero de texto, comunicación o archivo. Entre los cifrados más utilizados para este tipo de eventos están: el cifrado César, el cifrado Pigpen y el cifrado Vigenère. Los retos también pueden incluir codificaciones como el código Morse, Base64 y XOR.

La dificultad del reto yace en identificar el tipo codificación que se utiliza para el texto. Se pueden utilizar varias herramientas de código abierto y gratuitas para identificar de qué cifrado se trata. En el caso de este CTF se ha optado por identificar fallos en la implementación de un sistema «one-time-password» que codifica texto utilizando XOR.

6.1.1. OTP

El primer reto y único reto de la categoría de criptografía ha sido OTP. El reto se basa en una aplicación que primero cifra la bandera utilizando un sistema «one-time-password» y se le muestra al participante. Acto seguido, el programa pide una cadena de caracteres, los cifra con el mismo sistema y se imprime por pantalla. Para pasar el reto, el participante deberá revisar el código de la aplicación en busca de fallos en la implementación de este cifrado. El fallo se encuentra en que, pasados cierto número de caracteres, se reutiliza el código utilizado para cifrar la bandera. De esta forma se puede recuperar la llave utilizada para codificar la bandera y utilizarla para descifrarla. Para pasar este reto al participante se le entrega el código fuente de la aplicación, un Dockerfile y un ejecutable para desplegar la aplicación.

Para implementar este reto se ha utilizado Python en su versión 3.9, Docker [9] y la herramienta Socat [10].

6.2. Ingenieria inversa

Los retos que presenta la categoría de ingeniería inversa consisten en desensamblar algún archivo ejecutable (compilado o no). En esta categoría la bandera suele estar escondida dento del fichero ejecutable. En su defecto, desensamblando el fichero pueden encontrar pistas que lleven al participante a la bandera. Los retos pueden incluir codificaciones o cifrados propios de la categoría de criptografía.

La dificultad de esta categoría yace en interpretar el código ensamblador, siguiendo el flujo del programa de manera correcta. Se pueden utilizar varias herramientas de código abierto y gratuitas que ayudan a desensamblar,

tomar notas, cambiar nombres de variables, etc. Entre las más conocidas están IDA [11], radare2 [12] y Ghidra [13].

6.2.1. Intercept

El primer reto de la categoría de ingeniería inversa ha sido Intercept. Fundamentalmente se basa en leer e interpretar correctamente una serie de instrucciones en lenguaje ensamblador que se corresponden a una función de cifrado que utiliza la función XOR. Al participante se le entrega un fragmento del programa desensamblado y la bandera cifrada. El participante deberá crear un programa que le permita revertir el cifrado aplicado a la bandera basándose en lo que puede extraer del fichero que se le proporciona.

Para implementar este reto se ha creado un programa en C que cifra la bandera y de éste se ha extraído el código ensamblador utilizando el compilador x86-64 Clang en su versión 13.0.1 mediante la herramienta Godbolt [14].

6.2.2. Keygen

El reto Keygen es el segundo reto de categoría de ingeniería inversa. En este reto el participante deberá descifrar un fragmento de la bandera cifrado de una manera muy específica. El participante deberá leer, interpretar y seguir el flujo de ejecución del programa para encontrar de qué manera se cifra la bandera y cómo se compone esta. La bandera se divide en tres partes, dos de ellas que son estáticas y se pueden encontrar por el código de la aplicación, y una que se genera de manera dinámica haciendo uso de un hash SHA256 y que se genera cuando la aplicación se pone en marcha. Al tratarse de un hash y pasarle siempre el mismo valor, el código que se genera es siempre el mismo, de esta forma el participante puede revertirlo para recuperar esa parte de la bandera.

