



Escuela
Politécnica
Superior

Herramienta de auto diagnóstico



Máster Universitario en Ciberseguridad

Trabajo Fin de Grado

Autor:
Manuel Torres Mendoza

Tutor/es:
José Vicente Berná Martínez

Junio 2020



Universitat d'Alacant
Universidad de Alicante

TFM-Manuel Torres Mendoza

Motivación, justificación y objetivo general

Cada día las amenazas de seguridad son más potentes y complejas, por ello, a la par van surgiendo herramientas que nos permiten defender nuestros sistemas de manera más automatizada y eficiente. Atrás quedaron los días en los que una simple autenticación por medio de usuario y contraseña atribuía a nuestro sistema el adjetivo “seguro”.

Una de las múltiples herramientas imprescindibles para un profesional de ciberseguridad es el **IDS** (*Intrusion Detection System*), el cual monitorizará nuestros sistemas en busca de comportamientos sospechosos que puedan dar lugar a ciberataques. También existen herramientas que toman medidas contra estos comportamientos además de informar, por ejemplo, el **IPS** (*Intrusion Prevention System*). Lo cierto es que estas herramientas se centran en la monitorización de servicios de alto nivel, como pueden ser servidores web, correo, bases de datos, etc. dejando comportamientos mucho más comunes sin revisión alguna.

Pese a que estos servicios de alto nivel son muy importantes para la empresa, siendo atacados en primer lugar en la mayoría de ciberataques, son los ataques dirigidos a las acciones del día a día los que mayor porcentaje de éxito consiguen. Esto es debido a la popularización de los ataques basados en Ingeniería social.

Por sencillo que parezca, que un empleado abra un documento de facturación o conecte un dispositivo externo a su estación de trabajo es muy común. Este tipo de prácticas, que por sí mismas no tienen por qué perseguir ningún objetivo maligno o poco honesto, son uno de los mayores vectores de ataque para los ciberdelincuentes. Lo peor de todo es que las herramientas potentes nombradas con anterioridad no tienen el objetivo de detectar comportamientos tan aislados de los servicios críticos.

Aquí es donde surge la motivación de este documento, por lo que, a través de un análisis detallado del estado de las tecnologías actuales, se tratará de solventar la problemática anterior. Todo esto, intentando integrar los paradigmas tecnológicos actuales en una aplicación que represente una herramienta de diagnóstico, a nivel general, sin olvidarnos de los servicios críticos, para los encargados de ciberseguridad en un contexto empresarial.

Aunque, en un primer momento, la idea de este documento surgió del deseo de implementar una herramienta de diagnóstico enfocada a servidores y a los servicios críticos de estos, mi tutor me propuso centrarlo en acciones más cotidianas. Lo cierto es que al investigarlo me di cuenta de que había una total necesidad de este tipo de herramientas en el entorno tecnológico de muchas organizaciones, ya que estas no tenían control alguno, a diferencia de las herramientas de diagnóstico centradas en servicios y servidores.

TFM-Manuel Torres Mendoza

Índice de contenidos

Motivación, justificación y objetivo general	2
Índice de figuras	6
Índice de tablas	7
1. Introducción	8
2. Estudio de viabilidad	11
2.2. Análisis de mercado. Lean Canvas	12
2.3. Análisis de riesgos	13
3. Planificación	14
4. Estado del arte.	15
5. Objetivos	28
6. Metodología	30
7. Análisis y especificación	34
8. Diseño.....	43
9. Implementación	58
10. Pruebas y validación.....	73
11. Resultados	85
11.1. Cumplimiento de objetivos	86
11.2. Cumplimiento de la planificación.....	90
11.3. Vinculación con asignaturas.....	91
12. Conclusiones y trabajo futuro	92
12.1. Alcance logrado.....	92
12.2. Trabajo futuro	93
12.3. Aportación a mi desarrollo profesional.....	93
Referencias.....	94
Anexo 1.....	97
Anexo 2.....	99

Anexo 3.....	100
Anexo 4.....	102
Anexo 5.....	103
Anexo 6.....	106
Anexo 7.....	108
Anexo 8.....	110

TFM-Manuel Torres Mendoza

Índice de figuras

Figura 1. Captura EventLog Manager en plena ejecución.....	22
Figura 2. Captura de vista local de equipo con herramienta EventSentry SIEM.....	24
Figura 3. Captura de los incidentes de seguridad de un usuario con la herramienta Netwrix User Behavior Analytics.....	25
Figura 4. Fases de ejecución de un proyecto siguiendo la metodología Waterfall.	30
Figura 5. Diseño conceptual del software mediante la herramienta Draw.io.	32
Figura 6. Organización temporal de las tareas de implementación. Fuente:	33
Figura 7. Diferentes agentes que intervienen en el funcionamiento de la herramienta.....	46
Figura 8. Comportamiento de la herramienta local haciendo uso de diversas tecnologías.....	49
Figura 9. Comportamiento del agente externo haciendo uso de diversas tecnologías.....	51
Figura 10. Diagrama Entidad Relación para la base de datos del agente externo.....	52
Figura 11. Diagrama Entidad Relación para la base de datos de la herramienta local.	53
Figura 12. Ventana de servicio a través de Services.msc de Windows.....	59
Figura 13. Datos del Windows Registry en relación con el servicio Firewall.	63
Figura 14. Información del registro de eventos de Windows, concretamente, para Sysmon....	65
Figura 15. Petición a la API del Agente Externo para obtener el informe.	71
Figura 16. Alerta recibida en el correo del administrador e informe generado por la herramienta.....	75
Figura 17. Informe generado por la herramienta y alerta visualizada en el equipo local	78
Figura 18. Generación del informe por la herramienta y alerta recibida en el correo del administrador.....	80
Figura 19. Informe generado por el Agente Externo enviado al correo del administrador.....	82
Figura 20. Alerta recibida en el correo del administrador	84
Figura 21. Interfaz de configuración de la herramienta.....	85
Figura 22. Número de incidentes por día por usuario (barras) y tendencia de estos (líneas)....	87

Índice de tablas

Tabla 1. Análisis DAFO para el proyecto. Elaboración propia.	11
Tabla 2. Análisis Lean Canvas para el proyecto. Elaboración propia.	12
Tabla 3. Análisis de riesgos en lo que respecta a la realización del proyecto.....	13
Tabla 4. Planificación temporal del proyecto.....	14
Tabla 5. Resumen de las características que ofrece el software analizado en este apartado....	26
Tabla 6. Categorías de eventos interesantes para la detección de posibles riesgos de seguridad.	61
Tabla 7. Configuración inicial de la herramienta.....	73
Tabla 8. Actividades realizadas en la máquina y su resultado esperado.	74
Tabla 9. Resultados obtenidos tras la ejecución de las actividades.....	75
Tabla 10. Configuración inicial de la herramienta. Fuente: Elaboración propia.	76
Tabla 11. Actividades realizadas en la máquina y su resultado esperado.	77
Tabla 12. Resultados obtenidos tras la ejecución de las actividades.....	77
Tabla 13. Configuración inicial de la herramienta.....	78
Tabla 14. Actividades realizadas en la máquina y su resultado esperado.	79
Tabla 15. Resultados obtenidos tras la ejecución de las actividades.....	79
Tabla 16. Servicios probados a través del agente externo y resultado esperado.	80
Tabla 17. Resultado obtenido tras la ejecución de los análisis a los servicios públicos por parte del agente externo	81
Tabla 18. Configuración del horario de actividad para la Organización A	82
Tabla 19. Actividades realizadas y resultado esperado.	83
Tabla 20. Resultados obtenidos tras la ejecución de las actividades.....	84
Tabla 21. Comparativa de la planificación esperada con la resultante.....	90

1. Introducción

En los últimos años, la ciberseguridad ha tomado un papel importante para las organizaciones, en muchos casos, debido al crecimiento de los ciberataques. Las consecuencias son variadas, ya que pueden ir desde la pérdida de recursos temporales, monetarios o humanos, hasta la desaparición de la organización. Por ello, se ha enfatizado la protección de los sistemas que pueden sufrir este tipo de ataques, de forma que se cumplan las propiedades de un sistema seguro: Disponibilidad, Integridad, Confidencialidad, Autenticidad y Trazabilidad.

El cumplimiento de estas propiedades se lleva a cabo a través de distintas herramientas que se instalan, tanto en los sistemas que ejecutan servicios críticos, como en los equipos comunes utilizados por el personal. En realidad, hay una gran variedad en cuanto a las herramientas que se instalan en los sistemas críticos, además, son bastante más potentes que las utilizadas en los equipos comunes. Por ejemplo, podemos destacar sistemas de detección de intrusiones (IDS), políticas de control de acceso como SELinux o Apparmor, monitorización continua de la mano de herramientas como Nagios, Munin, etc. En cambio, los equipos comunes, con suerte, cuentan con un antivirus y un firewall, normalmente con la configuración básica, la cual no les servirá ante un ataque dirigido o moderno.

En principio, esta priorización de la seguridad en los sistemas críticos tiene sentido, ya que de estos depende la operatividad de la organización, en cambio, el bienestar de un equipo común con el que opera un empleado no es realmente determinante. Lo cierto es que la presencia de malware en uno de estos equipos comunes puede poner en riesgo la gran mayoría de los sistemas de una organización, incluso de otras, siempre y cuando la política de seguridad de la organización no presente medidas de protección para estos casos. Esto es debido a que el malware actual, en muchas ocasiones, va saltando de equipo en equipo, consiguiendo credenciales por el camino para alcanzar sistemas críticos [1].

Por lo tanto, podemos identificar a los equipos comunes de los usuarios como un buen punto de entrada para ciberataques, debido a su poca protección y aislamiento del resto de los sistemas, de hecho, se considera la más común por la gran mayoría de expertos en ciberseguridad. Pero, además, existe otro punto a favor de la utilización de este vector de ataque, el propio usuario del sistema, ya que estos pueden no contar con una formación extensa en el área de la ciberseguridad, por lo que será más fácil para un atacante tratar de usar técnicas de Ingeniería Social contra ellos, que infiltrarse en un sistema de forma remota atacando una vulnerabilidad cualquiera. Los ejemplos más típicos ya los conocemos, el empleado que inserta un dispositivo externo desconocido e infectado en la máquina de trabajo, el empleado que visita el correo personal en el trabajo y pincha en un enlace de un mensaje phishing que le dice que su cuenta bancaria ha sido suspendida, etc.

La formación y concienciación de los empleados es un área que se ha ido mejorando en los últimos años por parte de las organizaciones, llevando a cabo acciones que intentan implicar al empleado en los temas de ciberseguridad, además de hacerle entender los riesgos implícitos de sus acciones. Por otro lado, también se lleva a cabo la creación de documentos normativos como la Política de Seguridad, la cual refleja una serie de reglas, en fomento de la ciberseguridad, que los empleados deben de cumplir en el desarrollo de sus tareas cotidianas.

Lo cierto es que la política de seguridad es un documento muy beneficioso, ya que define una serie de buenas prácticas para llevar a cabo las tareas que realizan los empleados. Además, esta puede ser única para cada organización, ya que el contexto de cada una puede determinar que una acción de seguridad tenga o no lugar. Por el contrario, estos documentos tienen varios puntos en contra, dejando de lado que pueden tener una gran extensión o complejidad técnica para la que un empleado normal no está capacitado, reflejan medidas generalistas. Estas medidas son muy difíciles de monitorizar para las herramientas de seguridad, de hecho, en muchas ocasiones no tendrá sentido, ya que, como hemos comentado, según el contexto de la organización pueden ser importantes o no.

Por lo tanto, este TFM tendrá el objetivo de investigar el desarrollo de una herramienta que permita la monitorización del cumplimiento de las normas de carácter generalista y cotidiano, que así mismas pueden formar parte de las políticas de seguridad de las organizaciones. Además, también se implementará una gran parte de esta para ponerla en práctica en entornos principalmente empresariales. Por otro lado, resulta evidente que la herramienta deberá tener en cuenta los servicios que ejecute la máquina, aunque este no sea su objetivo principal, debe de ser capaz de detectar los posibles puntos de entrada de un sistema. Del mismo modo, ante una incidencia de seguridad, la herramienta debería de poder realizar una serie de medidas preventivas para que esta no suponga un riesgo para toda la organización. Por ejemplo, si un usuario inserta un dispositivo externo en la máquina y se ha definido que se mande un aviso y se apague la máquina, esto podría evitar la expansión del malware en el resto de la organización.

Por último, también se tendrá en cuenta que la herramienta debe de facilitar la recopilación de información en informes que puedan ser fácilmente interpretados por los expertos en ciberseguridad.

2. Estudio de viabilidad

En esta sección del documento se va a justificar el grado de idoneidad que tiene el proyecto desde varios puntos de vista, utilizando la siguiente metodología:

1. Análisis introspectivo. Nos proporcionará una visión global del proyecto, tanto de los puntos a favor y contra de este como nuestra situación para llevarlo a cabo.
2. Análisis de mercado. Nos proporcionará una visión general de la idea de negocio en la que se fundamenta el proyecto y cómo de viable sería si se fuese a publicar de manera abierta.
3. Análisis de riesgos. Identificaremos los riesgos más posibles de nuestro proyecto y la manera de lidiar con ellos.

2.1. Análisis introspectivo. DAFO

FACTORES INTERNOS DEL PROYECTO		FACTORES EXTERNOS DEL PROYECTO	
DEBILIDADES (-)		AMENAZAS (-)	
1	Poca experiencia de desarrollo en el área	1	Variabilidad de normas según la organización
2	Incapacidad de financiación	2	
3	Tiempo reducido debido a formación	3	
4	Falta de conocimiento en entornos Microsoft	4	
5	Integración de tecnologías compleja	5	
6	Dependiente del sistema operativo	6	
FORTALEZAS (+)		OPORTUNIDADES (+)	
1	Buenas capacidades técnicas	1	No hay competidores directos
2	Buena actitud para el desarrollo del proyecto	2	Aumento del interés por los registros de eventos por parte de las organizaciones (P.ej. SOC)
3	La herramienta proporciona un mecanismo de trazabilidad	3	
4	La herramienta puede evitar un ataque	4	
5	La infraestructura que se utiliza es mínima	5	

Tabla 1. Análisis DAFO para el proyecto. Fuente: Elaboración propia.

2.2. Análisis de mercado. Lean Canvas

Problemas:	Solución:	Proposición de valor única:	Ventaja diferencial:	Segmento de clientes:
<p>Dificultad para controlar acciones cotidianas por parte de los empleados.</p> <p>Dificultad para definir una serie de acciones genéricas que se consideren un incidente de seguridad según la organización.</p> <p>Dificultad para la recolección de estos incidentes de seguridad genéricos y cotidianos en informes.</p>	<p>Detección de incidentes de seguridad.</p> <p>Posibilidad de respuesta predefinida ante incidentes.</p> <p>Posibilidad de personalización de los tipos de eventos que se identificarán como incidentes de seguridad.</p> <p>Generación de informes de fácil entendimiento.</p>	<p>Servicio automático que recolecte los eventos del sistema e identifique incidentes de seguridad de forma ajena al usuario. Todo esto, en función de las características definidas previamente por el usuario.</p>	<p>Herramienta que detecta incidentes de seguridad generalistas causados, en muchas ocasiones, por el propio usuario del sistema. Ante este tipo de incidencias, se podrán tomar medidas de forma automática.</p>	<p>Organizaciones, tanto públicas como privadas, interesadas en el control de la política de seguridad por parte de los empleados.</p>
	<p>Métricas clave:</p> <p>Número de incidentes capturados y no capturados a través del feedback posterior con los clientes.</p>		<p>Canales:</p> <p>Contacto con empresas dedicadas a la consultoría de ciberseguridad para la recomendación.</p> <p>Redes sociales y página web.</p>	
<p>Estructura de costes:</p> <p>Servidor, tiempo de desarrollo y mantenimiento.</p>			<p>Flujo de ingresos:</p> <p>Donaciones</p>	

Tabla 2. Análisis Lean Canvas para el proyecto. Fuente: Elaboración propia.

2.3. Análisis de riesgos

Tipo de riesgo	Riesgo	Estrategia
Tecnológico	Equipo con entorno Windows queda inutilizable	<p>Prevención: Monitorización del estado del hardware y registro de malos funcionamientos.</p> <p>Plan de contingencia: Se instalará una máquina virtual Windows 10 en otro equipo.</p>
Personal	Enfermedad	<p>Prevención: Buena dieta y abrigarse bien.</p>
Tecnológico	Cambio en dependencia	<p>Prevención: Mientras que no pertenezca al sistema operativo, se utilizarán las mismas versiones.</p> <p>Minimización: Estar al tanto de los posibles cambios en las dependencias para hacer el sistema compatible a la vez que se va desarrollando.</p>
Tecnológico	Desconocimiento de la tecnología	<p>Prevención: Aprender a través de recursos online, preferiblemente documentación oficial.</p> <p>Plan de contingencia: Pedir consejo en foros de tecnología disponibles en Internet.</p>
Tecnológico	Pérdida del software	<p>Prevención: Realización y prueba de copias de seguridad, además del uso de un repositorio de control de versiones.</p>
Personal	Falta de tiempo debido a formación paralela	<p>Prevención: Intentar dedicar 1-2 horas al día.</p> <p>Plan de contingencia: Diseñar la solución completa, pero recortar implementación.</p>

Tabla 3. Análisis de riesgos en lo que respecta a la realización del proyecto. Fuente: Elaboración propia.

3. Planificación

En esta sección del documento se determinan una serie de intervalos de tiempo en los que se deberían de realizar las partes del TFM. De esta manera, se intentarán optimizar los recursos temporales con los que se parten.

Contenidos	Tiempo total	Fecha límite fin
Motivación, justificación y objetivo general Introducción Estudio de viabilidad	15 días	Mediados de febrero
Estado del arte Objetivos	15 días	Mediados de marzo
Metodología Análisis y especificación Diseño	7 días	Finales de marzo
Implementación Pruebas y validación	1 mes	Finales de abril
Conclusiones y trabajos futuros	15 días	Mediados de mayo

Tabla 4. Planificación temporal del proyecto. Fuente: Elaboración propia.

4. Estado del arte.

En esta sección del documento se detalla el problema a solventar, ratificado con datos y estudios relacionados con el tema. También, se destacan las soluciones que existen actualmente en el mercado para intentar solventar el problema de forma parcial o total. Una vez conocida la problemática, se propondrá una posible solución sin entrar en las especificaciones tecnológicas de esta, dejando este tema para otro capítulo.

4.1. Estudio de la problemática

Hoy en día, la gran mayoría de las empresas, se encuentren o no centradas en el marco tecnológico, tienen una gran variedad de dispositivos electrónicos que pueden interactuar con la red. Ya no se habla solo de los equipos de sobremesa asociados al trabajo de los empleados, sino de portátiles, impresoras o pequeños ordenadores de una sola placa (P.ej. Raspberries), además del mercado emergente **IoT** (*Internet of Things*), representado por dispositivos con hardware embebido que tienen capacidad para conectarse a la red. Por ejemplo, es muy común que haya cafeteras, frigoríficos o microondas en los comedores de las empresas, es muy probable que estos se encuentren conectados a la red empresarial en los siguientes años, si no lo están ya. De hecho, la tendencia de este mercado se espera que se mantenga en aumento durante los próximos años [2].

Esto nos lleva al primer problema, ya que, tradicionalmente, los expertos en seguridad únicamente han tenido que ocuparse de mantener bien configurados y seguros los activos TI más críticos para el funcionamiento del negocio, por ejemplo, servidores y computadores pertenecientes a la infraestructura de la compañía. En definitiva, dispositivos que podían representar un vector de ataque para ciberdelincuentes.

En cambio, en el panorama tecnológico actual, hay múltiples dispositivos, como los citados anteriormente basados en IoT, que pueden suponer una entrada para un atacante, tanto interno como externo. Además del **gran número de nuevos dispositivos**, la **variabilidad** del software y protocolo de comunicación de estos hace **muy difícil** la **gestión manual** por parte **del experto**.

Es evidente que esta variedad de tecnologías no solo implica que se deben de conocer para su configuración y gestión de incidencias, sino que se tiene que estar al tanto de sus probables vulnerabilidades, las cuales, sin lugar a duda surgirán en un momento específico. Esto nos abre dos frentes:

- Por un lado, necesitamos configurar de forma óptima los sistemas que constituyen nuestra infraestructura TI, no solo para que se preste el servicio deseado, sino para no habilitar brechas de seguridad. Según IBM X-force Threat Intelligence Index de 2019, los incidentes relacionados con malas configuraciones de los sistemas ocurren un 20% más cada año [3].
- Por otro lado, hay que estar continuamente al tanto de las vulnerabilidades que van surgiendo en los sistemas que conforman nuestra infraestructura TI, ya sea para buscar una solución alternativa o actualizar a una versión que no cuente con la vulnerabilidad. Esto debe de ser realmente importante para las organizaciones, sin ir más lejos, el equipo IBM X-Force Red's Vulnerability Management Services afirma encontrar un promedio de 1.440 vulnerabilidades **únicas** por organización [3].

Lo cierto es que la gestión de las vulnerabilidades actuales y futuras de los activos TI de la organización siempre han sido una tarea importante del experto en ciberseguridad. Esto es debido a que se van actualizando protocolos y funcionalidades internas de los servicios que proporcionan los sistemas TI de manera constante. Por ello, la infraestructura TI necesita una **monitorización continua** que nos ofrezca informes detallados de las incidencias de seguridad en nuestro entorno. Esto es algo que, en el contexto de sistemas TI, se implementa a través de software de monitorización (P.ej. Nagios, Munin, etc.), pero lo cierto es que **no hay un software de monitorización de acciones triviales** a nivel del equipo de usuario adscrito a la organización. Es decir, podemos consultar la carga de nuestros servidores, pero no si un empleado ha enchufado su teléfono móvil, posiblemente infectado con malware, a su equipo de trabajo [4].

El riesgo que conllevan este tipo de acciones debería de ser conocido por los empleados de una organización, nada más lejos de la realidad. A menudo, los **empleados** se muestran **reticentes a dejar de hacer acciones inseguras** que, según ellos, eran más eficientes en favor de prácticas que disminuyen en gran medida el riesgo de que ocurra un incidente de seguridad. Por suerte, a lo largo de los últimos años se ha entendido la importancia del área de formación y concienciación de todos los miembros de la organización, debido, en gran medida, a los múltiples ciberataques ocasionados por acciones inconscientes de los empleados. De hecho, en 2015 ya se decía que el 23.5% de los ataques fueron lanzados por estas acciones inintencionadas [5].

Bien es sabido que los cambios en la metodología de trabajo son más sencillos si se hace entender a las personas el porqué de estos, pero, aun así, se necesitan documentos formales y oficiales que reflejen las políticas de la organización en esta área, de tal forma que se obligue de manera normativa a la ejecución de estas por parte de los empleados.

En multitud de compañías se realiza un control y registro de las navegaciones web ilícitas por parte de los usuarios, mediante, por ejemplo, servidores proxy. En cambio, **no encontramos la aplicación de este tipo de medidas correctivas** en cuanto a las **directivas y normas** que marca la **política de seguridad** de una compañía. De tal forma que un encargado de ciberseguridad pueda consultar las violaciones que se producen sobre la política de seguridad de forma rápida mediante informes generados automáticamente, aplicando las medidas correctivas que procedan. De aquí se puede inferir que este es un motivo de peso para que se haga tanto hincapié en las acciones de formación y concienciación sobre ciberseguridad, creando planes dirigidos, formación asociada, continua, etc. para los usuarios. De esta manera, se intenta que los usuarios no realicen este tipo de acciones proporcionándoles conocimiento sobre los riesgos implícitos de estas, ya que la organización no puede controlarlos, o al menos, les resulta demasiado costoso.

En gran medida, esta problemática es debida a la **heterogeneidad del contexto que rodea a la infraestructura TI** de una organización. Por ejemplo, la política de seguridad no va a ser la misma en una central nuclear que en un concesionario de coches, pudiendo ser acciones críticas del primer caso, normales o “asequibles” en el segundo.

Si indagamos en la **gestión de la seguridad** que tienen las **organizaciones**, veremos que, en su gran mayoría, las que tienen personal técnico cualificado implementan una **gran protección** en sus **servicios**, en multitud de ocasiones a través de la integración de **mecanismos de seguridad adicionales**. Hoy en día, lo normal es que se realice una securización intensiva para cada capa de la pila del protocolo tecnológico usado, a grandes rasgos, en el nivel de red podemos encontrar Firewalls, ACLs, dispositivos exclusivamente de seguridad como ASAs, seguridad en switches, etc. Mientras que, a nivel de aplicación, se tiene software cuyo objetivo exclusivamente es proveer más seguridad, por ejemplo, extensiones de servicios, firewall de host, etc. Incluso servicios como IDS e IPS que buscan indicios de una intrusión en el sistema. **De nada sirve todo esto** si un **administrador del sistema** se **deja su sesión abierta**, se **filtra la contraseña de administrador local** entre los empleados de la compañía, o un **usuario introduce un malware** en la red empresarial. Según un estudio realizado por Kaspersky en 2017, reflejaba que el 44% de las empresas admiten que sus empleados no siguen las políticas de seguridad definidas [6].

