VulneraNET: Técnicas para la resolución cooperativa de vulnerabilidades

David del Pozo*, Alberto Pastor*, Francisco J. Blanco† and Ana M. Vázquez‡
*División I+D+i
Germinus XXI (Grupo Gesfor)
Avenida Manoteras, 32
28050 Madrid
{dpozog,apastorm}@grupogesfor.com
†Departamento de Ingeniería de Servicios Telemáticos
Universidad Politécnica de Madrid
Avenida Complutense nº 30
28040 Madrid
fcojavibr@dit.upm.es
‡Departamento de informática
Universidad Carlos III de Madrid
Avenida Universidad, 30
28911 Leganés
amvazque@inf.uc3m.es

***Resumen*—Este artículo presenta el proyecto VulneraNET, que propone mejorar la seguridad de las aplicaciones web proporcionando herramientas a los desarroladores que les faciliten (i) comprobar la seguridad de las aplicaciones desarrolladas, (ii) corregir las vulnerabilidades detectadas de forma cooperativa y (iii) compartir el conocimiento de de seguridad de aplicaciones web. Con este fin, la plataforma VulneraNET emplea técnicas web 2.0 y semánticas para facilitar la detección, gestión y resolución de vulnerabilidades, con el fin de reducir el esfuerzo y tiempo de corrección de vulnerabilidades.**

Las herramientas de VulneraNET pueden ser utilizadas a diferentes niveles, de acuerdo a las personas involucradas en el proceso de desarrollo y pruebas de un proyecto (desarrolladores, jefes de proyecto y auditores de seguridad) y sin necesidad de tener amplios conocimientos en el área de la seguridad.

***Palabras Clave*—seguridad, vulnerabilidad, wiki semántico, google wave, wapiti, lapse**

I. Introducción

La seguridad es uno de los principios más importantes en las aplicaciones web, ya que están expuestas a los ataques de cualquier usuario malintencionado. En caso de producirse un ataque con éxito contra la aplicación web, la empresa propietaria de la misma puede perder información o ésta puede ser sustraída, comprometiendo gravemente la información de la empresa y de sus clientes. Todo esto desencadenaría numerosas pérdidas económicas, a la vez que un grave desprestigio y perdida de confianza por parte de sus clientes y del sector empresarial.

Un estudio del Instituto Nacional de Estándares y Tecnología de Estados Unidos estima que los errores informáticos causan unas pérdidas de 59,5 billones de dólares anualmente [1]. No todos los fallos pueden ser detectados, pero el mismo estudio afirma que una infraestructura de pruebas que asegure la identificación efectiva y temprana de los defectos del software podría ahorrar casi un tercio de dichas perdidas.

La seguridad, pese a todo, es deliberadamente descuidada por muchos desarrolladores. Se piensa que realizar pruebas de seguridad es un esfuerzo innecesario en un proyecto, ya que el cliente final no puede ver los resultados de una forma tangible. Además, las pruebas de seguridad resultan tediosas, se deben tener grandes conocimientos sobre la materia y se pierde demasiado tiempo para los ajustados plazos a los que se suelen comprometer los fabricantes de software.

La mayoría de herramientas actuales están centradas en realizar pruebas específicas de algún aspecto de seguridad en concreto, requieren la interacción humana para realizar las pruebas y el usuario debe tener conocimientos tanto de la herramienta como de seguridad para llevar las pruebas a cabo.

Basándose en estas premisas, el proyecto VulneraNET [3] innova en la seguridad mediante la creación de una herramienta colaborativa que facilita la predicción, detección y corrección de vulnerabilidades de aplicaciones y servicios web, abarcando todo el ciclo de vida, desde su desarrollo hasta su auditoría y corrección de fallos. La herramienta será desarrollada y distribuida como Software Abierto. La Internet del futuro abre nuevos horizontes, pero también nuevas amenazas si no se disponen de medidas de seguridad adecuadas. Hoy en día se puede ya hablar ciber-terroristas, por lo que es necesario investigar en técnicas y herramientas para que los ciber-policías puedan detectar, predecir y corregir dichas vulnerabilidades en tiempos muy cortos. Precisamente éste es el objetivo del proyecto, orientado a la prevención de la seguridad de sistemas de clientes por auditores de seguridad, y a comunidades de software abierto que se auto-protegen en un entorno hostil y lleno de atacantes potenciales como Internet. Un estudio reciente de vulnerabilidades arrojaba algunas cifras inquietantes [2]:

- una vulnerabilidad de un navegador web es explotada después de un día de que sea conocida en el 94 % de las veces en 2008, frente a un 79 % de 2007.
- en vulnerabilidades publicadas para un sistema operativo, se explota en el mismo día en el 80 % de las

publicadas en 2008, frente a un 70 % de 2007.

El proyecto VulneraNET automatiza las pruebas de seguridad con el objetivo de reducir el esfuerzo y el tiempo de corrección de una vulnerabilidad, ofreciendo un panel de control que aglutine las vulnerabilidades encontradas por las diferentes herramientas de seguridad que se integran en VulneraNET y muestre los resultados de una forma clara para un desarrollador que no tenga conocimientos de seguridad, de forma que sea posible identificar dónde se encuentra la vulnerabilidad para poder tomar las medidas necesarias para solventarla.

Para centralizar toda esta información VulneraNET hace uso de Google Wave [4], pudiendo tener un control de las vulnerabilidades y realizar una gestión y seguimiento de las mismas. La plataforma Google Wave además permite su comunicación con otros servicios y aplicaciones externas. Para que sea posible la intercomunicación de las herramientas el proyecto define un formato de intercambio mediante el cual se pueden comunicar las diferentes herramientas de seguridad que están integradas en el proyecto.