Para implementar este reto se ha creado una aplicación en Python que simula una tienda de videojuegos para dar un poco más de realismo al reto. En esta tienda se pueden comprar y vender videojuegos, además de canjear códigos. También existe una opción de validar los códigos que se crean, y es en esta parte donde se encuentra la bandera que el participante busca. Una vez recuperada la bandera se puede verificar si es correcta utilizando esta misma opción que ofrece el programa.

6.3. Forense

Los retos de la categoría forense se basan en indagar mayoritariamente en sistemas de ficheros y capturas de paquetes de red. En esta categoría la bandera está escondida dentro del fichero que se le proporciona al participante. Los retos pueden incluir codificaciones o cifrados, simulando un entorno real donde las comunicaciones y los datos se intentan mantener fuera del alcance de posibles intrusos. La dificultad de los retos está en encontrar algún indicio o pista dentro de un archivo con una gran cantidad datos, ya que lo que se busca puede haber sido borrado u ocultado.

Existen varias herramientas de código abierto y gratuitas que ayudan a buscar de manera efectiva presentando los datos interesantes en primer plano y de manera vistosa. Algunas de las herramientas más conocidas son Wireshark [15] y Autopsy [16], y conviene saber utilizar ambas.

6.3.1. Baby Shark

El primer reto en ser implementado ha sido «Baby Shark», de la categoría forense. El reto consiste en analizar un archivo con extensión pcapng que contiene el tráfico intercambiado entre dos nodos de una red. Al igual que en la resta de retos, el participante deberá buscar la bandera de formato *UAB...*. Esta bandera se encuentra entre los paquetes que han intercambiado ambos nodos, por lo que el concursante deberá revisar los contenidos de los flujos TCP y HTTP. Al participante tan solo se le hace entrega de un fichero de tipo pcapng con la captura del tráfico de la red.

Para implementar este reto se ha utilizado el lenguaje de programación Python y las herramientas de código abierto Curl y Wget. Con la ayuda de Pyhton se ha creado un servidor web con el que recibir peticiones de archivos para posteriormente enviarlos al cliente. Para simular las peticiones de otro nodo se han utilizado las herramientas Curl y Wget. Tanto el cliente como el servidor web se han simulado a partir de la interfaz «loopback», por lo que ambos nodos presentan la dirección IP 127.0.0.1.

6.3.2. Baby Shark 2

El segundo reto en ser implementado ha sido la segunda parte de «Baby Shark», «Baby Shark 2», también de la categoría forense. Como su predecesor, el reto consiste en un archivo pcapng que contiene el tráfico intercambiado entre dos nodos de una red. Sin embargo, esta vez la conexión entre ambos nodos ha sido cifrada mediante el protocolo TLS. El participante deberá buscar entre los paquetes anteriores y posteriores a la conexión TLS para encontrar algo que le ayude a descifrar dichos mensajes y recuperar la bandera. Al participante tan solo se le hace entrega de un fichero de tipo pcapng con la captura del tráfico de la red.

Para implementar este reto se ha utilizado el lenguaje de programación Python y las herramientas de código abierto Curl, Wget y Openssl. Con la ayuda de Pyhton se ha creado un servidor web con el que recibir peticiones de archivos para posteriormente enviarlos al cliente. Este servidor, además de establecer la conexión mediante TLS y servir los archivos de las peticiones que le lleguen, también guarda la «secret key» utilizada en el «handshake» de TLS [17] (ver fig. 2) para posteriormente emitirla en una cabecera HTTP hecha a medida. Previamente en la conexión se han intercambiado mediante HTTP un certificado y la clave privada de la conexión, para hacer más trivial (y posible) el descifrado.

