Universidad Zaragoza

Curso Académico: 2021/22

62240 - Explotación de vulnerabilidades en sistemas software

Información del Plan Docente

Año académico: 2021/22
Asignatura: 62240 - Exploiting Software Vulnerabilities
Centro académico: 110 - Escuela de Ingeniería y Arquitectura
Titulación: 534 - Máster Universitario en Ingeniería Informática
Créditos: 3.0
Curso: 2 y 1
Periodo de impartición: Primer semestre
Clase de asignatura: Optativa
Materia:

1. Información Básica

1.1. Objetivos de la asignatura

Al cursar esta asignatura el alumnado será capaz de analizar código y sistemas software para la identificación y solución de las vulnerabilidades y problemas de seguridad más comunes. Así, serán capaces de aplicar diversas técnicas para analizar la seguridad y comprometer los sistemas software que sean vulnerables, demostrando fehacientemente la problemática existente y proponiendo soluciones de mejora.

Estos planteamientos y objetivos están alineados con los siguientes Objetivos de Desarrollo Sostenible (ODS) de la Agenda 2030 de Naciones Unidas (https://www.un.org/sustainabledevelopment/es/), de tal manera que la adquisición de los resultados de aprendizaje de la asignatura proporciona capacitación y competencia para contribuir en cierta medida a su logro. En concreto, se encuentran alineados con los siguientes objetivos:

- Objetivo 9.1: Desarrollar infraestructuras fiables, sostenibles, resilientes y de calidad, incluidas infraestructuras regionales y transfronterizas, para apoyar el desarrollo económico y el bienestar humano, haciendo especial hincapié en el acceso asequible y equitativo para todos.
- Objetivo 11.2: De aquí a 2030, proporcionar acceso a sistemas de transporte seguros, asequibles, accesibles y sostenibles para todos y mejorar la seguridad vial, en particular mediante la ampliación del transporte público, prestando especial atención a las necesidades de las personas en situación de vulnerabilidad, las mujeres, los niños, las personas con discapacidad y las personas de edad.

1.2. Contexto y sentido de la asignatura en la titulación

El desarrollo de software es uno de los pilares fundamentales del desarrollo de cualquier producto industrial porque o bien forma parte del producto, o bien forma parte del desarrollo del producto, o bien por ambos. En este desarrollo, cada vez más se trabaja en equipos multidisciplinares, con gente proveniente de diferentes disciplinas y con diferentes capacidades. Sin embargo, muchos de estos desarrollos tienen defectos en su forma, tanto a nivel de diseño como a nivel de implementación. Estos defectos, que pueden derivar en vulnerabilidades, pueden llegar a explotarse comprometiendo así la seguridad del sistema (atentando contra algunas de sus propiedades de confidencialidad, integridad y disponibilidad). Cada vez es más frecuente la publicación de vulnerabilidades en software ampliamente utilizado, como OpenSSL (vulnerabilidades POODLE y Heartbleed, entre otras) o la consola de Unix Bash (vulnerabilidad Shellshock, entre otras).

Para minimizar estos defectos potencialmente explotables en sistemas software, conviene conocer las vulnerabilidades más frecuentes introducidas durante el desarrollo de código, así como los mecanismos para evitarlas. En este curso, además, se introducirán los conceptos subyacentes detrás de las vulnerabilidades, los ataques posibles y las defensas para evitar su explotación, así como algunas técnicas avanzadas para el análisis de software y explotación automáticas. Por último, se introducirá una metodología para construcción de códigos de explotación de vulnerabilidades.

1.3. Recomendaciones para cursar la asignatura

Conocimientos de programación y de arquitectura de computadores a nivel de un graduado en Informática.

