



Presentación

Seguridad en el Diseño del Software

Datos

Profesores

Rafael Álvarez [teoría]
(ralvarez@dccia.ua.es)

Tutorías
lunes de 11 a 15

(politécnica II, planta superior, columnas verdes)
*confirmar cita por email o CV (UACloud...)

José Vicente Aguirre [prácticas]
(jaguirre@ua.es)

Tutorías
(consultarle a él)

(politécnica III, planta superior, pasillo DCCIA)
*confirmar cita por email o CV (UACloud...)

Asignatura

- Materiales en CV
- Teoría (3 créditos)
 - martes de 11 a 13h
A2/D27
- Prácticas (3 créditos)
 - martes de 9 a 11h
L15 (Pol. I)
 - martes de 13 a 15h
L15 (Pol. I)

Contextualización

Contexto

La seguridad es un *proceso continuo* dentro de la ingeniería del software que requiere del *conocimiento y aplicación* de las *técnicas apropiadas* para evitar los posibles *defectos y vulnerabilidades*.

El diseño de software seguro es, cada vez, más importante dada la creciente complejidad de los sistemas software actuales.

Objetivos específicos:

- Aplicar criterios de seguridad en el desarrollo del software.
- Evaluar y analizar software para determinar sus características de seguridad.
- Conocer y aplicar las herramientas, técnicas y construcciones de seguridad adecuadas.
- Conocer y evitar las vulnerabilidades y ataques más comunes.

Desarrollo

Temario

1. Introducción
2. Privacidad y seguridad de la información
3. Diseño de software seguro
4. Análisis de casos reales

Evaluación

- Evaluación continua:
 - 50% teoría
 - 50% prácticas
- Necesario sacar un 4 para promediar.
- 2ª convocatoria:
 - Una única prueba para el apartado de teoría
 - Ejercicios prácticos que se propongan
- Dado el carácter continuo en la evaluación de la asignatura, la asistencia será obligatoria.

Las 10 leyes de la seguridad de la información

Ley 1: Seguridad en el cliente

- ¿Qué es el cliente?
 - La máquina a la que tiene acceso el usuario (o atacante):
 - Arquitectura cliente/servidor.
 - Sistemas sin red.
- ¿Qué es la seguridad en el cliente?
 - Cualquier tipo de mecanismo de seguridad que se aplica únicamente en el cliente:
 - Protocolo en C/S.
 - Software DRM.
 - Etc.

Ley 1: Seguridad en el cliente

- PROBLEMA: el usuario tiene control absoluto del cliente.
 - El atacante acabará por romper el sistema.
 - Como mucho se puede retrasar / dificultar el proceso.
 - Se agrava con software o hardware producido en masa.
- Ejemplos:
 - DRM.
 - Anticopia.
 - Detección de modificación.
 - Ocultación de información.

Ley 1: Seguridad en el cliente

La seguridad exclusivamente en el cliente NO funciona

Ley 2: Intercambio de claves

- ¿Con quién intercambiamos la clave?
 - Un atacante puede suplantar la IP de destino:
 - Autoridades certificadoras.
 - Autenticación SSL.
 - Una clave compartida previamente.
- Dudas:
 - ¿Cómo llegan las claves a donde se necesitan?
 - ¿Es posible un ataque Man-in-the-middle?
 - ¿Hay alguna 3ª parte de confianza?
 - ¿Puede ser atacada esta 3ª parte de confianza?

Ley 2: Intercambio de claves

- Ejemplos:
 - Routers / VPNs.
 - MITM.
 - Ataque físico.
 - SSL / HTTPS.
 - Certificados no firmados.

**El intercambio de claves
necesita de información
compartida previamente**

Ley 3: Código malicioso

- Malware:

- Programas dañinos:
 - Virus
 - Troyanos
 - Gusanos
 - Etc.
- Cambian con gran velocidad.
- Se evitan con antivirus.

- Antivirus:

- Funcionan mediante detección de firmas o heurísticas de comportamiento.
- Desarrollo de firmas muy lento.
- Los antivirus NO protegen en otros casos.

Ley 3: Código malicioso

- Casos especiales:
 - Malware nuevo.
 - Malware hecho a medida.
 - Falsa sensación de seguridad.
- Solución:
 - Actualizaciones.
 - Comportamientos.
 - La seguridad absoluta no es práctica.

Es imposible protegerse
al 100% del código malicioso

Ley 4: Detección de malware

- Detección por firmas:
 - Búsqueda de ciertas partes de código.
 - Las variaciones no son detectadas.
- Mutaciones muy rápidas:
 - Otros autores / competitividad.
 - Compresión / cifrado.
 - Motores de mutación.

Cualquier código puede ser transformado para evitar la detección

Ley 5: Firewalls

- Protegen contra ciertos ataques y proporcionan *logs*.
- No todo pasa por el firewall:
 - Empleados maliciosos.
 - Seguridad física.
 - VPN.
 - WiFi sin control.
 - USB con virus.
 - Etc.
- Permiten cierto tipo de tráfico rechazando lo demás.
 - Se ha de seguir una política en su configuración.
- Tipos de protección:
 - Bloqueo de puertos.
 - Filtrado de protocolos.
 - Antivirus.
 - Bloqueo de ficheros.

