Security for Developers

Recomendaciones

Federico Pacheco

 @FedeQuark

 www.federicopacheco.com.ar

 info@federicopacheco.com.ar

Recomendaciones

- Probar si el software hace lo que NO debe hacer
- Probar si el software hace lo que debe hacer sin provocar efectos secundarios
- Cada defecto aumenta la probabilidad de descubrir defectos nuevos
- Diseñar las pruebas para detectar los máximos defectos con el mínimo esfuerzo
- Aplicar la creatividad en el testing de seguridad
- Priorizar la búsqueda y resolución de lo más importante

Recomendaciones - El rol del desarrollador

- Puede contribuir con la seguridad de distintas formas
  - Concientización propia y de otros
  - Asumir responsabilidad de asegurar su código
  - Adoptar procesos de desarrollo seguro
  - Adoptar técnicas de programación seguras y saber cómo generar código seguro
  - Conocer donde se producen las vulnerabilidades, como detectarlas y evitarlas
  - No testearse a sí mismo

- Entrenamiento en Seguridad
  - Mantenerse actualizado

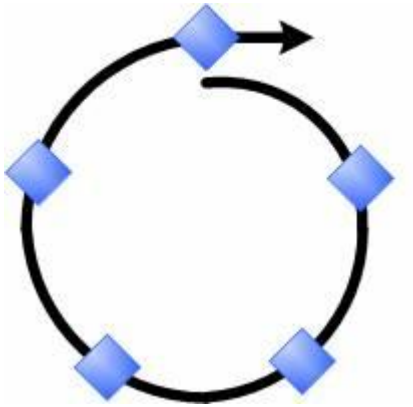
- Desarrollo de nuevas competencias
  - Capacidad de integración y trabajo en equipos multidisciplinarios
  - Pensar como un atacante
  - Revisión de código

Recomendaciones - Otros roles

- Gerente de Proyectos
  - Considerar la seguridad desde el inicio del proyecto
  - Incluir seguridad en todas las instancias posibles del SDLC
- Cliente
  - Solicitar requerimientos de seguridad y pedir que los productos sean seguros
- Empresa de desarrollo
  - Utilizar métricas de seguridad
  - Entrenar al personal en seguridad
  - Ejecutar revisiones y evaluaciones independientes
  - Aplicar procesos de ingeniería de software

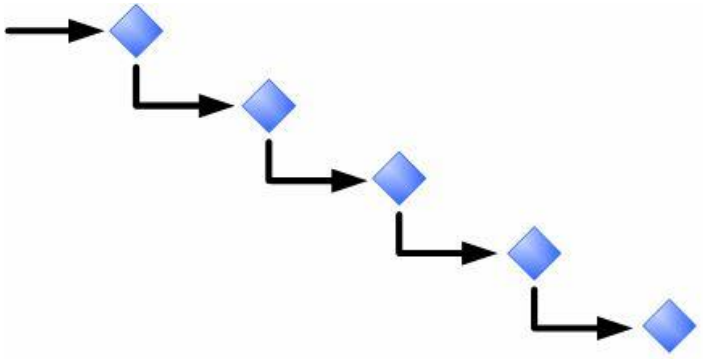
Inclusión de la seguridad en el SDLC

- Al inicio del proceso
  - Análisis de riesgo
  - Modelado de amenazas
  - Prácticas de diseño seguro