Además se pretende innovar mediante el uso de tecnologías web 2.0 y semánticas en el ámbito de la seguridad para poder obtener procedimientos de corrección de vulnerabilidades que faciliten al desarrollador su tarea. Para que el uso de estas tecnologías sea posible, el proyecto plantea la construcción de un wiki de seguridad y una red social en la que los desarrolladores podrán aportar el conocimiento a la aplicación. La red social podrá ser privada (empleados de la misma empresa auditando o desarrollando una aplicación) o pública (empleada por voluntarios que contribuyen a la detección y predicción de vulnerabilidades en software abierto). De esta forma, se aprovecha el conocimiento de los expertos en seguridad para que éste sea compartido entre los demás desarrolladores y se realiza una retroalimentación de los procesos de solución propuestos por el sistema para ir perfeccionando las soluciones a las distintas vulnerabilidades. VulneraNET además investigará el uso de un modelo reputacional para evaluar y calificar los diferentes componentes existentes a lo largo del proceso de resolución de vulnerabilidades mediante información obtenida de la red social e Internet, para ello se investigará en técnicas de extracción de datos y minería de opiniones.

II. Arquitectura de VulneraNET

VuneraNET provee un entorno colaborativo para la detección y resolución de vulnerabilidades de seguridad. Con este propósito, la arquitectura de VulneraNET (ver Fig. 1) se ha diseñado como un conjunto de herramientas que se comunican mediante un formato de intercambio de vulnerabilidades. Actualmente, VulneraNET investiga en herramientas para la detección de vulnerabilidades, tanto de caja negra (sección V) como de caja blanca (sección VI), y en herramientas de gestión del conocimiento de resolución de vulnerabilidades, como un wiki semántico (sección III). La interacción entre las diferentes personas involucradas en las pruebas y corrección de vulnerabilidades, se realiza mediante la plataforma colaborativa Google Wave (sección IV), que permite la colaboración tanto en la notificación de vulnerabilidades, ejecución de pruebas, asignación social de vulnerabilidades pendientes y gestión del conocimiento corporativo.

El formato definido para el intercambio de información en VulneraNET posibilita la integración sencilla de nuevas herramientas en la plataforma. Este formato contiene información detallada sobre una vulnerabilidad como puede ser el tipo, fecha de detección, persona o aplicación que la encontró, posibles soluciones, gravedad, estado, etc. VulneraNET integra las diferentes herramientas a través de dicho formato, pudiendo generar a partir de él informes de vulnerabilidades y trasferir información a las demás herramientas de plataforma.

Dentro del proyecto VulneraNET se está investigando en los dos principales enfoques de detección de vulnerabilidades en aplicaciones web, las pruebas de caja blanca y las pruebas de caja negra, los estudios realizados sobre estas técnicas se están materializado en herramientas de seguridad que están siendo evolucionadas para la inclusión de las técnicas de detección estudiadas y que finalmente serán integradas en la plataforma (la herramienta Wapiti (sección V-A) en el caso de caja negra y LAPSE+ (sección VI-A) en el caso de caja blanca).

Por otra parte, VulneraNET también reúne diferentes herramientas de resolución de vulnerabilidades como la red social de profesionales de seguridad, el wiki semántico (sección III) que recoge información sobre vulnerabilidades, incluyendo procesos de resolución de vulnerabilidades y Google Wave (sección IV), que permite la comunicación de diferentes participantes en un proyecto para hacer más sencillo el proceso de resolución de vulnerabilidades. Google Wave además contiene el panel de control que permite ver de forma clara las vulnerabilidades encontradas en una aplicación web sobre la que se hagan las pruebas, generar informes y realizar el seguimiento de las vulnerabilidades. La integración con Google Wave para facilitar la cooperación entre usuarios se detalla en la sección IV-A.

Como se comentó anteriormente el proyecto está enfocado a las diferentes personas que trabajan en la seguridad informática, tanto los auditores de seguridad como los desarrolladores que se encargan de corregir los bugs. Las herramientas pueden ser utilizadas conjuntamente o por separado, con lo que hace más flexible su uso. Sin embargo, la ventaja diferencial de este proyecto radica en usar las herramientas en conjunto para la perfecta gestión de conocimiento de los diferentes profesionales que desarrollan las actividades de seguridad de una aplicación web.

III. Wiki semántico

Un objetivo principal del proyecto VulneraNET es la investigación de mejoras en la gestión del conocimiento de consultoría de seguridad mediante la aplicación de técnicas de etiquetado social para procesos y vulnerabilidades. Para llevar a cabo esta investigación se ha definido una herramienta colaborativa de resolución de incidencias, donde se tratará de validar el uso de redes sociales para facilitar la auditoría de seguridad de un equipo de auditores y mejorar la comunicación con el equipo de desarrollo [31]. La capacidad organizativa de los portales que implementan los conceptos wiki y wiki semántico, así como su facilidad de creación de comunidades muy activas y abiertas con una enorme capacidad de generación y edición de contenidos, unido a su gran impacto en la red de redes, los convierten en una aplicación idónea para el objeto de estudio [27].

