UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf

Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Implementación de la norma de seguridad PCI-DSS versión 3.0 sobre aplicación web ASP.NET desplegada en Azure

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

Autor: Israel Pérez Gómez
Tutor: Julio Pons Terol
Co-tutor: Javier Jorge Cerdá

Curso 2015-2016

Resumen

El almacenamiento de datos bancarios requiere de un estricto control y seguimiento para asegurar en lo posible la integridad e inaccesibilidad de los datos sensibles. Estos requisitos se detallan en el Estándar de seguridad de datos para la industria de tarjeta de pago (PCI-DSS). El objetivo del TFG es implementar una serie de medidas que permitan superar las auditorías que certifiquen el cumplimiento de la norma PCI-DSS para la aplicación web y móvil "Pay[in]", cuyo despliegue se lleva a cabo en Azure, infraestructura en la nube de Microsoft. Puesto que la aplicación completa del estándar implica incrementar en gran medida la envergadura del proyecto, el desarrollo de la solución contiene únicamente las siguientes tres medidas: adición de una capa de seguridad al sistema de claves de administración a través de la herramienta de seguridad "Latch", configuración del almacén de claves criptográficas de Azure y la implementación de métodos de cifrado y uso de certificados para la protección de los datos en las comunicaciones a través de redes públicas.

Palabras clave: Estándar PCI-DSS, Microsoft Azure, seguridad, computación en la nube, almacén de claves criptográficas de Azure, Latch

Resum

L'emmagatzemament de dades bancàries requerix d'un estricte control i seguiment per a assegurar en la mesura que es puga la integritat i inaccessibilitat de les dades sensibles. Estos requisits es detallen en l'Estàndard de seguretat de dades per a la indústria de targeta de pagament (PCI-DSS). L'objectiu del TFG és implementar una sèrie de mesures que permeten superar les auditories que certifiquen el compliment de la norma PCI-DSS per a l'aplicació web i mòbil "Pay[in]", el desplegament del qual es du a terme en Azure, infraestructura en el núvol de Microsoft. Ja que l'aplicació completa de l'estàndard implica incrementar en gran manera l'envergadura del projecte, el desenrotllament de la solució conté únicament les següents tres mesures: adició d'una capa de seguretat al sistema de claus d'administració a través de la ferramenta de seguretat Latch", configuració del magatzem de claus criptogràfiques d'Azure i la implementació de mètodes de xifrat i ús de certificats per a la protecció de les dades en les comunicacions a través de xarxes públiques.

Paraules clau: Estàndard PCI-DSS, Microsoft Azure, seguretat, computació en el núvol, magatzem de claus criptogràfiques d'Azure, Latch

Abstract

Banking data storage requires strict control and monitoring to ensure as far as possible the integrity and inaccessibility of sensitive data. These requirements are specified in Payment Card Industry Data Security Standard (PCI DSS). The aim of this DFP is to implement a series of measures for overcoming the audits to certify PCI DSS compliance for the web and mobile application Pay[In], whose deployment is carried out in Azure, cloud infrastructure of Microsoft. Given that the complete standard implementation implies largely to increase the size of the project, the development of the solution contains only the following three issues: addition of a security layer to the key management system through the security tool Latch, configuration of Azure cryptographic keys warehouse and the implementation of encryption methods and use of certificates for the protection of data communications over public networks.

Key words: PCI-DSS Standard, Microsoft Azure, Security, Cloud computing, Key Vault Azure, Latch

Índice general

Índice de figuras

Índice de tablas

CAPÍTULO 1
Introducción

1.1 Motivación

Durante los últimos dos siglos hemos vivido una revolución tecnológica sin precedentes. Sus efectos se han visto incrementados de forma exponencial desde finales de la década de los años 30 del siglo XX, cuando Konrad Zuse, pionero en el campo de la computación, construyó su primera máquina programable, la Z1. Ello supuso la transición del uso de tecnologías analógicas a digitales.

Hoy en día no entendemos nuestra vida sin el uso de la tecnología y, cada vez más, nos rodeamos de un mayor número de dispositivos electrónicos que nos facilitan de una u otra forma la realización de todo tipo de tareas, ya sea para llevarlas a cabo de forma más eficaz, cómoda o en un menor tiempo. Y es lógico deducir que estos dispositivos van a seguir evolucionando y adquiriendo un mayor número de funcionalidades relacionadas directamente con las necesidades de la sociedad.

La empresa para la que se está desarrollando este proyecto, Pay[In][1], está trabajando en una aplicación web y móvil que permite a sus usuarios realizar pagos con tarjeta bancaria a través de su dispositivo móvil a modo de punto de venta (TPV) virtual. Además de poder cargar, recargar, devolver y consultar sus títulos de transportes en la ciudad de Valencia. Esto último en desarrollo.

En nuestro caso, este proyecto persigue ofrecer el servicio de pagos móviles con un mayor nivel de seguridad, integridad de datos y un menor tiempo de respuesta. Para ello es necesario cumplir una serie de pautas que minimicen al máximo todos los riesgos asociados al tratamiento de información sensible y permitan además reducir el tiempo de las operaciones.

En la actualidad, el software sobre el que se desarrolla el proyecto ya permite realizar las operaciones bancarias necesarias, pero con una serie de limitaciones ya que es la propia entidad bancaria quien se encarga de realizar los pagos, puesto que la herramienta solamente facilita la información del importe de la transferencia. Este procedimiento ralentiza el proceso de pago ya que depende completamente de la disponibilidad de los servicios del banco.

La solución a este problema radica en gestionar y almacenar los datos bancarios desde la misma aplicación, convirtiéndola en una pasarela de pagos al uso. Esta gestión requiere del cumplimiento de la norma de seguridad de datos de la industria de tarjetas de pago (PCI-DSS). Cabe destacar que el software mencionado previamente consiste en una aplicación web y móvil cuyo servicio está desplegado sobre la infraestructura en la nube de Microsoft (Azure). Dicha plataforma certifica el cumplimiento de la norma, por

lo que se hace necesario revisar qué puntos del estándar ya están cubiertos y cuales han de ser implementados todavía.

1.2 Objetivos

La finalidad de este proyecto es el desarrollo de medidas acordes al estándar PCI DSS que posibiliten la certificación de la norma, con el fin de permitir a la empresa operar con los datos bancarios de sus usuarios e independizar las operaciones de la plataforma ofrecida por la entidad bancaria. Para ello, en primer lugar se analiza la norma y se compara con los requisitos ya cumplidos por Azure. De esta forma se puede verificar qué requisitos no se han cubierto y han de subsanarse.

En segundo lugar, puesto que el cumplimiento de los requisitos requiere de mucho tiempo y abarca ámbitos tan dispares como la configuración de redes, la protección de las comunicaciones, la elaboración de protocolos de actuación, el control de inventario de los equipos informáticos en la empresa, entre muchos otros, se hace imposible abarcar toda la aplicación de la norma en este TFG. En su lugar se ha optado por el desarrollo de tres apartados de la norma:

- Protección y almacenamiento de claves criptográficas mediante la configuración del almacén de claves criptográficas de Azure (*Key Vault*).
- Cifrado de datos en las comunicaciones a través de redes públicas, mediante la implementación de métodos de cifrado y uso de certificados.
- Política de uso y gestión de contraseñas, incorporando una capa de seguridad adicional al sistema de gestión de claves de administrador mediante la integración de la herramienta de seguridad Latch[1] [2].

Puesto que hoy en día no tiene sentido la realización de un proyecto sin un análisis de costes, se estudia la política de precios de Microsoft Azure y el impacto económico de la implementación de los apartados contemplados en ese proyecto.

1.3 Estructura de la memoria

Este TFG está organizado en siete capítulos. A continuación se explica de forma breve el trabajo realizado en cada uno de ellos.

- **Capítulo 1, Introducción:**

  Este capítulo contiene los antecedentes que han motivado la elaboración de este proyecto, así cómo los objetivos a alcanzar en su desarrollo.

- **Capítulo 2, El estándar PCI DSS versión 3.0:**

  En el segundo capítulo se explica qué es y en qué consiste el estándar. Además, se reflexiona sobre su necesidad en el entorno tecnológico y se indica el proceso de evaluación de la misma para la obtención de de su certificación.

[1] Aplicación que protege tus servicios online bloqueando temporalmente el mecanismo de inicio de sesión.

- **Capítulo 3, Computación en la nube:**

  En el tercer capítulo se lleva a cabo una breve introducción a la computación en la nube, ahondando en Azure, la nube de Microsoft, ya que es la infraestructura sobre la que se apoya el proyecto real y nuestro desarrollo.

- **Capítulo 4, Análisis de los requisitos de implementación de la norma PCI DSS en Azure:**

  Puesto que Azure cumple con el estándar, se hace necesario concretar que requisitos de la norma han de ser implementados por nuestra parte y cuáles ya están cubiertos. En este cuarto capítulo se realiza este análisis, con el fin de elaborar una guía rápida de requisitos a cumplir.

- **Capítulo 5, Implementación de la norma:**

  El quinto capítulo contiene el desarrollo del proyecto, en concreto la integración de la herramienta Latch como medida adicional de seguridad, la implementación y configuración del módulo de almacenamiento de claves criptográficas de Azure y el uso de métodos de cifrado para transmitir datos sensibles a través de redes públicas.

- **Capítulo 6, Coste de los servicios en la nube de Microsoft Azure:**

  En este capítulo se analiza la política de precios de Microsoft Azure y el coste estimado de la implantación de nuestro desarrollo.

- **Capítulo 7, Conclusión:**

  En este último capítulo se hace un análisis de las metas alcanzadas durante el desarrollo del TFG, además de las dificultades encontradas y las posibles mejoras o ampliaciones que puedan realizarse.

También se incluyen dos apéndices en los que se exponen varios *sripts* sobre el despliegue y configuración del almacén de claves de Microsoft Azure y un listado de los requisitos de la norma PCI DSS cuya implementación ha de llevarse a cabo por Pay[In].

CAPÍTULO 2
El estándar PCI DSS versión 3.0

2.1 Introducción a la norma

La Norma de Seguridad de Datos para la Industria de Tarjeta de Pago (PCI DSS) es un estándar de seguridad desarrollado por un comité fundado por las compañías más importantes de tarjetas bancarias, tales como American Express, Discover Financial Services, JCB International, MasterCard y Visa, Inc. Esta comisión se autodenomina Consejo de Estándares de Seguridad para la Industria de Tarjeta de Pago (PCI SSC).

Figura 2.1: Logotipo de las PCI DSS.

Su principal objetivo consiste en mitigar el fraude relacionado con las tarjetas de crédito y aumentar la seguridad de los datos de los titulares almacenados en ellas. Estos datos se definen de la siguiente forma:

Datos de cuentas	
Los datos de titulares de tarjetas incluyen:	**Los datos confidenciales de autenticación incluyen:**
• Número de cuenta principal (PAN) • Nombre del titular de la tarjeta • Fecha de vencimiento • Código de servicio	• Contenido completo de la pista (datos de la banda magnética o datos equivalentes que están en un chip) • CAV2/CVC2/CVV2/CID • PIN/Bloqueos de PIN

Tabla 2.1: Datos sensibles almacenados en tarjetas bancarias

Los requisitos que en ella se detallan son de obligatorio cumplimiento para toda entidad que realice operaciones con tarjetas de pago, es decir, que procese, transmita o almacene información de estas características. Entre ellos se pueden encontrar comerciantes, procesadores, instituciones financieras y proveedores de servicios.

De no cumplir con la norma y a pesar de que el PCI SSC no impone ninguna consecuencia por su incumplimiento, cada una de las marcas de tarjetas bancarias tiene su propio programa de cumplimiento pudiendo imponer multas, demandas por incumplimiento de contrato, sanciones, reclamaciones de seguros o cancelaciones de

cuentas. Todo ello sumado de la consiguiente pérdida de reputación y los permisos para procesar estos datos.

Los requisitos que establecen son los mínimos a cubrir, ya que se pueden mejorar mediante controles y prácticas adicionales con el fin de mitigar otros riesgos, cumplir leyes locales, regionales o sectoriales, puesto que no sustituyen otros requisitos legales.

A su vez se establecen distintos niveles de cumplimiento de la norma en función del número de transacciones anuales. Existiendo también la posibilidad de que el nivel se asigne de forma directa en función de la distribución geográfica de la organización, del grado de consolidación de sus sistemas de información y/o del historial de incidentes de seguridad. Los criterios que determinan este nivel se detallan en la siguiente tabla, variando en función de los distintos miembros del comité:

Nivel	VISA	MasterCard	AMEX	Discover	JCB
1	Más de 6 millones de transacciones anuales	Más de 6 millones de transacciones anuales. O designados. O con historial de incidentes	Mas de 2,5 millones de transacciones anuales	Más de 1 millón de transacciones JCB o con historial de incidentes graves	No están clasificados por volumen de transacciones
2	Entre 1 y 6 millones de transacciones VISA	De 1 a 6 millones de transacciones MasterCard anuales	Entre 50.000 y 2,5 millones de transacciones AMEX o por asignación	Menos de 1 millón de transacciones JCB anuales	
3	Entre 20.000 y 1 millón de transacciones *e-commerce* anuales	Entre 20.000 y 1 millón de transacciones *e-commerce* anuales	Menos de 50.000 transacciones AMEX por año	N/A	La clasificación se realiza mediante el uso de criterios de riesgo
4	Menos de 20.000 transacciones *e-commerce* anuales o menos de 1 millón en total	El resto de usuarios MasterCard	N/A	N/A	

Tabla 2.2: Clasificación por niveles según la norma PCI DSS

El documento[3], publicado en noviembre de 2013, contiene la norma desglosada en su versión 3.0. Desde entonces, se han realizando modificaciones recogidas en dos informes, uno para la versión 3.1[4] y otro para la versión 3.2[5], habiendo sido publicado este último en abril de 2016.

2.2 La necesidad de un estándar

En el resumen ejecutivo del informe CCN-CERT IA-09/16 Ciberamenazas 2015/Tendencias 2016[6, pág 6], se indica:

> La sofisticación de las técnicas usadas, la disponibilidad de nuevas o renovadas herramientas (incluyendo la prestación de servicios delincuenciales bajo demanda -*on demand*-) y la pulcritud en la perpetración de tales acciones constituyen una preocupación en franco crecimiento.
>
> El año 2015, de manera análoga a lo ocurrido en 2014, evidenció que las organizaciones cibercriminales están dispuestas a invertir grandes cantidades de dinero en la preparación de sus acciones. Del mismo modo, el denominado Cibercrimen como Servicio CaaS (*Cybercrime-As-A-Service*) ha incrementado su penetración y profesionalización, habiéndose percibido una cierta "competencia" entre los propios ciberdelincuentes, lo que obliga a sus autores a prestar a sus "clientes" un "servicio" cada vez más fiable.

Se observa la creciente tendencia del interés por parte de organizaciones delictivas hacia el sector tecnológico. Estas ven en él una fuente muy lucrativa, propiciando incluso una mejora y especialización de sus técnicas.

En dicho documento también se menciona uno de los ataques más llamativos de robo de datos bancarios conocido hasta la fecha, en el que fueron robados 56 millones de números de tarjetas de crédito y débito, junto a 53 millones de correos electrónicos de clientes de la empresa Home Depot.

Según el informe de medios de pago y fraude online en España de abril de 2016[7], llevado a cabo por la Asociación Digital de la Economía Digital (ADIGITAL), un 20 % de las empresas encuestadas ha sufrido una tasa de fraude anual de entre el 0,25 y el 5 %, habiendo superando esta cifra un 1,7 %. Además, en el total de transacciones cuyo origen es fraudulento, las real izadas desde dispositivos móviles superan en un 2,5 % a operaciones realizadas de forma tradicional.

En el informe sobre la transformación digital de la banca española[8, pág 18] publicado en 2015 y realizado por el Departamento de Investigación del Instituto de Estudios Bursátiles (IEB), ya se refleja este hecho. Ya en el periodo comprendido entre 2010 y 2014 se detecta un crecimiento promedio anual del 9,2 % en la inclusión de usuarios de banca por Internet. También se hace hincapié en la creciente dificultad de mantener segurizados todos los sistemas de una organización, entendiendo como ésta no solo a grandes entidades bancarias fuertemente protegidas, sino a aquellos intermediarios cuyos sistemas estén peor protegidos.

El sector de la banca se considera crítico ya que cualquier fallo puede ocasionar cuantiosas pérdidas económicas. A pesar de que las medidas de seguridad empleadas dotan a los entornos de gran resistencia y resiliencia frente a ataques, nunca son suficientes ya que constantemente se detectan nuevas vulnerabilidades que son explotadas por grupos criminales.

La mayoría de las acciones llevadas a cabo por estos grupos tienen por objeto el beneficio económico, cuya motivación consiste principalmente en obtener beneficio de forma ilícita de todo aquello que tenga valor. Es por ello que los datos bancarios son un blanco suculento. Y más si tenemos en cuenta la diversificación que pueden sufrir, ya que al poder ser almacenados por otros actores bajo sistemas propios, si no están bien protegidos, es más probable que se produzca una fuga de información tras un ataque.

Por tanto se hace patente la necesidad de disponer de métodos que permitan a las entidades proporcionar a sus clientes un entorno seguro tanto para la realización de sus operaciones, como para almacenar, gestionar y transmitir sus datos bancarios con un alto nivel de seguridad, sin olvidar la constante evolución del sector tecnológico que implica una continua revisión y actualización de los protocolos de seguridad adoptados.

Así pues, la existencia del estándar PCI DSS y su constante adaptación se hacen imprescindibles para controlar y minimizar cualquier riesgo asociado al uso de datos sensibles en la banca *online*.

2.3 Requisitos de la norma

Tal y como se especifica en el documento que contiene la norma[3, pág 5], ésta se divide en doce requisitos agrupados a su vez en seis secciones denominadas "objetivos de control":

- **Desarrollar y Mantener una Red Segura:**
  - Requisito 1: Instalar y mantener una configuración de cortafuegos para proteger los datos de los titulares de las tarjetas.
  - Requisito 2: No usar contraseñas del sistema y otros parámetros de seguridad predeterminados provistos por los proveedores.
- **Proteger los Datos de los propietarios de tarjetas:**
  - Requisito 3: Proteger los datos almacenados de los propietarios de tarjetas.
  - Requisito 4: Cifrar los datos de los propietarios de tarjetas e información confidencial transmitida a través de redes públicas abiertas.
- **Mantener un Programa de Gestión de Vulnerabilidades:**
  - Requisito 5: Usar y actualizar regularmente un software antivirus.
  - Requisito 6: Desarrollar y mantener sistemas y aplicaciones seguras.
- **Implementar Medidas sólidas de control de acceso:**
  - Requisito 7: Restringir el acceso a los datos tomando como base la necesidad del funcionario de conocer la información.
  - Requisito 8: Asignar una identificación única a cada persona que tenga acceso a los componentes del sistema.
  - Requisito 9: Restringir el acceso físico a los datos de los propietarios de tarjetas.
- **Monitorizar y probar regularmente las redes:**
  - Requisito 10: Rastrear y monitorizar todo el acceso a los recursos de la red y datos de los propietarios de tarjetas.
  - Requisito 11: Probar regularmente los sistemas y procesos de seguridad.
- **Mantener una Política de Seguridad de la Información:**
  - Requisito 12: Mantener una política que contemple la seguridad de la información.

