UNIVERSIDAD DE OVIEDO

ASTURIAS
CAMPUS DE EXCELENCIA
INTERNACIONAL
| AD FUTURUM |

MÁSTER UNIVERSITARIO EN INGENIERÍA WEB

TRABAJO FIN DE MÁSTER

"AMPLIACIÓN DE MÓDULO DE SEGURIDAD PARA INTERNET INFORMATION SERVICES 7"

MIGUEL OTERO GAFARELO

Fdo. D. Jose Manuel Redondo López

Oviedo, Julio de 2014

Agradecimientos

A mi director de proyecto, Redondo, por la oportunidad que me dio y por la ayuda recibida.

A Nacho, por el apoyo y por las buenas ideas.

A Penélope, por la revisión de la documentación.

A mis padres y a mi tía, Marili, por todo.

Resumen

El proyecto consiste en el desarrollo un módulo para el servidor web Microsoft IIS 7. Un módulo, en el ámbito de IIS, es un programa que se ejecuta en un determinado momento del procesamiento de cada una de las peticiones HTTP que le llegan al servidor, pudiendo analizarlas y/o modificar las respuestas enviadas al cliente. El objetivo de este proyecto es proporcionar al usuario una herramienta de seguridad web independiente de la implementación de la aplicación web en sí.

Las funcionalidades desarrolladas son las siguientes:

- Detección y bloqueo de los ataques de inyección de código SQL Injection y Cross-Site Scripting mediante uso de expresiones regulares.
- Eliminación de la persistencia de cookies.
- Chequeo de la dirección IP del cliente contra la API de una DNS Blacklist externa.
- Desarrollo de una interfaz gráfica acoplada al programa de administración de IIS para la configuración de las funcionalidades citadas.

La principal ventaja de este módulo es que se puede activar para cualquier aplicación gestionada por IIS 7, con lo que el programador puede delegar parte de la responsabilidad en cuanto a la implementación de medidas de seguridad de las aplicaciones en dicho módulo.

De forma general, cabe destacar también que todas las funcionalidades son activables y desactivables por el usuario, de forma que el hecho de tener el módulo activado para una aplicación web no obliga a que se utilicen todas sus funcionalidades.

Palabras clave

Servidor web “Internet Information Services” (IIS)

Módulos IIS

Seguridad web

Análisis de peticiones HTTP

Inyección de código

Cookies HTTP

DNS Blacklist

Abstract

The project consists in the development of a module for the Microsoft IIS 7 web server. A module, in the context of IIS, is a program that runs in a certain moment of each HTTP request processing, being able to analyze them and/or modify the responses sent to the client. The aim of this project is to provide to the user a web security tool that is independent of the web application itself.

The functionalities developed are the following:

- Detection and blocking of code injection attacks SQL Injection and Cross Site Scripting by means of regular expressions
- Cookies persistence removal.
- Check of the client IP address against an external DNS Blacklist API.
- Development of a user interface built in the IIS management program for the configuration of the previous functionality.

The main perk of this module is that it can be enabled for any application managed by IIS 7, thus the programmer can delegate part of the responsibility for the implementation of the application security in the aforementioned module.

In general, it is worth highlighting that the user can enable or disable each of the functionalities, so the fact that the module is enabled for a certain web application does not force all of the functionalities to be used.

Keywords

"Internet Information Services" (IIS) web server

IIS modules

Web security

HTTP requests analysis

Code injection

HTTP cookies

DNS Blacklist

Índice General

CAPÍTULO 1. MEMORIA DEL PROYECTO 19

1.1 RESUMEN DE LA MOTIVACIÓN, OBJETIVOS Y ALCANCE DEL PROYECTO 19
1.2 RESUMEN DE TODOS LOS ASPECTOS 21
1.2.1 Introducción 21
1.2.2 Aspectos Teóricos 21
1.2.3 Planificación del Proyecto y Resumen de Presupuestos 21
1.2.4 Análisis 21
1.2.5 Diseño 21
1.2.6 Implementación del Sistema 22
1.2.7 Desarrollo de las Pruebas 22
1.2.8 Manuales del Sistema 22
1.2.9 Conclusiones y Ampliaciones 22
1.2.10 Presupuesto 22
1.2.11 Referencias Bibliográficas 22
1.2.12 Apéndices 22

CAPÍTULO 2. INTRODUCCIÓN 23

2.1 JUSTIFICACIÓN DEL PROYECTO 23
2.2 OBJETIVOS DEL PROYECTO 24
2.3 ESTUDIO DE LA SITUACIÓN ACTUAL 25
2.3.1 Evaluación de Alternativas 25

CAPÍTULO 3. ASPECTOS TEÓRICOS 27

3.1 PROTOCOLO HTTP 27
3.2 COOKIE HTTP 28
3.3 INTERNET INFORMATION SERVICES 29
3.4 EXPRESIÓN REGULAR 30
3.5 DIRECCIÓN IP 30
3.6 DNS BLACKLIST 31
3.7 INYECCIÓN DE CÓDIGO 32

CAPÍTULO 4. PLANIFICACIÓN DEL PROYECTO Y RESUMEN DE PRESUPUESTOS 33

4.1 PLANIFICACIÓN 33
4.2 RESUMEN DEL PRESUPUESTO 34

CAPÍTULO 5. ANÁLISIS 35

5.1 DEFINICIÓN DEL SISTEMA 35
5.1.1 Determinación del Alcance del Sistema 35
5.2 REQUISITOS DEL SISTEMA 36
5.2.1 Obtención de los Requisitos del Sistema 36
5.2.2 Identificación de Actores del Sistema 37
5.2.3 Especificación de Casos de Uso 38
5.3 IDENTIFICACIÓN DE LOS SUBSISTEMAS EN LA FASE DE ANÁLISIS 42
5.3.1 Descripción de los Subsistemas 42
5.3.2 Descripción de los Interfaces entre Subsistemas 42
5.4 DIAGRAMA DE CLASES PRELIMINAR DEL ANÁLISIS 44

5.4.1 Diagrama de Clases 44
5.4.2 Descripción de las Clases....45
5.5 ANÁLISIS DE CASOS DE USO Y ESCENARIOS 49
5.5.1 Caso de Uso 1: "Activar/desactivar comprobación de URL" 49
5.5.2 Caso de Uso 2: "Activar/desactivar comprobación de elementos de formulario" 50
5.5.3 Caso de Uso 3: "Activar/desactivar comprobación de cookies"....51
5.5.4 Caso de Uso 4: "Activar/desactivar eliminación de persistencia en cookies" 52
5.5.5 Caso de Uso 5: "Activar/desactivar comprobación de IPs registradas en DNS blacklist" 53
5.5.6 Caso de Uso 6: "Activar/desactivar protección frente a SQL Injection"....54
5.5.7 Caso de Uso 7: "Activar/desactivar protección frente a XSS" 55
5.5.8 Caso de Uso 8: "Activar/desactivar uso de regla personalizada" 56
5.5.9 Caso de Uso 9: "Editar reglas de validación" 57
5.5.10 Caso de Uso 10: "Activar/desactivar recepción de email para IPs registradas en DNS Blacklist" 58
5.5.11 Caso de Uso 11: "Editar destinatario de email para IPs registradas en DNS blacklist" 59
5.5.12 Caso de Uso 12: "Editar número máximo de intentos de acceso desde IPs registradas en DNS Blacklist"....60
5.5.13 Caso de Uso 13: "Recibir email de acceso desde IP registrada en DNS Blacklist" 61
5.5.14 Caso de Uso 14: "Enviar petición HTTP para análisis" 62
5.5.15 Caso de Uso 15: "Recibir respuesta HTTP de pipeline" 63
5.6 ANÁLISIS DE INTERFACES DE USUARIO 64
5.6.1 Descripción de la Interfaz....64
5.6.2 Descripción del Comportamiento de la Interfaz....66
5.7 ESPECIFICACIÓN DEL PLAN DE PRUEBAS 66
5.7.1 Pruebas Unitarias....67
5.7.2 Pruebas de Integración 67
5.7.3 Pruebas de Sistema 67
5.7.4 Clasificación de Pruebas según Caso de Uso....67

CAPÍTULO 6. DISEÑO DEL SISTEMA....75

6.1 ARQUITECTURA DEL SISTEMA....75
6.1.1 Diagramas de Paquetes 75
6.1.2 Diagramas de Componentes....76
6.1.3 Diagramas de Despliegue 77
6.2 DISEÑO DE CLASES 80
6.2.1 Diagramas de Clases....80
6.3 DIAGRAMAS DE SECUENCIA 82
6.3.1 Activación/desactivación de opción de módulo....82
6.3.2 Edición de parámetros de módulo 83
6.3.3 Enviar/recibir petición HTTP....84
6.3.4 Analizar petición HTTP 85
6.4 DIAGRAMAS DE ACTIVIDADES 86
6.5 DISEÑO DE LA INTERFAZ....87
6.6 ESPECIFICACIÓN TÉCNICA DEL PLAN DE PRUEBAS....90
6.6.1 Integración del módulo en IIS....90
6.6.2 Funcionalidad relativa a la URL....91
6.6.3 Funcionalidad relativa a los elementos de formulario....93
6.6.4 Funcionalidad relativa a las Cookies....94
6.6.5 Funcionalidad relativa a las IPs....96
6.6.6 Interfaz....97

CAPÍTULO 7. IMPLEMENTACIÓN DEL SISTEMA 101

7.1 ESTÁNDARES Y NORMAS SEGUIDOS 101
7.2 LENGUAJES DE PROGRAMACIÓN 102
7.3 HERRAMIENTAS Y PROGRAMAS USADOS PARA EL DESARROLLO 103
7.3.1 Visual Studio 2013 103
7.3.2 IIS 7.5 103
7.3.3 GitHub 103
7.4 CREACIÓN DEL SISTEMA 104
7.4.1 Problemas Encontrados 104
7.4.2 Descripción Detallada de las Clases 105

CAPÍTULO 8. DESARROLLO DE LAS PRUEBAS 117

8.1.1 Integración del módulo en IIS 117
8.1.2 Funcionalidad relativa a la URL 118
8.1.3 Funcionalidad relativa a los elementos de formulario 120
8.1.4 Funcionalidad relativa a las Cookies 121
8.1.5 Funcionalidad relativa a las IPs 123
8.1.6 Interfaz 125

CAPÍTULO 9. MANUALES DEL SISTEMA 129

9.1 MANUAL DE INSTALACIÓN 129
9.1.1 Windows Server 129
9.1.2 Windows 7 133
9.2 MANUAL DE EJECUCIÓN 135
9.2.1 Activación de Interfaz 135
9.2.2 Publicación de la Aplicación Web 139
9.2.3 Activación del Módulo en la Aplicación 141
9.3 MANUAL DE USUARIO 143
9.3.1 Ejemplo de Funcionamiento 146
9.4 MANUAL DEL PROGRAMADOR 149
9.4.1 Interfaz 149
9.4.2 Módulo 149

CAPÍTULO 10. CONCLUSIONES Y AMPLIACIONES 151

10.1 CONCLUSIONES 151
10.2 AMPLIACIONES 152

CAPÍTULO 11. PRESUPUESTO 153

11.1 PRESUPUESTO DEL CLIENTE 153
11.2 PRESUPUESTO DE COSTES 154

CAPÍTULO 12. REFERENCIAS BIBLIOGRÁFICAS 155

12.1 LIBROS Y ARTÍCULOS 155
12.2 REFERENCIAS EN INTERNET 155
12.2.1 Referencias consultadas 155
12.2.2 Referencias utilizadas en el Documento 155

CAPÍTULO 13. APÉNDICES 157

13.1 GLOSARIO 157

13.2 CONTENIDO ENTREGADO 159
13.2.1 Contenidos 159
13.2.2 Ficheros de Configuración 159
13.3 CÓDIGO FUENTE 160
13.3.1 Clases del Módulo 160
13.3.2 Clases de Configuración 169
13.3.3 Clases de la Interfaz 173

Índice de Figuras

Ilustración 3-1: Formato de petición HTTP 27
Ilustración 3-2: Formato de respuesta HTTP 27
Ilustración 3-3: Cookie HTTP persistente 28
Ilustración 3-4: Pipeline de IIS 7 29
Ilustración 4-1: Diagrama de Gantt sobre la planificación del proyecto 33
Ilustración 5-1: Diagrama de casos de uso de Usuario 38
Ilustración 5-2: Diagrama de casos de uso de HTTP.sys y WAS 39
Ilustración 5-3: Situación del módulo dentro del pipeline de IIS 43
Ilustración 5-4: Diagrama de clases preliminar 44
Ilustración 5-5: Situación de la interfaz del módulo dentro del programa de administración de IIS 7 64
Ilustración 5-6: Interfaz - Pestaña URL 64
Ilustración 5-7: Interfaz - Pestaña Formulario 65
Ilustración 5-8: Interfaz - Pestaña Cookies 65
Ilustración 5-9: Interfaz - Pestaña IP Blacklist 66
Ilustración 6-1: Diagrama de paquetes 75
Ilustración 6-2: Diagrama de componentes 76
Ilustración 6-3: Diagrama de despliegue 77
Ilustración 6-4: Diagrama de clases del módulo 80
Ilustración 6-5: Diagrama de clases de la interfaz del módulo 81
Ilustración 6-6: Diagrama de secuencia para casos de uso relativos a la activación/desactivación de opciones desde la interfaz 82
Ilustración 6-7: Diagrama de secuencia para casos de uso relativos a la edición de parámetros desde la interfaz 83
Ilustración 6-8: Diagrama de secuencia del envío y la recepción de una petición HTTP al módulo y desde el módulo 84
Ilustración 6-9: Diagrama de secuencia del procesamiento de una petición HTTP por parte del módulo 85
Ilustración 6-10: Diagrama de actividades del procesamiento de una petición HTTP por parte del módulo 86
Ilustración 6-11: Pestaña URL 87
Ilustración 6-12: Pestaña Formulario 87
Ilustración 6-13: Pestaña Cookies 88
Ilustración 6-14: Pestaña IP 88
Ilustración 6-15: Vista general de la interfaz integrada en el administrador de IIS 89
Ilustración 9-1: Administrador del servidor 129
Ilustración 9-2: Lista de funciones del servidor 130
Ilustración 9-3: Servicios de IIS 131
Ilustración 9-4: Página por defecto de IIS 132
Ilustración 9-5: Panel de control 133
Ilustración 9-6: Programas y características 133
Ilustración 9-7: Características de Windows 134

Ilustración 9-8: Archivos de esquemas de configuración 135
Ilustración 9-9: Secciones de configuración a incluir en IIS 136
Ilustración 9-10: Comprobación de la presencia de las nuevas secciones de configuración 136
Ilustración 9-11: Añadido del ensamblado de la interfaz a la caché global mediante la gacutil 137
Ilustración 9-12: Añadido de la interfaz al archivo de administración de IIS 137
Ilustración 9-13: Añadido de la interfaz a todas las aplicaciones administradas por IIS 137
Ilustración 9-14: Icono de la interfaz dentro del Administrador de IIS 138
Ilustración 9-15: Directorio wwwroot de IIS 139
Ilustración 9-16: Agregar sitio web al servidor IIS 139
Ilustración 9-17: Opciones para agregar sitio web 140
Ilustración 9-18: Acceso a la aplicación web desde el navegador 141
Ilustración 9-19: Vista “Características” de la aplicación web 141
Ilustración 9-20: Agregado de nuevo módulo a la aplicación web 142
Ilustración 9-21: Icono del administrador de IIS 143
Ilustración 9-22: Icono de la interfaz dentro del Administrador de IIS 144
Ilustración 9-23: Interfaz del módulo 144
Ilustración 9-24: Validación de la dirección de correo 145
Ilustración 9-25: Validación de elementos de formulario 146
Ilustración 9-26: Inicio de sesión correcto en la aplicación 147
Ilustración 9-27: Datos introducidos en el formulario (contraseña irrelevante ya que no se va a llegar a procesar) 147
Ilustración 9-28: Error 403 mostrado en el visor de eventos 148

Capítulo 1. Memoria del Proyecto

1.1 Resumen de la Motivación, Objetivos y Alcance del Proyecto

Los objetivos del proyecto son:

- Desarrollo de un módulo de seguridad acoplable al servidor web IIS 7
- Documentación de las funcionalidades implementadas
- Documentación del trabajo llevado a cabo por el alumno
- Presentación de dicho trabajo ante tribunal universitario

La motivación del proyecto viene dada por el interés personal del alumno en los aspectos de la seguridad informática, así como por la situación actual de la seguridad web en particular. Esta situación consiste básicamente en la tendencia a poner especial atención a la funcionalidad en sí de una aplicación web en detrimento de la seguridad de la misma. La consecuencia de esto es la aparición de numerosas aplicaciones web que funcionan correctamente en condiciones normales pero que, sin embargo, presentan un comportamiento erróneo o no previsto al intentar trabajar con datos cuyo formato o sintaxis no fueron considerados por el programador. Por normal general, estas situaciones se dan cuando el usuario introduce datos con el fin de causar este comportamiento en la aplicación y, así, averiguar los puntos débiles de la misma.

A raíz de la situación descrita, se pretende proporcionar una herramienta genérica dependiente únicamente del servidor web que, o bien exima de parte de la responsabilidad del programador en cuanto a la inclusión de medidas de seguridad en las aplicaciones desarrolladas, o bien sirva de refuerzo para las medidas implementadas en dichas aplicaciones.

El alcance del proyecto comprende el cumplimiento de los objetivos citados al principio de esta sección, consistiendo la funcionalidad en los siguientes aspectos:

- Detección y bloqueo de ataques de inyección de código tales como SQL Injection (SQLI), destinado a la manipulación ilegítima de las bases de datos usadas por la aplicación web, y Cross-site Scripting (XSS), destinado a modificar el comportamiento de la aplicación para con otros usuarios de la misma.
- Bloqueo de clientes web potencialmente sospechosos en base a su dirección IP. A grandes rasgos, la dirección IP es un mecanismo de identificación para máquinas que acceden a una red que utilice el protocolo IP. Esta funcionalidad consiste en hacer consultas a un servicio web externo (una DNSBL) que recoja las direcciones IP desde las que se ha llevado a cabo algún ataque, y bloquear los accesos a las aplicaciones web según los resultados obtenidos de dicho servicio.

- Eliminación de la persistencia en las cookies intercambiadas entre cliente y servidor. Las cookies son un sistema utilizado en el protocolo HTTP para proveer de cierto estado al mismo, es decir, hacer que el comportamiento de una determinada aplicación web se vea determinado por las acciones realizadas previamente por el usuario en anteriores accesos; por ejemplo, no teniendo que introducir nombre de usuario y contraseña cada vez que se quiera acceder a una determinada sección de la cuenta de un usuario en una aplicación web. Este sistema puede provocar vulnerabilidades si no se tienen en cuenta ciertas medidas de seguridad, como el encriptado de contraseñas de usuario. Concretamente, la persistencia en las cookies puede provocar que un usuario consiga acceso ilegítimo a la cuenta de otro que haya iniciado sesión en la aplicación web desde el mismo ordenador.
- Desarrollo de una interfaz de usuario para el manejo de las diferentes funcionalidades. El módulo requerirá de cierta configuración por parte del usuario para llevar a cabo las tareas deseadas, por lo que se proporcionará una interfaz gráfica acoplada al mismo programa desde el que se administrarán las páginas web; en este caso, el servidor web Microsoft Internet Information Services.

1.2 Resumen de Todos los Aspectos

1.2.1 Introducción

En este apartado se entra en detalle en los objetivos, justificación y situación actual del proyecto. Se explican, además, soluciones alternativas al mismo problema para el que el proyecto pretende dar solución.

1.2.2 Aspectos Teóricos

En este apartado se explican varios conceptos relativos al proyecto los cuales es necesario entender, al menos de una forma general, para comprender cómo funciona el módulo objeto de este proyecto, así como el servidor web IIS.

1.2.3 Planificación del Proyecto y Resumen de Presupuestos

En este apartado se resume la planificación llevada a cabo para el desarrollo del proyecto, acompañada del correspondiente diagrama de Gantt. Asimismo, se presenta una tabla con el presupuesto para llevar a cabo el proyecto junto con un texto explicativo de los valores presentes en dicha tabla.

1.2.4 Análisis

En este apartado se explica el alcance del proyecto y los requisitos del mismo para llevar a cabo lo documentado en dicho alcance, y se presentan diagramas preliminares del sistema, los cuales serán perfeccionados y detallados en la fase de diseño. Igualmente, se presentan los casos de uso y la interfaz gráfica con la que contará el sistema, así como una primera aproximación a la especificación del plan de pruebas, el cual será detallado también en la fase de diseño.

1.2.5 Diseño

En este apartado se presenta y explica la estructura final del sistema a través de diagramas UML que representarán la arquitectura del mismo y sus diferentes flujos de actividades. De la misma forma, se presentará el diseño final de la interfaz gráfica y se detallarán las pruebas a llevar a cabo sobre el sistema para asegurar el correcto funcionamiento del mismo.

1.2.6 Implementación del Sistema

En este apartado se documenta la fase de implementación del sistema; se citan y explican brevemente los lenguajes de programación utilizados, así como los entornos de desarrollo y herramientas usadas durante el desarrollo del sistema. Esta sección contiene también una lista de los principales problemas encontrados durante la fase de implementación.

1.2.7 Desarrollo de las Pruebas

En este apartado se documentan los resultados de las pruebas, tanto positivos como negativos, y las medidas llevadas a cabo para corregir aquellos aspectos del programa que provocaban los resultados negativos.

1.2.8 Manuales del Sistema

En este apartado se explican los conceptos necesarios para utilizar el programa, desde la instalación del mismo hasta la forma de activarlo y configurarlo para una u otra aplicación web. Además se provee un manual para desarrolladores con el objetivo de proporcionar una guía básica para la modificación de su funcionamiento.

1.2.9 Conclusiones y Ampliaciones

En este apartado se explica la situación una vez finalizado el proyecto y lo sacado en claro durante el desarrollo del mismo. Se citan además algunas de las posibles ampliaciones a llevar a cabo sobre el mismo.

1.2.10 Presupuesto

En este apartado se presentan dos tablas de presupuestos. El primero destinado a presentar al cliente del proyecto, y el segundo, de coste, en el que se desglosan todos los conceptos que generaron gastos durante el desarrollo.

1.2.11 Referencias Bibliográficas

En este apartado se incluyen los recursos bibliográficos (tanto libros y artículos como referencias en la red), utilizados para el desarrollo del proyecto y de su documentación.

1.2.12 Apéndices

En esta sección se presentan el glosario y diccionario de datos, la estructura de carpetas del código entregado y una transcripción del mismo.

Capítulo 2. Introducción

2.1 Justificación del Proyecto

El proyecto se ha llevado a cabo motivado, por una parte, por el interés personal del alumno en aspectos de seguridad web, y por otra parte como puesta en práctica de trabajos anteriores con módulos para IIS 7. La situación actual también juega un papel importante en los motivos que llevaron a desarrollar este proyecto, pues en muchas ocasiones la seguridad en las aplicaciones web actuales no se tiene en cuenta a la hora de ser desarrolladas, lo que puede causar, y causa, pérdidas de tiempo y dinero a medio y largo plazo.

Se espera que, una vez finalizado el proyecto, se tenga una herramienta de seguridad web genérica para el servidor IIS 7. Esta genericidad viene dada por el hecho de que la funcionalidad del módulo podrá ser aprovechada por cualquier aplicación web gestionada por IIS, lo que eximirá a los programadores de cada una de estas aplicaciones de implementar las medidas de seguridad que ya incluya el módulo.

Entrando más en detalle, esta herramienta será capaz de detectar y bloquear patrones sospechosos en las peticiones realizadas al servidor web que puedan suponer un intento de inyección de código y, en consecuencia, modificación del comportamiento de las aplicaciones gestionadas por dicho servidor. Además, se proveerá de mecanismos para bloquear accesos realizados desde direcciones IP sospechosas y para evitar la creación de elementos intercambiables entre cliente y servidor (cookies) que puedan suponer un futuro acceso ilegítimo por parte de otro usuario.

2.2 Objetivos del Proyecto

- **Desarrollo de un módulo de seguridad para IIS 7**: Este objetivo consiste en el desarrollo de la herramienta software en sí. A continuación se describen los subobjetivos correspondientes:
    - **Detección y bloqueo de ataques de inyección de código:** El módulo será capaz, mediante el uso de expresiones regulares, de detectar patrones sospechosos incluidos en las peticiones HTTP enviadas al servidor. En caso de detección positiva, dicho módulo cortará el flujo de procesamiento de la petición HTTP en cuestión y generará un error a mostrar al usuario de la aplicación web.
    - **Detección y bloqueo de accesos desde direcciones IP sospechosas:** El módulo realizará consultas a una DNSBL externa a través de la API de la misma para comprobar que las direcciones IP desde la que se realizan accesos al servidor web no estén en lista negra. En caso de que estén, se podrán bloquear las peticiones HTTP realizadas desde estas direcciones.
    - **Eliminación de la persistencia en cookies HTTP:** El módulo podrá eliminar la persistencia en las cookies intercambiadas entre cliente y servidor con el fin de limitar posibles accesos ilegítimos a cuentas de usuario de las aplicaciones web.
    - **Interfaz gráfica:** Se proveerá al módulo de una interfaz gráfica integrada en el programa de administración de IIS 7 con el fin de que el usuario pueda configurar las funcionalidades anteriores según convenga para cada aplicación web.
- **Documentación del módulo:** Se proporcionarán manuales de instalación y uso para guiar al usuario a través de la activación y utilización del módulo. Se proveerá también de una guía para programadores con el objetivo de facilitar futuras ampliaciones o modificaciones en la funcionalidad del módulo.
- **Documentación del trabajo realizado:** Además de documentar el funcionamiento del módulo desarrollado cara a su posterior instalación, uso y posible modificación, se habrá de proporcionar documentación del trabajo realizado, incluyendo las diferentes fases del proyecto, planificación y presupuestos, así como futuras ampliaciones sobre el proyecto y referencias bibliográficas utilizadas.
- **Presentación del proyecto:** Una vez finalizados los objetivos anteriores, se habrá de presentar el proyecto ante un tribunal universitario encargado de evaluarlo.

2.3 Estudio de la Situación Actual

Actualmente existen varias soluciones que abordan el tema de la seguridad en aplicaciones web. Todas estas soluciones se basan en una premisa: Los datos enviados por un usuario son potencialmente peligrosos, por lo que es necesario validarlos en el servidor antes de procesarlos.

La principal ventaja del módulo frente a gran parte de estas soluciones es su genericidad: Estas soluciones implican modificar cada una de las aplicaciones a nivel de código para conseguir la seguridad deseada, mientras que el módulo objeto del proyecto puede ser activado para cualquier aplicación administrada por IIS, a modo de Web Application Firewall.

A continuación se evalúan algunas de las alternativas a la utilización del módulo para conseguir una protección similar frente a ataques web.

2.3.1 Evaluación de Alternativas

2.3.1.1 Uso de un Web Application Firewall (WAF) existente

Descripción

Los WAF se definen como plugins o extensiones de un servidor web, y se encargan de aplicar una serie de reglas a la comunicación HTTP entre este servidor y los clientes. Estas reglas suelen ser personalizables por el usuario y su principal función es proteger contra ataques de inyección de código. Por esta definición, el módulo desarrollado se podría clasificar como un WAF de IIS 7.

Ejemplos de WAF son el ModSecurity o el SecureSphere

Ventajas

- Protege eficazmente contra ataques de inyección de código.
- Es más complejo que el módulo desarrollado y sus funcionalidades son más sofisticadas.

Inconvenientes

- La misma complejidad que hace que su funcionalidad sea más sofisticada que la del módulo, hace que sea más difícil de configurar que éste.

2.3.1.2 Uso de HttpRequest.ValidateInput

Descripción

ValidateInput es un método proporcionado por la clase HttpRequest de .NET. Esta clase será, de hecho, utilizada en la implementación del módulo objeto del proyecto. La funcionalidad de este método, como su nombre indica, es validar los datos enviados por el usuario antes de comenzar el procesamiento de los mismos.

Ventajas

- *ValidateInput* es más eficaz en cuanto a la detección de ataques XSS que las reglas de detección implementadas en el módulo

Inconvenientes

- Es necesario implementarlo en cada aplicación web.
- No incluye ninguna de las otras funcionalidades presentes en el módulo.
- Es más complejo de configurar que éste.

2.3.1.3 Implementación de mecanismos de seguridad en el código

Descripción

Hoy en día, la mayor parte de entornos de desarrollo ponen a disposición del programador herramientas que permiten implementar diferentes medidas de seguridad en las aplicaciones. Cara a la seguridad web, gran parte de estas herramientas son las que "escapan" los datos introducidos por el usuario en formularios y demás elementos web, de forma que nada de lo que escriba pueda afectar al funcionamiento de la aplicación. Librerías como ADO.NET o la clase PreparedStatement de Java, son ejemplos de herramientas de programación para evitar ataques de inyección de código.

Ventajas

- Protección eficaz contra ataques de inyección de código.

Inconvenientes

- Necesidad de implementar las medidas de seguridad para cada aplicación.

Capítulo 3. Aspectos Teóricos

3.1 Protocolo HTTP

HTTP (Hypertext Transfer Protocol), es el protocolo que dicta el formato de los intercambios de información que se dan entre dos máquinas conectadas a una red de Internet. Este protocolo obliga a que estos intercambios o transacciones sean del formato petición-respuesta, de forma que, para que una máquina obtenga la información que otra máquina tiene publicada en la red, primero ha de enviarle una petición con un formato concreto, a la cual esta otra máquina le enviará la respuesta correspondiente.

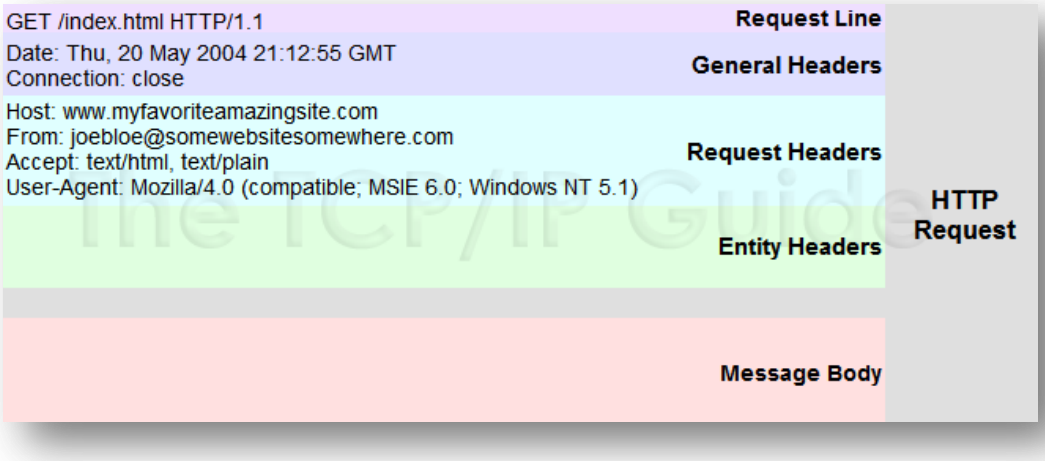


Ilustración 3-1: Formato de petición HTTP

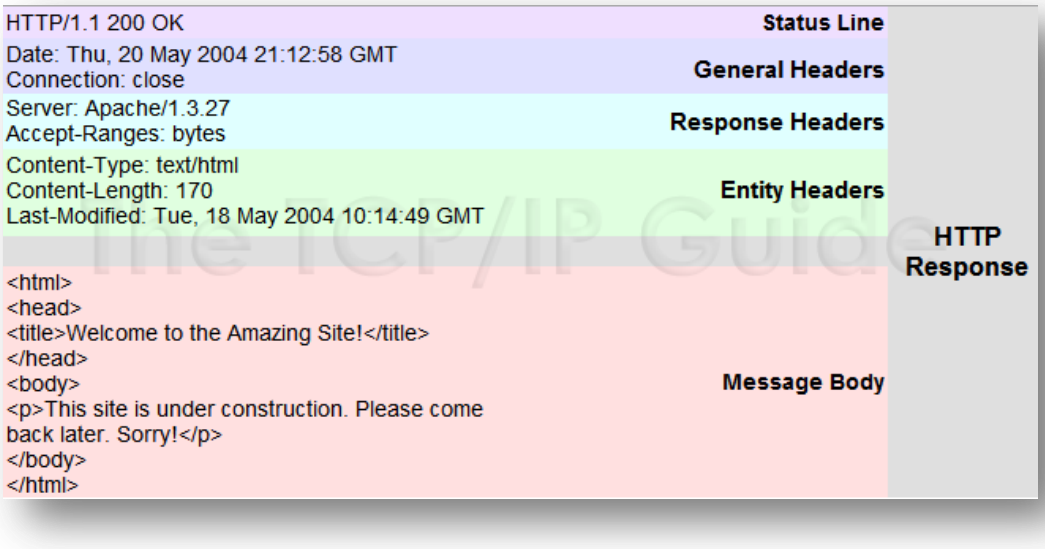


Ilustración 3-2: Formato de respuesta HTTP

Cabe destacar que HTTP es un protocolo sin estado, es decir, que cada par petición-respuesta es independiente de los realizados anteriormente, de forma que a la hora de obtener contenidos de una página web, estos son, en principio, siempre los mismos. Para solventar esta falta de estado se idearon las cookies HTTP.

Para más información acerca del protocolo HTTP, consultar las referencias [1] y [2] .

3.2 Cookie HTTP

Como ya se dijo, las cookies HTTP aparecen como solución parcial a la falta de estado del protocolo HTTP. A nivel superficial, una cookie es un fichero de texto con formato clave-valor que se intercambia entre el cliente y el servidor. Un ejemplo cercano de uso de cookies son los mecanismos de inicio de sesión: dada la falta de estado de HTTP, un usuario tendría que estar introduciendo su identificador y contraseña cada vez que intentase acceder a una sección de la aplicación web que requiriese identificación. Gracias a las cookies, dicho usuario sólo tiene que introducir sus credenciales una vez, y hasta que no cierre sesión o esta caduque, puede realizar todos los accesos que quiera sin necesidad de repetirlas.

El mecanismo de creación de cookies es el siguiente:

1. El usuario introduce sus credenciales
2. La aplicación las valida y, desde el servidor, crea la cookie
3. La cookie es enviada al cliente, donde queda alojada para futuras peticiones HTTP, junto con la respuesta HTTP correspondiente
4. El cliente envía la cookie en las siguientes peticiones HTTP a la aplicación y ésta lo identifica como el cliente que introdujo sus credenciales anteriormente

Cabe destacar que las cookies tienen una fecha de caducidad establecida en el servidor; es decir, dejan de ser válidas en el momento en que se alcanza esta caducidad o que el usuario las invalida (por ejemplo, cerrando sesión en la aplicación web). Aquellas cookies que no tienen establecida una caducidad determinada se denominan cookies de sesión, y quedan obsoletas una vez se cierra el navegador.

Para más información acerca de cookies HTTP, consultar la referencia [3]

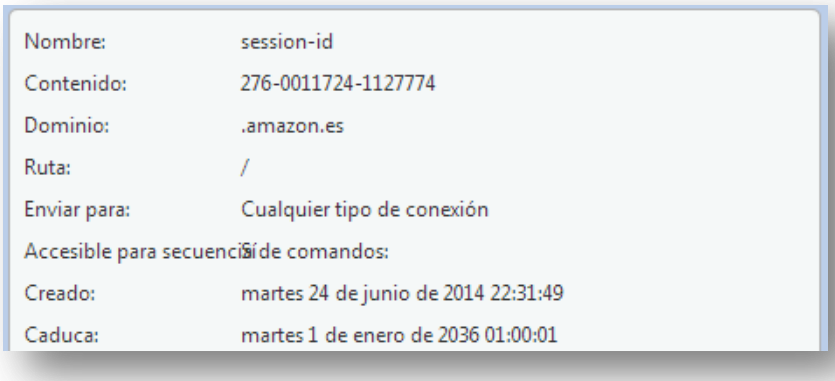


Ilustración 3-3: Cookie HTTP persistente

3.3 Internet Information Services

Internet Information Services (IIS) es un servidor web desarrollado por Microsoft. A grandes rasgos, un servidor web es un software que pone a disposición de la red contenidos de la máquina en la que está instalado. Cuenta con funcionalidades relativas a la gestión de credenciales, accesos a bases de datos, generación de errores y resolución de nombres, entre otras cosas.

Una característica de IIS, a partir de la versión 7 del mismo, es su motor de procesamiento de peticiones HTTP. Dicho motor presenta una arquitectura en "pipeline", de forma que las peticiones llegan al servidor web y son resueltas a medida que van siendo procesadas a través de una serie de módulos que desempeñan diferentes funcionalidades. Estos módulos son ejecutados en diferentes fases del ciclo de vida de una transacción petición-respuesta HTTP.

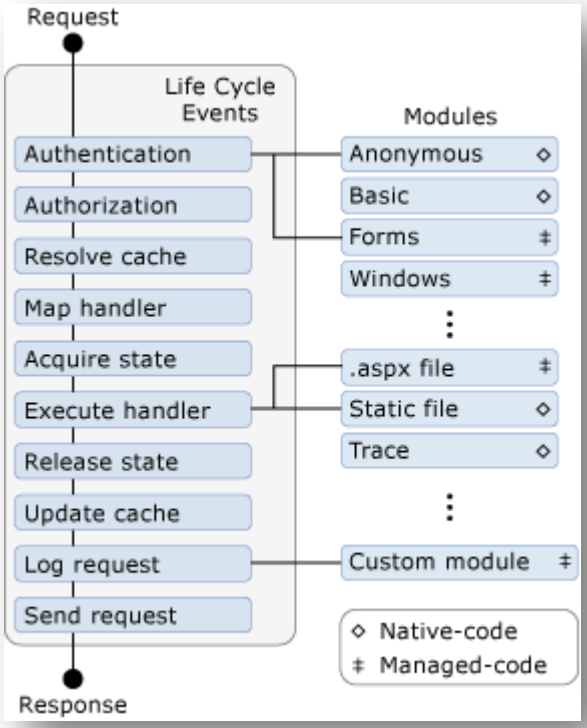


Ilustración 3-4: Pipeline de IIS 7

3.4 Expresión regular

Una expresión regular es una secuencia de caracteres que representa un formato o estructura que ha de seguir una cadena de texto. Dicho de otro modo, cada expresión regular dicta una serie de normas de sintaxis que tiene que cumplir una determinada cadena de texto para ser validada por dicha expresión.