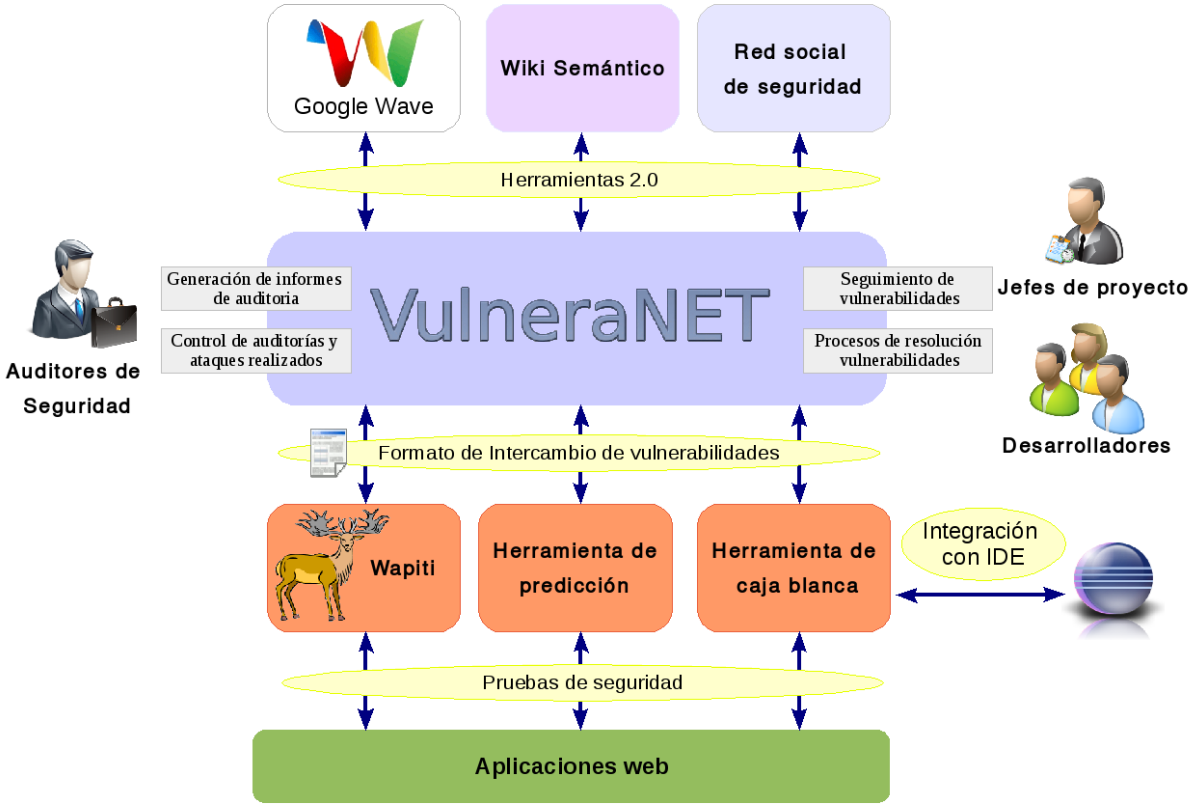

Figura 1. Visión general de VulneraNET

Así, wiki podría definirse como un concepto de aplicación Web cuyas páginas pueden ser editadas por múltiples voluntarios a través de un navegador, permitiendo a los diferentes usuarios de la aplicación crear, modificar o borrar información compartida. Cada página de un wiki se identifica por un título unívoco, al señalar dicho título con dos corchetes en cualquier lugar del wiki se crea automáticamente un enlace a dicha página, creando una estructura global de páginas que permite a los usuarios saltar de unas a otras a través de tales referencias [28]. Con estas sencillas reglas se construye un portal con gran capacidad organizativa, permitiendo a la comunidad compartir grandes cantidades de información, depurarla y enriquecerla continuamente. La principal utilidad de un wiki es que permite crear y mejorar las páginas de forma instantánea, dando gran libertad al usuario con una interfaz muy simple. Con un clic, un usuario puede empezar a editar una página y con otro clic, guardarla para que el resto de usuarios puedan cargarla de forma inmediata con los cambios realizados. Esto hace que un número considerable de usuarios participe en su edición, a diferencia de los sistemas tradicionales donde resulta más difícil que los usuarios del sitio contribuyan a mejorarlo.

El wiki software es un programa que permite ejecutar una aplicación Web basada en el concepto wiki. Existen diferentes versiones de software wiki, la mayoría de ellos son software libre y de código abierto. Es común que estos sean modulares, proporcionando APIs que permiten a los programadores desarrollar nuevas mejoras para la aplicación sin necesidad de familiarizarse con el código base. De los software wiki más populares se pueden destacar los siguientes Foswiki, MoinMoin, TikiWiki, XWiki, DokuWiki y MediaWiki, este último es el que se ha usado.

Existe una visión innovadora del concepto wiki que explota su capacidad organizativa por medio de conceptos de la Web semántica, llamado wiki semántico [29]. Un wiki semántico nace de la implementación de un modelo de conocimiento en el concepto wiki. Los wikis semánticos, a diferencia de las aplicaciones wiki, proporcionan la capacidad de formalizar la información acerca de los datos almacenados en sus páginas Web y establecer una jerarquía de relaciones entre estas, de forma que la información contenida en la aplicación puede ser exportada o consultada como una base de datos, aplicando el modelo de Entidad-Relación [22] [24].

Las tecnologías semánticas permiten proveer de una nueva dimensión de conocimiento a la información, posibilitando la integración y estandarización de conocimiento sobre las aplicaciones o recursos a aplicar la seguridad modelando el significado a los términos y conceptos necesarios [30]. Por lo tanto, esta nueva visión constituye una mejora notable a la hora de trabajar con la información contenida en la aplicación Web por medio de agentes software externos, facilitando la automatización del procesado de la información añadida al wiki y el intercambio de dicha información. La formalización de los términos incluidos en el wiki permite a estos agentes procesar la información de forma automática, evitando los tediosos procesos de parseado, minería e identificación del texto plano extraído de una página Web. Los usuarios que colaboran en un wiki suelen ser muy heterogéneos debido a las características del wiki de apertura y facilidad formándose grandes comunidades donde los contenidos aportados se van enriqueciendo debido a la diversidad de tales usuarios.