2.4 Proceso de evaluación de los requisitos de las PCI DSS

Las organizaciones que persigan recibir o mantener la certificación han de seguir un procedimiento de validación que permita analizar si cumplen todos y cada uno de los requisitos. Dada la complejidad de éstos es común que requieran orientación para su implementación. Para ello pueden solicitar ayuda a empresas certificadas por el PCI SSC como:

- **QSA (*Qualified Security Assessors*, Evaluadores de seguridad certificados):**

  Organizaciones autorizadas para asesorar en la implantación de los requerimientos PCI DSS, ayudar en la elaboración del cuestionario de autoevaluación y realizar las auditorías de cumplimiento de los requerimientos PCI DSS.

- **ASV (*Approved Scanning Vendors*, Proveedores aprobados de escaneo):**

  Organizaciones capacitadas para realizar auditorías de vulnerabilidades trimestrales.

Para llevar a cabo el proceso de verificación se establece el siguiente protocolo de actuación:

1. **Confirmación del alcance de la evaluación de las PCI DSS:**

   La entidad ha de determinar qué servicios o componentes de su sistema pueden comprometer la seguridad de datos confidenciales de autenticación o de titulares de tarjeta.

2. **Realizar la evaluación:**

   Se realiza la evaluación de las PCI DSS en la empresa según los procedimientos de pruebas para cada requisito establecido en la norma.

3. **Realizar correcciones en caso de no cumplir algún requisito:**

   Se realizarán tantas veces como sea necesario, hasta que se superen satisfactoriamente todas las pruebas realizadas.

4. **Redactar SAQ (*Self Assessment Questionnaire*, Cuestionario de autoevaluación) o ROC (*Report on Compliance*, Informe sobre cumplimiento):**

   El cuestionario de la evaluación o SAQ está destinado a empresas de menor dimensión que puedan suplir el trámite sin necesidad de recurrir a los servicios de una empresa auditora externa. Este informe hace función de guía para auditar su propio entorno. En la versión 3.1 de la norma, el PCI SSC define nueve modelos en función de los recursos e infraestructura de la organización.

   El informe sobre cumplimiento o ROC contiene el resultado de las auditorías de seguridad y los niveles de cumplimentación de los requisitos marcados por la norma e incluye la documentación de todos los controles de compensación[1]. Este ha de ser emitido y firmado por un QSA o auditor interno independiente del equipo.

[1]Son controles que a pesar de no estar descritos de una forma explícita en el estándar permiten proporcionar un nivel de seguridad similar o superior al control original. Su uso se reserva a circunstancias excepcionales en las que debido a limitaciones técnicas o administrativas sea muy difícil o imposible realizar un control según el estándar.

5. **Redactar la declaración de cumplimiento para proveedores de servicios o comerciantes:**

   Este documento recoge los resultados de los cuestionarios de evaluación y de todas las pruebas realizadas. Ha de estar firmado por un representante de la organización con el fin de garantizar el cumplimiento de los requisitos exigidos en la norma.

6. **Presentar el SAQ o el ROC, así como la declaración de cumplimiento junto a cualquier otro documento solicitado, como por ejemplo los informes de análisis de ASV:**

   Estos documentos deberán presentarse en las organizaciones que lo requieran para demostrar el cumplimiento de la norma.

CAPÍTULO 3
Computación en la nube

En este capítulo se explica el concepto de computación en la nube, profundizando en Azure, el modelo implementado por Microsoft.

3.1 Introducción a la computación en la nube

La evolución de las computadoras y su infraestructura está vinculada estrechamente con los cambios en su uso y las tareas que desempeñan. En sus inicios estas máquinas eran grandes, costosas y de difícil operación, pero a lo largo tiempo han ido aumentando su potencia, fiabilidad y eficiencia a la vez que han reducido su tamaño y coste. Todo ello, sumado al aumento del ancho de banda en Internet, ha permitido cambiar el modelo de tratamiento de la información que se ha venido utilizando en las últimas décadas. Estas mejoras han permitido a empresas tecnológicas tales como Google, Microsoft o Amazon entre otras, disponer de su propia infraestructura de servidores disponibles para cualquier usuario, en cualquier momento, desde cualquier lugar con conexión a Internet y a un coste cada vez mas bajo.

El concepto fundamental que describe la computación en la nube es la entrega de recursos informáticos a través de la red tales como correo electrónico, almacenamiento, aplicaciones web, etc. Estos son ofrecidos y consumidos como servicios a través de Internet de forma transparente a los usuarios finales, que desconocen la infraestructura que opera por debajo.

Según el NIST (*National Institute of Standards and Technology*, Instituto Nacional de Estándares y Tecnología) la computación en la nube[14] o *cloud computing* consiste en un modelo que facilita una red ubicua[1], conveniente y bajo demanda para compartir un conjunto de recursos informáticos configurables (por ejemplo, redes, servidores, almacenamiento, aplicaciones y servicios) que pueden ser asignados rápidamente y desplegados con un esfuerzo mínimo de gestión o interacción por parte del proveedor del servicio. Este modelo de nube se compone de cinco características esenciales, tres modelos de servicio y cuatro modelos de implementación, enumeradas a continuación.

[1]Concepto introducido por Mark Weiser en 1988 y definido como un modelo de interacción en el que el procesamiento de información por parte de los individuos se realiza de forma casi transparente y en cualquier lugar y momento. Hoy en día esto es posible gracias a la cantidad de dispositivos conectados a la red de los que disponemos.

3.1.1. Características esenciales

Las cinco características esenciales que cumple la computación en la nube son las siguientes:

1. **Autoservicio bajo demanda (*On-demand self-service*):**

   Un consumidor puede acceder a los servicios según sea necesario y sin requerir la interacción de un proveedor del servicio.

2. **Amplio acceso a la red (*Broad network access*):**

   Debido a que los servicios están disponibles a través de Internet no es necesario estandarizar dispositivos o servidores específicos. De esta forma los usuarios pueden acceder a los recursos desde cualquier tipo de dispositivo.

3. **Compartición de recursos (*Resource pooling*):**

   Los recursos del proveedor de servicios están agrupados para atender a múltiples consumidores usando un modelo de "tenencia"múltiple (*multi-tenant*)[2], con diferentes recursos físicos y virtuales asignados de forma dinámica de acuerdo a la demanda de los consumidores.

4. **Rápida elasticidad (*Rapid elasticity*):**

   Los recursos pueden ser aprovisionados y liberados de forma escalable[3], incluso de forma automática. Desde el punto de vista del consumidor los recursos pueden parecer ilimitados, pero se adaptan en función de la demanda.

5. **Servicio medido (*Measured service*):**

   Los sistemas en la nube se controlan y optimizan de forma automática aprovechando su constante monitorización. De esta forma solo se paga por los recursos usados, evitando además un desaprovechamiento de recursos energéticos y un gasto innecesario.

3.1.2. Modelos SaaS, IaaS, PaaS

La computación en la nube basa su arquitectura en la separación entre hardware, plataforma y aplicaciones, existiendo tres modelos o niveles de servicios:

1. **SaaS (*Software as service*, *Software* como servicio):**

   Se encuentra en la capa más alta y consiste en la provisión de aplicaciones completas como servicio.

   Un proveedor de servicios, no teniendo porqué ser la misma organización propietaria de la infraestructura, ofrece el *software* como servicio empleando para ello una aplicación que se encarga de mantener y operar. Muchas veces esta aplicación está desarrollada por él mismo.

2. **PaaS (*Platform as service*, Platforma como servicio):**

   Pertenece a la capa intermedia y se centra en un modelo en el que se proporciona un servicio de plataforma con lo necesario para dar soporte al ciclo de vida de

[2] Principio de arquitectura de *software* en el que una única instancia de la aplicación se ejecuta en el servidor pero sirve a múltiples clientes.

[3] Propiedad de un sistema que indica su capacidad de reacción y adaptación sin pérdida de calidad a las circunstancias cambiantes.

aplicaciones y servicios web. Es el proveedor quien se encarga de todos los aspectos de desarrollo, escalabilidad, seguridad, rendimiento, etc. El *hardware* de la capa de infraestructura es transparente a él, por lo que solo ha de preocuparse de sus desarrollos a nivel de *software*.

3. **IaaS (*Infrastructure as service*, Infraestructura como servicio):**

   Se corresponde con la capa de menor nivel de las tres y ofrece almacenamiento básico y capacidad de cómputo como servicio. La diferencia básica entre este modelo y el PaaS radica en que la IaaS permite una mayor flexibilidad en su uso pero requiere un mayor grado de configuración y mantenimiento por parte del cliente.

   Este modelo permite desplazar al proveedor gran parte de la gestión de las máquinas evitando el mantenimiento del *hardware* y con el ahorro en costes al pagar solo por los recursos consumidos.

3.1.3. Modelos de implementación

El Instituto Nacional de Estándares y Tecnología establece cuatro modelos de implementación posibles:

1. **Nube privada (*Private cloud*):**

   Este modelo de infraestructura en la nube se aprovisiona para uso exclusivo de una única organización. Puede ser gestionada por la misma o por terceros, permitiendo administrar internamente los servicios para mantener la privacidad de su información y un control sobre quién puede acceder a dichos servicios.

2. **Nube comunitaria (*Community cloud*):**

   Este modelo de infraestructura en la nube es compartida por varias organizaciones y da soporte a a una comunidad con unas mismas necesidades técnicas. Puede ser gestionada por las propias organizaciones o por terceros.

3. **Nube pública (*Public cloud*):**

   Una infraestructura de nube pública está orientada al uso abierto para el público en general. Puede ser propiedad, operada y gestionada de forma gubernamental, académica, empresarial o una combinación de ellas.

4. **Nube híbrida (*Hybrid cloud*):**

   Este tipo de infraestructura está formada por dos o más tipos de nubes (privada, comunitaria o pública), independientes entre sí pero compatibilizadas a la vez por la tecnología propietaria o algún estándar que permita la portabilidad de la información.

3.2 Computación en la nube como infraestructura en la empresa

Desde un punto de vista empresarial, los beneficios que puede suponer implantar su infraestructura en la nube son elevados. En primer lugar, el ahorro económico es sustancial puesto que se elimina la inversión inicial de la propia infraestructura y su mantenimiento quedando reducido al gasto por uso y tarificación de los proveedores. Por otra parte el consumo energético se ve reducido ya que este modelo de tarificación

se centra en el uso de los recursos premiando por lo tanto el consumo responsable de los mismos. Además, el propio proveedor puede destinar la misma infraestructura para otros clientes o simplemente desactivarla en caso de no ser usada.

También es un aliciente para una empresa el que los proveedores de servicios en la nube dispongan de certificaciones de seguridad, garantizando un control de acceso y segurización de los sistemas. De esta forma se reducen los recursos propios que ha de destinar la organización para proteger parte de su infraestructura.

Por otra parte, los proveedores garantizan la integridad de los datos almacenados de forma que las empresas disponen de un mayor grado de seguridad frente a la corrupción o eliminación accidental de sus datos.

3.3 Microsoft Azure

Microsoft Azure es un entorno gestionado para la ejecución y el despliegue de aplicaciones y servicios en la nube. Es en esencia un modelo PaaS e IaaS de computación en la nube. Desde el 1 de enero de 2010 Microsoft ofrece este servicio al público y desde entonces está en constante evolución, añadiendo y mejorando funcionalidades y servicios para ganar cuota de mercado a los ya grandes del sector, como son Google o Amazon. Como se puede observar en el siguiente gráfico, Azure es uno de los servicios en la nube que más ha crecido durante el año 2015 junto a otras empresas como Amazon o Salesforce:

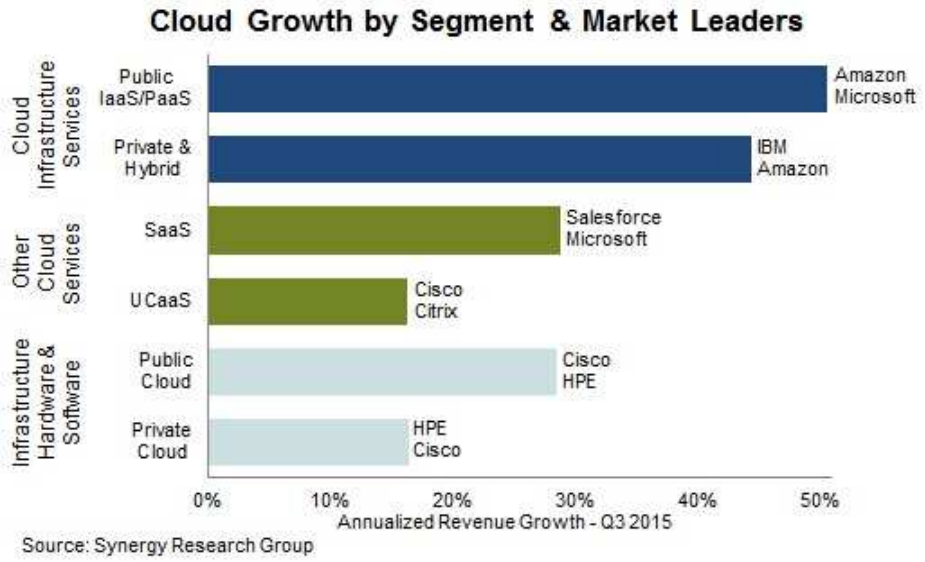

Figura 3.1: Crecimiento de la nube por segmentos y lideres de mercado en 2015.

3.3.1. Operando a nivel de plataforma (modelo PaaS)

El entorno está orientado a desarrolladores y ofrece computación y almacenamiento bajo demanda, despliegue, administración y escalado de sus aplicaciones web. Estas se alojan en los centros de datos de Microsoft distribuidos por todo el globo, a los que se accede vía Internet. Tal y como se ha explicado anteriormente este modelo supone un gran número de ventajas ya que permite reducir el tiempo de lanzamiento de los productos y adaptarse en tiempo real a la demanda de estos en la red. Además, Microsoft garantiza un 99 % o más la disponibilidad de sus servicios junto a la seguridad de acceso

e integridad de los datos. Todo ello supone por tanto una solución muy atractiva para grandes empresas, pymes y desarrolladores independientes. Sin embargo, la ventaja fundamental es la posibilidad de centrarse únicamente en los desarrollos, sin necesidad de montar y mantener su propio centro de datos.

La plataforma permite ejecutar aplicaciones basadas en Windows Server pudiendo estar desarrolladas sobre .NET *Framework* en lenguajes como C# y Visual Basic, o implementadas sin .NET en C++, Java, Ruby, Python y PHP, es decir, lenguajes de programación que sirvan para ejecutar aplicaciones sobre Windows Server 2008, 2008 R2 y 2012. También permite protocolos de Internet como HTTP, XML, REST, SOAP y JSON permitiendo una mayor flexibilidad e interoperabilidad entre distintos desarrollos ya que estos protocolos son estándar.

3.3.2. Operando a nivel de infraestructura (modelo IaaS)

Por otra parte, Microsoft Azure permite además disponer de un mayor control sobre su infraestructura. A pesar de que para el cliente el *hardware* es transparente, se le ofrece la posibilidad de elegir qué máquina virtual utilizar, el tipo de instancia, ya sea GNU/Linux o Windows, la capacidad de cómputo(CPU), la memoria RAM disponible o la capacidad de almacenamiento. En este último caso existen distintos métodos, ya sean motores de bases de datos, almacenamiento de datos no estructurados mediante blobs[4], colas y tablas no relacionales.

3.3.3. Servicios de Microsoft Azure

Azure agrupa sus servicios en distintas categorías, cada una a su vez alberga a uno o más servicios disponibles. A continuación se indican y describen de forma breve todos los servicios de Azure actualmente disponibles. Si se desea profundizar en alguno de ellos puede hacerse a través del siguiente enlace[5].

- **Proceso:**
  - **Máquinas virtuales de Azure:**
    Se ofrece la posibilidad de crear máquinas virtuales a petición, ya sea a partir de una imagen estándar de Microsoft y sus asociados o una suministrada por el propio cliente. Microsoft también ofrece soporte para montar y configurar máquinas virtuales basadas en Linux, entre muchas otras opciones.
  - **Aplicaciones web:** Este servicio ofrece un entorno web administrado a través del portal de administración de Azure y las API[6]. Ofrece flexibilidad para poder agregar o quitar instancias de forma dinámica permitiendo el equilibrio de carga entre ellas. Además, se permite aumentar los recursos destinados a las instancias.

[4]El Almacenamiento de blobs de Azure es un servicio que permite almacenar grandes cantidades de datos de objetos no estructurados, como texto o datos binarios, a los que se puede acceder desde cualquier lugar del mundo a través de HTTP o HTTPS. Para más información véase: `https://azure.microsoft.com/es-es/documentation/articles/storage-dotnet-how-to-use-blobs/`

[5]Servicios Azure: `https://azure.microsoft.com/es-es/services/`

[6]API (*Application Programming Inteface*, Interfaz de programación de aplicaciones), es un conjunto de subrutinas, funciones y procedimientos ofrecidos a través de una librería de funciones para ser utilizadas por otro *software* de forma abstracta.

- **Servicios en la nube:**
  Permite crear aplicaciones escalables de bajo coste administrativo ya que son gestionadas de forma autónoma por los servicios en la nube de Azure. Para ello el cliente desarrolla una aplicación mediante una tecnología compatible y el código se ejecuta en máquinas virtuales o instancias, que ejecutan una versión de Windows Server.
  Existen dos roles a elegir para la creación de una instancia, rol web o rol de trabajo. Se diferencian principalmente en que el rol web dispone de IIS[7] y el rol de trabajo no.

- **Administración de datos:**

  Azure ofrece distintas formas de almacenar y administrar los datos en Azure.

  - **Bases de datos SQL de Azure:**
    Bases de datos SQL ofrece las características de un sistema de administración de bases de datos relacional. Las tareas de administración son gestionadas por la plataforma, incluso la realización de copias de seguridad automáticas y la capacidad de restauración en el momento.
  - **Tablas:**
    Este servicio ofrece almacenamiento no estructurado conocido como almacén clave/valor. Permite almacenar propiedades de tipos como cadenas de texto, enteros y fechas. No permite operaciones complejas como es el caso de las bases de datos SQL, sin embargo, ofrecen acceso rápido a los datos y gran escalabilidad.
  - **Blobs:**
    Lows Blobs de Azure permiten almacenar datos binarios sin estructurar. Facilita un acceso rápido a los datos almacenados y soporta tamaños por Blob de hasta 1 TB(un terabyte).
  - **Importación/exportación:**
    Permite el envío de unidades de disco duro cifradas por Bitlocker[8]
    directamente a centros de datos de Azure. Ya que en ocasiones es necesario trasladar un gran volumen de datos a Azure y su volcado directamente en la red requeriría gran cantidad de tiempo y ancho de banda.
  - **Servicio de archivos:**
    Servicio que permite utilizar el protocolo Bloque de mensajes del servidor (SMB) en la nube. Ello le permite compartir archivos entre distintas máquinas virtuales. Además, también permite el acceso simultáneo a un mismo fichero.
  - **En máquinas virtuales:**
    Permite ejecutar a los clientes su propio sistema de base de datos, además de tecnologías NoSQL[9].