6.3.3. Luks Skywalker

El tercer reto y el mas dificil de la categoría forense es «Luks Skywalker». El reto consiste en un archivo que contiene, en su interior, un sistema de ficheros FAT32 cifrado con LUKS [18]. Para pasar este reto, el participante deberá hacerse con la bandera que se encuentra escondida en el sistema de ficheros. Para ello se deberá aplicar

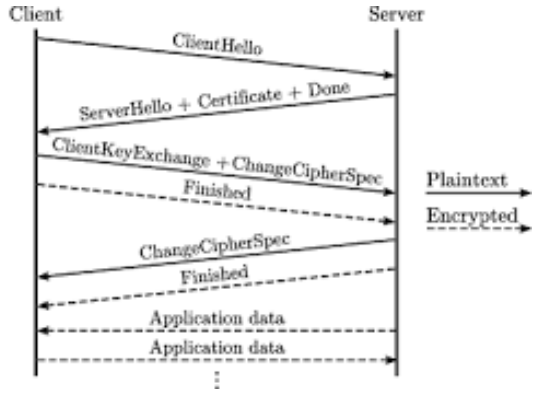

Fig. 2: Handshake de TLS versión 1.3.

fuerza bruta para encontrar la contraseña utilizada para el cifrado que aplica LUKS. Acto seguido se deberá utilizar una herramienta de extracción de datos sobre el sistema de ficheros, con el fin de encontrar datos que hayan sido borrados. Una vez se hayan recuperado los archivos borrados, se deberán aplicar un par de técnicas de estenografía para extraer de la bandera de dichos archivos. Al participante se le hace entrega de un archivo sin formato de un tamaño aproximado de 100MB junto con una pista sobre la contraseña que descifra el archivo.

Para implementar este reto han sido necesarias las herramientas nativas de Linux cryptsetup, mkfs y mount. Adicionalmente se ha utilizado Python junto con algunas librerías externas para llevar a cabo la parte de estenografía.

6.4. Explotación binaria

Los retos de la categoría de explotación binaria son desafíos en los que los participantes tienen que explotar una vulnerabilidad específica, como desbordamientos de «buffer», cadenas de formato, explotación del heap o de pila, etc. con el fin de modificar el flujo de ejecución del programa mientras éste se esta ejecutando. En estos desafíos los participantes obtienen un archivo binario que debe ser depurados para identificar el flujo de ejecución y poder modificarlo. Los binarios suelen ser ejecutables de Windows o Linux. Para llevar a cabo estos retos se pueden utilizar herramientas de código abierto gratuitas como Inmunity Debugger [19] o GDB [20]. También se pueden utilizar librerías de Python como pwntools [21] para hacer «scripts» de explotación binaria automatizados o plugins para GDB hechos por la comunidad como peda [22] para hacer la información más visual y entendible.

6.4.1. Overwrite

El primer reto de esta sección es Overwrite. El reto consiste en desplegar una «shell» dentro del contenedor donde se está ejecutando el binario y recuperar la bandera. Para pasar el reto el participante deberá conectarse al puerto 1337 y provocar un desbordamiento del «buffer» [23] con el fin de modificar el valor de una variable que se encuentra contigua en memoria. De esta forma se debe modificar el flujo del programa para que cumpla una condición dentro de un «if» y se ejecute una «shell». Al participante se le hace entrega del código del programa en C y la imagen del contenedor donde se ejecuta dicho binario.

Para implementar este reto se ha utilizado el compilador gcc en su versión 9.2, Docker [9] y la herramienta Socat [10].

6.4.2. xCentage

El segundo y último reto de la categoría de explotación binaria es xCentage. Igual que en el reto anterior, xCentage consiste en desplegar una «shell» dentro del contenedor donde se ejecuta este binario, con el fin de recuperar la bandera. Para pasar el reto el participante deberá identificar y aprovecharse de una vulnerabilidad conocida como «Uncontrolled format string vulnerability» [24] para filtrar la dirección lógica del «buffer». Con esta dirección y utilizando una técnica de desbordamiento del «buffer» [25] se debe modificar el registro rip para que apunte a la dirección de memoria donde se encuentra el «shellcode» que hemos introducido durante el desbordamiento. Al participante se le hace entrega de una copia del código en C de la aplicación y la imagen del contenedor donde se encuentra la prueba.