Lo más preocupante son las medidas que anunciaron las compañías entrevistadas para solucionar los problemas de seguridad, donde vemos que las opciones más votadas son: Utilizar software de seguridad más sofisticado (43%) y entrenar a los empleados en el área de la ciberseguridad (35%) [6]. Como ya hemos visto en el documento, estas medidas no necesariamente mejorarán la situación, por lo que se puede decir que la **gestión de seguridad** de las empresas **no ha puesto el foco sobre el principal problema**, controlar que sus políticas de seguridad se están aplicando correctamente. De esta forma, no solo se controla la seguridad de los servicios a un alto nivel, sino también las acciones más triviales que realizan los usuarios en el sistema.

Por otro lado, es importante destacar una solución de monitorización y gestión de la seguridad que, hoy en día, ya implementan muchas organizaciones, el **SOC** (*Security Operation Center*). Una unidad organizativa, interna o externa, que utiliza un software específico para normalizar las salidas de los sistemas TI pertenecientes a la infraestructura de la organización, tales como: servidores, equipos locales, PLCs, Routers, etc. De esta forma, se pueden unificar las salidas de múltiples sistemas en un único panel que de la información necesaria para actuar a los expertos de seguridad [7]. Aun así, de forma habitual, **este software sigue centrándose** en los **sistemas de alto nivel**, dejando **sin control** las normas de la **política de seguridad**. En los SOC, para la recolección y gestión de eventos, normalmente se utiliza un tipo de herramienta software denominada **SIEM** (*Security Information Event Management*), la cual se explicará en detalle en el siguiente apartado.

Igualmente, esta **problemática también** se **aplica** en los **hogares**, ya que, aunque el número de usuarios que acceden a los dispositivos están más acotados, no siempre podemos controlarlos. Los ejemplos anteriores de desactivación de actualizaciones automáticas o el fallo de estas, inyección de dispositivos externos, presencia de Key Loggers, etc. pueden ser llevados a cabo por miembros de la unidad familiar, e incluso, usuarios puntuales. Hay que destacar que los hogares con dispositivos tecnológicos con acceso a redes públicas también necesitan de una política de seguridad, muchas veces de mayor dureza, debido a la antigüedad de los dispositivos de red, ausencia de actualizaciones, dispositivos IoT, etc. Además, nunca se sabe si se va a ser objetivo de un ataque dirigido, por lo tanto, la herramienta de autodiagnóstico también cobra sentido en el hogar, ya que esta puede dar detalles sobre los problemas actuales, o, por otro lado, proporcionar un informe a un miembro de la familia con conocimientos más avanzados. Sin ir más lejos, un estudio de TrendMicro detectó más de 23 millones de incidentes de seguridad en redes del hogar a través de su telemetría en 2017 [8], donde se incluían diversos ataques basados en contraseñas por defecto, routers vulnerables, servicios disponibles públicamente, etc.

Obviamente, somos consciente de que fortalecer y comprobar la seguridad de los sistemas de entrada a la infraestructura TI de una organización es también algo primordial. Tanto es así que existen multitud de herramientas de diagnóstico para estos, y aunque aportan grandes beneficios, también tienen puntos flacos. Por ejemplo, normalmente, cuando hacemos uso de este tipo de herramientas nos encontramos con dos entornos posibles:

1. **Host local.** En este caso se realiza un análisis más laxo, ya que los servicios activos suelen ser mínimos.

Sin embargo, no se suele comprobar los servicios a los que este tiene acceso. Un ejemplo de esto sería que un equipo del departamento de contabilidad pueda acceder a equipos de TI, lo que sería una prueba de una mala configuración de VLAN. Por otro lado, la comprobación de los servicios asociados a la red empresarial a los que puede acceder este equipo también resulta muy importante, igual nos encontramos con que puede acceder a paneles de administración.

2. **Servidor.** El caso de uso más común, ya que, en teoría, resulta el punto de entrada más obvio de cara a un ciberatacante. Se realiza un análisis más extenso, a varios niveles de las capas del sistema.

En este tipo de entorno, nos podemos encontrar, a grandes rasgos, con dos tipos de soluciones:

- a. La herramienta proporciona un diagnóstico para un servicio concreto, detallando informes sobre el comportamiento de este, problemas de configuración y seguridad. Un ejemplo de este tipo de herramientas serían Apache2Buddy y MysqlTunner.
- b. La herramienta proporciona un diagnóstico general, incluso se pueden realizar pruebas sobre los servicios públicos atacando su interfaz asociada. Un ejemplo de este tipo de herramientas serían detectores de vulnerabilidades como Nessus.

Como podemos observar, **cada tipo de solución** tiene un **problema asociado** respectivamente. Ya que, por un lado, si la herramienta se centra en un único servicio dejaremos de lado el resto, siendo **una tarea muy costosa** el investigar y ejecutar una herramienta de diagnóstico para cada servicio del sistema. Por otro lado, aunque analicemos el estado de la interfaz pública, el origen siempre será nuestra propia máquina local, por lo que **la prueba realmente no será real**.

Por último, resulta evidente que proporcionar un diagnóstico detallado sobre el estado del sistema es una buena acción preventiva, pero en muchas ocasiones no será suficiente para detener un ataque, incluso, puede que ya sea tarde. Es por ello que, aunque una herramienta de diagnóstico debe, principalmente, detectar problemas, tendremos que plantearnos si necesitamos realizar una serie de acciones automáticas. Estas acciones pueden parar en seco posibles ataques o incluso formar de manera implícita a los usuarios del sistema.

4.2. Herramientas actuales

Como se ha nombrado anteriormente en este documento, **no hay una solución única** que englobe todos los problemas identificados en los apartados ya vistos. Aun así, **existe software** de carácter más **general que podemos personalizar** para acotar el aviso de incidentes de seguridad a eventos que realmente nos interesen según los requisitos de la organización.

A grandes rasgos, hay tres tipos de software que pueden realizar este tipo de acciones, los cuales se describen a continuación:

4.2.1. Monitores de eventos

En primer lugar, si queremos detectar eventos relacionados con acciones indebidas por parte de los usuarios, por ejemplo, inserción de dispositivo externo, desactivación de actualizaciones automáticas, uso indebido de aplicaciones, etc. debemos monitorizar el registro de eventos del sistema operativo.

Los **Monitores de eventos** no estarán enfocados plenamente en la seguridad como los SIEM, sino que tendrán una perspectiva más general, recopilando eventos de todo tipo. Algo que puede venir realmente bien para realizar una monitorización más minuciosa de las acciones que se realizan en un equipo. Además, resulta difícil clasificar todo lo que puede ser un incidente de seguridad, ya que, como se ha comentado anteriormente, puede haber acciones que una organización considere como incidente de seguridad, mientras que para otro no tenga mayor importancia. Es importante destacar que, un monitor de eventos con capacidad de filtrado puede ejercer de herramienta de autodiagnóstico, de hecho, si se encuentra bien configurada puede ser mejor que una solución comercial, ya que identifica lo que verdaderamente haría daño a nuestra infraestructura TI. Un ejemplo de este tipo de herramienta es:

SolarWinds Event Log Consolidator / Manager [9]

Con esta herramienta podemos monitorizar un gran número de equipos con sistemas operativos basados en Windows, de forma que obtenemos un registro centralizado de los eventos. Como se ha comentado, se recogen eventos de manera general, por lo que una funcionalidad muy interesante que nos ofrece esta herramienta es la de crear filtros para identificar cierto tipo de eventos. De esta forma se pueden englobar los eventos que interesan a nuestra organización y programar un aviso vía email si estos se suceden en los equipos que hayamos definido.

También permite la ejecución de acciones predefinidas ante ciertos eventos, algo muy importante ante situaciones de descontrol o posibles ataques. Por último, esta herramienta nos ofrece la posibilidad de visualizar informes con grafos para proporcionar una fácil lectura al experto de ciberseguridad, algo que será común a todas las herramientas que veamos en este apartado.

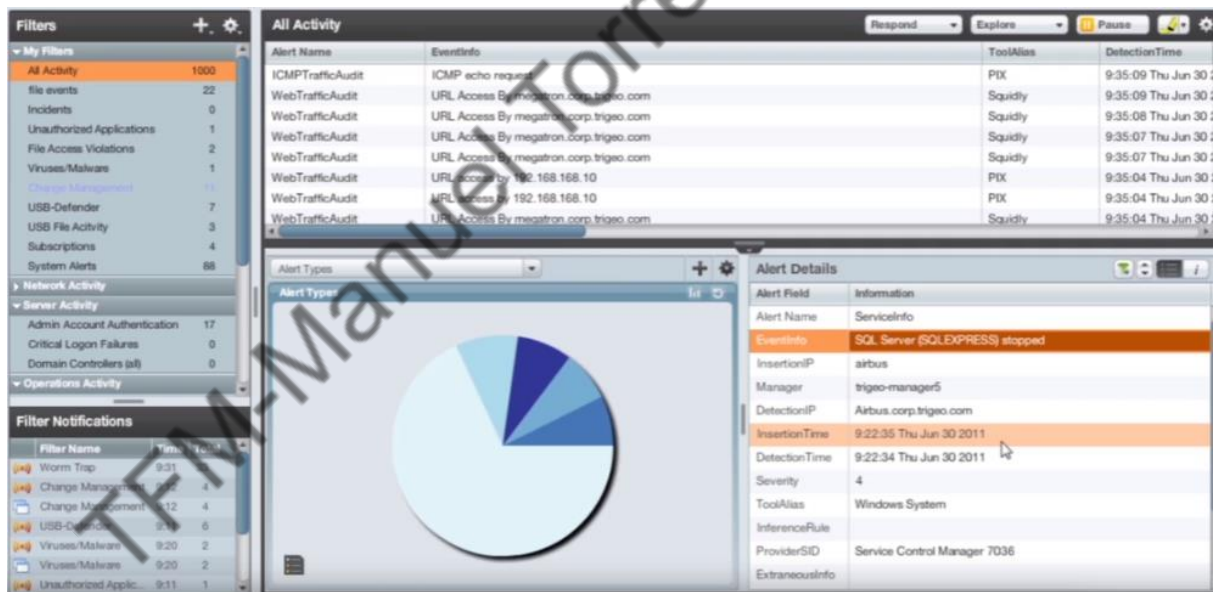


Figura 1. Captura EventLog Manager en plena ejecución. Fuente: [SolarWinds](#).

4.2.2. SIEM (*Security Information and Event Management*)

Los **SIEM**, ante la complejidad de identificar todos los tipos de eventos que pueden suponer un incidente en esta área, teniendo en cuenta el contexto de la organización, son implementados, en multitud de ocasiones, desde cero por la organización que requiera de estos servicios. Por otro lado, no se suelen centrar solo en el registro de eventos del sistema, sino que también utilizan correlación de eventos para la detección de ataques, detección de cambios de integridad, además de obtener eventos de múltiples dispositivos que forman la infraestructura TI (P.ej. Dispositivos de red, SAS, etc.).

Aunque existen más soluciones comerciales genéricas, un ejemplo sería la siguiente:

EventSentry SIEM [10]

Esta herramienta representa una solución centrada exclusivamente en la seguridad de las máquinas que monitoriza. Utiliza un agente que se instala en la máquina que se quiere monitorizar, este registrará, tanto los eventos del sistema como los de los servicios específicos que ejecuta. Además, creará un dominio virtual al que se podrá acceder de forma remota por medio de un navegador web para visualizar el estado actual del sistema.

Por otro lado, también implementa diversas acciones ante la ocurrencia de alertas de seguridad, por ejemplo, la posibilidad de parar, iniciar o programar un servicio concreto de manera remota. Esto es algo que causa controversia, ya que muchas implementaciones intentan independizar el registro de incidencias de seguridad de las respuestas activas a estas, ya que esto puede suponer un gran problema de seguridad para los sistemas monitorizados si el nodo central se ve comprometido.

Por último, como se puede observar en la siguiente figura, también ofrece informes con gráficos para facilitar la identificación de problemas por parte del experto en ciberseguridad.

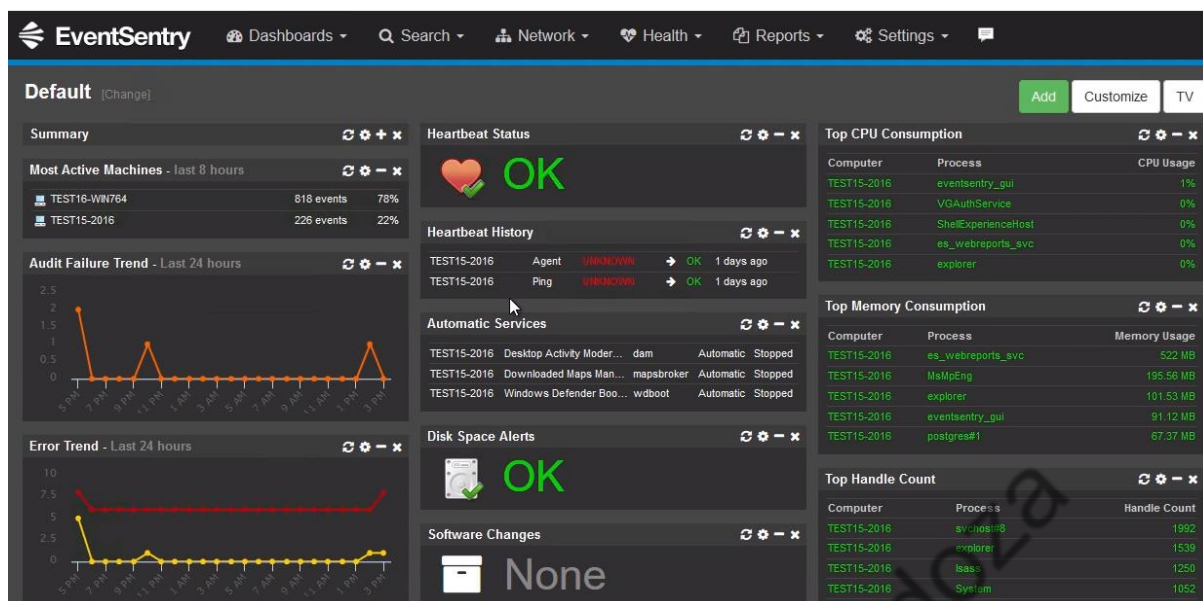


Figura 2. Captura de vista local de equipo con herramienta EventSentry SIEM. Fuente: [EventSentry](#).

4.2.3. UEBA (User and Entity Behavior Analysis)

Por último, se va a destacar un tipo de herramienta **UEBA**, estas suelen identificar una alerta de seguridad por medio de heurísticas internas que toman los eventos del sistema como entrada. Lo positivo de este tipo de herramientas, es que muchas toman en cuenta acciones más triviales de los usuarios, por ejemplo, ejecuciones de aplicaciones que no han sido definidas en una lista blanca, creación de ficheros ejecutables en carpetas compartidas, etc. Por el contrario, no realiza ninguna prueba contra los servicios, ni tampoco permite llevar a cabo alguna respuesta activa ante el incidente. Un ejemplo de este tipo de herramienta sería la siguiente:

Netwrix User Behavior Analytics [11]

Esta herramienta utiliza un sistema de detección de incidentes de seguridad basado en prioridades, por lo que se debe de asignar una prioridad a cada tipo de activo que queremos monitorizar. Por otro lado, también podremos visualizar las acciones de los usuarios, aunque lo más interesante será el informe que nos proporciona, donde podemos ver los incidentes de seguridad clasificados por prioridad. Además de diagnóstico, nos ofrece la posibilidad de gestionar las incidencias, por ejemplo, asociando una resolución a la incidencia para la creación de un histórico.

← User Profile (ENTERPRISE\J.Smith)

Home > Behavior Anomalies (ENTERPRISE\J.Smith)

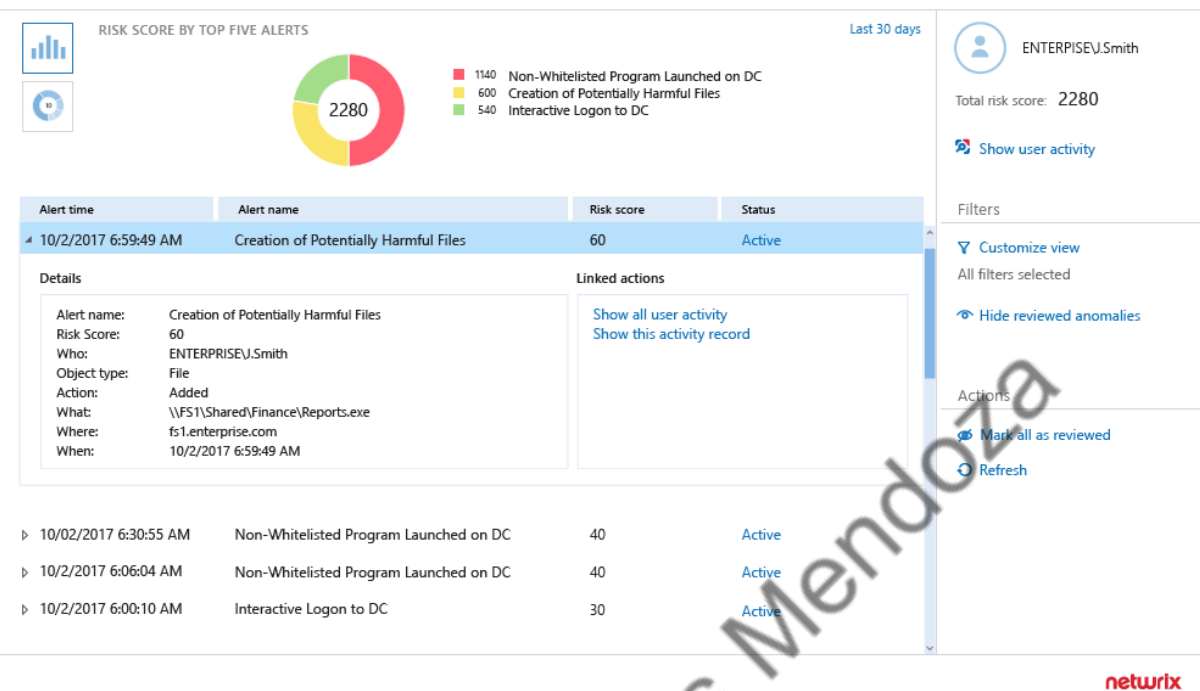


Figura 3. Captura de los incidentes de seguridad de un usuario con la herramienta Netwrix User

Behavior Analytics. Fuente: [Netwrix](#).

A continuación, se puede observar una tabla comparativa entre las tecnologías mencionadas en este apartado:

Tecnología	SolarWinds Event Log	EventSentry SIEM	Netwrix User Behavior Analytics
Objetivo	Obtener un registro central de eventos de sistema para que, a través de un sistema de filtrado, se obtenga el estado actual del equipo.	A través de un registro central de eventos de sistema, servicios y aplicaciones, realizar una correlación de todos ellos para detectar intentos de ataques. También permite ejecutar operaciones remotas a modo de respuesta activa ante un incidente.	Recoge un registro central de eventos del sistema que tienen que ver con el comportamiento del usuario. De esta manera, se identifican acciones poco seguras en base a la prioridad de los activos.
Requisitos	<ul style="list-style-type: none"> - 1 equipo que ejecute el agente. - 1 equipo que ejecute el servidor. 	<ul style="list-style-type: none"> - 1 equipo que ejecute el agente. - 1 equipo que ejecute el servidor. 	<ul style="list-style-type: none"> - 1 equipo que ejecute el cliente de administración. - 1 equipo que ejecute el servidor. - Equipos a monitorizar
Licencia	<ul style="list-style-type: none"> - Versión básica gratuita. - Versión Manager de pago. 	<ul style="list-style-type: none"> - Versión de pago con periodo de prueba gratuito. 	<ul style="list-style-type: none"> - Versión de pago con periodo de prueba gratuito.
Ventajas	<ul style="list-style-type: none"> - A través de los filtros, se pueden detectar alertas de seguridad cotidianas (P.ej. Inyección de dispositivo externo). - Permite avisar ante incidentes de seguridad. - Buena opción si no se dispone de un gran número de equipos. - Versión de pago permite llevar a cabo acciones de respuesta activa ante incidentes (P.ej. bloquear IP). 	<ul style="list-style-type: none"> - Detección de intentos de ataque. - Posibilidad de realizar acciones de respuesta activa importantes (P.ej. parada de servicio). 	<ul style="list-style-type: none"> - Detección de comportamientos poco seguros por parte de los usuarios. - Integración con SIEM para proporcionar una protección completa. - Detección de ataques internos. - Enfocado a grandes infraestructuras TI
Desventajas	<ul style="list-style-type: none"> - Solo para equipos Windows. - Enfocado para una gran infraestructura TI. - Los filtros no vienen predefinidos, lo que hace necesario contar con conocimientos sobre los tipos de eventos del sistema y su relación con las acciones de un usuario. - Si se quiere utilizar la mayoría de las funcionalidades se debe de pagar. - Su arquitectura se basa en un punto central (servidor) que actúa como maestro de la información recibida, no es una solución centrada en una única máquina. 	<ul style="list-style-type: none"> - Solo para equipos Windows. - No detecta vulnerabilidades. - Levanta servicios en la máquina que monitoriza que pueden significar otros puntos de ataque. - Enfocado para una gran infraestructura TI. - También se basa en cliente-servidor, no en un solo componente en la máquina. 	<ul style="list-style-type: none"> - Solo para equipos Windows. - No detecta vulnerabilidades. - No detecta ataques externos. - También se basa en una arquitectura de varios agentes. - Enfocado a una gran infraestructura.

Tabla 5. Resumen de las características que ofrece el software analizado en este apartado. Fuente:

Elaboración propia.

Como se puede observar en la tabla resumen, la gran mayoría de las herramientas están enfocadas a trabajar en una infraestructura mediana o grande, con múltiples dispositivos, de ahí que se dividan tareas entre varios componentes para proveer mayor cohesión al sistema. Esto puede ser un proceso demasiado costoso si nuestra infraestructura es muy pequeña (P.ej. Un hogar o Pyme) y puede no tener sentido de implementar.

4.3. Conclusiones

A raíz de las problemáticas comentadas en este capítulo, podemos identificar una serie de funcionalidades que debe de llevar a cabo la herramienta que se desarrolle en este proyecto.

Por un lado, desde un punto de vista técnico, deberá de llevar a cabo una serie de comprobaciones del registro de eventos del sistema para detectar incidentes, todo esto en función de las prioridades del usuario. A través de esta funcionalidad, la herramienta se centra en la detección de incidentes creados de manera directa o indirecta por un usuario.

Por otro lado, un componente externo debe de realizar un análisis de los servicios públicos que tenga el usuario. A través de esta funcionalidad, la herramienta se centra en la prevención de incidentes provocados por un posible atacante.

Desde el punto de vista comercial, se ha visto que el software dirigido a solventar este tipo de problemáticas suele estar centrado en una de las dos áreas comentadas anteriormente (incidente interno o prevención). Además, suelen ir enfocados a infraestructuras bastante más complejas que las asociadas a hogares o pymes. Por lo tanto, es una oportunidad de negocio que la herramienta abarque este sector de mercado, ya que se encuentra prácticamente sin explotar.

5. Objetivos

El **objetivo general** de este proyecto es el de desarrollar una herramienta que permita detectar, tanto vulnerabilidades como acciones triviales de los usuarios que puedan representar un riesgo de seguridad para el sistema.

Por otro lado, para lograr este objetivo principal se necesitan lograr otros a medida que se vaya desarrollando el proyecto:

- Estudio de la gestión de eventos en sistemas operativos, principalmente Windows.
- Estudio de herramientas que permitan la gestión de eventos, tanto del sistema operativo como de los servicios y aplicaciones instalados en este.
- Realizar la especificación de la herramienta a desarrollar a través de los requisitos identificados.
 - Identificación de funcionalidades.
 - Definición de arquitectura.
 - Estudio y selección de tecnología.
 - Diagramas asociados.
- Estudio y diseño de una solución que permita a un componente externo a la red realizar una serie de pruebas para la identificación de vulnerabilidades.
- Implementación de la herramienta de autodiagnóstico.
- Implementación del componente externo.
- Pruebas y validación en diferentes entornos.
- Desarrollo de conclusiones.