[7] IIS(*Internet Information Server*,) Es un servidor web y servicios para el sistema operativo Microsoft Windows que permite convertir un equipo en un servidor web.

[8] Programa de cifrado de disco de Microsoft que permite proteger todos los datos almacenados en disco duro ya que lo cifra en su totalidad.

[9] Sistemas de gestión de bases de datos que no utilizan el modelo clásico SQL para las consultas. Los datos almacenados no requieren estructuras fijas.

- **Redes:**

  La infraestructura de Azure está repartida en distintos centros de datos alojados en diversas localizaciones por todo el mundo.

  - **Redes virtuales:**
    Las redes virtuales de Azure permiten a los usuarios tratar a sus aplicaciones como si se ejecutaran de forma local.
  - ***Express Route*:**
    El servicio de *Express Route* usa una red virtual de Azure, sin embargo, redirige las conexiones a través de líneas dedicadas más rápidas que no van a través de la red pública de Internet.
  - **Administrador de tráfico:**
    El servicio de administrador de tráfico reparte la carga en la red entre varios centros de datos en caso de que alguna instancia de la aplicación sufra alguna sobrecarga o deje de estar disponible.

- **Servicios de desarrollador:**

  Azure ofrece herramientas de desarrollo para ayudar a crear y mantener aplicaciones en la nube.

  - **SDK de Azure:**
    Los SDK de Azure ayudan a crear, implementar y administrar aplicaciones de Azure. Facilitan el trabajo a los desarrolladores.
  - ***Visual Studio Team Services*:**
    El servicio *Visual Studio Team Services* ofrece un conjunto de herramientas que ayudan a desarrollar aplicaciones en Azure.
  - ***Application Insights*:**
    El servicio de *Application Insights:* supervisa el seguimiento el rendimiento y el uso de las aplicaciones web o de dispositivo.
  - **Automatización:**
    El servicio de automatización permite ejecutar operaciones de forma autónoma y automática, sin necesidad de intervención del usuario. Para ello utiliza *runbooks*[10].
  - **Administración de API:**
    El servicio de administrador de API de Azure facilita a las organizaciones la publicación de API de forma segura y escalable.

- **Identidad y acceso:**

  Azure ofrece servicios para facilitar el seguimiento de la identidad y control de acceso de los usuarios.

  - **Directorio activo:**
    El servicio de directorio activo de Azure permite el inicio de sesión a los usuarios. Les proporciona *tokens*[11] de acceso para las aplicaciones que requieran la validación de su identidad.

[10] *Runbook*: Rutina de compilación de los porcedimientos y operaciones realizados por el administrador del sistema.

[11] Un *token* es una ristra de caracteres alfanuméricos. El proceso de autenticación basado en estos elementos permite evitar tener que mantener la información de la sesión abierta o *cookies*.

- **Autenticación de factor múltiple:**
  Añade a la aplicación la funcionalidad para realizar el proceso de autenticación mediante el uso de dos métodos de verificación de identidad.

- **Móvil:**

  Azure dispone de soluciones para la creación de aplicaciones móviles que facilitan su desarrollo y mantenimiento.

  - **Aplicaciones móviles**:
    Azure facilita la creación de aplicaciones móviles ayudando a almacenar datos en la nube, autenticar a los usuarios, entre otros, sin la necesidad que desarrollar una gran cantidad de código.
  - **Centros de notificaciones:**
    El servicio de centros de notificaciones ofrece la gestión de notificaciones para las aplicaciones que interactúan con dispositivos móviles. Facilita la comunicación de las notificaciones a gran cantidad de usuarios.

- **Copia de seguridad:**

  Las copias de seguridad y su restauración son necesarias en cualquier organización que gestione información. Por ello Azure también ofrece distintos servicios para llevar a cabo estas tareas.

  - **Recuperación de sitios:**
    El servicio de recuperación de sitios permite la protección de aplicaciones mediante la coordinación de la replicación y recuperación entre sitios. Se encarga de supervisar el estado de los servicios de forma continua y facilita la automatización de la recuperación de servicios en caso de interrupción del sitio.
  - **Copia de seguridad de Azure:**
    El servicio de copia de seguridad de Azure realiza copias de seguridad de los datos locales a la nube de Microsoft. Además las copias de seguridad se almacenan de forma segura ya que se cifran antes de transmitirse y se emplean sistemas de protección redundante para evitar la pérdida de datos.

- **Mensajería e integración:**

  Azure ofrece varias formas de solucionar los problemas derivados del tratamiento de datos.

  - **Colas:**
    Las colas permiten una fácil integración entre distintas partes de una aplicación y un fácil escalado.
  - **Bus de servicio:**
    El bus de servicio permite el intercambio de datos entre aplicaciones que se ejecutan en distintas regiones.
  - **Servicio de BizTalk:**
    El servicio de BizTalk permite transformar mensajes XML[12]z en la nube para facilitar la comunicación entre aplicaciones que usen diferentes formatos de mensajería.

[12]XML es un lenguaje de etiquetado extensible orientado a estructurar, almacenar e intercambiar información

- **Asistencia de proceso:**

  Azure ofrece asistencia para los servicios que no necesitan ejecutarse todo el tiempo.

  - **Programador:**
    El servicio de programador permite la programación de trabajos que se ejecutarán de forma automática, en un momento y duración específicos.

- **Rendimiento:**

  Las aplicaciones suelen requerir el acceso a los mismos datos de forma continua. Una solución consiste en ofrecer servicios para el almacenamiento en caché de datos.

  - **Almacenamiento en caché de Azure:**
    El almacenamiento en caché de Azure permite reducir el tiempo de acceso a los datos ya que almacena una copia de aquellos a los que se tiene acceso con mayor frecuencia.
  - **Red de entrega de contenido:**
    La red de entrega de contenido permite copiar los datos de un blob en distintos sitios de todo el mundo a modo de caché. De esta forma se consigue un acceso más rápido a los datos ya que se reparte la carga de trabajo para su descarga.

- ***Big Data* y *Big Compute*:**

  El volumen de datos que gestionan las empresas es cada vez más grande. El procesamiento de esa información requiere técnicas avanzadas en paralelización y gran potencia de cálculo. Para ello Azure ofrece soluciones a nivel de servicio para ello.

  - ***HDInsight:***
    El servicio de HDInsight ayuda con el procesamiento masivo de grandes cantidades de datos.
  - **Informática de alto rendimiento:**
    La informática de alto rendimiento o HPC ofrece la posibilidad de gestionar los recursos de Azure como un clúster para el tratamiento de información de forma distribuida.

- **Multimedia:**

  El servicio multimedia es una plataforma para aplicaciones que abastecen de contenido multimedia a clientes en todo el mundo.

- **Tienda de Azure:**

  La tienda de Azure permite buscar y comprar aplicaciones o conjuntos de soluciones comerciales de Azure e integrarlos en las aplicaciones.

3.3.4. Administración de Microsoft Azure

Existen distintas formas y herramientas para administrar Azure. La más compleja consiste en realizar todas las operaciones a través de PowerShell o línea de comandos. Esto es así ya que requiere conocer los comandos y argumentos necesarios para su uso correcto. Sin embargo, es la única forma de acceder al 100 % de los servicios ya que al publicarse una nueva actualización o servicio no siempre están disponibles desde el portal web, pero sí desde PowerShell o la línea de comandos.

La otra forma de administrar Azure consiste en los portales web, el clásico y el nuevo. Ambos funcionan conjuntamente ya que algunos servicios, como por ejemplo el directorio activo de Azure, no están disponibles en ambos portales. En este caso solo es funcional en el clásico. Su uso es más sencillo e intuitivo y la tendencia es que vayan migrando todos los servicios al nuevo portal, pero, mientras se implementa todo ambos conviven y es necesario acceder a los dos para realizar muchas de las gestiones. Sin embargo, la implementación de este proyecto se ha realizado mediante el uso de PowerShell debido a que aún no se podían realizar todas las operaciones sobre el almacén de claves de Azure a través de ninguno de los portales. A continuación se muestra una captura del portal clásico frente a otra del nuevo.

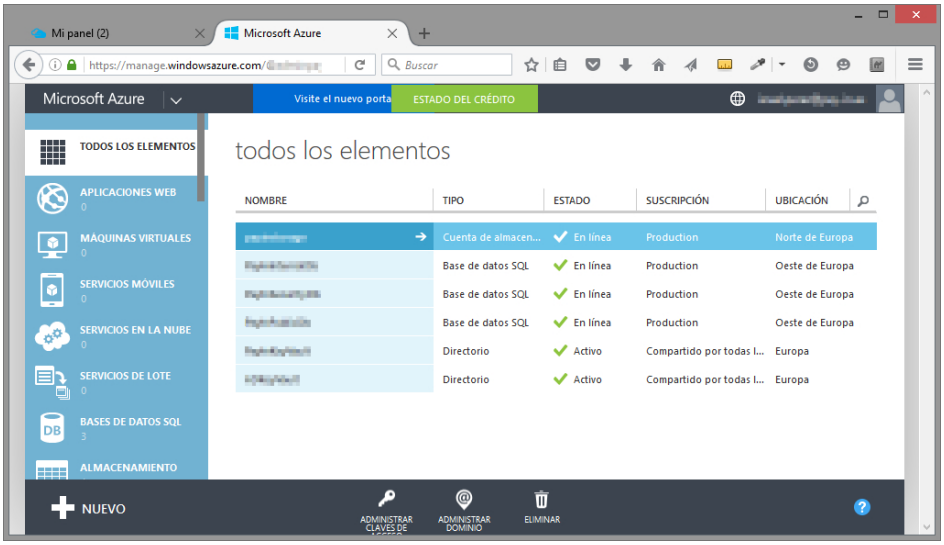

Figura 3.2: Captura del portal clásico de Azure.

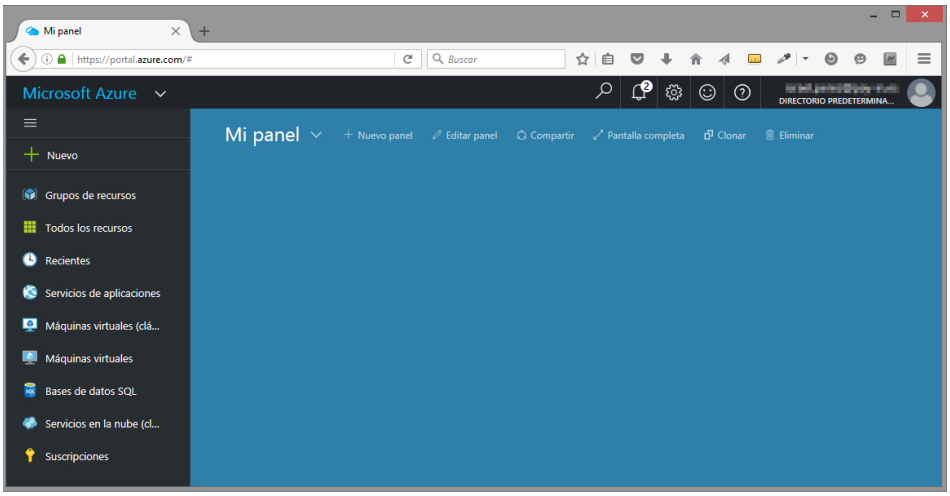

Figura 3.3: Captura del nuevo portal de Azure.

CAPÍTULO 4

Análisis de los requisitos de implementación de la norma PCI DSS en Azure

En este capítulo se analizan los puntos de la norma que no cumple la infraestructura en la nube de Microsoft o requieren nuestra gestión.

4.1 Situación actual

Anualmente, Microsoft Azure auditoriza sus productos para certificar el cumplimiento de las PCI DSS. Actualmente dispone de la certificación para la versión 3.1 como proveedor de servicio de nivel 1, correspondiente con un volumen superior a 6 millones de transacciones anuales. El periodo de vigencia del certificado es de un año, a contar desde la fecha de la firma de del documento de declaración de cumplimiento (AoC, *Attestation Of Compilance*)[1]. Este documento es de acceso público y se encuentra en el enlace a pie de página[2]. En este caso el archivo comprimido contiene varios documentos AOC ya que la constante evolución de Azure obliga a realizar revisiones de la norma sobre los nuevos servicios que se van desarrollando, con el fin de ofrecer estos servicios certificados en mayor brevedad posible.

La fecha en la que se hizo efectiva la superación de los requisitos es del día 30 de junio de 2016, por lo que es previsible que para una fecha cercana al día 30 de junio de 2017 se lleve a cabo una nueva auditoría para validar la última versión de la norma.

Puesto que la versión actual de la norma PCI DSS (versión 3.2) es superior a la que acredita Azure, se hace necesario decantarse por la aplicación de una de las dos. En nuestro caso y debido a que los recursos requeridos por el proyecto se despliegan en su totalidad en Azure, se decide aplicar la versión que éste certifica.

En el inicio de este proyecto la última versión acreditada por Azure era la 3.0, sin embargo, la revisión de la norma a la versión 3.1 no supone grandes cambios, ya que en el documento en el que se detalla el sumario de cambios solo se indican aclaraciones y pequeñas modificaciones sobre los puntos de la versión anterior.

[1] La declaración de cumplimiento es un formulario para los comerciantes y proveedores de servicios en el que se recogen los resultados de los cuestionarios de autoevaluación y cualquier otro tipo de pruebas realizadas. El documento requiere la firma de un representante de la organización garantizando el cumplimiento de los requisitos exigidos en la norma.

[2] `https://gallery.technet.microsoft.com/Azure-PCI-DSS-Responsibilit-02d4b4b2`

Toda la información relativa a las distintas versiones de la norma, así como las modificaciones entre las distintas versiones se encuentra accesible a través de la web de la PCI SSC en el enlace a pie de página[3].

4.2 Competencias sobre la aplicación de los requisitos

A pesar de que Microsoft Azure está certificada en el cumplimiento de la norma, ello no exime a Pay[In] su responsabilidad de cumplir e implementar por su cuenta o de forma conjunta algunos de los requisitos. Dada la naturaleza del proyecto y a pesar de que el despliegue se realiza sobre la infraestructura de Azure, también se dispone de una aplicación móvil desarrollada sobre la plataforma Android, totalmente independiente de Azure. También, muchos de los requisitos de la norma tienen como objetivo establecer y controlar todos los aspectos relacionados con las comunicaciones, protocolos de actuación, proceso de desarrollo, monitorización de vulnerabilidades constante, entre otros, por lo que se hace imperativo que la empresa se implique activamente.

Existe una guia (Guía PCI del cliente de Windows Azure)[21] en la que se especifica qué puntos de los requisitos han de ser aplicadas íntegramente por la empresa, cuáles por parte de Microsoft y cuáles de forma conjunta. A pesar de su utilidad, este documento ya no está disponible en su página[4] y no existe una versión suficientemente actualizada. La versión de la norma PCI DSS a la que hace referencia es la 2.0 y está fechada en enero de 2014. Es por ello que se hace necesario un análisis entre las distintas versiones para detectar los cambios y, en caso de no estar contemplados, tomar una decisión respecto a su implementación. *A priori*, y en caso de no existir un criterio, se opta por la implementación propia. Sin embargo, es fácil detectar si es o no necesaria ya que la responsabilidad de la empresa está muy acotada dado que toda la infraestructura y despliegue, exceptuando la aplicación móvil, están alojados en la nube de Microsoft.

4.3 Comparación de versiones

Tal y como se ha explicado en el punto anterior, la guía en la que se indica qué parte interesada ha de encargarse de unos u otros requisitos, tiene su última actualización en la versión 2.0 de la norma. Ello implica la comprobación de cada uno de los cambios realizados entre esta y la siguiente versión, con el fin de determinar si los nuevos puntos que se hayan añadido han de ser gestionados por la empresa. En primer lugar es necesario comprobar qué cambios se han realizado entre las versiones 2.0 y 3.0. Para ello se estudian el documento con la versión 2.0 de la norma y el sumario de cambios entre esta versión y la 3.0. Cabe destacar que en dicho sumario solamente se enumeran los puntos que han sufrido alguna modificación, ya sea actualización de requisitos, nueva incorporación o eliminación por ambigüedad o desuso.

A pesar de que se va a revisar la norma en su versión 3.0, ésta ya se considera obsoleta, por lo que será una mera operación de transición para actualizar la guía del cliente a una versión más actualizada.

Al inicio del proyecto de Pay[In], se llevó a cabo una auditoría independiente para conocer el estado general de la empresa respecto al cumplimiento de las PCI DSS. Aunque el contenido de este informe es de carácter confidencial, ofrece una visión general sobre el cumplimiento o no de los requisitos de la norma en su versión 3.0. En él

[3]Véase documentación de la norma PCI DSS: `https://es.pcisecuritystandards.org/document_library`

[4]https://azure.microsoft.com/es-es/blog/announcing-pci-dss-compliance-and-expanded-iso

se contempla la infraestructura y funcionalidad de la plataforma Pay[In], de esta forma se acotan las responsabilidades de implementación de algunos requisitos por parte de la empresa. Este documento también se ha tenido en cuenta a la hora de realizar el análisis y comparación entre las versiones 2.0 y 3.0.

El resultado de este análisis se ha adjuntado a modo de listado en el apéndice B.1. En él se especifican únicamente aquellos puntos de cuya implementación ha de encargarse Pay[In], ya sean de carácter cooperativo o individual. Y además se indica si el punto en cuestión ha sido modificado respecto a la versión anterior o añadido nuevo.

Por último, se revisa el sumario de cambios entre las versiones 3.0 y 3.1 de la norma[22], con el fin de determinar si hay alguna modificación sustancial que requiera añadir o eliminar algún requisito cuya implementación dependa de Pay[In]. Como resultado de esta operación se observa que la mayoría de cambios son aclaraciones del texto sin entrar en una modificación de los requisitos. Por lo tanto, los requisitos listados en el apéndice B.1 se consideran actualizados a la versión 3.1 de la norma. Cabe destacar que la función del listado es la de determinar los puntos de la norma que ha de implementar Pay[In], sin entrar al nivel técnico de su implementación. Por ello y aunque los requisitos puedan cambiar sutilmente de una versión a otra no es el objeto de este análisis. Se obvian las aclaraciones de los requisitos en posteriores versiones hasta realizar un estudio previo a su implementación.

CAPÍTULO 5
Implementación de la norma

En este capítulo se explican las medidas adoptadas para cumplir los requisitos especificados de la norma. En cada sección se detalla el funcionamiento de las tecnologías empleadas, así como el proceso de implementación y despliegue en la plataforma.

5.1 Implementación y configuración del módulo de almacenamiento de claves criptográficas de Azure

El requisito 3.5 de la norma establece que han de documentarse e implementarse los procedimientos que protejan las claves utilizadas para, en última instancia, proteger los datos del titular de la tarjeta almacenados contra su posible divulgación o uso indebido.