Para implementar este reto ha sido necesario el compilador gcc en su versión 9.2, Docker [9] y la herramienta Socat [10].

6.5. Web

Los retos de la categoría web son desafíos en los que los participantes tienen que explotar una vulnerabilidad específica, dentro de un entorno web. Existe un amplio abanico de vulnerabilidades web que pueden estar presentes o no dependiendo de la funcionalidad de la página web, el lenguaje que use o el «framework» que utilice. Para llevar a cabo estos retos se pueden utilizar herramientas de código abierto gratuitas como gobuster [26] o ufuzz [27] para la fase de enumeración, o Burpsuite [28] para la fase de reconocimiento y explotación.

6.5.1. Equality

El primer reto de la categoría Web ha sido Equality. Este reto consiste en hacer una petición POST a la web que se presenta. En esta petición viajará un parámetro llamado «code», en el que se debe encontrar nuestro «payload». Para pasar el reto el participante deberá identificar y aprovecharse de una vulnerabilidad conocida como «PHP Type Juggling» [29], con el objetivo de pasar la condición de un «if» y que de esta manera se muestre la bandera. Para realizar con éxito esta prueba se debe leer con detenimiento cómo se comporta la función «strcmp» de PHP dependiendo del «input» que se le proporciona y qué devuelve ésta. De la misma forma se debe prestar atención a cómo se verifica la condición del «if», ya que en esta no se utiliza una comparación estándar (==) y no una comparación estricta (===). Al participante se le hace entrega de una copia del código PHP de la web, un fichero Dockerfile y un ejecutable para desplegar la el contenedor con la prueba.

Para implementar este reto ha sido necesario el len-

guaje PHP en su versión 7.2, Docker [9] y Apache [30].

6.5.2. Polluted

El segundo reto de la categoría Web ha sido Polluted. Este reto consiste hacer una petición a la web modificando ciertos parámetros para inyectar texto en formato JSON. Para pasar el reto el participante deberá identificar y aprovecharse de una vulnerabilidad conocida como «Prototype Pollution» [31]. Esta vulnerabilidad se basa en modificar el objeto raíz a partir del cual se crean todos los demás objetos. Para ello se debe inyectar en la «Cookie» la siguiente cadena: *__proto__": {"flag": true}*. De esta forma, todos los objetos creados y por crear contendrán el atributo «flag» con valor «true», lo que les permitirá acceder a la bandera. Al participante se le hace entrega de una copia del código en Nodejs de la web, un fichero Dockerfile y un ejecutable para desplegar el contenedor con la prueba.

Para implementar este reto ha sido necesario el lenguaje Nodejs [32] en su versión 16, Docker [9] y la librería Express [33].

7 Resultados

Durante la fase final del proyecto, se llevó a cabo una pequeña campaña de marketing para dar visibilidad al evento entre los estudiantes del grado de ingeniería informática. Se pusieron en circulación un par de mensajes con toda la información del evento que se difundieron por los distintos grupos de mensajería electrónica de la carrera de Ingeniería Informática. En uno de estos mensajes se formulaba una pequeña encuesta para que los participantes decidieran las fechas de la competición. (ver Fig. 3)

Fig. 3: Encuesta de selección de fecha.

En la figura anterior se puede apreciar que cinco de los seis participantes de la encuesta escogieron las fechas del veintisiete de mayo al veintinueve de mayo y tan solo uno del veinte de mayo al veintidós de mayo.

Llegada la fecha del evento, se ponen a disposición de los participantes las normas y los retos de la competición, cada reto con un fichero de texto con las explicaciones necesarias para el despliegue de la prueba. Transcurridas las fechas de la competición se cierra el acceso a los retos y se hace un recuento de puntos. De las seis personas que respondieron la encuesta, participaron activamente dos. En conjunto resolvieron cinco retos distintos y se reportaron siete banderas.