Los objetivos anteriores son generales del desarrollo del proyecto, existen otros **enfocados en la solución** que se va a proporcionar a los usuarios de esta herramienta:

- Capacidad de identificar acciones cotidianas de un usuario del sistema que suponen un riesgo de seguridad.
- Capacidad de identificar malas configuraciones (a nivel básico) en la red interna.
- Capacidad de identificar malas configuraciones (a nivel básico) en los servicios públicos.
- Capacidad de alertar a un usuario ante una incidencia de seguridad en el sistema.
- Capacidad de generar informes que aporten la información necesaria a una persona con conocimientos técnicos, no necesariamente muy altos, para solucionar un problema de seguridad.
- Capacidad de que el usuario pueda visualizar avisos o sufrir acciones en forma de respuesta activa a un incidente de seguridad.
- Disminuir el número de incidentes de seguridad que ocurren en el sistema.

6. Metodología

La metodología de trabajo que se ha decidido utilizar para este proyecto es Waterfall [12]. Esta metodología de desarrollo se centra en la división del proyecto en distintas fases que serán ejecutadas de manera secuencial.

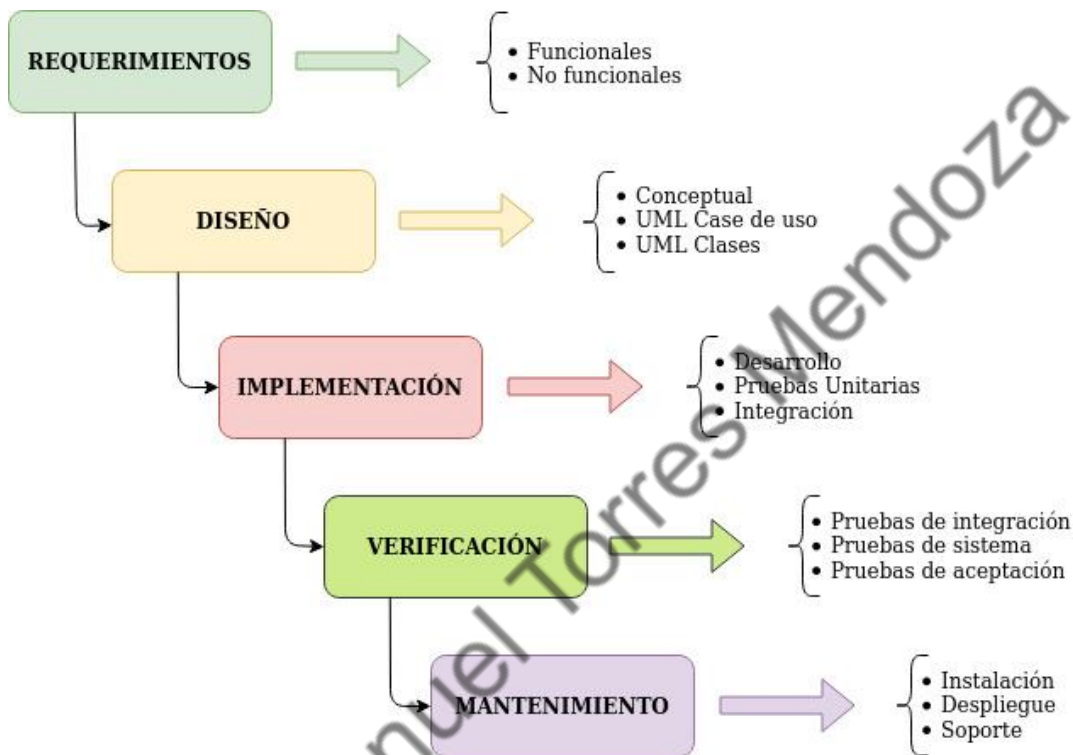


Figura 4. Fases de ejecución de un proyecto siguiendo la metodología Waterfall. Fuente: Elaboración propia.

Waterfall propone una serie de ventajas al utilizarse en este proyecto con respecto a otras metodologías ágiles estudiadas (P.ej. SCRUM, Proceso Unificado, etc.):

- + **No se basa en un proceso iterativo**, lo que permite avanzar de forma más directa en el desarrollo del proyecto. Lo cual, aunque hace necesario que se deba de llevar a cabo una muy buena realización de cada fase del proyecto si no se quieren problemas posteriores debidos a dependencias, hace posible que cuando se complete una fase no se vuelva atrás, permitiendo una mayor organización de los recursos disponibles.
- + **No está enfocada en la realización de reuniones continuas**. Esto es algo que no tiene sentido en mi proyecto, ya que soy el único desarrollador.

- + **Se centra en la planificación temporal**, lo cual es un factor muy importante para este proyecto.
- + Uno de sus objetivos es la **documentación del software**, a mi entender, algo que se alinea al completo con los objetivos de un TFM. Además de revisiones puntuales, las cuales pueden ser realizadas por el tutor.
- + La fase de **mantenimiento** es de **gran importancia** para este proyecto, ya que como se ha comentado, hay una gran dependencia con respecto al funcionamiento de las tecnologías que se utilizan en la herramienta. Debido a esto, un mantenimiento continuo que tenga en cuenta los cambios que van surgiendo en las dependencias resulta de gran utilidad.

Otras metodologías que se han estudiado han sido las siguientes:

- Modelo en espiral [13]. Ha sido descartada, debido a que, en general, va enfocada a proyectos de gran complejidad.
- RAD (*Rapid Application Development*) [14]. Aunque tiene un enfoque dinámico, centrado en una rápida especificación y diseño e implementación continua, puede alargar mucho la finalización del proyecto. Además, está muy basada en la realización de prototipos, lo cual no tiene sentido en mi caso.

6.1. Herramientas utilizadas

Para llevar a cabo los mecanismos de organización del desarrollo del proyecto que marca la metodología Waterfall, he utilizado varias herramientas. Se podrían categorizar en los siguientes apartados:

- **Análisis y diseño del proyecto:** Me ayudarán a realizar las primeras fases del ciclo de vida del proyecto de manera rápida y comprensible. De esta forma, tal y como marca la metodología, mi proyecto será entendible por otros expertos en el área.
 - Utilizaré Draw.io para todos los tipos de diseño que utilice para la definición inicial del proyecto.

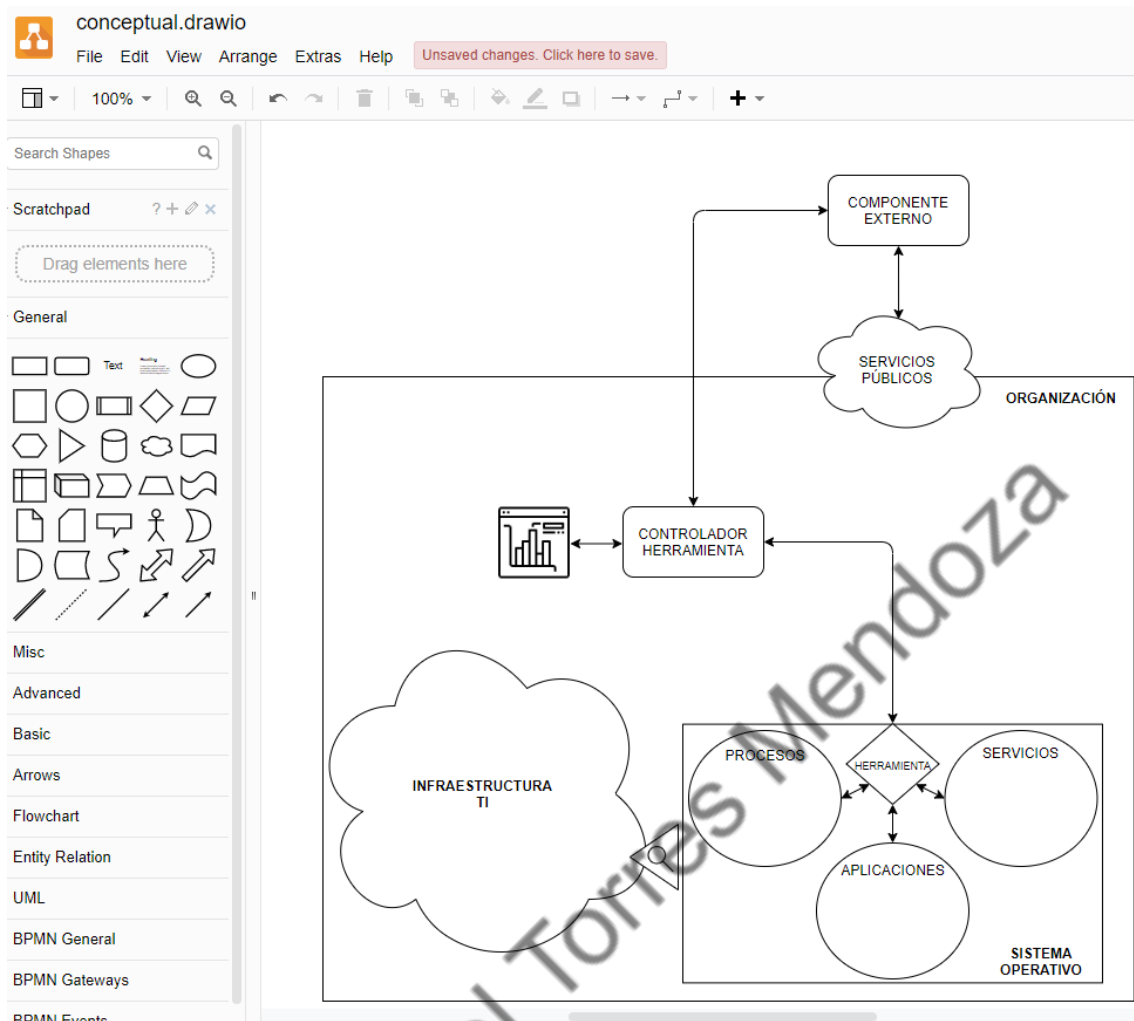


Figura 5. Diseño conceptual del software mediante la herramienta Draw.io. Fuente: Elaboración propia.

- **Organización temporal de las tareas de implementación:** Obtendré una división bastante completa de las tareas de implementación que debo de llevar a cabo, teniendo en cuenta, tanto plazos de entrega como prioridad.
 - Utilizaré Trello para este objetivo.

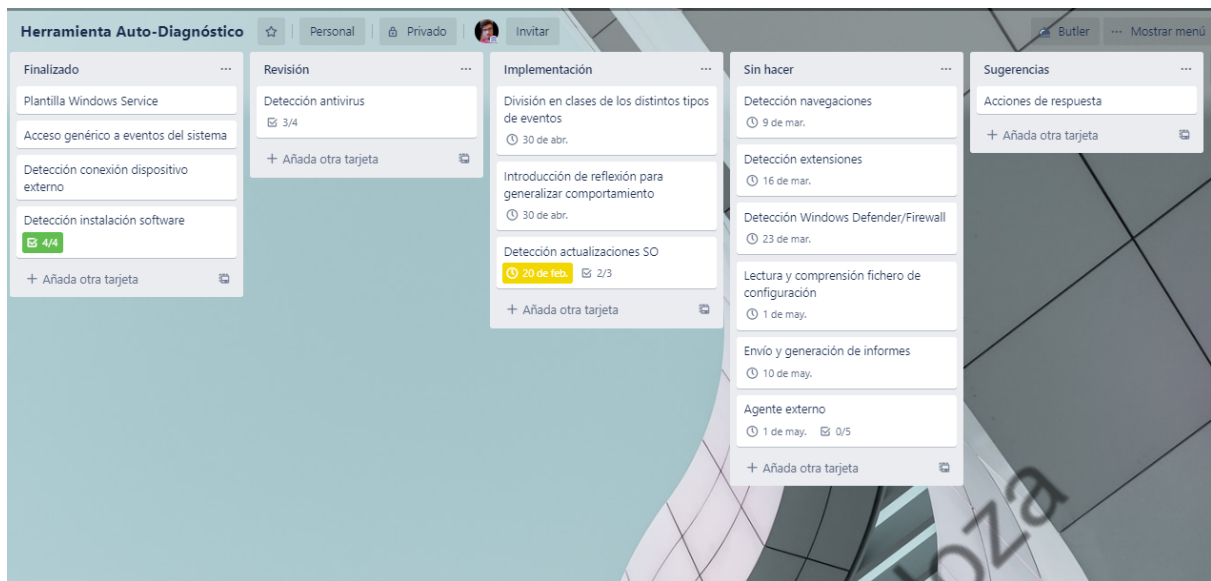


Figura 6. Organización temporal de las tareas de implementación. Fuente: Elaboración propia.

Por último, quiero destacar que, para la **definición de requisitos** de la herramienta, una vez identificados los agentes que intervienen y la arquitectura, utilizaré herramientas de ofimática común.

7. Análisis y especificación

En esta sección del documento se definen los requisitos de comportamiento que debe seguir el software que se va a desarrollar a lo largo del proyecto. Para este propósito, se ha seguido el estándar IEEE 830 [15] con unas ligeras variaciones que se expondrán a lo largo del documento.

7.1. Alcance

Este proyecto, se basa en el desarrollo de una herramienta de autodiagnóstico centrada en el marco de la ciberseguridad. Tendrá el objetivo de, a raíz de una serie de especificaciones dadas por un usuario **Administrador**, ser capaz de detectar comportamientos ilícitos por parte de los usuarios de ese sistema concreto, ya sean el propio administrador u otro, a los cuales se determinará **Usuario Sistema**. Estos comportamientos irán dirigidos a acciones generalistas como ya se ha comentado a lo largo del documento.

Esta herramienta está pensada para ser implementada como un servicio ejecutado en segundo plano, que obtenga los datos de comportamiento pertinentes de un fichero de configuración, por lo tanto, la creación de interfaces gráficas no será un requisito esencial de este software.

Por otra parte, el Administrador podrá definir que un agente externo, localizado en Internet, ejecute análisis periódicos a sus servicios públicos (P.ej. el servidor web), previamente definidos. Este agente externo tiene que ser accesible desde el sistema local que ejecuta la herramienta, tanto para proporcionarle información de los servicios, como para recibir informes.

Una vez que se lleven a cabo estas dos funciones, la herramienta debe de ser capaz de generar un informe detallado al Administrador.

Por lo tanto, podemos definir el ámbito de funcionamiento del software a desarrollar como híbrido, ya que se ejecuta, tanto en la máquina local de un usuario, como en un servidor público de Internet.

7.2. Características de los usuarios

7.2.1. Administrador

Va a necesitar de cierto conocimiento técnico, aunque este no será muy elevado, debe de saber editar un fichero de configuración con los parámetros concretos que proporciona la herramienta y ser consciente de la causa del error si este sucede.

Por otro lado, debe de ser capaz de razonar ciertas conclusiones que se obtendrán del informe final que proporcione la herramienta. Estas pueden incluir acciones necesarias para subsanar problemas actuales de seguridad (P.ej. Activa las actualizaciones automáticas, se ha detectado el puerto 80/HTTP abierto a Internet, etc.).

Deberá de tener permisos de administración en el sistema operativo.

7.2.2. Usuario Sistema

No necesitará ningún tipo de conocimiento técnico, ya que su única función será la de realizar tareas cotidianas, pudiendo realizar una que suponga un incidente de seguridad y ser detectada por la herramienta.

No tiene por qué tener permisos de administración en el sistema operativo, ya que, si los tiene puede parar la ejecución de la herramienta.

7.3. Restricciones

7.3.1. Red

El sistema en el que se ejecute la herramienta debe de tener acceso a Internet, en especial, al agente externo que realiza los análisis de seguridad, siempre y cuando se quiera contar con esta funcionalidad.

7.3.2. Software

Se debe de contar con un sistema operativo Windows 10, ya sea de 64 o 32 bits para que la herramienta pueda funcionar.

Por otro lado, también se debe de contar con el siguiente paquete de software:

- Python 3.
- Sysmon

7.3.3. Legislativa

El cliente deberá de haber aceptado, previamente a la instalación del software, que se realicen pruebas de seguridad externas (P.ej. Escaneo de puertos) que pueden atentar a la disponibilidad y rendimiento del sistema.

Hay que tener en cuenta que la elección de los servicios públicos a analizar por parte de la organización puede llevar a un problema legal. Esto es debido a que, si la organización cliente nos proporciona servicios públicos que no le pertenecen, el análisis de estos por parte de nuestra herramienta puede llevar a pensar a la organización dueña de los servicios públicos objetivo que está sufriendo un ciberataque (P.ej. DDoS), pudiendo llevar a cabo acciones legales en nuestra contra. Por ello, se especificará en el capítulo de diseño del agente externo como se va a resolver esta problemática.

7.4. Suposiciones y dependencias

En este apartado se tendrán en cuenta diversos factores que si cambian pueden causar que se tengan que revisar los requisitos definidos.

- Actualización de sistema operativo que afecte a la gestión de eventos.
- Actualización de sistema operativo que afecte al registro de Windows.
- Actualización del lenguaje de programación empleado que afecte al funcionamiento de las funciones utilizadas.
- Actualización o descubrimiento de vulnerabilidad en una de las librerías utilizadas.

7.5. Definiciones y organización del documento

Los requisitos del sistema serán representados mediante tablas que contendrán la siguiente información a modo de tuplas clave-valor:

Identificador - [RF | RNF].[Núm.] → P.ej. RF.1

Se utilizarán las siglas RF para Requisitos Funcionales, mientras que RNF para Requisitos No Funcionales.

Usuario - [Usuario Sistema | Administrador]

Sólo válido para requisitos funcionales.

Nombre - Título del requisito.

Tipo - [Obligatorio | Opcional]

Se entenderá que un requisito es obligatorio cuando su prioridad es muy alta y debe de ser completado antes de la fecha de finalización del proyecto.

Se entenderá que un requisito es opcional cuando su prioridad es media/baja y el proyecto puede ser entregado sin este.

Ámbito - [Agente Externo | Herramienta | Ambos]

Relación de un requisito, ya sea funcional o no, con el agente externo que ejecuta pruebas a servicios públicos, la herramienta que se ejecuta en el sistema local del usuario o ambos.

Descripción - Explicación del requisito.

7.6. Requisitos

7.6.1. Funcionales

Identificador:	RF.1
Usuario:	Administrador
Nombre:	Elección de eventos a monitorizar
Tipo:	Obligatorio
Ámbito:	Herramienta
Descripción:	Se podrá elegir entre una lista de acciones que pueden suponer un incidente de seguridad (P.ej. inyección de dispositivo externo, desactivación Antivirus, etc.) para que la herramienta las tome en cuenta en el diagnóstico.

Identificador:	RF.2
Usuario:	Administrador
Nombre:	Elección de uso de agente externo
Tipo:	Obligatorio
Ámbito:	Ambos
Descripción:	Se podrá elegir si se requiere que el agente externo realice un análisis de los servicios públicos o no. Si la respuesta es afirmativa se podrá elegir la hora de la ejecución.

Identificador:	RF.3
Usuario:	Administrador
Nombre:	Elección de servicios a monitorizar
Tipo:	Obligatorio
Ámbito:	Ambos
Descripción:	Se podrá elegir, entre una lista de servicios, los que se quiera tener en cuenta para el análisis periódico del agente externo, siendo las pruebas realizadas únicas para cada uno de estos.

Identificador:	RF.4
Usuario:	Administrador
Nombre:	Generación de informes
Tipo:	Opcional
Ámbito:	Herramienta
Descripción:	Se podrá elegir si se quiere que la herramienta genere informes periódicos a modo de resúmenes de autodiagnóstico de seguridad.

Identificador:	RF.5
Usuario:	Administrador
Nombre:	Generación de avisos
Tipo:	Opcional
Ámbito:	Herramienta
Descripción:	Se podrá elegir si se quiere que la herramienta genere avisos cuando detecte algún tipo de evento especificado como incidente de seguridad. Estos avisos podrán ser enviados a un mail definido por el administrador.

Identificador:	RF.6
Usuario:	Administrador
Nombre:	Elección formato del informe
Tipo:	Opcional
Ámbito:	Herramienta
Descripción:	Se podrá elegir, entre una lista de formatos, el que quiera para la generación de los informes.

Identificador:	RF.7
Usuario:	Administrador
Nombre:	Elección de acciones de respuesta a incidentes
Tipo:	Opcional
Ámbito:	Herramienta
Descripción:	Se podrá elegir, entre una lista de acciones, la que se quiera ejecutar en respuesta a un incidente de seguridad.

7.6.2. No Funcionales

Identificador:	RNF.1
Nombre:	Franja de tiempo de detección de eventos asociados a incidentes de seguridad.
Ámbito:	Herramienta
Tipo:	Obligatorio
Descripción:	El intervalo de tiempo máximo en el que la herramienta tiene que ser capaz de detectar un incidente de seguridad tiene que ser menor o igual a 5 minutos. En este intervalo no se incluye el envío de avisos ni posibles retardos en las comunicaciones.

Identificador:	RNF.2
Nombre:	Seguridad
Ámbito:	Ambos
Tipo:	Obligatorio
Descripción:	<ul style="list-style-type: none"> - Comunicaciones Las comunicaciones entre la herramienta y el agente externo deben de ir cifradas utilizando un criptosistema seguro. - Protección de la información Los datos asociados a los análisis de seguridad de los servicios públicos de las organizaciones que se almacenen de manera persistente deben de estar cifrados - Accesibilidad Los datos asociados a los análisis de seguridad de la organización deben de ser accesibles por esta tras una autenticación. - Disponibilidad El sistema tiene que estar disponible cerca del 99% del tiempo. - Trazabilidad Tanto la herramienta como el agente externo deben de ser capaces de mantener un registro continuo de las acciones que realizan, de tal forma que puedan ser examinados por los administradores o un analista forense en caso de ser necesario.

Identificador:	RNF.3
Nombre:	Eliminación de los datos en Agente Externo
Ámbito:	Agente externo
Tipo:	Obligatorio
Descripción:	Se debe de proporcionar un mecanismo por el cual un cliente puede eliminar los datos asociados a los análisis de seguridad de sus servicios públicos almacenados en el agente externo. De esta forma, se cumplirá con el Derecho a Ser Olvidado del RGPD.

Identificador:	RNF.4
Nombre:	Copias de seguridad
Ámbito:	Agente externo
Tipo:	Opcional
Descripción:	Se deben de programar copias de seguridad de los datos de manera periódica.

Identificador:	RNF.5
Nombre:	Escalabilidad
Ámbito:	Ambos
Tipo:	Opcional
Descripción:	<p>La herramienta debe de permitir la adición de nuevos tipos de eventos de seguridad para monitorizar a través de actualizaciones propias sin que se vea afectado en gran medida el rendimiento.</p> <p>El agente externo también debe de permitir realizar análisis de seguridad a servicios públicos de manera que la adición de nuevos objetivos no suponga problemas de rendimiento o disponibilidad.</p>

Identificador:	RNF.6
Nombre:	Auditoría
Ámbito:	Ambos
Tipo:	Opcional
Descripción:	<p>La herramienta debe de poder enviar información estadística de los eventos asociados a incidentes de seguridad capturados.</p> <p>El agente externo, a través de la información almacenada de manera persistente, será capaz de proporcionar información estadística sobre las vulnerabilidades encontradas.</p>

8. Diseño

8.1. Solución Conceptual

La solución a estos problemas pasaría por construir una **herramienta** que, en primer lugar, se instale de manera independiente en cada sistema, abstrayéndolos del resto de la infraestructura TI, lo que causa que el **experto en ciberseguridad** pueda consultar rápidamente los dispositivos que le interesan sin sorpresas, además de revisar los **servicios** que ofrece cada dispositivo (P.ej. Servidor Web, SNMP, SSH, etc.). Con toda esta información, el experto ya podría identificar posibles problemas de configuración y/o seguridad.

Aun así, investigar cada servicio ofrecido por cada dispositivo en un entorno cambiante podría ser interminable. Por ello, la herramienta debería de ofrecer un **resumen** de las **vulnerabilidades** que pueden tener los servicios identificados, eso sí, **a bajo nivel**, ya que queremos centrarnos en incidentes generalistas. El objetivo no es centrarse en detectar vulnerabilidades complejas, ya que para esto existe software específico muy bueno. Claramente, la información de vulnerabilidades debe de estar actualizada. Además, de forma ideal la herramienta podría **aconsejar sobre la posible solución** de la vulnerabilidad a nivel de configuración del servicio para facilitar la labor del experto, o al menos, proporcionar una referencia para más información.

Por otro lado, se debería de controlar el cumplimiento de la política de seguridad por parte de los usuarios. Esto es algo complejo, ya que todas las organizaciones definen sus políticas en función de su misión, funcionamiento y entorno. Por ello, la herramienta puede **definir** una serie de **normas estándar**, las cuales sabemos que van a suponer un riesgo de seguridad en cualquier entorno (P.ej. usuario desactiva el antivirus, conecta USB, etc.). De forma que el administrador pueda **definir** una serie de **acciones personalizadas** (normas a seguir) acorde con la política de seguridad de la organización.