2. Competencias y resultados de aprendizaje

2.1. Competencias

- CB-06 - Poseer y comprender conocimientos que aporten una base u oportunidad de ser originales en el desarrollo y/o aplicación de ideas, a menudo en un contexto de investigación.
- CB-09 - Que los estudiantes sepan comunicar sus conclusiones y los conocimientos y razones últimas que las sustentan a públicos especializados y no especializados de un modo claro y sin ambigüedades.
- CB-10 - Que los estudiantes posean las habilidades de aprendizaje que les permitan continuar estudiando de un modo que habrá de ser en gran medida autodirigido o autónomo.
- CG-09 - Capacidad para comprender y aplicar la responsabilidad ética, la legislación y la deontología profesional de la actividad de la profesión de Ingeniero en Informática.
- CG-11 - Capacidad para adquirir conocimientos avanzados y demostrado, en un contexto de investigación científica y tecnológica o altamente especializado, una comprensión detallada y fundamentada de los aspectos teóricos y prácticos y de tecnológica o altamente especializado, una comprensión detallada y fundamentada de los aspectos teóricos y prácticos y dela metodología de trabajo en uno o más campos de estudio.
- CG-13 - Capacidad para evaluar y seleccionar la teoría científica adecuada y la metodología precisa de sus campos de estudio para formular juicios a partir de información incompleta o limitada incluyendo, cuando sea preciso y pertinente, una reflexión sobre la responsabilidad social o ética ligada a la solución que se proponga en cada caso

Conseguir adquirir las siguientes competencias específicas:

- CTI-01 - Capacidad para modelar, diseñar, definir la arquitectura, implantar, gestionar, operar, administrar y mantener aplicaciones, redes, sistemas, servicios y contenidos informáticos.
- CTI-02 - Capacidad para comprender y saber aplicar el funcionamiento y organización de Internet, las tecnologías y protocolos de redes de nueva generación, los modelos de componentes, software intermediario y servicios.
- CTI-03 - Capacidad para asegurar, gestionar, auditar y certificar la calidad de los desarrollos, procesos, sistemas, servicios, aplicaciones y productos informáticos.
- CTI-04 - Capacidad para diseñar, desarrollar, gestionar y evaluar mecanismos de certificación y garantía de seguridad en el tratamiento y acceso a la información en un sistema de procesamiento local o distribuido.

2.2. Resultados de aprendizaje

Cada estudiante deberá ser capaz de:

- RA1: Reconocer las vulnerabilidades más comunes en sistemas software.
- RA2: Evaluar la seguridad de un sistema software.
- RA3: Dominar diferentes técnicas de análisis de sistemas software.
- RA4: Crear pruebas de concepto que permitan comprometer la seguridad de sistemas software vulnerables.

2.3. Importancia de los resultados de aprendizaje

Los egresados serán capaces de reconocer las vulnerabilidades más comunes en sistemas software, además de proponer soluciones de mejora para evitarlas o, incluso, de desarrollar pruebas de concepto que permitan una explotación de la vulnerabilidad.

A día de hoy, la detección y explotación de vulnerabilidades es un campo en auge en el ecosistema de desarrollo software, siendo numerosas las empresas que ofrecen recompensas de diversa índole a aquellos que mejoran la seguridad de sus productos. Adicionalmente, el perfil de expertos analistas de código y de búsqueda de vulnerabilidades es muy demandado por las empresas tecnológicas, tanto en el campo defensivo (*blue team*) como en el campo ofensivo (*red team*). Entendemos que conocer estas vulnerabilidades y sus principios subyacentes, así como técnicas de defensa, habilita a los egresados para diseñar e implementar sistemas de una manera más segura. Todo esto aumenta además su empleabilidad en el mercado laboral.

3. Evaluación

3.1. Tipo de pruebas y su valor sobre la nota final y criterios de evaluación para cada prueba

El estudiante deberá demostrar que ha alcanzado los resultados de aprendizaje previstos mediante las siguientes actividades de evaluación:

Se plantearán enunciados prácticos de análisis de programas vulnerables que deberán de ser resueltos en el laboratorio. Estos trabajos se calificarán con una nota cuantitativa de 0 a 10. Se valorará especialmente una correcta explicación y desarrollo del análisis realizado, fundamentado en los conceptos estudiados en la asignatura.