Por lo tanto el primer problema a resolver es dónde se pueden almacenar las claves que cifran y descifran los datos bancarios. Este medio ha de ser seguro y debe cumplir con la norma PCI DSS, además de permitir un control exhaustivo de acceso al mismo. Un segundo problema subyacente es el proceso a seguir para el almacenamiento de las claves en el medio, ya que este ha de ser seguro y ha de poder ser evitado cualquier tipo de fuga de información durante dicho procedimiento. De nada servirá disponer de un soporte de información seguro si el propio proceso de introducción de los datos no lo es.

La solución adoptada consiste en generar y almacenar las claves de cifrado en el almacén de claves criptográficas de Azure (*Key Vault*). El cuál,tal y como se ha explicado anteriormente, es un módulo de la plataforma de servicios en la nube de Azure y cumple con el nivel 1 de la norma PCI DSS, contemplando en su diseño las necesidades de seguridad requeridas. En las siguientes secciones se ahonda en las propiedades y funcionamiento de este almacén, incluyendo su implementación y despliegue en la plataforma Azure.

5.1.1. Funcionamiento del almacén de claves criptográficas de Azure

El almacén de claves de Azure[23] está diseñado para almacenar y proteger claves criptográficas y secretos[1]. Para ello cifra estos datos sensibles mediante claves asimétricas protegidas por módulos de seguridad de hardware (HSM[2]).

Figura 5.1: Logotipo del almacén de claves de Azure (*Key Vault*).

Este sistema permite la independencia entre los procesos de cifrado y operaciones criptográficas y los realizados por el software desarrollado en la empresa, ya que opera de forma transparente e interna.

El control de acceso está basado en AAD (*Azure Active Directory*, directorio activo de Azure), siendo necesario el registro previo de un usuario o aplicación en la plataforma para tener acceso a los recursos del almacén. De esta forma se permite gestionar conexiones de distintos usuarios al almacén de claves, con la posibilidad de asignar diversos permisos a cada cuenta.

Además, poder asignar estos permisos de acceso permite que terceros puedan gestionar sus claves y secretos de forma autónoma. Esto añade un grado de confidencialidad y autonomía aumentando a su vez la seguridad ya que solo ellos podrán acceder a sus propios recursos.

Acerca de las claves y secretos

Es necesario distinguir entre el concepto de clave y secreto ya que el Almacén de claves los trata de distinta forma ofreciendo un mayor o menor número de operaciones posibles a realizar sobre ellos.

En el caso de las claves criptográficas el almacén solo admite algoritmos y claves RSA permitiendo el uso de módulos HSM[3], aunque en versiones futuras se podrán incorporar otros tipos de clave como simétricas o de curva elíptica. Existen dos posibilidades de gestión para las claves RSA, por un lado se encuentra la considerada clave "débil" que consiste en una clave RSA de 2048 bits que se procesa mediante *software* por el almacén de claves, pero se cifra para su almacenamiento mediante una clave del sistema que se encuentra en un HSM. La otra posibilidad es que sea una clave RSA-HSM procesada directamente por un HSM y siendo protegida en uno de los espacios de seguridad del HSM del almacén de claves de Azure.

Una de las características que dotan a este sistema de gran independencia y seguridad es la posibilidad de generar claves RSA de forma totalmente transparente al

[1]Cualquier tipo de contraseña, clave de autenticación, clave de cuenta de almacenamiento, clave de cifrado y archivo en formato .pfx.

[2]Dispositivo criptográfico basado en *hardware* que genera, almacena y protege claves criptográficas. Al estar basado en *hardware* ofrece un alto rendimiento en operaciones de cálculo con números primos de gran tamaño.

[3]HSM (*Hardware Security Module*", Módulo de Seguridad Hardware). Es un dispositivo criptográfico basado en hardware que genera, almacena y protege claves criptográficas y suele aportar aceleración hardware para operaciones criptográficas ya que está diseñado específicamente para esa función.

administrador, ya que se crean internamente en el almacén y nunca se puede acceder a la parte privada de la clave. Existe la posibilidad de exportar la clave pero nunca de forma que pueda ser usada fuera del sistema.

Las operaciones que se pueden realizar sobre las claves son las siguientes:

- **Creación (*Create*):**

  Permite la creación de una clave en el almacén de claves de Azure. El valor de la clave se genera de forma aleatoria por el propio almacén y se almacena sin ser entregada.

- **Importación (*Import*):**

  Permite la importación de una clave existente al Almacén de claves de Azure.

- **Actualización (*Update*):**

  Permite la modificación de los metadatos de la clave.

- **Eliminación:**

  Permite la eliminación de la clave.

- **Listado (*List*):**

  Permite enumerar todas las claves de un determinado almacén. Únicamente se muestran los atributos públicos, en ningún caso se mostrará la clave en si misma.

- **Listado de versiones (*List versions*):**

  Permite listar las versiones de una misma clave.

- **Obtención (*Get*):**

  Permite obtener la parte pública de una clave determinada.

- **Respaldo (*Backup*):**

  Permite la exportación de la clave en un formato protegido. Este formato impide su uso fuera del sistema del Almacén de claves de Azure, puesto que el material devuelto está cifrado por un HSM del almacén o el propio almacén.

- **Restauración (*Rrestore*):**

  Permite la importación de una clave desde una copia de seguridad.

Una vez creada la clave, se pueden realizar las siguientes operaciones:

- **Firma y verificación (*Sign and verify*):**

  Consiste en la firma o comprobación de una función *hash*[4]. Actualmente el almacén no permite la generación propia del hash, por lo que es necesario generarlo de forma local.

- **Cifrado/encapsulado de clave (*Key encryption / wrapping*):**

  Una clave ya almacenada en el sistema puede utilizarse para cifrar otra clave.

[4]Algoritmo matemático que transforma cualquier conjunto aleatorio de datos en una nueva serie de caracteres con longitud fija. Se considera una función resumen que compacta la cadena de entrada a un valor fijo, con la cualidad de no poder recuperar el conjunto de datos originales.

- **Cifrado y descifrado (*Encrypt and decrypt*):**

  Una clave ya almacenada puede emplearse para cifrar o descifrar un bloque de datos. Cabe destacar que el tamaño de los datos se determina por el tipo de clave y el algoritmo de cifrado empleado.

Los secretos, por otra parte, son secuencias de octetos de tamaño máximo 25.000 Bytes. No se requiere ningún tipo de estructura para los datos, siendo almacenados de forma segura siempre y cuando cumplan con el límite de tamaño máximo. En caso de que los datos sean confidenciales la responsabilidad del cifrado recae sobre el cliente, puesto que no se realiza ninguna operación sobre la información almacenada.

Las operaciones que se pueden realizar sobre los secretos son las siguientes:

- **Creación (*Set*):**

  Permite la creación de un nuevo secreto con la información aportada.

- **Obtención (*Get*):**

  Permite la obtención de un secreto.

- **Listado (*List*):**

  Permite la enumeración de los secretos de un determinado Almacén de claves de Azure.

- **Eliminación (*Delete*):**

  Permite la eliminación de un secreto.

Todas las operaciones están ligadas a los permisos asignados a través del control de acceso, existiendo una notable diferencia entre la versatilidad de las claves y la simplicidad de los secretos.

Las claves y secretos se gestionan de forma independiente, existiendo un control de acceso distinto para cada tipo. La recuperación de los secretos se realiza de forma individual mediante un enlace generado de forma automática. Cada uno tiene un identificador único, así como una versión asignada. Si se realiza alguna modificación se crea una nueva versión del objeto, pasando a ser esta última la versión activa.

5.1.2. Implementación del Almacén de claves de Azure

Existen distintos métodos para crear y administrar almacenes de claves. A través de CLI (*Command line*, Línea de comandos), PowerShell[5] o mediante una plantilla de ARM[6] (*Azure Resource Manager*, administrador de recursos de Azure). En un futuro también se podrán llevar a cabo estos procedimientos de forma gráfica a través del portal web de Azure[7], pero por el momento aún no ha sido desarrollado por Microsoft. En nuestro caso, se ha elegido la implementación mediante PowerShell ya que en el momento de iniciar el desarrollo era el único entorno que permitía una interacción con Azure, además de la generación de *scripts* para su posterior exportación.

[5] Lenguaje de *scripting* e intérprete de línea de comandos creado por Microsoft basado en tareas y diseñado para la administración de sistemas.

[6] El administrador de recursos de Azure permite trabajar con los recursos desplegados en Azure de forma agrupada. Para la implementación se emplean una o varias plantillas que almacenan la información de configuración y despliegue de la infraestructura. La sintaxis básica de la plantilla es JSON pero se emplean además funciones y expresiones que permiten una mayor flexibilidad.

[7] `https://portal.azure.com/`

El motivo por el que estos *scripts* han de poder ser exportados a otro computador es la necesidad de realizar el despliegue en un entorno seguro, ya que este proceso requiere la introducción de valores de configuración sensibles a los que únicamente han de tener acceso los administradores.

Para poder trabajar de una forma cómoda, se utiliza el entorno de *script* integrado de PowerShell o PowerShell ISE, que permite ejecutar comandos, escribir, comprobar y depurar *scripts*. Además, es necesaria la instalación de Azure PowerShell, conjunto de módulos que ofrece *cmndlet* para administrar Azure desde esta plataforma. Puesto que no es objeto de este proyecto profundizar en el funcionamiento de esta herramienta y en el lenguaje PowerShell, se dan por supuestos conceptos como la propia sintaxis del lenguaje o la configuración del sistema para su correcto funcionamiento, aunque pueda puntualizarse algún dato relevante en caso de considerarse necesario. En caso de requerir mayor información sobre este lenguaje y la configuración del entorno se puede acceder a [25, 26].

Los *scripts*

El proceso de desarrollo se divide en dos grupos de operaciones reflejadas en dos *scripts* distintos. Por un lado se dispone la configuración del servicio, es decir, la creación del almacén de claves y la generación de las mismas. Por otro lado, se configura la cuenta de aplicación en el directorio activo de Azure para controlar el acceso al almacén de claves.

Para poder realizar dichas operaciones es necesario que un administrador de la cuenta de Microsoft Azure o alguien con privilegios suficientes se autentifique con su usuario y contraseña. Esto permite establecer una conexión segura entre la máquina de despliegue y Azure. Para realizar esta conexión, ambos *scripts* ejecutan el *cmndlet* de autentificación de un usuario en Azure: ”Login-AzureRmAccount".

Con el fin de evitar la modificación de cualquier despliegue previo así como de informar del estado del sistema, previamente a la realización de cada una de las operaciones se verifica que no exista ya un recurso con las mismas características.

Por otra parte, los valores de los parámetros para los *cmdlet*[8] se introducen a través de variables. De esta forma es mucho más sencillo ya que solo será necesario añadirlos al inicio del *script* una única vez. En otras ocasiones las variables permiten pasar valores generados entre unos *cmdlets* y otros sin necesidad de intervenir en el proceso. Cabe destacar que las variables en PowerShell van precedidas por el caracter "$".

Los *scripts* completos se encuentran en el apéndice A y están basados en el código de muestra facilitado por Microsoft[28].

[8]Un *cmdlet* es un comando utilizado en el entorno de Windows PowerShell. Para más información véase también [27].

Operaciones para la creación y configuración del servicio

- **Creación del grupo de recursos**[9]

```
$rgExists = Get-AzureRmResourceGroup `
            -Name $resourceGroupName `
            -ErrorAction SilentlyContinue
if (-not $rgExists)
{
    New-AzureRmResourceGroup `
        -Name $resourceGroupName `
        -Location $location
}
else
{
    Write-Host "Resource group $resourceGroupName exists!"
}
```

En caso de que no exista un grupo de recursos se crea uno nuevo mediante el *cmdlet* "New-AzureRmResourceGroup" introduciendo los siguientes parámetros:

- **Nombre (*Name*):**
  Nombre único para todo nuestro despliegue en Azure con el que distinguiremos el grupo de recursos.
- **Región (*Location*):**
  La región es el emplazamiento del centro de datos en el que se va a gestionar el recurso. Ha de elegirse de entre todas las disponibles teniendo en cuenta la distáncia y disponibilidad, ya que no todos los servicios pueden ser desplegados en cualquier región. En nuestro caso elegimos Europa Occindetal.

- **Creación del almacén de claves:**

```
$vaultExists = Get-AzureRmKeyVault `
               -VaultName $vaultName `
               -ErrorAction SilentlyContinue
if (-not $vaultExists)
{
    Write-Host "Creating vault $vaultName"
    $vault = New-AzureRmKeyVault `
               -VaultName $vaultName `
               -ResourceGroupName $resourceGroupName `
               -Sku premium `
               -Location $location
}
else
{
   Write-Host "The Key Vault $vaultName exists!"
}
```

La creación del almacén de claves requiere el *cmdlet* "New-AzureRmKeyVault". Los parámetros más relevantes son los siguientes:

- **Nombre del almacén (*VaultName*):**
  Nombre único para todo nuestro despliegue en Azure con el que distinguiremos el almacén de claves,

[9]Estructura lógica en Microsoft Azure que contiene los recursos que se le asignen. Ello permite administrar varios recursos a la vez. Todo recurso o conjunto de ellos ha de estar contenido en uno.

- **Nombre del grupo de recursos (*ResourceGroupName*):**
  Nombre de un grupo de recursos existente, en nuestro caso el creado anteriormente.
- **Familia de SKU (*Stock-keeping unit*):**
  Este parámetro indica el nivel de servicio del almacén de claves que se usará. Existen dos niveles, el servicio *standard*, que permite secretos y claves protegidas mediante software, o el servicio *premium*, que permite la compatibilidad para claves protegidas con HSM. En nuestro implementación seleccionamos el nivel *premium* para poder utilizar la implementación con claves protegidas mediante HSM.

- **Creación de la clave:**

```
$keyExists = Get-AzureKeyVaultKey
                -Name $keyName `
                -VaultNAme $vaultName
if(-not $keyExists) {
    Write-Host "Setting key $keyName in vault $vaultName using HSM
        destination"
    $key = Add-AzureKeyVaultKey `
            -VaultName $vaultName `
            -Name $keyName `
            -Destination HSM
}
else
{
   Write-Host "The Key $keyName in vault $vaultName exists!"
}
```

Para la creación de la clave RSA se emplea el *cmdlet* "Add-AzureKeyVaultKey" cuyos parámetros son:

- **Nombre del almacén de claves (*VaultName*):**
  Contiene el nombre identificador del almacén de claves en el que queremos crear la clave.
- **Destino (*Destination*):**
  Permite seleccionar si la clave va a estar protegida mediante *software* o HSM. En nuestro caso y debido a que hemos seleccionado previamente el "SKU"*premium* en la creación del almacén, seleccionamos el tipo "HSM".

- **Generación de la dirección de acceso a la clave:**

```
Write-Host "Place the following into both CSCFG files for the
    SampleAzureWebService project:" -ForegroundColor Cyan

'<App name="KeyUrl" value="' + $key.Id.Substring(0, $key.Id.LastIndexOf
    ('/')) + '" />'
```

Por último se construye en formato XML la dirección web de la clave generada en el almacén de claves. Este dato será necesario para cuando se configure la aplicación en la parte del servidor.

Operaciones para la configuración de la aplicación en el directorio activo de Azure

- **Creación de la cuenta de aplicación en el directorio activo de Azure:**

```
if(-not $applicationPassword)
{
  $applicationPassword = GenerateSymmetricKey
}
$identifierUri = "https://webdelaaplicacion/Account/Login"
$homePage = "https://webdelaaplicacion.es"

Write-Host "Creating a new AAD Application"
$ADApp = New-AzureRmADApplication `
            -DisplayName $applicationName `
            -HomePage $homePage `
            -IdentifierUris $identifierUri `
            -Password $applicationPassword

Write-Host "Creating a new AAD service principal"
$servicePrincipal = New-AzureRmADServicePrincipal `
                    -ApplicationId $ADApp.ApplicationId
```

Para poder acceder al almacén de claves es necesario disponer de una cuenta en AAD ya que la autenticación pasa por verificar las credenciales de una cuenta de este tipo.

La creación de la aplicación en el directorio activo de Azure se realiza mediante el *cmdlet* "New-AzureRmADApplication". Puesto que la cuenta que se va a crear es de una aplicación, se necesitan los siguientes parámetros:

- **Nombre a mostrar (*DisplayName*):**
  Nombre de la aplicación a mostrar.
- **Página principal (*HomePage*):**
  Página web principal a la que hace referencia la cuenta de aplicación.
- **Identifiacores de recursos uniforme (*IdentifierUris*):**
  Especifica un conjunto de URI (*Uniform Resource Identifiers*, identificadores de recursos uniforme) para la aplicación.
- **Contraseña (*Password*):**
  Especifica la contraseña de la cuenta de aplicación.

Para que la cuenta recién creada en AAD pueda ser utilizada desde un servicio de Azure es necesario que se vincule a una cuenta de servicio de inicio de sesión. Esto se realiza mediante el *cmdlet* "New-AzureRmADServicePrincipal" y es necesario introducir como parámetro el identificador de la cuenta de aplicación de AAD, *ApplicationId*. En nuestro caso lo obtenemos de la variable resultado generada tras la creación de la cuenta de AAD accediendo al valor *ApplicationId*.

Existe un procedimiento más seguro que el uso de una clave simétrica y consiste en el uso de un certificado que confirme la identidad del usuario que crea la cuenta en AAD. Para ello es necesario importar el certificado en formato *.pfx y sustituir el parámetro *Password* por los siguientes:

- **Certificado a importar (*KeyValue*):**
  Contiene la clave privada del certificado.
- **Tipo de clave (*KeyType*):**
  Especifica el tipo de clave.

- **Fecha de inicio (*StartDate*):**
  Indica la fecha de inicio de validez del certificado.
- **Fecha de fin (*EndDate*):**
  Indica la fecha de fin de validez del certificado.