Las expresiones regulares constan de una serie de caracteres o conjuntos de caracteres que dictan cada uno una determinada regla que han de seguir las cadenas de texto para ser aceptadas por la expresión en sí. Por ejemplo, los corchetes representan rangos, de forma que la expresión regular "[a-h]" acepta cualquier cadena de caracteres formada por una única letra situada en ese intervalo. El signo más (+) representa una repetición indeterminada de las cadenas de texto afectadas por la regla anterior, de forma que la expresión regular "[a-h]+" engloba todas aquellas cadenas de texto formadas por repeticiones de las letras contenidas en el rango anterior, como "aaa", "abha" o "adabchh".

Para más información acerca de las expresiones regulares, consultar la referencia [4]

3.5 Dirección IP

La dirección IP es una cadena numérica que sirve de identificador para cada dispositivo conectado a una red informática que use el protocolo IP (Internet Protocol). Además de servir de identificador, también ofrece información sobre la localización geográfica de la máquina.

Hay dos tipos de dirección IP según la versión del protocolo, la IPv4 y la IPv6, siendo la primera la más utilizada. La dirección IPv4 es un número de 4 bytes cuya representación decimal sigue el formato XXX.XXX.XXX.XXX, siendo el rango desde la dirección 0.0.0.0 hasta la 255.255.255.255

Cabe destacar que una dirección IP no identifica a una máquina de forma unívoca, pues hay dispositivos (routers) que dividen las direcciones IP en otras subdirecciones IP privadas, haciendo más compleja la identificación final. Por ejemplo, la dirección IP 156.35.98.20 puede corresponder a un router que genera direcciones IP locales desde la 192.168.1.0 hasta la 192.168.1.255 para cada máquina que esté conectado al mismo.

Para obtener más información más acerca de direcciones IP, consultar las referencias [5] y [6]

3.6 DNS Blacklist

Las DNS Blacklist (DNSBL) son listas que contienen direcciones IP asociadas a redes de spam o desde las que se han registrado ataques informáticos. Normalmente son publicadas por empresas o asociaciones relacionadas con la seguridad informática junto con redes denominadas "honeypot", destinadas a atraer la atención de atacantes para así registrar sus IPs y su forma de operar.

Algunas DNSBL están dotadas de una API a través de la cual una aplicación puede hacer consultas dinámicas a la misma para comprobar si alguna de las IPs desde la que se realizan accesos está registrada. Por ejemplo, haciendo una consulta a la misma si desde una determinada IP se han llevado a cabo varios intentos fallidos de inicio de sesión.

Normalmente los usuarios pueden contribuir a la expansión de las DNSBL con el objetivo de lograr una mayor base de datos de direcciones IP potencialmente peligrosas.

Para más información acerca de las DNS Blacklist, consultar las referencias [7] y [8]

Además, la página Wikipedia contiene una tabla comparativa de varias DNSBL con información relativa a cada una de ellas y su URL. Referencia [9]

3.7 Inyección de código

"Inyección de código" es un término relativo a la seguridad informática que engloba todos aquellos ataques que tienen por objetivo insertar código en una o varias aplicaciones de una máquina externa y hacer que se ejecute en dicha máquina, logrando con ello un funcionamiento erróneo de estas aplicaciones. Los objetivos de un ataque de inyección de código pueden ir desde inutilizar aplicaciones web, hasta a acceder a información sensible sobre éstas o sobre sus usuarios.

Hay diferentes tipos de ataques de inyección de código, pero los más famosos y extendidos son los dos siguientes:

- **Cross-site Scripting (XSS):** Este ataque se realiza normalmente introduciendo instrucciones javascript en elementos web en los que el usuario puede escribir texto. El código javascript es ejecutado en el navegador web de cada usuario que accede a la página web que lo contiene. Si un usuario malintencionado escribe un fragmento de código y consigue publicarlo en dicha página web, y otro usuario visita esta página, el código se ejecutará en su máquina, permitiendo al primer usuario conseguir información sensible, como las credenciales de inicio de sesión del segundo.
- **SQL Injection:** Este ataque también se aprovecha de los elementos web que permiten introducir texto al usuario, pero en esta ocasión el lenguaje es SQL. El objetivo principal de este ataque es manipular la base de datos con la que interactúa la aplicación web, pudiendo borrar datos de la misma, obtener información privada o alterarla de algún modo ilegítimo.

Para más información sobre ataques de inyección de código, consultar las referencias [10], [11] y [Ray10]

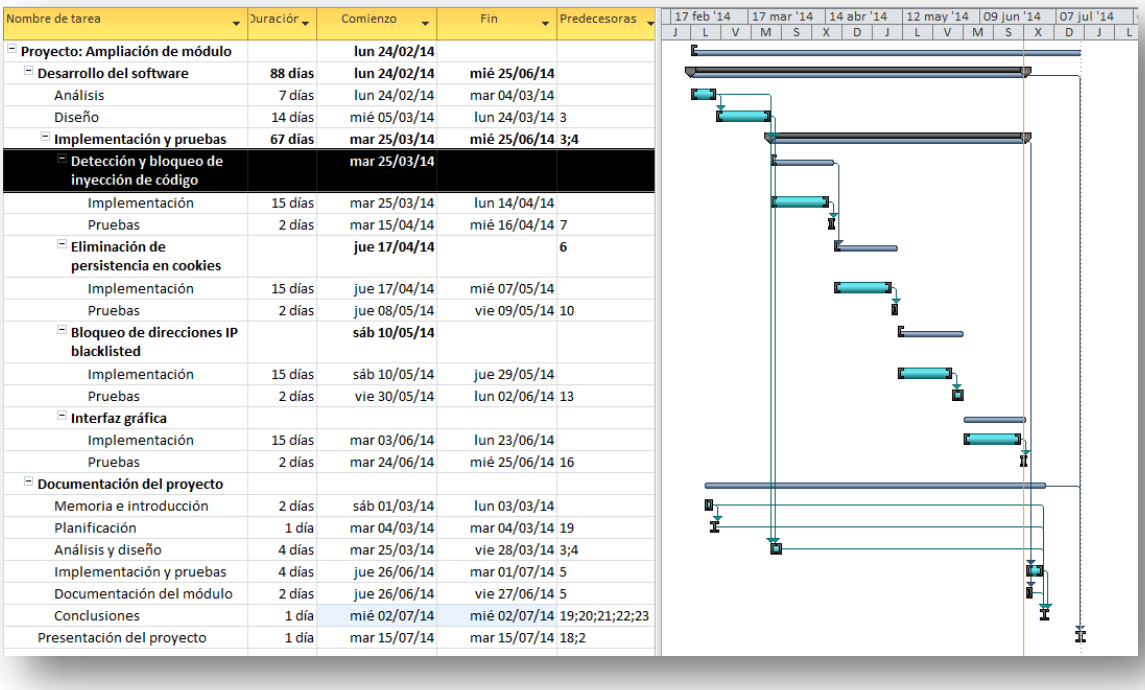
Capítulo 4. Planificación del Proyecto y Resumen de Presupuestos

4.1 Planificación

El proyecto se inicia a finales de febrero de 2014. El plazo límite para su finalización es la primera semana de julio del mismo año, aunque la presentación del mismo es a mediados de dicho mes.

La metodología a seguir es en cascada, con fases definidas de análisis, diseño, implementación y pruebas. Sin embargo, dada la naturaleza del proyecto, durante las fases de implementación y pruebas se seguirá una metodología ágil, en la que se desarrollará una funcionalidad del módulo y se probará completamente antes de pasar a desarrollar la siguiente funcionalidad.

Para ilustrar la planificación del proyecto, a continuación se muestra un diagrama de Gantt descriptivo.



Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
Proyecto: Ampliación de módulo		**lun 24/02/14**		
Desarrollo del software	**88 días**	**lun 24/02/14**	**mié 25/06/14**	
Análisis	7 días	lun 24/02/14	mar 04/03/14	
Diseño	14 días	mié 05/03/14	lun 24/03/14	3
Implementación y pruebas	**67 días**	**mar 25/03/14**	**mié 25/06/14**	**3;4**
Detección y bloqueo de inyección de código		**mar 25/03/14**		
Implementación	15 días	mar 25/03/14	lun 14/04/14	
Pruebas	2 días	mar 15/04/14	mié 16/04/14	7
Eliminación de persistencia en cookies		**jue 17/04/14**		**6**
Implementación	15 días	jue 17/04/14	mié 07/05/14	
Pruebas	2 días	jue 08/05/14	vie 09/05/14	10
Bloqueo de direcciones IP blacklisted		**sáb 10/05/14**		
Implementación	15 días	sáb 10/05/14	jue 29/05/14	
Pruebas	2 días	vie 30/05/14	lun 02/06/14	13
Interfaz gráfica				
Implementación	15 días	mar 03/06/14	lun 23/06/14	
Pruebas	2 días	mar 24/06/14	mié 25/06/14	16
Documentación del proyecto				
Memoria e introducción	2 días	sáb 01/03/14	lun 03/03/14	
Planificación	1 día	mar 04/03/14	mar 04/03/14	19
Análisis y diseño	4 días	mar 25/03/14	vie 28/03/14	3;4
Implementación y pruebas	4 días	jue 26/06/14	mar 01/07/14	5
Documentación del módulo	2 días	jue 26/06/14	vie 27/06/14	5
Conclusiones	1 día	mié 02/07/14	mié 02/07/14	19;20;21;22;23
Presentación del proyecto	1 día	mar 15/07/14	mar 15/07/14	18;2

Ilustración 4-1: Diagrama de Gantt sobre la planificación del proyecto

4.2 Resumen del Presupuesto

Dado que el proyecto es un trabajo de fin de máster universitario, el presupuesto mostrado a continuación no es real en el sentido de que no va a ser presentado a ningún cliente. No obstante es incluido en el presente documento como muestra de lo que costaría en una situación laboral real, suponiendo la existencia de un cliente que contrata al alumno para el desarrollo de este proyecto.

Los costes presentados han sido calculados en función al tiempo invertido en horas al proyecto, y a los honorarios por hora estimados para un ingeniero informático.

Item	Concepto	Cantidad	Precio Unitario	TOTAL
0	*Desarrollo software: Módulo de seguridad para IIS 7*	1,00	8.100,00 €	8.100,00 €
1	*Manuales y documentación*	3,00	60,00 €	180,00 €
	…		*Subtotal*	8.280,00 €
			IVA (21%)	1.738,80 €
			TOTAL	**10.018,80 €**

Capítulo 5. Análisis

5.1 Definición del Sistema

5.1.1 Determinación del Alcance del Sistema

El alcance del proyecto comprenderá el desarrollo de un módulo IIS que disponga de las siguientes funcionalidades, ya descritas en el Capítulo 1:

- Detección y bloqueo de ataques de inyección de código. Los elementos web a analizar serán los siguientes:
    - Elementos de formulario
    - URL
    - Cookies
    - Además, se proporcionará una forma de que el usuario personalice las reglas a aplicar en la validación del formato de los elementos anteriores.
- Bloqueo de clientes web potencialmente sospechosos en base a su dirección IP.
    - Se proporcionará además un sistema por el que avisar al usuario del módulo de que una determinada dirección IP registrada en lista negra, ha intentado realizar repetidos accesos al servidor.
- Eliminación de la persistencia en las cookies enviadas del cliente al servidor.
- Desarrollo de una interfaz de usuario para el manejo de las diferentes funcionalidades.

Se omitirán del alcance los siguientes aspectos:

- Independencia total del servidor web. El módulo a desarrollar será una extensión del servidor IIS, no siendo utilizable por ningún otro servidor.
- Análisis de la integridad de los scripts ejecutados en el cliente y enviados al servidor.

5.2 Requisitos del Sistema

5.2.1 Obtención de los Requisitos del Sistema

El cliente solicita un módulo integrable en el servidor web IIS que implemente una serie de opciones de seguridad:

- Protección frente ataques de inyección de código, principalmente SQL Injection y Cross-site scripting.
- Cierto grado de personalización de las reglas que permiten al servidor aceptar o rechazar una petición HTTP.
- Opción de eliminar la persistencia en las cookies para limitar el número de accesos ilegítimos a cuentas de sus clientes.
- Detección de intentos de acceso desde direcciones IP potencialmente peligrosas.

Además pide que el módulo venga acompañado de un interfaz gráfico a través del cual se puedan configurar las funcionalidades anteriores.

5.2.1.1 Requisitos Funcionales

Código	Nombre Requisito	Descripción del Requisito
R1	Acoplarse al pipeline de procesamiento de IIS 7	El módulo ha de poder ser integrado en el pipeline de procesamiento de peticiones HTTP de IIS 7
R2.1	Detectar inyección de código en URL	Deben detectarse y bloquearse aquellas peticiones HTTP que contengan inyección de código en la URL
R2.2	Detectar inyección de código en elementos de formulario	Deben detectarse y bloquearse aquellas peticiones HTTP que contengan inyección de código en alguno de los elementos de formulario
R2.3	Detectar inyección de código en las cookies enviadas al servidor	Deben detectarse y bloquearse aquellas peticiones HTTP que contengan inyección de código en las cookies enviadas al servidor
R2.4	Permitir personalización de las reglas de validación de datos enviados por el usuario	Se debe permitir que el usuario introduzca sus propias reglas a aplicar a los elementos enviados por el usuario mediante expresiones regulares
R3	Eliminar persistencia de cookies	El módulo ha de ser capaz de eliminar la persistencia de las cookies HTTP, de forma que las convierta en cookies de sesión (no válidas una vez que se cierre el navegador web)
R4.1	Detectar IP potencialmente peligrosa	Se ha de poder detectar aquellos intentos de acceso desde máquinas con dirección IP registrada en listas negras públicas
R4.2	Notificar al usuario de intento de acceso desde IP detectada como peligrosa mediante correo electrónico	Una vez se detecten varios intentos de acceso seguidos desde una IP registrada en lista negra, se ha de mandar un correo electrónico a la dirección especificada por el usuario informando de la circunstancia

5.2.1.2 Requisitos no Funcionales

Requisitos de Usuario

- El usuario debe tener experiencia manejando el servidor IIS 7
- Aunque no es requisito esencial, el usuario debería estar familiarizado con el uso de expresiones regulares

Requisitos Tecnológicos

- La versión de IIS sobre la que se instale el módulo ha de ser la 7 o posterior

5.2.2 Identificación de Actores del Sistema

- **Usuario:** Es un actor primario y secundario de la aplicación. Primario porque es el que interactúa con el módulo especificando la configuración con la que desea que funcione, y secundario porque, en el caso de
- **Servicio de activación de procesos Windows (WAS):** Este elemento es uno de los principales componentes del servidor IIS. Actúa de actor primario iniciando el proceso que envía la petición HTTP al pipeline para su procesamiento por parte de los módulos que lo forman.
- **HTTP.sys:** Este elemento es otro componente de IIS. Es un actor secundario ya que es el elemento al que el pipeline le pasa la respuesta HTTP a enviar al cliente, generada a partir del procesamiento realizado, en el cual participa el módulo objeto del proyecto.

5.2.3 Especificación de Casos de Uso

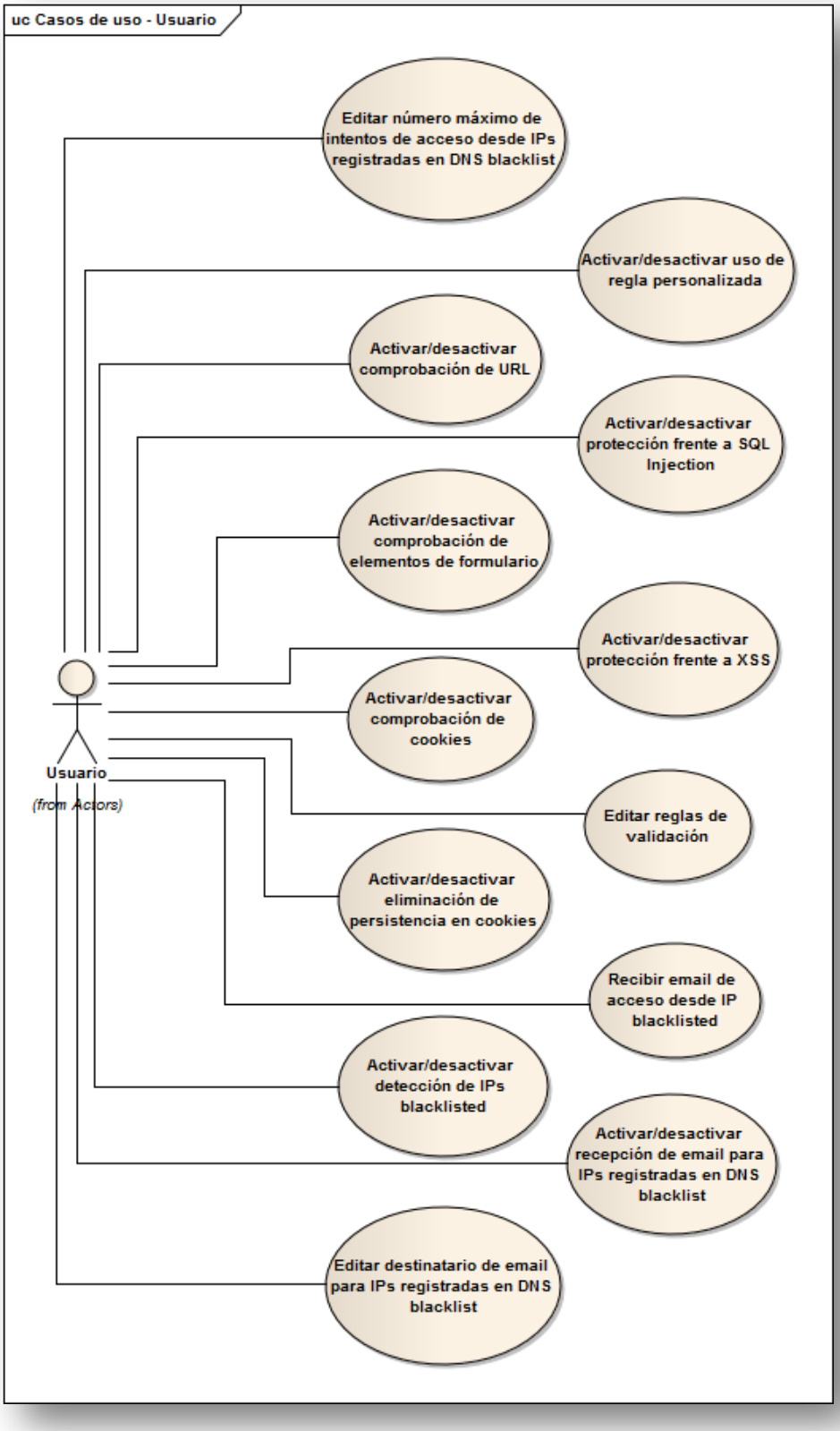


Ilustración 5-1: Diagrama de casos de uso de Usuario

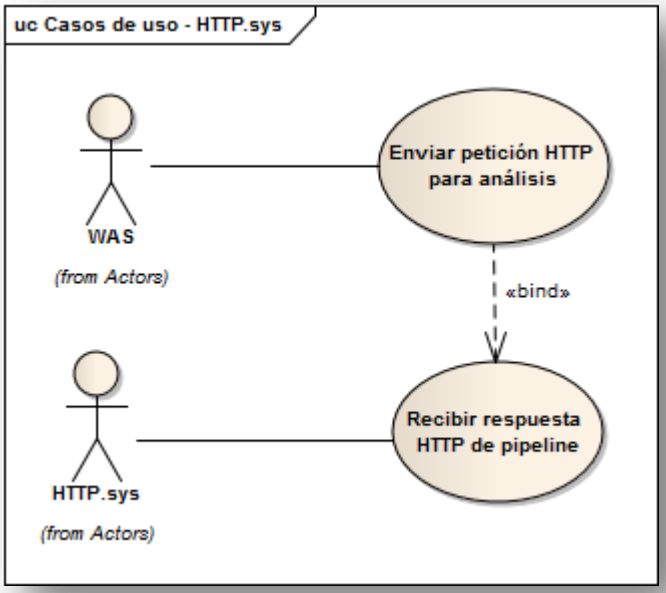


Ilustración 5-2: Diagrama de casos de uso de HTTP.sys y WAS

Nombre del Caso de Uso
Enviar petición HTTP para análisis
Descripción
El componente WAS de IIS pasará cada una de las peticiones HTTP entrantes al pipeline, en el cual se encuentra el módulo, para su análisis.

Nombre del Caso de Uso
Recibir respuesta HTTP de pipeline
Descripción
El componente HTTP.sys de IIS recibirá la respuesta HTTP generada a partir del procesamiento realizado por el pipeline, en el que se encuentra el módulo.

Nombre del Caso de Uso
Activar/desactivar comprobación de URL
Descripción
El usuario activará o desactivará desde la interfaz el checkbox correspondiente a la comprobación del formato de la URL.

Nombre del Caso de Uso
Activar/desactivar comprobación de elementos de formulario
Descripción
El usuario activará o desactivará desde la interfaz el checkbox correspondiente a la comprobación del formato de los elementos de formulario.

Nombre del Caso de Uso
Activar/desactivar comprobación de cookies
Descripción
El usuario activará o desactivará desde la interfaz el checkbox correspondiente a la comprobación del formato de las cookies.

Nombre del Caso de Uso
Activar/desactivar eliminación de persistencia en cookies
Descripción
El usuario activará o desactivará desde la interfaz el checkbox correspondiente a la eliminación de la persistencia en las cookies.

Nombre del Caso de Uso
Activar/desactivar detección de registradas en DNS blacklist
Descripción
El usuario activará o desactivará desde la interfaz el checkbox correspondiente a la comprobación en blacklists de las IPs desde las que se accede al servidor.

Nombre del Caso de Uso
Activar/desactivar protección frente a SQL Injection
Descripción
El usuario activará o desactivará desde la interfaz el checkbox correspondiente a la protección frente a ataques de SQL injection

Nombre del Caso de Uso
Activar/desactivar protección frente a XSS
Descripción
El usuario activará o desactivará desde la interfaz el checkbox correspondiente a la protección frente a ataques de CSS

Nombre del Caso de Uso
Activar/desactivar uso de regla personalizada
Descripción
El usuario introducirá una expresión regular para la validación de los diferentes elementos web en un campo de texto de la interfaz.

Nombre del Caso de Uso
Editar reglas de validación
Descripción
El usuario activará o desactivará desde la interfaz el checkbox correspondiente al uso de reglas de validación personalizadas

Nombre del Caso de Uso
Activar/desactivar recepción de email para IPs registradas en DNS blacklist
Descripción
El usuario activará o desactivará desde la interfaz el checkbox correspondiente al envío de emails para IPs registradas en DNS blacklist

Nombre del Caso de Uso
Editar destinatario de email para IPs registradas en DNS blacklist
Descripción
El usuario introducirá la nueva dirección de correo en la que quiere recibir los próximos emails de acceso desde IPs registradas en DNS blacklist

Nombre del Caso de Uso
Editar número máximo de intentos de acceso desde IPs registradas en DNS Blacklist
Descripción
El usuario establecerá el número máximo de intentos de acceso desde IPs registradas en DNS blacklist antes de mandar el email a la dirección especificada

Nombre del Caso de Uso
Recibir email de acceso desde IP blacklisted
Descripción
Tras haber superado el límite de accesos establecido para una dirección IP registrada en una blacklist, el usuario recibirá un email informando de la circunstancia.

5.3 Identificación de los Subsistemas en la Fase de Análisis

5.3.1 Descripción de los Subsistemas

La aplicación estará formada por los siguientes subsistemas:

- **Módulo:** Este subsistema es el responsable de que el módulo de IIS sea tal, y sea capaz de integrarse en el pipeline de procesamiento del mismo. En él se cargan las opciones establecidas en el fichero de configuración de la aplicación web, previamente modificado a través de la interfaz gráfica.
- **Validadores:** Los validadores son un sistema de clases que se encarga de aplicar cada una de las diferentes opciones de seguridad a cada uno de los elementos de la petición HTTP.
- **Analizador:** Está formado por una sóla clase, y su única tarea es aplicar las diferentes reglas de comprobación o protección contra ataques a los elementos HTTP. En definitiva, es la parte del sistema que decide cuándo una petición contiene o no inyección de código.
- **Interfaz gráfica:** La interfaz es una parte del sistema separada de lo que es la funcionalidad en sí del módulo. De hecho, es compilado en un ensamblado diferente, y se comunica con el módulo modificando el fichero de configuración de la aplicación web a la que afecta, el cual es accedido luego desde el módulo para recoger las opciones especificadas.

5.3.2 Descripción de los Interfaces entre Subsistemas

En esta sección cabe destacar dos aspectos en cuanto a la comunicación entre los subsistemas. El primero, la comunicación entre los componentes del módulo en sí, y el segundo, la situación del módulo como parte de un sistema mayor.

5.3.2.1 Comunicación entre Componentes del Módulo

La comunicación entre Módulo, Validadores y Analizador se reduce a mecanismos a través de los que .NET especifica la comunicación entre objetos, pues todos los subsistemas pertenecen al mismo ensamblado.

Sin embargo, la interfaz gráfica estará en un ensamblado diferente, y la comunicación con los otros tres subsistemas (más concretamente con el subsistema “Módulo”), se habrá de dar a través de un fichero de configuración único para cada aplicación web que implemente el módulo, de forma que desde la interfaz gráfica se establecerán los valores deseados en este fichero de configuración, y desde el módulo en sí se recogerán las opciones especificadas en dicho fichero.

5.3.2.2 Situación del Módulo dentro del Servidor IIS

No hay que olvidar que el módulo es una extensión de la funcionalidad de un programa mayor, el servidor IIS y, por lo tanto, conviene determinar cómo se comunican ambos elementos.

El servidor IIS utiliza una serie de componentes, o módulos, dispuestos en pipeline para resolver las peticiones HTTP que le llegan y así generar la respuesta correspondiente. Dado que el módulo objeto de este proyecto es prevenir contra ataques que puedan dañar la infraestructura de la aplicación, de las bases de datos o, incluso, del servidor, dicho módulo habrá de estar ubicado antes de que el pipeline de IIS comience a procesar los datos enviados junto con la petición HTTP.

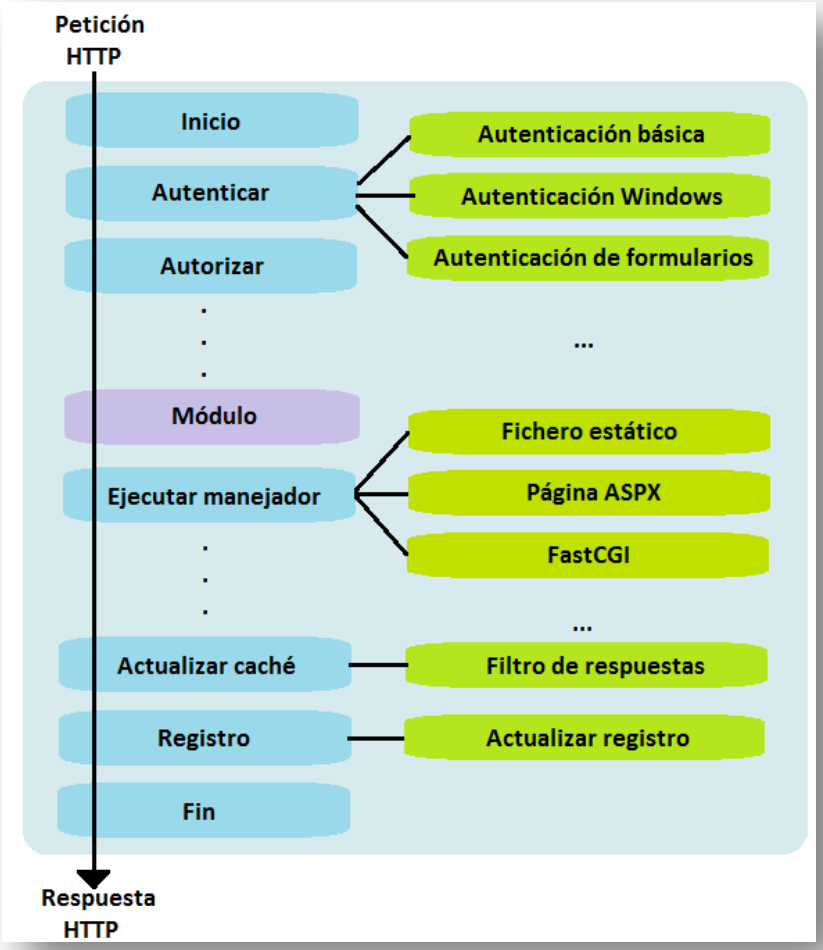


Ilustración 5-3: Situación del módulo dentro del pipeline de IIS

5.4 Diagrama de Clases Preliminar del Análisis

5.4.1 Diagrama de Clases

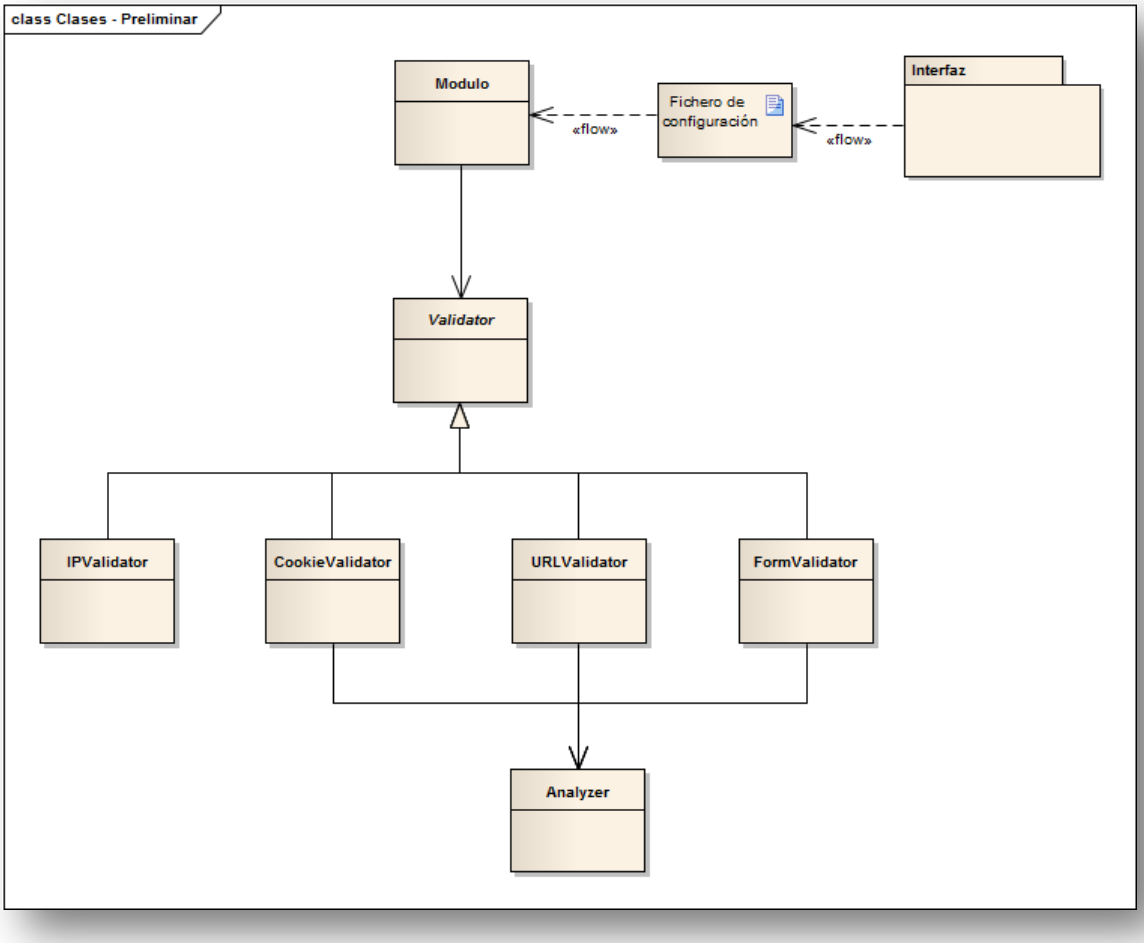


Ilustración 5-4: Diagrama de clases preliminar

5.4.2 Descripción de las Clases

5.4.2.1 Subsistema "Módulo"

Nombre de la Clase
SecurityModule
Descripción
Esta clase es la encargada de recoger la petición HTTP del pipeline de procesamiento y de comenzar a ejecutar las operaciones especificadas por el usuario desde la interfaz.
Atributos Propuestos
settings: Este atributo consistirá en un Dictionary de .NET con las reglas cargadas del fichero de configuración modificado desde la interfaz.
Métodos Propuestos
Init: Este método se encargará de capturar la petición HTTP para su análisis y de iniciar los diferentes validadores que se vayan a utilizar en dicho análisis. **Dispose:** Este método, junto con Init, es necesario para para que el módulo sea reconocido como tal por el servidor IIS. Aunque no se prevé que vaya a ser necesario, en este método se liberarían recursos utilizados por el módulo durante el procesamiento de la petición HTTP **OnPreRequestHandlerExecute**: Este método es el que sitúa la ejecución del módulo antes de que el pipeline de IIS empiece a procesar los datos enviados por el usuario, donde podría haber código inyectado. Desde él se llama a los validadores activados en ese momento para la aplicación.

5.4.2.2 Subsistema "Validadores"

Nombre de la Clase
Validator (abstracta)
Descripción
Esta clase abstracta representa cada uno de los validadores de los que estará provisto el módulo según sus diferentes funcionalidades.
Atributos Propuestos
-
Métodos Propuestos
Validate: Este método estará redefinido en cada una de las clases que hereden de Validator

Nombre de la Clase
URLValidator
Descripción
En esta clase se llevará a cabo la validación de la URL.
Atributos Propuestos
urlAnalyzer: El objeto de la clase "Analyzer" encargado de analizar la URL de cada petición HTTP. **urlWhitelistRegex:** El string que formará la expresión regular contra la cual se validará la URL de cada petición HTTP. Se le pasará al objeto anterior, "urlAnalyzer". **applyXSSProtection:** Atributo booleano que determinará si se aplica o no la expresión regular encargada de detectar ataques XSS. **applySQLIProtection:** Atributo booleano que determinará si se aplica o no la expresión regular encargada de detectar ataques SQL Injection.
Métodos Propuestos
Validate: Este método llamará al método "Analyze" del objeto "urlAnalyzer" para que valide la URL contra las reglas definidas desde la interfaz.

Nombre de la Clase
FormValidator
Descripción
En esta clase se llevará a cabo la validación de los elementos de formulario.
Atributos Propuestos
formAnalyzer: El objeto de la clase "Analyzer" encargado de analizar cada uno de los elementos de formulario de cada petición HTTP. **formWhitelistRegex:** El string que formará la expresión regular contra la cual se validará cada uno de los elementos de formulario de cada petición HTTP. Se le pasará al objeto anterior, "formAnalyzer". **applyXSSProtection:** Atributo booleano que determinará si se aplica o no la expresión regular encargada de detectar ataques XSS. **applySQLIProtection:** Atributo booleano que determinará si se aplica o no la expresión regular encargada de detectar ataques SQL Injection.
Métodos Propuestos
Validate: Este método llamará al método "Analyze" del objeto "formAnalyzer" para que valide cada uno de los elementos de formulario contra las reglas definidas desde la interfaz.

Nombre de la Clase
CookieValidator
Descripción
En esta clase se llevará a cabo la validación de las cookies HTTP. Así mismo, será la clase desde la que se elimine la persistencia de las mismas.
Atributos Propuestos
cookieAnalyzer: El objeto de la clase "Analyzer" encargado de analizar las cookies de cada petición HTTP. **cookieWhitelistRegex:** El string que formará la expresión regular contra la cual se validarán las cookies de cada petición HTTP. Se le pasará al objeto anterior, "cookieAnalyzer". **applyXSSProtection:** Atributo booleano que determinará si se aplica o no la expresión regular encargada de detectar ataques XSS. **applySQLIProtection:** Atributo booleano que determinará si se aplica o no la expresión regular encargada de detectar ataques SQL Injection.
Métodos Propuestos
Validate: Este método llamará al método "Analyze" del objeto "cookieAnalyzer" para que valide cada uno de los elementos de formulario contra las reglas definidas desde la interfaz.

Nombre de la Clase
IPValidator
Descripción
En esta clase se llevará a cabo la validación de la IP. Desde ella se harán llamadas a la API de una DNS BL pública para comprobar si la IP desde la que se realiza cada una de las peticiones HTTP está en lista negra. Además, desde esta clase se enviarán también correos electrónicos a la dirección especificada en la interfaz para informar sobre repetidos intentos de acceso desde una IP registrada en lista negra.
Atributos Propuestos
client: El objeto de la clase HttpClient de .NET que realizará accesos a la API de la DNS BL. **ipWhitelist:** Las IPs desde las que se han realizado peticiones HTTP y para las que no hay registrado ningún ataque en la DNS BL. Se crea este atributo con el objetivo de no hacer llamadas a la API para IPs que ya se ha comprobado que no están registradas en la DNS BL. **triesPerIP:** Diccionario cuyas claves son las direcciones IP para las que se ha registrado algún ataque en la DNS BL, y cuyos valores son el número de intentos de acceso a la aplicación que han realizado. **sendEmailEnabled:** Atributo booleano que dicta si se mandará o no un email cuando una determinada IP registrada en la DNS BL alcance el máximo número de intentos de acceso especificados desde la interfaz. **tries:** Entero con el número máximo de intentos de acceso para direcciones IP registradas en la lista negra antes de enviar un email a la dirección especificada en la interfaz.
Métodos Propuestos
Validate: Este método primero comprueba si la IP desde la que se realiza la petición HTTP está incluida en "ipWhitelist". Si no es así, se hace una llamada a la API de la DNS BL para comprobar si está registrada en la misma. En caso de que lo esté, se bloquea la petición y, si se ha alcanzado el máximo de intentos para esa IP, se manda un email informativo a la dirección especificada en la interfaz. En caso de que no lo esté, se añade la IP al atributo "ipWhitelist" y se continúa el procesamiento.

5.4.2.3 Subsistema "Analizador"

Nombre de la Clase
Analyzer
Descripción
Esta clase es la que aplicará las reglas especificadas desde la interfaz a través de expresiones regulares.
Atributos Propuestos
whitelistRegex: Este atributo contendrá una expresión regular con cuyo patrón tendrá que coincidir la sintaxis de cada uno de los elementos de la petición HTTP que se esté analizando. **blacklistRegexList:** Este atributo, una lista de Regex, contendrá las expresiones regulares a aplicar en cuanto a protección frente a ataques de inyección de código; en principio, XSS y SQL Injection
Métodos Propuestos
Analyze: Comprobará que la cadena de texto recibida es validada por la expresión regular del atributo "whitelistRegex", y que no coincide con ninguna de las especificadas en el atributo "blacklistRegexList". **AddBlacklistRegex:** Este método añadirá una nueva expresión regular al atributo "blacklistRegexList" con el fin de ser utilizada en la validación de cada cadena de texto que se le pase a la clase.