El problema radica en que tal heterogeneidad hace que

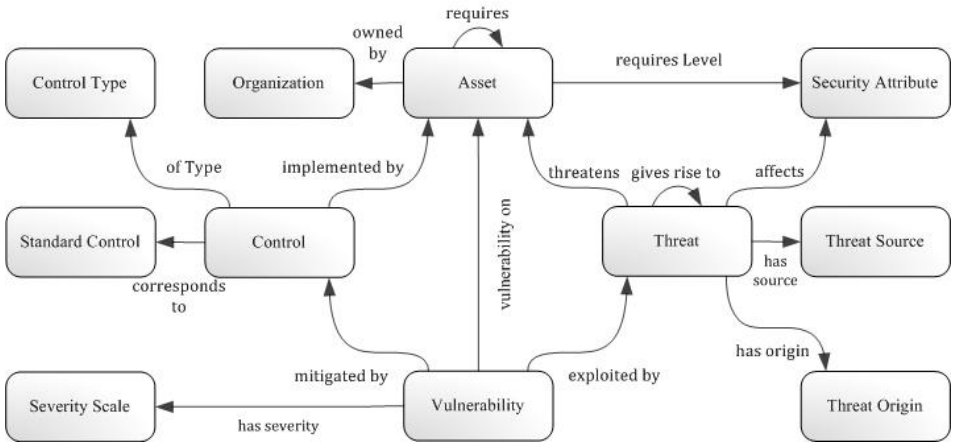

Figura 2. Ontología utilizada en VulneraNET

cada contenido difiera en demasía con los otros, incluso los más similares, ya que los usuarios no siguen ningún patrón, ni terminología, ni estructura en común. Esto es una de las cuestiones principales que intenta abordar la aplicación de wikis semánticos, añadiendo el uso de una estructura que formalice la creación y edición de contenidos en todo el wiki. El principal método por el que se genera una estructura formalizada es la inclusión de ontologías. En el área de la seguridad informática, el uso de ontologías de seguridad [26] [20] [23] provee de las siguientes capacidades:

- **Expresividad.** Una ontología y la cantidad de tecnologías subyacentes son capaces de describir cualquier clase de concepto y las relaciones entre ellos. Estos conceptos y sus propiedades quedarán jerarquizados formando una taxonomía y se les podrá asociar restricciones a éstos y a sus relaciones.
- **Organización.** Permite organizar el conocimiento, separa los requisitos de seguridad de su implementación técnica facilitando la gestión de la seguridad y el desarrollo y posterior acceso al modelo de seguridad ya que el sistema se realiza en dos fases: desarrollo del modelo sin tener en cuenta los elementos específicos y después la instanciación de dichos elementos. La ontología organiza y sistematiza cualquier tipo de fenómeno a cualquier nivel de detalle, reduciendo la lista de items a una lista de propiedades.
- **Mecanismos de formalización y compartición del conocimiento.** La falta de conocimiento explícito compartido en general en toda la ingeniería del software presenta una gran dificultad que repercute en gasto innecesario de tiempo y esfuerzo. La integración del conocimiento, su soporte automático y la posibilidad de su comunicación entre los agentes (humanos o software) disminuye la ambigüedad del lenguaje que frecuentemente lleva a errores, falta de entendimiento y realización de esfuerzos improductivos. Gracias a esta compartición y transmisión de la información se mitiga completamente la posible diferencia de conocimiento entre los miembros sobre el dominio del problema.
- **Inferencia lógica.** El procesamiento y análisis de las ontologías hace posible una capa lógica donde por ejemplo se pueden dibujar conclusiones o conseguir nueva información por la combinación de toda la información. La información implícita es hecha explícita gracias a la capacidad de las ontologías de razonar a través de un motor de inferencias.
- **Extensibilidad y reutilización.** Una ontología es siempre reutilizable pues se abstrae de los elementos específicos del sistema definiéndolos a través de conceptos y propiedades, esto supone escalabilidad pues con la misma ontología se pueden definir diez que cien elementos, lo único que se necesita es una instancia para cada uno. La idea es no empezar nunca de cero sino usar ontologías completas ya publicadas conteniendo la mayoría de los conceptos que se manejan en esa área específica. Una ontología es fácilmente modificable y ampliable de forma dinámica.
- **Interoperabilidad.** La capacidad que tienen las ontologías para especificar comportamientos sobre los objetos que representan los recursos y el manejo de conceptos comunes hace muy sencillo el intercambio de información entre entornos y aplicaciones heterogéneas. Gracias a la integración semántica y a la definición del comportamiento de los diferentes agentes mediante ontologías puede lograrse una capacidad de operar conjuntamente.

Concretamente, el empleo de ontologías en el proyecto VulneraNET está enfocado en la gestión de seguridad por medio de agentes inteligentes especificando el modo en que dichos agentes cooperan entre sí. La coordinación de estos agentes con la información especificada y compartida por parte de los diferentes usuarios que conforman el grupo de trabajo (auditores, analistas de seguridad, equipo de desarrollo, usuarios de la Web) lleva a emplear técnicas semánticas que permiten automatizar el proceso, disminuyendo la necesidad del componente humano en un proceso complicado y tedioso como es la monitorización de las incidencias de seguridad en tiempo real y el seguimiento de su resolución a lo largo de su ciclo para cada vulnerabilidad detectada. Además la utilización de ontologías permite la formalización del vocabulario de ataques, vulnerabilidades y demás componentes que toman parte en el seguimiento de incidencias en el campo de la seguridad de una aplicación Web, permitiendo también formalizar el proceso mismo del seguimiento de la incidencia

(ver figura 2).