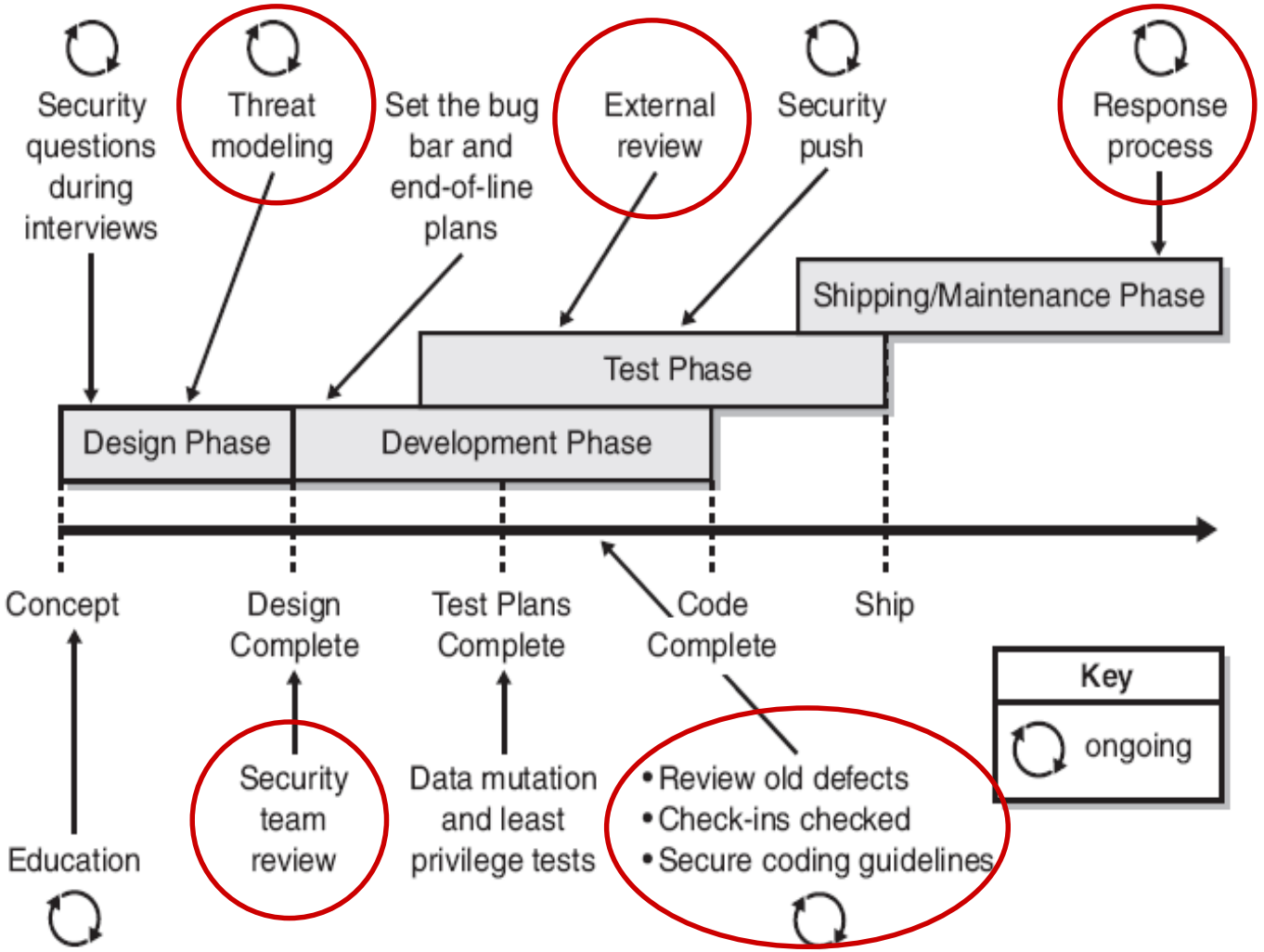
- Al momento de escribir código
  - Prácticas de programación segura
  - Auditoría de código fuente (white box)

- Al momento de realizar el deployment
  - Penetration testing (black box)