5.4.2.4 Subsistema "Interfaz gráfica"

Debido a que para implementar la interfaz de un módulo para IIS se necesita conocer más a fondo las clases .NET de las que es necesario heredar para que dicha interfaz se integre en el programa de administración de IIS, aún no se tiene una idea de las clases que será necesario implementar. Se definirán estas clases en la fase de diseño.

5.5 Análisis de Casos de Uso y Escenarios

5.5.1 Caso de Uso 1: "Activar/desactivar comprobación de URL"

Activar/desactivar comprobación de URL	
Precondiciones	El módulo debe estar activado para la aplicación web en cuestión
Postcondiciones	-En caso de haber activado la comprobación, las siguientes peticiones HTTP pasarán por el filtro de comprobación de la URL -En caso de haber desactivado la comprobación, las siguientes peticiones HTTP dejarán de pasar por el filtro de comprobación de la URL
Actores	Iniciado y terminado por el usuario
Descripción	El usuario: 1. Abrirá el programa de administración de IIS 2. Seleccionará la aplicación web para la que quiere activar o desactivar la comprobación 3. En la zona central del programa, seleccionará el icono del módulo 4. En la pestaña "URL" activará o desactivará la casilla de "Activar validación de URL" 5. Aun en dicha pestaña, hará click en el botón "Aplicar"
Variaciones (escenarios secundarios)	-
Excepciones	**-**
Notas	-

5.5.2 Caso de Uso 2: "Activar/desactivar comprobación de elementos de formulario"

Activar/desactivar comprobación de elementos de formulario	
Precondiciones	El módulo debe estar activado para la aplicación web en cuestión
Postcondiciones	-En caso de haber activado la comprobación, las siguientes peticiones HTTP pasarán por el filtro de comprobación de elementos de formulario -En caso de haber desactivado la comprobación, las siguientes peticiones HTTP dejarán de pasar por el filtro de comprobación de elementos de formulario
Actores	Iniciado y terminado por el usuario
Descripción	El usuario: 1. Abrirá el programa de administración de IIS 2. Seleccionará la aplicación web para la que quiere activar o desactivar la comprobación 3. En la zona central del programa, seleccionará el icono del módulo 4. En la pestaña "Formulario" activará o desactivará la casilla de "Activar validación de elementos de formulario" 5. Aun en dicha pestaña, hará click en el botón "Aplicar"
Variaciones (escenarios secundarios)	-
Excepciones	**-**
Notas	-

5.5.3 Caso de Uso 3: "Activar/desactivar comprobación de cookies"

Activar/desactivar comprobación de cookies	
Precondiciones	El módulo debe estar activado para la aplicación web en cuestión
Postcondiciones	-En caso de haber activado la comprobación, las siguientes peticiones HTTP pasarán por el filtro de comprobación de cookies -En caso de haber desactivado la comprobación, las siguientes peticiones HTTP dejarán de pasar por el filtro de comprobación de las cookies
Actores	Iniciado y terminado por el usuario
Descripción	El usuario: 1. Abrirá el programa de administración de IIS 2. Seleccionará la aplicación web para la que quiere activar o desactivar la comprobación 3. En la zona central del programa, seleccionará el icono del módulo 4. En la pestaña "Cookies" activará o desactivará la casilla de "Activar validación de cookies" 5. Aun en dicha pestaña, hará click en el botón "Aplicar"
Variaciones (escenarios secundarios)	-
Excepciones	**-**
Notas	-

5.5.4 Caso de Uso 4: "Activar/desactivar eliminación de persistencia en cookies"

Activar/desactivar eliminación de persistencia en cookies	
Precondiciones	-El módulo debe estar activado para la aplicación web en cuestión
Postcondiciones	-En caso de haber activado la eliminación, las cookies enviadas por el cliente en la petición HTTP que tuviesen establecida una caducidad, dejarán de tenerla. -En caso de haber desactivado la eliminación, la caducidad de las cookies enviadas por el cliente en la petición HTTP se dejará intacta.
Actores	Iniciado y terminado por el usuario
Descripción	El usuario: 1. Abrirá el programa de administración de IIS 2. Seleccionará la aplicación web para la que quiere activar o desactivar la comprobación 3. En la zona central del programa, seleccionará el icono del módulo 4. En la pestaña "Cookies" activará o desactivará la casilla de "Eliminar persistencia en cookies" 5. Aun en dicha pestaña, hará click en el botón "Aplicar"
Variaciones (escenarios secundarios)	-
Excepciones	**-**
Notas	-

5.5.5 Caso de Uso 5: "Activar/desactivar comprobación de IPs registradas en DNS blacklist"

Activar/desactivar comprobación de IPs registradas en DNS blacklist	
Precondiciones	El módulo debe estar activado para la aplicación web en cuestión
Postcondiciones	-En caso de haber activado la comprobación, las IPs de las que procedan las siguientes peticiones HTTP serán comprobadas a través de la API de una DNS BL. -En caso de haber desactivado la comprobación, las IPs de las que procedan las siguientes peticiones HTTP dejarán de ser comprobadas.
Actores	Iniciado y terminado por el usuario
Descripción	El usuario: 1. Abrirá el programa de administración de IIS 2. Seleccionará la aplicación web para la que quiere activar o desactivar la comprobación 3. En la zona central del programa, seleccionará el icono del módulo 4. En la pestaña "IP check" activará o desactivará la casilla de "Activar comprobación de IP contra DNS Blacklist" 5. Aun en dicha pestaña, hará click en el botón "Aplicar"
Variaciones (escenarios secundarios)	-
Excepciones	**-**
Notas	-

5.5.6 Caso de Uso 6: "Activar/desactivar protección frente a SQL Injection"

Activar/desactivar protección frente a SQL Injection	
Precondiciones	-El módulo debe estar activado para la aplicación web en cuestión -La casilla de comprobación para el elemento web en cuestión debe estar activada
Postcondiciones	-En caso de haber activado la protección, las siguientes peticiones HTTP pasarán por el filtro de comprobación de SQL Injection para el elemento web correspondiente. -En caso de haber desactivado la comprobación, las siguientes peticiones HTTP dejarán de pasar por el filtro de comprobación de SQL Injection para el elemento web correspondiente.
Actores	Iniciado y terminado por el usuario
Descripción	El usuario: 1. Abrirá el programa de administración de IIS 2. Seleccionará la aplicación web para la que quiere activar o desactivar la comprobación 3. En la zona central del programa, seleccionará el icono del módulo 4. En la pestaña "Formulario" o "Cookies" activará o desactivará la casilla de "Activar protección frente a SQLI" 5. Aun en dicha pestaña, hará click en el botón "Aplicar"
Variaciones (escenarios secundarios)	-
Excepciones	**-**
Notas	-

5.5.7 Caso de Uso 7: "Activar/desactivar protección frente a XSS"

Activar/desactivar protección frente a XSS	
Precondiciones	-El módulo debe estar activado para la aplicación web en cuestión -La casilla de comprobación para el elemento web en cuestión debe estar activada
Postcondiciones	-En caso de haber activado la protección, las siguientes peticiones HTTP pasarán por el filtro de comprobación de Cross-site scripting para el elemento web correspondiente. -En caso de haber desactivado la comprobación, las siguientes peticiones HTTP dejarán de pasar por el filtro de comprobación de Cross-site scripting para el elemento web correspondiente.
Actores	Iniciado y terminado por el usuario
Descripción	El usuario: 1. Abrirá el programa de administración de IIS 2. Seleccionará la aplicación web para la que quiere activar o desactivar la comprobación 3. En la zona central del programa, seleccionará el icono del módulo 4. En la pestaña "URL", "Formulario" o "Cookies" activará o desactivará la casilla de "Activar protección frente a XSS" 5. Aun en dicha pestaña, hará click en el botón "Aplicar"
Variaciones (escenarios secundarios)	-
Excepciones	**-**
Notas	-

5.5.8 Caso de Uso 8: "Activar/desactivar uso de regla personalizada"

Activar/desactivar uso de regla personalizada	
Precondiciones	-El módulo debe estar activado para la aplicación web en cuestión -La casilla de comprobación para el elemento web en cuestión debe estar activada
Postcondiciones	-En caso de haber activado el uso de regla personalizada, los elementos web para los que se haya activado esta opción, pasarán a ser validados utilizando la expresión regular especificada en el campo de texto. -En caso de haber desactivado el uso de regla personalizada, los elementos web para los que se haya activado esta opción, pasarán a ser validados utilizando la expresión regular por defecto.
Actores	Iniciado y terminado por el usuario
Descripción	El usuario: 1. Abrirá el programa de administración de IIS 2. Seleccionará la aplicación web para la que quiere activar o desactivar la comprobación 3. En la zona central del programa, seleccionará el icono del módulo 4. En la pestaña "URL", "Formulario" o "Cookies" activará o desactivará la casilla de "Regla personalizada" 5. Aun en dicha pestaña, hará click en el botón "Aplicar"
Variaciones (escenarios secundarios)	-
Excepciones	**-**
Notas	-

5.5.9 Caso de Uso 9: "Editar reglas de validación"

Editar reglas de validación	
Precondiciones	-El módulo debe estar activado para la aplicación web en cuestión -La casilla de comprobación para el elemento web en cuestión debe estar activada -Debe estar marcada la casilla "Regla personalizada" en la pestaña para cuyo elemento web se quiere editar la regla de validación.
Postcondiciones	-Para los elementos web del elemento en cuya pestaña se activó la casilla, se aplicará la nueva expresión regular.
Actores	Iniciado y terminado por el usuario
Descripción	El usuario: 1. Abrirá el programa de administración de IIS 2. Seleccionará la aplicación web para la que quiere activar o desactivar la comprobación 3. En la zona central del programa, seleccionará el icono del módulo 4. En la pestaña "URL", "Formulario" o "Cookies", introducirá la expresión regular que quiere aplicar en la validación del elemento web en cuestión 5. Aun en dicha pestaña, hará click en el botón "Aplicar"
Variaciones (escenarios secundarios)	• **Escenario Alternativo 1:** El usuario introduce una expresión regular incorrecta. o El módulo muestra un mensaje de error en cuanto el campo de texto pierde el foco. o El usuario introduce una expresión regular correcta. • **Escenario Alternativo 2:** El usuario deja el campo de texto vacío. o El módulo da por válidas todas las cadenas de texto (equivalente a la expresión regular "*")
Excepciones	**-**
Notas	-

5.5.10 Caso de Uso 10: "Activar/desactivar recepción de email para IPs registradas en DNS Blacklist"

Activar/desactivar recepción de email para IPs registradas en DNS blacklist	
Precondiciones	-El módulo debe estar activado para la aplicación web en cuestión -La casilla de comprobación de IP registrada en DNS blacklist debe estar activada
Postcondiciones	-En caso de haber activado la recepción de email, cada vez que una determinada IP registrada en DNS blacklist supere el máximo número de intentos permitido, se enviará un email informando de la circunstancia a la dirección especificada por el usuario. -En caso de haber desactivado la recepción de email, se dejará de recibir uno cada vez que una determinada IP registrada en DNS blacklist supere el máximo número de intentos permitido.
Actores	Iniciado y terminado por el usuario
Descripción	El usuario: 1. Abrirá el programa de administración de IIS 2. Seleccionará la aplicación web para la que quiere activar o desactivar la comprobación 3. En la zona central del programa, seleccionará el icono del módulo 4. En la pestaña "IP check" activará o desactivará la casilla de "Enviar email" 5. En la misma pestaña, en el campo de texto, introducirá la dirección de correo en la que quiere recibir los emails. 6. Aun en dicha pestaña, hará click en el botón "Aplicar"
Variaciones (escenarios secundarios)	• **Escenario Alternativo 1:** El usuario introduce una dirección de correo con formato incorrecto. o El módulo muestra un mensaje de error en cuanto el campo de texto pierde el foco. o El usuario introduce una dirección de correo válida.
Excepciones	**-**
Notas	-

5.5.11 Caso de Uso 11: "Editar destinatario de email para IPs registradas en DNS blacklist"

Editar destinatario de email para IPs registradas en DNS blacklist	
Precondiciones	-El módulo debe estar activado para la aplicación web en cuestión -Debe estar marcada la casilla "Enviar email" en la pestaña "IP check".
Postcondiciones	-Cuando se alcance el máximo de intentos de acceso permitidos desde una IP registrada en DNS blacklist, se enviará el correo a la nueva dirección de correo especificada por el usuario
Actores	Iniciado y terminado por el usuario
Descripción	El usuario: 1. Abrirá el programa de administración de IIS 2. Seleccionará la aplicación web para la que quiere activar o desactivar la comprobación 3. En la zona central del programa, seleccionará el icono del módulo 4. En la pestaña "IP check", introducirá la dirección de correo en la que quiere recibir los próximos emails 5. Aun en dicha pestaña, hará click en el botón "Aplicar"
Variaciones (escenarios secundarios)	• **Escenario Alternativo 1:** El usuario introduce una dirección de correo con formato incorrecto. o El módulo muestra un mensaje de error en cuanto el campo de texto pierde el foco. o El usuario introduce una dirección de correo válida.
Excepciones	**-**
Notas	-

5.5.12 Caso de Uso 12: "Editar número máximo de intentos de acceso desde IPs registradas en DNS Blacklist"

Editar número máximo de intentos de acceso desde IPs registradas en DNS blacklist	
Precondiciones	-El módulo debe estar activado para la aplicación web en cuestión -Debe estar marcada la casilla "Enviar email" en la pestaña "IP check".
Postcondiciones	-Cuando se alcance el máximo de intentos de acceso permitidos desde una IP registrada en DNS blacklist, se enviará el correo a la dirección de correo especificada por el usuario
Actores	Iniciado y terminado por el usuario
Descripción	El usuario: 1. Abrirá el programa de administración de IIS 2. Seleccionará la aplicación web para la que quiere activar o desactivar la comprobación 3. En la zona central del programa, seleccionará el icono del módulo 4. En la pestaña "IP check", seleccionará en el slider numérico el máximo de intentos. 5. Aun en dicha pestaña, hará click en el botón "Aplicar"
Variaciones (escenarios secundarios)	-
Excepciones	**-**
Notas	-

5.5.13 Caso de Uso 13: "Recibir email de acceso desde IP registrada en DNS Blacklist"

Editar número máximo de intentos de acceso desde IPs registradas en DNS Blacklist	
Precondiciones	-El módulo debe estar activado para la aplicación web en cuestión -Debe estar marcada la casilla "Enviar email" en la pestaña "IP check".
Postcondiciones	El usuario tiene un nuevo email en la bandeja de entrada de la dirección especificada en la interfaz
Actores	-Iniciado por el módulo -Terminado por el usuario
Descripción	El módulo: 1. Detecta un intento de acceso por parte de una IP registrada en DNS Blacklist que supera el máximo establecido desde la interfaz 2. Envía un correo a la dirección especificada por el usuario El usuario: 1. Recibe el email
Variaciones (escenarios secundarios)	-
Excepciones	• **La dirección de correo no existe:** o El usuario no recibe ningún email
Notas	-

5.5.14 Caso de Uso 14: "Enviar petición HTTP para análisis"

Enviar petición HTTP para análisis	
Precondiciones	El módulo debe estar activado para la aplicación o para el servidor.
Postcondiciones	El módulo procesa la petición HTTP validando sus datos.
Actores	Iniciado y terminado por WAS
Escenario principal	El Servicio de Activación de Procesos Windows (WAS): 1. Crea un proceso para resolver la petición HTTP entrante. 2. Envía la petición al proceso. 3. Al llegar la petición al módulo, éste la procesa. 4. El módulo pasa la petición al siguiente componente del pipeline para que continúe su procesamiento.
Escenarios secundarios	• **Escenario Alternativo 1**: Ya existe un proceso adecuado para la petición. o Se continúa en el paso 2. • **El módulo detecta un patrón peligroso en alguno de los elementos de la petición.** o El módulo interrumpe el procesamiento. o El módulo envía una respuesta HTTP al actor HTTP.sys con una excepción.
Excepciones	-
Notas	-

5.5.15 Caso de Uso 15: "Recibir respuesta HTTP de pipeline"

Recibir respuesta HTTP de módulo	
Precondiciones	El módulo debe estar activado para la aplicación o para el servidor. El módulo debe haber detectado un patrón peligroso en la petición HTTP.
Poscondiciones	HTTP.sys envía una respuesta comunicando el error al usuario de la aplicación.
Actores	Iniciado por módulo y terminado por HTTP.sys
Escenario principal	1. El módulo enviará una respuesta HTTP al componente HTTP.sys del servidor. 2. HTTP.sys renviará la respuesta al cliente.
Escenarios secundarios	-
Excepciones	• **HTTP.sys no responde*:*** El componente HTTP.sys del servidor no responde por un fallo interno ajeno al módulo. o El servidor se encarga de gestionar el error.
Notas	-

5.6 Análisis de Interfaces de Usuario

5.6.1 Descripción de la Interfaz

Siendo el módulo una ampliación de la funcionalidad del servidor web IIS, cabe destacar que la interfaz del mismo irá integrada en el programa de administración de dicho servidor.

Ilustración 5-5: Situación de la interfaz del módulo dentro del programa de administración de IIS 7

Como muestran las siguientes ilustraciones, la funcionalidad del módulo irá separada en pestañas según el elemento de las peticiones HTTP al que afecten.

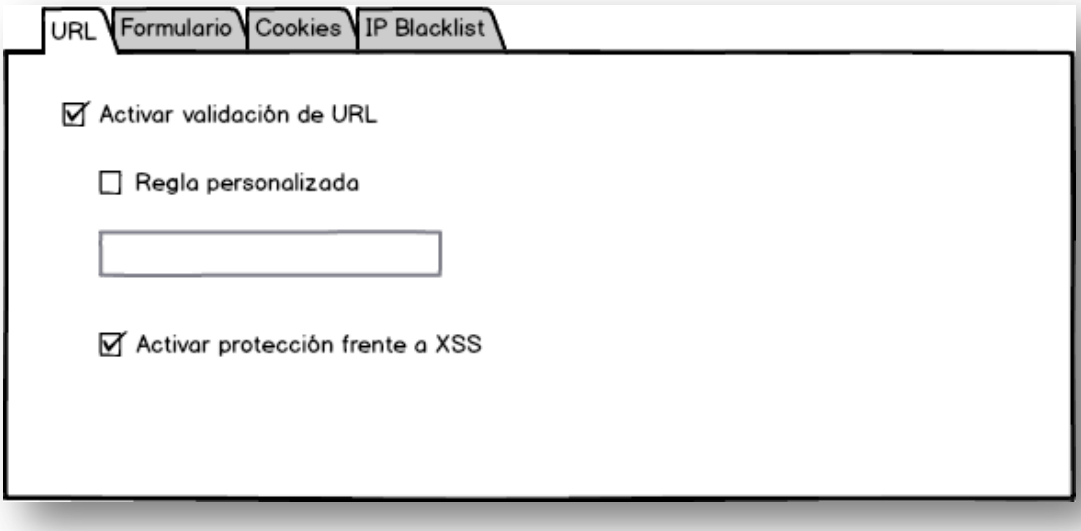


Ilustración 5-6: Interfaz - Pestaña URL

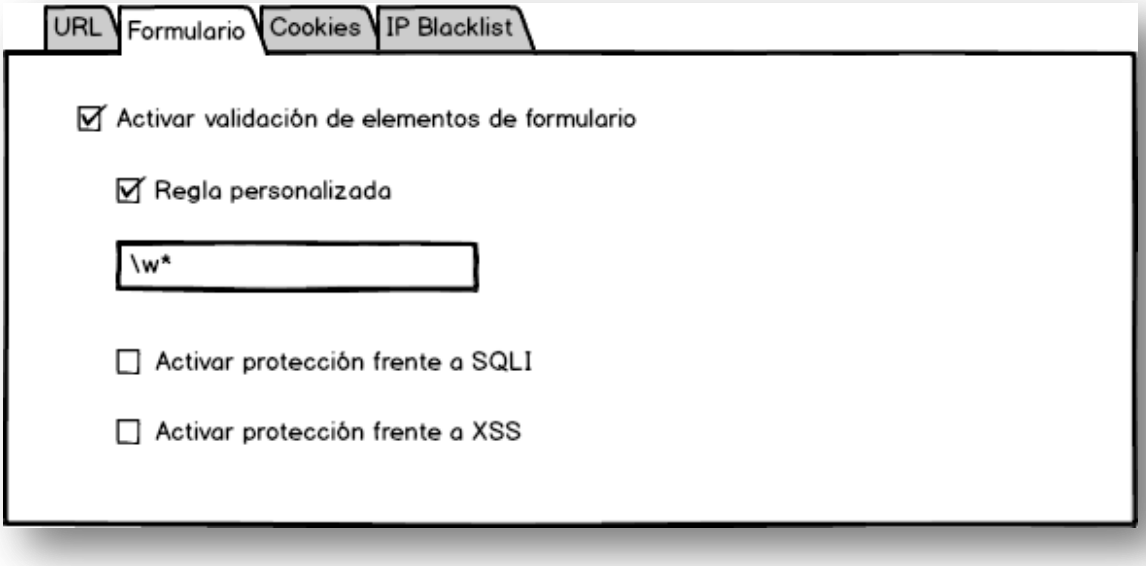


Ilustración 5-7: Interfaz - Pestaña Formulario

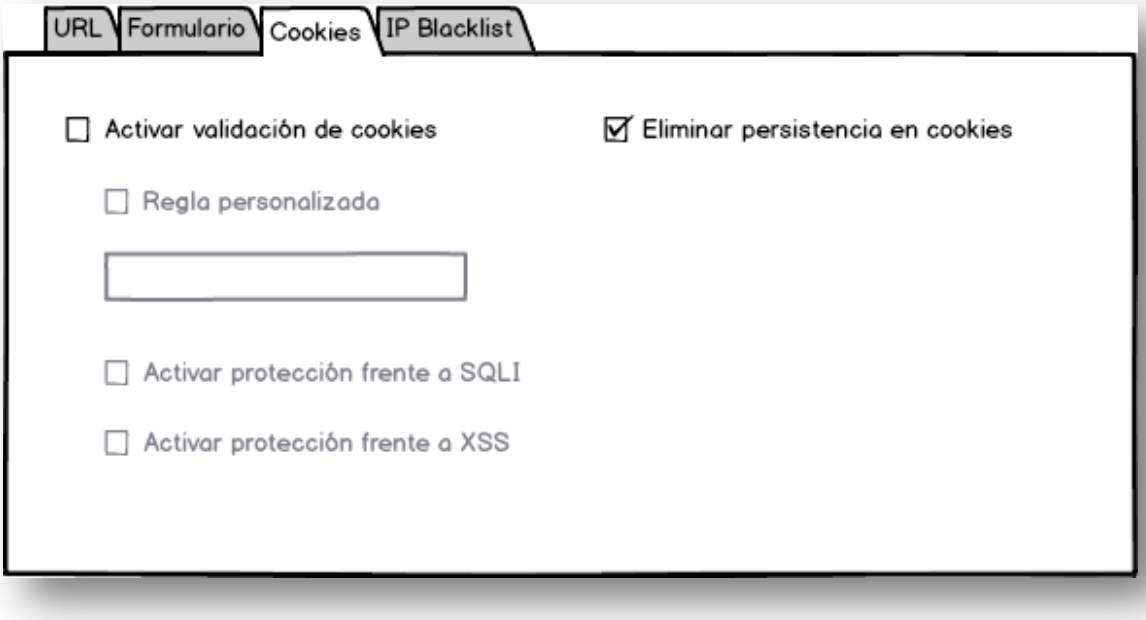


Ilustración 5-8: Interfaz - Pestaña Cookies

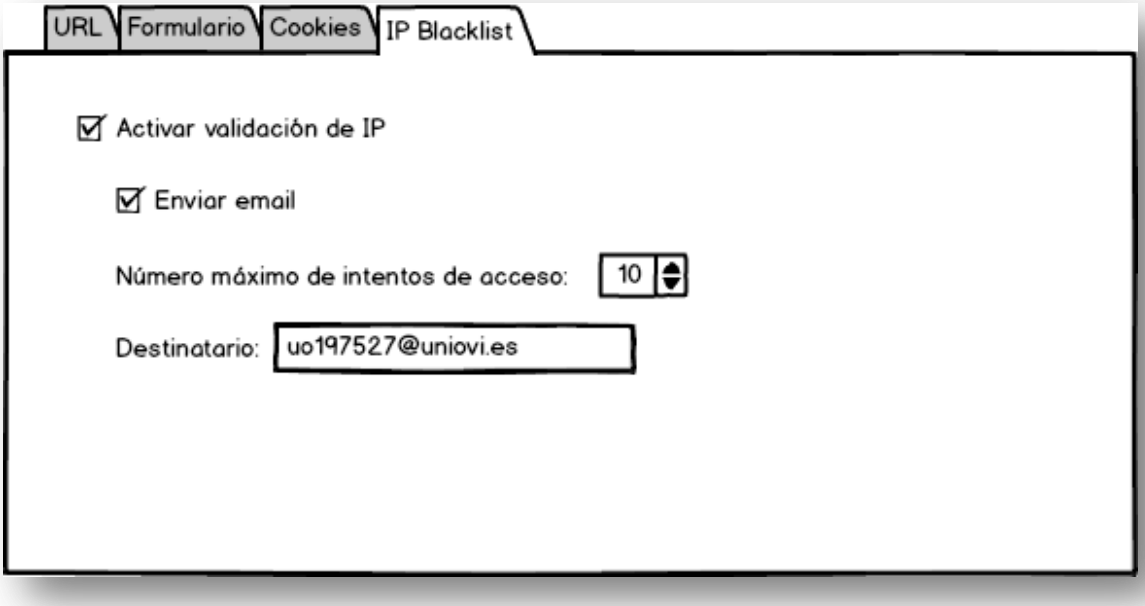


Ilustración 5-9: Interfaz - Pestaña IP Blacklist

5.6.2 Descripción del Comportamiento de la Interfaz

Como ya se ha dicho, la interfaz del módulo modificará el fichero de configuración de la aplicación web para la que esté instalado en IIS, y el módulo en sí se encargará de recoger la configuración especificada en dicho fichero.

Dada la poca complejidad de la interfaz, no se considera necesario proporcionar un menú de ayuda al usuario. Aun así, se dotará a la interfaz de cierta validación:

- Se revisarán las sintaxis de las expresiones regulares definidas en los campos de reglas personalizadas de las pestañas “URL”, “Formulario” y “Cookies”.
- En la pestaña “IP Blacklist” se comprobará que el campo de texto para el destinatario de los email siga una sintaxis de dirección de correo correcta.

5.7 Especificación del Plan de Pruebas

En esta sección se describen los diferentes tipos de pruebas que se van a llevar a cabo sobre el módulo, tanto durante su implementación como después de la misma. Como ya se dijo en el Capítulo 4, se irán implementando y probando cada funcionalidad antes de pasar a implementar la siguiente. Las pruebas especificadas en esta fase de análisis se han establecido en base a las clases y a los casos de uso identificados hasta el momento. A medida que se vaya avanzando en el desarrollo del proyecto, es probable que estas pruebas cambien. La versión definitiva de las mismas se especificará en el Capítulo 8 del presente documento.

5.7.1 Pruebas Unitarias

Se llevarán a cabo pruebas unitarias sobre cada clase del módulo cada vez que se termine de implementar una funcionalidad del mismo. Algunos de los métodos pueden ser probados mediante las pruebas unitarias de .NET proporcionadas por el Visual Studio, pero otros necesitan recibir una petición HTTP, por lo que será necesario simular accesos a una aplicación web de prueba que tenga instalada una versión funcional del módulo.

5.7.2 Pruebas de Integración

Con el objetivo de probar el correcto funcionamiento del módulo combinando varias de sus funcionalidades, estas pruebas se realizarán a partir de la segunda funcionalidad que se implemente. Se llevarán a cabo realizando peticiones HTTP a la aplicación de prueba mencionada en la sección anterior. Básicamente se probará que las diferentes funcionalidades no se entorpezcan entre sí.

5.7.3 Pruebas de Sistema

Una vez finalizada la implementación, interfaz incluida, se llevarán a cabo pruebas de sistema para comprobar que todas las funcionalidades del módulo cumplen su función de forma correcta. Se llevarán a cabo con varias aplicaciones web de prueba ejecutándose sobre la misma instancia de IIS. De la misma forma, se comprobará que la interfaz actualiza correctamente el fichero de configuración de cada aplicación y que, en caso de introducir valores erróneos en la misma, ésta avisa al usuario de la circunstancia y no le permite guardar la configuración.

5.7.4 Clasificación de Pruebas según Caso de Uso

Caso de Uso 1: Activar/desactivar comprobación de URL	
Prueba	**Resultado Esperado**
Con la comprobación desactivada, enviar petición con URL bien formada	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Con la comprobación desactivada, enviar petición con URL mal formada	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Con la comprobación activada, enviar petición con URL bien formada	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Con la comprobación activada, enviar petición con URL mal formada	Se interrumpe el procesamiento de la petición y se genera un error.

Caso de Uso 2: Activar/desactivar comprobación de elementos de formulario	
Prueba	**Resultado Esperado**
Con la comprobación desactivada, enviar petición con elementos de formulario bien formados	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Con la comprobación desactivada, enviar petición con elementos de formulario mal formados	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Con la comprobación activada, enviar petición con elementos de formulario bien formados	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Con la comprobación activada, enviar petición con elementos de formulario mal formados	Se interrumpe el procesamiento de la petición y se genera un error.

Caso de Uso 3: Activar/desactivar comprobación de cookies	
Prueba	**Resultado Esperado**
Con la comprobación desactivada, enviar petición con cookies bien formadas	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Con la comprobación desactivada, enviar petición con cookies mal formadas	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Con la comprobación activada, enviar petición con cookies bien formadas	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Con la comprobación activada, enviar petición con cookies mal formadas	Se interrumpe el procesamiento de la petición y se genera un error.

Caso de Uso 4: Activar/desactivar eliminación de persistencia en cookies	
Prueba	**Resultado Esperado**
Con la eliminación de persistencia desactivada, enviar petición con cookies persistentes	La caducidad de las cookies permanece sin cambios
Prueba	**Resultado Esperado**
Con la eliminación de persistencia desactivada, enviar petición con cookies de sesión	La caducidad de las cookies permanece sin cambios
Prueba	**Resultado Esperado**
Con la eliminación de persistencia activada, enviar petición con cookies de sesión	La caducidad de las cookies permanece sin cambios
Prueba	**Resultado Esperado**
Con la eliminación de persistencia activada, enviar petición con cookies persistentes	La caducidad de las cookies desaparece, convirtiéndose en cookies de sesión

Caso de Uso 5: Activar/desactivar comprobación de IPs registradas en DNS blacklist	
Prueba	**Resultado Esperado**
Con la comprobación desactivada, enviar petición desde IP no registrada en DNS blacklist	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Con la comprobación desactivada, enviar petición con la IP modificada para que coincida con una registrada en DNS blacklist	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Con la comprobación activada, enviar petición desde IP no registrada en DNS blacklist	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Con la comprobación activada, enviar petición con la IP modificada para que coincida con una registrada en DNS blacklist	Se interrumpe el procesamiento de la petición y se genera un error.

Caso de Uso 6: Activar/desactivar protección frente a SQL Injection	
Prueba	**Resultado Esperado**
Con la protección desactivada, enviar petición con todos los elementos web sin código SQL inyectado	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Con la protección desactivada, enviar petición con algunos elementos web con código SQL inyectado	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Con la protección activada, enviar petición con todos los elementos web sin código SQL inyectado	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Con la protección activada, enviar petición con código SQL inyectado en alguno de los elementos de formulario	Se interrumpe el procesamiento de la petición y se genera un error.
Prueba	**Resultado Esperado**
Con la protección activada, enviar petición con código SQL inyectado en algunas de las cookies	Se interrumpe el procesamiento de la petición y se genera un error.

Caso de Uso 7: Activar/desactivar protección frente a Cross-site scripting	
Prueba	**Resultado Esperado**
Con la protección desactivada, enviar petición con todos los elementos web sin código Javascript inyectado	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Con la protección desactivada, enviar petición con algunos elementos web con código Javascript inyectado	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Con la protección activada, enviar petición con todos los elementos web sin código Javascript inyectado	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Con la protección activada, enviar petición con código Javascript inyectado en la URL	Se interrumpe el procesamiento de la petición y se genera un error.
Prueba	**Resultado Esperado**
Con la protección activada, enviar petición con código Javascript inyectado en alguno de los elementos de formulario	Se interrumpe el procesamiento de la petición y se genera un error.
Prueba	**Resultado Esperado**
Con la protección activada, enviar petición con código Javascript inyectado en algunas de las cookies	Se interrumpe el procesamiento de la petición y se genera un error.

Caso de Uso 8: Activar/desactivar uso de regla personalizada	
Prueba	**Resultado Esperado**
Con la casilla desactivada, enviar petición cuyos elementos no coincida ninguno con la expresión regular especificada	Los elementos son procesados por la expresión regular por defecto
Prueba	**Resultado Esperado**
Con la casilla desactivada, enviar petición cuyos elementos coincidan algunos con la expresión regular especificada	Los elementos son procesados por la expresión regular por defecto
Prueba	**Resultado Esperado**
Con la casilla activada, enviar petición cuya URL coincida con la expresión regular especificada	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Con la casilla activada, enviar petición cuya URL no coincida con la expresión regular especificada	Se interrumpe el procesamiento de la petición y se genera un error.
Prueba	**Resultado Esperado**
Con la casilla activada, enviar petición cuyos elementos de formulario coincidan con la expresión regular especificada	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Con la casilla activada, enviar petición cuyos elementos de formulario no coincidan con la expresión regular especificada	Se interrumpe el procesamiento de la petición y se genera un error.
Prueba	**Resultado Esperado**
Con la casilla activada, enviar petición cuyas cookies coincidan con la expresión regular especificada	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Con la casilla activada, enviar petición cuyas cookies no coincidan con la expresión regular especificada	Se interrumpe el procesamiento de la petición y se genera un error.

Caso de Uso 9: Editar reglas de validación	
Prueba	**Resultado Esperado**
Enviar petición cuyos elementos web coincidan con la nueva expresión regular	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Enviar petición cuyos elementos web no coincidan con la nueva expresión regular	Se interrumpe el procesamiento de la petición y se genera un error.

Caso de Uso 10: Activar/desactivar recepción de email para IPs registradas en DNS blacklist ***Caso de Uso 13: Recibir email de acceso desde IP registrada en DNS blacklist***	
Prueba	**Resultado Esperado**
Con la opción de envío desactivada, enviar peticiones superando el número máximo de intentos con la IP modificada para que coincida con una registrada en la DNS blacklist	Se interrumpe el procesamiento de la petición y se genera un error.
Prueba	**Resultado Esperado**
Con la opción de envío activada, enviar peticiones sin superar el número máximo de intentos con la IP modificada para que coincida con una registrada en la DNS blacklist	Se interrumpe el procesamiento de la petición y se genera un error.
Prueba	**Resultado Esperado**
Con la opción de envío activada, enviar peticiones superando el número máximo de intentos con la IP modificada para que coincida con una registrada en la DNS blacklist	Se interrumpe el procesamiento de la petición, se genera un error y se envía un correo a la dirección especificada

Caso de Uso 11: Editar destinatario de email para IPs registradas en DNS blacklist	
Prueba	**Resultado Esperado**
Con la opción de envío activada, enviar peticiones superando el número máximo de intentos con la IP modificada para que coincida con una registrada en la DNS blacklist	Se interrumpe el procesamiento de la petición, se genera un error y se envía un correo a la dirección especificada

Caso de Uso 12: Editar número máximo de intentos de acceso desde IPs registradas en la DNS blacklist	
Prueba	**Resultado Esperado**
Con la opción de envío activada, enviar peticiones sin superar el nuevo número máximo de intentos con la IP modificada para que coincida con una registrada en la DNS blacklist	Se interrumpe el procesamiento de la petición y se genera un error.
Prueba	**Resultado Esperado**
Con la opción de envío activada, enviar peticiones superando el nuevo número máximo de intentos con la IP modificada para que coincida con una registrada en la DNS blacklist	Se interrumpe el procesamiento de la petición, se genera un error y se envía un correo a la dirección especificada

Caso de Uso 14: "Enviar petición HTTP para análisis" ***Caso de Uso 15: "Recibir respuesta HTTP de pipeline"***	
Prueba	**Resultado Esperado**
Con el módulo desactivado, enviar petición HTTP sin elementos web susceptibles de ser bloqueados	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Con el módulo desactivado, enviar petición HTTP con elementos web susceptibles de ser bloqueados	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Con el módulo activado, enviar petición HTTP sin elementos web susceptibles de ser bloqueados	El procesamiento de la petición HTTP sigue su curso
Prueba	**Resultado Esperado**
Con el módulo activado, enviar petición HTTP con elementos web susceptibles de ser bloqueados	Se interrumpe el procesamiento de la petición y se genera un error

Capítulo 6. Diseño del Sistema

6.1 Arquitectura del Sistema

6.1.1 Diagramas de Paquetes

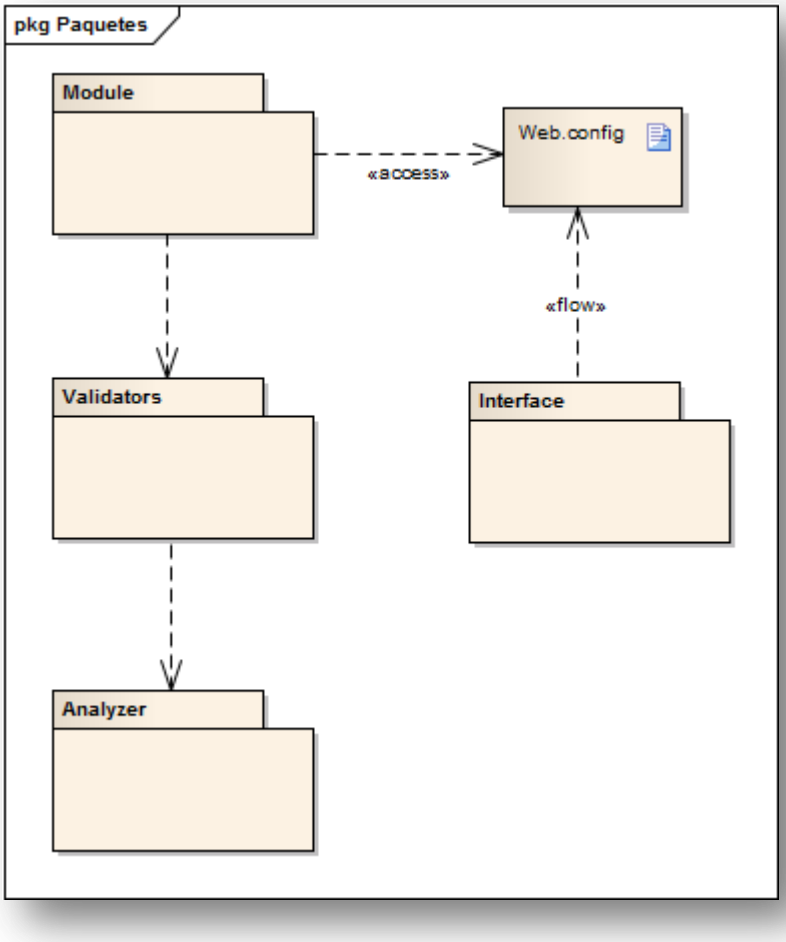


Ilustración 6-1: Diagrama de paquetes

6.1.1.1 Paquete "Interface"

En este paquete se encuentran las clases que conforman la interfaz del módulo que se integrará con el programa de administración de IIS. Este paquete actualiza la configuración para el módulo editando el fichero Web.config de la aplicación web en cuestión.