La formalización de los términos anteriormente citados (Vulnerabilidades, amenazas, activos, procesos, etc.) permiten facilitar la gestión de la información de vulnerabilidades y amenazas que afectan a una aplicación Web dada. En este caso el uso de técnicas semánticas facilita a los agentes software la traducción de la información compartida por los usuarios de la aplicación, detectada por cualquiera de los módulos de detección de seguridad (análisis de código, análisis de trafico, etc.) y/o exportada a la base de datos desde un sitio externo a un formato de intercambio de vulnerabilidades de seguridad. La información puesta en este formato podrá ser entonces intercambiada entre los módulos del suit de seguridad o entre diferentes suites o programas de seguridad, así como entre diferentes aplicaciones Web dedicadas a la publicación de incidencias de seguridad.

Una última utilidad que cabe reseñar es el seguimiento en la edición de cada página y, por lo tanto, el seguimiento de cada concepto añadido al wiki. Así asociada a cada página existe otra página donde se puede discutir, criticar o realizar comentarios sobre los contenidos expuestos, las ediciones o actualizaciones realizadas, la adecuación o completitud de la información escrita o, por ejemplo, orientar sobre el contenido que queda por meter.

En la búsqueda de una herramienta colaborativa de resolución de incidencias esta perspectiva es idónea pues en las páginas de contenidos se tendrá toda la información de relevancia, con los datos técnicos, que se irán modificando, enriqueciendo y actualizando en un ambiente colaborativo a medida que los usuarios vayan accediendo y el proceso de resolución de la incidencia vaya avanzando. Al mismo tiempo en las páginas de discusión se realizarán los comentarios y críticas pertinentes a los contenidos, pudiendo especificar a los usuarios responsables que realizaron tales cambios o que serán los encargados de realizar las modificaciones. Para ello, la discusión será dividida en una lista organizada de entradas, cada una de las cuales expone una actividad a realizar sobre los contenidos y exponiendo las tareas hechas para llevar a cabo dicha actividad, especificando el usuario responsable de la tarea. Esta forma de gestionar los contenidos permite tener las dos áreas perfectamente separadas: la exposición del contenido técnico frente a las posibles discusiones, adecuándose ambas a un ambiente colaborativo.

IV. Google Wave

VulneraNET utiliza Google Wave [4] como herramienta colaborativa para la participación de los diferentes usuarios en la visualización de las vulnerabilidades de forma centralizada utilizando todas las ventajas que la plataforma provee.

Google Wave es una herramienta on-line de comunicación y colaboración en tiempo real. Cada conversación se conoce como *wave* u *ola* y se compone de diferentes mensajes que puede ser tanto una conversación como un documento, y en ellos los usuarios pueden participar y trabajar juntos usando texto formateado, fotos, vídeos, mapas, *widgets*, etc. Además, permite una participación activa por parte de los usuarios de la *ola*. Cualquier participante puede editar, responder o comentar algo en cualquier parte del mensaje, las respuestas se pueden publicar entre medias de otras, como respuesta concreta de una (creando subniveles de respuesta) o dentro de otra respuesta, además de poder privatizarla para elegir que usuarios pueden leerla. Google Wave permite a los usuarios acceder al historial de la ola, pudiendo ver de forma cronológica la evolución de la conversación.

Google Wave proporciona un API que permite utilizar y mejorar sus servicios a través de inserciones y extensiones. Las inserciones permiten introducir las olas de Google Wave en aplicaciones web externas. Las extensiones permiten mejorar las prestaciones y servicios que provee Google Wave mediante el uso de robots, que se comportan con un participante más de la ola y permiten automatizar ciertos comportamientos, y mediante el uso de *gadgets*, que están orientados a aumentar la experiencia de colaboración, permitiendo interactuar a los usuarios de nuevas formas a las que ya provee de forma básica Google Wave.

IV-A. Extensión de Google Wave para VulneraNET

En esta sección se mostrará un caso práctico de uso de VulneraNET. Dentro de el entorno de Google Wave se ha implementado una extensión en forma de robot [19]. Los robots dentro de Google Wave tienen el aspecto de un usuario normal y pueden participar en una conversación y modificar el contenido de la misma. Este robot permite al usuario obtener un informe con estadísticas e información que muestra de forma gráfica distintas vulnerabilidades, utilizando un fichero XML con vulnerabilidades generado por Wapiti como fichero de entrada.

Para poder ejecutar el robot en Google Wave lo primero que hay que hacer es añadirlo a la lista de contactos (su dirección es vulneranet@appspot.com). A continuación basta con incluirlo como participante en una *wave* para que se ejecute automáticamente, entonces aparecerá un formulario en el mensaje inicial de esa conversación con un campo de entrada de texto en el que indicar la URL del fichero con el informe de Wapiti en XML y una casilla de verificación que permite al usuario presentar el informe en un solo mensaje dentro del *wave* o en varios (modo múltiple, una vulnerabilidad por mensaje).

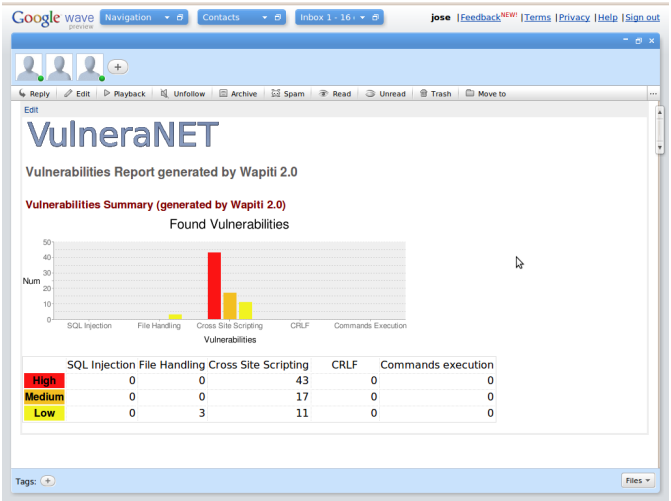

Figura 3. Informe dentro de Google Wave

En este ejemplo el informe aparece en el modo múltiple, con un mensaje introductorio y uno por cada riesgo que

incluye el fichero de entrada. El usuario podrá interactuar sobre cada vulnerabilidad usando directamente las opciones que Google Wave proporciona, por ejemplo, el usuario podrá: responder directamente sobre el mensaje en el que esté descrita esa vulnerabilidad para poner un comentario, crear una nueva wave con esa vulnerabilidad para poder interactuar de forma independiente al *wave* actual, editar la información que aparece sobre la vulnerabilidad, etc.

