ENERO DE 2024

MANUAL DE PROYECTO: CTF

TÉCNICAS GENERALES DE ATAQUES II

PROF. CÉSAR ANTONIO RÍOS OLIVARES
WWW.CESARANTONIO.PRO

CĒSAR ANTONIO
— #HAZQUESUCEDA —

Proyecto 1: CTF - Proyecto de Desarrollo de Capture The Flag (CTF) en Python para Ciberseguridad

Objetivo:

El objetivo principal de este proyecto es que los estudiantes adquieran conocimientos prácticos en ciberseguridad a través del desarrollo completo de un Capture The Flag (CTF) desde cero. Al finalizar el proyecto, los estudiantes deberán comprender y aplicar conceptos esenciales de seguridad informática, programación en Python, y otras herramientas relacionadas para la creación de un entorno desafiante y educativo.

Introducción:

Un Capture The Flag (CTF) es un juego en el que los participantes resuelven una serie de desafíos para obtener banderas (flags) que representan la conquista de un objetivo específico. Estos desafíos pueden incluir problemas de criptografía, forense, hacking web, reversing, y más. Este proyecto proporcionará a los estudiantes la oportunidad de diseñar y construir su propio CTF, permitiéndoles explorar diversas áreas de la ciberseguridad.

Requisitos:

1. Conocimientos básicos de programación en Python.
2. Familiaridad con conceptos de seguridad informática, como criptografía, networking, y hacking ético.
3. Acceso a un entorno de desarrollo Python.
4. Acceso a recursos para almacenar y gestionar desafíos y soluciones de CTF.

Requerimientos Técnicos:

a) Desarrollo del Backend:

1. Implementación de un servidor web en Python para alojar los desafíos.
2. Creación de scripts y servicios que actúen como desafíos, abarcando diversas categorías.
3. Configuración de un sistema de autenticación para el acceso a desafíos privados.

b) Diseño de Desafíos:

1. Desarrollo de desafíos encriptados que requieran habilidades de hacking y criptografía.
2. Creación de desafíos de forense que impliquen análisis de archivos y tráfico de red.
3. Implementación de desafíos de reversing para fomentar la comprensión del código binario.

c) Interfaz de Usuario (UI):

1. Diseño de una interfaz web para navegar y seleccionar desafíos.
2. Implementación de un sistema de puntuación en tiempo real para los participantes.
3. Integración de pistas y descripciones para cada desafío.

d) Seguridad del CTF:

1. Aplicación de medidas de seguridad para prevenir explotaciones no deseadas o ataques.
2. Supervisión y registro de actividades en el servidor para análisis forense.

Resultado Esperado:

Al finalizar el proyecto, los estudiantes habrán desarrollado un CTF funcional que abarque diversas áreas de la ciberseguridad. Habrán adquirido habilidades prácticas en programación Python, seguridad informática, y diseño de desafíos realistas.

Entregables:

1. Código fuente del servidor web y desafíos.
2. Documentación detallada sobre la implementación y funcionamiento del CTF.
3. Descripciones y soluciones para cada desafío.
4. Informe de análisis forense de posibles intentos de explotación.

Referencias bibliográficas:

Ciberseguridad:

- Stallings, W., & Brown, L. (2017). Computer Security: Principles and Practice (4th ed.). Pearson.

Desarrollo en Python:

- Downey, A. (2015). Think Python: How to Think Like a Computer Scientist (2nd ed.). O'Reilly Media.

SCRUM:

- Schwaber, K., & Sutherland, J. (2017). The Scrum Guide. Scrum.org.
- Sutherland, J. (2014). Scrum: The Art of Doing Twice the Work in Half the Time. Crown Business.

ANEXOS

A. Ideas para desafíos para el CTF

1. **Desafío de Criptografía: "Mensaje Secreto"**
   - Se proporciona un archivo cifrado con un algoritmo de cifrado simétrico.
   - Los participantes deben descifrar el mensaje utilizando técnicas de criptoanálisis.
   - Pista: La clave de cifrado está relacionada con un evento histórico.