Para solucionar las problemáticas asociadas a las herramientas de diagnóstico tradicionales, en primer lugar, necesitamos un **componente** que se sitúe, de manera estratégica, **fuera de la intranet del equipo**, de esta forma se podrán probar las medidas de seguridad de los servicios públicos. De manera implícita, este componente realizará un análisis de una gran cantidad de servicios, no centrándose en exclusiva en uno de ellos. Incluso puede ser el administrador de la herramienta quien defina las pruebas a realizar, en función de los servicios que este ofrece.

En definitiva, la herramienta podrá realizar un análisis preventivo de la seguridad asociada a la infraestructura TI de la red donde se encuentra el equipo, tanto desde el punto de vista de un atacante externo, como el interno.

El experto en ciberseguridad debería ser capaz de definir '*triggers*' o **acciones desencadenantes** en respuesta a alertas de seguridad. Estas pueden basarse en cualquier acción detallada anteriormente, la cual sea considerada como situación de riesgo extremo para la organización. Por ejemplo, si se detecta algún tipo de comportamiento malicioso en un equipo, el **controlador de la herramienta** podría ser capaz de ejecutar una orden predefinida por el experto de ciberseguridad, una posible opción sería apagar el equipo. Esta funcionalidad aportaría un gran control al experto sobre la infraestructura TI de manera optimizada, no teniendo que ir físicamente al equipo afectado. El enfoque de la acción desencadenante puede variar según la alerta de seguridad, por ejemplo, si un usuario acepta la navegación en un sitio web sin certificado digital, en lugar de apagar el equipo, se le podría mostrar un aviso de que está cometiendo una infracción y explicarle por qué. De esta forma, la herramienta también sería compatible con un posible plan de formación y concienciación de la organización.

Por último, es evidente que la herramienta debe de **proporcionar** una serie de **informes** y estadísticas al experto en ciberseguridad, de manera que, según los resultados y las prioridades de la organización, se escojan las soluciones apropiadas. Debido al funcionamiento basado en alertas, la mejor manera de informar al experto estaría basada en un panel de control de monitorización continua, de tal forma que los resultados estén actualizados al orden de pocos minutos. Por el contrario, esta solución sería bastante costosa, ya que seguramente implicaría separar el controlador de la herramienta en otro sistema que tenga una interfaz que pueda mostrar este panel de control. Por lo tanto, debido a que la idea inicial es enfocarse en organizaciones medianas/pequeñas y hogares, la mejor opción es la de mantener la posibilidad de obtener información continua sobre las alertas de seguridad e informes diarios, pero a través de correo o consultas en el equipo local.

Proporcionar información sobre alertas de seguridad, de manera más o menos síncrona (con un retraso aceptable), al experto en ciberseguridad es una prioridad, ya que obtener un informe detallado que te diga que se ha producido una infección en el sistema hace 9 horas, sirve de poco en cuanto a evitar el problema, únicamente de cara a un análisis forense posterior.

Uno de los problemas que se han visto en el *Capítulo 5. Estado del Arte*, ha sido que el **software actual** que puede lograr unos objetivos similares al de este proyecto va **enfocado** plenamente a organizaciones con una **infraestructura extensa**, dejando de lado infraestructuras más modestas donde obtener un servidor dedicado resulta muy difícil. Por lo que **dotar al software** de los **menos sistemas** independientes posibles es una **prioridad en el diseño**. Por lo tanto, aunque de manera conceptual se distinga entre **Herramienta y Controlador**, realmente **pueden pertenecer al mismo componente**, siendo su cometido distinto de manera conceptual.

De esta manera, vemos como se elimina la necesidad de contar con una gran variedad de componentes para llevar a cabo la solución del problema, ya que simplemente se necesita instalar la herramienta en un equipo. Por lo que, no necesitamos servidores ni clientes intermedios que recogen información, concluyendo que la solución será compatible con organizaciones de todo tipo y hogares.

A continuación, se puede observar un diagrama conceptual de lo propuesto como arquitectura conceptual:

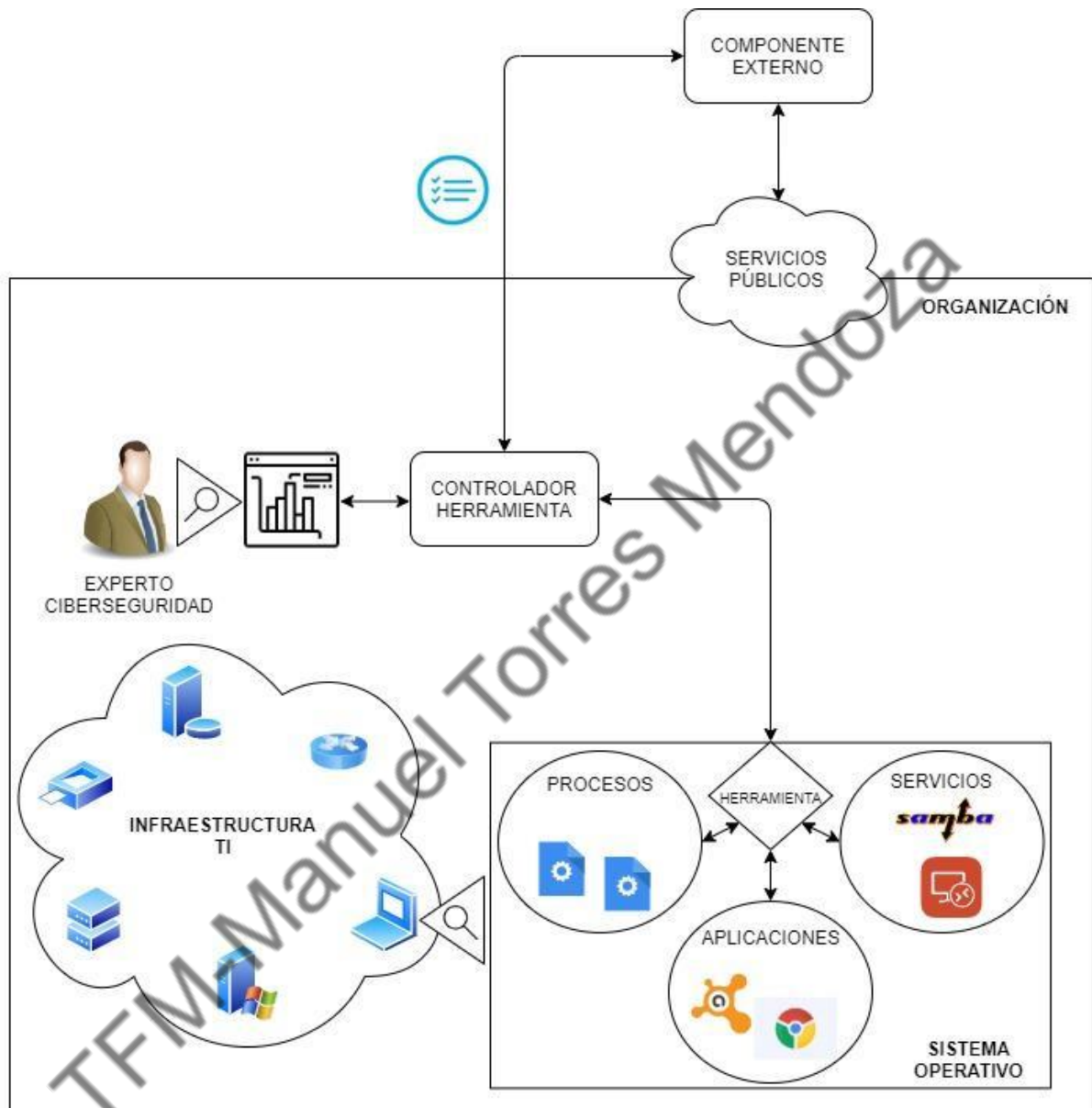


Figura 7. Diferentes agentes que intervienen en el funcionamiento de la herramienta. Elaboración propia.

8.2. Solución Tecnológica

En primer lugar, debido a que el software va a contar con dos agentes diferentes como son el agente externo y la herramienta local, a continuación, se van a detallar las tecnologías que se van a utilizar para cada uno.

8.2.1. Herramienta

Hay que destacar que, debido a que el mercado al que va enfocado la solución se centra en dispositivos con sistema operativo Windows, la herramienta será compatible, en una primera versión, únicamente con este sistema, concretamente centrándose en la versión Windows 10. Por lo tanto, toda la elección de tecnologías se habrá hecho con respecto a esta decisión.

La herramienta se va a implementar como un **servicio de Windows 10** que se ejecute en **segundo plano**, de esta manera, se realizarán comprobaciones de manera periódica sobre el registro de eventos del sistema, pudiendo avisar al Administrador si se detecta un incidente de seguridad. Esto es posible debido a que Windows 10 mantiene un registro de eventos referido en la mayoría de las ocasiones como *Windows Event Logging Service* [16], el cual puede ser analizado y filtrado por el Visor de Eventos de Windows [17], una funcionalidad gráfica que nos será de gran ayuda en la implementación, validación y pruebas.

Aunque **Windows 10** extrae la información de los **registros** de una serie de ficheros, estos tienen un formato propietario y sería más costoso tratarlos de manera manual. Por lo que, ha sido un **factor determinante** para **escoger** el **lenguaje** de programación que será usado para el desarrollo de la herramienta que **Python** [18] **proporcionase librerías** específicas para el **acceso** al **registro** de **eventos** y demás información del sistema operativo. Por supuesto, este no ha sido el único factor para escogerlo, algunos más se listan a continuación:

1. Lenguaje independiente del sistema operativo que se utilice, ya sea Windows, Linux, MacOS, etc.
2. Lenguaje dinámico con muchas posibilidades de uso.
3. Ofrece librerías para la implementación de servicios en entornos Windows 10.
4. Gran rendimiento.
5. Compatibilidad con entornos web, algo que será tratado en profundidad en el siguiente apartado.

La **configuración de la herramienta** por parte del administrador será a partir de un **fichero**, aunque, debido a que es **necesario** brindar al usuario la posibilidad de visualizar informes anteriores, se tiene que implementar una funcionalidad de **persistencia**. De esta manera, no solo se podrá almacenar información pasada, sino también detectar cambios en la configuración (P.ej. desactivación de análisis externo, generación de informes, etc.). Por lo tanto, se va a utilizar un **SGBD relacional** para este propósito, concretamente, debido a que la base de datos será exclusivamente de uso local en un dispositivo final, se ha escogido **SQLite** [19].

Como se puede ver a continuación, la herramienta interactúa con varias partes del sistema operativo, ya que la detección de procesos y/o programas independientes será heterogénea en la mayoría de los casos.

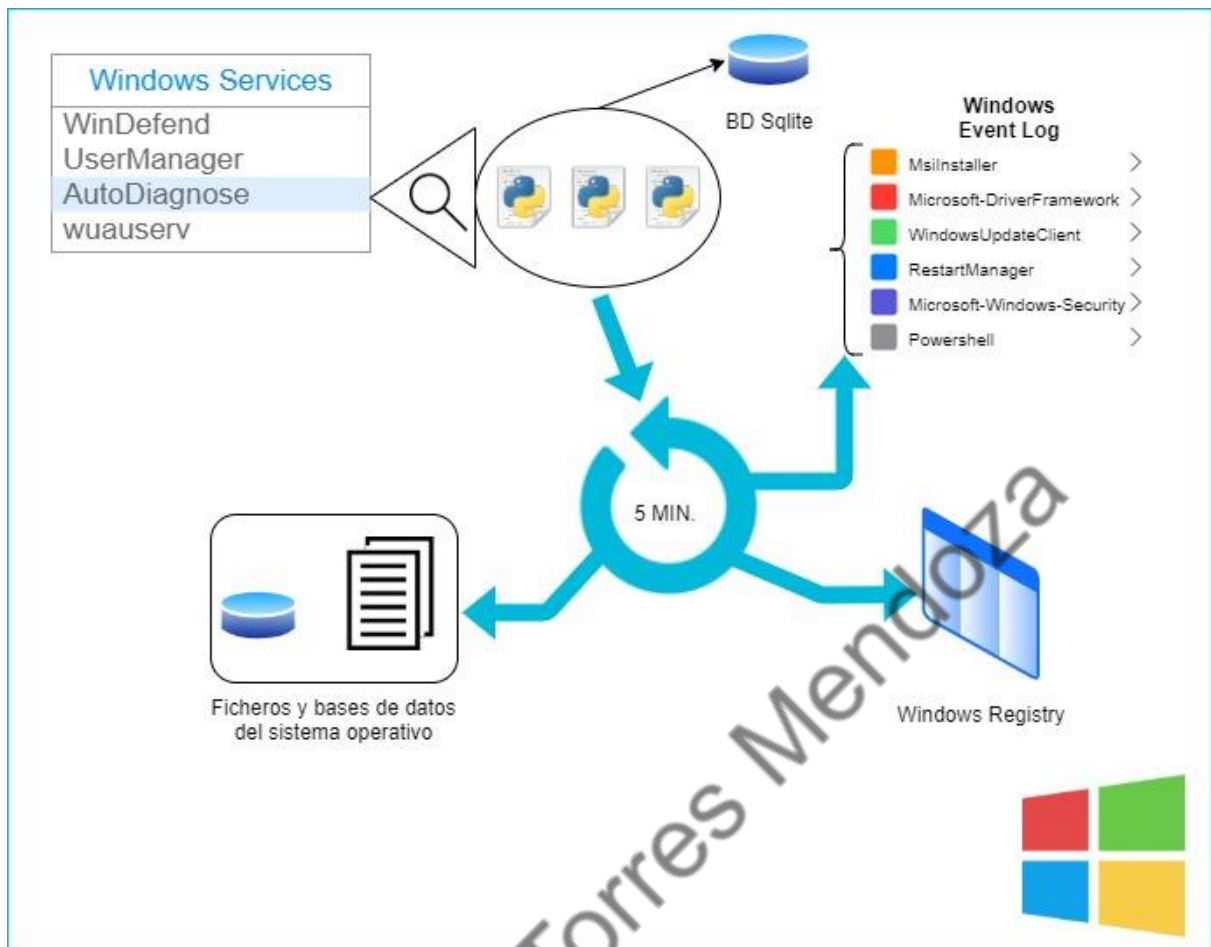


Figura 8. Comportamiento de la herramienta local haciendo uso de diversas tecnologías. Fuente: Elaboración propia.

8.2.2. Agente externo

El agente externo debe de ser un **servidor con acceso a Internet**, ya que las diversas herramientas repartidas por las organizaciones y hogares clientes deben de poder comunicarse con él.

Se debe de tener cuenta una perspectiva de uso de recursos elevada debido a que el cometido del servidor se basa en realizar análisis de vulnerabilidades, de manera periódica, a diversas organizaciones, algo que, según el caso puede ser bastante costoso. Por lo tanto, se ha escogido un sistema operativo **Linux** como entorno de ejecución de este componente.

El ámbito tecnológico de este componente podría tomar dos vertientes:

- **Mediante servicio web.** Esta opción nos permite, no sólo automatizar la gestión de la comunicación entre diferentes clientes, sino también la seguridad de la comunicación mediante el uso de **TLS**. Por otro lado, la integración de una API REST nos puede permitir la compatibilidad con otros sistemas, ya sean propios o externos.
- **Mediante servicio propio.** Esta opción, aunque permite mayor personalización en el comportamiento y comunicación del agente, es claramente más compleja.

Por lo tanto, se ha escogido implementar la funcionalidad del agente externo mediante un servidor web que integre una **API REST**, de tal forma que los clientes puedan intercambiar información con los verbos HTTP de manera sencilla y estándar. Lo cual, nos permite **cumplir** con el requisito no funcional **RNF.2**.

Claramente, se identifica la problemática de escoger un lenguaje de programación que permita su uso en *backend* y tenga librerías para realizar tareas relacionadas con el análisis de vulnerabilidades de los servicios. Al poderse utilizar también **Python** para estos propósitos, la solución a esta se resuelve rápidamente, además, al utilizarse el mismo lenguaje de programación tanto en el cliente como en el servidor, se obtienen ventajas en muchos aspectos. Aunque, lo cierto es que el *backend* del servidor web no debería de ser el encargado de realizar los análisis de vulnerabilidades, ya que la carga de cada hilo de ejecución podría llevar a grandes problemas de rendimiento y disponibilidad. Una mejor opción es la de **asignar** las tareas relacionadas con el **análisis de vulnerabilidades a un programa externo**, también alojado en el servidor, que se encargue de ejecutarlas a una hora concreta y **almacenar** los **resultados** en la **base de datos** común, en la cual el *backend* buscará la información que los clientes pidan mediante la API. Esta última decisión de diseño se ha tomado para intentar **cumplir** con los requisitos no funcionales **RNF.2 (Disponibilidad)** y **RNF.5**.

Por otro lado, como ya se ha comentado, el agente externo deberá de ser capaz de almacenar cierta información de manera persistente, para esto se va a utilizar **SGBD** relacional, cuyo diseño será mostrado en el siguiente apartado.

Como tecnología de servicio web se va a utilizar **Apache** [20], mientras que para el servicio de base de datos **MariaDB** [21].

En resumen, como se puede ver en la siguiente figura, el cliente que ejecute la herramienta contactará con el agente externo para obtener la información del informe de vulnerabilidades. Llegada la hora, el agente externo llevará a cabo los análisis oportunos, ya sea directamente o a través de servicios intermedios. Posteriormente, el cliente obtendrá la información del informe, que será mostrada al usuario por medio de la herramienta.

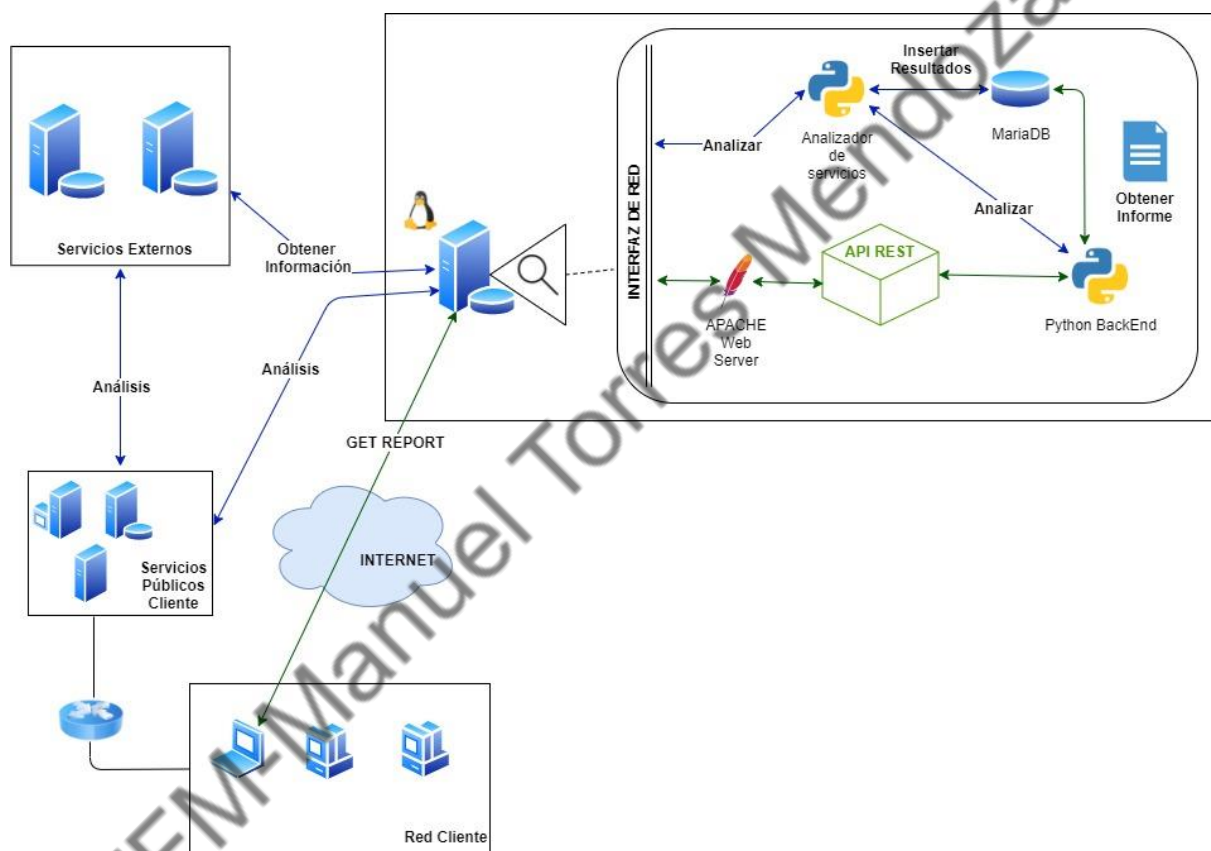


Figura 9. Comportamiento del agente externo haciendo uso de diversas tecnologías. Fuente:
Elaboración propia.

Hay que tener en cuenta un aspecto muy importante, este es el relacionado con la definición de los servicios públicos a analizar pertenecientes a la infraestructura del cliente. Aunque, a primera vista resulta lógico pensar que esta información, que puede ser cambiante en el tiempo, sea introducida de manera dinámica por la herramienta a través de peticiones POST a la API del componente externo, esto puede ser un problema si los

servicios definidos no pertenecen a la infraestructura del cliente. Esto es debido a que, ya sea por error de configuración o intencionalidad, el agente externo realizaría un análisis a una infraestructura ajena a nuestra clientela, por lo que se podrían tomar medidas legales alegando, por ejemplo, un intento de denegación de servicio. Por ello, se ha decidido que la información de los servicios públicos a analizar sea proporcionada y verificada previamente por la organización cliente, por ejemplo, a través de una entrevista previa o trámite telemático.

8.3. Diseño de la persistencia de los datos

Para el diseño de la base de datos, debido a que se va a usar un SGBD relacional tanto en el agente externo como en la herramienta local, se ha optado por emplear un diseño Entidad Relación [22]. A continuación, podemos ver el diagrama inicial, tanto del agente externo, como de la herramienta local.

8.3.1. Diagrama Entidad Relación Agente externo

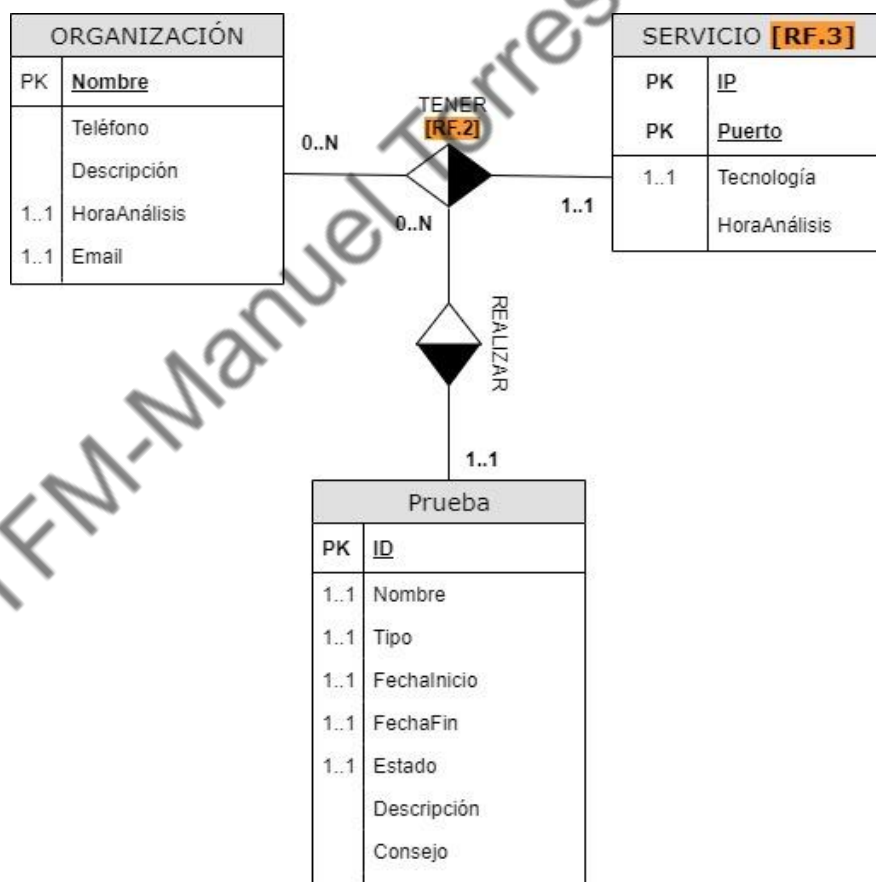


Figura 10. Diagrama Entidad Relación para la base de datos del agente externo. Fuente: Elaboración propia.