Este procedimiento se llevará a cabo cuando se disponga de un certificado válido de empresa. Por el momento y mientras se realizan los trámites para su obtención, las pruebas de implementación han sido realizadas utilizando únicamente una contraseña generada de forma automática mediante la siguiente función:

```
Function GenerateSymmetricKey()
{
    $key = New-Object byte[](32)
    $rng = [System.Security.Cryptography.RNGCryptoServiceProvider]::
        Create()
    $rng.GetBytes($key)
    return [System.Convert]::ToBase64String($key)
}
```

- **Autorización de la aplicación y administrador para que puedan usar las claves y secretos:**

```
# Specify full privileges to the vault for the application
Write-Host "Setting access policy"
Set-AzureRmKeyVaultAccessPolicy -VaultName $vaultName `
      -ObjectId $servicePrincipal.Id `
      -PermissionsToKeys all `
      -PermissionsToSecrets all

# Specify full privileges to the vault for the Azure administrator
if($administratorID)
{
    Write-Host "Setting access policy for Azure administrator"
    Set-AzureRmKeyVaultAccessPolicy -VaultName $vaultName `
         -ObjectId $administratorID `
         -PermissionsToKeys all `
         -PermissionsToSecrets all
}
else
{
    Write-Host "administratorId not exists, access policy for Azure
        administrator is not set"
```

Con el fin de realizar operaciones sobre las claves o secretos es necesario disponer de los permisos necesarios. Para ello se emplea el *cmdlet* "Set-AzureRmKeyVaultAccessPolicy". Los parámetros son los siguientes:

- **Nombre del almacén de claves (*VaultName*):**
  Se corresponde con el nombre del almacén de claves sobre el que van a aplicarse los permisos.
- **Identificador de objeto (*ObjectId*):**
  Es el identificador de objeto único para todo Azure que identifica de forma inequívoca la cuenta de usuario o servicio. En nuestra implementación se asignan permisos a dos cuentas, una la de aplicación y otra la del administrador de la suscripción de Azure.
  En el caso del identificador de la cuenta de la aplicación, este es accesible desde la variable que contiene el objeto (*$servicePrincipal*) ya que previamente se han guardado los resultados de la operación de creación en ella. Para

acceder al identificador del administrador es necesario utilizar el *cmdlet* "Get-AzureRmADUser" en un terminal PowerShell. Este comando muestra un listado de los usuarios disponibles y su correspondiente identificador de objeto.

- **Permisos para claves (*PermissionsToKeys*):**
  Especifica un conjunto de permisos que pueden ser asignados a las claves.
- **Permisos para secretos (*PermissionsToSecrets*):**
  Especifica un conjunto el conjunto de permisos que pueden ser asignados a los secretos.

- **Generación de las credenciales de acceso para el proyecto web:**

```
1 Write-Host "Paste the following settings into the web.config file for the
      HelloKeyVault project:" -ForegroundColor Cyan
2 '<add key="HSMKeyVaultUrl" value="' + $vault.VaultUri + '"/>'
3 '<add key="HSMClientId" value="' + $servicePrincipal.ApplicationId +
     '"/>'
4 '<add key="HSMClientSecret" value="' + $applicationPassword + '"/>'
```

En la finalización del *script* se construyen las clave-valor con los parámetros de configuración necesarios para establecer la comunicación con el almacén de claves. Éstas han de añadirse al fichero "web.config"del proyecto.

5.1.3. Despliegue en Azure

Como planteamiento previo al despliegue, se asume que la computadora desde la que se va a llevar a cabo la operación está en un entorno seguro al que solo pueden tener acceso los administradores. Aunque en este TFG las demostraciónes se realizan a través de una máquina física, el despliegue que se llevará a cabo para la aplicación final del desarrollo en la empresa se realizará desde una máquina virtual en la cuenta de Azure destinada a producción. A esta máquina virtual se accederá a través de escritorio remoto. Para más información sobre como llevar a cabo esta operación puede accederse a[29].

El procedimiento para realizar el despliegue consiste en seguir los siguientes pasos:

Configuración de los *scripts*

En primer lugar es necesario asignar los valores que necesitemos en las variables de los *scripts* puesto que por motivos de seguridad no se completan hasta que no es estrictamente necesario. En el caso del *script* para la configuración de la aplicación de Azure Active Directory las variables son:

- **vaultName:**
  Nombre del nuevo almacén de claves de Azure.
- **resourceGroupName:**
  Nombre del nuevo grupo de recursos.
- **applicationName:**
  Nombre de la cuenta de aplicación del directorio activo de Azure.

- **administratorID:**

  Número de identificación de la cuenta de administrador. Este identificador se puede obtener desde el terminal de PowerShell utilizando el *cmdlet* "Get-AzureRmADUser". El único motivo por el que se necesita este identificador de objeto es para poder asignar permisos con el fin de permitir la gestión del almacén de claves desde fuera de la aplicación de Azure Active Directory. Esto es así debido a que la aplicación solamente podrá realizar un número limitado de operaciones, y en caso de ser necesario modificar, crear, actualizar o eliminar el almacén de claves se necesita de un usuario con permisos suficientes. Para ello, en la asignación de privilegios del *script* se deberán asignar los permisos necesarios en vez todos, como se realiza de forma predeterminada.

El *script* de configuración del servicio incluye una variable más:

- **keyName:**

  Nombre de la nueva clave.

A continuación se muestran capturas de los dos *scripts* en las que se indica la configuración de los valores de las variables.

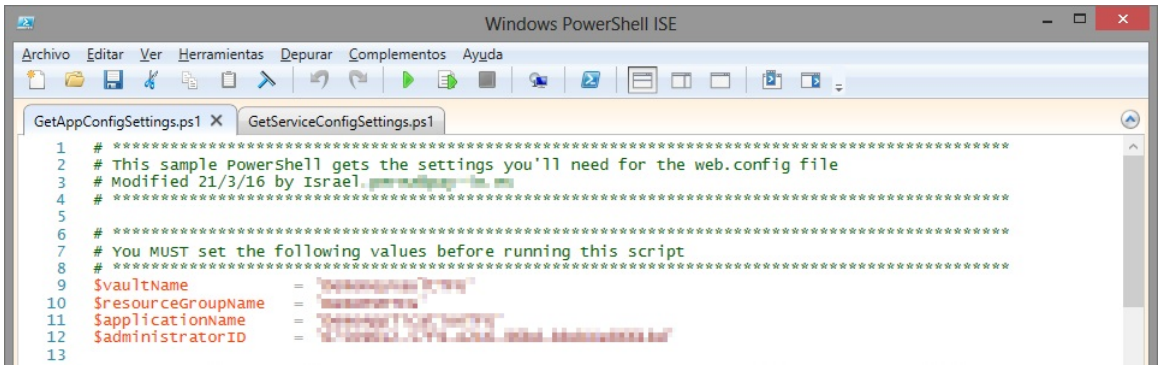

Figura 5.2: Configuración de parámetros en el *script* de aplicación.

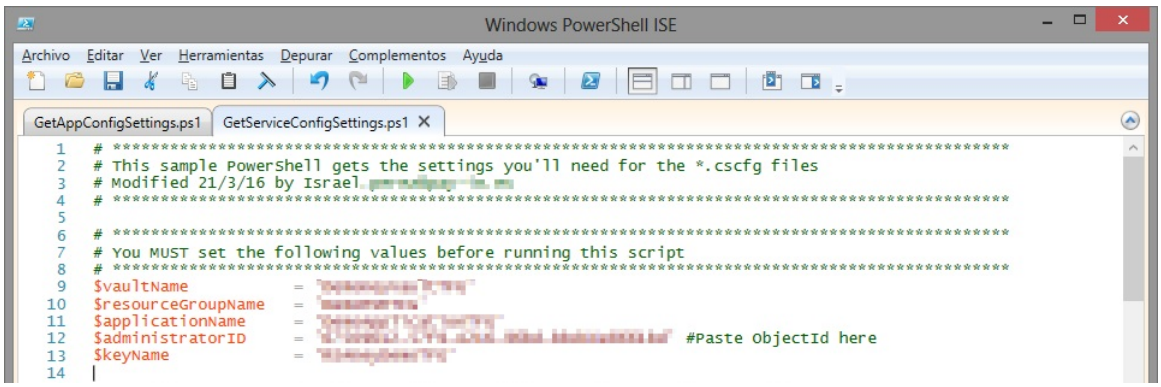

Figura 5.3: Configuración de parámetros en el *script* de configuración del servicio del almacén de claves.

Ejecución del *script* para la configuración del servicio

Este es el *script* correspondiente a la creación del almacén de claves, en nuestro caso tiene el nombre "GetServiceConfigSettings.ps1". La ejecución puede realizarse directamente desde el menú contextual pulsando con el botón derecho del ratón sobre el mismo y seleccionando la entrada “Ejecutar en PowerShell".

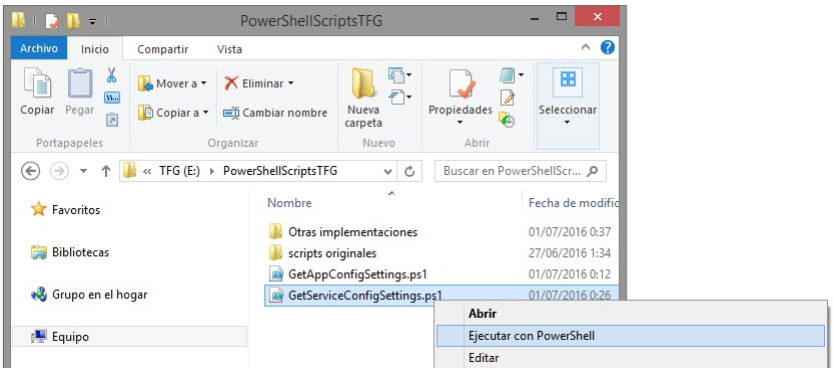

Figura 5.4: Ejecución de *script* desde menú contextual.

Sin embargo, para llevar a cabo la ejecución se utiliza el entorno de *script* integrado de Windows Powershell ya que se necesita comprobar la salida por terminal para controlar el proceso de ejecución y los valores generados como resultado. En caso de que se haya producido algún error será mostrado en el terminal. En caso de realizarse de forma satisfactoria será en este mismo terminal en el que se muestren los valores requeridos para configurar la aplicación .Net que usará el servicio del almacén de claves.

Por lo tanto, para ejecutar el *script* abrimos el fichero desde la herramienta y lo ejecutamos pulsando sobre el botón *play* de color verde o simplemente pulsando la tecla de función "F5".

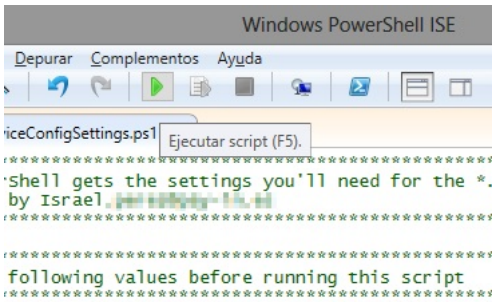

Figura 5.5: Captura del botón de ejecución de *scripts* en el entorno integrado de *scripting* de PowerShell.

Nada más iniciar la ejecución, el entorno nos solicita las credenciales de la cuenta de administrador de Azure para poder continuar el proceso de despliegue. Tras la introducción del usuario, el cuál se corresponde con el correo electrónico asociado a la cuenta de Azure, se solicita el tipo de cuenta. En nuestro caso se corresponde con una cuenta personal ya que el tipo de cuenta profesional o educativa está vinculada directamente con Microsoft y no nuestro caso. Si nunca se ha autenticado en Microsoft Azure desde la máquina en la que se ejecutan los *scripts* se deberán introducir dos veces las credenciales, ya que en la primera se asociará una suscripción de Azure y en la última se realizará el proceso de identificación del usuario.

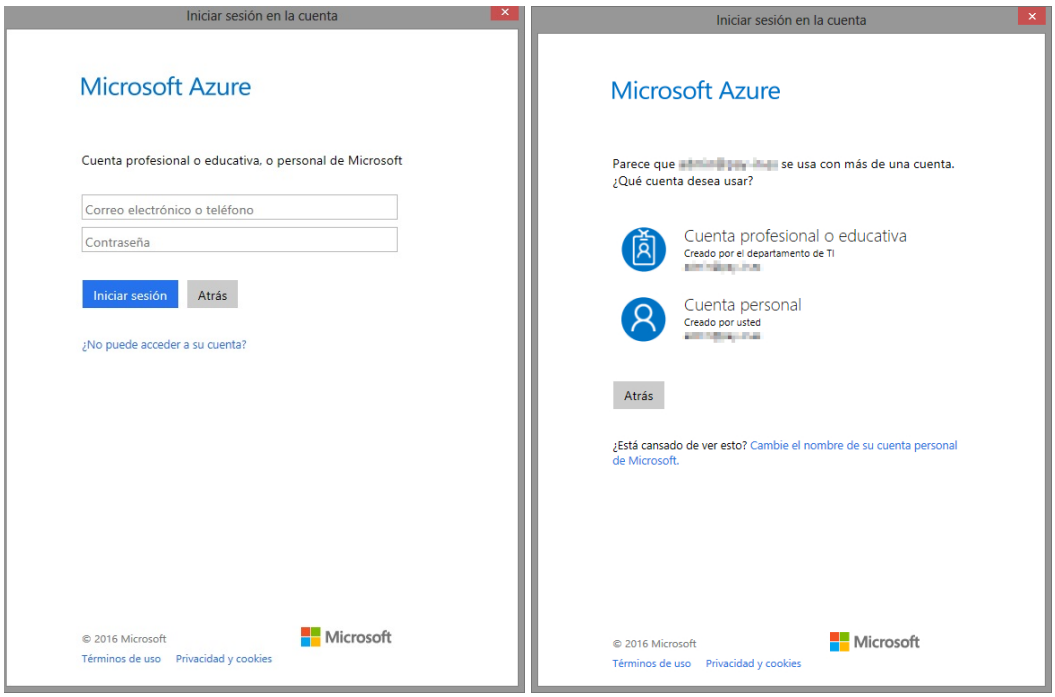

Figura 5.6: Ventanas de inicio de sesión de Microsoft Azure.

Tras la introdución de las credenciales el proceso continúa normalmente. En cuyo caso y si no ocurre ningún problema, el terminal mostrará un mensaje similar al siguiente:

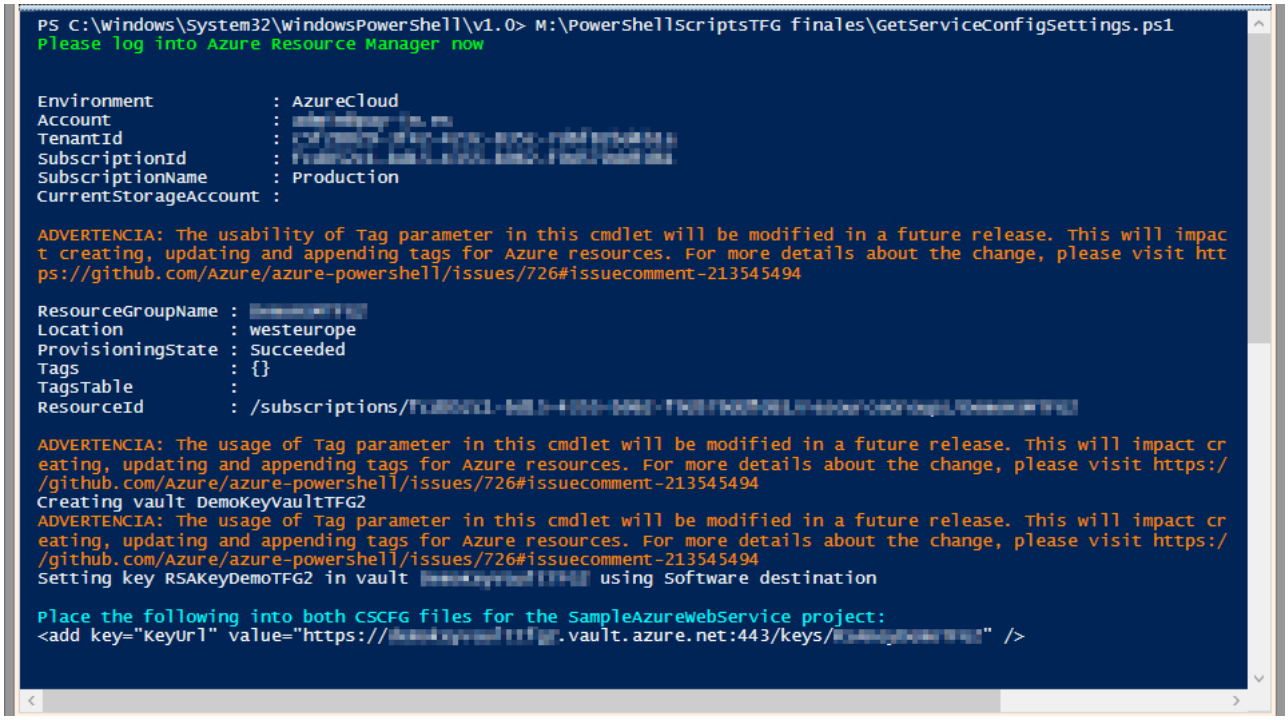

Figura 5.7: Resultado de la ejecución del *script* de configuración del almacén de claves

Es necesario almacenar la ruta que se ha generado ya que será necesaria en el proceso de configuración de la aplicación .Net. Esta ruta contiene la dirección web del almacén de claves.

Ejecución del *script* para la cuenta de aplicación

El procedimiento para la ejecución del segundo *script* es idéntico al primero, a excepción de los resultados que se muestren en el terminal.

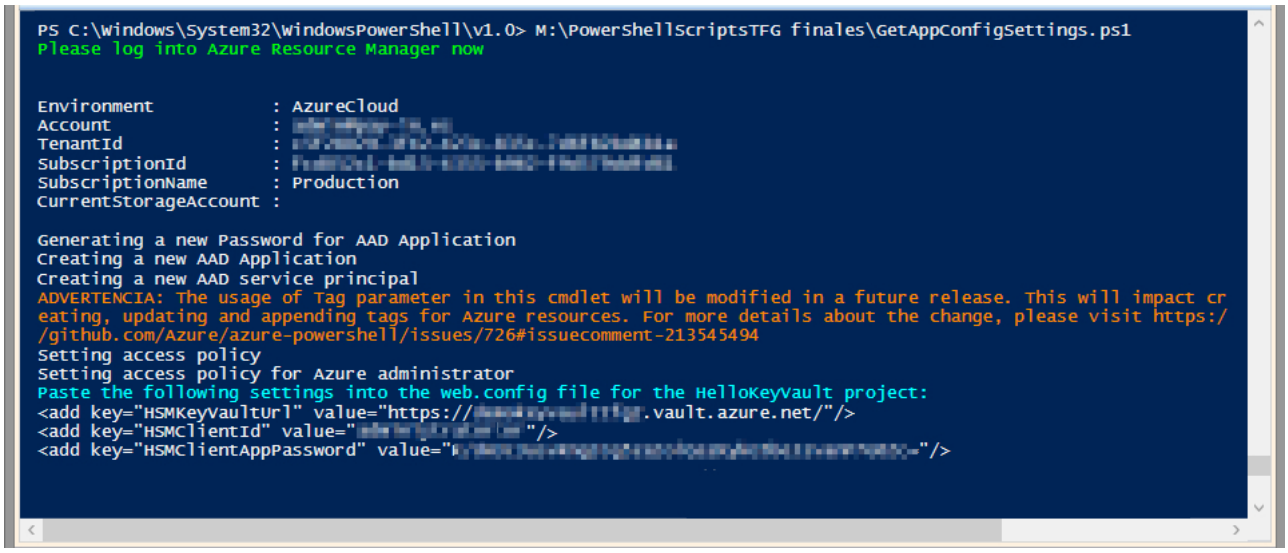

Figura 5.8: Resultado de la ejecución del *script* de configuración de la aplicación

En este caso se generan tres claves, de las cuales se han de almacenar las correspondientes a la contraseña de la aplicación, *HSMClientAppPassword* y al identificador del cliente, *HSMClientId*. Esta tupla identificador/contraseña permite la acreditación a través de la aplicación de Azure Active Directory, generada previamente en el *script*.