6.1.1.2 Paquete "Module"

Este paquete, formado por una única clase, es la parte del programa que se integra en el pipeline de procesamiento de IIS, recogiendo las peticiones HTTP que le llegan para su posterior análisis en función de la configuración especificada desde la interfaz, a la cual accede consultando el archivo Web.config de la aplicación web.

6.1.1.3 Paquete "Validators"

En este paquete se encuentran las clases validadoras que realizarán las operaciones pertinentes sobre las peticiones HTTP según el elemento al que afecten; URL, elementos de formulario, cookies o IP.

6.1.1.4 Paquete "Analyzer"

Este paquete contiene también una sóla clase, la cual se encarga de validar las cadenas de texto presentes en cada elemento web contra las expresiones regulares que se le especifiquen.

6.1.2 Diagramas de Componentes

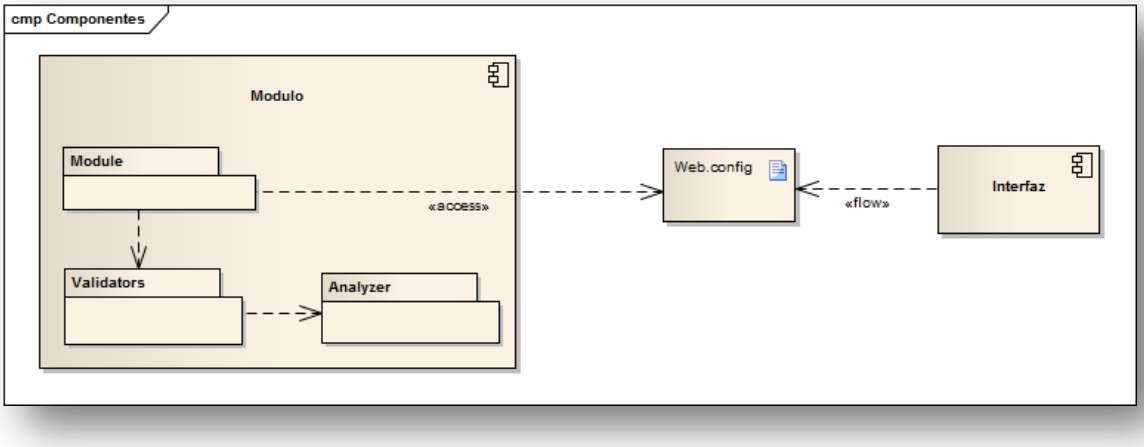


Ilustración 6-2: Diagrama de componentes

6.1.3 Diagramas de Despliegue

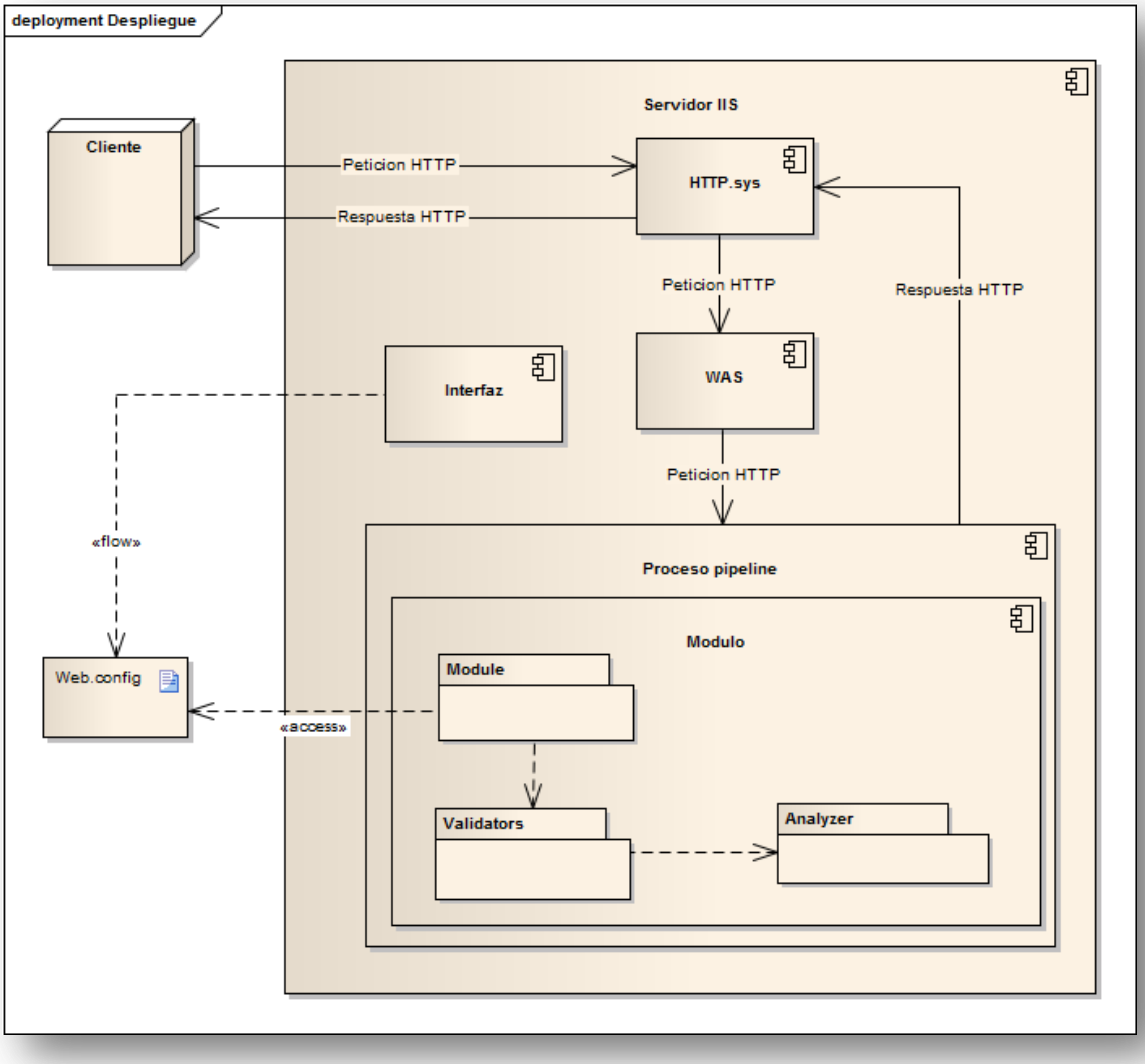


Ilustración 6-3: Diagrama de despliegue

6.1.3.1 Elemento "Cliente"

Este elemento representa la máquina desde la cual se realizan peticiones HTTP al servidor y a la que llegan las respuestas HTTP del mismo.

6.1.3.2 Elemento "Servidor IIS"

El servidor es el componente que contiene al resto de elementos que se encargan de, a partir de una petición HTTP, generar una respuesta a mandar al cliente, entre los que se encuentra el módulo objeto de este proyecto.

6.1.3.3 Elemento "HTTP.sys"

HTTP.sys es uno de los principales componentes de IIS, y es el encargado de recoger las peticiones HTTP le llegan del cliente. Asimismo, una vez que el pipeline del servidor ha generado la respuesta HTTP correspondiente, HTTP.sys es el que se encarga de mandársela al cliente.

6.1.3.4 Elemento "WAS"

Este componente inicia el proceso pipeline con los módulos adecuados para procesar la petición HTTP que le llega de HTTP.sys. En caso de que ya exista un proceso capaz de procesar la petición que recibe, WAS se limita a enviársela.

6.1.3.5 Elemento "Proceso pipeline"

El pipeline es el conjunto de módulos designados por el servidor IIS que tienen la responsabilidad de procesar las peticiones HTTP que le llegan a una determinada aplicación o conjunto de aplicaciones web. Como ya se dijo en el Capítulo 3, la petición HTTP va pasando por estos módulos, donde cada uno desempeña una función en particular. Una vez ha finalizado el procesamiento de la petición por parte de todos los módulos, la respuesta HTTP obtenida es enviada a HTTP.sys para que se la reenvíe al cliente.

6.1.3.6 Elemento "Interfaz"

Este elemento es el ensamblado que se integra con el programa de administración de IIS para proporcionar una interfaz gráfica del módulo. En vez de comunicarse directamente con éste, la interfaz modifica el fichero de configuración de la aplicación web, el cual es consultado posteriormente por el módulo en sí para determinar de qué modo se va a llevar a cabo su funcionalidad.

6.1.3.7 Elemento "Web.config"

Representa el fichero de configuración de cada aplicación web. Entre otras cosas, es modificado por la interfaz del módulo y consultado por dicho módulo para aplicar unas u otras operaciones a la petición HTTP.

6.1.3.8 Elemento "Módulo"

Este elemento representa el ensamblado que lleva a cabo la funcionalidad del módulo objeto de este proyecto. Una vez le llega la petición HTTP del anterior módulo del pipeline la procesa, y una vez termina se la pasa al siguiente.

6.2 Diseño de Clases

6.2.1 Diagramas de Clases

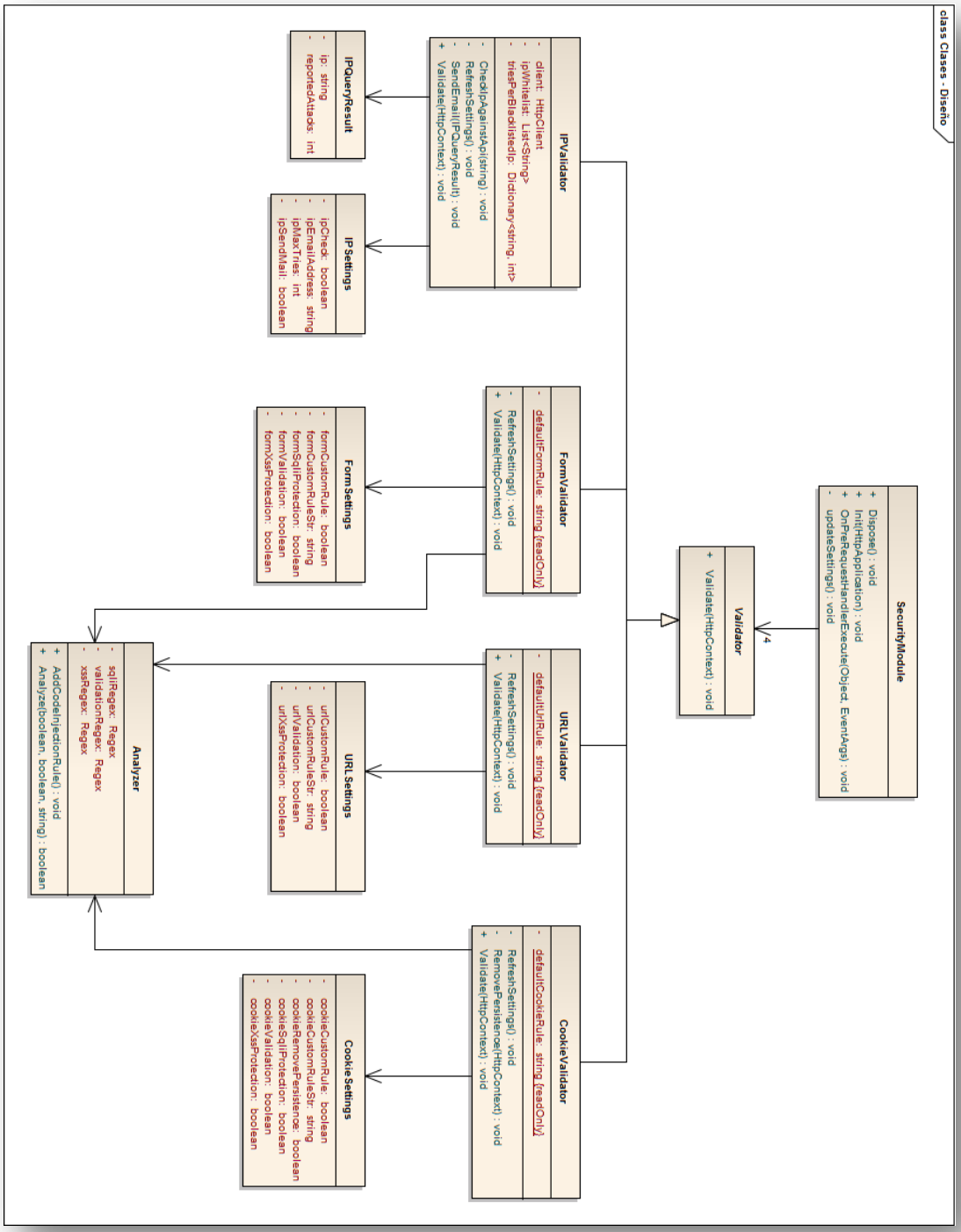


Ilustración 6-4: Diagrama de clases del módulo

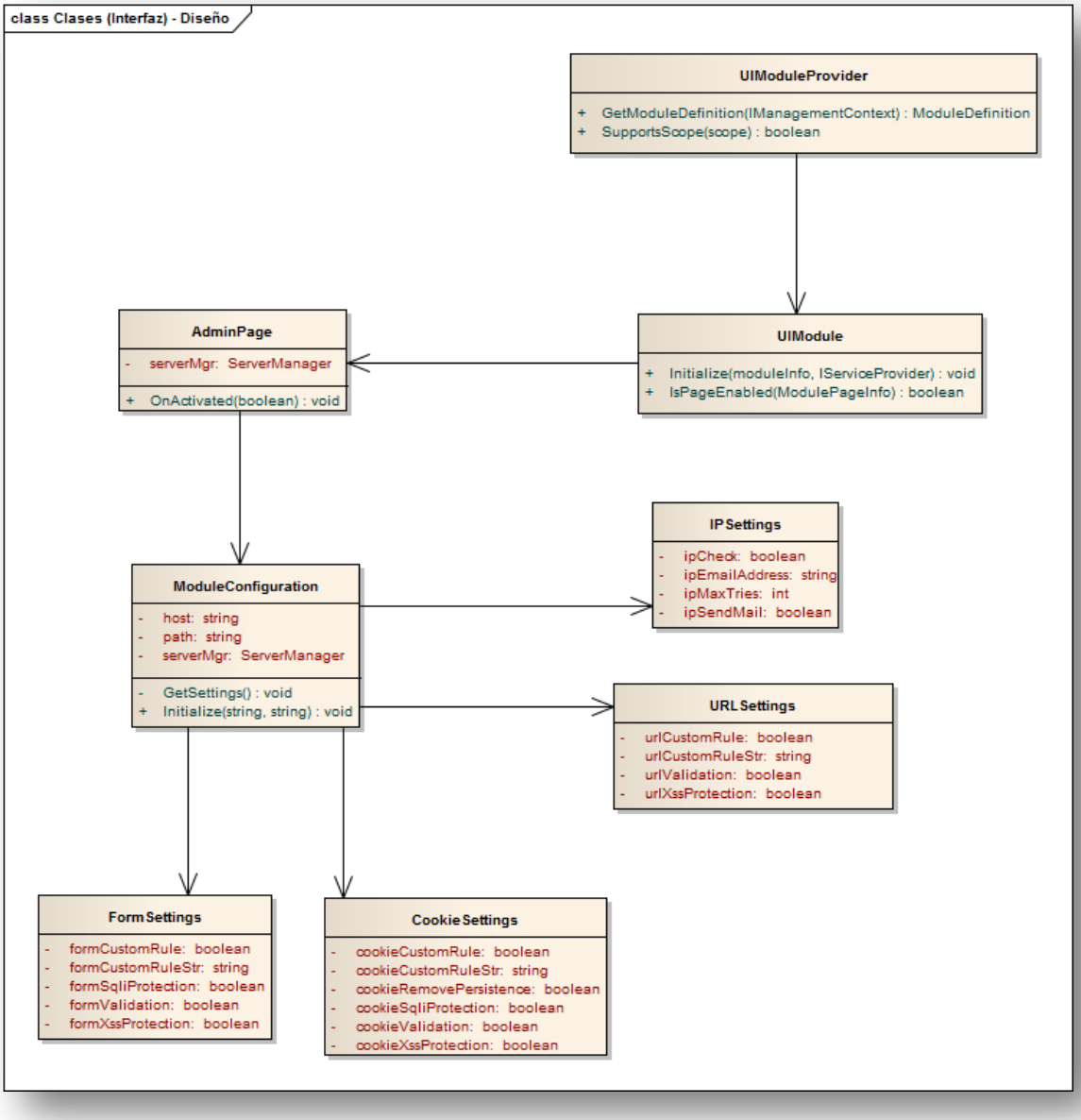


Ilustración 6-5: Diagrama de clases de la interfaz del módulo

6.3 Diagramas de Secuencia

En esta sección se muestran los diagramas de secuencia de cada caso de uso que ilustran las operaciones llevadas a cabo por los diferentes actores y elementos del sistema para desarrollar uno u otro caso de uso. La mayor parte de ellos consisten en la activación o desactivación de opciones desde la interfaz por parte del usuario, y las operaciones son siempre muy similares. Por ello, en cuanto a operación con la interfaz se refiere, sólo se incluyen dos diagramas de secuencia para representar los siguientes casos generales:

- Activación/desactivación de opción de módulo
- Edición de parámetros de módulo

En cuanto a los diagramas que demuestran el funcionamiento del módulo integrado con el servidor IIS son dos, correspondientes a los siguientes casos:

- Enviar/recibir petición HTTP
- Analizar petición

6.3.1 Activación/desactivación de opción de módulo

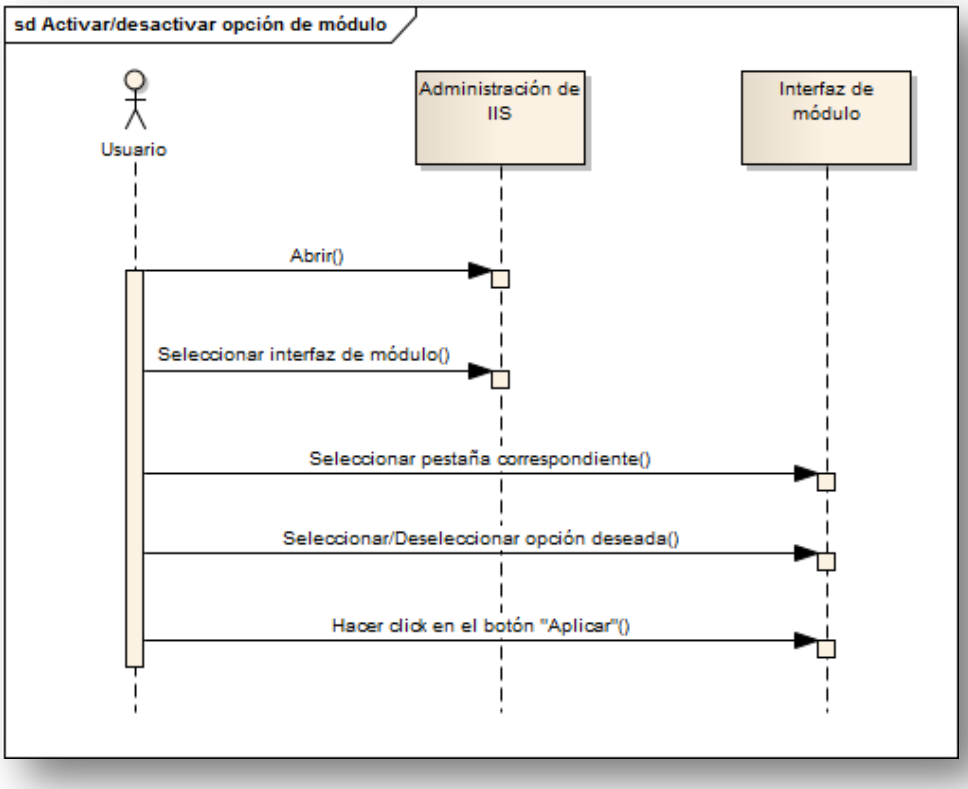


Ilustración 6-6: Diagrama de secuencia para casos de uso relativos a la activación/desactivación de opciones desde la interfaz

6.3.2 Edición de parámetros de módulo

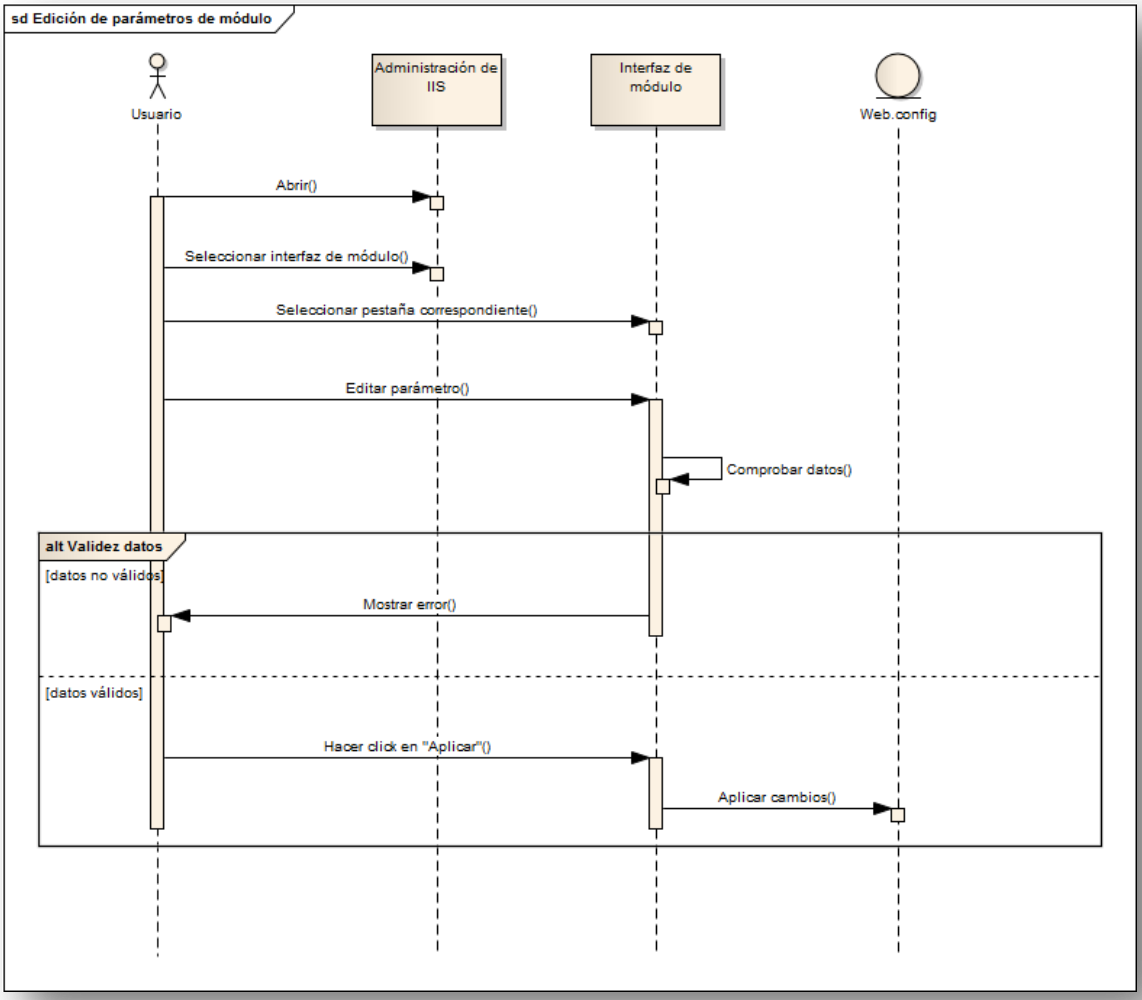


Ilustración 6-7: Diagrama de secuencia para casos de uso relativos a la edición de parámetros desde la interfaz

6.3.3 Enviar/recibir petición HTTP

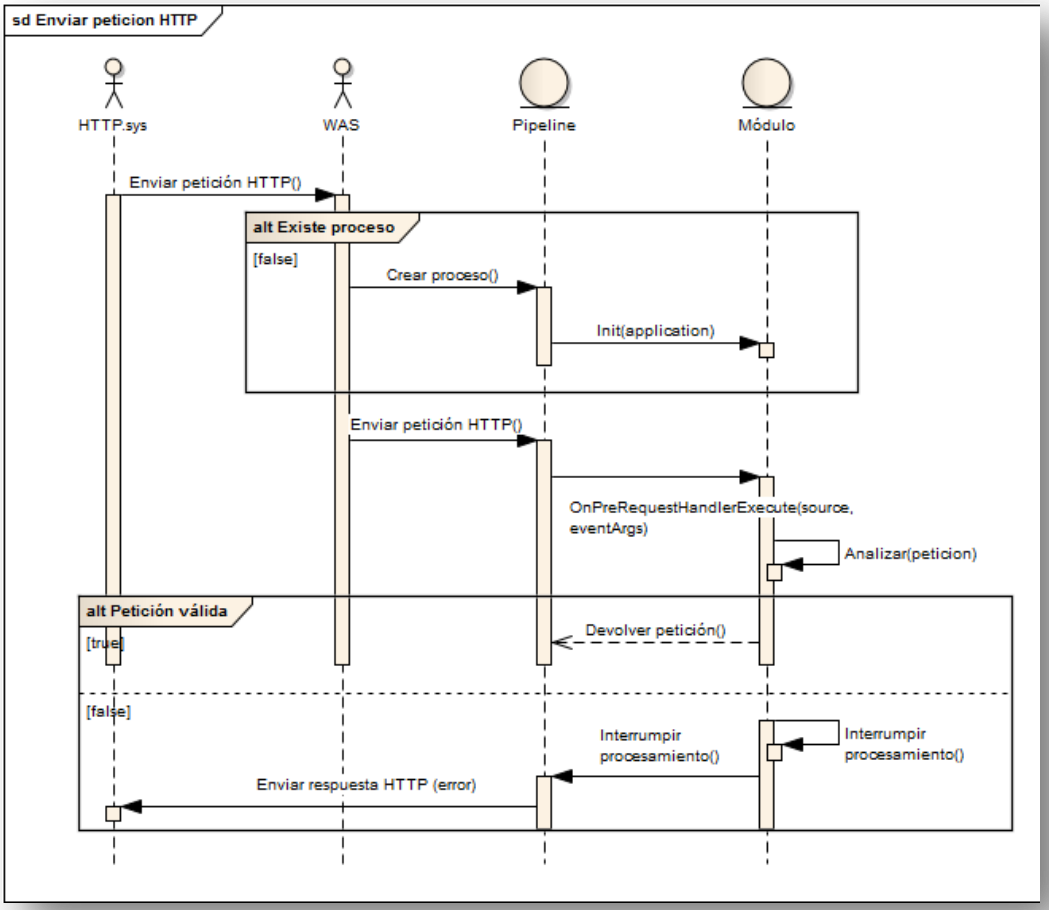


Ilustración 6-8: Diagrama de secuencia del envío y la recepción de una petición HTTP al módulo y desde el módulo

6.3.4 Analizar petición HTTP

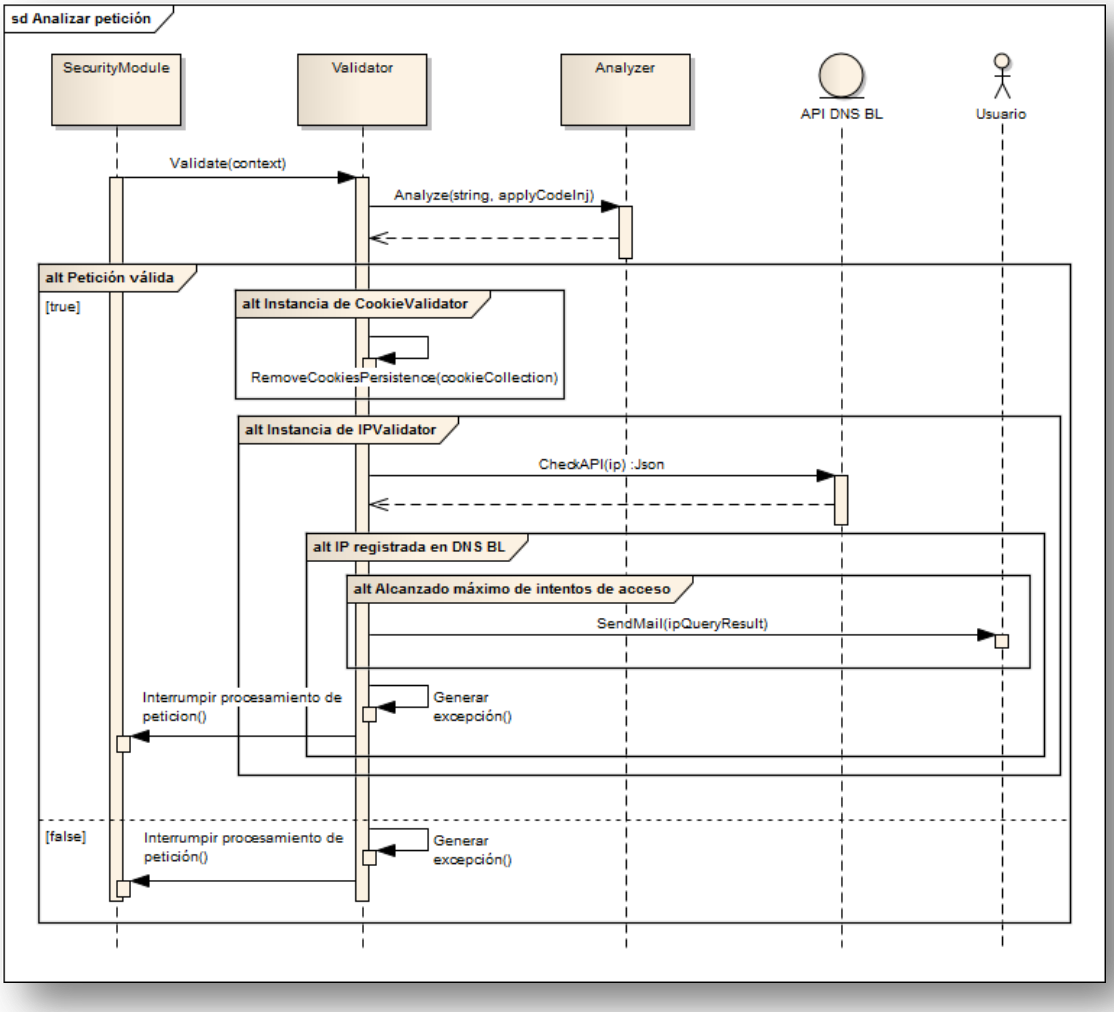


Ilustración 6-9: Diagrama de secuencia del procesamiento de una petición HTTP por parte del módulo

6.4 Diagramas de Actividades

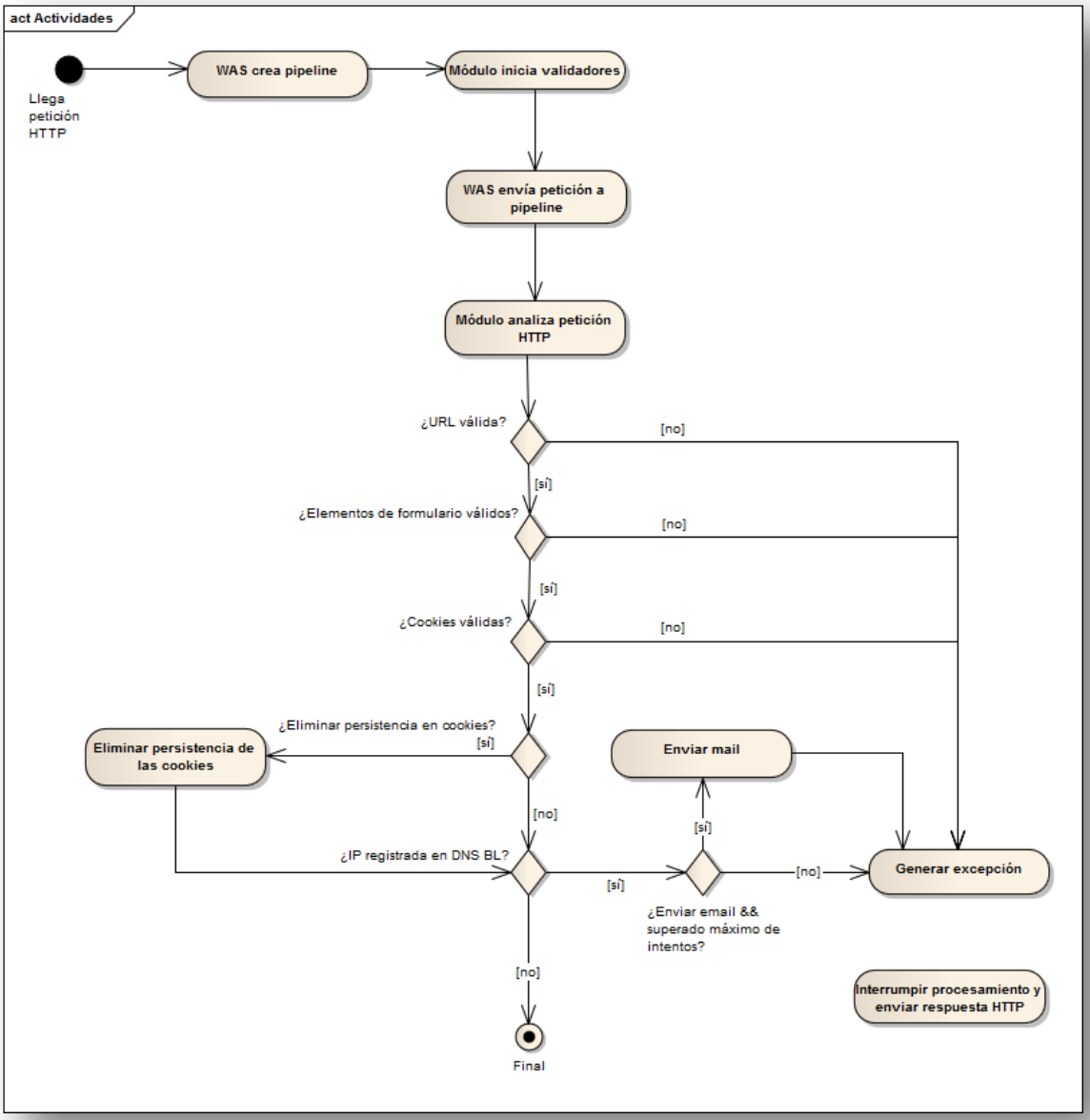


Ilustración 6-10: Diagrama de actividades del procesamiento de una petición HTTP por parte del módulo

6.5 Diseño de la Interfaz

Como ya se mostró en la fase de análisis, la interfaz del módulo estará dividida en pestañas según los distintos elementos que comprobará el mismo.

En el caso del campo de texto para la regla personalizada de las pestañas “URL”, “Formulario” y “Cookies”, y para el campo “Destinatario” de la pestaña “IP”, la interfaz proporcionará cierto grado de validación; en el caso de las reglas personalizadas, se comprobará que sean expresiones regulares válidas, y en el caso del campo “Destinatario”, se comprobará que el texto introducido sigue el formato válido de una dirección email.