Como se puede observar en la anterior figura, el agente externo almacenará información sobre las organizaciones clientes y los servicios públicos que debe de analizar, realizando las pruebas oportunas.

8.3.2. Diagrama Entidad Relación Herramienta local

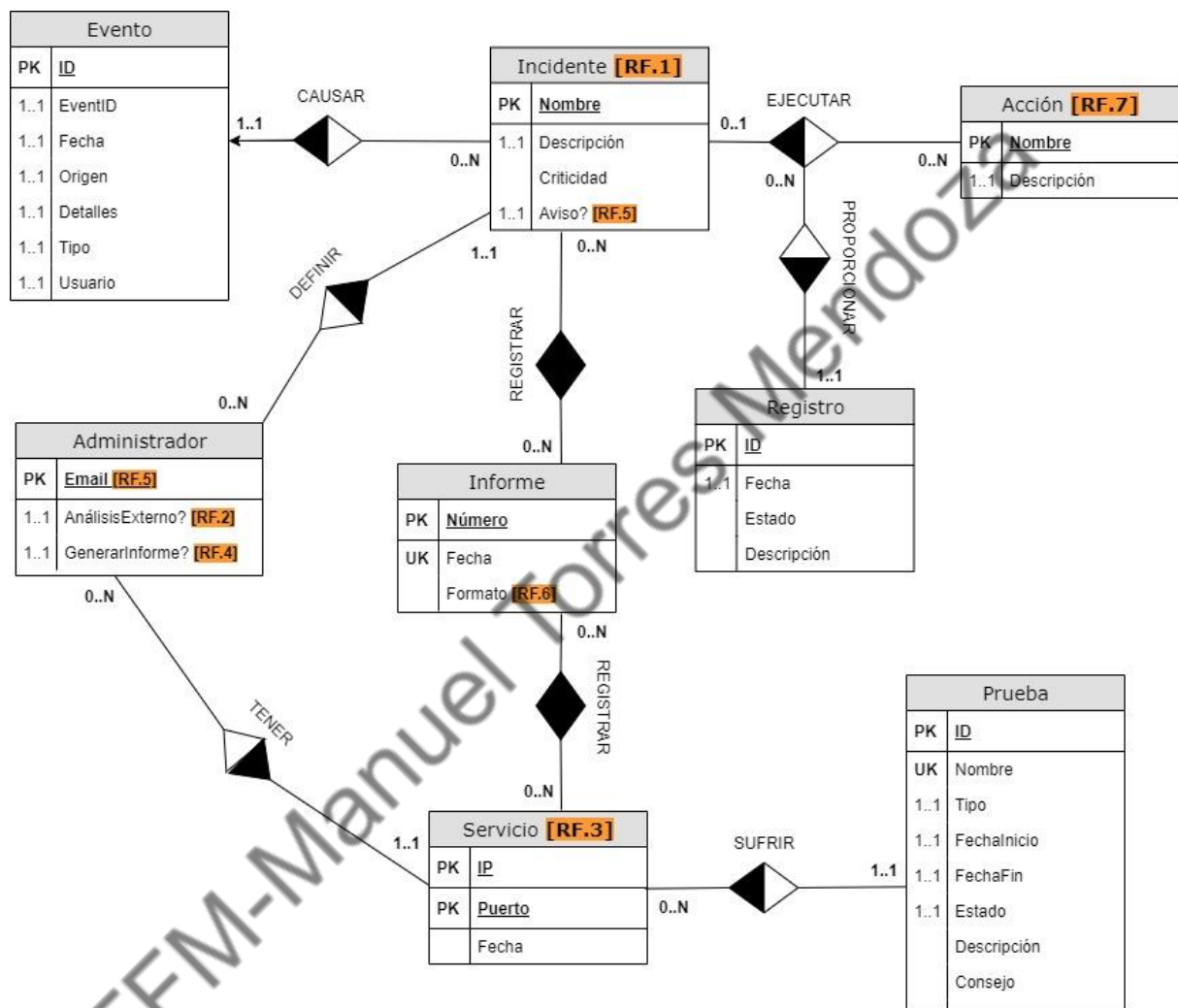


Figura 11. Diagrama Entidad Relación para la base de datos de la herramienta local. Fuente:
Elaboración propia.

Nota: En un primer diseño únicamente habrá una sola entidad Administrador. Además, los incidentes, acciones y servicios serán escogidos por el administrador en el fichero de configuración sobre una lista dada.

Como se puede apreciar en las anteriores figuras, hay **partes del diagrama** que han sido definidas de esa forma **para cumplir con los requisitos** definidos en los apartados anteriores del diseño, por ello se puede ver una referencia a estos.

8.4. Diseño API REST

A continuación, se definen los endpoints con los que trabajará la API del agente externo. Como se puede observar, en esta primera versión no habrá grandes posibilidades de interacción por parte del cliente.

Ruta:	/api/:key/report/:timestamp
Verbo HTTP:	GET
Parámetro:	:key → Clave única asociada a un cliente concreto para el acceso a la API, esta habrá sido proporcionada previamente. [RNF.2] :timestamp → Cadena de números que representa el informe realizado para un día concreto. P.ej. 20200222 se referiría al informe generado el día 22 de febrero de 2020.
Descripción:	Obtener el informe de los análisis realizados sobre los servicios públicos del cliente el día que haga referencia al timestamp.

Ruta:	/api/service/:service/
Verbo HTTP:	PUT
Parámetro:	:service → IP y puerto del servicio codificado en Base 64.
Cuerpo del mensaje:	:key :time → Nueva hora para ejecutar el análisis del servicio en formato 24 horas. P.ej. 1905 se referiría a las 19:05h.
Descripción:	Modificar la hora a la que se quiere realizar el análisis del servicio.

Ruta:	/api/:key/service/:service
Verbo HTTP:	DELETE
Parámetro:	:key :service → IP y puerto del servicio codificado en Base 64.2020.
Descripción:	Eliminar un servicio.

Ruta:	/api/:key/organization/:organization
Verbo HTTP:	DELETE
Parámetro:	:key :organization → Nombre identificativo del cliente.
Descripción:	Eliminar todos los datos identificativos de ese cliente. [RFN.5]

8.5. Pruebas y validación

Por último, el diseño de las pruebas que se realizarán para verificar el correcto funcionamiento de los requisitos del software constará de tres partes:

- **Herramienta local.** Estas pruebas irán enfocadas a validar el comportamiento de la herramienta que se ejecute localmente en el sistema del usuario.

Principalmente, se tratará de verificar la **detección de los incidentes de seguridad** definidos por el usuario Administrador, **forzando la ejecución** de estos. Por ejemplo, si un incidente de seguridad definido es el de desactivar el antivirus, se llevará a cabo esta acción y se comprobará si la herramienta lo detecta. En caso negativo, se comprobará el visor de eventos para ver el caso concreto que se ha pasado por alto.

Por otro lado, se deberá de probar si, tanto la **generación de informes** como **envío de avisos** funciona al nivel esperado.

De manera secundaria, se deberán de realizar **pruebas de carga** para validar que la ejecución de la herramienta no compromete el rendimiento del equipo, teniendo en cuenta el **RNF.5**.

- **Agente externo.** Estas pruebas irán enfocadas a validar el comportamiento del servidor externo.

De igual forma que en el caso de la detección de eventos, se realizarán **pruebas** para cada uno de los **servicios** a los que se vaya a **analizar**. Una vez que el agente externo extraiga la información del análisis de un servicio, se comprobará con la salida que proporcionen otras herramientas utilizadas para este caso, por ejemplo, en el caso de que se realice un escaneo de puertos, se comprobará la salida con una efectuada por la herramienta NMAP [23].

La **generación y envío de información** deberá de ser puesta a prueba a través de probar los casos de uso de la API REST con los verbos HTTP que correspondan.

Por último, también se deberá de llevar a cabo alguna **prueba de carga** para intentar cumplir con **RNF.2 (Disponibilidad)**.

- **Validación conjunta.** Estas pruebas irán enfocadas a validar el comportamiento del software en su totalidad, de forma que se ejecuten de forma simultánea tanto la herramienta local como el agente externo.

En este caso, se llevarán a cabo las **pruebas de persistencia** en fechas que no correspondan a la actual.

Por otro lado, la **sincronización** de los dos componentes será un factor crítico debido al uso de una red que cuenta con retardos potenciales como Internet. Esta validación será emulada por medio de dos equipos diferentes en una LAN.

TFM-Manuel Torres Mendoza

9. Implementación

En esta sección se detallan las evidencias de la implementación que se ha llevado a cabo para desarrollar los componentes software definidos en los apartados anteriores. Debido a la metodología *Waterfall* que se ha seguido, la implementación ha tenido un orden secuencial, implementando, en primer lugar, la herramienta local, y en segundo, el agente externo, cada uno con su respectivo entorno, como se detalla a continuación.

9.1. Herramienta local

9.1.1. Preparación de entorno

Se ha utilizado un equipo Microsoft **Windows 10** Education x64 con CPU AMD FX-6350 a 3.9 GHz y 8 GB de RAM DDR4.

El IDE utilizado es **PyCharm Community Edition 2019** [24] con su configuración básica. El intérprete de **Python** utilizado es la versión **3.7**.

Para la implementación de la persistencia se ha utilizado el paquete **SQLite 3** [25] de Python, en conjunto con **DB Browser for SQLite 3.11.2** [26] para la visualización de los resultados de manera más directa.

Se debe de destacar el papel de la librería **PyWin32** [27], ya que, en un principio se necesitaban ciertos funcionamientos propios de sistemas Windows, por ejemplo, integrar un comportamiento de servicio, acceso al registro de eventos y Windows Registry. Para todas estas acciones PyWin ha ofrecido una gran facilitación.

9.1.2. Servicio de Windows

Como se puede observar en el **Anexo 1**, se necesita que el organismo central de la herramienta implemente ciertos métodos de la librería que hereda (`win32serviceutil`) para funcionar como un servicio. Si nos fijamos, estos se centran en el envío de señales típicas de apagado y encendido del servicio. Un **factor diferencial** será **definir cada cuanto tiempo se va a volver a ejecutar el servicio**, ya que puede **derivar en que la alerta sea exitosa o tardía**. Aunque, en un primer momento se pensó en la ejecución cada cinco minutos, tras unas pruebas unitarias, se ha comprobado que bajar este número no afectada en gran medida al rendimiento, por lo que **se ha seguido el siguiente enfoque**:

- Cada detección (P.ej. inserciones de dispositivos externos en la máquina), buscará unos registros concretos, ya sean eventos o datos de aplicación. Debido a la ejecución secuencial de la herramienta, si una comprobación dura mucho tiempo pueden darse cuellos de botella que causen, incluso, que las siguientes detecciones bajen la ratio de éxito.
- Por lo tanto, cada detección almacena la hora a la que se ha ejecutado por última vez, comprobándola con la hora de la ejecución actual, centrándose en detectar únicamente los eventos que pertenezcan a esa franja de tiempo.
- En la primera ejecución se detectarán eventos que hayan ocurrido tres minutos antes.
- Tras la finalización de la iteración (fin de todas las comprobaciones), se esperará un minuto para volver a empezar.

Es importante diferenciar entre la **importancia de tener un servicio de Windows** y un proceso en segundo plano, ya que, tanto para personal técnico como no técnico, resulta mucho más **fácil interactuar** con servicios a través de la funcionalidad *Services.msc* de Windows.

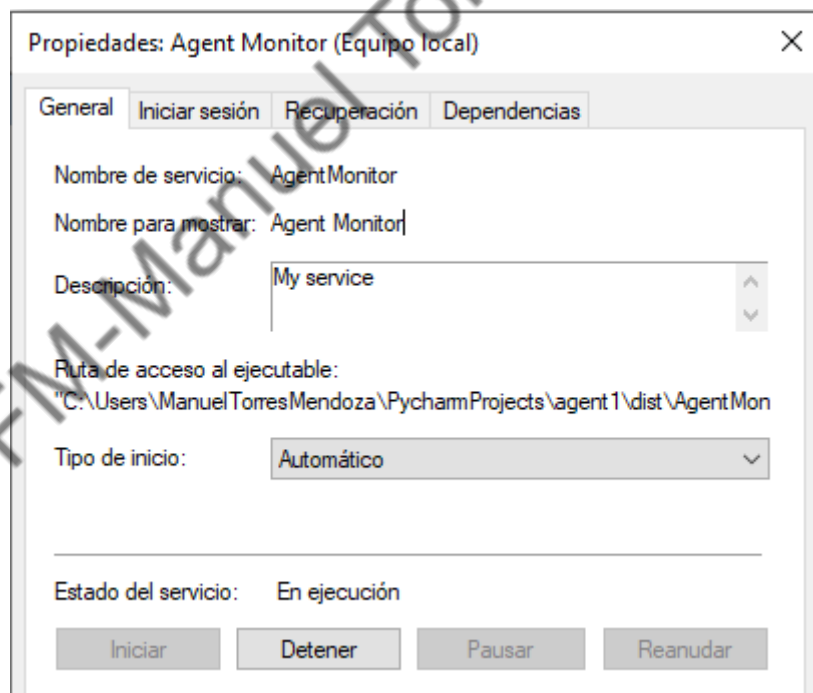


Figura 12. Ventana de servicio a través de Services.msc de Windows.

9.1.3. Identificación y captura de eventos

Como ya se ha comentado en este documento, Windows tiene un registro central de eventos gestionado por el servicio Windows Logging. Dentro de este, se identifican las siguientes categorías:

- **Eventos de Windows.** Estos son los principales del sistema, la mayoría de los eventos que influyen directamente sobre este vendrán reflejados aquí. Se dividen en los siguientes tipos:
 - **Aplicación.** Recopilan eventos sobre las aplicaciones reconocidas de Windows y las **instalaciones de software de aplicación.**
 - **Sistema.** Actualizaciones de sistema, eventos del kernel y hardware, además de servicios estrechamente relacionados con el funcionamiento del sistema (P.ej. DHCP, DNS).
 - **Seguridad.** Recopila acceso por parte de los usuarios a ficheros críticos del sistema y/o credenciales asociados a perfiles.
- **Eventos de Servicio y Aplicaciones.** En este tipo de eventos encontramos una gran cantidad de subcategorías, pero la mayoría asociadas a software concreto.

Lo cierto es que **no todo el software tiene su categoría de evento asociado**, sino que, algunos, lo redirigen a ciertos ficheros, que pueden estar o no, definidos en el Windows Registry (P.ej. Avast Antivirus).

Pese a esto, se identifican varios eventos interesantes asociados con los objetivos del proyecto, los cuales se muestran a continuación:

<u>Categoría del evento</u>	<u>Descripción</u>
Aplicación/MsiInstaller	Instalaciones de editores reconocidos por Windows
Aplicación/Microsoft-Windows-RestartManager	Instalaciones en general
Aplicación/Microsoft-Windows-User Profiles Services	Suele aparecer en instalaciones de editores no reconocidos por Windows
Sistema/Microsoft-Windows-WindowsUpdateClient	Actualizaciones del sistema.
Microsoft-Windows-Windows Firewall With Advanced Security/Firewall	Eventos relacionados con los firewalls de Windows
Microsoft-Windows-DriverFrameworks-Usermode/Operational	Inserciones de dispositivos externos
Microsoft-Windows-WindowsUpdateClient/Operational	Errores en actualizaciones
Microsoft-Windows-Sysmon/Operational	Conexiones de red

Tabla 6. Categorías de eventos interesantes para la detección de posibles riesgos de seguridad.

Fuente: Elaboración propia.

Otro aspecto que destacar es la **representación de estos eventos por parte de Windows**, ya que estos son mostrados tanto, mediante objetos compatibles con Active Directory (AD), como XML, siguiendo los esquemas de representación de eventos de Windows [28]. Un ejemplo de la estructura que siguen se puede consultar en el **Anexo 2**.

En general, se pueden diferenciar **dos secciones** interesantes **del evento**, la primera es **System**, la cual nos proporciona información general del evento relacionada con su identificación (P.ej. EventID, Version, Level, etc.) y hora del suceso. Por otro lado, la segunda es **EventData**, esta contiene información descriptiva del evento concreto, por lo que puede contener o no datos. Lo cierto es que el **contenido de estos campos no** es para nada **uniforme** entre los distintos tipos de eventos, sobre todo entre los que pertenecen a la categoría de Eventos de Windows y Eventos de Servicios y Aplicaciones, donde incluso las horas del suceso pueden tener formato diferente (UTC u Hora local).

Sumado a esto, las **características que identifican a un evento como interesante o no son específicas de este**. Por ejemplo, en el caso de los eventos que recogen inserciones de dispositivos externos, se necesita validar que tengan un Level igual a 5 y un OpCode igual a 1, mientras que, para los eventos de instalaciones, además del identificador, también se debe de comprobar que lleven consigo cierta cadena de texto. Incluso, para detectar situaciones específicas, puede ser imprescindible **analizar la dependencia entre eventos**. Obviamente, también deben de entrar en el intervalo de tiempo que nos interesa (P.ej. Últimos 3 minutos). Los casos concretos de cada situación a detectar se detallarán en el apartado de Pruebas y validación.

Debido a todo esto, la mejor solución que se ha encontrado es la de centrar la **implementación de la detección de eventos** en un **paradigma orientado a objetos**, donde cada uno de estos representará un tipo de evento que podrá comprobar si es válido o no de manera independiente, centralizando comportamiento común en una superclase. De esta manera, se podrá **introducir reflexión en el código** para que este sea mucho más extensible y limpio. Un ejemplo de esto se puede observar en el **Anexo 3**.

Por último, como se ha podido ver en el Anexo 3, se destaca que se han tenido que utilizar dos métodos distintos para buscar Eventos de Windows (searchNormalEvents) y Eventos de Servicios y Aplicaciones (searchEvents), esto es debido a particularidades de la librería.

9.1.4. Extracción de información del Windows Registry

Muchas situaciones peligrosas que pueden ocurrir en el sistema **no son detectables a través del registro de eventos de Windows**, ya que, o bien el componente software no emite eventos, o directamente no se encuentra activo. Este es un caso para tener en cuenta, ya que, por ejemplo, se puede detectar que se ha desactivado el Firewall a través del evento que se genera cuando el suceso ocurre, pero este solo se emite una vez. Por lo tanto, para determinar el estado de **servicios críticos del sistema**, como **wuauserv** (Actualizaciones) o **mpssvserv** (Firewall) se debe consultar el Windows Registry.

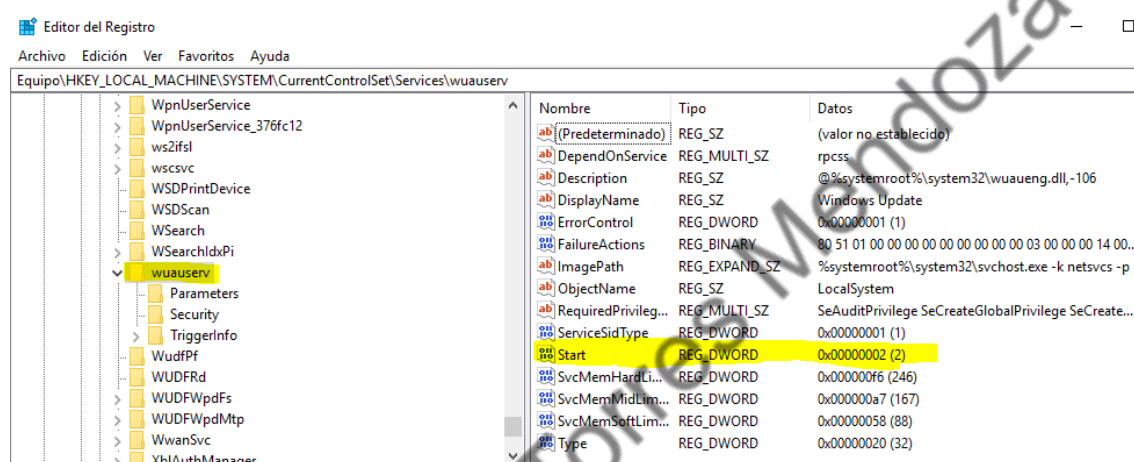


Figura 13. Datos del Windows Registry en relación con el servicio Firewall. Fuente: Elaboración propia.

Para obtener información relevante sobre los servicios críticos de Windows nombrados anteriormente, se ha implementado la sección de código que se puede observar en el **Anexo 4**.

Es importante destacar que **otra opción sería listar los servicios actualmente activos**, pero esto no es factible de cara a los servicios wuauserv y mpssvserv, debido a que, pese a no estar desactivados, cambian a estado de parada en determinadas ocasiones, lo que podría emitir falsos positivos. Por otro lado, para servicios que siempre deben de estar activos, por ejemplo, un antivirus, sí que se ha seguido este enfoque, como se expone en el siguiente apartado.

Por otro lado, a través del Windows Registry, se puede obtener información sobre el **software instalado**, eso sí, de manera dispersa, ya que puede estar recogido en diversas rutas del registro (P.ej. SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall o SOFTWARE\WOW6432Node\Microsoft\Windows\CurrentVersion\Uninstall). Esta funcionalidad se ha utilizado para implementar una lista de software instalado en el sistema que mejore la detección de instalaciones, ya que, de otra manera, hay **instalaciones** que provienen **de editores no reconocidos** y que utilizan **instaladores propios** que **no son detectados fácilmente por el registro de eventos**. Por lo que, de manera periódica, se actualizará la lista de software, si se detecta una instalación se iniciará el análisis de eventos para detectar el tipo de instalación, en función de si ha sido:

1. A través de un editor reconocido de Windows.
2. A través de un editor no reconocido de Windows.
3. A través del instalador de Windows.
4. A través de un instalador propio del editor.

9.1.5. Análisis del estado del Antivirus

El caso del antivirus es diferente, ya que, para el análisis íntegro de este se han tenido que hacer varias comprobaciones.

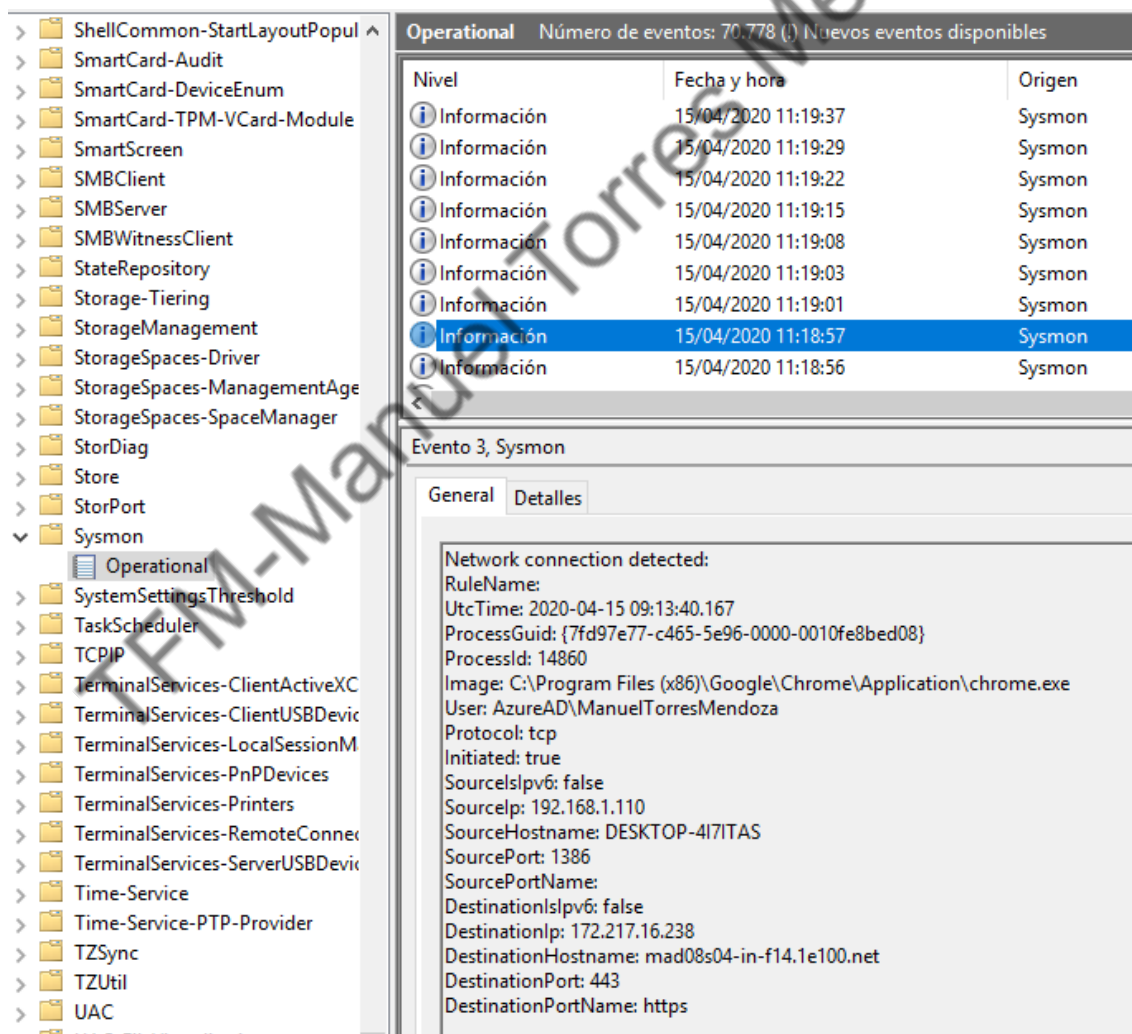
1. En primer lugar, se hace uso del Windows Registry para determinar si se encuentra instalado, comprobando que su descripción concuerde con las conocidas por los proveedores compatibles (P.ej. Avast Antivirus, Norton, etc.).
2. Se obtiene, si se puede, la versión del software (También del Windows Registry) y se realiza una petición HTTP a un sitio web para obtener la última versión estable de este.
3. Se comparan las dos versiones y se avisa únicamente si se tiene una versión inferior a la última estable.
4. Se comprueba que el servicio asociado al antivirus detectado figure como servicio activo en la máquina.