Configuración de la aplicación .NET

Debido a la política de empresa no es posible mostrar algunas partes del código implementado, sin embargo, el procedimiento es general y puede explicarse correctamente.

Puesto que el desarrollo está basado en roles web de Azure, el fichero de configuración que almacena los parámetros de conexión se encuentra en el directorio principal del proyecto. De nombre "Web.config", almacena información en formato XML.

Es en este fichero en el que hemos de añadir las dos instrucciones obtenidas durante la ejecución de los *scripts*. Ambas han de estar dentro de la jerarquía de la etiqueta <*appSettings*>.

Una vez almacenadas las credenciales en el documento XML es necesario acceder a ellas desde la clase encargada de invocar las operaciones del almacén de claves. Para ello se utiliza la función *ConfigurationManager.AppSettings*, la cual permite acceder al recurso contenido en el fichero de configuración XML y obtiene el valor cuya clave coincida con el argumento introducido en la función:

```
1 vaultUrl = ConfigurationManager.AppSettings["HSMKeyVaultUrl"];
2 var clientId = ConfigurationManager.AppSettings["HSMClientId"];
3 var clientSecret = ConfigurationManager.AppSettings["HSMClientAppPassword"];
```

Para que el proyecto en .Net se comunique con el almacén de claves necesita establecer una conexión con el directorio activo de Azure. Para ello, mediante la función

GetADToken que obtiene un *token* de autenticación válido, llama a la función *AcquireTokenAsyn* de la clase *AuthenticationContext*[10] usando como parámetros el destinatario del *token* y la aserción del cliente que se utilizará para su adquisición. Este *token* no es más que una firma cifrada que permite al servidor identificar al usuario.

```
#region GetADToken
  private async Task<string> GetADToken(string authority, string resource,
      string scope)
  {
    var authContext = new AuthenticationContext(authority);
    ClientCredential clientCred = new ClientCredential(
      ConfigurationManager.AppSettings["HSMClientId"],
      ConfigurationManager.AppSettings["HSMClientAppPassword"]);
    AuthenticationResult result = await authContext.AcquireTokenAsync(resource,
        clientCred);

    if (result == null)
      throw new InvalidOperationException("Failed to obtain the JWT token");

    return result.AccessToken;
  }
#endregion GetADToken
```

El resultado de esta función es el *token*, parámetro requerido por la función *AuthenticationCallback* para establecer la conexión con el almacén de claves. Función que pertenece a la clase *KeyVaultClient*[11]

```
keyVaultClient = new KeyVaultClient(new KeyVaultClient.AuthenticationCallback(
    GetADToken));
```

De esta forma se establece una comunicación directa con el almacén de claves permitiendo realzar las operaciones cuyos permisos de la cuenta de aplicación permitan. A continuación se muestra el código de las funciones que permiten cifrar, descifrar y firmar:

Función de cifrado

Para realizar la operación de cifrado se pasan como parámetros de la función el texto a cifrar y el nombre de la clave. Internamente la función llama al método *EncryptAsync*, encargado de solicitar la operación al almacén de claves pasando como parámetros el texto a cifrar, el tipo de algoritmo que en nuestro caso es RSA-OAEP (RSA-*Optimal asymmetric encryption padding*, RSA con relleno de cifrado asimétrico óptimo) y el texto codificado en *Base64*[12].

```
#region EncryptKeyOperationResult
  public async Task<KeyOperationResult> EncryptKeyOperationResult(string text,
      string keyName)
  {
    var keyUrl = ConfigurationManager.AppSettings[keyName];
    if (keyUrl == null)
      throw new Exception("Key Url don't exists in App.Config file");

    var algorithm = JsonWebKeyEncryptionAlgorithm.RSAOAEP;
    var textBytes = Encoding.ASCII.GetBytes(text);
```

[10]Documentación de las clases de la librería *IdentityModel.Clients.ActiveDirectory*: `https://msdn.microsoft.com/es-es/library/microsoft.identitymodel.clients.activedirectory.aspx`

[11]Documentación de la clase *KeyVaultClient*: `https://msdn.microsoft.com/es-es/library/azure/microsoft.azure.keyvault.keyvaultclient.aspx`

[12]La codificación en base 64 consiste en el uso de 4 caracteres imprimibles (en formato US-ASCII).

```
10      //var plainText = Convert.ToBase64String(textBytes);
11
12      var operationResult = await keyVaultClient.EncryptAsync(keyUrl, algorithm,
            textBytes);
13
14      return operationResult;
15    }
16  #endregion EncryptKeyOperationResult
```

Función de descifrado

Por otra parte la operación de descifrado es similar a la de cifrado a excepción de que el resultado es justo la operación inversa, en este caso, dado el nombre de la clave de cifrado y el texto cifrado se devuelve dicho texto descifrado.

```
1  #region DecryptKeyOperationReturn
2    public async Task<KeyOperationResult> DecryptKeyOperationReturn(string text,
         string keyName)
3    {
4      var keyUrl = ConfigurationManager.AppSettings[keyName];
5      if (keyUrl == null)
6        throw new Exception("Key Url don't exists in App.Config file");
7
8      var keyBundle = keyVaultClient.GetKeyAsync(keyUrl).Result;
9      var algorithm = JsonWebKeyEncryptionAlgorithm.RSAOAEP;
10     var textByte = Convert.FromBase64String(text);
11
12     var operationResult = await keyVaultClient.DecryptDataAsync(keyBundle,
           algorithm, textByte);
13
14     return operationResult;
15   }
16 #endregion DecryptKeyOperationReturn
```

Función de firma

Por último, la operación de firma se encarga de obtener una clave pública del resumen *hash*[13] de un texto dado en función de una de las claves almacenadas en el almacén de claves de Azure.

```
1  #region SignKeyOperationResult
2    public async Task<KeyOperationResult> SignKeyOperationResult(string text,
         string keyName)
3    {
4      //Devuelve el hash firmado del texto con la clave privada keyName
5
6      var keyUrl = ConfigurationManager.AppSettings[keyName];
7      if(keyUrl == null)
8        throw new Exception("Key Url don't exists in App.Config file");
9
10     var algorithm = JsonWebKeySignatureAlgorithm.RS256;
11     var digest = GetSHA256Hash(text);
12
13     var operationResult = await keyVaultClient.SignAsync(keyUrl, algorithm,
           digest);
```

[13]Una función *hash* o de resumen es un algoritmo que permite obtener a partir de un unos datos una salida alfanumérica de longitud generalmente fija. Esta cadena representa un resumen de los datos originales y no es posible revertir el proceso para obtener de nuevo el conjunto de datos originales a través del *hash* obtenido. No es un método de cifrado de datos, sin embargo, permite asegurar la integridad de los datos.

```
14
15      return operationResult;
16    }
17  #endregion SignKeyOperationResult
```

Cabe destacar que el almacén de claves requiere que la función *hash* se ejecute en el lado del cliente ya que por el momento esta funcionalidad no está desarrollada. Para ello utilizamos una función basada en la librería de .Net, *Security.Cryptography.SHA256CryptoServiceProvider*, que permite obtener este valor con facilidad:

```
1  #region GetSHA256Hash
2    private byte[] GetSHA256Hash(string text)
3    {
4      SHA256CryptoServiceProvider provider = new SHA256CryptoServiceProvider();
5      var textBytes = Encoding.UTF8.GetBytes(text);
6      var hashBytes = provider.ComputeHash(textBytes);
7
8      return hashBytes;
9    }
10 #endregion GetSHA256Hash
```

5.1.4. Pruebas de funcionamiento

Una vez configurado todo el proyecto es el momento de realizar las comprobaciones oportunas para determinar su correcto funcionamiento. Para ello se llevan a cabo unas pruebas de funcionamiento basadas en un proyecto de prueba[28] que permite conectarse a un almacén de claves y probar todas sus funciones.

Tras la ejecución del programa se lanza una interfaz de línea de comandos a modo de testigo de todas las operaciones que se van realizando. De forma predeterminada se ejecutan todas las operaciones posibles sobre el almacén de claves. Este proyecto de prueba también contiene los *scripts* originales de PowerShell a partir de los cuales se han programado los nuestros.

A pesar de que el proyecto es capaz de conectarse por si mismo al almacén de claves tras su configuración, se ha necesitado modificar parte para probar la clase que se utilizará en nuestra implementación. Esta clase contiene funciones propias desarrolladas por Pay[In] para gestionar las operaciones de acceso y manipulación del almacén de claves de Azure.

La modificación ha supuesto intercalar esta clase en las llamadas a las funciones del almacén de claves con el fin de verificar el correcto paso de argumentos y devolución de los resultados de las operaciones de forma correcta y en el formato requerido.

Tras la ejecución del programa se muestran los resultados de las operaciones en la interfaz de línea de comandos, en la que se puede observar que se han realizado tres opearaciones.

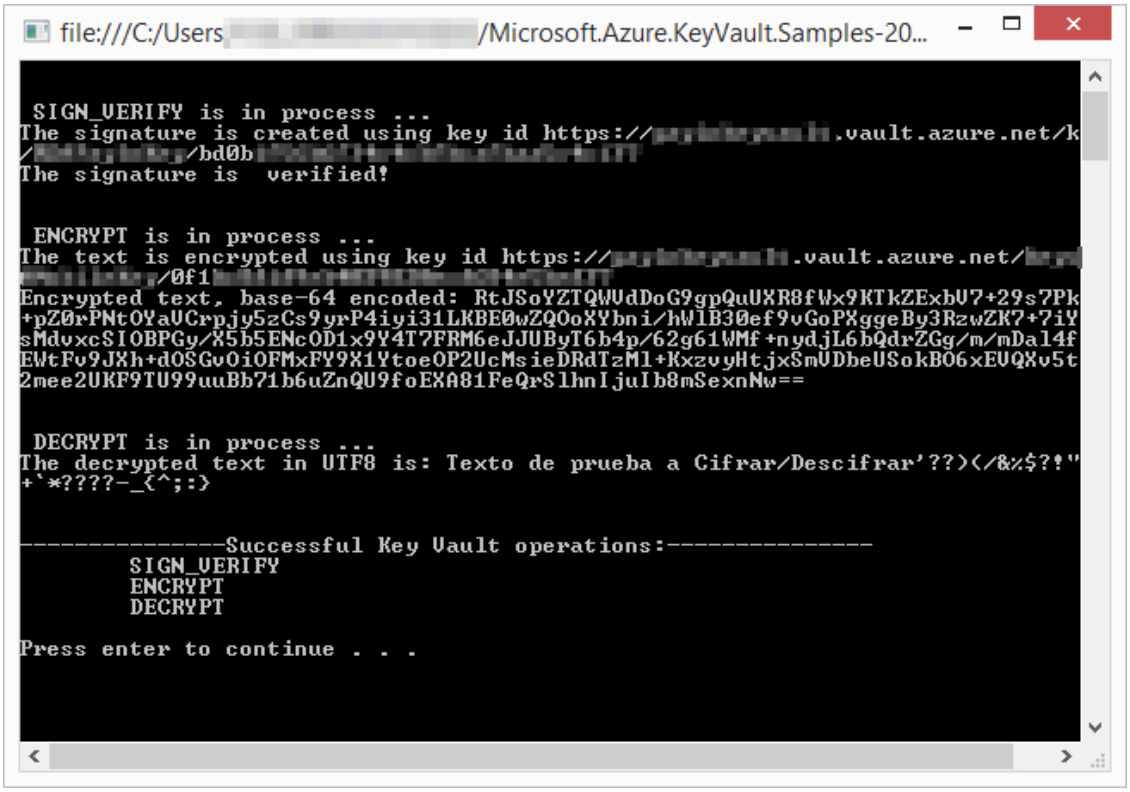

Figura 5.9: Resultado de la ejecución del proyecto de prueba. Interfaz de línea de comandos.

- La primera de ellas consiste en la operación de firma. El programa prueba el correcto funcionamiento del método *SignKeyOperationResult*. Para ello se ejecuta el método y se verifica el resultado mediante el uso de la función *ValidateKeyOperationResult*. Esta última se encarga de comprobar que el *hash* resultado de la función *SignKeyOperationResult* se ha obtenido a partir de los datos originales y la clave almacenada en el almacén de claves.

  En caso de que el resultado de la operación sea *true* se muestra el mensaje "*The signature is verified!*". A continuación se muestra el código de la función de validación:

```
#region ValidateKeyOperationResult
    public async Task<bool> ValidateKeyOperationResult(string text, string sign
        , string keyName)
    {
      // Se debe devolver comprobar que el hash firmado del texto con la clave
          publica keyName coincide con la firma que se pasa
      var keyUri = keyVaultClient.GetKeyAsync(keyName).Result.Key.Kid;
      var algorithm = JsonWebKeySignatureAlgorithm.RS256;
      var digest = GetSHA256Hash(text);
      var signBytes = Convert.FromBase64String(sign);

      return await keyVaultClient.VerifyAsync(keyUri, algorithm, digest,
          signBytes);
    }
#endregion ValidateKeyOperationResult
```

- La segunda es la operación de cifrado en la que se muestra el resultado en la interfaz de línea de comandos en codificación base 64.

- La última operación es el descifrado de la operación anterior en la que se devuelve como resultado el texto resultado de la operación de descifrado. Si la operación ha sido exitosa el texto devuelto ha de ser idéntico al introducido en la operación de cifrado.

5.2 Transmisión de datos cifrados entre móvil y servidor

Tras la implementación y configuración del almacén de claves de Azure en este apartado se explica el procedimiento empleado para transferir de forma segura los datos entre el servidor y el dispositivo móvil.

Con el fin de evitar el acceso a los datos bancarios de los usuarios a través de las redes públicas se ha decidido usar cifrado de claves asimétricas para hacerlos ilegibles a no ser que se disponga de la clave privada para descifrar la información.

La comunicación entre la aplicación móvil y el *backend* de nuestro despliegue en Azure ya utiliza el protocolo HTTPS[14]. Sin embargo, se ha considerado necesario cifrar estos datos para asegurar su inaccesibilidad incluso una vez han llegado al servidor y son tratados.

Los datos del usuario se deberán introducir por primera y única vez en la aplicación móvil durante el proceso de activación de una tarjeta bancaria. Es en ella en la que se cifrarán mediante la clave pública de una firma RSA almacenada en el almacén de claves de Azure y accesible únicamente por el *backend* de nuestra solución.

Por otra parte ha de considerarse que esta comunicación ha de ser bidireccional puesto que también es necesario transmitir datos sensibles desde el servidor a la aplicación móvil. Por lo tanto es necesaria otra clave RSA alojada en el dispositivo móvil. La implementación del apartado móvil no se considera en este TFG por lo que aunque se va a explicar de forma teórica no se mostrará código ni pruebas de funcionamiento al respecto. Este análisis se centra en el apartado servidor, concretamente en la importación de claves RSA para permitir el cifrado y descifrado de la información transmitida hacia el dispositivo móvil.

Los motivos por los que se ha elegido el uso de cifrado asimétrico son varios, por un lado partimos de la robustez del cifrado, pese a que a pesar de suponer una mayor carga de proceso y tiempo de ejecución, al no requerirse de forma constante, no llega a ralentizar la comunicación. El otro motivo es la imposibilidad de utilizar otro tipo de cifrado en el almacén de claves de Azure ya que aún está en proceso de desarrollo y por el momento solo soporta ese tipo de cifrado.

[14] *HTTPS (Hypertext Transfer Protocol Secure, Protocolo seguro de transferencia de hipertexto)*

5.3 Importación de una clave al almacén de claves

Para poder cifrar los datos del lado del *backend* y enviarlos al dispositivo móvil, el servidor ha de disponer de la parte pública de la clave de cifrado y el dispositivo móvil la clave privada.

Por lo tanto, partiendo de que la clave RSA se genera en el móvil ha de importarse al servidor. Para ello se emplea el *script* cuyo contenido está disponible en el anexo A.3.

A continuación se explica de forma breve su funcionamiento:

En primer lugar hay que completar los valores de las variables que se utilizarán como parámetros de la función principal, siendo los siguientes:

- **vaultName:** Contiene el nombre del almacén de claves al que se va a importar la clave.
- **mobileKeyName:** Nombre de la clave RSA.
- **mobileKeyFPath:** Ruta de acceso al fichero que contiene la clave.
- **password:** Contraseña con la que se ha cifrado el fichero de la clave.
- **outFilePath:** Ubicación en disco del fichero que contendrá la ruta web a Azure de la clave importada.

Tras la configuración de las variables y asegurarnos de que el fichero con la clave está correctamente ubicado, se procede con la ejecución del *script*. Cabe destacar que el formato del fichero que contenga la clave a importar ha de ser ".pfx" ya que es el único formato que acepta el almacén de claves. Tras la ejecución, si el proceso ha ido bien, se mostrará la ruta de la nueva clave importada que tendremos que agregar en el fichero "Web.Config" del proyecto .Net.

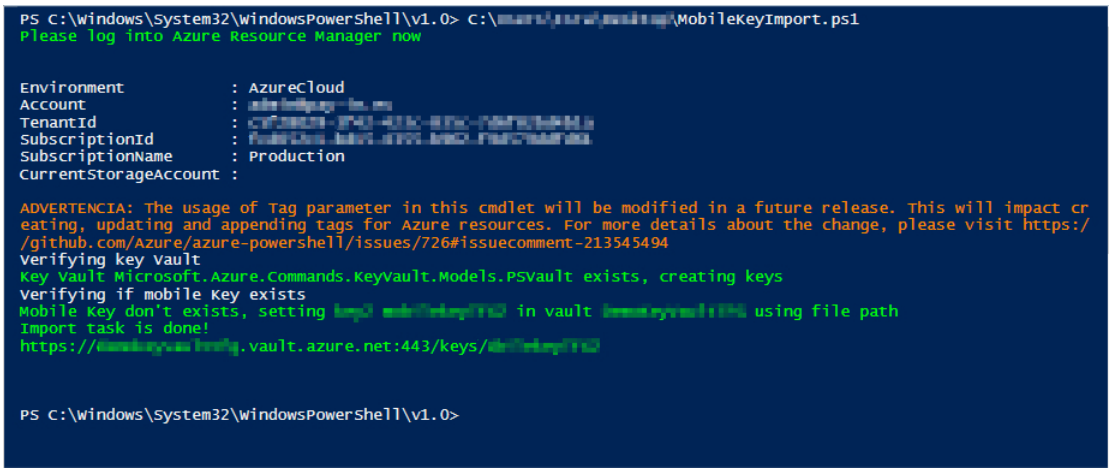

Figura 5.10: Resultado de la ejecución del *script* de importación.

De esta forma, podemos cifrar los datos sensibles que necesitemos enviar al móvil sabiendo que solo en él se podrán descifrar, ya que solamente el dispositivo móvil dispone de la clave privada.

5.4 Integración de la herramienta Latch

Con el fin de tener un mayor control sobre la máquina virtual que dará acceso al entorno de producción de Pay[In], se ha decidido integrar la herramienta de seguridad Latch.