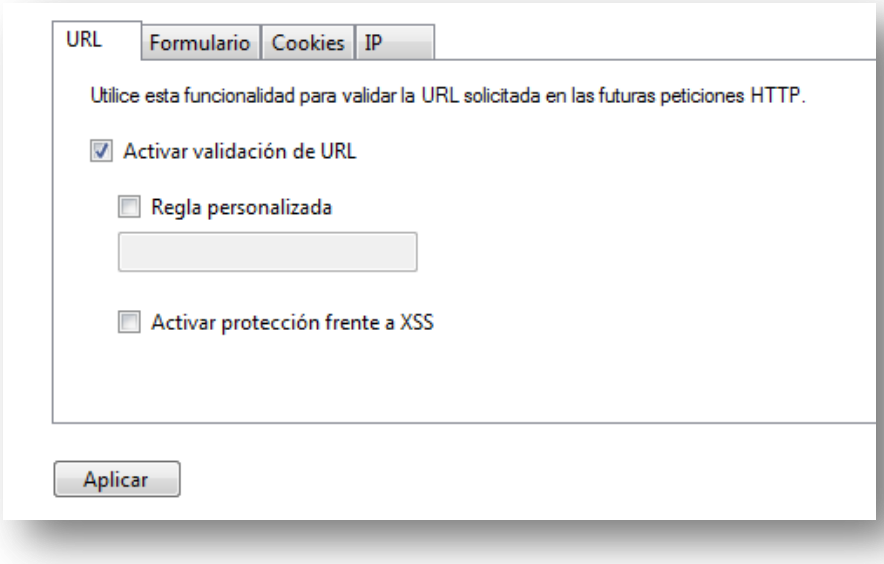


Ilustración 6-11: Pestaña URL

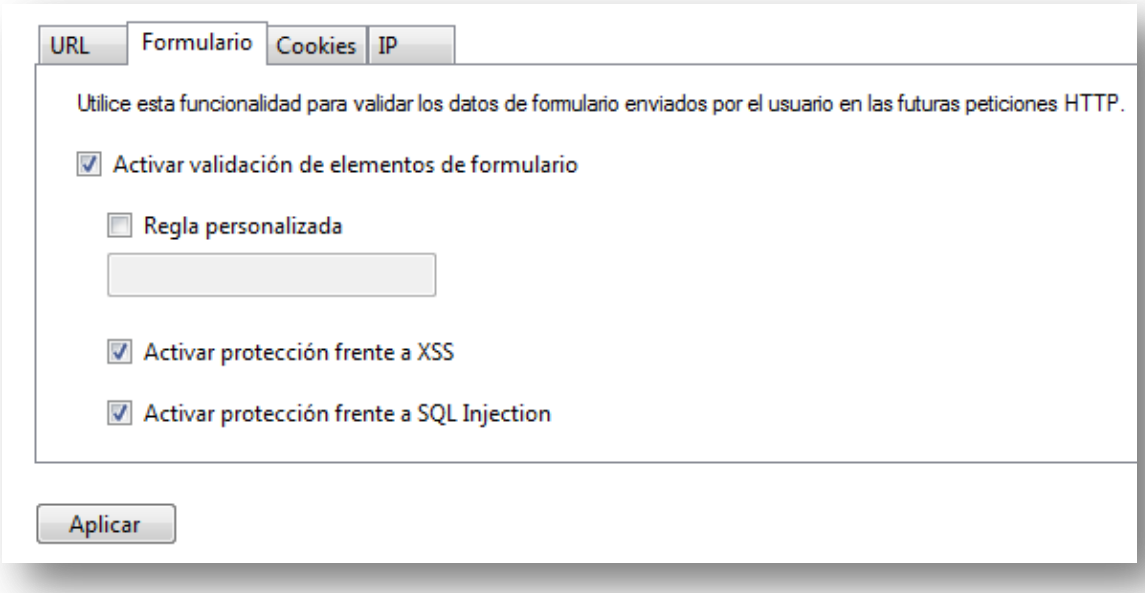


Ilustración 6-12: Pestaña Formulario

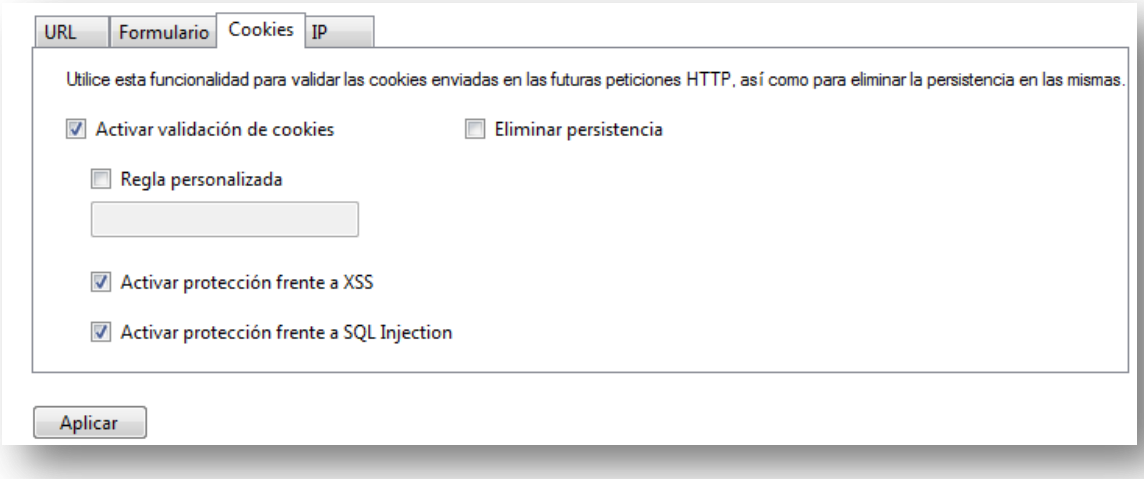


Ilustración 6-13: Pestaña Cookies

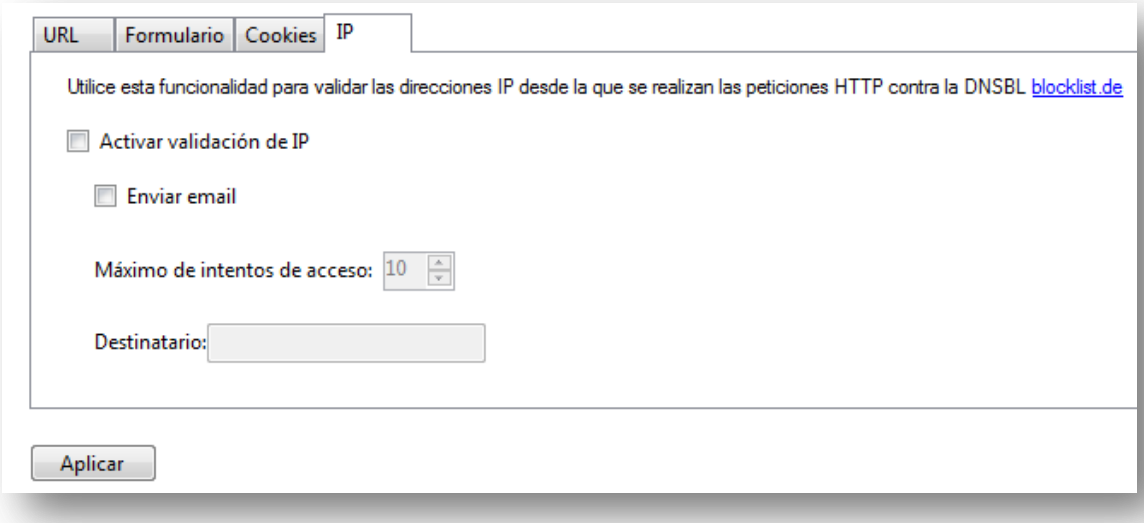


Ilustración 6-14: Pestaña IP

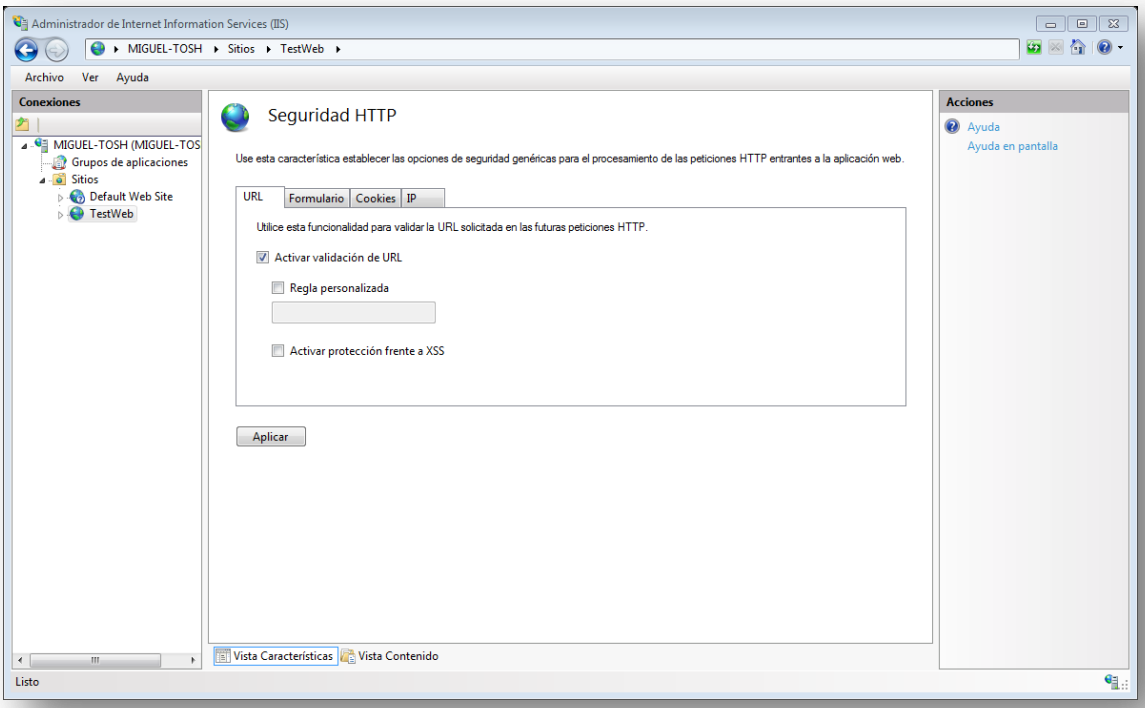


Ilustración 6-15: Vista general de la interfaz integrada en el administrador de IIS

6.6 Especificación Técnica del Plan de Pruebas

Las pruebas se realizarán bajo un sistema operativo Windows 7 Home Premium. Como ya se dijo en el Capítulo 5, parte de las pruebas se realizarán haciendo peticiones al servidor IIS. En esos casos, cliente y servidor serán la misma máquina (localhost).

Las pruebas se realizarán a medida que se vayan desarrollando funcionalidades. Estas pruebas serán unitarias una vez desarrollada la primera funcionalidad, y unitarias y de integración por cada una de las siguientes funcionalidades. Una vez finalizadas todas las funcionalidades se realizarán pruebas de sistema para verificar el funcionamiento correcto de todo el sistema.

Antes de empezar con las pruebas de la funcionalidad en sí, se implementará el módulo "vacío" y se comprobará que se puede activar para una aplicación web de IIS. En cuanto a las pruebas de la funcionalidad, en caso de no superar alguna, se revisarán las clases y métodos implicados y se procederá a arreglar el error. No obstante, este error quedará reflejado en la presente documentación. Para aquellas reparaciones que impliquen modificar expresiones regulares, se repetirán aquellas pruebas que utilicen dichas expresiones, independientemente de si ya se habían superado o no.

6.6.1 Integración del módulo en IIS

En esta fase se implementará un módulo que se limite a integrarse en el pipeline de IIS y recoja las peticiones HTTP que le llegan a una determinada aplicación. La única funcionalidad del mismo será imprimir en la respuesta enviada por el servidor, los parámetros de la petición HTTP a partir de la que se generó.

A continuación se muestra una tabla con las pruebas a llevar a cabo:

Número	Prueba	Mecanismo de notificación de resultados	Resultado Esperado
1	Activar módulo	En las respuestas HTTP enviadas al cliente se imprimirá la cadena "Module enabled" al principio de la misma	Se muestra la cadena "Module enabled" al principio de la respuesta
3	Desactivar módulo	La ausencia de resultados es el mecanismo de notificación	No se muestra la cadena "Module enabled" al principio de la respuesta
7	Extraer URL de la petición	En las respuestas HTTP enviadas al cliente se imprimirá la URL solicitada al principio de la misma	Se muestra la URL al principio de la respuesta
8	Extraer elementos de formulario de la petición	En las respuestas HTTP enviadas al cliente a raíz de una petición POST, se imprimirán los datos introducidos por el usuario mediante formulario	Se muestran los datos de formulario enviados por el usuario al principio de la respuesta.

Número	Prueba	Mecanismo de notificación de resultados	Resultado Esperado
9	Extraer cookies de la petición	En las respuestas HTTP enviadas al cliente se imprimirán las cookies enviadas por el cliente.	Se muestran los valores de las cookies al principio de la respuesta
10	Extraer IP origen de la petición	En las respuestas HTTP enviadas al cliente se imprimirá la IP desde la que se envió la petición.	Se muestra la dirección IP desde la que se realizó la petición al principio de la respuesta
11	Interrumpir procesamiento de la petición	Se provocará temporalmente el lanzamiento de una excepción HTTP con código 403 en función de los elementos extraídos de la petición. En este caso, se lanzará si en alguno de los elementos de formulario se encuentra la cadena "throwException"	Se genera un error 403 y no se permite acceder al recurso web solicitado

6.6.2 Funcionalidad relativa a la URL

En esta fase se implementará y probará el análisis de las URL solicitadas en las peticiones HTTP. Este análisis incluye:

- Validación del formato de la URL
  - Validación por defecto
  - Validación especificada por el usuario: Dado que aún no se habrá implementado la interfaz, simplemente se modificará la expresión regular a aplicar por parte de la clase Analyzer
- Detección de ataques de inyección de código XSS

6.6.2.1 Pruebas Unitarias

A continuación se muestran las tablas correspondientes a las pruebas unitarias de las clases desarrolladas para esta funcionalidad. En el caso de la clase Analyzer, es usada también por los validadores de elementos de formulario y de cookies, pero dado que las expresiones regulares utilizadas por esta clase le son pasadas por los validadores, se probará una sóla vez.

Clase: Analyzer	
Método: Boolean Analyze(String str, boolean applyXss, boolean applySqli)	
Datos introducidos	**Resultado Esperado**
"http://www.paginaweb.com/", true, false	True
"https://www.paginaweb.com/", true, false	True
"http://www.paginaweb.com/elemento1.aspx" , true, false	True
"http://www.paginaweb.com/elemento1/elemento2.aspx" , true, false	True

Clase: Analyzer	
Método: Boolean Analyze(String str, boolean applyXss, boolean applySqli)	
Datos introducidos	**Resultado Esperado**
“htTP://wWw.paginaWEB.cOm/ElEmentO1/ElEmEnto2.aSpx” , true, false	True
“http://pagina4web”, true, false	True
“http://798423.com”, true, false	True
“http://www.pagina.com/operacion?v=PfapQg9iv”, true, false	True
“http://localhost”, true, false	True
“http://localhost:80”, true, false	True
“http://localhost:80/elemento.aspx”, true, false	True
“http://www.paginaweb.com<script>alert(1)</script>”, true, false	False
“http://www.paginaweb.com<sCriPt>alErt(1)</scCRIpt>”, true, false	False
“http://www.paginaweb.com\"><script>alert(document.cookie)</script>”, true, false	False
“http://www.paginaweb.com/javascript:alert(1)” , true, false	False
“http://www.paginaweb.com/';alert(1)” , true, false	False
“http://www.paginaweb.com/<elementoRaro.com”, true, false	False
Método Regex SetValidationRegex(String validationStr)	
Datos introducidos	**Resultado Esperado**
\s*	\s* (Regex)
abcd+	abcd+ (Regex)
.+	.+ (Regex)
localhost:?/[a-z0-9]+(.[a-z]+)*	localhost:?/[a-z0-9]+(.[a-z]+)* (Regex)

El método SetValidationRegex es un método void, pero con el objetivo de realizar las pruebas se hará que devuelva temporalmente la expresión regular que se acaba de construir a partir de la cadena pasada como parámetro.

Nótese que el método Validate de la clase URLValidator recibe un objeto HttpContext contenedor de la petición, y que lo único que hace es pasarle a la clase Analyzer la URL solicitada junto con un booleano indicando si se aplica o no la detección de XSS (no se aplica detección de SQLI al análisis de las URL). Por lo tanto, se probarán con las mismas URLs que las utilizadas en las pruebas unitarias de la clase Analyzer, pero jugando con el valor booleano del XSS que se le pasa dentro de Validate al Analyzer.

Clase: URLValidator	
Método: void Validate(HttpContext context)	
Datos introducidos	**Resultado Esperado**
“http://www.paginaweb.com/”, true, false	Petición procesada
“https://www.paginaweb.com/”, false, false	Petición procesada
“http://www.paginaweb.com/elemento1.aspx” , true, false	Petición procesada
“http://www.paginaweb.com/elemento1/elemento2.aspx” , true, false	Petición procesada
“htTP://wWw.paginaWEB.cOm/ElEmentO1/ElEmEnto2.aSpx” , true, false	Petición procesada
“http://pagina4web”, true, false	Petición procesada
“http://798423.com”, false, false	Petición procesada

Clase: URLValidator	
Método: void Validate(HttpContext context)	
Datos introducidos	**Resultado Esperado**
“http://www.pagina.com/operacion?v=PfapQg9iv”, true, false	Petición procesada
“http://localhost”, true, false	Petición procesada
“http://localhost:80”, true, false	Petición procesada
“http://localhost:80/elemento.aspx”, true, false	Petición procesada
“http://www.paginaweb.com<script>alert(1)</script>”, true, false	Se genera excepción
“http://www.paginaweb.com<sCriPt>alErt(1)</scCRIpt>”, false, false	Petición procesada
“http://www.paginaweb.com\"><script>alert(document.cookie)</script>”, false, false	Petición procesada
“http://www.paginaweb.com/javascript:alert(1)” , true, false	Se genera excepción
“http://www.paginaweb.com/';alert(1)” , true, false	Se genera excepción
“http://www.paginaweb.com/<elementoRaro.com”, true, false	Se genera excepción

6.6.3 Funcionalidad relativa a los elementos de formulario

Toda la funcionalidad relativa a los datos enviados por el usuario de la aplicación a través de formularios se implementa en la clase FormValidator. La única diferencia con respecto a la clase anterior, aparte del elemento web al que afectan, es que se proporciona también protección frente a ataques de inyección SQL.

6.6.3.1 Pruebas Unitarias

Clase: FormValidator	
Método: void Validate(HttpContext context)	
Datos introducidos	**Resultado Esperado**
“contraseña”, false, false	Petición procesada
“_*89contraseña”, false, false	Petición procesada
“’OR 1=1”, false, false	Petición procesada
“<script>alert(1)</script>”, false, false	Petición procesada
“contraseña”, true, false	Petición procesada
“_*89contraseña”, true, false	Petición procesada
“’OR 1=1”, true, false	Petición procesada
“<script>alert(1)</script>”, true, false	Se genera excepción
“contraseña”, false, true	Petición procesada
“_*89contraseña”, false, true	Petición procesada
“’OR 1=1”, false, true	Se genera excepción
“<script>alert(1)</script>”, false, true	Petición procesada
“contraseña”, true, true	Petición procesada
“_*89contraseña”, true, true	Petición procesada
“’OR 1=1”, true, true	Se genera excepción
“<script>alert(1)</script>”, true, true	Se genera excepción

6.6.3.2 Pruebas de Integración

A continuación se especifican las pruebas de integración a realizar una vez finalizada la funcionalidad relativa a los elementos de formulario y a la URL.

Número	Prueba	Resultado Esperado
1	Procesar petición con URL válida.	Se procesa correctamente la petición accediendo al recurso web solicitado.
2	Procesar petición con URL no válida.	Se genera un error HTTP con código de estado 403.
3	Procesar petición con datos enviados en formulario válidos.	Se procesa correctamente la petición accediendo al recurso web solicitado.
4	Procesar petición con datos enviados en formulario no válidos.	Se genera un error HTTP con código de estado 403.

6.6.4 Funcionalidad relativa a las Cookies

El validador de las cookies posee la misma funcionalidad que el de los elementos de formulario, con el añadido de que se elimina la persistencia de las cookies permanentes, conviertiéndolas en cookies de sesión.

6.6.4.1 Pruebas Unitarias

Clase: CookieValidator	
Método: void Validate(HttpContext context)	
Datos introducidos	**Resultado Esperado**
“sessionId=5fas41hgdf35h4136”, false, false	Petición procesada
“discount=2.0”, false, false	Petición procesada
“sessionId=<script>”, false, false	Petición procesada
“sessionId=’or%201=1”, false, false	Petición procesada
“sessionId=5fas41hgdf35h4136”, true, false	Petición procesada
“discount=2.0”, true, false	Petición procesada
“sessionId=<script>”, true, false	Petición procesada
“sessionId=’or%201=1”, true, false	Se genera excepción
“sessionId=5fas41hgdf35h4136”, false, true	Petición procesada
“discount=2.0”, false, true	Petición procesada
“sessionId=<script>”, false, true	Se genera excepción
“sessionId=’or%201=1”, false, true	Petición procesada
“sessionId=5fas41hgdf35h4136”, true, true	Petición procesada
“discount=2.0”, true, true	Petición procesada
“sessionId=<script>”, true, true	Se genera excepción
“sessionId=’or%201=1”, true, true	Se genera excepción

Clase: CookieValidator	
Método: void RemoveCookiesPersistence(HttpContext context)	
Datos introducidos	**Resultado Esperado**
Null	-
HttpCookieCollection (Expires = DateTime.MAXVALUE)	HttpCookieCollection (Expires = DateTime.MINVALUE)
HttpCookieCollection (Expires = DateTime.MINVALUE)	HttpCookieCollection (Expires = DateTime.MINVALUE)
HttpCookieCollection (Expires = 01/01/2015 00:00:00)	HttpCookieCollection (Expires = DateTime.MINVALUE)
HttpCookieCollection (Expires =null)	HttpCookieCollection (Expires = DateTime.MINVALUE)

6.6.4.2 Pruebas de Integración

Número	Prueba	Resultado Esperado
1	Procesar petición con URL válida.	Se procesa correctamente la petición accediendo al recurso web solicitado.
2	Procesar petición con URL no válida.	Se genera un error HTTP con código de estado 403.
3	Procesar petición con datos enviados en formulario válidos.	Se procesa correctamente la petición accediendo al recurso web solicitado.
4	Procesar petición con datos enviados en formulario no válidos.	Se genera un error HTTP con código de estado 403.
5	Procesar petición sin cookies.	Se procesa correctamente la petición accediendo al recurso web solicitado.
6	Procesar petición con cookies válidas de sesión.	Se procesa correctamente la petición accediendo al recurso web solicitado.
7	Procesar petición con cookies no válidas de sesión.	Se genera un error HTTP con código de estado 403.
8	Procesar petición con cookies válidas permanentes.	Se procesa correctamente la petición accediendo al recurso web solicitado y las cookies dejan de ser válidas una vez se cierra el navegador.
9	Procesar petición con cookies no válidas permanentes.	Se genera un error HTTP con código de estado 403.

6.6.5 Funcionalidad relativa a las IPs

En esta sección se definen las pruebas a llevar a cabo en cuanto a la validación de las IPs desde las que se realizan las peticiones HTTP. Esta validación incluye:

- Chequeo de la dirección IP contra la DNSBL de www.blocklist.de
- Envío de email en caso de repetidos intentos desde una misma IP.

6.6.5.1 Pruebas Unitarias

Clase: IPValidator	
Método: void Validate(HttpContext context)	
Datos introducidos	**Resultado Esperado**
127.0.01 (localhost)	Petición procesada
156.35.98.20	Petición procesada
:::1 (localhost)	Petición procesada
37.187.89.77 (registrada en la DNSBL)	Se genera excepción
116.10.191.199 (registrada en la DNSBL)	Se genera excepción
Método SendMail(IPQueryResult result)	
Datos introducidos	**Resultado Esperado**
Attacks=0	Petición procesada
Attacks=214	Email recibido
Attacks=2279	Email recibido

6.6.5.2 Pruebas de Integración

Número	Prueba	Resultado Esperado
1	Procesar petición con URL válida.	Se procesa correctamente la petición accediendo al recurso web solicitado.
2	Procesar petición con URL no válida.	Se genera un error HTTP con código de estado 403.
3	Procesar petición con datos enviados en formulario válidos.	Se procesa correctamente la petición accediendo al recurso web solicitado.
4	Procesar petición con datos enviados en formulario no válidos.	Se genera un error HTTP con código de estado 403.
5	Procesar petición sin cookies.	Se procesa correctamente la petición accediendo al recurso web solicitado.
6	Procesar petición con cookies válidas de sesión.	Se procesa correctamente la petición accediendo al recurso web solicitado.
7	Procesar petición con cookies no válidas de sesión.	Se genera un error HTTP con código de estado 403.

Número	Prueba	Resultado Esperado
8	Procesar petición con cookies válidas permanentes.	Se procesa correctamente la petición accediendo al recurso web solicitado y las cookies dejan de ser válidas una vez se cierra el navegador.
9	Procesar petición con cookies no válidas permanentes.	Se genera un error HTTP con código de estado 403.
10	Procesar petición enviada desde IP no registrada en la DNSBL	Se procesa correctamente la petición accediendo al recurso web solicitado.
11	Procesar petición enviada desde registrada en la DNSBL sin superar el máximo de intentos establecido.	Se genera un error HTTP con código de estado 403.
12	Procesar petición enviada desde registrada en la DNSBL superando el máximo de intentos establecido.	Se genera un error HTTP con código de estado 403 y se recibe un email en la dirección especificada

6.6.6 Interfaz

La interfaz será la última parte del proyecto que se implementará, y al ser una interfaz, las pruebas unitarias se limitan a operar con la misma. Por estas dos razones, las pruebas que se realizarán una vez desarrollada la interfaz, serán las de sistema, pues ya estará todo implementado.

6.6.6.1 Pruebas de Sistema

Número	Prueba	Resultado Esperado
1	Con la validación de URL desactivada, enviar petición con una URL no válida.	Se procesa correctamente la petición accediendo al recurso web solicitado.
2	Con la validación de URL activada, enviar petición con una URL válida.	Se procesa correctamente la petición accediendo al recurso web solicitado.
3	Con la validación de URL activada, enviar petición con una URL no válida.	Se genera un error HTTP con código de estado 403.
4	Con una regla de validación personalizada, enviar petición con una URL que coincida con la nueva regla.	Se procesa correctamente la petición accediendo al recurso web solicitado.

Número	Prueba	Resultado Esperado
5	Con una regla de validación personalizada, enviar petición con una URL que no coincida con la nueva regla.	Se genera un error HTTP con código de estado 403.
6	Con la detección de XSS activada, enviar petición con una URL con código javascript inyectado.	Se genera un error HTTP con código de estado 403.
7	Con la detección de XSS activada, enviar petición con una URL sin código javascript inyectado.	Se procesa correctamente la petición accediendo al recurso web solicitado.
8	Con la validación de elementos de formulario desactivada, enviar petición con elementos de formulario no válidos.	Se procesa correctamente la petición accediendo al recurso web solicitado.
9	Con la validación de elementos de formulario activada, enviar petición con elementos de formulario válidos.	Se procesa correctamente la petición accediendo al recurso web solicitado.
10	Con la validación de elementos de formulario activada, petición con elementos de formulario no válidos.	Se genera un error HTTP con código de estado 403.
11	Con una regla de validación personalizada, enviar petición con elementos de formulario que coincidan con la nueva regla.	Se procesa correctamente la petición accediendo al recurso web solicitado.
12	Con una regla de validación personalizada, enviar petición con elementos de formulario que no coincidan con la nueva regla.	Se genera un error HTTP con código de estado 403.
13	Con la detección de XSS activada, enviar petición con elementos de formulario con código javascript inyectado.	Se genera un error HTTP con código de estado 403.
14	Con la detección de XSS activada, enviar petición con elementos de formulario sin código javascript inyectado.	Se procesa correctamente la petición accediendo al recurso web solicitado.

Número	Prueba	Resultado Esperado
15	Con la detección de SQLI activada, enviar petición con elementos de formulario con código SQL inyectado	Se genera un error HTTP con código de estado 403.
16	Con la detección de SQLI activada, enviar petición con elementos de formulario sin código SQL inyectado	Se procesa correctamente la petición accediendo al recurso web solicitado.
17	Con la validación de cookies desactivada, enviar petición con cookies no válidas.	Se procesa correctamente la petición accediendo al recurso web solicitado.
18	Con la validación de cookies activada, enviar petición con cookies válidas.	Se procesa correctamente la petición accediendo al recurso web solicitado.
19	Con la validación de cookies activada, enviar petición con cookies no válidas.	Se genera un error HTTP con código de estado 403.
20	Con una regla de validación personalizada, enviar petición con cookies que coincidan con la nueva regla.	Se procesa correctamente la petición accediendo al recurso web solicitado.
21	Con una regla de validación personalizada, enviar petición con cookies que no coincidan con la nueva regla.	Se genera un error HTTP con código de estado 403.
22	Con la detección de XSS activada, enviar petición con cookies con código javascript inyectado.	Se genera un error HTTP con código de estado 403.
23	Con la detección de XSS activada, enviar petición con cookies sin código javascript inyectado.	Se procesa correctamente la petición accediendo al recurso web solicitado.
24	Con la detección de SQLI activada, enviar petición con cookies con código SQL inyectado	Se genera un error HTTP con código de estado 403.
25	Con la detección de SQLI activada, enviar petición con cookies sin código SQL inyectado	Se procesa correctamente la petición accediendo al recurso web solicitado.

Número	Prueba	Resultado Esperado
26	Con la eliminación de persistencia de las cookies desactivada, enviar petición con cookies persistentes	Se procesa correctamente la petición accediendo al recurso web solicitado y las cookies permanecen inalteradas.
27	Con la eliminación de persistencia de las cookies activada, enviar petición con cookies de sesión	Se procesa correctamente la petición accediendo al recurso web solicitado y las cookies permanecen inalteradas.
28	Con la eliminación de persistencia de las cookies activada, enviar petición con cookies persistentes	Se procesa correctamente la petición accediendo al recurso web solicitado y la caducidad de las cookies desaparece, convirtiéndose así en cookies de sesión.
29	Con la comprobación de IP desactivada, enviar petición con la IP modificada para que aparezca registrada en la DNSBL	Se procesa correctamente la petición accediendo al recurso web solicitado.
30	Con la comprobación de IP activada, enviar petición con IP válida	Se procesa correctamente la petición accediendo al recurso web solicitado.
31	Con la comprobación de IP activada, enviar petición con la IP modificada para que aparezca registrada en la DNSBL	Se genera un error HTTP con código de estado 403.
32	Con la opción de mandar email desactivada, enviar petición con la IP modificada para que aparezca en la DNSBL superando el número máximo de intentos establecidos	Se genera un error HTTP con código de estado 403 por cada petición realizada.
33	Con la opción de mandar email desactivada, enviar petición con la IP modificada para que aparezca en la DNSBL sin superar el número máximo de intentos establecidos	Se genera un error HTTP con código de estado 403 por cada petición realizada.
34	Con la opción de mandar email activada, enviar petición con la IP modificada para que aparezca en la DNSBL superando el número máximo de intentos	Se genera un error HTTP con código de estado 403 por cada petición realizada y se recibe un email en la dirección de correo especificada.

Capítulo 7. Implementación del Sistema

7.1 Estándares y Normas Seguidos

Aunque en la especificación del lenguaje C# no se define ningún estándar de codificación, durante la implementación del proyecto se ha utilizado la convención descrita en http://msdn.microsoft.com/es-es/library/ff926074.aspx con el fin de proporcionar un código legible y mantenible a futuros programadores.

Con el mismo propósito, se ha decidido realizar la programación en inglés (clases, miembros de clases, métodos y comentarios)

7.2 Lenguajes de Programación

El lenguaje de programación empleado para el desarrollo del proyecto es C# de .NET, versión 4.0. El motivo de no haber usado la última versión, la 4.5, es que la versión del servidor IIS para la que se desarrolló el proyecto es la 7, la cual sólo admite funcionar bajo las versiones de .NET desde la 2.0 hasta la 4.0.

De las librerías utilizadas, cabe destacar las siguientes:

- **System.Web:** Esta librería, incluida en .NET, fue utilizada tanto para extraer los parámetros de las peticiones HTTP necesarios para analizarlas, como para implementar la interfaz del módulo. Al ser requisito que se integre con el programa de administración de IIS, fue necesario hacer que algunas de las clases de la interfaz heredasen de los componentes de .NET que implementan pantallas y secciones de configuración en dicho programa.
- **System.Text.RegularExpressions:** Esta librería, también incluida en el framework de .NET, fue necesaria para implementar expresiones regulares. Se utiliza en la clase Analyzer del módulo y en la interfaz del mismo para realizar la validación de las reglas especificadas por el usuario.
- **Newtonsoft.Json:** Esta librería se usa para interactuar con documentos JSON. El motivo de incluirla es utilizar la API de la DNS BL. Esta API pertenece al servicio web RESTful de http://www.blocklist.de/en/api.html , y su funcionamiento básico consiste en pasarle una petición GET con la IP que se quiere comprobar y el formato en el que se quieren obtener los resultados, JSON en este caso.

7.3 Herramientas y Programas Usados para el Desarrollo

7.3.1 Visual Studio 2013

La herramienta de desarrollo utilizada durante el desarrollo del proyecto ha sido Visual Studio, versión 2013. Con ella se ha implementado toda la funcionalidad del módulo, así como su interfaz, y se han realizado las pruebas unitarias.

7.3.2 IIS 7.5

El servidor para el que se ha desarrollado el módulo, IIS, también ha servido de herramienta para probarlo. Se ha desarrollado una aplicación web de prueba que se ha integrado en el servidor y a la que se le ha instalado el módulo objeto del proyecto.

7.3.3 GitHub

Para la gestión del código, se ha utilizado un repositorio git público alojado en el servicio GitHub. Las ramas creadas corresponden a cada una de las funcionalidades implementadas; validación de elementos susceptibles de contener inyección de código, eliminación de persistencia en cookies, comprobación de IP y desarrollo de interfaz. Asimismo, se ha incluido una rama extra para la aplicación web sobre la que se probaría el módulo.

El enlace al repositorio git es el siguiente:

https://github.com/miguel-otero/IIS7_SecurityModule

7.4 Creación del Sistema

7.4.1 Problemas Encontrados

- **Versiones de .NET:** Uno de los problemas encontrados fue a la hora de probar la interfaz. Ésta, una vez compilada, se ha de añadir a la caché global de ensamblados de Windows (GAC) para poder ser detectada por el servidor IIS. Para ello se utiliza la herramienta gacutil desde el intérprete de comandos, la cual añade el ensamblado seleccionado a la GAC. Sin embargo, aunque el mensaje proporcionado por la herramienta indicaba que el ensamblado se había añadido correctamente, desde IIS no era capaz de cargarlo. El problema era que la herramienta de gacutil utilizada añadía el ensamblado a una GAC correspondiente a .NET 4.5, a la cual no era capaz de acceder IIS. Una vez se cambió la versión de la herramienta utilizada, IIS detectó el ensamblado y mostró la interfaz del módulo.
- **Uso de IIS 7:** Para el desarrollo del proyecto fue necesario documentarse sobre el funcionamiento de IIS 7, concretamente de todo el sistema de módulos, su forma de procesar peticiones HTTP y sobre cómo añadir nuevos módulos al pipeline. Esta última cuestión dio problemas, pues en un principio no se conseguía hacer funcionar el módulo para una aplicación web. Finalmente se descubrió el formato con el que había que especificar a IIS que se quería activar un módulo para una aplicación.
- **Mecánica de las cookies HTTP:** No se conocía detalladamente el funcionamiento de las cookies HTTP intercambiadas entre cliente y servidor, por lo que fue necesario documentarse sobre ellas para saber qué se podía y qué no se podía hacer con ellas. Al principio esto dio problemas a la hora de trabajar con la caducidad de las mismas, pues en .NET, para indicar que una cookie es de sesión, hay que asignarle a la caducidad el mínimo valor del tipo DateTime.