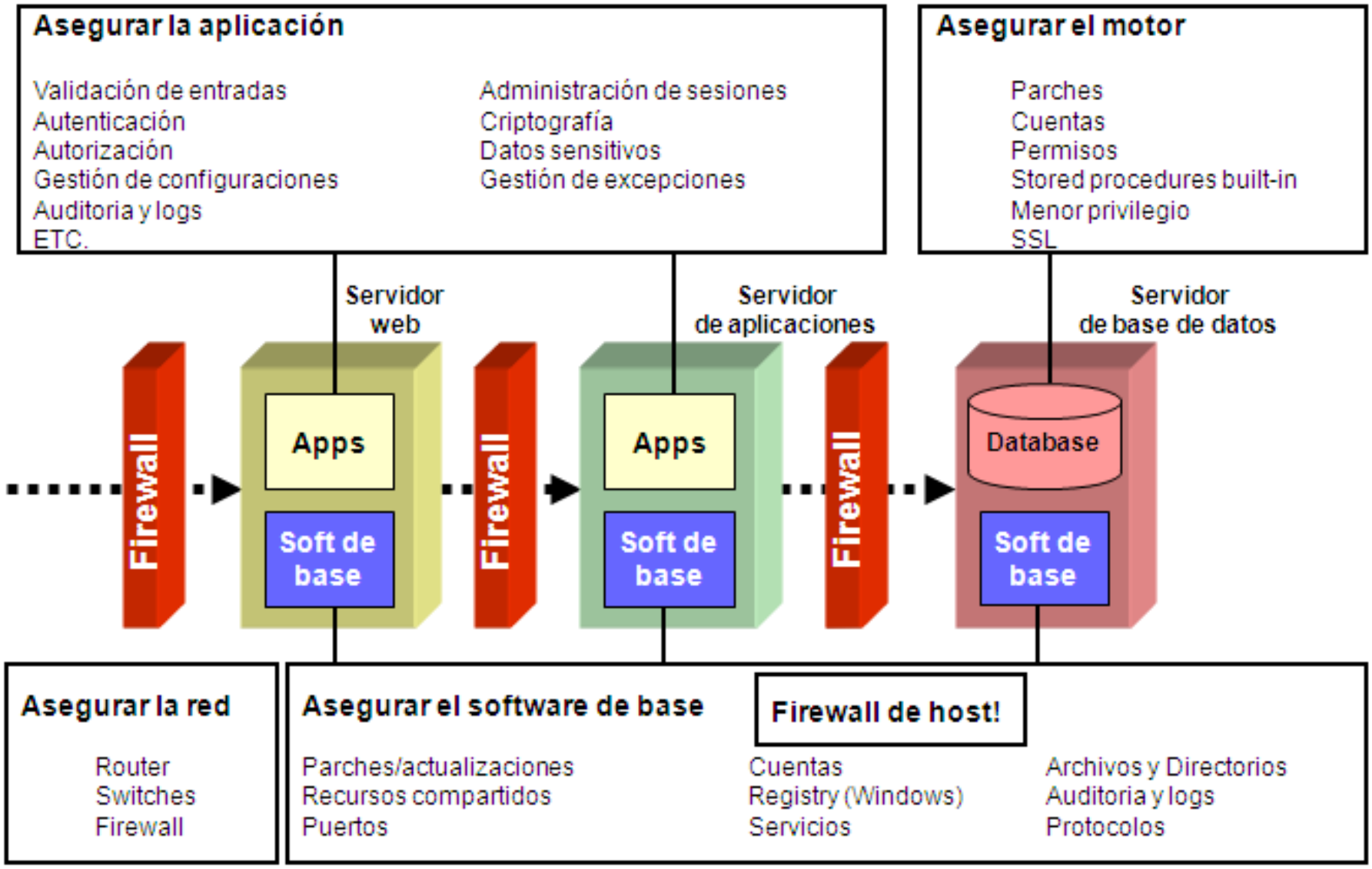


- A lo largo de la vida útil del software

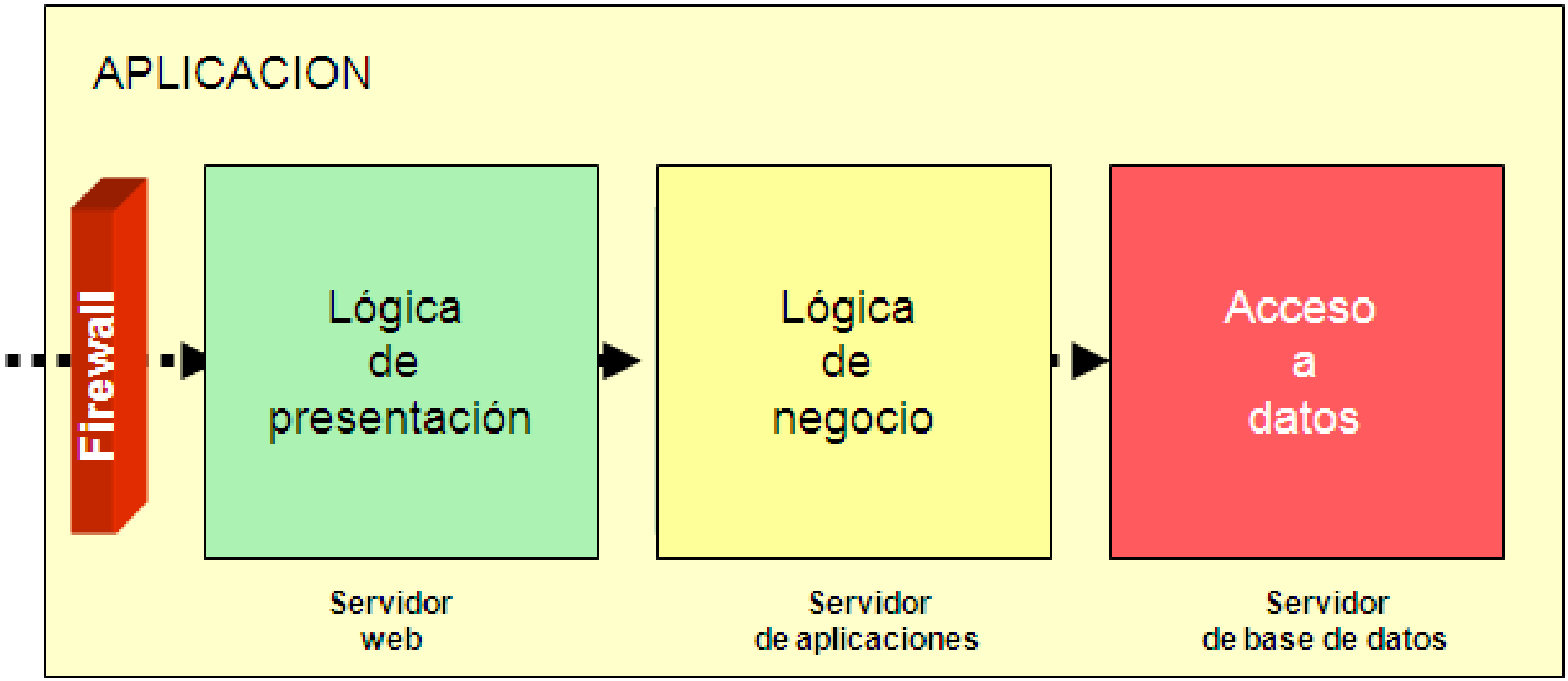
Inclusión de la seguridad en el SDLC (II)



Defensa en Profundidad



Defensa en Profundidad



Asegurar la aplicación

Validación de entradas

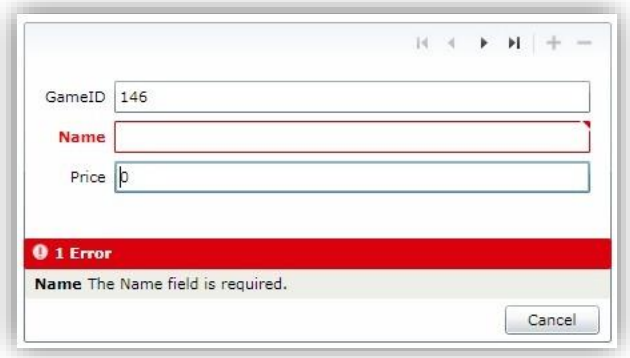
Autenticación

Autorización y manejo de sesiones

Datos sensibles y criptografía

Validación de Entradas

- Correctamente resuelto puede solucionar distintas vulnerabilidades
  - Cross-site Scripting
  - Command Execution
    - Buffer Overflow
    - Format String Attack
    - LDAP Injection
    - OS Commanding
    - SQL Injection