Figura 5.11: Logotipo de Latch

Latch es una herramienta desarrollada por la empresa Eleven Paths[15], cuyo presidente es el reconocido *hacker* español, Chema Alonso[16].

El principal problema de un control de acceso tradicional es que requiere únicamente el uso de credenciales de acceso para entrar en el sistema. Existen otros métodos más sofisticados como la autenticación en dos pasos[17], ya sea mediante el uso de un *token* de seguridad RSA por *hardware* o a través de un dispositivo móvil en el que se recibe un código de acceso. El problema radica en que todos estos elementos pueden ser accesibles en algún momento, aunque sea por un corto periodo de tiempo, por lo que pueden llegar a conocerse ambas credenciales.

La función principal de Latch es la de añadir una capa de seguridad adicional al entorno sobre el que se implanta, a modo de pestillo virtual. El el usuario puede deshabilitar la cuenta vinculada a través de un dispositivo móvil en caso de no querer utilizarla. De esta forma, aunque se conozcan las credenciales de acceso, el servicio denegará la operación impidiendo así un acceso no autorizado Además desde Latch no se gestiona ni almacena ninguna información sobre la identidad del usuario.

El requisito de la norma PCI DSS al que hace referencia esta mejora es el 7.1. En él se indica que ha de limitarse el acceso a los componentes del sistema y a los datos del titular de la tarjeta a aquellos individuos cuyas tareas requieran dicho acceso.

5.4.1. Funcionamiento

El funcionamiento de Latch es el siguiente. El usuario, mediante la aplicación móvil, puede bloquear o desbloquear la cuenta que tenga asociada. Cada vez que realiza un intento de acceso a la cuenta en el equipo "latcheado" el *plugin* instalado en él realiza una consulta a los servidores de Latch para verificar el estado del servicio. Si el estado es activo permite el acceso al equipo, de lo contrario deniega el acceso y avisa al usuario para que cambie la contraseña en la mayor brevedad posible.

[15]Eleven Paths: `https://www.elevenpaths.com/es/index.html`

[16]Chema Alonso: `https://www.elevenpaths.com/es/quienes-somos/nuestro-equipo/chema-alonso/index.html`

[17]Proceso de verificación de identidad en dos procedimientos distintos. El primero suele ser mediante usuario y contraseña, el segundo suele mediante un código de acceso enviado a un dispositivo móvil. Para poder entrar en la cuenta asociada es necesario introducir ambas credenciales.

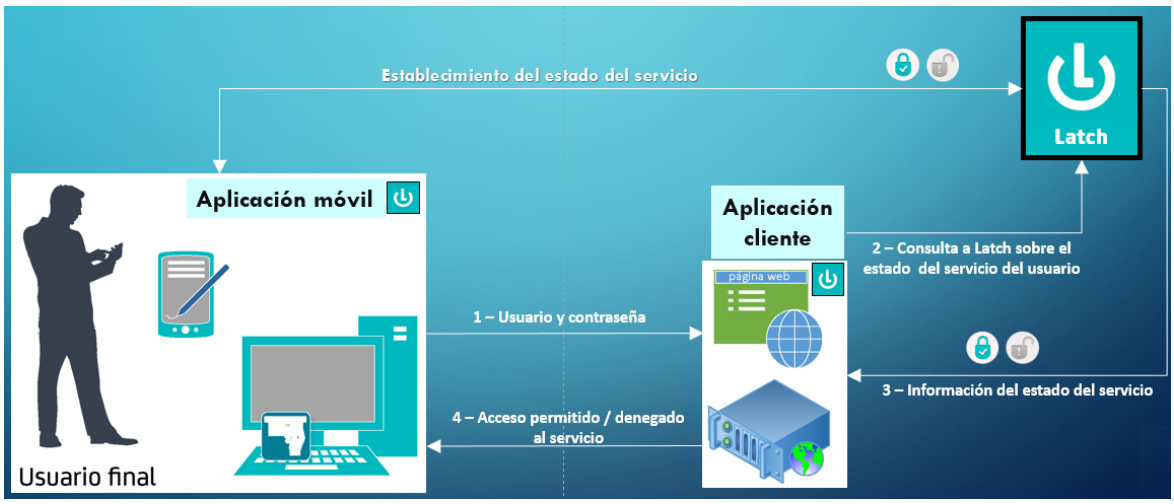

Figura 5.12: Esquema de funcionamiento de Latch

Por lo tanto en este proceso intervienen los siguientes actores:

- **Aplicación en el dispositivo móvil del usuario:**

  La aplicación permite al usuario bloquear, desbloquear y monitorizar el estado de su cuenta.

- **Aplicación cliente de Latch(*plugin*):**

  La aplicación cliente se instala en el sistema a proteger y acepta o deniega los accesos en función de la respuesta del servidor de Latch.

- **Servidor Latch:**

  Es el encargado de almacenar el estado del servicio y responder a las solicitudes de la aplicación cliente.

El tipo de dispositivos o aplicaciones que pueden utilizar Latch está limitado a los *plugins* desarrollados. Estos son de descarga gratuita y están accesibles en la web de Latch[18], a excepción de la versión para Windows-*Enterprise Edition* que requiere una suscripción de pago. Actualmente existen cerca de 40 entre *plugin* y SDK[19].

5.4.2. Integración en el sistema

La integración de la herramienta Latch requiere configurar distintos parámetros e instalar tanto una aplicación móvil como un *plugin* en la máquina virtual. Todo ello se explica en los siguientes apartados.

Apartado de la web

Para la integración de Latch en el sistema es necesario disponer previamente de una cuenta en la plataforma, ya que es desde donde se va a iniciar la configuración de la cuenta. Para ello y en caso de no disponer de ella hemos de crear una cuenta de desarrollador. Desde la página principal seleccionamos el área para desarrolladores y

[18] *Plugins*: `https://latch.elevenpaths.com/www/plugins_sdks.html`

[19] SDK (*Software Development Kit*, Kit de desarrollo de *software*). Conjunto de herramientas de desarrollo de *software* que permiten crear aplicaciones para un sistema concreto.

clicamos sobre el botón "Registrarse como desarrollador". A continuación se muestra el formulario de registro:

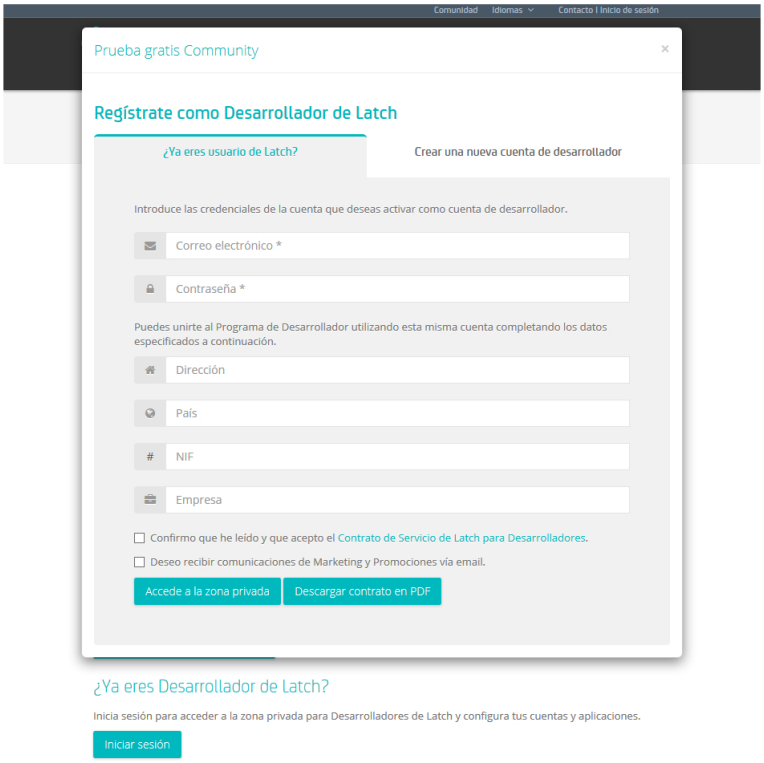

Figura 5.13: Proceso de registro de una cuenta de desarrollador en Latch.

Una vez introducidos los datos y aceptadas las condiciones del servicio carga una nueva página en la que se indica el envío de un correo de confirmación en el que hay un código de validación que deberemos introducir:

Figura 5.14: Mensaje de activación de la cuenta de desarrollador.

El correo muestra el *token* de activación. Bastará con copiarlo y pegarlo en el campo "*Activation code*" de la página web o clicar directamente en el enlace "*Clic aquí para activar tu cuenta*".

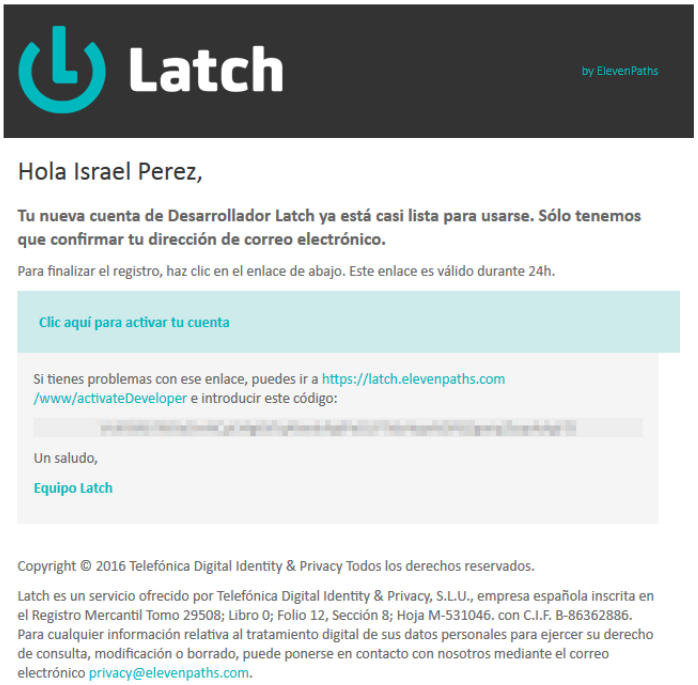

Figura 5.15: Correo de activación de la cuenta de desarrollador.

Una vez activada se muestra un mensaje en el que se indica que la activación se ha realizado satisfactoriamente y ya se puede acceder al portal web.

Para crear una nueva aplicación que genere los *tokens* es necesario seleccionar el apartado "*Mis aplicaciones*" en la barra lateral izquierda del panel de control.

Figura 5.16: Menú mis aplicaciones en el panel de desarrollador de Latch.

A continuación pulsamos en el botón "*Añadir una nueva aplicación*" y se solicita el nombre de la aplicación a mostrar en el panel.

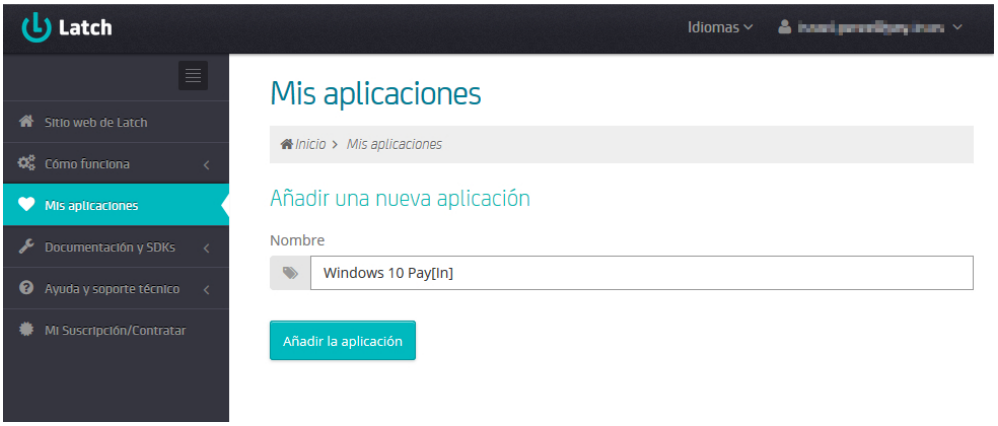

Figura 5.17: Creación de aplicación en panel de desarrollador de Latch (1).

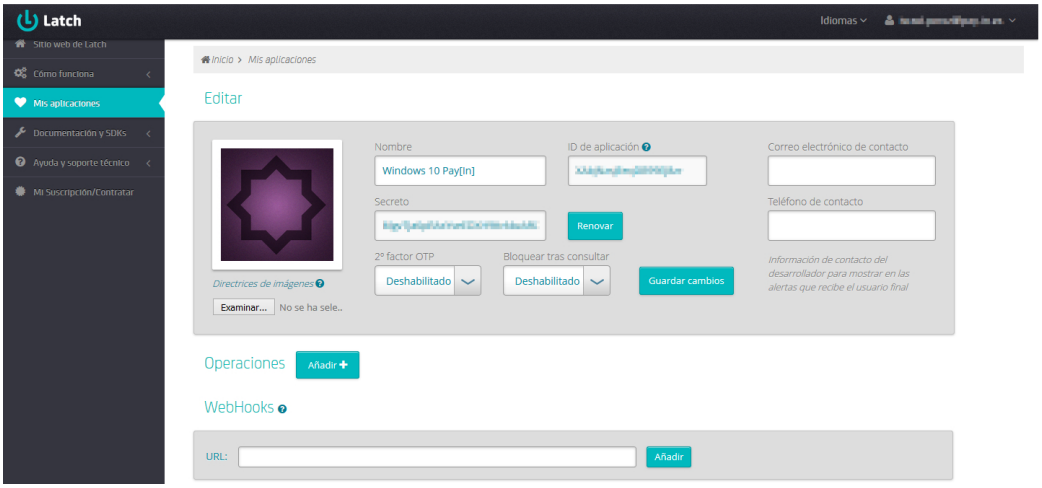

Figura 5.18: Creación de aplicación en panel de desarrollador de Latch (2).

Una vez generada la aplicación se muestra la interfaz de edición en la que se puede observar el identificador de aplicación y el secreto. Estos dos valores son el identificador y la firma que nos van a permitir configurar la llave de acceso en la aplicación cliente. Además hay otros parámetros que pueden ser configurados, como la activación de un segundo factor en el control de acceso o el bloqueo de la cuenta tras consulta. Existen más opciones de configuración pero no son necesarias para llevar a cabo esta implementación.

Apartado del *plugin*

Una vez creada y configurada la aplicación en el portal de Latch se procede con la configuración del equipo. Para ello es necesario descargar el *plugin* en la máquina virtual y ejecutar su instalación.

El *plugin* se encuentra en el menú *"Plugins y SDKs"*, en el desplegable de la barra lateral izquierda *"Documentación y SDKs"*. Una vez en la página buscamos el fichero de nombre *Windows Personal Edition*.

Figura 5.19: Descarga del *plugin* estámdar para Windows.

Aceptamos los términos de uso y descargamos el fichero. Para realizar la instalación simplemente extraemos el fichero comprimido y lanzamos el ejecutable con privilegios de administrador, siguiendo las instrucciones del menú de instalación. Una vez finalizado el proceso de instalación buscamos la aplicación *"Latch para Windows"* en el buscador de aplicaciones y la ejecutamos.

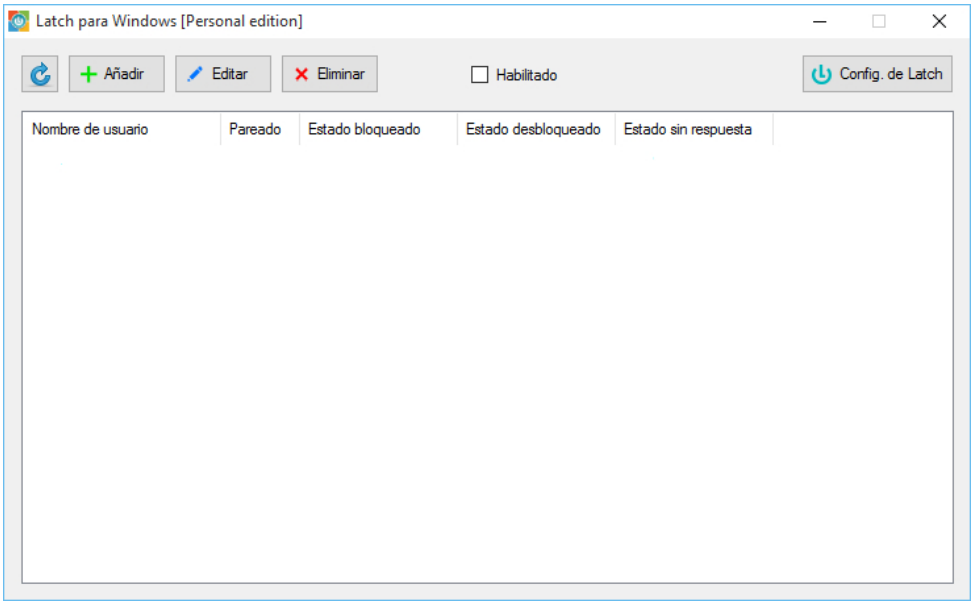

Figura 5.20: Aplicación cliente.

Para proceder con la configuración de la aplicación cliente, en primer lugar hay que habilitar la pestaña "*Habilitado*" para que el servicio funcione. A continuación seleccionamos el botón "*Config. de Latch*" y nos muestra una nueva ventana en la que debemos introducir el identificador de la aplicación y el *token* generados en el panel de control de la *web* de Latch.

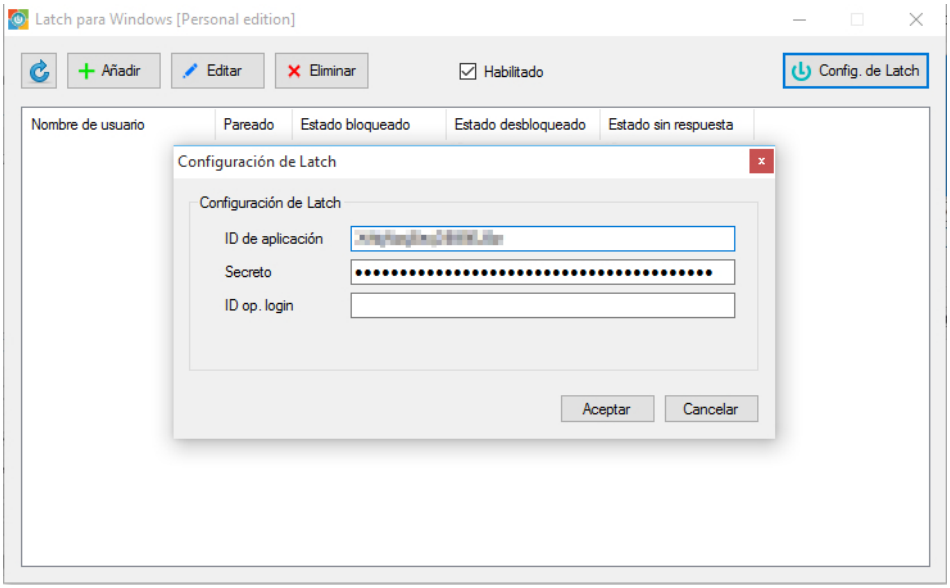

Figura 5.21: Configuración de la aplicación cliente.