7.4.2 Descripción Detallada de las Clases

7.4.2.1 Módulo

SecurityModule

Nombre	**Tipo**	**Descripción**	**Hereda de…**
SecurityModule	Derivada	Recoge la petición HTTP para enviársela a cada uno de los validadores	IHttpModule
Responsabilidades			
Número	**Descripción**		
1	Recoger la petición HTTP enviada por el pipeline		
2	Llamar a cada validador para que procese la petición		
Métodos			
Acceso | Modo	**Tipo de Retorno**	**Nombre**	**Parámetros y tipos**
Público | Normal	Void	Init	-HttpApplication app
Público | Normal	Void	Dispose	-
Público | Normal	Void	OnPreRequestHandlerExecute	-Object source -EventArgs e
Privado | Normal	Void	UpdateSettings	-
Atributos			
Acceso	**Modo**	**Tipo o Clase**	**Nombre**
Privado	Normal	Validator	urlValidator
Privado	Normal	Validator	formValidator
Privado	Normal	Validator	cookieValidator
Privado	Normal	Validator	ipValidator
Observaciones			
-			

Validator

Nombre	**Tipo**	**Descripción**	**Hereda de…**
Validator	Abstracta	Implementa la interfaz que utilizarán los validadores individuales del módulo	-
Responsabilidades			
Número	**Descripción**		
1	Proporcionar el método Validate que redefinirán cada uno de los validadores que hereden de esta clase		
Métodos			
Acceso | Modo	**Tipo de Retorno**	**Nombre**	**Parámetros y tipos**
Público | Virtual	Void	Validate	-HttpContext context
Observaciones			
-			

URLValidator

Nombre	**Tipo**	**Descripción**	**Hereda de…**
URLValidator	Derivada	Recoge las opciones especificadas desde la interfaz para la validación de URLs, y las valida llamando a la clase Analyzer	Validator
Responsabilidades			
Número	**Descripción**		
1	Recoger las opciones del módulo con respecto al análisis de las URL solicitadas en la petición HTTP.		
2	Llamar a la clase Analyzer para que valide las URL en función de las opciones especificadas.		
Métodos			
Acceso | Modo	**Tipo de Retorno**	**Nombre**	**Parámetros y tipos**
Privado | Normal	Void	RefreshSettings	-
Público | Redefinido	Void	Validate	-HttpContext context
Atributos			
Acceso	**Modo**	**Tipo o Clase**	**Nombre**
Privado	Estático y final	String	defaultUrlRule
Privado	Normal	Analyzer	analyzer
Observaciones			
-			

FormValidator

Nombre	**Tipo**	**Descripción**	**Hereda de…**
FormValidator	Derivada	Recoge las opciones especificadas desde la interfaz para la validación de elementos de formulario, y las valida llamando a la clase Analyzer	Validator
Responsabilidades			
Número	**Descripción**		
1	Recoger las opciones del módulo con respecto al análisis de los elementos de formulario enviados en la petición HTTP.		
2	Llamar a la clase Analyzer para que valide los elementos de formulario en función de las opciones especificadas.		
Métodos			
Acceso | Modo	**Tipo de Retorno**	**Nombre**	**Parámetros y tipos**
Privado | Normal	Void	RefreshSettings	-
Público | Redefinido	Void	Validate	-HttpContext context
Atributos			
Acceso	**Modo**	**Tipo o Clase**	**Nombre**
Privado	Estático y final	String	defaultFormRule
Privado	Normal	Analyzer	analyzer
Observaciones			
-			

CookieValidator

Nombre	**Tipo**	**Descripción**	**Hereda de…**
CookieValidator	Derivada	Recoge las opciones especificadas desde la interfaz para la validación de cookies y eliminación de persistencia de las mismas, y las valida llamando a la clase Analyzer	Validator
Responsabilidades			
Número	**Descripción**		
1	Recoger las opciones del módulo con respecto al procesamiento de cookies que acompañan a la petición HTTP.		
2	Llamar a la clase Analyzer para que valide las cookies en función de las opciones especificadas.		
3	Eliminar la persistencia de las cookies que acompañan a la petición HTTP si así se especifica en la interfaz		
Métodos			
Acceso | Modo	**Tipo de Retorno**	**Nombre**	**Parámetros y tipos**
Privado | Normal	Void	RefreshSettings	-
Público | Redefinido	Void	Validate	-HttpContext context
Privado | Normal	Void	RemoveCookiesPersistence	-HttpContext context
Atributos			
Acceso	**Modo**	**Tipo o Clase**	**Nombre**
Privado	Estático y final	String	defaultCookiesRule
Privado	Normal	Analyzer	analyzer
Observaciones			
-			

IPValidator

Nombre	**Tipo**	**Descripción**	**Hereda de…**
IPValidator	Derivada	Recoge las opciones especificadas desde la interfaz para la comprobación de IPs, y las valida llamando a la API de la DNS BL	Validator
Responsabilidades			
Número	**Descripción**		
1	Recoger las opciones del módulo con respecto a la comprobación de IPs desde las que se realizan las peticiones HTTP.		
2	Llamar a la API de la DNS BL con la IP que se quiere consultar y comprobar el número de ataques registrados.		
3	Enviar un email al usuario si así está especificado desde la interfaz.		
Métodos			
Acceso | Modo	**Tipo de Retorno**	**Nombre**	**Parámetros y tipos**
Privado | Normal	Void	RefreshSettings	-
Público | Redefinido	Void	Validate	-HttpContext context
Privado | Normal	Void	SendEmail	-IPQueryResult result
Privado | Normal	Void	CheckIpAgainstApi	-String ip
Atributos			
Acceso	**Modo**	**Tipo o Clase**	**Nombre**
Privado	Normal	HttpClient	client
Privado	Normal	List<String>	ipWhitelist
Privado	Normal	Dictionary<string, int>	triesPerBlacklistedIp
Observaciones			
-			

IPQueryResult

Nombre	Tipo	Descripción	Hereda de…
IPQueryResult	-	Modela el resultado Json obtenido de la API de la DNS BL a raíz de una consulta realizada a la misma	-
Responsabilidades			
Número	**Descripción**		
1	Modelar el documento Json devuelto por la API de la DNS BL después de realizar una consulta sobre una IP		
Atributos			
Acceso	**Modo**	**Tipo o Clase**	**Nombre**
Privado	Normal	String	ip
Privado	Normal	Int	attacks
Observaciones			
-			

Analyzer

Nombre	Tipo	Descripción	Hereda de…
Analyzer	-	Valida, utilizando expresiones regulares, las cadenas de texto presentes en los diferentes elementos web a partir de las opciones especificadas por los validadores	-
Responsabilidades			
Número	**Descripción**		
1	Validar el formato de un elemento web		
2	Detectar ataques de inyección de código si así está especificado		
Métodos			
Acceso | Modo	**Tipo de Retorno**	**Nombre**	**Parámetros y tipos**
Público | Normal	Boolean	Analyze	-boolean checkXss -boolean checkSqli -string str
Público | Normal	Void	AddCodeInjectionRule	-string str
Atributos			
Acceso	**Modo**	**Tipo o Clase**	**Nombre**
Privado	Normal	Regex	validationRegex
Privado	Normal	Regex	xssRegex
Privado	Normal	Regex	sqliRegex
Observaciones			
-			

7.4.2.2 Interfaz

UIModuleProvider

Nombre	**Tipo**	**Descripción**	**Hereda de…**
UIModuleProvider	Derivada	Esta clase sirve de conexión entre el ensamblado y la definición del módulo dentro de IIS	ModuleProvider
Responsabilidades			
Número	**Descripción**		
1	Publicar la interfaz como parte de IIS		
Métodos			
Acceso | Modo	**Tipo de Retorno**	**Nombre**	**Parámetros y tipos**
Público | Redefinido	Bool	SupportsScope	ManagementScope scope
Público | Redefinido	ModuleDefinition	GetModuleDefinition	IManagementContext context
Observaciones			
-			

UIModule

Nombre	**Tipo**	**Descripción**	**Hereda de…**
UIModule	Derivada	Esta clase es la responsable de que la interfaz del módulo sea accesible desde la vista “Características” de una aplicación web desde el programa de administración de IIS	Module
Responsabilidades			
Número	**Descripción**		
1	Proporcionar acceso a la interfaz desde el programa de administración de IIS		
Métodos			
Acceso | Modo	**Tipo de Retorno**	**Nombre**	**Parámetros y tipos**
Protegido | Redefinido	Void	Initialize	-IServiceProvider serviceProvider -ModuleInfo moduleInfo
Protegido | Redefinido	Void	IsPageEnabled	-ModulePageInfo pageInfo
Observaciones			
-			

AdminPage

Nombre	**Tipo**	**Descripción**	**Hereda de…**
AdminPage	Derivada	Implementa el contenedor del programa de administración de IIS que contendrá la interfaz del módulo	ModulePage
Responsabilidades			
Número	**Descripción**		
1	Servir de conexión entre el programa de administración de IIS y la interfaz del módulo		
Métodos			
Acceso | Modo	**Tipo de Retorno**	**Nombre**	**Parámetros y tipos**
Público | Redefinido	Void	OnActivated	-boolean initialActivation
Atributos			
Acceso	**Modo**	**Tipo o Clase**	**Nombre**
Privado	Normal	ServerManager	serverMgr
Privado	Normal	ModuleConfiguration	configuration
Observaciones			
-			

ModuleConfiguration

Nombre	**Tipo**	**Descripción**	**Hereda de…**
ModuleConfiguration	Derivada	Implementa el formulario de Windows que se incrustará en el programa de administración de IIS y formará la interfaz del módulo.	UserControl
Responsabilidades			
Número	**Descripción**		
1	Ofrecer el formulario de Windows que formará la interfaz del módulo		
2	Aplicar los cambios realizados desde la interfaz al fichero de configuración de la aplicación		
Métodos			
Acceso | Modo	**Tipo de Retorno**	**Nombre**	**Parámetros y tipos**
Público | Normal	Void	GetSettings	-HttpApplication app
Privado | Normal	Void	Initialize	-string host -string path
Atributos			
Acceso	**Modo**	**Tipo o Clase**	**Nombre**
Privado	Normal	String	Host
Privado	Normal	ServerManager	serverMgr
Privado	Normal	String	Path
Privado	Normal	URLSettings	urlSettings
Privado	Normal	FormSettings	formSettings
Privado	Normal	CookiesSettings	cookiesSettings
Privado	Normal	IPSettings	ipSettings
Observaciones			
-			

7.4.2.3 Configuración

URLSettings

Nombre	Tipo	Descripción	Hereda de…
URLSettings	-	Modela las opciones especificadas desde la interfaz para el procesamiento de las peticiones cara a la URL	-
Responsabilidades			
Número	**Descripción**		
1	Modelar las opciones especificadas en la pestaña URL de la interfaz para su utilización por parte del módulo.		
Atributos			
Acceso	**Modo**	**Tipo o Clase**	**Nombre**
Privado	Normal	Boolean	urlValidation
Privado	Normal	Boolean	urlCustomRule
Privado	Normal	Boolean	urlXssProtection
Privado	Normal	String	urlCustomRuleStr
Observaciones			
-			

FormSettings

Nombre	Tipo	Descripción	Hereda de…
FormSettings	-	Modela las opciones especificadas desde la interfaz para el procesamiento de las peticiones cara a los elementos de formulario	-
Responsabilidades			
Número	**Descripción**		
1	Modelar las opciones especificadas en la pestaña Formulario de la interfaz para su utilización por parte del módulo.		
Atributos			
Acceso	**Modo**	**Tipo o Clase**	**Nombre**
Privado	Normal	Boolean	formValidation
Privado	Normal	Boolean	formCustomRule
Privado	Normal	Boolean	formXssProtection
Privado	Normal	Boolean	formSqliProtection
Privado	Normal	String	formCustomRuleStr
Observaciones			
-			

CookiesSettings

Nombre	**Tipo**	**Descripción**	**Hereda de…**
CookiesSettings	-	Modela las opciones especificadas desde la interfaz para el procesamiento de las peticiones cara a las cookies HTTP	-
Responsabilidades			
Número	**Descripción**		
1	Modelar las opciones especificadas en la pestaña Cookies de la interfaz para su utilización por parte del módulo.		
Atributos			
Acceso	**Modo**	**Tipo o Clase**	**Nombre**
Privado	Normal	Boolean	cookieValidation
Privado	Normal	Boolean	cookieCustomRule
Privado	Normal	Boolean	cookieXssProtection
Privado	Normal	Boolean	cookieSqliProtection
Privado	Normal	Boolean	cookieRemovePersistence
Privado	Normal	String	cookieCustomRuleStr
Observaciones			
-			

IPSettings

Nombre	**Tipo**	**Descripción**	**Hereda de…**
IPSettings	-	Modela las opciones especificadas desde la interfaz para el procesamiento de las peticiones cara a las direcciones IP desde las que se realizan.	-
Responsabilidades			
Número	**Descripción**		
1	Modelar las opciones especificadas en la pestaña IP Check de la interfaz para su utilización por parte del módulo.		
Atributos			
Acceso	**Modo**	**Tipo o Clase**	**Nombre**
Privado	Normal	Boolean	ipValidation
Privado	Normal	Boolean	ipSendMail
Privado	Normal	Boolean	ipEmailAddress
Privado	Normal	Boolean	ipMaxTries
Observaciones			
-			

Capítulo 8. Desarrollo de las Pruebas

En este capítulo se muestran los resultados de las pruebas llevadas a cabo sobre el sistema. Como ya se dijo en el capítulo 6, se documentan tanto las pruebas superadas como las fallidas. En este último caso, al final de cada sección se incluye una tabla con las reparaciones llevadas a cabo y el resultado de las mismas.

8.1.1 Integración del módulo en IIS

Número	Prueba	Mecanismo de notificación de resultados	Resultado Esperado	Resultado Obtenido
1	Activar módulo	En las respuestas HTTP enviadas al cliente se imprimirá la cadena "Module enabled" al principio de la misma	Se muestra la cadena "Module enabled" al principio de la respuesta	Correcto
3	Desactivar módulo	La ausencia de resultados es el mecanismo de notificación	No se muestra la cadena "Module enabled" al principio de la respuesta	Correcto
7	Extraer URL de la petición	En las respuestas HTTP enviadas al cliente se imprimirá la URL solicitada al principio de la misma	Se muestra la URL al principio de la respuesta	Correcto
8	Extraer elementos de formulario de la petición	En las respuestas HTTP enviadas al cliente a raíz de una petición POST, se imprimirán los datos introducidos por el usuario mediante formulario	Se muestran los datos de formulario enviados por el usuario al principio de la respuesta.	Correcto

Número	Prueba	Mecanismo de notificación de resultados	Resultado Esperado	Resultado Obtenido
9	Extraer cookies de la petición	En las respuestas HTTP enviadas al cliente se imprimirán las cookies enviadas por el cliente.	Se muestran los valores de las cookies al principio de la respuesta	Correcto
10	Extraer IP origen de la petición	En las respuestas HTTP enviadas al cliente se imprimirá la IP desde la que se envió la petición.	Se muestra la dirección IP desde la que se realizó la petición al principio de la respuesta	Correcto
11	Interrumpir procesamiento de la petición	Se provocará temporalmente el lanzamiento de una excepción HTTP con código 403 en función de los elementos extraídos de la petición. En este caso, se lanzará si en alguno de los elementos de formulario se encuentra la cadena “throwException”	Se genera un error 403 y no se permite acceder al recurso web solicitado	Correcto

8.1.2 Funcionalidad relativa a la URL

8.1.2.1 Pruebas Unitarias

Clase: Analyzer		
Método: Boolean Analyze(String str, boolean applyXss, boolean applySqli)		
Datos introducidos	**Resultado Esperado**	**Resultado Obtenido**
“http://www.paginaweb.com/”, true, false	True	True
“https://www.paginaweb.com/”, true, false	True	True
“http://www.paginaweb.com/elemento1.aspx” , true, false	True	True
“http://www.paginaweb.com/elemento1/elemento2.aspx” , true, false	True	True
“htTP://wWw.paginaWEB.cOm/ElEmentO1/ElEmEnto2.aSpx” , true, false	True	True
“http://pagina4web”, true, false	True	True
“http://798423.com”, true, false	True	True
“http://www.pagina.com/operacion?v=PfapQg9iv”, true, false	True	True
“http://localhost”, true, false	True	True
“http://localhost:80”, true, false	True	True

Clase: Analyzer		
Método: Boolean Analyze(String str, boolean applyXss, boolean applySqli)		
Datos introducidos	**Datos introducidos**	**Datos introducidos**
“http://localhost:80/elemento.aspx”, true, false	True	True
“http://www.paginaweb.com<script>alert(1)</script>”, true, false	False	False
“http://www.paginaweb.com<sCriPt>alErt(1)</scCRIpt>”, true, false	False	False
“http://www.paginaweb.com\"><script>alert(document.cookie)</script>”, true, false	False	False
“http://www.paginaweb.com/javascript:alert(1)” , true, false	False	False
“http://www.paginaweb.com/';alert(1)” , true, false	False	False
“http://www.paginaweb.com/<elementoRaro.com”, true, false	False	False
Método Regex SetValidationRegex(String validationStr)		
Datos introducidos	**Resultado Esperado**	
\s*	\s* (Regex)	Correcto
abcd+	abcd+ (Regex)	Correcto
.+	.+ (Regex)	Correcto
localhost:?/[a-z0-9]+(.[a-z]+)*	localhost:?/[a-z0-9]+(.[a-z]+)* (Regex)	Correcto

Clase: URLValidator		
Método: void Validate(HttpContext context)		
Datos introducidos	**Resultado Esperado**	**Resultado Obtenido**
“http://www.paginaweb.com/”, true, false	Petición procesada	Petición procesada
“https://www.paginaweb.com/”, false, false	Petición procesada	Petición procesada
“http://www.paginaweb.com/elemento1.aspx” , true, false	Petición procesada	Petición procesada
“http://www.paginaweb.com/elemento1/elemento2.aspx” , true, false	Petición procesada	Petición procesada
“htTP://wWw.paginaWEB.cOm/ElEmentO1/ElEmEnto2.aSpx” , true, false	Petición procesada	Petición procesada
“http://pagina4web”, true, false	Petición procesada	Petición procesada
“http://798423.com”, false, false	Petición procesada	Petición procesada
“http://www.pagina.com/operacion?v=PfapQg9iv”, true, false	Petición procesada	Petición procesada
“http://localhost”, true, false	Petición procesada	Petición procesada

Clase: URLValidator		
Método: void Validate(HttpContext context)		
Datos introducidos	**Datos introducidos**	**Datos introducidos**
“http://localhost:80”, true, false	Petición procesada	Petición procesada
“http://localhost:80/elemento.aspx”, true, false	Petición procesada	Petición procesada
“http://www.paginaweb.com<script>alert(1)</script>”, true, false	Se genera excepción	Se genera excepción
“http://www.paginaweb.com<sCriPt>alErt(1)</scCRIpt>”, false, false	Petición procesada	Petición procesada
“http://www.paginaweb.com\"><script>alert(document.cookie)</script>”, false, false	Petición procesada	Petición procesada
“http://www.paginaweb.com/javascript:alert(1)”, true, false	Se genera excepción	Se genera excepción
“http://www.paginaweb.com/';alert(1)” , true, false	Se genera excepción	Se genera excepción
“http://www.paginaweb.com/<elementoRaro.com”, true, false	Se genera excepción	Se genera excepción

8.1.3 Funcionalidad relativa a los elementos de formulario

8.1.3.1 Pruebas Unitarias

Clase: FormValidator		
Método: void Validate(HttpContext context)		
Datos introducidos	**Resultado Esperado**	**Resultado Obtenido**
“contraseña”, false, false	Petición procesada	Petición procesada
“_*89contraseña”, false, false	Petición procesada	Petición procesada
“’OR 1=1”, false, false	Petición procesada	Petición procesada
“<script>alert(1)</script>”, false, false	Petición procesada	Petición procesada
“contraseña”, true, false	Petición procesada	Petición procesada
“_*89contraseña”, true, false	Petición procesada	Petición procesada
“’OR 1=1”, true, false	Petición procesada	Petición procesada
“<script>alert(1)</script>”, true, false	Se genera excepción	Se genera excepción
“contraseña”, false, true	Petición procesada	Petición procesada
“_*89contraseña”, false, true	Petición procesada	Petición procesada
“’OR 1=1”, false, true	Se genera excepción	Petición procesada
“<script>alert(1)</script>”, false, true	Petición procesada	Petición procesada
“contraseña”, true, true	Petición procesada	Petición procesada
“_*89contraseña”, true, true	Petición procesada	Petición procesada
“’OR 1=1”, true, true	Se genera excepción	Se genera excepción
“<script>alert(1)</script>”, true, true	Se genera excepción	Se genera excepción

Reparaciones

Clase: FormValidator		
Método: void Validate(HttpContext context)		
Datos introducidos	**Solución**	**Resultado Obtenido**
"'OR 1=1", false, true	Modificar expresión regular de detección de SQLI	Se genera excepción

8.1.3.2 Pruebas de Integración

Número	Prueba	Resultado Esperado	Resultado Obtenido
1	Procesar petición con URL válida.	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
2	Procesar petición con URL no válida.	Se genera un error HTTP con código de estado 403.	Correcto
3	Procesar petición con datos enviados en formulario válidos.	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
4	Procesar petición con datos enviados en formulario no válidos.	Se genera un error HTTP con código de estado 403.	Correcto

8.1.4 Funcionalidad relativa a las Cookies

8.1.4.1 Pruebas Unitarias

Clase: CookieValidator		
Método: void Validate(HttpContext context)		
Datos introducidos	**Resultado Esperado**	**Resultado Obtenido**
"sessionId=5fas41hgdf35h4136", false, false	Petición procesada	Petición procesada
"discount=2.0", false, false	Petición procesada	Petición procesada
"sessionId=<script>", false, false	Petición procesada	Petición procesada
"sessionId='or%201=1", false, false	Petición procesada	Petición procesada
"sessionId=5fas41hgdf35h4136", true, false	Petición procesada	Petición procesada
"discount=2.0", true, false	Petición procesada	Petición procesada
"sessionId=<script>", true, false	Petición procesada	Petición procesada
"sessionId='or%201=1", true, false	Se genera excepción	Se genera excepción
"sessionId=5fas41hgdf35h4136", false, true	Petición procesada	Petición procesada
"discount=2.0", false, true	Petición procesada	Petición procesada
"sessionId=<script>", false, true	Se genera excepción	Se genera excepción

Clase: CookieValidator		
Método: void Validate(HttpContext context)		
Datos introducidos	**Datos introducidos**	**Datos introducidos**
“sessionId=’or%201=1”, false, true	Petición procesada	Petición procesada
“sessionId=5fas41hgdf35h4136”, true, true	Petición procesada	Petición procesada
“discount=2.0”, true, true	Petición procesada	Petición procesada
“sessionId=<script>”, true, true	Se genera excepción	Se genera excepción
“sessionId=’or%201=1”, true, true	Se genera excepción	Se genera excepción

Método: void RemoveCookiesPersistence(HttpContext context)		
Datos introducidos	**Resultado Esperado**	**Resultado Obtenido**
Null	-	Correcto
HttpCookieCollection (Expires = DateTime.MAXVALUE)	HttpCookieCollection (Expires = DateTime.MINVALUE)	Correcto
HttpCookieCollection (Expires = DateTime.MINVALUE)	HttpCookieCollection (Expires = DateTime.MINVALUE)	Correcto
HttpCookieCollection (Expires = 01/01/2015 00:00:00)	HttpCookieCollection (Expires = DateTime.MINVALUE)	Correcto
HttpCookieCollection (Expires =null)	HttpCookieCollection (Expires = DateTime.MINVALUE)	Correcto

8.1.4.2 Pruebas de Integración

Número	**Prueba**	**Resultado Esperado**	**Resultado Obtenido**
1	Procesar petición con URL válida.	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
2	Procesar petición con URL no válida.	Se genera un error HTTP con código de estado 403.	Correcto
3	Procesar petición con datos enviados en formulario válidos.	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
4	Procesar petición con datos enviados en formulario no válidos.	Se genera un error HTTP con código de estado 403.	Correcto

Número	Prueba	Resultado Esperado	Resultado Obtenido
5	Procesar petición sin cookies.	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
6	Procesar petición con cookies válidas de sesión.	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
7	Procesar petición con cookies no válidas de sesión.	Se genera un error HTTP con código de estado 403.	Correcto
8	Procesar petición con cookies válidas permanentes.	Se procesa correctamente la petición accediendo al recurso web solicitado y las cookies dejan de ser válidas una vez se cierra el navegador.	Correcto
9	Procesar petición con cookies no válidas permanentes.	Se genera un error HTTP con código de estado 403.	Correcto

8.1.5 Funcionalidad relativa a las IPs

8.1.5.1 Pruebas Unitarias

Clase: IPValidator		
Método: void Validate(HttpContext context)		
Datos introducidos	**Resultado Esperado**	**Resultado Obtenido**
127.0.01 (localhost)	Petición procesada	Petición procesada
156.35.98.20	Petición procesada	Petición procesada
:::1 (localhost)	Petición procesada	Petición procesada
37.187.89.77 (registrada en la DNSBL)	Se genera excepción	Se genera excepción
116.10.191.199 (registrada en la DNSBL)	Se genera excepción	Se genera excepción
Método SendMail(IPQueryResult result)		
Datos introducidos	**Resultado Esperado**	**Resultado Obtenido**
Attacks=0	Petición procesada	Petición procesada
Attacks=214	Email recibido	Email recibido
Attacks=2279	Email recibido	Email recibido

8.1.5.2 Pruebas de Integración

Número	Prueba	Resultado Esperado	Resultado Obtenido
1	Procesar petición con URL válida.	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
2	Procesar petición con URL no válida.	Se genera un error HTTP con código de estado 403.	Correcto
3	Procesar petición con datos enviados en formulario válidos.	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
4	Procesar petición con datos enviados en formulario no válidos.	Se genera un error HTTP con código de estado 403.	Correcto
5	Procesar petición sin cookies.	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
6	Procesar petición con cookies válidas de sesión.	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
7	Procesar petición con cookies no válidas de sesión.	Se genera un error HTTP con código de estado 403.	Correcto

Número	Prueba	Resultado Esperado	Resultado Obtenido
8	Procesar petición con cookies válidas permanentes.	Se procesa correctamente la petición accediendo al recurso web solicitado y las cookies dejan de ser válidas una vez se cierra el navegador.	Correcto
9	Procesar petición con cookies no válidas permanentes.	Se genera un error HTTP con código de estado 403.	Correcto
10	Procesar petición enviada desde IP no registrada en la DNSBL	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
11	Procesar petición enviada desde registrada en la DNSBL sin superar el máximo de intentos establecido.	Se genera un error HTTP con código de estado 403.	Correcto
12	Procesar petición enviada desde registrada en la DNSBL superando el máximo de intentos establecido.	Se genera un error HTTP con código de estado 403 y se recibe un email en la dirección especificada	Correcto

8.1.6 Interfaz

8.1.6.1 Pruebas de Sistema

Número	Prueba	Resultado Esperado	Resultado Obtenido
1	Con la validación de URL desactivada, enviar petición con una URL no válida.	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
2	Con la validación de URL activada, enviar petición con una URL válida.	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
3	Con la validación de URL activada, enviar petición con una URL no válida.	Se genera un error HTTP con código de estado 403.	Correcto
4	Con una regla de validación personalizada, enviar petición con una URL que coincida con la nueva regla.	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
5	Con una regla de validación personalizada, enviar petición con una URL que no coincida con la nueva regla.	Se genera un error HTTP con código de estado 403.	Correcto
6	Con la detección de XSS activada, enviar petición con una URL con código javascript inyectado.	Se genera un error HTTP con código de estado 403.	Correcto
7	Con la detección de XSS activada, enviar petición con una URL sin código javascript inyectado.	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
8	Con la validación de elementos de formulario desactivada, enviar petición con elementos de formulario no válidos.	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
9	Con la validación de elementos de formulario activada, enviar petición con elementos de formulario válidos.	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
10	Con la validación de elementos de formulario activada, petición con elementos de formulario no válidos.	Se genera un error HTTP con código de estado 403.	Correcto

Número	Prueba	Resultado Esperado	Resultado Obtenido
11	Con una regla de validación personalizada, enviar petición con elementos de formulario que coincidan con la nueva regla.	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
12	Con una regla de validación personalizada, enviar petición con elementos de formulario que no coincidan con la nueva regla.	Se genera un error HTTP con código de estado 403.	Correcto
13	Con la detección de XSS activada, enviar petición con elementos de formulario con código javascript inyectado.	Se genera un error HTTP con código de estado 403.	Correcto
14	Con la detección de XSS activada, enviar petición con elementos de formulario sin código javascript inyectado.	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
15	Con la detección de SQLI activada, enviar petición con elementos de formulario con código SQL inyectado	Se genera un error HTTP con código de estado 403.	Correcto
16	Con la detección de SQLI activada, enviar petición con elementos de formulario sin código SQL inyectado	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
17	Con la validación de cookies desactivada, enviar petición con cookies no válidas.	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
18	Con la validación de cookies activada, enviar petición con cookies válidas.	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
19	Con la validación de cookies activada, enviar petición con cookies no válidas.	Se genera un error HTTP con código de estado 403.	Correcto
20	Con una regla de validación personalizada, enviar petición con cookies que coincidan con la nueva regla.	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto

Número	Prueba	Resultado Esperado	Resultado Obtenido
21	Con una regla de validación personalizada, enviar petición con cookies que no coincidan con la nueva regla.	Se genera un error HTTP con código de estado 403.	Correcto
22	Con la detección de XSS activada, enviar petición con cookies con código javascript inyectado.	Se genera un error HTTP con código de estado 403.	Correcto
23	Con la detección de XSS activada, enviar petición con cookies sin código javascript inyectado.	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
24	Con la detección de SQLI activada, enviar petición con cookies con código SQL inyectado	Se genera un error HTTP con código de estado 403.	Correcto
25	Con la detección de SQLI activada, enviar petición con cookies sin código SQL inyectado	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
26	Con la eliminación de persistencia de las cookies desactivada, enviar petición con cookies persistentes	Se procesa correctamente la petición accediendo al recurso web solicitado y las cookies permanecen inalteradas.	Correcto
27	Con la eliminación de persistencia de las cookies activada, enviar petición con cookies de sesión	Se procesa correctamente la petición accediendo al recurso web solicitado y las cookies permanecen inalteradas.	Correcto
28	Con la eliminación de persistencia de las cookies activada, enviar petición con cookies persistentes	Se procesa correctamente la petición accediendo al recurso web solicitado y la caducidad de las cookies desaparece, convirtiéndose así en cookies de sesión.	Correcto

Número	Prueba	Resultado Esperado	Resultado Obtenido
29	Con la comprobación de IP desactivada, enviar petición con la IP modificada para que aparezca registrada en la DNSBL	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
30	Con la comprobación de IP activada, enviar petición con IP válida	Se procesa correctamente la petición accediendo al recurso web solicitado.	Correcto
31	Con la comprobación de IP activada, enviar petición con la IP modificada para que aparezca registrada en la DNSBL	Se genera un error HTTP con código de estado 403.	Correcto
32	Con la opción de mandar email desactivada, enviar petición con la IP modificada para que aparezca en la DNSBL superando el número máximo de intentos establecidos	Se genera un error HTTP con código de estado 403 por cada petición realizada.	Correcto
33	Con la opción de mandar email desactivada, enviar petición con la IP modificada para que aparezca en la DNSBL sin superar el número máximo de intentos establecidos	Se genera un error HTTP con código de estado 403 por cada petición realizada.	Correcto
34	Con la opción de mandar email activada, enviar petición con la IP modificada para que aparezca en la DNSBL superando el número máximo de intentos	Se genera un error HTTP con código de estado 403 por cada petición realizada y se recibe un email en la dirección de correo especificada.	Correcto

Capítulo 9. Manuales del Sistema

9.1 Manual de Instalación

Dado que el módulo consiste en dos ensamblados, (la interfaz y el módulo en sí), no requiere de una instalación propiamente dicha, sino de una serie de operaciones sobre el servidor IIS para que reconozca estos ensamblados. Esa parte se explicará en el manual de ejecución. En esta sección se detallará la instalación del único software necesario: el servidor IIS.

Partiremos de una instalación de Windows Server 2008 o posterior (W7 incluido)

9.1.1 Windows Server

En el caso de una versión Server, abrimos el programa administrador del servidor y, desde la sección "Funciones" seleccionamos "Agregar funciones"

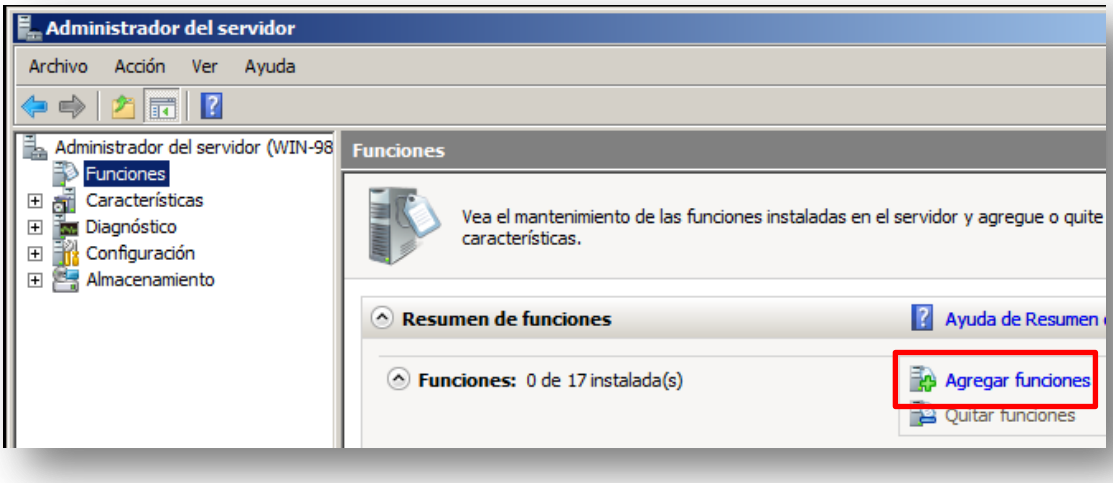


Ilustración 9-1: Administrador del servidor

En la ventana que nos aparezca hacemos click en “Siguiente” hasta que se muestre una lista de las posibles funciones a instalar en el servidor.

Ilustración 9-2: Lista de funciones del servidor

En la siguiente pantalla se mostrarán las diferentes características que se pueden activar para el servidor IIS. Las que están activadas por defecto son suficientes, pero podemos seleccionar más si prevemos que las vamos a utilizar. No obstante, no se recomienda instalar aquellos servicios para los que no tenemos pensar dar ninguna utilidad, ya que en caso de necesitarlos podemos activarlos posteriormente sin necesidad de repetir la instalación del servidor. Si no tenemos claro para qué sirve una determinada característica o servicio, al seleccionarlo nos aparece una descripción del mismo en la parte derecha del asistente de instalación.

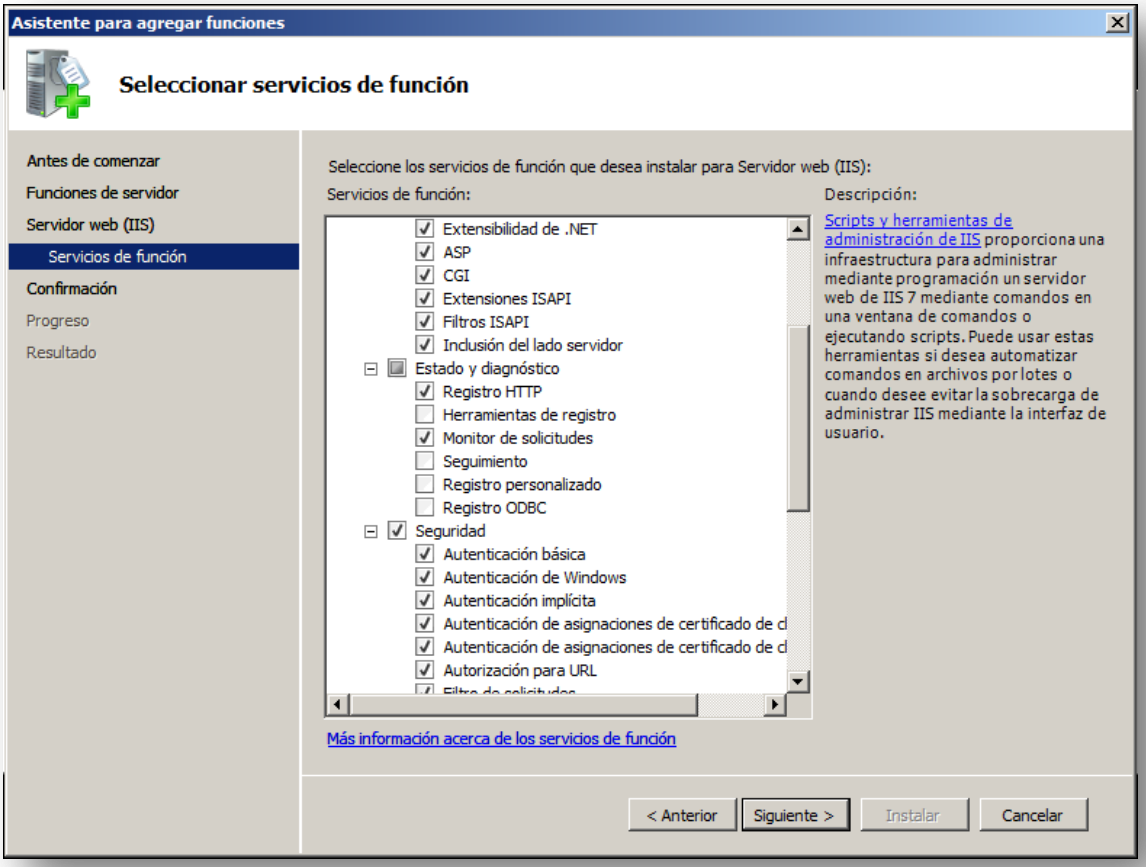


Ilustración 9-3: Servicios de IIS

Una vez hayamos seleccionado las características deseadas para el servidor, hacemos click en siguiente hasta que comience la instalación. Una vez termine, ya tenemos instalada una instancia del servidor IIS, y, si no tenemos ocupado el puerto 80, al acceder a localhost nos aparecerá la página por defecto de IIS.

Ilustración 9-4: Página por defecto de IIS

9.1.2 Windows 7

La instalación de IIS en Windows 7 difiere de la de Windows Server. Lo primero que haremos será ir al panel de control, y, desde ahí, acceder a la sección de programas y características.

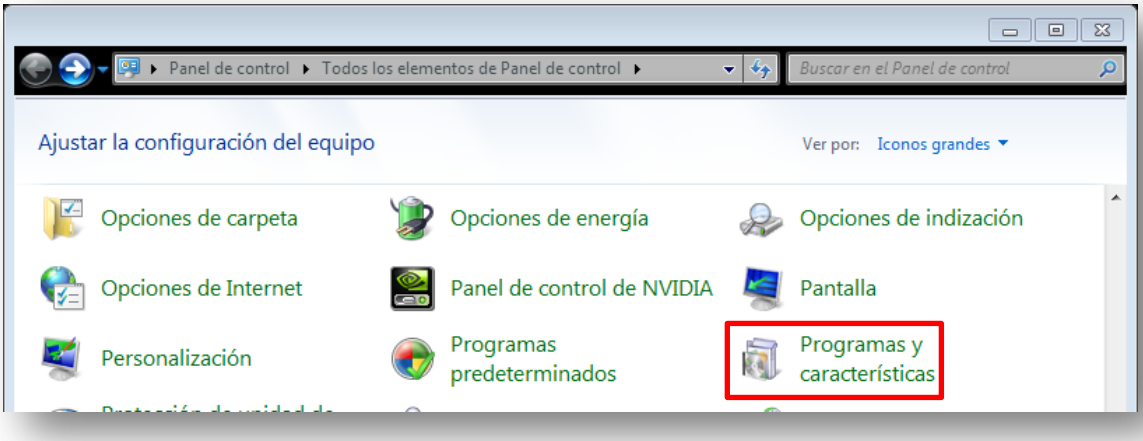


Ilustración 9-5: Panel de control

Una vez en esta sección, haremos click en "Activar o desactivar las características de Windows", para lo cual necesitaremos privilegios de administrador en la máquina.

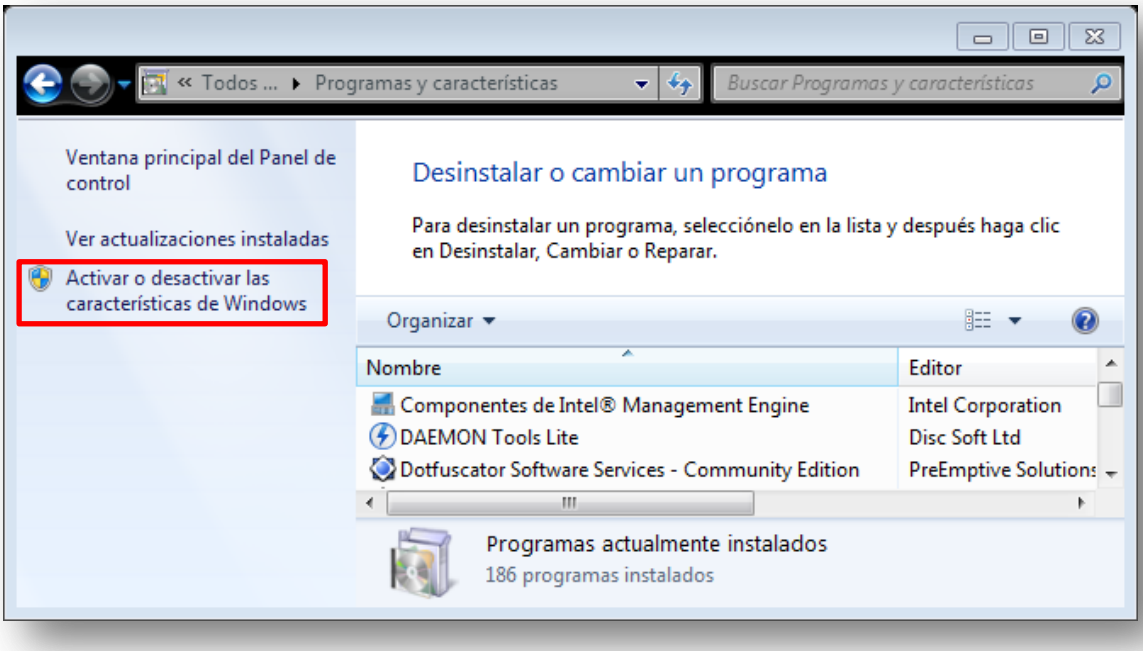


Ilustración 9-6: Programas y características

En la ventana que nos aparecerá, seleccionamos la casilla “Internet Information Services” junto con las características que deseemos activar para nuestro servidor.