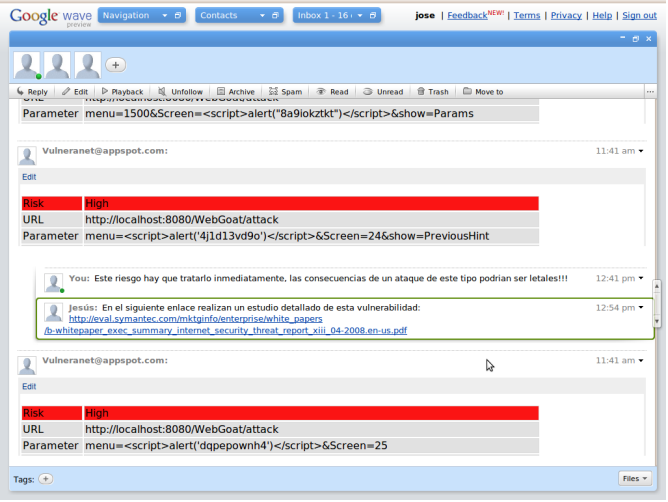

Figura 4. Usuarios interactuando con el informe

V. Enfoque de detección de vulnerabilidades de caja negra

Las técnicas de detección de vulnerabilidades de caja negra se basan en realizar ataques de la misma forma que un atacante real para poder descubrir las vulnerabilidades de una aplicación.

Existen diferentes técnicas basadas en este principio, el enfoque más utilizado, por ser el más eficiente, es el de la auditoría manual utilizando un *proxy* entre el servidor donde se encuentra la aplicación a evaluar y el navegador del auditor. Éste intercepta las peticiones y las respuestas del servidor y las manipula de forma manual con el fin de encontrar vulnerabilidades. Este enfoque permite descubrir todo tipo de vulnerabilidades, pero su éxito se basa en la pericia del auditor ya que para poder realizar este tipo de pruebas, son necesarios unos amplios conocimientos de seguridad [5].

Una de las herramientas más destacables que siguen este enfoque es WebScarab [6]. Esta herramienta desarrollada por la Fundación OWASP [8], permite realizar análisis de seguridad sobre aplicaciones que se comunican a través de los protocolos HTTP. Es una herramienta modular y puede ser extendida añadiendo *plugins*. Dispone de varios modos de operación, pero el más común es el de *proxy* entre la conexión y el navegador. Un auditor de seguridad puede revisar y modificar las respuestas desde el servidor hasta el navegador. Esta aplicación está orientada a personas que están familiarizadas con el protocolo HTTP y que tienen conocimientos amplios en el área de seguridad. La principal forma de trabajo con esta herramienta es prácticamente manual, requiere la intervención del usuario, aunque también dispone de funcionalidades automatizadas.

Otro enfoque muy utilizado es el *fuzzing*, técnica que consiste en inyectar una colección de cadenas maliciosas en las entradas de la aplicación que se desea probar, evaluando la respuesta obtenida. La gran ventaja de esta técnica es que puede detectar la mayoría de las vulnerabilidades relacionadas con inyecciones de código (XSS, inyecciones SQL, etc.), siendo éstas las más comunes según el Top 10 de vulnerabilidades de OWASP [7]. A diferencia de la anterior, esta técnica se puede automatizar. Lo normal es integrarla dentro de un proceso de auditoría manual puesto que evita realizar tareas muy repetitivas. Sin embargo, la evaluación de las respuestas de forma manual es mucho más exhaustiva que la automatizada, ya que ésta no es capaz de detectar muchas respuestas como vulnerables [5].

V-A. *OWASP Wapiti*

En este apartado se muestra la herramienta OWASP Wapiti,la cual se ha integrado dentro de VulneraNET. OWASP Wapiti [9] es un proyecto de código abierto liderado por Nicolas Surribas y Grupo Gesfor. La herramienta Wapiti utiliza técnicas de *fuzzing* para realizar los ataques y descubrir las vulnerabilidades. Wapiti no inyecta un grupo predefinido de cadenas maliciosas, sino cadenas generadas dinámicamente, de esta forma se evitan muchos falsos positivos. Permite auditar la seguridad de aplicaciones web de una forma automática y sencilla. La ventaja que tiene Wapiti respecto a otras herramientas del mismo estilo es que realiza el escaneo del sitio web sin la necesidad de la intervención del usuario. Las vulnerabilidades que es capaz de detectar son errores de manejo de ficheros, inyecciones de bases de datos, inyecciones XSS, inyecciones LDAP, inyecciones CRLF y de ejecución de comandos.

Wapiti puede ser fácilmente extendido para incluir nuevos tipos de ataque gracias a su arquitectura modular, cada ataque es implementado como un módulo independiente del resto. Wapiti también permite la configuración de *cadenas de texto maliciosas* que puede ser insertadas en los ataques existentes para expandirlos. Wapiti además genera informes con las vulnerabilidades encontradas que pueden ser exportados a diferentes formatos: HTML, XML y texto plano. Los informes están orientados a programadores y desarrolladores, un público que normalmente no tiene grandes conocimientos de seguridad, ya que proporcionan información sencilla y concisa sobre las vulnerabilidades para que éstas puedan ser resueltas.