El código para realizar esta funcionalidad se adjunta en el Anexo 5.

9.1.6. Identificación y análisis de conexiones

A través del servicio de Microsoft Windows **Sysmon** (System Monitoring) [29], se detectan todas las conexiones entrantes y salientes del equipo, mostrándose en el registro de eventos de Windows. Este tipo de eventos son muy interesantes, ya que recogen conexiones pertenecientes a la capa tres del modelo OSI, por lo que será independiente del protocolo de aplicación que se utilice.

Por lo que, se ha **implementado una funcionalidad que recoge las conexiones que se van realizando**, tanto externas, como internas, lo cual puede ser de gran interés para que el experto en ciberseguridad, según la política de la organización, decida si se está realizando un comportamiento poco ético del equipo.



The screenshot displays the Windows Event Viewer interface. On the left, the 'Sysmon' folder is expanded under 'Operational' events. The main pane shows a list of events with columns for 'Nivel', 'Fecha y hora', and 'Origen'. The selected event is 'Evento 3, Sysmon'.

Nivel	Fecha y hora	Origen
Información	15/04/2020 11:19:37	Sysmon
Información	15/04/2020 11:19:29	Sysmon
Información	15/04/2020 11:19:22	Sysmon
Información	15/04/2020 11:19:15	Sysmon
Información	15/04/2020 11:19:08	Sysmon
Información	15/04/2020 11:19:03	Sysmon
Información	15/04/2020 11:19:01	Sysmon
Información	15/04/2020 11:18:57	Sysmon
Información	15/04/2020 11:18:56	Sysmon

Evento 3, Sysmon

General Detalles

Network connection detected:
RuleName:
UtcTime: 2020-04-15 09:13:40.167
ProcessGuid: {7fd97e77-c465-5e96-0000-0010fe8bed08}
ProcessId: 14860
Image: C:\Program Files (x86)\Google\Chrome\Application\chrome.exe
User: AzureAD\ManuelTorresMendoza
Protocol: tcp
Initiated: true
SourceIsIpv6: false
SourceIp: 192.168.1.110
SourceHostname: DESKTOP-4I7ITAS
SourcePort: 1386
SourcePortName:
DestinationIsIpv6: false
DestinationIp: 172.217.16.238
DestinationHostname: mad08s04-in-f14.1e100.net
DestinationPort: 443
DestinationPortName: https

Figura 14. Información del registro de eventos de Windows, concretamente, para Sysmon. Fuente:

Elaboración propia.

Como se puede observar en la figura anterior, dado que en una gran cantidad de ocasiones el dominio de la IP se resuelve en el propio evento, es muy factible la **implementación de una funcionalidad** que, **dado el dominio, resuelva si es posible que este haya sido creado mediante técnicas DGA** (*Domain Generation Algorithm*). Esta técnica se basa en generar dominios de forma automatizada para intentar evadir la detección de firewalls de red y RPZ (*Response Policy Zone*) de los servicios DNS, ya que los equipos asociados a estos dominios suelen ser contenedores de malware usados para fines poco éticos (P.ej. contenedores de Ransomware).

Existe tanto un paquete en Python como una web para realizar consultas relacionadas con el porcentaje DGA de un dominio determinado. Es importante destacar que, pese a que hay una gran cantidad de listas de dominios DGA que se van actualizando continuamente [30], en ocasiones, son indetectables, ya que van cambiando muy rápidamente.

Se destaca que, tanto las IP como los dominios a los que el equipo se va conectando se almacenan, pero únicamente se comprueba su nivel DGA una vez al día, ya que se entiende que es muy poco probable que se produzca un falso negativo. Por otro lado, realizar comprobaciones DGA de manera continua podría poner en riesgo el rendimiento de la herramienta, debido, en gran medida, a las resoluciones DNS necesarias para resolver el dominio.

9.1.7. Acciones de respuesta

Se han implementado tres acciones de respuesta a determinados comportamientos encontrados por las detecciones:

- Apagado del equipo.
- Deshabilitación de los adaptadores de red.
- Mostrar ventana de aviso.

La mayoría se llevan a cabo a través de sencillos comandos CMD y gracias a los altos privilegios de la herramienta, pero, debido a la imposibilidad de que un servicio de Windows muestre una ventana gráfica, se ha tenido que independizar el comportamiento que ejecuta las acciones de respuesta en un proceso independiente. El servicio se comunicará con este cuando vea necesaria la ejecución de una acción de respuesta. Esto se lleva a cabo mediante **IPC** (*Interprocess Communication*) basada en tuberías.

9.1.8. Persistencia y Trazabilidad

Tal y como se especificó en la etapa de diseño, la herramienta local debe de tener una funcionalidad que le proporcione persistencia, ya sea para permitir el buen funcionamiento de las detecciones, o para permitir la generación de informes en una determinada fecha.

Esta funcionalidad se ha implementado a través del paquete SQLITE 3 de Python, creando una base de datos en el equipo local.

Por otro lado, debido a la poca legibilidad de los errores de Microsoft Windows, resulta esencial la creación de un log que vaya registrando las acciones que va realizando la herramienta.

9.1.9. Generación de informes

En base a su configuración inicial, la herramienta genera un informe diario con las detecciones nombradas con anterioridad, a través de generar una tarea programada de Windows a la hora que se haya definido. La tarea programada se basa en llamar a un script Python que se encarga de generar el informe del día actual y formatearlo en HTML.

9.1.10. Presentación al usuario

Al final, la herramienta se podrá configurar por medio de una ventana gráfica que ofrecerá una lista de detecciones, donde el usuario podrá elegir si activa la detección de manera total, parcial o no la activa. Además, este podrá elegir entre una lista de acciones de respuesta para determinados comportamientos.

9.2. Agente externo

9.2.1. Preparación del entorno

Se ha utilizado un equipo **Ubuntu** 20.04 (64 bits) con CPU Intel i7-7500U 2.7 GHz y 16 GB de RAM DDR4.

El IDE de desarrollo es similar al de la herramienta local, con **Pycharm Community Edition 2019** y **Python 3.7** como intérprete.

Para el desarrollo de la API se ha utilizado el Framework **Flask** [31], mientras que para la persistencia se ha utilizado el servicio **MariaDB** 10.4 [21].

9.2.2. API

La implementación de los métodos de la API sigue las indicaciones que se definieron en el apartado de diseño, donde se destacan los siguientes puntos:

- Se ha implementado una sanitización de las entradas de los métodos HTTP para prevenir inyecciones de código, ya sea SQL o comandos del operativo.
- Se ha seguido un paradigma orientado a objetos, ya que, en mi opinión resulta más eficiente para realizar acceso a datos.
- Se utiliza una API Key independiente por cliente para realizar la autenticación. Resulta de gran importancia definir un método con la suficiente entropía para generar las claves que se proporcionarán a los clientes. En mi caso, he utilizado el mecanismo de **Generación de bits Aleatorios basado en la Degradación Nuclear de Caesium-137 Barium** [33], ya que, es el método que mayor aleatoriedad proporciona actualmente.

Nota: Este NO es un servicio público, pero he conseguido acceso tras ponerme en contacto con la organización.

- También se han implementado otros métodos relacionados con la prevención del apagado de la herramienta en el equipo del cliente, los cuales serán explicados posteriormente en este documento.

Algunas secciones de la implementación de la API pueden ser consultadas en el **Anexo 6**.

9.2.3. Analyzer

Este componente software será el encargado de realizar las pruebas a los servicios públicos que haya definido la organización cliente. Actualmente, realiza tareas relacionadas con el descubrimiento de amenazas a bajo nivel, las cuales, se muestran a continuación:

1. Descubrimiento de puertos abiertos y/o cerrados no conocidos por el cliente a través de escaneo de puertos.
2. Obtención de información asociada a los puertos abiertos mediante la técnica de Banner Grabbing, centrándose en recuperar versiones de software y distribución.
3. Análisis de las cabeceras del servidor web (Si se tuviese).
 - 3.1. Validar que HTTPS esté activado.
 - 3.2. Validar que se realiza redirección del puerto 80 al 443.
 - 3.3. Validar el uso de cookies seguras (HttpOnly) si se utilizan.
 - 3.4. Presencia de cabeceras de seguridad que puedan ocasionar descubrimiento de vulnerabilidades como E-Tag y X-Powered-By.
 - 3.5. Contacto con servicio externo SecurityHeaders [34] para que este realice un análisis en mayor nivel de las cabeceras. Después, se obtienen las partes interesantes de este, por ejemplo, riesgos medios, altos y consejos, y se presentarán al usuario.

Tras la realización de cada prueba se almacenan los resultados en la base de datos.

Un aspecto que destacar es que este componente se ha desarrollado con la **idea de ser extensible**, pudiendo ir añadiendo más pruebas poco a poco. Esto se tratará en el apartado de Líneas Futuras.

La implementación del Analyzer se puede observar en el **Anexo 7**.

9.2.4. Integración con ejecución periódica (Watcher)

Al no tener una hora fija para realizar las pruebas a los servicios de los usuarios y dejarlos elegir la que más le conviene, surge una **problemática centrada en cómo gestionar la programación y ejecución de esos análisis**, teniendo en cuenta los requisitos de rendimientos y escalabilidad.

Aunque, **en un primer momento se pensó en el servicio Crontab** [34], esta opción se descartó debido a que **este no está pensado para desarrollar tareas pesadas** que deben de ser bastante síncronas. Por ello, **se ha implementado otro componente** de software, llamado **Watcher.py** como servicio del sistema de inicio Systemd, el cual, cada vez que se llame obtendrá la prueba más cercana a ejecutar y esperará hasta esta hora. Por otro lado, si se produce una actualización en las pruebas de los servicios, este se reiniciará y volverá a obtener el más cercano.

Evidentemente, este será un proceso crítico que no deberá pararse de manera continua mucho tiempo, por este motivo se ha implementado como un servicio, además de que la interacción con el resulta más fácil.

Se puede observar en el **Anexo 8** como se ha implementado.

9.2.5. Envío de alertas e informes

Aunque en este apartado hay muchas posibilidades, donde se destaca envío de SMS, llamada telefónica o mensajería vía Telegram, al final **se ha optado por el envío de mensajes de correo electrónico**, ya que es la opción más común y utilizada en el ámbito empresarial. La implementación, aunque se podría haber configurado mediante un servidor propio de correo, se ha configurado el SMTP de Google para utilizar una cuenta de correo electrónico de este proveedor, no necesitando de procesamiento extra que ralentice el servidor propio.

De esta forma, el usuario administrador puede recibir los siguientes mensajes:

- Informes diarios de los equipos locales donde se esté ejecutando la herramienta.
- Informes diarios de los servicios públicos configurados.
- Alertas que sucedan en los equipos locales donde se esté ejecutando la herramienta.

Se destaca que la primera vez que la herramienta local se inicie en la máquina, es decir, cuando se instale, no se mandará un mensaje de alerta al administrador, aunque haya sido en una hora prohibida. Esto es debido a que la instalación puede haberse realizado en una hora anormal para dejar el software preparado.

Los informes podrán ser generados y obtenidos a través de una petición GET a un endpoint de la API del Agente Externo, de esta manera, habrá una **compatibilidad con otros tipos de software** asociados a monitorización (P.ej. NOC o SOC).

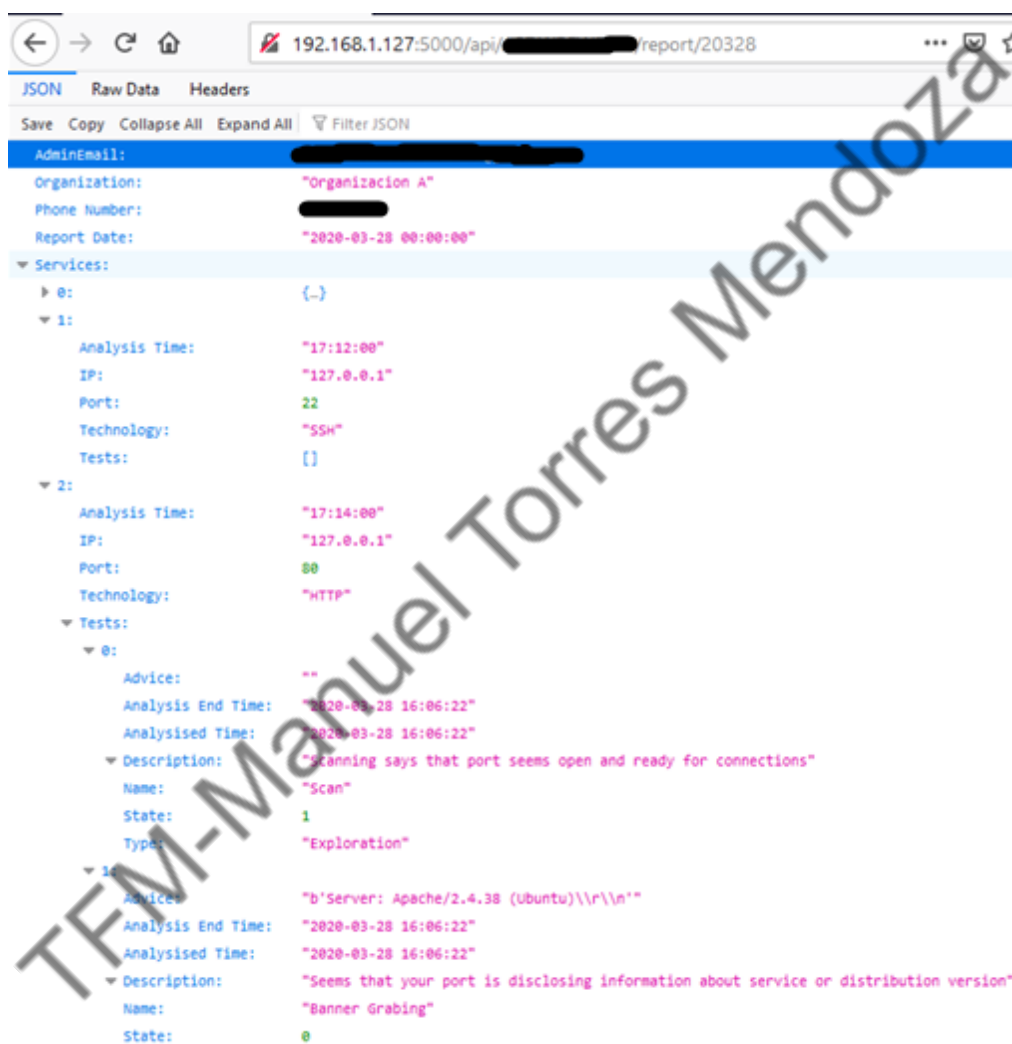


Figura 15. Petición a la API del Agente Externo para obtener el informe. Fuente: Elaboración propia.

9.2.6. Tratamiento de señales

Ante **la posibilidad de que un usuario del equipo** donde se esté ejecutando **la herramienta**, con suficientes privilegios, **interrumpa la ejecución de esta de manera premeditada**, se ha introducido un mecanismo de comprobación del estado de la herramienta de manera periódica.

Al iniciarse, la herramienta enviará una señal de inicio a un endpoint de la API del Agente Externo, este lo tratará como una prueba a servicio más, siendo compatible con el Watcher explicado anteriormente. Antes de que pasen los próximos 29 minutos, la herramienta deberá de mandar una señal Keep Alive para que el contador se reestablezca, en caso contrario, se enviará una alerta al administrador. Por último, cuando la herramienta pare, es decir, se llame al método Stop típico de los servicios, se mandará una señal de parada, de igual forma, a la API del Agente Externo.

Ahora bien, la política de seguridad del cliente puede variar en función de la hora a la que se registra un apagado o encendido, por lo que, a grandes rasgos, se pueden diferenciar las siguientes situaciones que resumen el comportamiento de esta funcionalidad.

Teniendo en cuenta que un cliente tiene definida una hora de inicio y fin de actividad:

- Si se inicia la herramienta antes de la hora de inicio o después de la hora de fin se mandará una alerta, ya que se está haciendo uso del equipo en una hora prohibida.
- Si se para la herramienta entre la hora de inicio y la hora de fin se enviará un aviso, ya que un usuario puede haber parado la herramienta adrede en horario laboral.
- Si se manda una señal Keep Alive antes de la hora de inicio o después de la hora de fin se mandará otra alerta al administrador, ya que se sigue utilizando el equipo en una hora prohibida.

Es importante destacar que la herramienta que se ejecutará en los equipos locales mandará una señal de actualización a los 20 minutos en lugar de los 29, por el hecho de tener un poco más de margen ante posibles adversidades.

10. Pruebas y validación

Una vez finalizada la implementación del software, se deben de realizar pruebas de manera exhaustiva sobre las funcionalidades que ofrece, no solo para asegurar que funcionan de una manera correcta, sino también para llegar a la conclusión de si sirve para resolver el problema para el que fue concebido.

Al igual que en la implementación, en primer lugar, se han efectuado las pruebas sobre la herramienta local, centrándose en las detecciones de situaciones poco seguras. En segundo lugar, se validan las funcionalidades relacionadas con la prueba de servicios públicos definidos en el Agente externo, mientras que, por último, se han realizado unas pruebas de integración conjuntas.

10.1. Herramienta local

Las pruebas para las distintas detecciones implementadas en la herramienta local han sido realizadas tanto de manera individual, es decir, una por detección, como de manera conjunta. Pero, para tratar de resumir, en esta sección se tratarán casos concretos, donde, en base a una configuración de la herramienta, se realizarán una serie de acciones en la máquina y se comparará si el resultado es el esperado.

10.1.1.Caso 1

Configuración de la herramienta		
Detección	Estado	Acción de respuesta
Dispositivos externos	Activado	Ninguna
Instalaciones	Activado	Ninguna
Firewall	Activado	Ninguna
Conexiones remotas	Activado	Ninguna
Antivirus	Activado	Ninguna
Actualizaciones	Activado	Ninguna
Generación de informe	21:05h.	-

Tabla 7. Configuración inicial de la herramienta. Fuente: Elaboración propia.

Acciones realizadas		
Hora	Descripción	Resultado esperado
20:51h	Se activa la herramienta en la máquina 192.168.1.110	Envío de informe diario a las 21:05h.
20:52h	Se desactiva el firewall de red privada mediante la característica Panel de control\Sistema y seguridad\Firewall de Windows Defender	Envío de alerta y adición en el informe diario.
20:52h	Se navega a la web de la Universidad de Alicante.	Adición en el informe diario como conexión NO dga.
20:53h	Se inserta un USB a la máquina	Envío de alerta y adición en el informe diario.
20:56h	Se deshabilita el servicio mpssvc (Firewall) del registro de Windows	Envío de alerta y adición en el informe diario.
21:00h	Se deshabilita el servicio wuauserv (Actualizaciones) del registro de Windows	Envío de alerta y adición en el informe diario.
21:00h	Se instala un software de un editor no reconocido	Envío de alerta y adición en el informe diario.

Tabla 8. Actividades realizadas en la máquina y su resultado esperado. Fuente: Elaboración propia.


RESULTADOS

20:54h	Se recibe alerta en relación con un dispositivo externo conectado en la máquina 192.168.1.110.
20:54h	Se recibe alerta en relación con que el firewall de red interna ha sido desconectado mediante GUI por el usuario ManuelTorresMendoza en la máquina 192.168.1.110.
20:56h.	Se recibe alerta en relación con que el servicio de firewall ha sido desactivado en la máquina 192.168.1.110.

21:00h	Se recibe alerta en relación con que el servicio de actualizaciones ha sido desactivado en la máquina 192.168.1.110.
21:00h	Se recibe alerta en relación con que se ha instalado 1 nuevo software en la máquina 192.168.1.110.
21:05h	Se recibe el informe diario de la máquina 192.168.1.110 con todos los eventos esperados. También se muestra una conexión a la IP 192.145.235.30 que corresponde con la conexión a www.ua.es (Aunque la query reversa de la IP apunta a otro dominio por causas ajenas).

Tabla 9. Resultados obtenidos tras la ejecución de las actividades. Fuente: Elaboración propia.

Tanto una de las alertas recibidas en el email del administrador, como el informe diario generado se muestra a continuación:



Incidente	Fecha	Evento	Criticidad
Dispositivos externo conectado	2020-04-20T20:53:19.167835200Z	Se ha producido una insercion de un dispositivo externo	9
Instalaciones detectadas	2020-04-20T21:00:38	1 nuevos programas instalados. La descripción de cada uno se muestra en los siguientes eventos.	5
Instalacion de un editor desconocido detectada	2020-04-20T21:00:38	Sin información del software	5
Las actualizaciones automáticas han sido desactivadas	2020-04-20 21:00:42.650919	Las actualizaciones automaticas han sido desactivadas	8
El servicio firewall se encuentra desactivado	2020-04-20 20:56:33.061251	El servicio firewall esta desactivado	9
Firewall desactivado por el usuario	2020-04-20T20:52:28.894311300Z	Se ha desconectado el firewall Internal Network Firewall mediante la GUI por el usuario 'azuread\\ManuelTorresMendoza'	9
Conexión externa	2020-04-20 20:52:19.947484	Ip: 192.145.235.30. Dominio: mail.dedicated1856.com	2

Figura 16. Alerta recibida en el correo del administrador e informe generado por la herramienta.

Fuente: Elaboración propia.

10.1.2.Caso 2

Configuración de la herramienta		
Detección	Estado	Acción de respuesta
Dispositivos externos	Desactivado	Ninguna
Instalaciones	Activado	Mostrar Aviso
Firewall	Desactivado	Ninguna
Conexiones remotas	Activado	Apagar PC
Antivirus	Activado	Apagar Red
Actualizaciones	Desactivado	Ninguna
Generación de informe	21:05h.	-

Tabla 10. Configuración inicial de la herramienta. Fuente: Elaboración propia.

Acciones realizadas		
Hora	Descripción	Resultado esperado
20:44h	Se activa la herramienta en la máquina 192.168.1.110	Envío de informe diario a las 21:05h.
20:45h	Se inserta un USB a la máquina	Ninguno.
20:46h	Se desactiva el firewall	Ninguno.
20:46h	Se instala Digimizer, un software de un editor reconocido por Microsoft Windows.	Envío de alerta, adición en el informe y generación de ventana de aviso en la máquina local.
20:48h	Se acaba con el servicio de Avast Antivirus mediante un kill al proceso.	Envío de alerta, adición en el informe diario y apagado de las interfaces de red para el equipo local.
20:50h	Se deshabilita el servicio wuauserv (Actualizaciones) del registro de Windows	Ninguno.

20:52h	Se simula la conexión a un dominio catalogado como DGA.	Adición en el informe diario y apagado del equipo.
--------	---	--

Tabla 11. Actividades realizadas en la máquina y su resultado esperado. Fuente: Elaboración propia.

RESULTADOS

20:46h	Se muestra una ventana de alerta al usuario con la sesión activa.
20:46h	Se recibe alerta con relación a que el software Digimizer ha sido instalado con éxito en la máquina 192.168.1.110.
20:50h.	Se recibe alerta en relación con que el servicio antivirus ha sido desactivado en la máquina 192.168.1.110.
20:50h.	Todos los adaptadores de red se encuentran deshabilitados en la máquina 192.168.1.110.
21:52h	La máquina 192.168.1.110 se apaga.
21:05h	El informe diario generado de la máquina 192.168.1.110 con todos los eventos esperados. También se muestra una conexión a vlurgeddydy.com, catalogado como DGA.