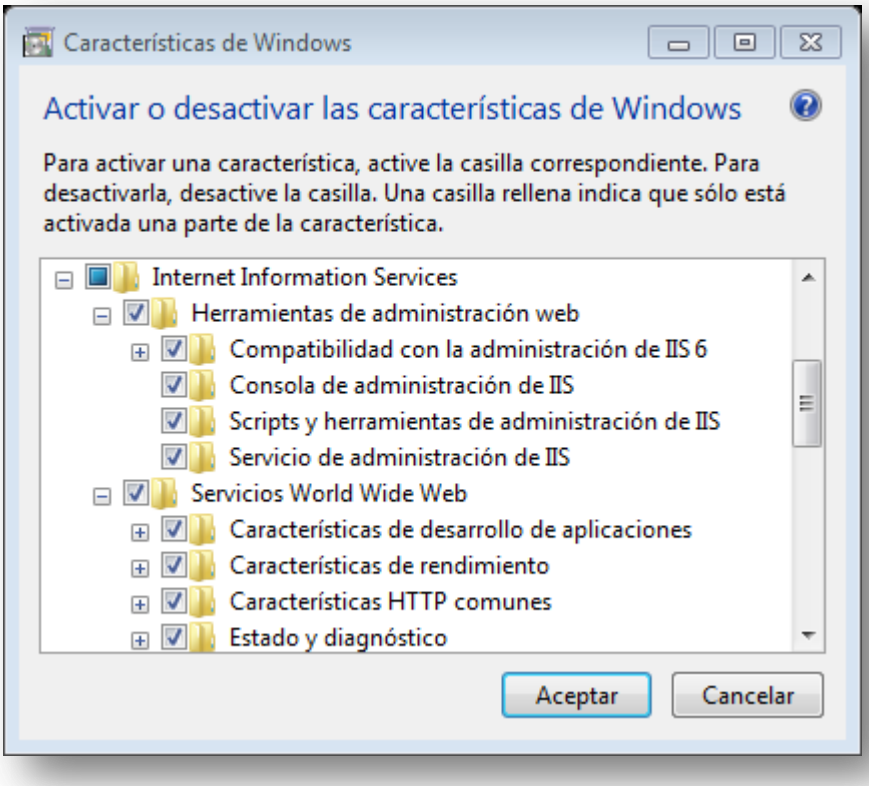


Ilustración 9-7: Características de Windows

Una vez hemos seleccionado las características, hacemos click en “Aceptar” y, en cuanto termine la instalación, ya tendremos una instancia del servidor IIS instalada y activa en nuestra máquina.

9.2 Manual de Ejecución

En este manual se explicará cómo integrar la interfaz del módulo en el programa de administración de IIS, y cómo activar el módulo para una determinada aplicación web. Además, se explicará la forma de publicar una aplicación en el servidor.

9.2.1 Activación de Interfaz

El primer paso consiste en copiar algunos de los archivos de configuración adjuntos a este documento en la carpeta %windir%/system32/inersrv/config/schema. Los archivos a copiar son:

- CookieSecModule.xml
- URLSecModule.xml
- IPSecModule.xml
- FormSecModule.xml

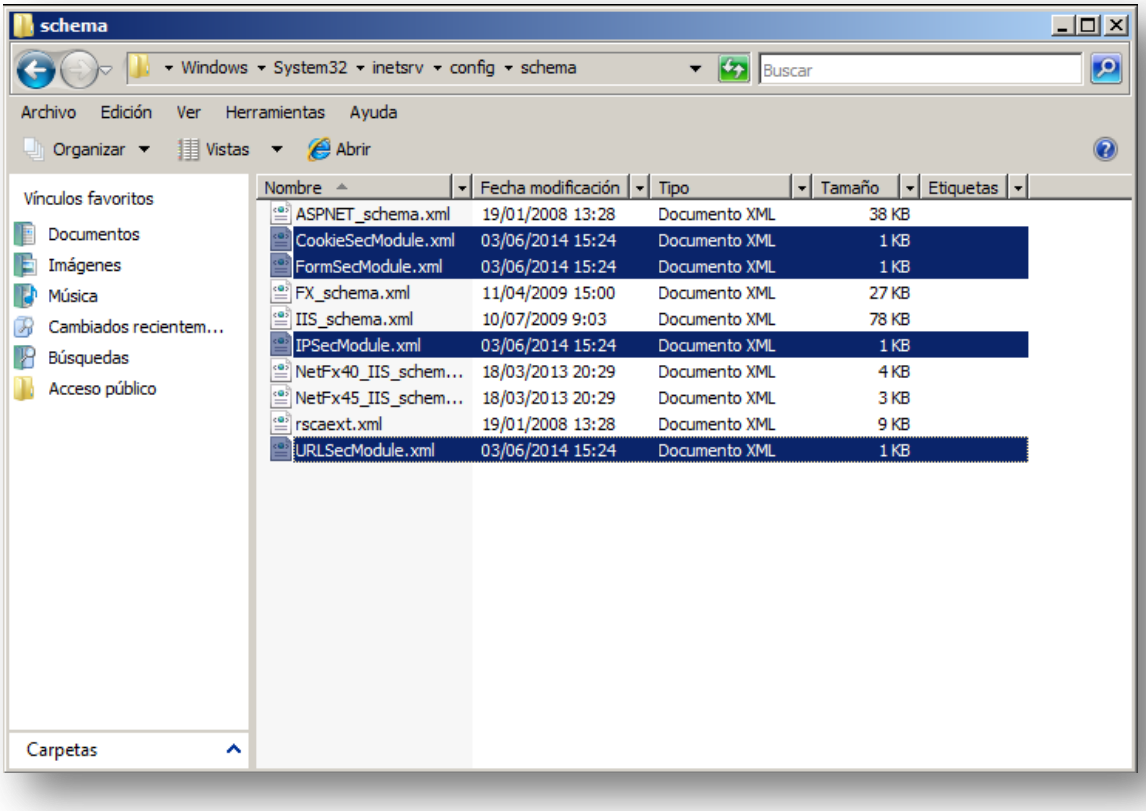


Ilustración 9-8: Archivos de esquemas de configuración

Después, en el fichero %windir%/system32/inersrv/config/ApplicationHost.config, dentro de la sección "system.webServer" escribimos las siguientes líneas:

```
53      <sectionGroup name="system.webServer">
54          <section name="urlSecModule" overrideModeDefault="Allow" />
55          <section name="formSecModule" overrideModeDefault="Allow" />
56          <section name="cookieSecModule" overrideModeDefault="Allow" />
57          <section name="ipSecModule" overrideModeDefault="Allow" />
58          <section name="asp" overrideModeDefault="Deny" />
```

Ilustración 9-9: Secciones de configuración a incluir en IIS

Una vez hecho esto, comprobamos para cada sección de configuración publicada que ha sido correctamente en el servidor. Para ello utilizaremos la herramienta "appcmd". Abrimos el intérprete de comandos, vamos a la carpeta %windir%/system32/inetsrv y utilizamos la herramienta con la siguiente sintaxis. El resultado de cada llamada debería ser el mismo que el mostrado en la imagen.

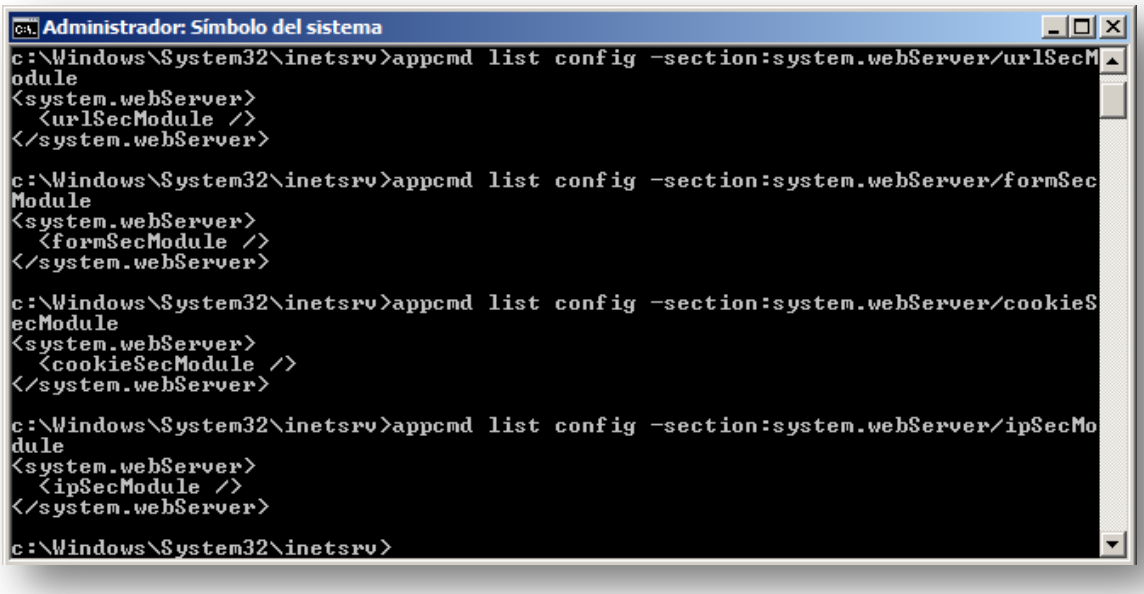


Ilustración 9-10: Comprobación de la presencia de las nuevas secciones de configuración

El siguiente paso es añadir el ensamblado de la interfaz del módulo a la caché global de ensamblados. Esto lo haremos mediante la herramienta "gacutil". Copiaremos el ensamblado (archivo SecurityModuleConf.dll) a una localización cualquiera (el escritorio sirve) y, desde la interfaz de comandos ejecutaremos la herramienta con la siguiente sintaxis, tras lo cual deberá aparecer un mensaje de que el ensamblado se ha añadido correctamente a la caché.

```
C:\Users\Administrador\Desktop>gacutil -i SecurityModuleConf.dll
Microsoft (R) .NET Global Assembly Cache Utility.  Version 3.5.21022.8
Copyright (c) Microsoft Corporation.  All rights reserved.

Assembly successfully added to the cache

C:\Users\Administrador\Desktop>_
```

Ilustración 9-11: Añadido del ensamblado de la interfaz a la caché global mediante la gacutil

Después, en la sección <configuration><moduleProviders> del archivo %windir%/system32/inetsrv/config/administration.config escribiremos el siguiente elemento:

```
        <add name="SecurityModuleConf" type=
        "IIS7Modules.UIModuleProvider, SecurityModuleConf,
        Version=1.0.0.0, Culture=neutral,
        PublicKeyToken=1351e6604a8f186e, processorArchitecture=MSIL"
        />
    </moduleProviders>
```

Ilustración 9-12: Añadido de la interfaz al archivo de administración de IIS

Por último, en la sección <configuration>/<location path=".">/<modules> del mismo archivo añadiremos el siguiente elemento:

```
        <add name="SecurityModuleConf" />
    </modules>
</location>
```

Ilustración 9-13: Añadido de la interfaz a todas las aplicaciones administradas por IIS

Abrimos el administrador de IIS y vamos a una aplicación cualquiera para comprobar que efectivamente aparece el módulo el área central.

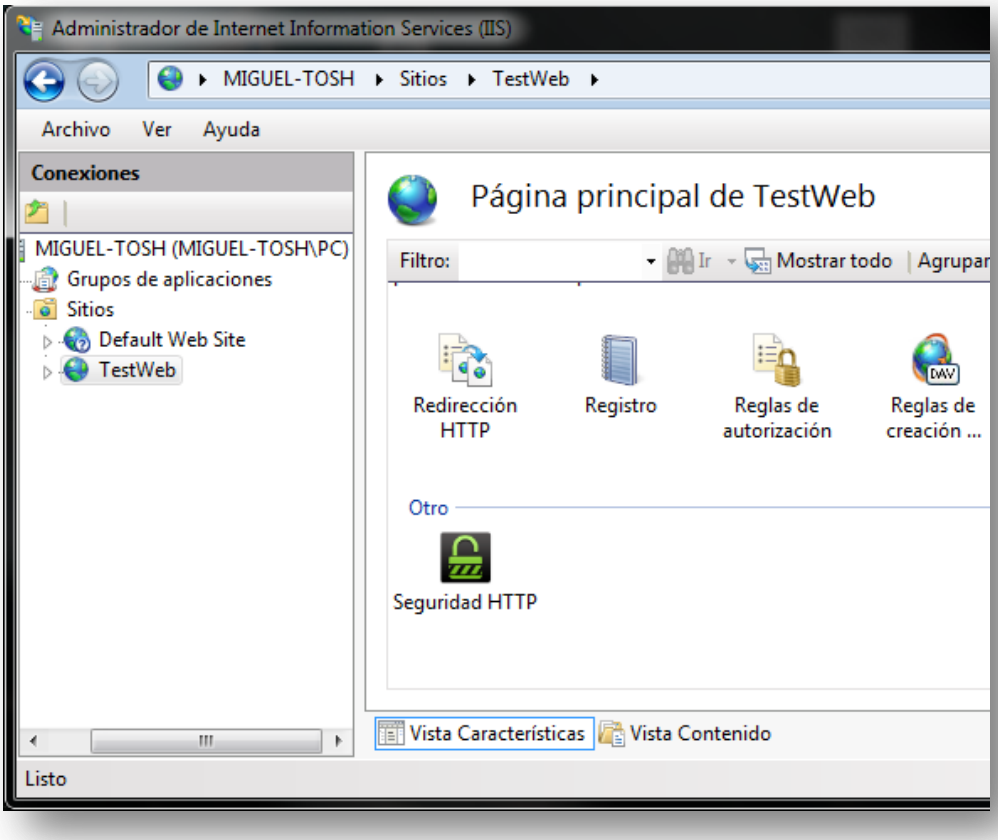


Ilustración 9-14: Icono de la interfaz dentro del Administrador de IIS

9.2.2 Publicación de la Aplicación Web

Para ilustrar cómo publicar una aplicación web partiremos de una ya desarrollada. Adjunta a este documento se proporciona una aplicación de prueba. Copiaremos la carpeta raíz de la misma en el directorio /inetpub/wwwroot.

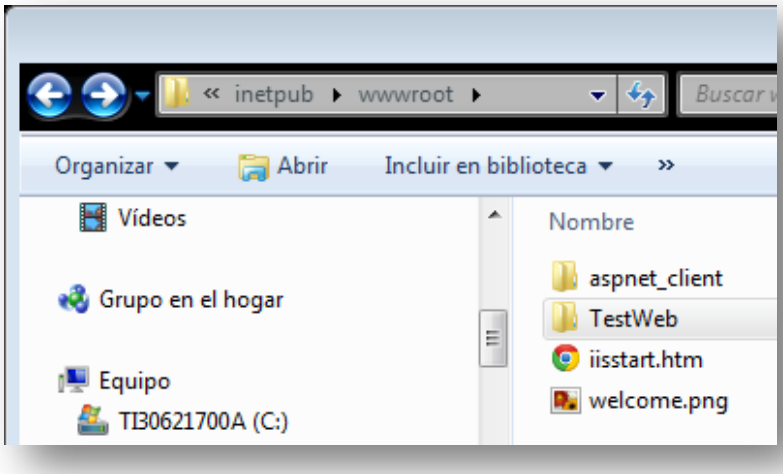


Ilustración 9-15: Directorio wwwroot de IIS

Una vez hecho esto abriremos el administrador de IIS, desplegaremos el nodo principal y, haciendo click derecho sobre la carpeta "Sitios" seleccionaremos "Agregar sitio web…"

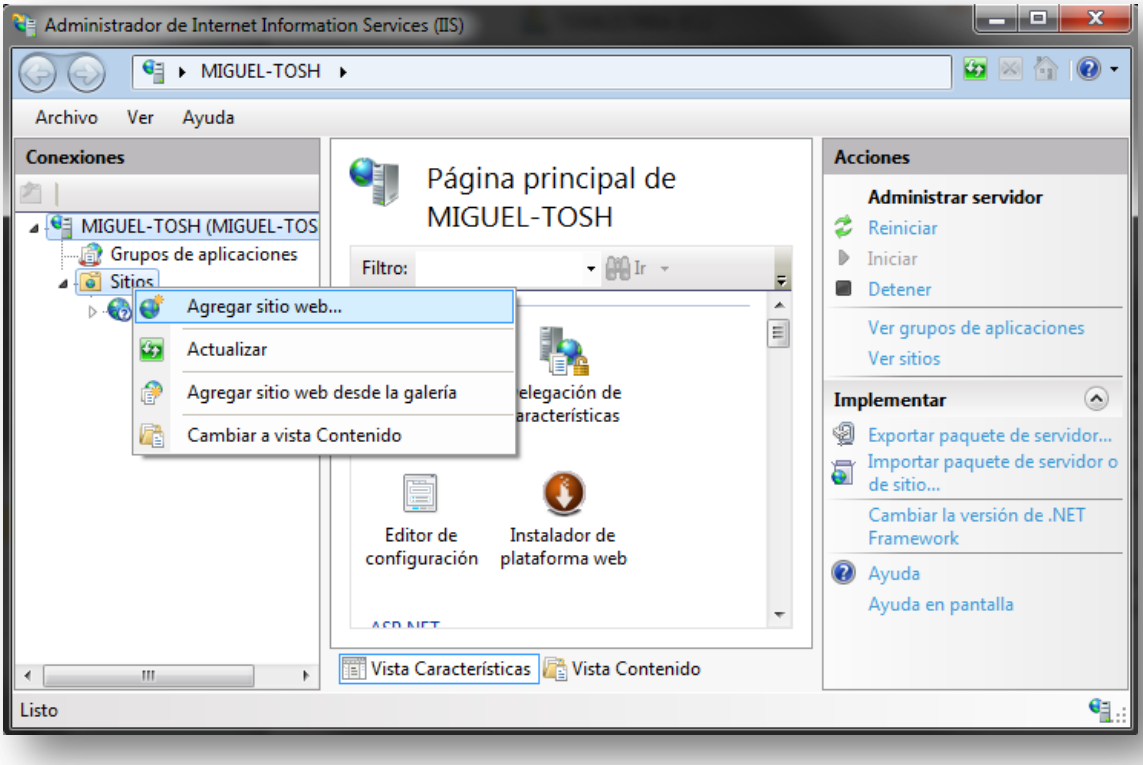


Ilustración 9-16: Agregar sitio web al servidor IIS

En la pantalla que nos aparecerá, estableceremos la ruta física del sitio web en el path de la carpeta que copiamos en el directorio wwwroot en el primer paso, le daremos un nombre al sitio web, y estableceremos un puerto que no esté utilizado (o dejamos el 80, pero como consecuencia tendremos que detener cualquier otra aplicación que se estuviese ejecutando en dicho puerto). El grupo de aplicaciones lo dejaremos en "DefaultAppPool".

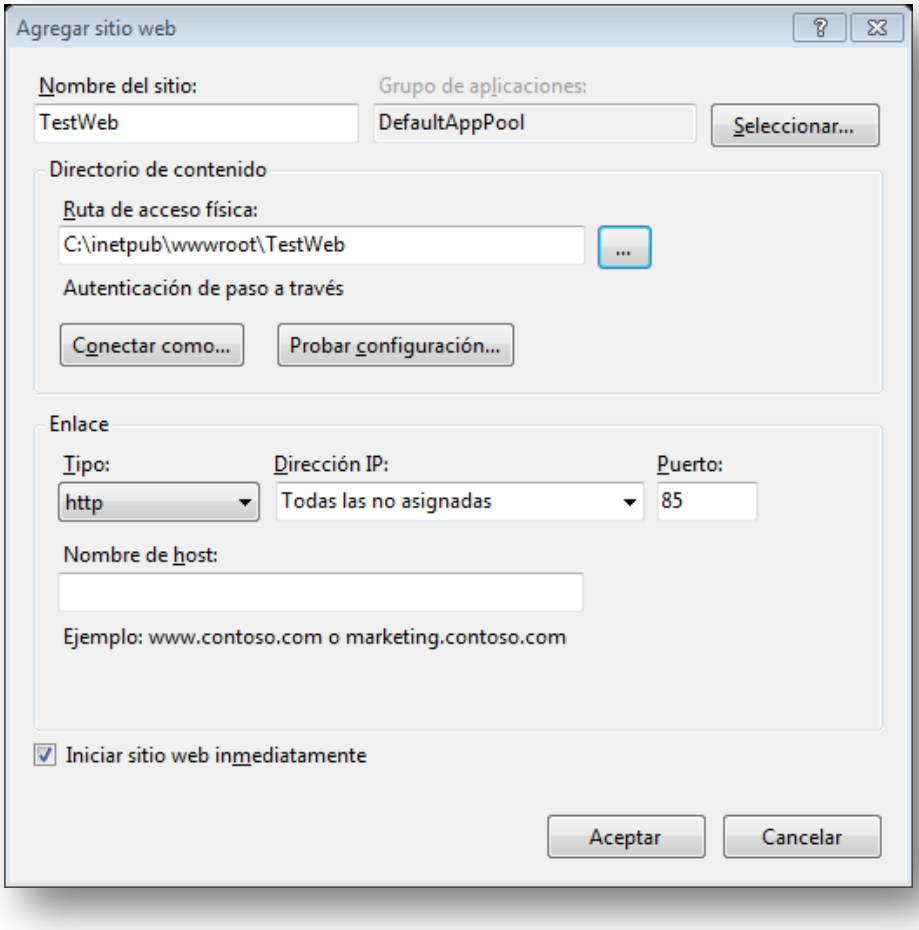


Ilustración 9-17: Opciones para agregar sitio web

Una vez hecho esto, comprobamos que podemos acceder a la aplicación publicada.

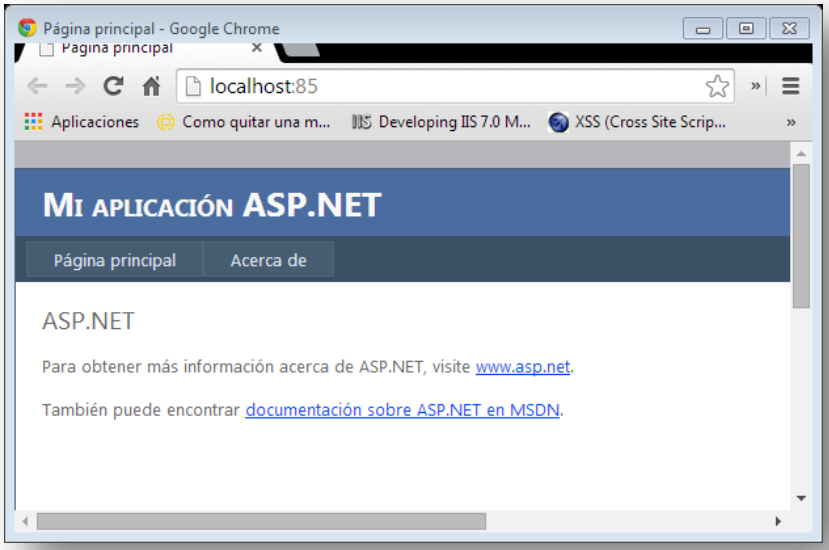


Ilustración 9-18: Acceso a la aplicación web desde el navegador

9.2.3 Activación del Módulo en la Aplicación

Para activar el módulo en la aplicación web recién publicada en IIS, copiaremos el ensamblado SecurityModule.dll al directorio /bin de la aplicación (si no existe, lo creamos). Una vez hecho esto, desde el administrador de IIS vamos a la característica "Módulos" de la aplicación web.

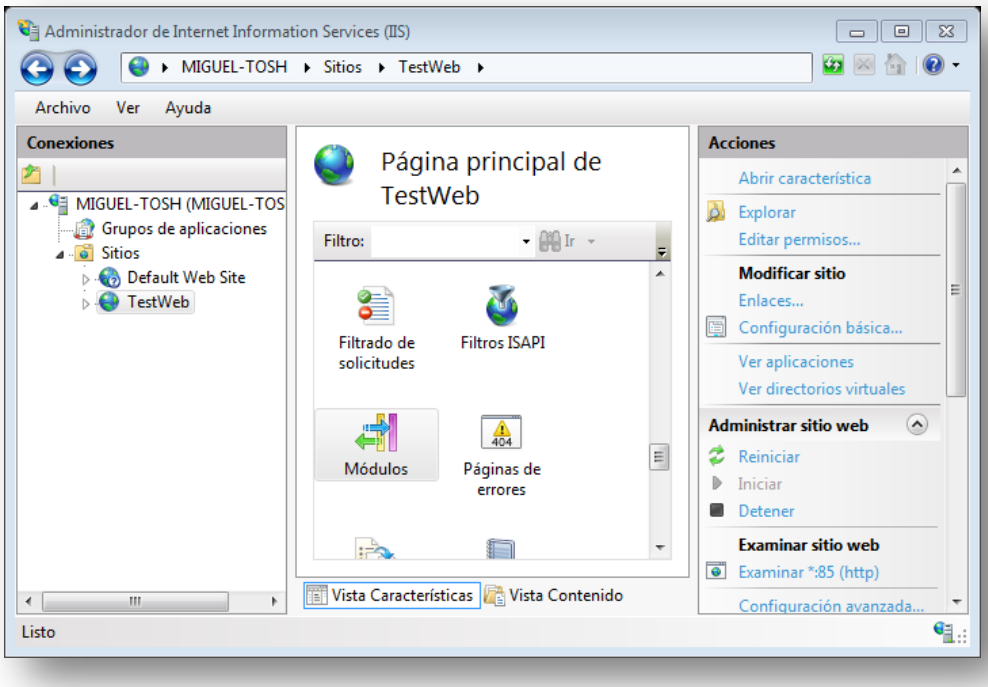


Ilustración 9-19: Vista "Características" de la aplicación web

Desde ahí hacemos click en "Agregar módulo administrado…" en la parte derecha de la interfaz. En la ventana que aparecerá introduciremos un nombre descriptivo cualquiera, pero en el tipo introduciremos "IIS7Modules.SecurityModule"

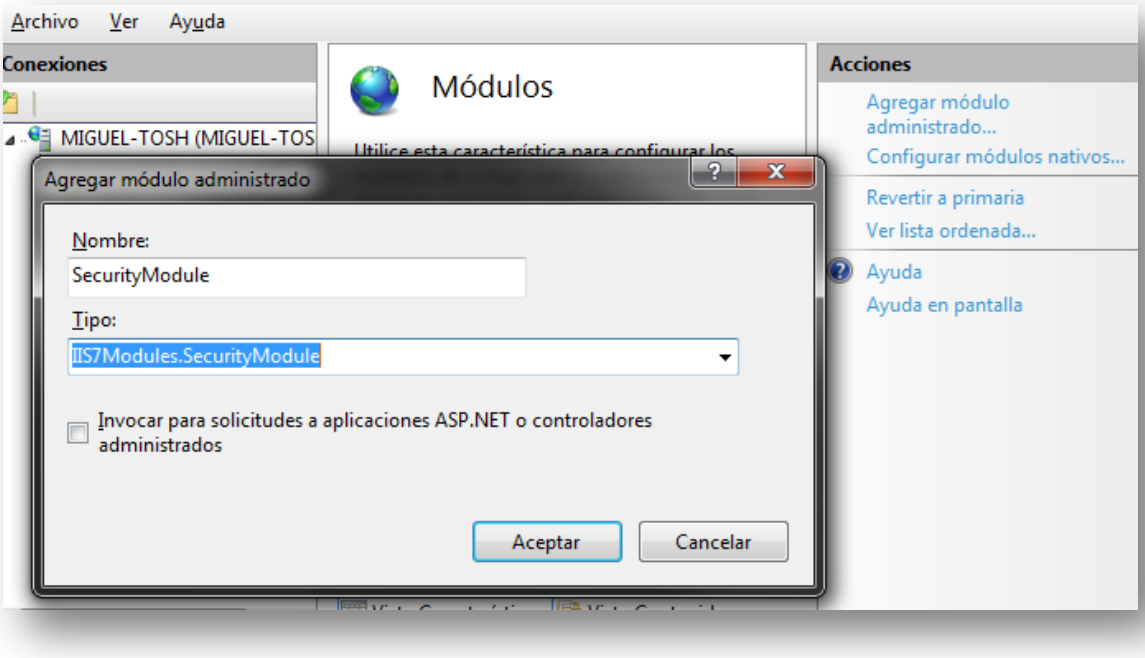


Ilustración 9-20: Agregado de nuevo módulo a la aplicación web

Una vez hecho esto, el módulo queda activado para la aplicación, pudiendo acceder ya a la interfaz y operar con el mismo.

9.3 Manual de Usuario

El módulo proporcionado tiene capacidad para detectar elementos web mal formados o con código inyectado. Estos elementos web son la URL, datos introducidos por el usuario mediante formularios, y las cookies HTTP. En cuanto a estas últimas, el módulo puede eliminar la persistencia de las mismas y convertirlas en cookies de sesión. Por último, el módulo proporciona protección contra peticiones HTTP cuyo origen sea una IP registrada en la DNSBL www.blocklist.de , con capacidad para enviar un correo electrónico al usuario de IIS informándole en caso de que una determinada IP haya realizado varios intentos.

Toda esta funcionalidad se configura desde la interfaz del módulo. Para acceder a la misma, primero hemos de abrir el Administrador de IIS.

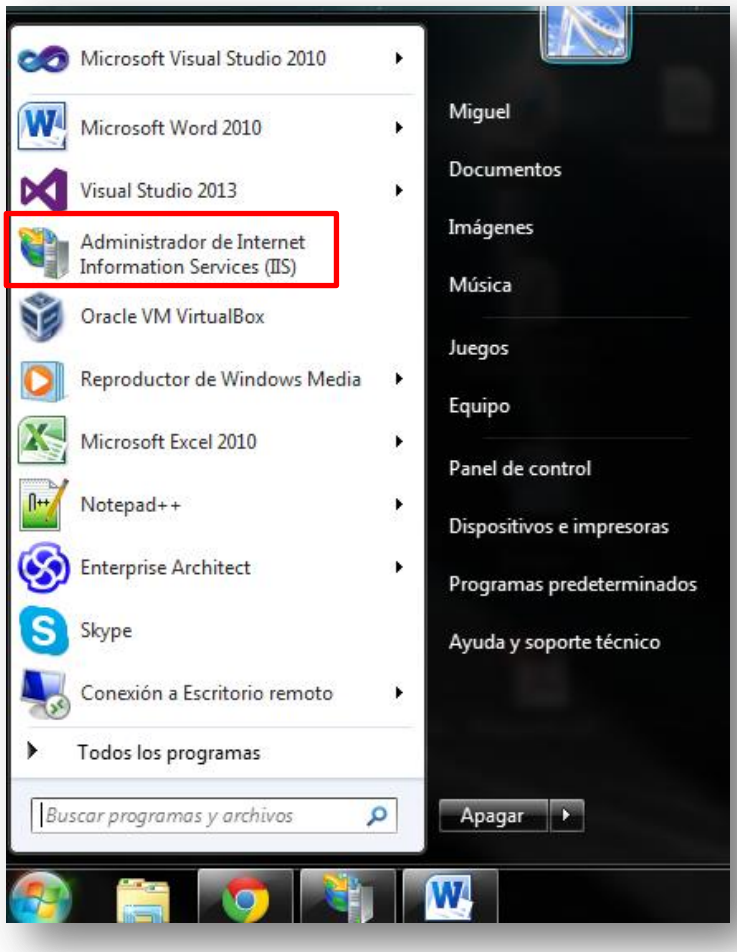


Ilustración 9-21: Icono del administrador de IIS

Una vez ahí, seleccionamos la aplicación web para la que queremos administrar la configuración del módulo y hacemos doble click sobre el icono del módulo, un candado verde que estará abajo del todo designado con el nombre "Seguridad HTTP".

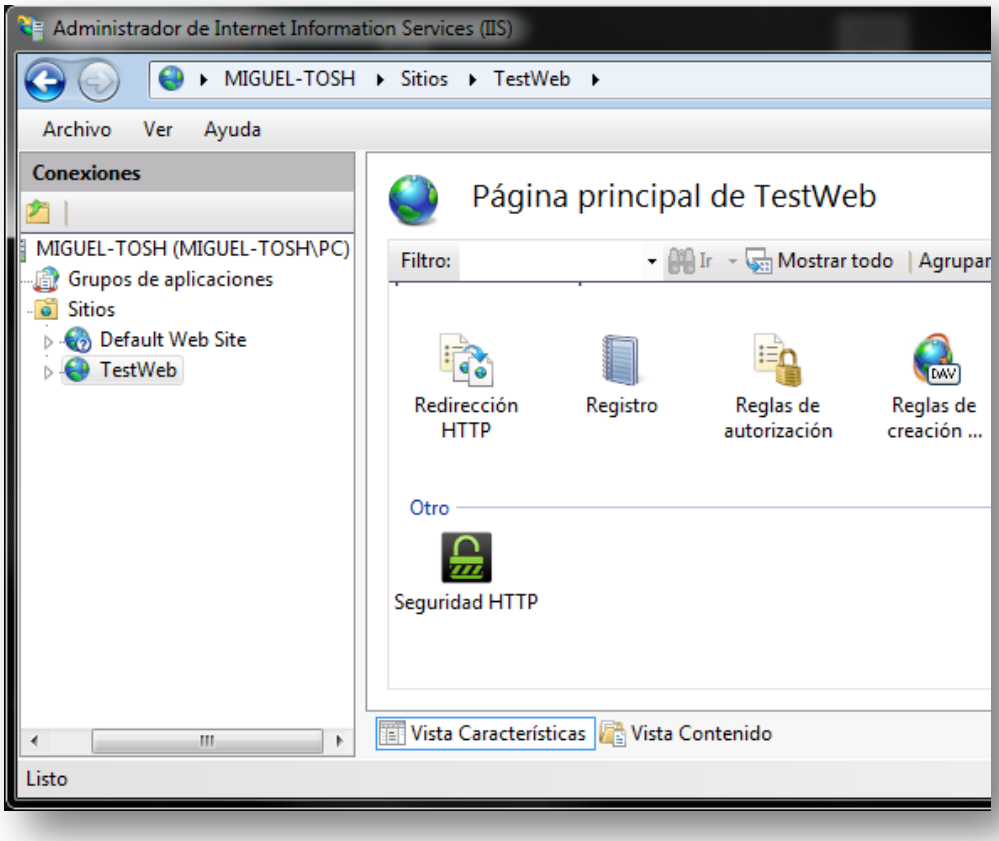


Ilustración 9-22: Icono de la interfaz dentro del Administrador de IIS

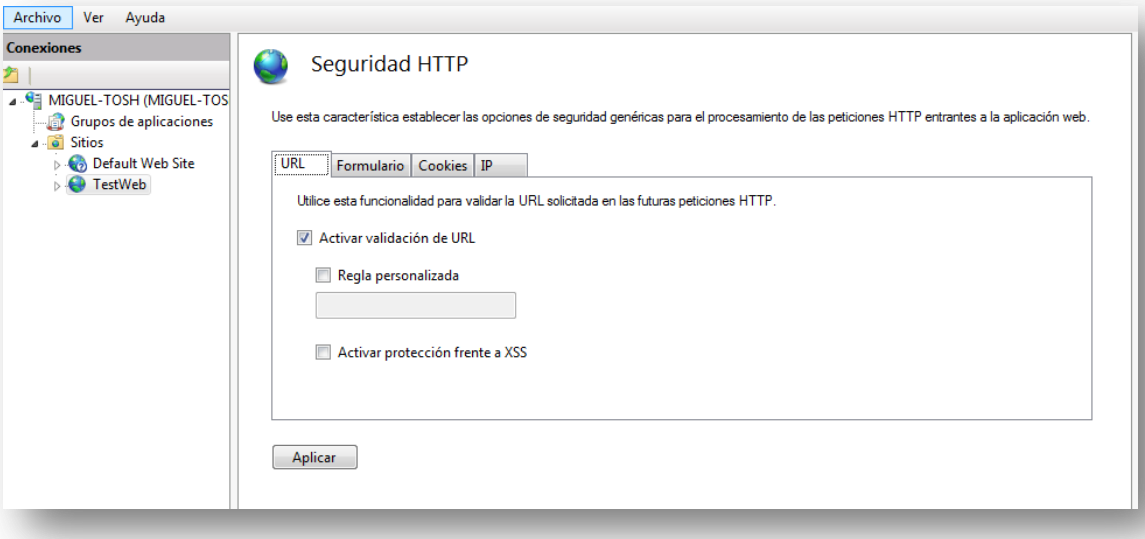


Ilustración 9-23: Interfaz del módulo

Una vez en la interfaz del módulo, podemos movernos por cualquiera de las cuatro pestañas para configurar el comportamiento del módulo con respecto a cada uno de los elementos que procesa.

- **Pestaña URL:** Proporciona validación de la URL, bien mediante una expresión regular definida por defecto, o bien mediante una que especifique el usuario si marca la casilla "Regla personalizada" y escribe una expresión regular válida (En caso de que no sea válida, se avisará del error con un indicador al lado del campo de texto). Además, se proporciona protección aparte de la validación en sí contra ataques de tipo cross-site scripting.
- **Pestaña Formulario:** Al igual que la URL, se proporciona validación para los datos enviados por el usuario de la aplicación mediante formularios, tanto por defecto como personalizada, y protección contra cross-site scripting. Además, también se provee de protección contra ataques SQL Injection.
- **Pestaña Cookies:** Contiene las mismas funcionalidades que la pestaña "Formulario", pero en este caso aplicadas a las cookies HTTP intercambiadas entre cliente y servidor. Asimismo, se da al usuario la opción de eliminar la persistencia en las mismas.
- **Pestaña IP:** En esta pestaña se configura el comportamiento del módulo cara a IPs registradas en la DNSBL www.blocklist.de . Si se desea, se puede especificar una dirección email a informar sobre repetidos intentos de acceso por parte de una misma IP. Este número de intentos es también configurable desde esta pestaña. Además, al igual que en el caso de las expresiones regulares introducidas por el usuario, se proporciona validación del formato de la dirección de correo proporcionada.

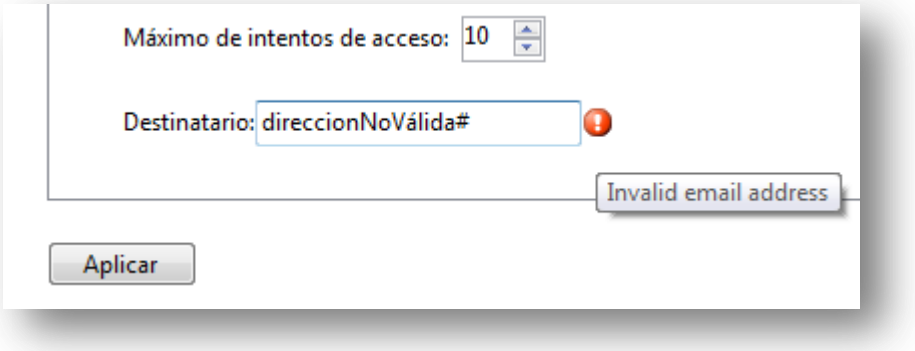


Ilustración 9-24: Validación de la dirección de correo

9.3.1 Ejemplo de Funcionamiento

A continuación se muestra un ejemplo de funcionamiento del módulo. Partimos de tenerlo activado para una aplicación web del servidor.

Como ya se ha explicado, el primer paso será acceder al administrador de IIS, desde el cual iremos a la interfaz del módulo. Una vez ahí, nos limitaremos a activar la validación de elementos de formulario, sin personalizar la expresión regular, pero activando la protección frente a SQLI, tal y como muestra la siguiente imagen.