OWASP (Open Web Application Security Project), una de las organizaciones sin animo de lucro más importante en el ámbito de la seguridad informática, ha reconocido la utilidad de esta herramienta incluyéndola como proyecto OWASP Alpha y distribuyendo Wapiti en la distribución GNU/Linux de seguridad OWASP Live CD [10]. Además, la herramienta OWASP Wapiti ha sido recientemente incluida en la distribución Backtrack [11]. BackTrack es una distribución GNU/Linux en formato LiveCD pensada y diseñada para la auditoría de seguridad y relacionada con la seguridad informática en general que goza de gran popularidad y aceptación.

Wapiti será evolucionado dentro de VulneraNET para la detección de errores de seguridad con AJAX y posteriormente integrado en VulneraNET gracias al formato de intercambio. La tecnología AJAX (Asynchronous JavaScript and XML),

está siendo usada cada vez más para crear aplicaciones web pues ofrece enormes beneficios de usabilidad para el usuario fin. Sin embargo, desde un punto de vista de seguridad estas aplicaciones tienen más problemas que una aplicación web convencional debido a que poseen una superficie de ataque mayor que las aplicaciones web normales (el número de entradas a la aplicación crece), tienen funciones internas expuestas como servicio y en ocasiones acceden desde la parte cliente a recursos sin mecanismos de codificación o seguridad.

VI. Enfoque de detección de vulnerabilidades de caja blanca

Las técnicas de detección de vulnerabilidades de caja blanca se basan en el análisis de código fuente. La aplicación de estas técnicas forma parte del desarrollo software donde los propios programadores someten a revisión el código que han generado para descubrir los problemas e inconsistencias que este contiene antes de su compilación. Aunque estas comprobaciones pueden ser realizadas por una persona, es habitual que esta tarea se automatice utilizando herramientas específicas que son capaces de analizar el programa completo. Las herramientas de análisis estático de código recorren el código fuente de un programa y automáticamente detectan errores y vulnerabilidades que no son advertidos normalmente por los compiladores y que posteriormente pueden suponer grandes problemas. Este tipo de procesos permiten resultados más exhaustivos que los realizados mediante pruebas de caja negra aunque ello también conlleva que, en proyectos de elevado tamaño, la dimensión del análisis puede suponer una labor ingente.

En la actualidad es posible encontrar diferentes propuestas orientadas al análisis estático. Algunas de ellas simplemente hacen un análisis semántico del código, comparando partes de éste con una biblioteca de vulnerabilidades conocidas como Flawfinder [12] o ITS4 [13]. Con diferencia, las propuestas más potentes para detección y localización de vulnerabilidades pasan por el análisis de flujo de datos. La técnica de análisis estático basado en el flujo de datos, construye un grafo de control del flujo que sirve para determinar las posibles propagaciones de datos manipulados de forma malintencionada hasta los puntos vulnerables del programa. Un ejemplo de este tipo de análisis es el propuesto por la Universidad Tecnológica de Viena denominado Pixy [14]. Una estrategia similar emplea la propuesta de la Universidad de Stanford, el llamado análisis de punteros (Points-to analysis) [15], en el que se obtienen los caminos por los que los datos manipulados pueden llegar hasta partes vulnerables del programa. Los esquemas basados en análisis de flujos y punteros pueden presentar la aparición de falsos positivos, sentencias de código identificadas por el proceso como potencialmente peligrosas, que en realidad no lo son. Por ello, son cada vez más habituales el empleo de técnicas sensibles al contexto que reducen el número de falsos positivos con respecto a otros esquemas.

VI-A. LAPSE+

LAPSE (Lightweight Analysis for Program Security in Eclipse) [16] es una herramienta para realizar la auditoría de seguridad en las aplicaciones Java y forma parte del proyecto *Griffin Software Security Project* [17] de la Universidad de Stanford. El analizador se distribuye como un plug-in de Eclipse [18] permitiendo mostrar, mediante vistas específicas, en qué parte del programa se da la vulnerabilidad de seguridad. LAPSE tiene como objetivo encontrar las vulnerabilidades de seguridad en las aplicaciones Web causadas por inadecuada o nula validación de las entradas del usuario. La implementación actual de LAPSE no constituye una de las herramientas más eficientes, los resultados documentados por sus desarrolladores muestran que algunos errores no se llegan a detectar y aparecen falsos positivos como resultado del análisis. Sin embargo, los autores de LAPSE proponen un análisis más completo, basado en punteros y al contexto como una futura mejora. Este análisis utiliza el lenguaje PQL para las vulnerabilidades en una sintaxis parecida a Java. Posteriormente las vulnerabilidades así definidas se traducen a un lenguaje lógico Datalog y se resuelven en una base de datos especifica. Este análisis a pesar de ser mucho más completo, es difícil de utilizar ya que no se integra bien en el entorno de desarrollo Eclipse.

Dadas las deficiencias presentes en el desarrollo LAPSE y basándose en la mejoras propuestas por los propios creadores se pretende la implementación de una nueva versión del plug-in de Eclipse, LAPSE+. Esta nueva herramienta tendrá en cuenta las vulnerabilidades en aplicaciones web cuyo origen se basa en la entrada de datos de usuario que no se valida de forma adecuada antes de ser utilizada por la aplicación, problema definido por el grupo SUIF como *tainted object propagation problem* (propagación de objetos envenenados). Como resultado, LAPSE+ constituirá una herramienta para la detección de las vulnerabilidades de seguridad en el código Java, teniendo en cuenta las vulnerabilidades más recientes en las aplicaciones web.

VII. Conclusiones

En este trabajo de investigación se ha presentado el proyecto VuneraNET, basado en herramientas y procesos colaborativos de detección, predicción y corrección de vulnerabilidades de aplicaciones web para desarrolladores y auditores de seguridad. Esta combinación e integración de herramientas de detección y resolución produce una sinergia que disminuye radicalmente el tiempo de corrección de vulnerabilidades.