La presentación y defensa de trabajos prácticos de programación se valorará con una calificación de prácticas que ponderará con un 70% de la nota final de la asignatura. Con esta prueba se evaluarán los resultados de aprendizaje RA1, RA2, RA4.

Por último, se realizará una prueba final de evaluación, consistente en una presentación de trabajos en grupo, que servirá para demostrar que se ha logrado alcanzar los resultados de aprendizaje requeridos en la asignatura. En esta prueba se resolverán problemas de naturaleza similar a los planteados en clase (análisis de código y realización de prueba de concepto). La calificación obtenida en esta prueba ponderará un 30% de la nota final de la asignatura. Con esta prueba se evaluarán los resultados de aprendizaje RA1, RA2, RA3, y RA4.

El estudiante que no opte por el procedimiento de evaluación descrito anteriormente, o bien no supere dichas pruebas durante el periodo docente, o bien quisiera mejorar su calificación, tendrá derecho a realizar una prueba global que será programada dentro del periodo de exámenes correspondiente a la primera o segunda convocatoria.

4. Metodología, actividades de aprendizaje, programa y recursos

4.1. Presentación metodológica general

El proceso de aprendizaje que se ha diseñado para esta asignatura se basa en lo siguiente:

- Clase de Teoría. Exposición de contenidos mediante presentación o explicación por parte de un profesor (posiblemente incluyendo demostraciones).
- Laboratorio Actividades desarrolladas en espacios especiales con equipamiento especializado (laboratorio, aulas informáticas, visita a obra o a lugares de interés arquitectónico).
- Presentación de trabajos en grupo. Exposición de ejercicios asignados a un grupo de estudiantes que necesita trabajo cooperativo para su conclusión
- Trabajos prácticos. Preparación de actividades para exponer o entregar en las clases prácticas o en la etapa de evaluación.

4.2. Actividades de aprendizaje

La dedicación del estudiante para alcanzar los resultados de aprendizaje en esta asignatura se estima en 75 horas distribuidas del siguiente modo:

- 26 horas, aproximadamente, de actividades presenciales: clases magistrales y de resolución de problemas y casos y clases prácticas de laboratorio.
- 30 horas, aproximadamente, de realización de trabajos de aplicación o investigación prácticos.
- 5 horas, aproximadamente, de tutela personalizada profesor-alumno.
- 10 horas, aproximadamente, de estudio personalde teoría.
- 4 horas, aproximadamente, de las pruebas de evaluación.

4.3. Programa

- Introducción: Gestión de vulnerabilidades, tipos de vulnerabilidades, herramientas y laboratorio de análisis. Cuestiones éticas
- Técnicas de análisis de aplicaciones: análisis estático, análisis dinámico. Fuzzing
- Vulnerabilidades software y técnicas de explotación: vulnerabilidades de corrupción de memoria (en heap, en pila), de enteros, de cadenas de formato, problemas de concurrencia
- Técnicas de defensa software
- Técnicas de explotación avanzadas. Ataques ROP, diseño de shellcodes personalizadas

4.4. Planificación de las actividades de aprendizaje y calendario de fechas clave

La organización docente de la asignatura prevista es la siguiente:

- Clases magistrales y de resolución de problemas y casos
- Clases prácticas de laboratorio. Son sesiones de trabajo en laboratorio, tuteladas por un profesor, en las que participan el estudiantado en grupos reducidos.

Los horarios de todas las clases y fechas de las sesiones de prácticas se anunciarán con suficiente antelación a través delas webs del centro y de la asignatura.

El calendario de clases, prácticas y exámenes, así como las fechas de entrega de trabajos de evaluación, se anunciará con suficiente antelación.

4.5. Bibliografía y recursos recomendados

La bibliografía recomendada para esta asignatura puede consultarse en este enlace de la Biblioteca de la Universidad de Zaragoza.