Tabla 12. Resultados obtenidos tras la ejecución de las actividades. Fuente: Elaboración propia.

A continuación, se observa tanto el informe diario generado, como la alerta recibida en el equipo local.

Incidente	Fecha	Evento	Criticidad
Instalaciones detectadas	2020-04-21T20:46:25	1 nuevos programas instalados	5
Instalacion segura detectada	2020-04-21 20:46:24	Se ha producido una instalacion Product: Digimizer -- Installation completed successfully.	5
Antivirus deshabilitado	2020-04-21 20:50:27.370401	El servicio antivirus se encuentra deshabilitado	9
Conexión DGA	2020-04-21 20:52:27.370401	Se ha detectado una conexión al dominio vlurgeddygdy.com catalogado como DGA	10



Figura 17. Informe generado por la herramienta y alerta visualizada en el equipo local. Fuente: *Elaboración propia.*

10.1.3.Caso 3

Configuración de la herramienta		
Detección	Estado	Acción de respuesta
Dispositivos externos	Desactivado	Ninguna
Instalaciones	Solo de origen desconocido	Ninguna
Firewall	Desactivado	Ninguna
Conexiones remotas	Desactivado	Ninguna
Antivirus	Solo instalación	Ninguna
Actualizaciones	Activado	Ninguna
Generación de informe	12:00h.	-

Tabla 13. Configuración inicial de la herramienta. Fuente: *Elaboración propia.*

Acciones realizadas		
Hora	Descripción	Resultado esperado
11:44h	Se activa la herramienta en la máquina 192.168.1.110	Envío de informe diario a las 12:00h.
11:45h	Se instala Digimizer, un software de un editor reconocido por Microsoft Windows	Ninguno.
11:46h	Se instala ICMeasure, un software de un editor no reconocido por Microsoft Windows	Envío de alerta y adición en el informe.
11:46h	Instalación de una actualización del sistema.	Envío de alerta y adición en el informe.
11:48h	Se desinstala el único antivirus del sistema.	Envío de alerta y adición en el informe.

Tabla 14. Actividades realizadas en la máquina y su resultado esperado. Fuente: Elaboración propia.

RESULTADOS

11:46h	Se recibe alerta con relación a que un software de un editor no reconocido ha sido instalado en el sistema.
11:46h	Se recibe una alerta con relación a que una actualización ha sido instalada en la máquina 192.168.1.110.
11:49h.	Se recibe una alerta con relación a que no se ha encontrado software antivirus en la máquina 192.168.1.110.
12:00h	El informe diario generado de la máquina 192.168.1.110 con todos los eventos esperados. También se muestran detalles sobre la actualización instalada.

Tabla 15. Resultados obtenidos tras la ejecución de las actividades. Fuente: Elaboración propia.

A continuación, se observa tanto el informe generado por la herramienta como la alerta recibida en la dirección de correo del administrador.

Incidente	Fecha	Evento	Criticidad
Instalaciones detectadas	2020-04-22T11:46:08	2 nuevos programas instalados	5
Instalacion de un editor desconocido detectada	2020-04-22 11:46:27	Software IC Measure	6
Actualización encontrada	2020-04-22 11:46:41.380408	Actualización de inteligencia de seguridad para Windows Defender Antivirus - KB2267602 (Versión 1.313.2033.0) realizado por System	1
No se ha detectado ningún antivirus instalado	2020-04-22 11:48:40.257720	Antivirus no encontrado	9

auto.diagnose.tool.tfm@gmail.com
para mí ▾

22 abr. 2020 11:46 (hace 23 horas)

2020-04-22 11:46:27 || Actualizacion encontrada: Actualización de inteligencia de seguridad para Windows Defender Antivirus - KB2267602 (Versión 1.313.1920.0) realizado por System

Figura 18. Generación del informe por la herramienta y alerta recibida en el correo del administrador.

10.2. Agente externo

Las pruebas en el Agente externo se han centrado en realizar análisis a un servicio público propio y comparar los resultados con los esperados.

IP	Puerto	Servicio	Hora análisis	Resultado esperado
192.168.1.127	21	FTP	15:12	El puerto se encuentra cerrado
192.168.1.127	22	SSH	15:12	El puerto se encuentra abierto, pero publica información sobre la versión de OpenSSH y distribución de la máquina.
192.168.1.127	80	HTTP	15:12	El puerto se encuentra abierto, pero publica información sobre la versión de Apache y la distribución de la máquina, además, las cabeceras de seguridad no están configuradas de forma correcta.
192.168.1.127	3306	MYSQL	15:12	El puerto se encuentra abierto, pero publica información sobre la versión de MariaDB y la distribución de la máquina.

Tabla 16. Servicios probados a través del agente externo y resultado esperado. Fuente: Elaboración propia.

RESULTADOS

Servicio	Fecha Inicio	Fecha Fin	Resultado
FTP	15:12:01h	15:12:02h	El informe diario muestra que el puerto no está abierto o preparado para conexiones entrantes.
SSH	15:12:02h	15:12:03h	El informe diario muestra que el puerto está abierto y revelando información sobre la versión de OpenSSH (7.9p1), distribución (Ubuntu).
HTTP	15:12:03h.	15:12:05h	El informe diario muestra que el puerto está abierto, revela información sobre la versión de Apache (2.4.38), distribución utilizada (Ubuntu), además de la siguiente información sobre cabeceras: <ul style="list-style-type: none">- HTTPS bien configurado.- No hay cabeceras Etag y X-Powered-By.- Puntuación F (A, B, C, D, E, F).- Las cabeceras STS, CSP, XFO, XCTO, RP, FP, ECT no se encuentran presentes.
MYSQL	15:12:01h	15:12:02	El informe diario muestra que el puerto está abierto y revela información sobre la versión de MariaDB (10.3.2) y la distribución (Ubuntu).

Tabla 17. Resultado obtenido tras la ejecución de los análisis a los servicios públicos por parte del agente externo. Fuente: Elaboración propia.

Algunos fragmentos del informe generado se muestran a continuación.

IP: 127.0.0.1 Puerto: 22 Tecnologia: SSH

Nombre	Tipo	Fecha Inicio	Fecha Fin	Estado	Descripción
Scan	Exploration	2020-04-22 15:12:02	2020-04-22 15:12:02	1	Scanning says that port seems open and ready for connections
Banner Grabing	Exploration	2020-04-22 15:12:02	2020-04-22 15:12:03	0	b'SSH-2.0-OpenSSH_7.9p1 Ubuntu-10\r\n'

IP: 127.0.0.1 Puerto: 80 Tecnologia: HTTP

Nombre	Tipo	Fecha Inicio	Fecha Fin	Estado	Descripción
HttpsEnabled	Web server Pentest	2020-04-22 15:12:03	2020-04-22 15:12:05	1	HTTPS is enabled and working
HttpsRedirection	Web server Pentest	2020-04-22 15:12:03	2020-04-22 15:12:05	1	HTTPS Redirection enabled and working
SecuredCookie	Web server Pentest	2020-04-22 15:12:03	2020-04-22 15:12:05	0	Header not found
ETag	Web server Pentest	2020-04-22 15:12:03	2020-04-22 15:12:05	0	Header not found
X-Powered-By	Web server Pentest	2020-04-22 15:12:03	2020-04-22 15:12:05	0	Header not found
Score	Web server headers	2020-04-22 15:12:03	2020-04-22 15:12:05	0	F
Strict-Transport-Security	Web server headers	2020-04-22 15:12:03	2020-04-22 15:12:05	0	Is an excellent feature to support on your site and strengthens your implementation of TLS by getting the User Agent to enforce the use of HTTPS. Recommended value "Strict-Transport-Security: max-age=31536000; includeSubDomains".

Figura 19. Informe generado por el Agente Externo enviado al correo del administrador. Fuente: Elaboración propia.

10.3. Integración

Las pruebas de integración se han centrado en ejecutar el Agente Externo en la máquina Ubuntu 20.04 a la vez que se utilizaba la herramienta local en un equipo Windows 10 de la misma red, comprobándose que el envío de señales de inicio, parada y Keep Alive se realizaba de la forma esperada.

Configuración		
Cliente	Inicio actividad	Fin actividad
Organización A	17:00	21:00

Tabla 18. Configuración del horario de actividad para la Organización A. Fuente: Elaboración propia.

Actividades realizadas		
Hora	Descripción	Resultado esperado
01:12h.	Se inicia la máquina 192.168.1.110.	Envío de alerta al mail del administrador.
17:01	Se inicia la máquina 192.168.1.110.	Ninguna
17:05	Se apaga la herramienta adrede a través de la parada del servicio.	Envío de alerta a la dirección de correo del administrador.
17:08	Se vuelve a iniciar el servicio.	Ninguna.
17:09	Se apaga la herramienta a través de un SIGKILL a los procesos.	Envío de alerta a la dirección de correo del administrador antes de los próximos 29 minutos.
20:00	Se vuelve a iniciar el servicio.	Ninguna.
21:37	Se envía señal Keep Alive al Agente Externo debido a que la herramienta sigue ejecutándose.	Envío de alerta a la dirección de correo del administrador.

Tabla 19. Actividades realizadas y resultado esperado. Fuente: Elaboración propia.

RESULTADOS

Hora	Resultado
01:12h.	Se recibe una alerta con relación a que se acaba de encender el equipo 192.168.1.110 a una hora prohibida.
17:05h	Se recibe una alerta con relación a que se acaba de apagar la herramienta en el equipo 192.168.1.110 a una hora prohibida.
17:37h	Se recibe una alerta con relación a que el equipo 192.168.1.110 parece apagado, debido a que no se ha recibido la señal Keep Alive desde hace 29 minutos.
21:37h.	Se recibe una alerta con relación a que el equipo 192.168.1.110 sigue encendido en una hora prohibida.

Tabla 20. Resultados obtenidos tras la ejecución de las actividades. Fuente: Elaboración propia.

A continuación, se puede observar un ejemplo de mensaje de alerta recibido en la bandeja de entrada del administrador.



Figura 20. Alerta recibida en el correo del administrador.

11. Resultados

La herramienta de autodiagnóstico es capaz de realizar una amplia gama de detecciones a través de una interfaz gráfica para que el usuario administrador introduzca la configuración, contando con un diseño simple y minimalista. Esta interfaz puede ser ejecutada tantas veces como se quiera con el objetivo de cambiar la configuración, aunque esto no será necesario si únicamente se busca que la herramienta se ejecute siempre de manera automática al iniciar el equipo.

Herramienta de autodiagnóstico	
Email Administrador:	mtm41@alu.ua.es
Dispositivos externos:	Activado
Instalaciones:	Solo de origen desconocido
Firewall:	Activado
Conexiones:	Activado
Antivirus:	Activado
Actualizaciones:	Activado
Clave API	vsGaNvG?/QXmh_q+ayL
Generación Informe:	17:00
Acción de respuesta	
	Ninguna
	Mostrar aviso
	Ninguna
	Apagar PC
	Desconectar red
	Ninguna
Ok	

Figura 21. Interfaz de configuración de la herramienta. Fuente: Elaboración propia.

Como se ha visto en la sección anterior *Pruebas y validación*, la herramienta también será capaz de mostrar determinadas alertas al usuario local cuando se produzca un incidente de seguridad definido en la configuración previa. Estos mensajes, mostrarán un mensaje personalizado según el tipo de incidente detectado.

11.1. Cumplimiento de objetivos

Se puede afirmar que los objetivos definidos al inicio del proyecto se cumplen, ya sea en mayor o menor medida, en su totalidad. A continuación, se justifica porque esto es así para cada objetivo especificado, además, esta justificación podrá ir acompañada con experiencias de usuarios reales de la aplicación y/o estadísticas.

Capacidad de identificar acciones cotidianas de un usuario del sistema que suponen un riesgo de seguridad. Disminuir el número de incidentes de seguridad que ocurren en el sistema.

Si bien es cierto que esta situación es muy general, la herramienta es capaz de detectar una buena cantidad de situaciones en las que el usuario puede poner en riesgo el sistema, e incluso, toda la red local, ya sea de manera intencionada o inintencionada. De hecho, la herramienta ha sido desplegada de manera local en 5 equipos pertenecientes a usuarios de perfil variado durante una semana, mostrando el siguiente resultado en cuanto a detecciones:

- **Un usuario avanzado.** Tiene conocimientos técnicos sobre TI.
- **Dos usuarios independientes.** No dispone de conocimientos técnicos sobre TI, pero la utilizan en el día a día con soltura para desempeñar varias acciones.
- **Dos usuarios básicos.** No disponen de conocimientos técnicos sobre TI y tampoco de suficiente soltura para una acción que no consista en navegar por Internet de forma básica.

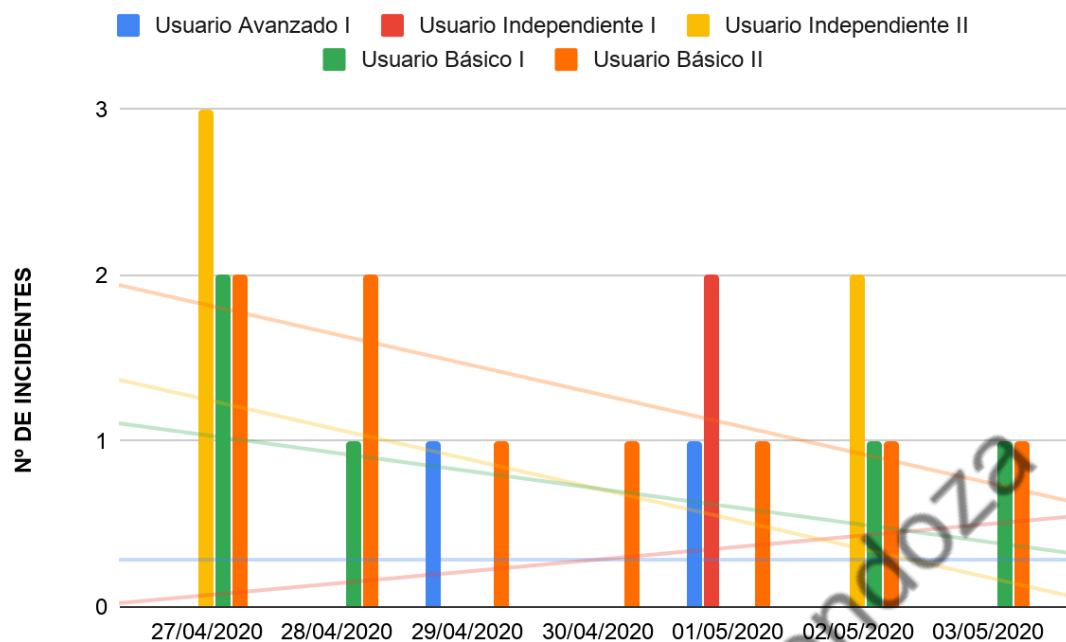


Figura 22. Número de incidentes por día por usuario (barras) y tendencia de estos (líneas). Fuente: Elaboración propia.

Como se puede observar, la herramienta es capaz de detectar incidentes de seguridad todos los días en diferentes perfiles de usuario durante un uso cotidiano. Además, aunque la duración de la muestra no es muy grande, sí que se observa una tendencia negativa en la mayoría de los casos. Esto podría ser debido al hecho de que los usuarios han entendido la alerta generada y han seguido el consejo.

Capacidad de identificar malas configuraciones (a nivel básico), tanto en la máquina local del usuario como en los servicios públicos de este.

Como ya se ha visto a lo largo del documento, la herramienta es capaz de detectar malas configuraciones locales, por ejemplo, las relacionadas con el firewall, actualizaciones, antivirus, etc. Incluso, las basadas en los accesos a otros equipos de la red interna a través del informe de conexiones que se genera, eso sí, debiendo de ser analizado por un usuario que conozca la situación de la red.

Por otro lado, el agente externo también es capaz de detectar malas configuraciones de los servicios públicos que un determinado cliente ha definido. Aunque el nivel actual de este componente no es tan alto como podría ser una prueba de intrusión, sí que se detectan malas configuraciones a un nivel básico, por ejemplo, puertos abiertos, revelación de información, e incluso, análisis de cabeceras y funcionamiento de servidores web.

Capacidad de alertar a un usuario ante una incidencia de seguridad en el sistema.
Capacidad de generar informes que aporten la información necesaria a una persona con conocimientos técnicos, no necesariamente muy altos, para solucionar un problema de seguridad.

En este sentido, tanto la herramienta local como el agente externo cumplen con su cometido, ya que, por un lado, la herramienta local genera, de manera diaria, informes con todos los incidentes de seguridad activados por el usuario administrador. Respecto a estos, los usuarios que han probado el software han comentado lo siguiente:

Usuario Avanzado I: “En mi caso, no se han generado muchas alertas, pero cuando lo han hecho he entendido el informe, era bastante directo”.

Usuario Independiente I: “He descubierto una conexión de la que no era consciente, también he visto que se han actualizado algunas cosas, pero supongo que eso es bueno”.

Usuario Independiente II: “La verdad es que de las cosas que me han salido, las he entendido, además, ya sabía de su existencia. En cambio, no sabía que había instalado un programa de un editor desconocido”.

Usuario Básico I: “Algunas cosas sí que las he entendido, por ejemplo, el tema del antivirus desactivado, pero lo de actualizaciones automáticas desactivadas me costó un poco más pillarlo”.

Usuario Básico II: “Viendo los informes, parece que mi ordenador no se actualiza bien, pero no tengo ni idea de lo que es un firewall”.

Capacidad de que el usuario pueda visualizar avisos o sufrir acciones en forma de respuesta activa a un incidente de seguridad.

Estas son las llamadas acciones de respuesta, como se ha tratado anteriormente, el usuario puede seleccionar que se muestre un aviso, se deshabiliten los adaptadores de red o que se apague el equipo. Normalmente, la mayoría de los usuarios que han probado la herramienta coinciden en que, como mucho, han seleccionado la acción de respuesta “Mostrar aviso”, ya que no querían que se les apagase el equipo o la red, lo que podía ocasionar una interrupción de alguna transacción importante. Por el contrario, coinciden en que esto sí que sería interesante para un perfil más directivo, en vistas de evitar posibles ataques de seguridad.

Por último, se destaca que el código fuente desarrollado se encuentra en su totalidad en mi Github personal por si se quiere investigar en detalle: <https://github.com/mtm41/TFM>

11.2. Cumplimiento de la planificación

La planificación se ha cumplido casi a la perfección, aunque, sí que es cierto que el volumen de trabajo ha sido cambiante a lo largo del proyecto, principalmente, teniendo subidas en la fase de inicio (Análisis de requerimientos, Viabilidad, Objetivos, etc.) y en la fase de implementación, mientras que el volumen de trabajo bajó en la fase de Diseño y Conclusiones.

Se considera que, aunque esta situación no es la ideal y debería de haber sido más uniforme, es causa colateral de las debilidades del proyecto, nombradas en el análisis introspectivo de la sección *Estudio de viabilidad*. A continuación, se muestra una tabla comparativa de la planificación esperada con la real.

Contenido	Fecha límite planificada	Fecha de entrega real
Motivación, justificación y objetivo general Introducción Estudio de viabilidad	Mediados de febrero	Miércoles, día 5 de febrero de 2020.
Estado del arte Objetivos	Mediados de marzo	Domingo, día 9 de febrero de 2020.
Metodología Análisis y especificación Diseño	Finales de marzo	Domingo, día 23 de febrero de 2020.
Implementación Pruebas y validación	Finales de abril	Jueves, día 23 de abril de 2020.
Conclusiones y trabajos futuros	Mediados de mayo.	Martes, día 5 de mayo de 2020.

Tabla 21. Comparativa de la planificación esperada con la resultante. Fuente: Elaboración propia.

11.3. Vinculación con asignaturas

La asignatura que mayor relación tiene con el proyecto es **Sistemas de Gestión de la Seguridad**. En ella se estudian varias fases de documentación sobre la gestión y el gobierno de los activos TI en una organización concreta, según las prioridades y la política de esta. Claramente, esta es la idea base en la que se fundamenta este proyecto, ya que se proporciona al usuario administrador una serie de funcionalidades para comprobar y asegurar que los demás usuarios cumplen con la política de la organización.

Por otro lado, si nos remitimos a los objetivos directos de las otras asignaturas impartidas en el curso, ninguna tiene relación con este proyecto, pero sí que es cierto que conocimientos específicos que se han aprendido de ellas, se han utilizado en este. Por ejemplo:

- **Análisis forense.** En esta asignatura se debe de trabajar con los sistemas de registro y eventos de Windows, algo fundamental para determinar la causa de una situación concreta. El proyecto también ha necesitado de un conocimiento amplio en lo que respecta a la manera de Windows de utilizar estos servicios, almacenar información e interpretarla.
- **Hacking Ético y Contramedidas.** En esta asignatura se dan, en especial, las fases de una prueba de intrusión y herramientas para llevarlo a cabo. En el proyecto, el agente externo que realiza pruebas sobre los servicios públicos del cliente emplea algunas técnicas definidas en la fase de exploración de una prueba de intrusión, por ejemplo, el Banner Grabbing o escaneo de puertos.

12. Conclusiones y trabajo futuro

12.1. Alcance logrado

Actualmente, el proyecto cumple con el alcance que se había predefinido, ya que, se ha logrado desarrollar una herramienta capaz de realizar detecciones de situaciones del día a día, que pueden repercutir de manera directa en la seguridad. Todo esto, teniendo en cuenta la variabilidad de la política de seguridad de cada caso.

Por otro lado, algo para tener en cuenta es la extensibilidad de la herramienta, a la que se pueden añadir nuevas funcionalidades de manera sencilla gracias al diseño basado en clases y reflexión. Esta situación se asocia tanto a la herramienta local, como al agente externo.

El agente externo completa las funcionalidades del proyecto, al multiplicar las posibilidades de este, por ejemplo, pudiendo efectuar un análisis desde fuera de la red, enviando alertas y gestionando el estado de las herramientas locales de un cliente.

Lo cierto es, que el carácter generalista o básico de las detecciones, limita un poco el proyecto, ya que, normalmente, el malware utiliza un número mayor de técnicas para detectar de las que puede realizar un usuario común. Aunque, en la gran mayoría de ocasiones, un error del usuario es el vector de entrada para dicho malware, por lo que esta herramienta puede, incluso, ser de mayor importancia que una que implemente una gran complejidad en sus detecciones.

También, cabe destacar el papel que puede jugar esta herramienta en un entorno empresarial, ya que utiliza una cantidad de recursos mínima y puede servir a personal técnico para determinar si las acciones reflejadas en la política de seguridad de la compañía se están llevando a cabo o no. De hecho, el funcionamiento de la herramienta también se puede extrapolar al hogar, donde, seguramente, el acceso a las máquinas sea más fácil para un usuario sin conocimientos que cometa errores, o incluso, un atacante con malas intenciones.

12.2. Trabajo futuro

A lo largo del desarrollo del proyecto, se han ido descubriendo posibles mejoras o adiciones que podrían ser interesantes para este:

- Aportar mayor control al agente externo, al más puro estilo Command and Control (C2C). De esta manera, en lugar de realizar acciones de respuesta predefinidas, se podría intentar cancelar la acción que ha causado el incidente de seguridad, o incluso, proporcionar acceso directo al usuario administrador a la máquina.
- Adición de un panel de control en el agente externo. De esta forma, tras una autenticación, el usuario administrador podría ver de manera más gráfica las estadísticas e informes generados tanto, por medio del análisis a los servicios públicos, como en las máquinas locales.
- Adición de un mayor número de mecanismos de notificación de alertas. Esto es debido a que, de momento, únicamente se envían emails para notificar de alertas, pero, fácilmente, se podría notificar a través de servicios de mensajería instantánea como Whatsapp o Telegram.
- Añadir el soporte para sistemas Linux para la herramienta local.
- Creación de una herramienta de autodiagnóstico de mayor tecnicidad, la cual emplee técnicas de detección de malware avanzado.
- Adición de un mayor número de pruebas en el agente externo.
- Adición de un mayor número de detecciones en la herramienta local.

12.3. Aportación a mi desarrollo profesional

En cuanto a lo que respecta a mi desarrollo profesional, este proyecto me ha proporcionado un mayor entendimiento del funcionamiento interno del sistema operativo Windows, en parte, desconocido para mí. Además, también he mejorado mis aptitudes para crear un documento de este calado, que incluye varias metodologías de índole ingenieril. Todo esto, siguiendo procesos de desarrollo de software como el IEEE 830 y la metodología *Waterfall*.