VulneraNET permite aumentar la productividad, mediante la reducción del tiempo de detección y corrección de vulnerabilidades gracias a las herramientas de detección de vulnerabilidades y al entorno colaborativo de trabajo en el cual se enmarca este proyecto, que permite gestionar de forma eficiente el conocimiento de seguridad de todas la personas involucradas en el desarrollo y pruebas de un proyecto informático.

Agradecimientos

Este proyecto ha sido cofinanciado por el Ministerio de Industria, Turismo y Comercio, dentro de la convocatoria del subprograma Avanza I+D, dentro de la prioridad temática de Internet del futuro y como Proyecto Tractor de Investigación Industrial y Desarrollo Experimental en Cooperación (TSI-020302-2009-64).

Referencias

[1] Chen, H., Zou, T., and Wang, D. Data-flow based vulnerability analysis and java bytecode. In 7th Conference on 7th WSEAS international Conference on Applied Computer, 2007

[2] Internet Security Systems X-Force support. Disponible en http://www.usatoday.com/tech/news/computersecurity/2008-07-29-internet-threats-f aster_N.htm
[3] Sitio web de VulneraNET, http://vulneranet.grupogesfor.com
[4] Google Wave, http://wave.google.com/about.html
[5] Guia de test de OWASP v.3, http://www.owasp.org/images/5/56/OWASP_Testing_Guide_v3.pdf
[6] Sitio web del proyecto OWASP WebScarab, http://www.owasp.org/index.php/Category:OWASP_WebScarab_Project
[7] OWASP Top 10, http://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project
[8] Sitio del proyecto OWASP, http://owasp.org
[9] Sitio web de Wapiti, http://www.ict-romulus.eu/web/wapiti
[10] Distribucion GNU/Linux OWASP Live CD: http://www.owasp.org/index.php/Category:OWASP_Live_CD_Project
[11] Distribución Linux de seguridad Backtrack, http://www.backtrack-linux.org
[12] Sitio web de Flawfinder, http://www.dwheeler.com/flawfinder
[13] Sitio web de ITS4, http://www.cigital.com/its4/
[14] Jovanovic, N., Kruegel, C., and Kirda, E. Pixy: a static analysis tool for detecting Web application vulnerabilities. Security and Privacy, IEEE 2006
[15] Livshits, V. B. and Lam, M. S. Finding security vulnerabilities in java applications with static analysis. USENIX Security Symposium, 2005
[16] Sitio Web LAPSE, http://suif.stanford.edu/ livshits/work/lapse/
[17] Sitio Web Griffin Software Security Project, http://suif.stanford.edu/ livshits/work/griffin/
[18] SitioWeb Eclipse, http://www.eclipse.org/
[19] Robot de Google Wave de VulneraNET, http://vulneranet.grupogesfor.com/google-wave
[20] Blanco, C., Lasheras, J., Valencia-García, R., Fernández-Medina, E., Toval, A., & Piattini, M. (2007). Revisión sistemática y comparación de ontologías en el marco de la seguridad Paper presented at the IV Congreso Iberoamericano de Seguridad Informática (CIBSI 07), Mar Del Plata. Argentina.
[21] Eyal Oren. SemperWiki: a semantic personal Wiki. In Proceedings of the Workshop on Semantic Desktop. Workshop at 4th International Semantic Web Conference (ISWC 2005), Galway, Ireland, 2005.
[22] Eyal Oren, Max Völkel, John Breslin, Stefan Decker. Semantic Wikis for Personal Knowledge Management. In Proceedings of the 17th International Conference on Database and Expert Systems Applications Krakow, Poland, 2006
[23] Bill TSOUMAS, Dimitris GRITZALIS, "Towards an Ontology-based Security Management,.Advanced Information Networking and Applications, International Conference on, pp. 985-992, 20th International Conference on Advanced Information Networking and Applications - Volume 1 (AINA'06), 2006
[24] Max Völkel, Eyal Oren. Towards a Wiki Interchange Format (WIF) -Opening Semantic Wiki Content and Metadata. In Proceedings of the Workshop on SemWiki. Workshop at ESWC, 2006.
[25] Artem Vorobiev and Jun Han, "Security Attack Ontology for Web Services", in Proceedings of the Second International Conference on Semantics, Knowledge, and Grid (SKG'06), IEEE Computer Society, 2006.
[26] J. Undercoffer, A. Joshi and J. Pinkston. Modeling Computer Attacks: An Ontology for Intrusion Detection. University of Maryland, Baltimore County. Department of Computer Science and Electrical Engineering, 2003.
[27] Buffa, M., Gandon, F., Ereteo, G., Sander, P., and Faron, C: SweetWiki: A semantic wiki. Web Semant., vol 6, no. 3, pages 84-97, year 2008.
[28] Ye, Shiren and Chua, Tat-Seng and Lu, Jie: Summarizing definition from Wikipedia. ACL-IJCNLP '09: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP, vol 1, pages 199-207, year 2009.
[29] Schaffert, S., Bry, F., Baumeister, J. and Kiesel, M.: Semantic Wikis. IEEE Softw., vol. 25, no. 4, pages 8-11, year 2008.
[30] Panagiotou, D. and Mentzas, G.: Exploiting Semantics in Collaborative Software Development Tasks. Proceeding of the 2008 conference on Knowledge-Based Software Engineering, pages 385-394, year 2008.
[31] Xiao, WenPeng and Chi, ChangYan and Yang, Min: On-line collaborative software development via wiki. WikiSym '07: Proceedings of the 2007 international symposium on Wikis, pages 177-183, year 2007.