2. **Desafío de Forense: "Rastro de Pistas"**
   - Los participantes reciben un archivo de memoria volátil de un sistema comprometido.
   - Deben analizar el archivo para encontrar pistas sobre cómo se realizó el ataque.
   - Pista: Hay mensajes ocultos en las áreas de la memoria.
3. **Desafío de Web Hacking: "SQL Injection"**
   - Se proporciona una aplicación web vulnerable con una función de búsqueda.
   - Los participantes deben explotar una inyección SQL para recuperar información oculta.
   - Pista: La vulnerabilidad está relacionada con la manipulación de la consulta de búsqueda.
4. **Desafío de Reversing: "Crackea el Código"**
   - Se presenta un ejecutable encriptado que realiza una operación específica.
   - Los participantes deben realizar ingeniería inversa para descubrir cómo funciona y encontrar la entrada correcta.
   - Pista: El algoritmo de encriptación es una variante de XOR.
5. **Desafío de Esteganografía: "Imagen Oculta"**
   - Se proporciona una imagen que contiene información oculta.
   - Los participantes deben utilizar técnicas de esteganografía para revelar el mensaje secreto.
   - Pista: La información está incrustada en los bits menos significativos.
6. **Desafío de Networking: "Captura y Analiza"**
   - Los participantes deben analizar un archivo de captura de red (pcap) para encontrar información crucial.
   - Se oculta una bandera en alguna transmisión de red.
   - Pista: Utiliza herramientas como Wireshark para examinar el tráfico.
7. **Desafío de Lógica: "Secuencia de Números"**
   - Se presenta una secuencia de números y una función desconocida.
   - Los participantes deben descubrir la lógica detrás de la secuencia y predecir el siguiente número.

- Pista: La función sigue una serie matemática simple.

8. **Desafío de Escalada de Privilegios: "Root Exploitation"**
    - Se proporciona un sistema virtual con un usuario limitado.
    - Los participantes deben encontrar una vulnerabilidad para escalar sus privilegios y obtener acceso de root.
    - Pista: Examina los permisos de los archivos y busca servicios ejecutándose con privilegios elevados.

B. Sugerencias para el Desarrollo del Backend para CTF en Python

1. Estructura del Proyecto:

- Crea un directorio principal para tu proyecto CTF.
- Divide el código en módulos para gestionar diferentes aspectos del backend, como autenticación, desafíos, puntuación, etc.

2. Dependencias:

- Utiliza Flask para crear el servidor web.
- Considere el uso de una base de datos (puede ser SQLite para empezar) para almacenar información sobre usuarios, desafíos y puntuaciones.

3. Autenticación:

- Implementa un sistema básico de autenticación de usuarios.
- Los usuarios pueden registrarse, iniciar sesión y cerrar sesión.
- Utiliza sesiones para mantener el estado de autenticación.

4. Desafíos:

- Crea una estructura para almacenar información sobre los desafíos (nombre, descripción, categoría, puntos, etc.).
- Desarrolla endpoints para que los usuarios vean la lista de desafíos disponibles.
- Implementa un mecanismo para que los usuarios soliciten y envíen las soluciones a los desafíos.

5. Puntuación:

- Asigna puntos a cada desafío.
- Lleva un registro de la puntuación de cada usuario.
- Actualiza la puntuación en tiempo real y refleja los cambios en la interfaz de usuario.

6. Interfaz Web:

- Diseña una interfaz web simple utilizando plantillas HTML.
- Proporciona páginas para ver la lista de desafíos, enviar soluciones y ver la puntuación.
- Utiliza AJAX para actualizar dinámicamente la interfaz sin recargar la página.

7. Seguridad:

- Valida y filtra la entrada del usuario para prevenir ataques de inyección (SQL, XSS, etc.).
- Implementa protecciones contra intentos de fuerza bruta en las credenciales de usuario.

8. Documentación:

- Documenta claramente la estructura del proyecto, el propósito de cada módulo y las API expuestas.
- Proporciona instrucciones para la configuración y ejecución del backend.

9. Pruebas:

- Realiza pruebas unitarias para cada módulo del backend.
- Realiza pruebas de integración para asegurarte de que todos los componentes funcionen correctamente juntos.

10. Extras:

- Agrega funcionalidades adicionales como pistas automáticas, temporizadores para ciertos desafíos, etc.

C. Diseño de Interfaz de Usuario para CTF

1. Página de Inicio:

- Bienvenida y descripción del CTF.
- Enlaces para registrar una nueva cuenta o iniciar sesión si ya se tiene una.
- Información básica sobre las reglas del CTF.

2. Panel de Usuario:

- Después de iniciar sesión, los usuarios son dirigidos a un panel personalizado.
- Vista de la puntuación actual y posición en la clasificación.
- Enlace para ver la lista completa de desafíos.

3. Lista de Desafíos:

- Mostrar una lista de desafíos disponibles con detalles como nombre, categoría y puntos.
- Indicadores visuales para indicar el estado de cada desafío (resuelto, no resuelto).
- Botones para solicitar detalles adicionales y para acceder al desafío.

4. **Página del Desafío:**

- Descripción del desafío, incluyendo información sobre la categoría y la puntuación.
- Área para ingresar la solución.
- Botón para enviar la solución.
- Pistas adicionales (pueden desbloquearse después de un tiempo o mediante una acción específica).