- Se busca la correcta validación de toda entrada
  - Asumir que todo input es malicioso y validarlo explícitamente
  - No confiar en datos que puedan ser manipulados o validados del lado del cliente
  - Considerar que los datos enviados por el cliente puede ser modificada con proxys locales
  - Validar los datos en un entorno confiable (lado del servidor)
  - Definir el criterio de validación

Validación de Entradas

- Considerar validaciones múltiples
- Validar antes de enviar datos al backend
- Codifique las salidas (encoding)
- Canonicalizar los datos antes de validarlos (desencodear)
- Verificar límites de longitud (overflow)
- Verificar tipo de datos esperado (tipo, largo, formato y rango)
- Rechazar cualquier cosa que no este específicamente permitida (whitelisting)

Validación de Entradas: caracteres especiales a filtrar

SQL injection

`	(comilla simple)
;	(punto y coma)
--	(signo menos)
<	(menor)
>	(mayor)
"	(comilla)
(	(paréntesis)
)	(paréntesis)
?	(signo pregunta)
&	(ampersand)
*	Asterisco
=	Igual

Varios etc.

XSS

<	&lt;
>	&gt;
(	&#40;
)	&#41;
#	&#35;
&	&#38;

Format string

%p
%f
%n

OS Commanding

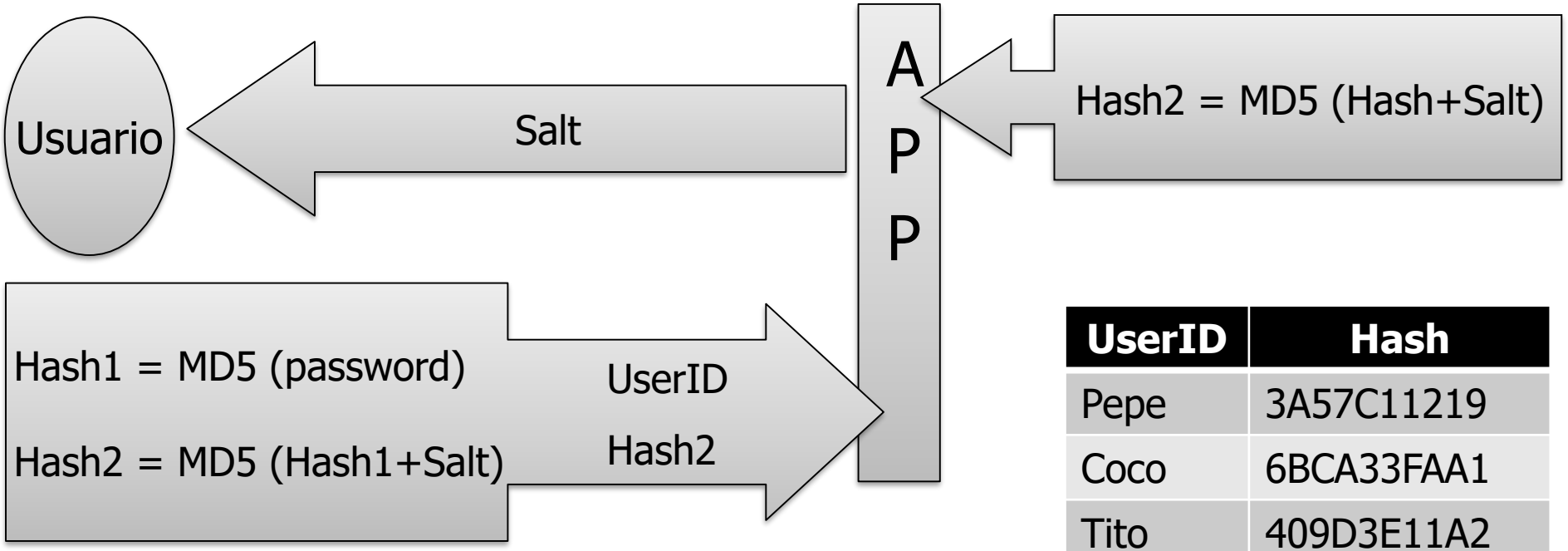
..	Punto punto
/	barra común
\	barra invertida
|	pipe
>	Mayor
<	Menor
?	Pregunta
*	Asterisco
$	Pesos

Etc.