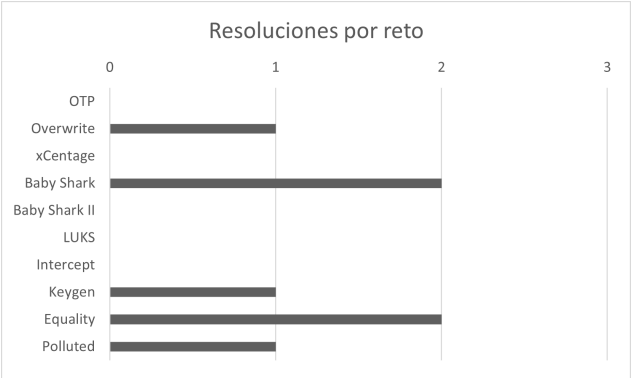

Fig. 4: Número de veces que se ha resuelto cada reto.

Como se puede ver en la figura 4, gran parte de los retos se quedaron sin resolver. Tanto Baby Shark de la categoría Forense, como Equality de la categoría Web, fueron resueltos por ambos participantes. Viendo que el reto Polluted, el reto con mayor puntuación, también fue resuelto por uno de los participantes, se podría interpretar que ambos tenían un muy buen nivel de entendimiento sobre las tecnologías web. A pesar que ambos participantes son estudiantes del grado de ingeniería informática, la baja participación del evento no nos permite extrapolar estos resultados al resto de alumnos. Tampoco se pueden extraer resultados de los retos no resueltos, ya que puede que los participantes no hayan intentado resolverlos.

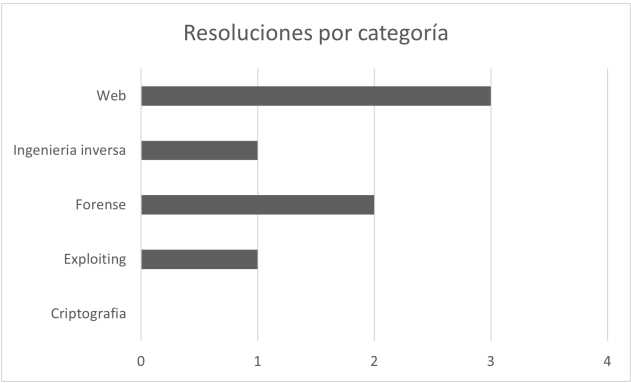

Fig. 5: Número de retos resueltos dentro de cada categoria

Finalmente, observando la figura 5 se puede apreciar que se han resuelto al menos un reto por categoría a excepción de Criptografía, que solo presenta el reto OTP. Tres de los nueve retos restantes presentan algún tipo de relación con la criptografía. Estos retos son: Baby Shark II, Intercept y Keygen. Volviendo a la figura 4, podemos ver que de los cuatro retos criptográficos, Keygen ha sido el único que se ha resuelto. Esto podría significar que los participantes no han sabido identificar problemas en las implementaciones criptográficas en situaciones alejadas del papel, que no han tenido tiempo de intentar todos los retos o que su tiempo han preferido dedicarlo a otros retos.

8 Lineas Futuras

Para complementar el aprendizaje de los estudiantes y aumentar la participación del alumnado, sería razonable

realizar este tipo de competiciones una vez al año. Sería ideal realizarlas en fecha intersemestral una vez acabados los exámenes, de esta forma los alumnos dispondrían de más tiempo libre para participar. Además convendría que los profesores integraran y/o fomentaran la participación en este tipo de eventos extracurriculares, ya que ayudan a desarrollar la curiosidad y la creatividad, además de salir con una mayor formación.

9 Conclusiones

Este proyecto ha puesto en perspectiva las nociones en materia de ciberseguridad con las que todo profesional del sector se encuentra en su día a día. A pesar de la baja participación a causa de la gran carga de trabajo que tienen los estudiantes a final de semestre, se puede observar en la figura 5 que la comprensión de los entornos Web de ambos participantes ha sido buena.