5. Página de Puntuación:

- Clasificación general que muestra la puntuación de todos los participantes.
- Destacar el propio lugar en la clasificación.
- Información sobre los últimos participantes que han resuelto desafíos.

6. Perfil del Usuario:

- Información personal del usuario (nombre, correo electrónico, etc.).
- Registro de actividades (desafíos resueltos, cambios en la puntuación, etc.).
- Opciones para cerrar sesión o modificar la cuenta.

7. Estilo y Diseño:

- Usa colores consistentes con el tema de ciberseguridad (tonos oscuros, colores fuertes).
- Utiliza iconos intuitivos para representar acciones y estados.
- Asegúrate de que la interfaz sea fácil de navegar y que las funciones clave estén fácilmente accesibles.

8. Respuestas a Acciones:

- Mensajes de confirmación o error después de enviar una solución.
- Notificaciones para eventos importantes, como la resolución de un desafío o cambios en la clasificación.

9. Extras:

- Considera agregar un temporizador para ciertos desafíos.
- Proporciona enlaces a recursos útiles relacionados con la ciberseguridad.
- Incluye un área de preguntas frecuentes (FAQ) para responder consultas comunes.

D. Seguridad del Capture The Flag (CTF)

1. **Asegurar el Servidor:**

- Actualizaciones y Parches: Mantén el sistema operativo y todas las aplicaciones actualizadas con los últimos parches de seguridad.
- Firewall: Configura un firewall para permitir únicamente el tráfico necesario.
- Limitar Servicios: Desactiva servicios y aplicaciones innecesarios en el servidor para reducir la superficie de ataque.

2. Validación y Filtrado de Entrada:

- SQL Injection y XSS: Implementa medidas para validar y filtrar la entrada del usuario para prevenir ataques de inyección SQL y cross-site scripting (XSS).
- Parámetros de URL: Valida y filtra todos los parámetros de URL para evitar manipulaciones maliciosas.

3. Protección contra Fuerza Bruta:

- Bloqueo de Cuentas: Implementa mecanismos para bloquear cuentas después de varios intentos fallidos de inicio de sesión.
- Limitar Acceso: Limita la cantidad de solicitudes que un usuario puede hacer en un período de tiempo para evitar ataques de fuerza bruta.

4. Almacenamiento Seguro de Contraseñas:

- Hashing y Salting: Almacena las contraseñas de los usuarios de manera segura utilizando funciones hash y salting.
- No almacenar en texto plano: Asegúrate de que las contraseñas no se almacenen en la base de datos en texto plano.

5. Comunicación Segura:

- SSL/TLS: Usa conexiones seguras (HTTPS) para proteger la comunicación entre el cliente y el servidor.
- HSTS: Implementa HTTP Strict Transport Security (HSTS) para forzar el uso de HTTPS.

6. Monitoreo de Actividades:

- Logs: Lleva un registro detallado de las actividades del servidor, incluyendo intentos de inicio de sesión, acceso a desafíos y cambios en la puntuación.
- Alertas de Seguridad: Configura alertas para notificar sobre actividades sospechosas.

7. Controles de Acceso:

- Roles y Permisos: Asigna roles y permisos de manera adecuada para limitar el acceso a funciones críticas.
- Acceso Mínimo Necesario: Garantiza que los usuarios tengan el acceso mínimo necesario para realizar sus funciones.

8. Pruebas de Seguridad:

- Penetration Testing: Realiza pruebas de penetración para identificar y corregir vulnerabilidades.
- Revisión de Código: Realiza revisiones de código para identificar posibles problemas de seguridad en el código fuente.

9. Contenedores (Opcional):

- Docker y Seguridad: Si utilizas contenedores (por ejemplo, Docker), asegúrate de seguir las mejores prácticas de seguridad para contenedores.

10. Backup y Restauración:

- Copias de Seguridad: Realiza copias de seguridad periódicas de la base de datos y otros datos importantes.
- Procedimientos de Restauración: Ten procedimientos claros para restaurar el sistema en caso de una violación de seguridad.

11. Divulgación Responsable:

- Establece un canal de comunicación para divulgación responsable donde los participantes puedan informar sobre posibles problemas de seguridad de manera responsable.

12. Educación y Concientización:

- Educa a los participantes sobre prácticas seguras en ciberseguridad y ética hacker.

E. Implementación de SCRUM para el Proyecto CTF

1. Definir el Producto:

- Identifica y prioriza los requisitos clave del proyecto CTF, como el desarrollo del backend, la interfaz de usuario, la seguridad y otros aspectos fundamentales.