Considerar todos los encodings

Autenticación

- No enviar contraseñas en texto claro
- No guardar contraseñas en texto claro en la bases de datos
- Usar valores salt en lo posible



UserID	Hash
Pepe	3A57C11219
Coco	6BCA33FAA1
Tito	409D3E11A2

Autenticación

- Ataques por fuerza bruta
  - Evitar que se pueda repetir intentos de autenticación
  - El bloqueo de cuentas puede promover ataques de denegación del servicio
  - Forzar intervención humana al superar el umbral de intentos fallidos (captcha)

- Autenticación débil o insuficiente
  - No implementar zonas de la aplicación a las que se pueda acceder anónimamente
  - No confiar en la seguridad por oscuridad
  - Aplicar políticas de passwords y de cuentas

- Mecanismos de recuperación de contraseñas
  - Implementar preguntas secretas de manera prudente
  - No mostrar ni enviar la contraseña antigua, solo permitir blanquear y reestablecer
  - Enviar link seguro con expiración

Autorización y Manejo de Sesiones

- Autorización
  - Distinguir y separar las zonas de la aplicación de distinto nivel de privilegios de acceso
  - Prever granularidad en el control de acceso a las distintas zonas, funciones y operaciones
  - Validar privilegios antes de permitir acceder a la página
  - Solicitar reautenticación para operaciones o transacciones criticas

- Manejo de Sesiones
  - Generar IDs de sesión aleatorio
  - Usar dos valores para tokens (uno antes y otro después de la autenticación)
  - Invalidar el ID de sesión cuando el usuario cierra la sesión
  - Definir periodos de expiración para las sesiones según uso

Datos Sensibles y Criptografía

- Datos sensibles
  - Cifrar los datos que circulan por redes o usar canales seguros
  - Evitar almacenar datos sensibles
  - Implementar control de acceso sobre datos sensibles
  - No guardar información sensible en cookies persistentes

- Criptografía
  - No implementar algoritmos criptográficos propietarios
  - Usar las funcionalidades estandarizadas (plataforma, lenguaje, framework)
  - Seleccionar algoritmo y longitud de clave adecuados según sensibilidad
  - Tomar precaución en la administración de claves

Recomendaciones

- Utilizar estándares de programación y metodologías
- Implementar Control de Cambios y Control de Versiones
- Implementar código reusable
- Implementar rutinas genéricas de validación por tipo de datos
- Implementar revisiones de código internas y externas
- Implementar control de calidad y testing a su proceso de desarrollo
- Considerar el filtrado a nivel de aplicación
- No agregar seguridad mediante funciones

Recomendaciones (II)

- Considerar el nivel de seguridad de una aplicación como mix de distintas cosas
- Si no se pueden cubrir todas las situaciones, cubrir las obvias
- Considerar la seguridad como proceso continuo, no como un producto
- Buscar seguridad en estadios tempranos para minimizar el costo de los cambios
- Realizar pruebas metódicas y repetibles
- Documentar todo lo posible
- Buscar tomar la menor cantidad de decisiones en tiempo de programación
  - Si hay que tomarlas, hacerlo con criterio de seguridad

Recomendaciones (III)

- No usar solo la lógica de la aplicación
- No confiar solo en los resultados de las herramientas (verificar manualmente)
- Considerar las evaluaciones externas
- Aprender de los errores
- Tener en cuenta los escenarios
- Considerar la seguridad como proceso continuo

¿Preguntas?

Federico Pacheco

 @FedeQuark

 www.federicopacheco.com.ar

 info@federicopacheco.com.ar