Referencias

- [1] Marvin the Robot (02/05/2016). Zcryptor: Ramsomware that spreads itself as a worm. Kaspersky Official Blog. Kaspersky. Recuperado de <https://www.kaspersky.com/blog/zcryptor-ransomware/12268/>
- [2] Statista Digital Market Outlook - Market Report. Smart Home Report 2019 (05/05/2020). Statista. Recuperado de <https://www.statista.com/outlook/279/100/smart-home/worldwide#market-revenue>
- [3] IBM X-force Threat Intelligence Index 2019 (10/03/2020). IBM. Recuperado de <https://www.ibm.com/downloads/cas/ZGB3ERYD>
- [4] Corinne Reichert (06/08/2019). AT&T workers bribed to infect firm's gear with malware, unlock phones, DOJ alleges - CNET. CNET News. Recuperado de <https://www.cnet.com/news/at-t-employees-bribed-to-infect-mobile-devices-with-malware-doj-alleges/>
- [5] Virtru Editorial Team (05/05/2020). Insider Threats in Cyber Security. Virtru Blog. Virtru. Recuperado de <https://www.virtu.com/blog/insider-threats-in-cyber-security/>
- [6] Kaspersky (05/05/2020). The Human Factor in IT Security: How Employees are Making Businesses Vulnerable from Within. Kaspersky official blog. Kaspersky. Recuperado de <https://www.kaspersky.com/blog/the-human-factor-in-it-security/>
- [7] Grupo Verne Actualidad (05/05/2020). Security Operations Center (SOC): una solución integral de ciberseguridad. Verne Group. Recuperado de <https://www.vernegrup.com/actualidad/security-operations-center-soc-una-solucion-integral-de-ciberseguridad>
- [8] Trend Micro (27/02/2018). A Look Into the Most Noteworthy Home Network Security Threats of 2017. USA: TrendMicro. Recuperado de <https://www.trendmicro.com/vinfo/us/security/research-and-analysis/threat-reports/roundup/a-look-into-the-most-noteworthy-home-network-security-threats-of-2017>
- [9] SolarWinds (05/05/2020). FREE Event Log Consolidator. Solar Winds. Recuperado de <https://www.solarwinds.com/free-tools/event-log-consolidator>
- [10] Event Sentry (05/05/2020). SIEM Monitoring Software for you Network. Event Sentry. Recuperado de <https://www.eventsentry.com/solutions/siem>

- [11] Netwrix (05/05/2020). Detect and Investigate Suspicious User Behavior. Netwrix. Recuperado de https://www.netwrix.com/user_behavior_analytics.html
- [12] Wikipedia EN (23/04/2020). Waterfall Model. Wikipedia. Recuperado de https://en.wikipedia.org/wiki/Waterfall_model
- [13] Wikipedia EN (14/03/2020). Spiral Model. Wikipedia. Recuperado de https://en.wikipedia.org/wiki/Spiral_model
- [14] Wikipedia EN (10/02/2020). Rapid Application Development. Wikipedia. Recuperado de https://en.wikipedia.org/wiki/Rapid_application_development
- [15] IEEE C/S2ESC - Software And Systems Engineering Standards Committee (09/12/2009). 830-1998 - IEEE Recommended Practice for Software Requirements Specifications. Recuperado de <https://standards.ieee.org/standard/830-1998.html>
- [16] Drew batchelor, Michael Satran y Mike Jacobs (31/05/2018). Event Logging - Win32 Apps. Microsoft. Recuperado de <https://docs.microsoft.com/en-us/windows/win32/eventlog/event-logging>
- Computers Step By Step (05/05/2020). Services - Windows Event Log Service. Recuperado de <https://computerstepbystep.com/windows-event-log-service-windows-7.html>
- [17] Wikipedia EN (08/04/2020). Event Viewer. Wikipedia. Recuperado de https://en.wikipedia.org/wiki/Event_Viewer
- [18] Guido van Rossum (05/05/2020). Python. Recuperado de <https://www.python.org/>
- [19] D. Richard Hipp (05/05/2020). Sqlite. Recuperado de <https://www.sqlite.org/index.html>
- [20] Apache Software Foundation (05/05/2020). Welcome! The Apache HTTP Project.. Recuperado de <https://httpd.apache.org/>
- [21] MadridDB Corporation AB (05/05/2020). MariaDB Foundation. Recuperado de <https://mariadb.org/>
- [22] Chen, Peter (March 1976). "The Entity-Relationship Model - Toward a Unified View of Data". *ACM Transactions on Database Systems*.
- [23] Gordon Lyon (05/05/2020). Nmap: The Network Mapper. Recuperado de <https://nmap.org/>
- [24] JetBrains (05/05/2020). PyCharm: The Python IDE for Professional Developers. Recuperado de <https://www.jetbrains.com/pycharm/download/#section=windows>
- [25] Python Documentation (20/04/2020). sqlite3 — DB-API 2.0 interface for SQLite databases — Python 3.7.7 documentation. Recuperado de <https://docs.python.org/3.7/library/sqlite3.html>

- [26] SQLite Browser Blog (04/04/2019). Version 3.11.2. Release. Recuperado de <https://sqlitebrowser.org/blog/version-3-11-2-released/>
- [27] Tim Golden (27/04/2020). PyWin32 Documentation. Timwolden. Recuperado de <http://timgolden.me.uk/pywin32-docs/contents.html>
- [28] Justin Hall, Andres Mariano Gorzelany (22/10/2018). How to get a list of XML data name elements in EventData. docs.microsoft.com. Recuperado de <https://docs.microsoft.com/en-us/windows/security/threat-protection/auditing/how-to-list-xml-elements-in-eventdata>
- [29] Mark Russinovich, Kent Sharkey, Luke Kim (28/04/2020). Sysmon - Windows Sys-Internals. docs.microsoft.com. Recuperado de <https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon>
- [30] Shateel A. Chowdhury (30/8/2019). Domain Generation Algorithm - DGA in malware. hackersterminal.com. Recuperado de <https://hackersterminal.com/domain-generation-algorithm-dga-in-malware/>
- [31] Armin Ronacher (05/05/2020). Flask Documentation. flask.palletsprojects.com. Recuperado de <https://flask.palletsprojects.com/en/1.1.x/>
- [32] John Walker (05/05/2020). How Hotbit Works. fourmilab.ch. Recuperado de <https://www.fourmilab.ch/hotbits/how3.html>

Anexo 1

En el código siguiente se observa la implementación central de la herramienta local, donde se destaca el comportamiento necesario para que esta implemente el comportamiento de un servicio Windows.

```
class AgentMonitor(win32serviceutil.ServiceFramework):
    _svc_name_ = "AgentMonitor"
    _svc_display_name_ = "Agent Monitor"
    _svc_description_ = "My service"

    def __init__(self, args):
        self.log = Logger()
        win32serviceutil.ServiceFramework.__init__(self, args)
        self.hWaitStop = win32event.CreateEvent(None, 0, 0, None)
        socket.setdefaulttimeout(60)

    def SvcDoRun(self):
        self.readConfig()
        self.log.write(States.PROGRAM_INIT.value)
        self.log.write(States.SENDING_SIGNAL_START_TO_AGENT.value)
        self.sendSignal('start')
        rc = None
        self.isrunning = True
        while rc != win32event.WAIT_OBJECT_0:
            self.main()
            rc = win32event.WaitForSingleObject(self.hWaitStop, 5000)

    def SvcStop(self):
        self.log.write(States.SENDING_SIGNAL_STOP_TO_AGENT.value)
        self.sendSignal('end')
        self.isrunning = False
        self.ReportServiceStatus(win32service.SERVICE_STOP_PENDING)
        win32event.SetEvent(self.hWaitStop)

    def main(self):
        lastDevicesCheck = None
        lastInstallationCheck = None
        lastUpdatesCheck = None
        lastFirewallCheck = None
        lastConnectionCheck = None
        while self.isrunning:
            self.log.write(States.ITERATION_INIT.value)
            self.c = DatabaseConnection()
            if self.configs['devices']:
                lastDevicesCheck =
self.mngExternalDeviceCon(lastDevicesCheck)
                if self.configs['installations'] or
self.configs['installations'] == 1:
```

```

        lastInstallationCheck =
self.mngInstalledSoftware(lastInstallationCheck)
        if self.configs['updates']:
            lastUpdatesCheck = self.mngUpdates(lastUpdatesCheck)
        if self.configs['firewall']:
            lastFirewallCheck =
self.mngFirewall(lastFirewallCheck)
        self.c.close()
        if self.configs['connections']:
            lastConnectionCheck =
self.mngNetworkConnections(lastConnectionCheck)
        if datetime.now() >= self.sendSignalTime:
            self.sendSignal('update')
        time.sleep(60)
        if datetime.now() >= self.sendSignalTime:
            self.sendSignal('update')

```

TFM-Manuel Torres Mendoza

Anexo 2

El siguiente fragmento de código XML contiene la representación de un evento en un sistema Microsoft Windows 10. Las librerías utilizadas en el desarrollo del proyecto tienen en cuenta este formato para extraer información relevante. En concreto, este evento está relacionado con una actualización de Windows Defender Antivirus.

```
- <Event
xmlns="http://schemas.microsoft.com/win/2004/08/events/event">
- <System>
  <Provider Name="Microsoft-Windows-WindowsUpdateClient"
  Guid="{945a8954-c147-4acd-923f-40c45405a658}" />
  <EventID>19</EventID>
  <Version>1</Version>
  <Level>4</Level>
  <Task>1</Task>
  <Opcode>13</Opcode>
  <Keywords>0x800000000000000018</Keywords>
  <TimeCreated SystemTime="2020-04-13T10:00:54.013217600Z" />
  <EventRecordID>43895</EventRecordID>
  <Correlation />
  <Execution ProcessID="10208" ThreadID="8112" />
  <Channel>System</Channel>
  <Computer>DESKTOP-4I7ITAS</Computer>
  <Security UserID="S-1-5-18" />
</System>
- <EventData>
  <Data Name="updateTitle">Actualización de inteligencia de seguridad
para Windows Defender Antivirus - KB2267602 (Versión
1.313.1426.0)</Data>
  <Data Name="updateGuid">{4a51a099-889d-4101-ab92-
5cd5b3517b8a}</Data>
  <Data Name="updateRevisionNumber">200</Data>
  <Data Name="serviceGuid">{9482f4b4-e343-43b6-b170-
9a65bc822c77}</Data>
</EventData>
</Event>
```

Anexo 3

En el siguiente fragmento de código se observa la implementación de la búsqueda y detección de eventos por parte de la herramienta local. El método searchEvents se centrará en los eventos asociados a servicios o aplicaciones, mientras que el searchNormalEvents en los eventos generales del sistema (Aplicación, sistema y seguridad). La reflexión de las clases de eventos juega un papel importante en todos ellos, intentando generalizar lo máximo posible.

```
# Method for searching software self generated events.
def searchEvents(self, date, logtype):
    importantEvents = []
    eventLog = win32evtlog.EvtOpenLog(logtype, 1, None)
    totalRecords =
win32evtlog.EvtGetLogInfo(eventLog, win32evtlog.EvtLogNumberOf
LogRecords)[0]
    eventResultSet = win32evtlog.EvtQuery(logtype, 1, '*', None)
    events = self.calculateOffset(totalRecords, eventResultSet)
    totalRecords = totalRecords - len(events)

    while len(events) != 0:
        for event in events:
            xml = win32evtlog.EvtRender(event, 1)
            root = ElementTree.fromstring(xml)
            sourceName = self.getClassFromSourceEvent(logtype, 1)
            evClass = globals()[sourceName]
            evIns = evClass(logtype, root[0], root[1])
            if evIns.checkEvent(date):
                importantEvents.append(evIns)
            events = self.calculateOffset(totalRecords, eventResultSet)
#Next iteration
            totalRecords = totalRecords - len(events)

    return importantEvents

# Method for searching events from Application, Security and System
categories
def searchNormalEvents(self, date, logtype, sourceName):
    importantEvents = []
    suspiciousEvents = []
    eventLog = win32evtlog.OpenEventLog(None, logtype)
    flags = win32evtlog.EVENTLOG_BACKWARDS_READ |
win32evtlog.EVENTLOG_SEQUENTIAL_READ
    totalRecords = win32evtlog.GetNumberOfEventLogRecords(eventLog)
    try:
        events = 1
        while events:
            events = win32evtlog.ReadEventLog(eventLog, flags, 0)
#Next iteration

            for ev_obj in events:
                sourceName =
```

```

self.getClassFromSourceEvent(ev_obj.SourceName, 0)
    if sourceName is not None:
        evClass = globals()[sourceName]
        evIns = evClass(logtype,
ev_obj.TimeGeneratedev_obj.StringInserts,    ev_obj.SourceName,
ev_obj.EventID)
        if evIns.checkEvent(date, suspiciousEvents,
importantEvents) \
            and ev_obj.SourceName != 'Microsoft-Windows-
RestartManager':
                importantEvents.append(evIns)
                if ev_obj.SourceName == 'MsiInstaller':
                    suspiciousEvents.append(evIns)

except:
    self.log.write(States.ERROR_READING_EVENTS.value)

return importantEvents

```

TFM-Manuel Torres Mendoza

Anexo 4

El siguiente fragmento de código se encargará de buscar en el registro de Windows para determinar si un determinado servicio se encuentra en estado Habilitado o Deshabilitado.

```
# This method will determine if service is enabled or not based on
windows registry information
def checkServiceIsDisabled(self, service, message):
    disabled = False
    isuphandle =
win32api.RegOpenKeyEx(win32con.HKEY_LOCAL_MACHINE,
"SYSTEM\\CurrentControlSet\\Services\\"
                                + service, 0,
win32con.KEY_READ)
    isuptype = win32api.RegQueryValueEx(isuphandle, "Start")[0]
    win32api.RegCloseKey(isuphandle)
    if isuptype == 4:
        disabled = True

    return disabled
```

Anexo 5

El siguiente fragmento de código realiza varias operaciones relacionadas con las detecciones del software antivirus, desde comprobar que se encuentra instalado consultando el registro de Windows, si el servicio se está ejecutando, hasta la comprobación de versión con la última registrada en un famoso repositorio de software.

```
# This method will perform a series of operations to conclude if
antivirus software is
# up to date or not
def mngAntivirus(self, installedApps):
    self.log.write(States.CHECK_ANTIVIRUS.value)
    good = False
    sql = "INSERT INTO Evento VALUES(?,?,?,?)"
    antivirusStatus_event = (str(datetime.now()),
'AntivirusCheck',
'Antivirus no encontrado', '', 'AntivirusNotInstalled')
    antivirus = self.checkAntivirusInstalled(installedApps)
    if antivirus:
        activeServices = self.ListServices()
        actualAntivirusVersions =
self.getAntivirusVersion(activeServices)
        if self.configs['antivirus'] != 1: # Checking only
installation option
            for antivirusNameInstalled in antivirus.keys():
                antivirusStatus_event = (str(datetime.now()),
'AntivirusCheck',
'Antivirus encontrado:
{}'.format(antivirusNameInstalled),
'',
'AntivirusInstalledWithoutVersion')
            for actualAntivirusVersion in
actualAntivirusVersions.keys():
                if actualAntivirusVersion.capitalize() in
antivirusNameInstalled.capitalize():
                    if antivirus[antivirusNameInstalled][0] >=
actualAntivirusVersions[actualAntivirusVersion]:
                        good = True
                    else:
                        antivirusStatus_event =
(str(datetime.now()), 'AntivirusCheck',
'Antivirus
encontrado: {} {}'.format(antivirusNameInstalled, antivirus[antivirusNameInstalled][0]),
'',
'AntivirusNotUpToDate')
                else:
                    good = True
```



```

        if not good:
            self.send_alert(antivirusStatus_event)
            self.execResponse(self.responses['antivirus'],
'antivirus')
            self.c.insert_values([antivirusStatus_event], sql)

def ListServices(self):
    accessSCM = win32con.GENERIC_READ

    # Open Service Control Manager
    hscm = win32service.OpenSCManager(None, None, accessSCM)

    # Enumerate Service Control Manager DB
    typeFilter = win32service.SERVICE_WIN32
    stateFilter = win32service.SERVICE_STATE_ALL
    statuses = win32service.EnumServicesStatus(hscm, typeFilter,
stateFilter)

    return statuses

# This method will request actual antivirus software version
def getAntivirusVersion(self, statuses):
    antivirusUrls = {"avast": "https://avast.en.softonic.com/",
                    "norton": "https://norton-
antivirus.en.softonic.com/",
                    "panda": "https://panda-free-
antivirus.en.softonic.com/",
                    "kaspersky": "https://kaspersky-anti-
virus.en.softonic.com/"}

    antivirusVersions = {}
    antivirusServiceFound = False

    for (short_name, desc, status) in statuses:
        if "Antivirus" in short_name:
            antivirusServiceFound = True
            canReadVersion = False
            antivirusRead = ""
            for antivirusTemplate in antivirusUrls.keys():
                if antivirusTemplate in short_name:
                    canReadVersion = True
                    antivirusRead = antivirusTemplate
            if canReadVersion:
                found = False
                fp =
urllib.request.urlopen(antivirusUrls[antivirusRead])
                mybytes = fp.read()

                mystr = mybytes.decode("utf8")
                fp.close()

                index = mystr.find("<h3 class=\"app-
specs__title\">Version")
                version = ""
                if index != -1:
                    i = 75

```

```

        while not found:
            if mystr[index + i] != '<':
                version += mystr[index + i]
            else:
                found = True
            i = i + 1
        antivirusVersions[antivirusRead] = version

    if not antivirusServiceFound:
        sql = "INSERT INTO Evento VALUES(?,?,?,?)"
        antivirusServiceEvent = (str(datetime.now()),
        'AntivirusServiceCheck', '', '', 'AntivirusNotPresent')
        self.send_alert(antivirusServiceEvent)
        self.execResponse(self.responses['antivirus'],
        'antivirus')
        self.c.insert_values([antivirusServiceEvent], sql)

    return antivirusVersions

```

TFM-Manuel Torres Mendoza

Anexo 6

En el siguiente fragmento de código se observan los métodos más relevantes de la implementación de la API del agente externo. Por ejemplo, obtención de informes, recepción de alertas y señales, etc.

```
@app.route('/api/<key>/report/<timestamp>', methods=['GET'])
def getReport(key, timestamp):
    org = Organization()
    if org.authenticate(html.escape(key)):
        report = Report(html.escape(timestamp), org.name)
        report_json = report.generate()

        return jsonify(report_json)
    else:
        return Response("NOT AUTHORIZED", status=403)

#POST
@app.route('/api/alert/<base64ip>', methods=['POST'])
def sendLocalAlert(base64ip):
    base64IpBytes = html.escape(base64ip).encode('ascii')
    message_bytes = base64.b64decode(base64IpBytes)
    ip_address = message_bytes.decode('ascii')

    api_key = str(html.escape(request.form.get('key')))
    organization = Organization()
    if organization.authenticate(api_key):
        localReport = str(html.escape(request.form.get('alert')))
        if sendReportToUser(organization.email, localReport,
            ip_address, True):
            return Response("Alert sent", status=200)
        else:
            return Response("Error sending alert", status=500)
    else:
        return Response("NOT AUTHORIZED", status=403)

#POST
@app.route('/api/report/<base64ip>', methods=['POST'])
def sendLocalReport(base64ip):
    base64IpBytes = html.escape(base64ip).encode('ascii')
    message_bytes = base64.b64decode(base64IpBytes)
    ip_address = message_bytes.decode('ascii')

    api_key = str(html.escape(request.form.get('key')))
    organization = Organization()
    if organization.authenticate(api_key):
        localReport = str(request.form.get('report'))
        if sendReportToUser(organization.email, localReport,
            ip_address, False):
            return Response("Report sent", status=200)
        else:
            return Response("Error sending report", status=500)
    else:
        return Response("NOT AUTHORIZED", status=403)
```

```

#POST
@app.route('/api/tool/<base64ip>', methods=['POST'])
def beginWorkingDay(base64ip):
    now = datetime.datetime.now().strftime('%H:%M')

    base64IpBytes = html.escape(base64ip).encode('ascii')
    message_bytes = base64.b64decode(base64IpBytes)
    ip_address = message_bytes.decode('ascii')

    api_key = str(html.escape(request.form.get('key')))
    state = str(html.escape(request.form.get('state')))

    if state == 'start' or state == 'end':
        organization = Organization()
        if organization.authenticate(api_key):
            organization.checkSchedule(ip_address, now, state)
            return Response("TIME UPDATED", status=200)
        else:
            return Response("NOT AUTHORIZED", status=403)
    else:
        return Response("BAD STATE", status=404)

@app.route('/api/<key>/organization/<organization>',
methods=['DELETE'])
def deleteOrganization(key, organization):
    base64OrganizationBytes =
html.escape(organization).encode('ascii')
    message_bytes = base64.b64decode(base64OrganizationBytes)
    message = message_bytes.decode('ascii')

    organizationForKey = Organization()
    organizationForKey.authenticate(html.escape(key))

    if organizationForKey.name == message:
        if organizationForKey.delete():
            resp = str('Organization {} has been successfully
deleted').format(message)
            return jsonify({'Message': resp})

    return jsonify({'Message': 'Deletion was not performed'})

```

Anexo 7

En el siguiente fragmento de código se observa la implementación de algunas pruebas que realiza el agente externo a los servicios públicos especificados por el cliente, destacando, escaneo de puertos, obtención de información relevante a través de Banner Grabbing o el análisis de las cabeceras del servidor web.

```
def TCPConnect(self, ip, port, delay, listeningPorts):

    TCPsock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    TCPsock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
    TCPsock.settimeout(delay)
    try:
        TCPsock.connect((ip, port))
        listeningPorts.append(port)
    except:
        exit(0)

def checkWebServer(self):
    webServer = False
    cmd = 'echo -en \"GET / HTTP/1.1\\n\\n\\n\" | nc {}'
    {}.format(self.ip, self.port)
    proc = subprocess.Popen([cmd], stdout=subprocess.PIPE,
shell=True)
    (out, err) = proc.communicate()

    if str(out).find('HTTP') > -1:
        webServer = True

    return webServer

def scan(self, ports):
    delay = 1000
    threads = []
    listeningPorts = []

    for port in ports:
        t = threading.Thread(target=self.TCPConnect,
args=(self.ip, port, delay, listeningPorts))
        threads.append(t)

    for thread in threads:
        thread.start()

    for thread in threads:
        thread.join()

    return listeningPorts

def checkCookie(self, output):
    secureCookie = False
    cmd = 'echo {} | grep cookie'.format(output)
    proc = subprocess.Popen([cmd], stdout=subprocess.PIPE,
shell=True)
```

```
(out, err) = proc.communicate()
if len(str(out)) > 2:
    if str(out).find('HttpOnly') > -1:
        secureCookie = True

return secureCookie
```

TFM-Manuel Torres Mendoza

Anexo 8

En el siguiente fragmento de código se observa parte de la implementación del Watcher ejecutado como servicio en el agente externo, el cual, mediante una consulta SQL y reinicios, obtendrá siempre el siguiente servicio a probar.

```
if __name__ == '__main__':
    print('Starting up...')

    notify(Notification.READY)

    while True:
        conn = None

        try:
            conn = DatabaseConnection().conn
            cur = conn.cursor()

            sql = 'SELECT ip, puerto, organizacion, horaAnalisis,
domainName, tecnologia FROM Servicio ' \
                'WHERE NOT EXISTS (SELECT * FROM Prueba WHERE
Servicio.ip=Prueba.servicioIp AND
Servicio.puerto=Prueba.servicioPuerto AND
Servicio.organizacion=Prueba.organizacion ' \
                'AND DATE_FORMAT(Prueba.fechaInicio, "%Y-%m-%d") =
CURDATE()) ORDER BY horaAnalisis LIMIT 1;'
            cur.execute(sql)
            resultSet = cur.fetchone()

            sleep = 1000000

            ip = ''
            port = ''
            organization = ''
            timeForAnalysis = ''
            domain = ''
            tecnologia = ''
            if resultSet is not None:
                ... # Obtención de valores de la base de datos

            cur.close()
            conn.close()

            if resultSet is not None:
                if sleep > 0:
                    time.sleep(sleep)
                    command = 'python3
/root/PycharmProjects/TFM/analyzer/main.py {} \'\{\}\' \'\{\}\' {}
{}'.format(port, ip, organization, domain, tecnologia)
                    proc = subprocess.Popen([command],
stdout=subprocess.PIPE, shell=True)
                    (out, err) = proc.communicate()
                else:
                    time.sleep(sleep)
            except Exception as ex:
                print('ERROR {0}'.format(ex))
```

```
        break
    finally:
        if conn is not None:
            conn.close()
```

TFM-Manuel Torres Mendoza