En cuanto a mi desarrollo personal y de aprendizaje, este proyecto me ha servido para saber gestionar mi tiempo y mis recursos a largo y medio plazo, aprendiendo a priorizar aquellas tareas que son más urgentes pero sin dejar de lado las importantes. En la parte técnica, el uso continuado de Docker [9] a lo largo del proyecto me ha ayudado a agilizar el despliegue y mantenimiento de los contenedores, la gran diversidad de retos que he tenido que implementar me han servido para indagar en todas las áreas de ciberseguridad, aprendiendo tanto fallos como medidas mitigatorias para todo tipo de vulnerabilidades.

Todos los objetivos han sido cumplidos: se han diseñado e implementado 10 retos entre diversas categorías, 5 de ellos desplegados en contenedores, todo para finalmente culminar en una competición de ciberseguridad donde el principal objetivo ha sido aprender divirtiéndonos.

Agradecimientos

Me gustaría dar las gracias a mi tutor Sergio Castillo-Pérez por todo el tiempo que ha dedicado a este proyecto revisando los retos, proporcionando ideas y haciendo de guía, sin él este Trabajo de Final de Grado no sería lo que es hoy. También me gustaría agradecer a mi familia y amigos por sus consejos y preocupación durante el transcurso de todo el proyecto, gracias a ellos he sabido actuar con calma y ser organizado en todo momento.

Referencias

[1] “picoCTF - CMU Cybersecurity Competition.” [Online]. Available: https://picoctf.org/

[2] “Microsoft Bounty Programs | MSRC.” [Online]. Available: https://www.microsoft.com/en-us/msrc/bounty

[3] “Home | Google Bug Hunters.” [Online]. Available: https://bughunters.google.com/

[4] “Vulnerability Reporting - Amazon Web Services (AWS).” [Online]. Available: https://aws.amazon.com/security/vulnerability-reporting/

[5] “DEF CON® Hacking Conference - Capture the Flag Archive.” [Online]. Available: https://defcon.org/html/links/dc-ctf.html

[6] “iCTF: the International Capture The Flag Competition.” [Online]. Available: https://shellphish.net/ictf/

[7] “RuCTFE.” [Online]. Available: https://ructfe.org/index/

[8] “CTFtime.org / CSAW CTF Qualification Round.” [Online]. Available: https://ctftime.org/ctf/18/

[9] “Docker.” [Online]. Available: https://www.docker.com/

[10] E. Amoany, “Socat: An advanced sysadmin tool for your toolbox,” publisher: Red Hat, Inc. Section: Enable Sysadmin. [Online]. Available: https://www.redhat.com/sysadmin/getting-started-socat

[11] “IDA Freeware.” [Online]. Available: https://hex-rays.com/ida-free/

[12] “Radare2: The libre unix-like reverse engineering framework.” [Online]. Available: https://www.radare.org/r/

[13] “Ghidra.” [Online]. Available: https://ghidra-sre.org/

[14] M. Godbolt, “Compiler Explorer.” [Online]. Available: https://godbolt.org/

[15] “Wireshark · Go Deep.” [Online]. Available: https://www.wireshark.org/

[16] “Autopsy | Digital Forensics.” [Online]. Available: https://www.autopsy.com/

[17] “Explicación de handshake de tls | Protocolo de enlace SSL.” [Online]. Available: https://www.cloudflare.com/es-es/learning/ssl/what-happens-in-a-tls-handshake/

[18] “Encrypting block devices using LUKS Red Hat Enterprise Linux 8.” [Online]. Available: https://access.redhat.com/documentation/en-us/red-hat-enterprise-linux/8/html/security-hardening/encrypting-block-devices-using-luks-security-hardening

[19] “Immunity Debugger.” [Online]. Available: https://www.immunityinc.com/products/debugger/