2. Crear el Backlog del Producto:

- Descompón el proyecto en elementos de trabajo más pequeños que se puedan asignar a sprints (iteraciones).
- Backlog de producto podría incluir tareas como "Desarrollo del Backend", "Diseño de la Interfaz de Usuario", "Implementación de la Seguridad", etc.

3. Planificación del Sprint:

- Selecciona un conjunto de elementos del backlog para incluir en el primer sprint.
- Establece metas claras para el sprint, como completar el desarrollo del backend o diseñar la interfaz de usuario.

4. Reuniones Diarias (Daily Scrum):

- Realiza reuniones diarias cortas para mantener al equipo actualizado sobre el progreso.
- Discute cualquier obstáculo o problema que pueda estar afectando el avance del sprint.

5. Desarrollo Iterativo:

- Fomenta el desarrollo iterativo y la entrega continua de funcionalidades.
- Implementa una versión mínima viable (MVP) del sistema y amplía las funcionalidades en sprints posteriores.

6. Revisiones de Sprint:

- Al final de cada sprint, realiza revisiones del trabajo completado con los stakeholders, incluyendo la revisión de desafíos desarrollados, seguridad implementada y la interfaz de usuario.

7. Retrospectivas del Sprint:

- Después de cada sprint, realiza retrospectivas para evaluar el rendimiento del equipo y discutir mejoras para el próximo sprint.

8. Adaptación Continua:

- Adapta el plan de desarrollo y el backlog del producto según sea necesario en función de los cambios en los requisitos o las prioridades.

9. Roles en SCRUM:

- Asigna roles como el Scrum Master, el Product Owner y el equipo de desarrollo.
- El Scrum Master puede ser responsable de supervisar la seguridad del proyecto y garantizar la implementación adecuada de prácticas ágiles.

10. Herramientas de Gestión:

- Utiliza herramientas de gestión de proyectos SCRUM para rastrear el progreso, gestionar el backlog y facilitar la colaboración del equipo.
- La aplicación de SCRUM puede proporcionar una estructura ágil y flexible para el desarrollo del proyecto CTF, permitiendo entregas incrementales y adaptación continua a medida que evolucionan los requisitos y las prioridades. Recuerda que SCRUM se puede adaptar según las necesidades específicas de tu equipo y proyecto.

Ejemplo de cómo podrías organizar el desarrollo del proyecto CTF en un periodo de 6 semanas:

Planificación de SCRUM para el Proyecto CTF (6 Semanas)

Sprint 1 (Semana 1 y 2): Inicio y Planificación Inicial

- Objetivo: Establecer bases y planificación inicial.
- Identificación y priorización de requisitos clave.
- Creación del backlog del producto.
- Planificación de sprints iniciales.
- Inicio del desarrollo del backend básico y configuración del entorno.

Sprint 2 (Semana 3): Desarrollo del Backend Básico

- Objetivo: Implementar funcionalidades básicas del backend.
- Desarrollo del servidor web en Python.
- Implementación de la autenticación y la gestión de usuarios.
- Configuración de la base de datos para almacenar información sobre desafíos y usuarios.

Sprint 3 (Semana 4): Diseño de la Interfaz de Usuario (UI)

- Objetivo: Crear una interfaz de usuario inicial.
- Diseño de la estructura general de la interfaz web.
- Creación de páginas para ver la lista de desafíos y la página del desafío.
- Implementación de la navegación básica.

Sprint 4 (Semana 5): Desarrollo de Desafíos y Puntuación

- Objetivo: Implementar los primeros desafíos y la lógica de puntuación.
- Desarrollo de scripts y servicios para actuar como desafíos.
- Integración de la lógica de puntuación en tiempo real.
- Implementación de la visualización de la puntuación en la interfaz.

Sprint 5 (Semana 6): Refinamiento y Pruebas

- Objetivo: Refinar el sistema y realizar pruebas.
- Revisión y refinamiento de la interfaz de usuario.
- Pruebas de seguridad y corrección de posibles vulnerabilidades.
- Ajustes finales en el diseño y la funcionalidad.

Sprint 6 (Semana 6): Entrega Final y Documentación

- Objetivo: Preparar la entrega final y documentación.
- Realizar la última ronda de pruebas.
- Preparar la documentación completa del proyecto.
- Realizar una revisión final del sistema.

Este es solo un ejemplo y puedes ajustar la duración de los sprints según las necesidades específicas de tu equipo y proyecto. Además, es crucial tener reuniones diarias cortas (Daily Scrum) para mantenerse actualizado sobre el progreso y abordar cualquier problema que surja. La adaptabilidad y la colaboración continua son clave en un entorno SCRUM.