Ley 5: Firewalls

- ¿Es seguro?
 - Renombrado de ficheros.
 - Compresión y cifrado.
 - Ataque a sitios autorizados.
- Los fabricantes siempre van por detrás respecto a los ataques.
- Es muy difícil precisar la configuración:
 - Tunneling.
- Ingeniería social.
 - Email (I love you).
 - Suplantación.
 - Virus/Troyanos.
- Servidores expuestos.
 - DMZ con acceso interno.
 - Seguridad interna reducida.

Ley 5: Firewalls

- Ataque directo al firewall:
 - Vulnerabilidades.
 - Problemas de configuración
 - No actualizado.
- Agujeros en el cliente:
 - Navegadores web.
 - Chat.
 - Email.
 - FTP, Telnet, SSH, etc.

Los firewalls no protegen
completamente

Ley 6: Detección de intrusismo

- ¿Qué es un IDS?
 - Detectan ataques o tráfico no adecuado.
 - Comparan con firmas.
 - Comparan con estándares.
 - Contrastes estadísticos.
 - Pueden cooperar con firewalls.
- Problemas:
 - Hay usuarios por detrás.
 - Mantenimiento inadecuado.
- Ventajas:
 - Pasivos.
 - Alarma silenciosa.

Ley 6: Detección de intrusismo

- Es necesario vigilar los logs.
- El atacante posee los mismos conocimientos:
 - Puede modificar el código para no ser detectado.
 - Puede restringirse a aquellas cosas permitidas.

Cualquier IDS se puede evadir

Ley 7: Criptosistemas secretos

- Cualquiera puede diseñar un criptosistema y no ser capaz de romperlo:
 - Programadores no son buenos beta testers.
 - Escritores no son buenos correctores.
- Para producir un criptosistema seguro se debe:
 - Conocer todos los posibles ataques.
 - Saber cuándo son aplicables a su algoritmo.
 - Ser capaz de predecir futuros ataques.
- Ejemplo: S-Boxes en DES.

Ley 7: Criptosistemas secretos

- La mayoría de los criptosistemas fracasan:
 - La criptografía es difícil.
 - Falta de revisión.
 - Que no se haya roto NO significa que sea seguro.
 - Los algoritmos novedosos requieren ataques novedosos.
- Un autor puede:
 - Publicar sus resultados.
 - Usarlos de forma privada.
 - Crear un producto comercial.
- Autores noveles:
 - Mucha ingenuidad.
 - Optimismo generalizado.

Ley 7: Criptosistemas secretos

Todo algoritmo que no haya sido sometido a revisión escrupulosa merece una gran sospecha.

Ley 8: Sin clave

- Crear criptosistemas seguros es difícil.
 - Existen pocos y se reutilizan para muchas cosas.
 - Si el mejor ataque es fuerza bruta y es suficientemente seguro, ¿por qué cambiar?
- ¿Un sistema TDES sin necesidad de memorizar claves?
 - ¿qué clave utiliza?
 - ¿cómo se obtiene?
 - ¿dónde se guarda?
- Criptografía clásica es sin clave.
- Los algoritmos difícilmente se mantienen secretos.
 - Basta el algoritmo para descifrar si no hay clave.

Ley 8: Sin clave

Sin clave no se tiene ~~criptografía~~ lo que se tiene es
codificación.

Ley 9: Passwords locales

- Algunos programas ofrecen “recordar” tus passwords.
 - ¿Cómo lo guardan?
 - ¿Es seguro?
 - ¿Usan clave?
 - ¿Es buena idea?
- Ejemplos:
 - Cliente de email:
 - Ataque por *sniffing*.
 - Ataque por robo.
 - Sistema de ficheros cifrado.

Ley 9: Password locales

No es seguro almacenar password de forma local salvo que se protejan con otra clave.

Ley 10: Auditoría externa

- Prueba funcional:
 - Comprobar que el programa realiza lo que debe.
- Prueba de seguridad:
 - Comprobar que el programa realiza lo que debe de *forma segura*.
- Dificultad de defensa:
 - Un usuario casual.
 - Un hacker de élite.
- ¿Por qué ha de ser externa?
- Productos comerciales:
 - ¿Cuántos son sometidos a auditoría de seguridad?
 - ¿Es sólo importante en sistemas seguros?
 - La presión del mercado obliga a producir rápido.
 - Los derechos de propiedad intelectual impiden una auditoría externa.
 - Los errores se repiten una y otra vez.

Ley 10: Auditoría externa

- Ejemplo OpenBSD:
 - No comercial.
 - Énfasis en auditoria de código y seguridad.
 - Es considerado el sistema más seguro en la actualidad.
 - La auditoria de seguridad engloba la funcional.

Para que un sistema pueda comenzar a ser considerado seguro, ha de satisfacer una auditoria externa