[20] “GDB: The GNU Project Debugger.” [Online]. Available: https://sourceware.org/gdb/

[21] “pwntools - CTF toolkit,” Jun. 2022. [Online]. Available: https://github.com/Gallopsled/pwntools

[22] L. Le, “Peda - gdb plugin,” Jun. 2022. [Online]. Available: https://github.com/longld/peda

[23] B. Bierbaumer, J. Kirsch, T. Kittel, A. Francillon, and A. Zarras, "Smashing the Stack Protector for Fun and Profit," in *ICT Systems Security and Privacy Protection*, ser. IFIP Advances in Information and Communication Technology, L. J. Janczewski and M. Kutyłowski, Eds. Cham: Springer International Publishing, 2018, pp. 293–306.

[24] "Format String Software Attack | OWASP Foundation." [Online]. Available: https://owasp.org/www-community/attacks/Format_string_attack

[25] "Buffer Overflow | OWASP Foundation." [Online]. Available: https://owasp.org/www-community/vulnerabilities/Buffer_Overflow

[26] O. J. Reeves, "Gobuster v3.1.0," Jun. 2022. [Online]. Available: https://github.com/OJ/gobuster

[27] Phikshun, "Ufuzz - web fuzzer," May 2021. [Online]. Available: https://github.com/phikshun/ufuzz

[28] "Burp Suite - Application Security Testing Software." [Online]. Available: https://portswigger.net/burp

[29] "PHP: Manipulación de tipos - Manual." [Online]. Available: https://www.php.net/manual/es/language.types.type-juggling.php

[30] "Welcome to The Apache Software Foundation!" [Online]. Available: https://apache.org/

[31] "What is prototype pollution? | Tutorial & examples." [Online]. Available: https://learn.snyk.io/lessons/prototype-pollution/javascript/

[32] Node.js, "Node.js - an asynchronous event-driven javascript runtime." [Online]. Available: https://nodejs.org/en/about/

[33] "Express - Node.js web application framework." [Online]. Available: http://expressjs.com/

APÉNDICE

A METODOLOGÍA

Descripción del hito	Categoría	Progreso	Inicio	Días
Preparación TFG				
Búsqueda de retos	Riesgo medio	100%	27/02/2022	11
Elaboración del informe inicial				
Reunión con el tutor	Hito	100%	19/02/2022	0
Definición de objetivos y abarque del proyecto		100%	27/02/2022	3
Planificación temporal de las tareas		100%	02/03/2022	3
Planteamiento de la metodología a seguir		100%	05/03/2022	3
Redacción del informe inicial		100%	27/02/2022	10

Fig. 6: Metodología fase 1

Informe de seguimiento I				
Clasificación de los retos	Hito	100%	10/03/2022	1
Creación de las fichas técnicas de los retos		100%	11/03/2022	5
Implementación de los retos de criptografía		100%	16/03/2022	13
Implementación de los retos de ingeniería inversa		100%	29/03/2022	13
Implementación de los retos forenses		100%	11/04/2022	13
Implementación de los retos de <<exploiting>>		100%	24/04/2022	13
Implementación de los retos web	Según lo previsto	100%	07/05/2022	20
Redacción del informe de seguimiento (1ª parte)		100%	04/04/2022	7

Fig. 7: Metodología fases 2 y 3

Informe de seguimiento II				
Puesta en marcha de la competición	Según lo previsto	100%	27/05/2022	3
Interpretación de los resultados	Hito	100%	30/05/2022	3
Redacción del informe de seguimiento (2ª parte)	Según lo previsto	100%	15/05/2022	7

Fig. 8: Metodología fase 4

Informe final				
Redacción de los <<writeups>>	Según lo previsto	100%	27/05/2022	14
Redacción del informe final	Según lo previsto	100%	22/05/2022	14

Fig. 9: Metodología fase 5