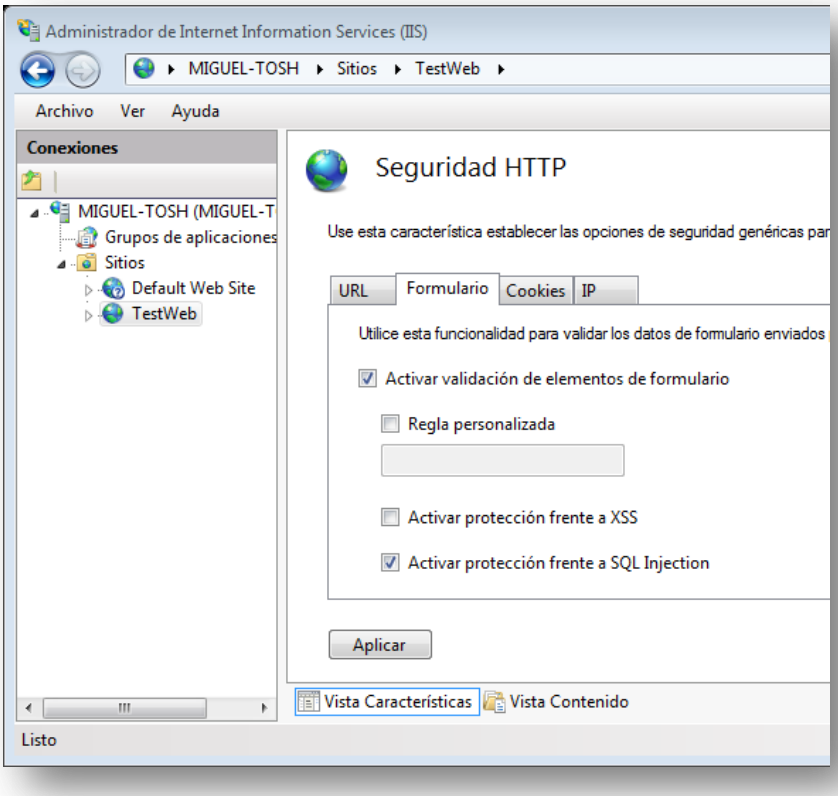


Ilustración 9-25: Validación de elementos de formulario

Aplicamos las opciones establecidas y las probamos accediendo a la aplicación web desde el navegador, concretamente a la pantalla de login para poder enviar datos a la aplicación mediante formulario. Introduciendo datos válidos, comprobamos que iniciamos sesión correctamente.

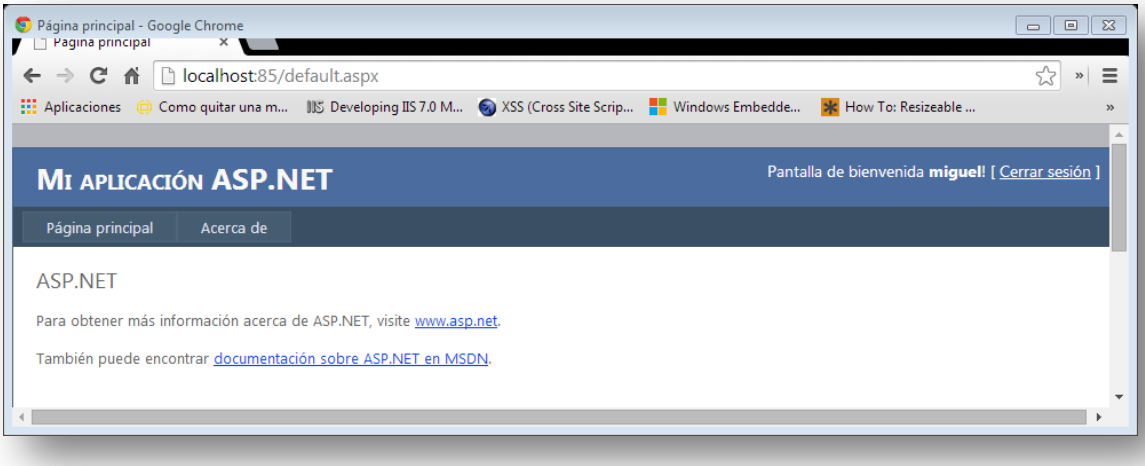


Ilustración 9-26: Inicio de sesión correcto en la aplicación

Sin embargo, comprobamos que intentando inyectar código SQL en la aplicación, el navegador nos muestra un error.

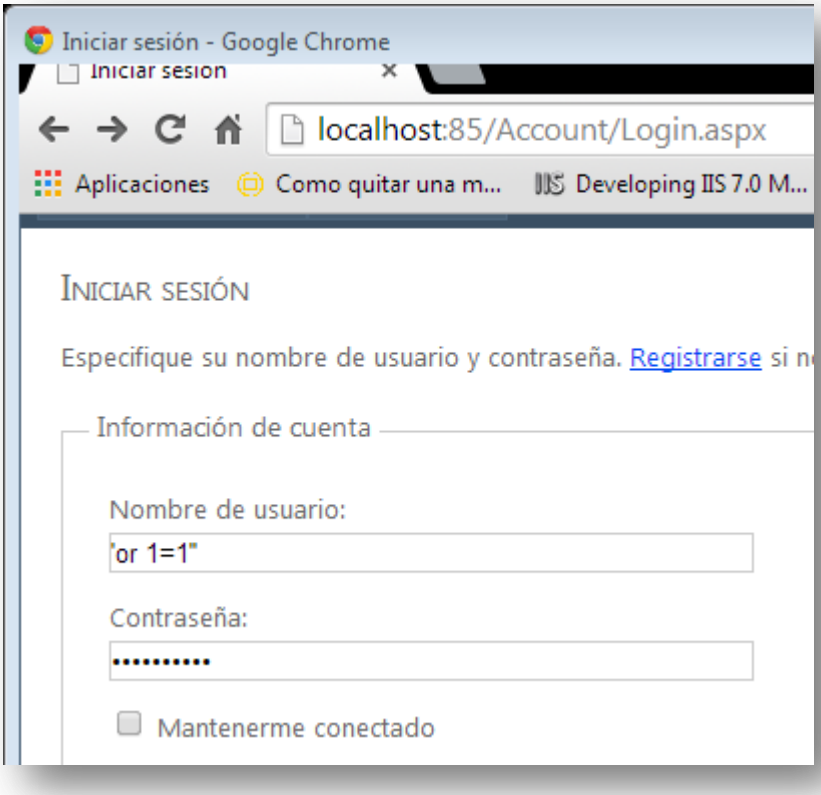


Ilustración 9-27: Datos introducidos en el formulario (contraseña irrelevante ya que no se va a llegar a procesar)

Es posible que, si no hemos configurado IIS para que muestre mensajes de error personalizados, no aparezcan detalles del error en el navegador. Para obtenerlos, o bien configuramos IIS para que muestre una página de error personalizada para el código HTTP 403, o bien comprobamos en el Visor de Eventos de Windows la naturaleza del mismo.

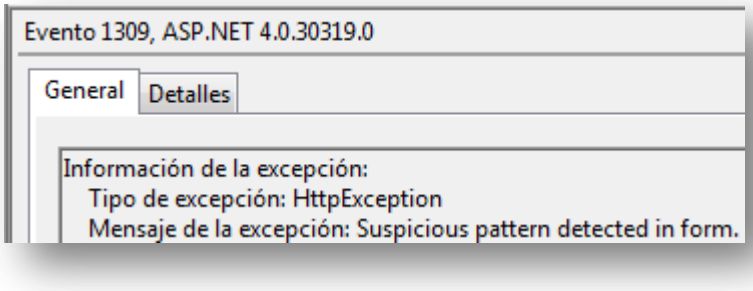


Ilustración 9-28: Error 403 mostrado en el visor de eventos

9.4 Manual del Programador

Cara a ampliar la funcionalidad del módulo, la primera consideración a tener en cuenta es que está formado por dos ensamblados diferentes: el de la interfaz, y el del módulo en sí.

Cabe destacar que en ambos ensamblados se han separado unas funcionalidades de otras según el elemento web al que afecten.

9.4.1 Interfaz

En cuanto a la interfaz, las clases UIModuleProvider, UIModule y AdminPage no deberían modificarse, pues son las responsables de que la interfaz se integre con el programa administrador de IIS. La clase ModuleConfiguration es la que implementa la interfaz en sí (elementos gráficos), con lo que en caso de añadir más pestañas o elementos web a configurar, esta es la clase que deberemos modificar.

En caso de que queramos añadir más elementos web, probablemente deberemos añadir otra clase Settings que modele las opciones a implementar de los mismos.

9.4.2 Módulo

En caso de que se quiera ampliar la funcionalidad del módulo, una vez se hayan hecho los cambios pertinentes en la interfaz, se debería añadir una nueva clase Validator. En la clase SecurityModule hay una lista con los validadores actuales, la cual se recorre llamando al método Validate() de cada uno de ellos, con lo cual habrá que añadir el nuevo validador a esta lista.

Si se quiere modificar el funcionamiento de alguna de las funcionalidades, se debería empezar modificando el método Validate() de la clase Validator a la que corresponde.

Capítulo 10. Conclusiones y Ampliaciones

10.1 Conclusiones

Se ha desarrollado un módulo integrable con el servidor IIS que lleva a cabo varias opciones relacionadas con la seguridad de las aplicaciones gestionadas por dicho servidor:

- **Validación del formato de elementos web**: Se comprueba que los elementos de formulario introducidos por el usuario, URL y valor de las cookies sigan una determinada sintaxis mediante expresiones regulares. Además, se proporciona la posibilidad de personalización de las reglas a aplicar por parte del usuario.
- **Búsqueda de ataques de inyección de código**: Relacionado con el punto anterior, se comprueba que los elementos mencionados no contienen ningún ataque de inyección de código.
- **Eliminación de persistencia en cookies**: Se elimina la persistencia de las cookies, de forma que pasan a ser todas de sesión.
- **Comprobación de IP contra una DNS BL**: Se comprueba que la IP desde la que se realizan las peticiones no esté registrada en la DNS BL de http://www.blocklist.de/ , enviando un correo al usuario del módulo en caso de repetidos intentos desde una misma IP.
- **Interfaz gráfica integrada en el programa de administración de IIS**: Se ha desarrollado una interfaz gráfica acoplada al programa de administración de IIS desde la que se pueden activar o desactivar las funcionalidades anteriores, así como editar ciertos parámetros del procesamiento de la petición por parte del módulo.

Además, se ha proporcionado documentación sobre el módulo, así como sobre el desarrollo en sí del proyecto.

De lo anterior se concluye que se han cumplido los objetivos del proyecto.

10.2 Ampliaciones

Como ampliaciones futuras al proyecto se consideran las siguientes posibilidades:

- **Desarrollo de un WAF totalmente independiente**: La principal ampliación sería replicar la funcionalidad actual del módulo en un programa totalmente independiente. El módulo desarrollado es un plugin del servidor IIS, con lo que únicamente funciona para aplicaciones gestionadas por este. Servidores igualmente populares como Apache o Tomcat no se pueden aprovechar de las ventajas del módulo desarrollado. El motivo de no haber llevado a cabo esta ampliación es que no era objetivo del proyecto, así como la complejidad que conlleva, ya que, al ser totalmente independiente del servidor web, el programa tendría que comunicarse directamente con el hardware de red de la máquina en la que estuviese instalado.
- **Analizar los scripts ejecutados en el cliente y mandados al servidor**: Otro elemento que se podría haber analizado son los scripts enviados desde el servidor al cliente, ejecutados en la máquina del servidor, y enviados de nuevo al cliente con los datos del usuario ya introducidos. Si el usuario cuenta con las herramientas para capturar peticiones y modificar estos scripts, y el servidor no comprueba que los scripts devueltos por el cliente son los mismos que le envió previamente, es posible que el usuario consiga ejecutar código ilegítimo en la máquina del servidor. No se llevó a cabo esta ampliación por el tiempo y complejidad que supone.
- **Revisar integridad de las cookies**: Además de la posibilidad que ya permite el módulo de eliminar la persistencia de las cookies, también se podría validar que las cookies enviadas por el cliente al servidor, son las mismas que le ha enviado previamente el servidor al cliente. Esto permitiría cubrir un abanico de ataques web mayor, pues ataques como el secuestro de sesión serían mucho más difíciles de llevar a cabo. Se intentó implementar esta funcionalidad, pero por varios problemas encontrados y por falta de tiempo se decidió omitir del proyecto.
- **Seleccionar los elementos de formulario a los que se quiere afectar:** Por último, sería útil que desde la interfaz del módulo se mostrasen todos los elementos de formulario de la aplicación web para la que está activado. Actualmente, la regla de validación (tanto la personalizada por el usuario como la aplicada por defecto), así como las reglas de detección de ataques de inyección de código, se aplican indistintamente a todos los datos enviados por el usuario a través de formularios web, cuando es probable que el usuario del módulo necesitase una validación concreta para cada elemento de los formularios. Esta ampliación no se llevó a cabo por falta de tiempo.

Capítulo 11. Presupuesto

11.1 Presupuesto del Cliente

A continuación se presenta el presupuesto final a mostrar al cliente. Los ítems principales del mismo son dos; el desarrollo en sí del módulo y la documentación del mismo. Los subítems del desarrollo corresponden a las diferentes fases del mismo; análisis, diseño, implementación y pruebas, siendo los honorarios los mismos para todas las fases.

En cuanto a la metodología a seguir, al cliente se le especificará que no se abordará la fase de implementación (ni, por consiguiente, la de pruebas), hasta que no apruebe y firme el análisis y diseño llevados a cabo previamente. El objetivo de esto es evitar cambios en los requisitos o en el diseño final de la aplicación antes de pasar a la implementación de la misma, o al menos no sin antes negociar un nuevo presupuesto.

Item	Sub-item	Concepto	Cantidad	Precio Unitario	TOTAL
1		**Desarrollo software: Módulo de seguridad para IIS 7**			8.100,00 €
	1.1	Análisis	21 horas	30,00 €	630,00 €
	1.2	Diseño	42 horas	30,00 €	1.260,00 €
	1.3	Implementación	167 horas	30,00 €	5.010,00 €
	1.4	Pruebas	40 horas	30,00 €	1.200,00 €
2		**Manuales y documentación**			180,00 €
		Desarrollo	4 horas	30,00 €	120,00 €
		Impresión de ejemplares	3 ejemplares	20,00 €	60,00 €
				Subtotal	8.280,00 €
				IVA (21%)	1.738,80 €
				TOTAL	**10.018,80 €**

Por último cabe destacar que en este presupuesto no se incluyen conceptos tales como los gastos de administración o la amortización de los equipos, sino que sólo se muestran aquellos ítems que el cliente ha pedido. El coste real del proyecto está incluido en los honorarios percibidos por el desarrollador, y su desglose se muestra en la siguiente sección, el presupuesto de costes.

11.2 Presupuesto de Costes

A continuación se muestra una tabla con el coste real del proyecto, desglosando de los gastos de administración la amortización, servicios de comunidad y otros elementos que es necesario utilizar o amortizar para llevar a cabo el proyecto. En este presupuesto no se incluyen los honorarios percibidos por el desarrollador.

Item	Sub-item	Concepto	Cantidad	Precio Unitario	TOTAL
1		***Gastos de administración***			**737,08 €**
	1.1	Seguro	4 meses	17,33 €	69,32 €
	1.2	Licencia Visual Studio 2013	1 licencia	99,80 €	99,80 €
	1.3	Licencia Enterprise Architect 10	1 licencia	39,80 €	39,80 €
	1.4	Agua, luz y recogida de basura	4 meses	90,00 €	360,00 €
	1.5	Equipo informático (amortización)	4 meses	26,09 €	104,36 €
	1.6	Teléfono/Internet	4 meses	15,95 €	63,80 €
2		**Documentación**			**60,00 €**
	2.1	Encuadernación	3 ejemplares	20,00 €	60,00 €
				TOTAL	**797,08 €**

Capítulo 12. Referencias Bibliográficas

12.1 Libros y Artículos

[Stuttard11]Stuttard, Dafydd; Pinto, Marcus. "The Web Application Hacker's Handbook". Wiley Publishing Inc. 2011. ISBN 978-1-118-02647-2

[Ray10]Ray, Donald; Ligatti, Jay. "Defining Code-injection Attacks" http://www.cse.usf.edu/~ligatti/papers/code-inj.pdf

12.2 Referencias en Internet

12.2.1 Referencias consultadas

"Funcionamiento de IIS" http://www.iis.net/learn 03/2014

"Página de preguntas-respuestas para programadores" http://stackoverflow.com/ 04/2014-06/2014

"Red de Microsoft de desarrolladores" http://msdn.microsoft.com/es-es/dn308572.aspx 04/2014-06/2014

"Foro de MSDN" http://social.msdn.microsoft.com/Forums/en-US/home 05/2014

12.2.2 Referencias utilizadas en el Documento

[1] "Explicación sobre el funcionamiento del protocolo HTTP" http://www.w3.org/Protocols/rfc2616/rfc2616.html 03/2014

[2] "Especificación HTTP 1.1" http://www.w3.org/Protocols/rfc2616/rfc2616.html 03/2014

[3] "Explicación sobre el funcionamiento de las cookies HTTP" http://www.allaboutcookies.org/ 04/2014

[4] "Introducción a las expresiones regulares" http://www.regular-expressions.info/ 03/2014

[5] "Funcionamiento básico de las direcciones IP" http://computer.howstuffworks.com/internet/basics/question549.htm 04/2014

[6] "Especificación del protocolo IP" http://tools.ietf.org/html/rfc760 04/2014

[7] "Bases y funcionamiento de las DNS BL" http://www.rblmon.com/blog/what-are-rbls-and-how-do-they-work/ 05/2014

[8] "Revisión de DNSBL" http://www.faqs.org/rfcs/rfc6471.html#b 05/2014

[9] "Comparación de DNSBL" http://en.wikipedia.org/wiki/Comparison_of_DNS_blacklists 05/2014

[10] "Explicación sobre la inyección de código" https://www.owasp.org/index.php/Code_Injection 05/2014

[11]"Ejemplos de ataques de SQL Injection" http://www.unixwiz.net/techtips/sql-injection.html 05/2014

Capítulo 13. Apéndices

13.1 Glosario

- **Cross-site scripting (XSS):** Ataque de inyección de código consistente en introducir código Javascript en un recurso web para que se ejecute en las máquinas cliente que accedan a dicho recurso.
- **Expresión regular**: Conjunto de símbolos que dicta unas determinadas reglas a seguir por las cadenas de caracteres para ser reconocidas por dicho conjunto de símbolos, o expresión regular.
- **HTTP**: Protocolo de aplicación utilizado en Internet. Basado en un mecanismo de peticiones por parte de los clientes, y respuestas por parte de los servidores.
- **HTTP Cookie**: Fichero de texto intercambiado entre cliente y servidor con el fin de suplir la falta de estado del protocolo HTTP
- **IIS**: Servidor web de Microsoft. Destaca por tener una arquitectura de módulos en estructura de pipeline, a través de la cual las peticiones HTTP van pasando y van siendo procesadas para generar una respuesta a enviarle al cliente.
- **Inyección de código:** Conjunto de ataques informáticos consistentes en introducir y ejecutar código ilegítimo en una máquina remota.
- **SQL Injection:** Ataque de inyección de código consistente en introducir código SQL en una petición HTTP para lograr que se ejecute de forma ilegítima en una base de datos con el objetivo de afectar a su funcionamiento u obtener información de la misma.

13.2 Contenido Entregado

13.2.1 Contenidos

En esta sección se describe la estructura de carpetas de los contenidos anexos a la documentación del proyecto.

Directorio	Contenido
./	Contiene el directorio de los proyectos de Visual Studio, el de los ficheros de configuración, y los archivos EAP, Mockup, Project y Excel correspondientes a los diagramas, los diseños de interfaz, la planificación y los presupuestos del proyecto.
./proyectos_vs	Contiene los directorios correspondientes a los dos proyectos de Visual Studio 2013 del módulo y de su interfaz
./proyectos_vs/ConfModuloSeguridad	Contiene el proyecto de Visual Studio 2013 correspondiente a la interfaz del módulo desarrollado.
./proyectos_vs/ModuloSeguridad	Contiene el proyecto de Visual Studio 2013 correspondiente al módulo desarrollado.
./ficheros_configuracion	Contiene los ficheros xml de esquemas IIS para detectar la configuración del módulo para cada aplicación web.

13.2.2 Ficheros de Configuración

En la carpeta “ficheros_configuracion” se encuentran los cuatro .xml que representan los esquemas de configuración de IIS que permitirán a dicho servidor interpretar correctamente las opciones especificadas por cada aplicación web para el funcionamiento del módulo. Estos archivos, cuyo nombre describe a qué elemento web afectan, son los siguientes:

- CookieSecModule
- FormSecModule
- IPSecModule
- URLSecModule

Estos archivos han de ser copiados al directorio
%windir%/Windows/System32/inetsrv/config/schema

13.3 Código Fuente

13.3.1 Clases del Módulo

13.3.1.1 SecurityModule

```
namespace IIS7Modules
{

    public class SecurityModule : IHttpModule
    {
        List<Validator> validators;

        #region IHttpModule methods

        public void Init(HttpApplication application)
        {
            validators = new List<Validator>();
            validators.Add(new URLValidator());
            validators.Add(new FormValidator());
            validators.Add(new CookieValidator());
            validators.Add(new IPValidator());
            application.PreRequestHandlerExecute += new
EventHandler(OnPreRequestHandlerExecute);
        }

        public void Dispose() { }

        #endregion

        #region Module events handlers

        public void OnPreRequestHandlerExecute(Object source, EventArgs e)
        {
            HttpApplication app = (HttpApplication)source;
            HttpContext context = app.Context;
            foreach (Validator validator in validators)
            {
                validator.Validate(context);
            }
        }
        #endregion
    }
}
```

13.3.1.2 Validator

```
namespace IIS7Modules
{
    public abstract class Validator
    {
        public virtual void Validate(HttpContext context) { }
    }
}
```

13.3.1.3 URLValidator

```
namespace IIS7Modules
{
    class URLValidator : Validator
    {

        private readonly static String urlRegex = @"^(https?|ftp)\://[a-z0-9\-
\.]+\.?:?[a-z]+(:[a-z0-9]*)?/?([a-z0-9\-
\._\?\,\'/\\\+&amp;%\$#\=~])*[^\.\,\)\(\s]$";
        private readonly static String xssRegex =
@"(?i)(<script[^>]*>[\s\S]*?<\/script[^>]*>|<script[^>]*>[\s\S]*?<\/script[[\s\S]
]*[\s\S]|<script[^>]*>[\s\S]*?<\/script[\s]*[\s]|<script[^>]*>[\s\S]*?<\/script|<
script[^>]*>[\s\S]*?)";
        private Analyzer urlAnalyzer;
        private URLSettings urlSettings;
        ServerManager serverManager;
        Configuration configuration;

        public URLValidator() : base()
        {
            urlAnalyzer = new Analyzer(urlRegex, xssRegex, null);
            serverManager = new ServerManager();
            configuration =
serverManager.GetWebConfiguration(System.Web.Hosting.HostingEnvironment.SiteName)
;
        }

        public override void Validate(HttpContext context)
        {
            RefreshSettings();

            if (urlSettings.urlValidation)
            {
                if (urlSettings.urlCustomRule)
                {
                    urlAnalyzer.SetValidationRegex(urlSettings.urlCustomRuleStr);
                }
                else
                {
                    urlAnalyzer.SetValidationRegex(urlRegex);
                }

                HttpRequest request = context.Request;
                string url = request.Url.ToString();
                if (!urlAnalyzer.Analyze(url, urlSettings.urlXssProtection,
false))
```

```
                    throw new HttpException(403, "Suspicious pattern detected in
URL.");
                }
            }

            private void RefreshSettings()
            {
                urlSettings =
(URLSettings)configuration.GetSection("system.webServer/urlSecModule",
typeof(URLSettings));
            }
        }
}
```

13.3.1.4 FormValidator

```
namespace IIS7Modules
{
    class FormValidator : Validator
    {
        private readonly static String formRegex = @"^[/a-z0-9\s\+\.@-
_\*ñ:\?+áéíóúü=¿]+$";
        private readonly static String xssRegex =
@"(?i)(<script[^>]*>[\s\S]*?<\/script[^>]*>|<script[^>]*>[\s\S]*?<\/script[[\s\S]
]*[\s\S]|<script[^>]*>[\s\S]*?<\/script[\s]*[\s]|<script[^>]*>[\s\S]*?<\/script|<
script[^>]*>[\s\S]*?)";
        private readonly static String sqliRegex = @"(/\*!?|\*/|[';]--|--
[\s\r\n\v\f]|(?:--[^-]*?-)|([^\-&])#.*?[\s\r\n\v\f]|;?\\x00)";
        private Analyzer formAnalyzer;
        private FormSettings formSettings;
        ServerManager serverManager;
        Configuration configuration;

        public FormValidator() : base()
        {
            formAnalyzer = new Analyzer(formRegex, xssRegex, sqliRegex);
            serverManager = new ServerManager();
            configuration =
serverManager.GetWebConfiguration(System.Web.Hosting.HostingEnvironment.SiteName)
;
        }

        public override void Validate(HttpContext context)
        {
            RefreshSettings();

            HttpRequest request = context.Request;
            if (formSettings.formValidation && request.HttpMethod.Equals("POST"))
            {
                if (formSettings.formCustomRule)
                {

formAnalyzer.SetValidationRegex(formSettings.formCustomRuleStr);
                }
                else
                {
                    formAnalyzer.SetValidationRegex(formRegex);
                }
```

```
                NameValueCollection form = request.Form;
                for (int i = 0; i < form.Count; i++)
                {
                    if (!String.IsNullOrEmpty(form.Get(i)))
                    {
                        if (!formAnalyzer.Analyze(form.Get(i),
formSettings.formXssProtection, formSettings.formSqliProtection))
                        {
                            throw new HttpException(403, "Suspicious pattern
detected in form.");
                        }
                    }
                }
            }
        }

        private void RefreshSettings()
        {
            formSettings =
(FormSettings)configuration.GetSection("system.webServer/formSecModule",
typeof(FormSettings));
        }
    }
}
```

13.3.1.5 CookieValidator

```
namespace IIS7Modules
{
    class CookieValidator : Validator
    {
        private readonly static String cookieRegex = @".*";
        private readonly static String xssRegex =
@"(?i)(<script[^>]*>[\s\S]*?<\/script[^>]*>|<script[^>]*>[\s\S]*?<\/script[[\s\S]
]*[\s\S]|<script[^>]*>[\s\S]*?<\/script[\s]*[\s]|<script[^>]*>[\s\S]*?<\/script|<
script[^>]*>[\s\S]*?)";
        private readonly static String sqliRegex = @"(/\*!?|\*/|[';]--|--
[\s\r\n\v\f]|(?:--[^-]*?-)|([^\-&])#.*?[\s\r\n\v\f]|;?\\x00)";
        private Analyzer cookieAnalyzer;
        private CookieSettings cookieSettings;
        ServerManager serverManager;
        Configuration configuration;

        public CookieValidator() : base()
        {
            cookieAnalyzer = new Analyzer(cookieRegex, xssRegex, sqliRegex);
            serverManager = new ServerManager();
            configuration =
serverManager.GetWebConfiguration(System.Web.Hosting.HostingEnvironment.SiteName)
;
        }

        public override void Validate(HttpContext context)
        {
            RefreshSettings();

            if (cookieSettings.cookieValidation)
            {
                if (cookieSettings.cookieCustomRule)
                {

cookieAnalyzer.SetValidationRegex(cookieSettings.cookieCustomRuleStr);
                }
                else
                {
                    cookieAnalyzer.SetValidationRegex(cookieRegex);
                }

                HttpRequest request = context.Request;
                for (int i = 0; i < context.Response.Cookies.Count; i++)
                {
                    HttpCookie cookie = context.Response.Cookies.Get(i);
                    if (!String.IsNullOrEmpty(cookie.Value))
                    {
                        if (!cookieAnalyzer.Analyze(cookie.Value,
cookieSettings.cookieXssProtection, cookieSettings.cookieSqliProtection))
                        {
                            throw new HttpException(403, "Suspicious pattern
detected in cookie.");
                        }
                    }
                }
            }

            if(cookieSettings.cookieRemovePersistence)
            {
                RemoveCookiesPersistence(context);
```

```
            }
        }

        public void RemoveCookiesPersistence(HttpContext context)
        {
            for (int i = 0; i < context.Response.Cookies.Count; i++)
            {
                HttpCookie cookie = context.Response.Cookies.Get(i);
                if (!cookie.Expires.Equals(DateTime.MinValue))
                {
                    cookie.Expires = DateTime.MinValue;
                    context.Response.SetCookie(cookie);
                }
            }
        }

        private void RefreshSettings()
        {
            cookieSettings =
(CookieSettings)configuration.GetSection("system.webServer/cookieSecModule",
typeof(CookieSettings));
        }
    }
}
```

13.3.1.6 IPValidator

```
namespace IIS7Modules
{
    class IPValidator : Validator
    {
        HttpClient client;
        List<String> whitelist;
        Dictionary<string, int> triesPerIP;
        private IPSettings ipSettings;
        ServerManager serverManager;
        Configuration configuration;


        public IPValidator() : base()
        {
            client = new HttpClient();
            whitelist = new List<String>();
            triesPerIP = new Dictionary<string, int>();
            serverManager = new ServerManager();
            configuration =
serverManager.GetWebConfiguration(System.Web.Hosting.HostingEnvironment.SiteName)
;
        }

        public override void Validate(HttpContext context)
        {
            RefreshSettings();

            if (ipSettings.ipValidation)
            {
                string ip = context.Request.UserHostAddress;
                //string ip = "27.153.165.154"; //Blacklisted IP
                if (!whitelist.Contains(ip))
```

```
                {
                    IPQueryResult result = CheckIPAgainstAPI(ip);
                    if (result.attacks > 0)
                    {
                        if (ipSettings.ipSendMail)
                        {
                            if (!triesPerIP.ContainsKey(ip))
                            {
                                triesPerIP.Add(ip, 0);
                            }
                            triesPerIP[ip]++;
                            if (triesPerIP[ip] == ipSettings.ipMaxTries)
                            {
                                SendEmail(result);
                            }
                        }
                        throw new HttpException(403, "IP registrada en DNS BL");
                    }
                    else
                    {
                        whitelist.Add(ip);
                    }
                }
            }
        }

        private void SendEmail(IPQueryResult result)
        {
            MailMessage mail = new MailMessage("iissecmodule@gmail.com",
ipSettings.ipEmailAddress);
            SmtpClient client = new SmtpClient("smtp.gmail.com", 587);
            client.EnableSsl = true;
            client.DeliveryMethod = SmtpDeliveryMethod.Network;
            client.UseDefaultCredentials = false;
            client.Credentials = new NetworkCredential("iissecmodule@gmail.com",
"iisSecModule_1");
            mail.Subject = "Blacklisted IP report";
            mail.Body = "IP " + result.ip + ", with " + result.attacks + "
attacks of "
                + result.reports + " reports, has tried to access.";
            client.Send(mail);
        }

        private IPQueryResult CheckIPAgainstAPI(String ip)
        {
            string api_url =
"http://api.blocklist.de/api.php?ip={IP}&format=json".Replace("{IP}", ip);
            HttpResponseMessage response = client.GetAsync(api_url).Result;
            response.EnsureSuccessStatusCode();
            string content = response.Content.ReadAsStringAsync().Result;
            IPQueryResult result =
JsonConvert.DeserializeObject<IPQueryResult>(content);
            result.ip = ip;
            return result;
        }

        private void RefreshSettings()
        {
            ipSettings =
(IPSettings)configuration.GetSection("system.webServer/ipSecModule",
typeof(IPSettings));
        }
```

```
    }


    class IPQueryResult
    {
        public string ip { get; set; }
        public int attacks { get; set; }
        public int reports { get; set; }
    }
}
```

13.3.1.7 Analyzer

```
namespace IIS7Modules
{
    public class Analyzer
    {
        private Regex validationRegex;
        private Regex xssRegex;
        private Regex sqliRegex;

        public Analyzer(string validationStr, string xssStr, string sqliStr)
        {
            validationRegex = new Regex(validationStr, RegexOptions.IgnoreCase);
            if(!String.IsNullOrEmpty(xssStr))
            {
                xssRegex = new Regex(xssStr, RegexOptions.IgnoreCase);
            }
            if (!String.IsNullOrEmpty(sqliStr))
            {
                sqliRegex = new Regex(sqliStr, RegexOptions.IgnoreCase);
            }
        }

        public Boolean Analyze(String str, bool applyXss, bool applySqli)
        {
            bool ret = true;
            if (!validationRegex.Match(str).Success)
            {
                ret = false;
            }
            else if(applyXss && xssRegex.Match(str).Success)
            {
                ret = false;
            }
            else if(applySqli && sqliRegex.Match(str).Success)
            {
                ret = false;
            }
            return ret;
        }

        public void SetValidationRegex(String validationStr)
        {
            validationRegex = new Regex(validationStr, RegexOptions.IgnoreCase);
        }
    }
}
```

13.3.2 Clases de Configuración

13.3.2.1 URLSettings

```
namespace IIS7Modules.Settings
{
    class URLSettings : ConfigurationSection
    {
        public bool urlValidation
        {
            get { return (bool)base["urlValidationEnable"]; }
            set { base["urlValidationEnable"] = value; }
        }

        public bool urlCustomRule
        {
            get { return (bool)base["urlCustomRule"]; }
            set { base["urlCustomRule"] = value; }
        }

        public string urlCustomRuleStr
        {
            get { return (string)base["urlCustomRuleStr"]; }
            set { base["urlCustomRuleStr"] = value; }
        }

        public bool urlXssProtection
        {
            get { return (bool)base["urlXssProtection"]; }
            set { base["urlXssProtection"] = value; }
        }
    }
}
```

13.3.2.2 FormSettings

```
namespace IIS7Modules.Settings
{
    class FormSettings : ConfigurationSection
    {
        public bool formValidation
        {
            get { return (bool)base["formValidationEnable"]; }
            set { base["formValidationEnable"] = value; }
        }

        public bool formCustomRule
        {
            get { return (bool)base["formCustomRule"]; }
            set { base["formCustomRule"] = value; }
        }

        public string formCustomRuleStr
        {
            get { return (string)base["formCustomRuleStr"]; }
            set { base["formCustomRuleStr"] = value; }
        }

        public bool formXssProtection
        {
            get { return (bool)base["formXssProtection"]; }
            set { base["formXssProtection"] = value; }
        }

        public bool formSqliProtection
        {
            get { return (bool)base["formSqliProtection"]; }
            set { base["formSqliProtection"] = value; }
        }
    }
}
```

13.3.2.3 CookieSettings

```
namespace IIS7Modules.Settings
{
    class CookieSettings : ConfigurationSection
    {
        public bool cookieValidation
        {
            get { return (bool)base["cookieValidationEnable"]; }
            set { base["cookieValidationEnable"] = value; }
        }

        public bool cookieCustomRule
        {
            get { return (bool)base["cookieCustomRule"]; }
            set { base["cookieCustomRule"] = value; }
        }

        public string cookieCustomRuleStr
        {
            get { return (string)base["cookieCustomRuleStr"]; }
            set { base["cookieCustomRuleStr"] = value; }
        }

        public bool cookieXssProtection
        {
            get { return (bool)base["cookieXssProtection"]; }
            set { base["cookieXssProtection"] = value; }
        }

        public bool cookieSqliProtection
        {
            get { return (bool)base["cookieSqliProtection"]; }
            set { base["cookieSqliProtection"] = value; }
        }

        public bool cookieRemovePersistence
        {
            get { return (bool)base["cookieRemovePersistence"]; }
            set { base["cookieRemovePersistence"] = value; }
        }
    }
}
```

13.3.2.4 IPSettings

```
namespace IIS7Modules.Settings
{
    class IPSettings : ConfigurationSection
    {
        public bool ipValidation
        {
            get { return (bool)base["ipValidationEnable"]; }
            set { base["ipValidationEnable"] = value; }
        }

        public bool ipSendMail
        {
            get { return (bool)base["ipSendMail"]; }
            set { base["ipSendMail"] = value; }
        }

        public int ipMaxTries
        {
            get { return (int)base["ipMaxTries"]; }
            set { base["ipMaxTries"] = value; }
        }

        public string ipEmailAddress
        {
            get { return (string)base["ipEmailAddress"]; }
            set { base["ipEmailAddress"] = value; }
        }
    }
}
```

13.3.3 Clases de la Interfaz

13.3.3.1 UIModuleProvider

```
class UIModuleProvider : ModuleProvider
    {
        public override Type ServiceType
        {
            get { return null; }
        }

        public override bool SupportsScope(ManagementScope scope)
        {
            return true;
        }

        public override ModuleDefinition GetModuleDefinition(IManagementContext
context)
        {
            return new ModuleDefinition(Name,
typeof(UIModule).AssemblyQualifiedName);
        }
    }
```

13.3.3.2 UIModule

```
class UIModule : Module
    {
        protected override void Initialize(IServiceProvider serviceProvider,
Microsoft.Web.Management.Server.ModuleInfo moduleInfo)
        {
            base.Initialize(serviceProvider, moduleInfo);
            System.IO.Stream icoStream =
this.GetType().Assembly.GetManifestResourceStream("IIS7Modules.utilities_locker.p
ng");
            System.Drawing.Bitmap ico = new System.Drawing.Bitmap(icoStream);
            icoStream.Close();
            IControlPanel controlPanel =
(IControlPanel)GetService(typeof(IControlPanel));
            controlPanel.RegisterPage(new ModulePageInfo(this, typeof(AdminPage),
               "Seguridad HTTP", "Especifique aquí las opciones pertinentes para
la seguridad de las peticiones HTTP entrantes", ico, ico));
        }

        protected override bool IsPageEnabled(ModulePageInfo pageInfo)
        {
            Connection conn = (Connection)GetService(typeof(Connection));
            ConfigurationPathType pt = conn.ConfigurationPath.PathType;
            return pt == ConfigurationPathType.Site || pt ==
ConfigurationPathType.Application;
        }
    }
```

13.3.3.3 AdminPage

```
class AdminPage : ModulePage
    {
        private ServerManager serverMgr;
        private ModuleConfiguration c;

        public AdminPage()
        {
            serverMgr = new ServerManager();
            c = new ModuleConfiguration(serverMgr);

            Controls.Add(c);
        }


        protected override void OnActivated(bool initialActivation)
        {
            base.OnActivated(initialActivation);

            if (initialActivation)
                c.Initialize(Connection.ConfigurationPath.SiteName,
                             Connection.ConfigurationPath.ApplicationPath +
                             Connection.ConfigurationPath.FolderPath);
        }
    }
```