Un vez realizado este procedimiento nuestra máquina virtual estará sincronizada con el servidor Latch. Para finalizar será necesario "parear"[20] la aplicación móvil con la aplicación cliente.

[20]Término utilizado en la herramienta Latch indicando la sincronización entre los dispositivos.

Apartado móvil

Una vez iniciada la aplicación introducimos los datos de nuestra cuenta y accedemos a la ventana principal. En la parte inferior de esta última pulsamos en el botón "*Añadir nuevo servicio"* y esto nos redirige a la interfaz de pareado del móvil.

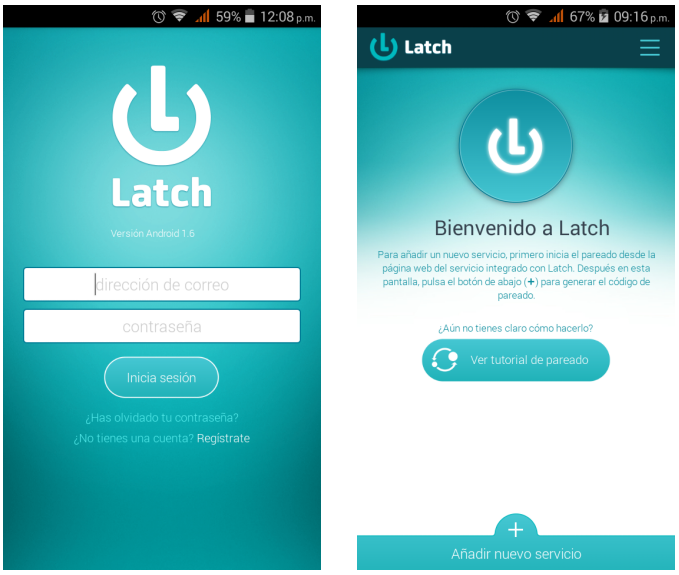

Figura 5.22: Interfaces de *login* y principal de la aplicación móvil.

A continuación se solicita generar un nuevo código y tras pulsar en el botón de generación la siguiente interfaz muestra el código a introducir en la aplicación cliente.

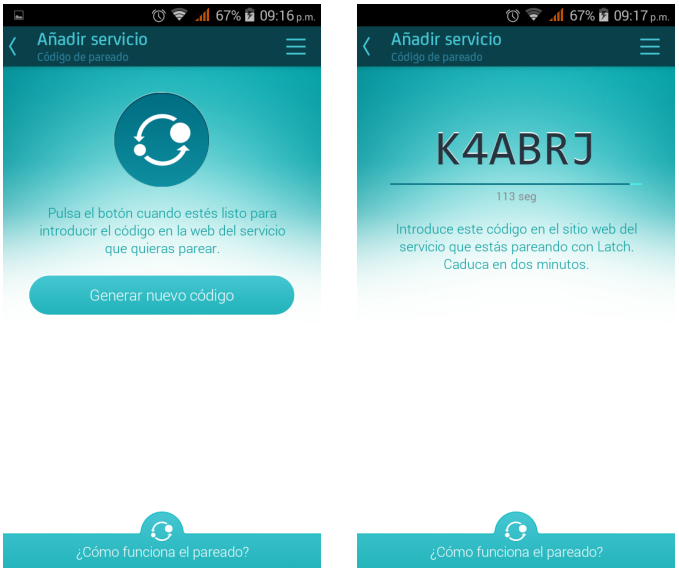

Figura 5.23: Interfaz del código de pareado de la aplicación móvil.

Para concluir el pareado introducimos el código en la aplicación cliente y pulsamos el botón "*Parear*". Si el proceso ha sido satisfactorio se mostrará el mensaje "*La cuenta ya está pareada*", tal y como se muestra a continuación:

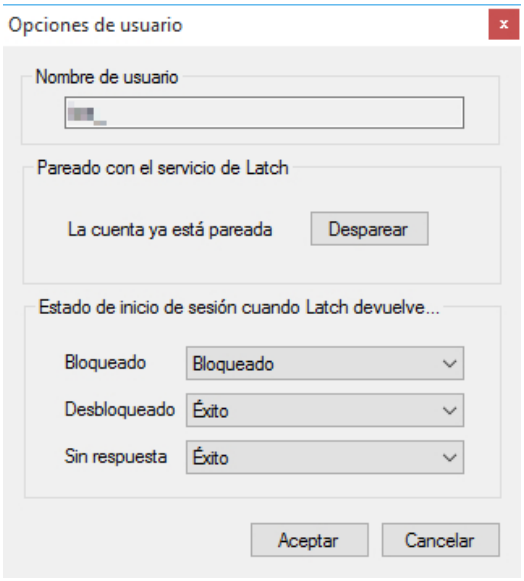

Figura 5.24: Pareado del dispositivo móvil con la aplicación cliente de Latch

Si el proceso se ha realizado correctamente en la aplicación móvil se mostrará la nueva cuenta asociada activada de forma predeterminada.

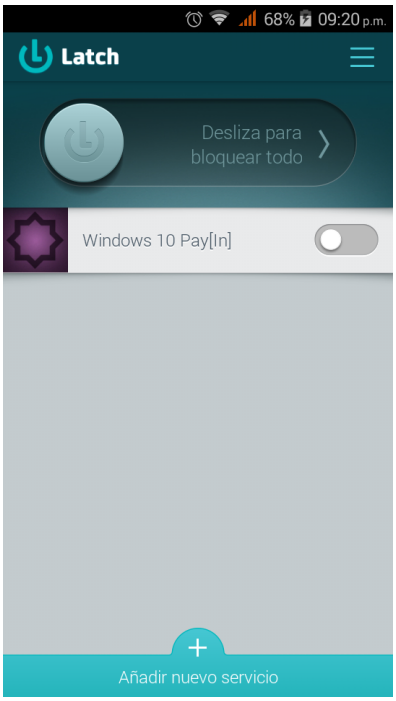

Figura 5.25: Interfaz de cuenta asociada de la aplicación móvil.

Para verificar el correcto funcionamiento del candado virtual realizamos una prueba de acceso habiendo bloqueado la cuenta previamente. Para ello reiniciamos la máquina virtual y bloqueamos la cuenta desde la aplicación móvil deslizando el botón de bloqueo hacia la derecha.

Si ahora intentamos iniciar sesión en la máquina virtual con la cuenta bloqueada, se muestra el siguiente mensaje en la interfaz de inicio de sesión de Windows 10: "*La cuenta a que se hace referencia está bloqueada y no se puede utilizar*". Además se recibe una notificación en la aplicación móvil indicando el intento de conexión.

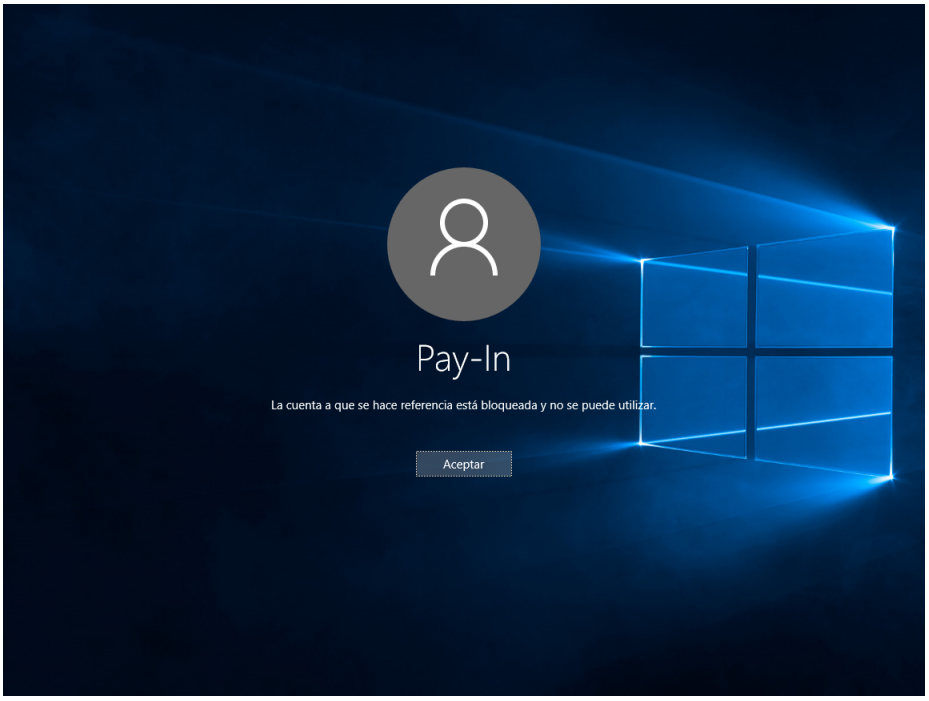

Figura 5.26: Bloqueo de inicio de sesión.

5.4.3. Precauciones y consejos sobre el uso de Latch

Es necesario tener en cuenta una serie de principios a la hora de utilizar y comprender la solución Latch.

- Este sistema no sustituye a los controles de acceso tradicionales, los complementa añadiendo un elemento más a la seguridad del control de acceso.
- Uno de los requisitos indispensables para su correcto funcionamiento es que disponga de conexión directa a Internet. Esto es debido a que debe conectarse a los servidores de Latch para realizar la comprobación del estado de la cuenta. Por lo que en caso de que la cuenta haya sido bloqueada previamente y no se disponga de conexión a Internet, será imposible acceder al sistema.
- En el caso de utilizar Windows 10 como sistema operativo en la máquina a proteger es necesario que la cuenta de usuario sea local, ya que si es de Microsoft siempre se autenticará contra sus servidores y no funcionará correctamente.
- Como todo componente *software* no está exento de fallos o vulnerabilidades, ya que en su implementación se hace uso de librerías externas, como OpenSSL[21] o cURL[22], las cuales pueden verse afectadas por vulnerabilidades en el propio sistema en el que se implanten. Es por ello que una medida para mitigar posibles fallos de seguridad es mantener los sistemas actualizados siempre a su última versión y descargar el *software* de sitios confiables.

[21] Conjunto de herramientas y bibliotecas de funciones criptográficas.

[22] Proyecto de *software* orientado a la transferencia de archivos. Su objetivo es automatizar transferencias de archivos o secuencias de operaciones no supervisadas.

5.4.4. Coste

A pesar de la existencia de un capítulo completo dedicado al coste de la plataforma Microsoft Azure, se ha decidido no incluir este apartado en él puesto que la integración de Latch en la infraestructura de Pat[In] no suponen coste económico alguno. Esto es así debido a que solo se necesita gestionar una única cuenta de usuario y aplicación. Latch es gratuito para la gestión de un máximo de cincuenta cuentas de usuario y dos aplicaciones por cuenta.

Sin embargo, en caso de que la máquina virtual sobre la que se implantase Latch fuese Windows Server, se requeriría el *plugin* premium por lo que sería necesario contratar una cuenta de pago y el coste variaría en función del número de cuentas de usuarios y aplicaciones a gestionar.

CAPÍTULO 6

Coste de los servicios en la nube de Microsoft Azure

En este capítulo se estudia el impacto económico que supone para la empresa la aplicación de las medidas expuestas en el capítulo cuarto. También se analiza la política de precios en la que se basa el modelo de negocio de Azure.

6.1 Política de precios de Microsoft Azure

Microsoft Azure, tiene una política de precios basada en la optimización de los recursos y su aprovechamiento. Sus principales características son:

- No hay costes por adelantado
- No hay tarifas de cancelación
- Pago por uso
- Facturación por minuto

De esta forma Azure se convierte en un producto muy atractivo para posibles consumidores ya que no obliga a pagar por un servicio de golpe. Ofrece la posibilidad de probar la plataforma y en caso de no cumplir con las expectativas pagar por los recursos utilizados solo durante el tiempo que han estado activos. Además existe la opción de crear una cuenta gratuita[1] con un crédito de 170€, sin embargo, tiene limitaciones en los productos que se pueden implementar, como por ejemplo el almacén de claves, por lo que no sería una opción para nuestra implementación.

6.1.1. Opciones de compra

Azure dispone de distintas opciones de compra a la hora de adquirir sus recursos. A continuación se detallan las más importantes:

- **Suscripción de pago por uso:**

  Este modelo consiste en el pago por uso de los recursos, es decir, pagar por los servicios que esté usando el cliente en base a factores como el tiempo de uso o el cómputo. De esta forma permite optimizar los recursos, tanto a nivel de consumo

[1] Enlace a cuenta gratuita de Azure: `https://azure.microsoft.com/es-es/free/`

ya que solo se utilizan cuando son necesarios, como a nivel de usuario ya que no ha de pagar un alquiler por un recurso sea o no aprovechado. En caso de cancelación del servicio se deberá abonar el importe de lo consumido hasta la fecha pero sin costes añadidos.

Esta es la suscripción que tiene contratada Pay[In] en su cuenta de Azure por lo que basaremos el presupuesto en ella.

- **Suscripción de previo pago:**

  Opción de pago por adelantado de 12 meses con el incentivo de un 5 % de descuento sobre el importe total. Sin embargo, hay que remarcar que en caso de cancelación del servicio no se permite el reembolso del importe total, a excepción de si se realiza una cancelación durante los primeros 30 días desde su compra, aceptando el pago de los recursos utilizados. Además, si se termina el plazo de 12 meses y aún se dispone de fondos éstos se perderán.

- **Revendedores de Microsoft:**

  Esta opción permite a los socios autorizados de Microsoft ofrecer sus servicios a terceros a través de la plataform Azure. Se factura a los asociados mensualmente y estos a su vez facturan a sus clientes.

- **Contrato *Enterprise*:**

  Este modelo está orientado a empresas de gran envergadura que disponen ya de un contrato de este tipo. Está basado en el pago por anticipado ofreciendo opciones flexibles de facturación y descuentos en el total de la factura. Se permite el pago anual de las facturas siempre que no se exceda de ciertos límites de uso.

- **Opción de proceso de Microsoft Azure:**

  Este tipo de suscripción está orientado a la migración de cargas de trabajo desde los recursos locales de una entidad a la nube de Microsoft a un ritmo moderado y no establecido. La ventaja principal es el descuento de hasta un 60 % sobre el precio de venta.

6.2 Coste estimado de la estructura desplegada

Microsoft Azure dispone de un simulador de presupuestos[2] en el que se pueden ir añadiendo los recursos requeridos, así como el tipo de cuenta asociada y otros parámetros relativos a cada uno de los distintos servicios ofrecidos. A través de esta herramienta es fácil obtener un presupuesto muy aproximado al gasto real de la implementación una vez desplegada. A continuación se muestra una captura de la página de la herramienta:

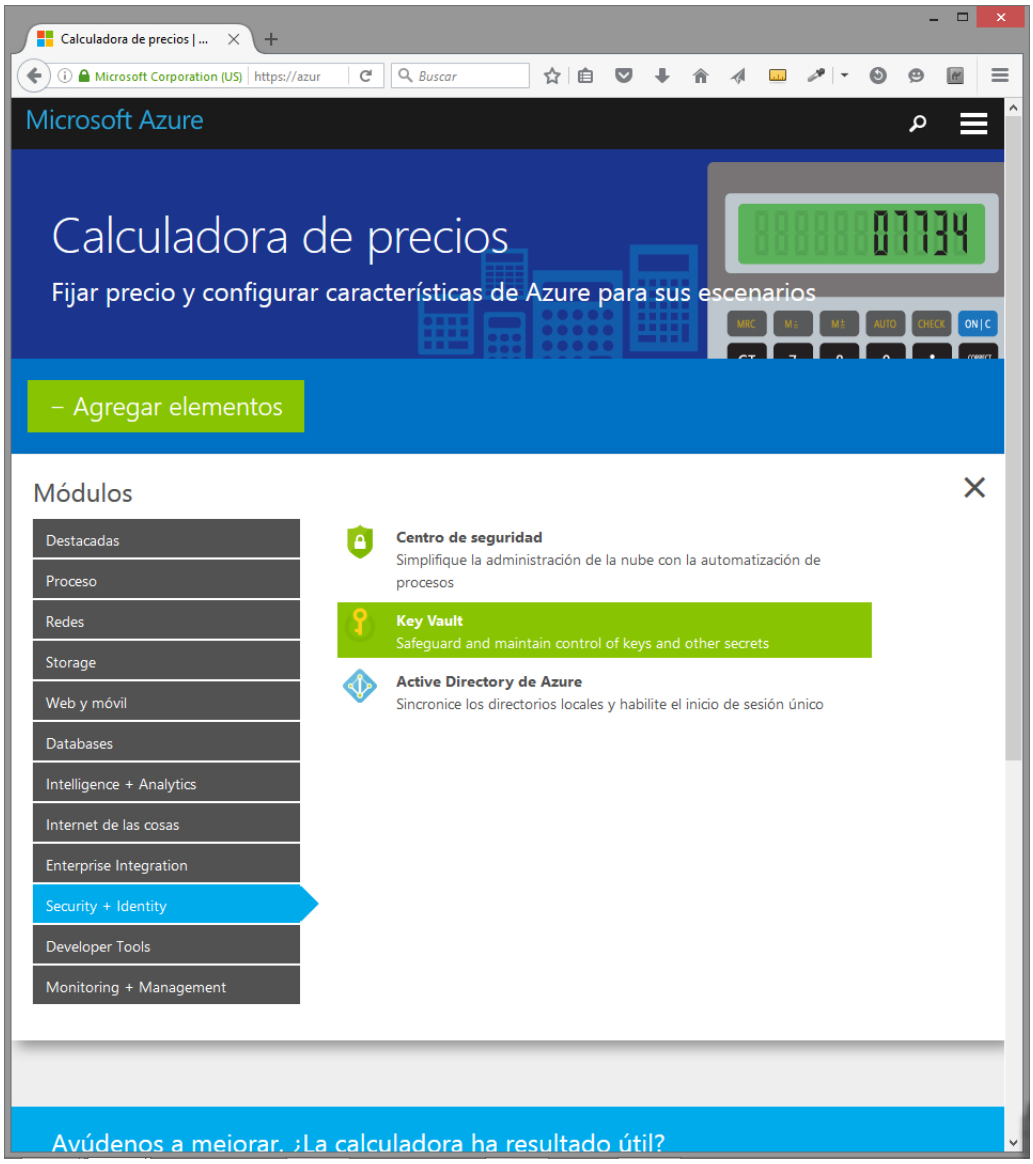

Figura 6.1: Captura de la calculadora de precios de Microsoft Azure

Para obtener el presupuesto se han de ir seleccionando y configurando los parámetros de los recursos que se deseen contratar. Tal y como se puede observar en la captura el almacén de claves (*Key Vault*) ya está preseleccionado. Tras clicar sobre él se añade al panel principal y se pueden modificar los parámetros de su configuración. En el caso del almacén de claves se debe introducir la región en la que se desplegará, el número de operaciones estándar y el número de claves protegidas por un HSM.

[2]Calculadora de precios de Microsoft Azure: `https://azure.microsoft.com/es-es/pricing/calculator/`

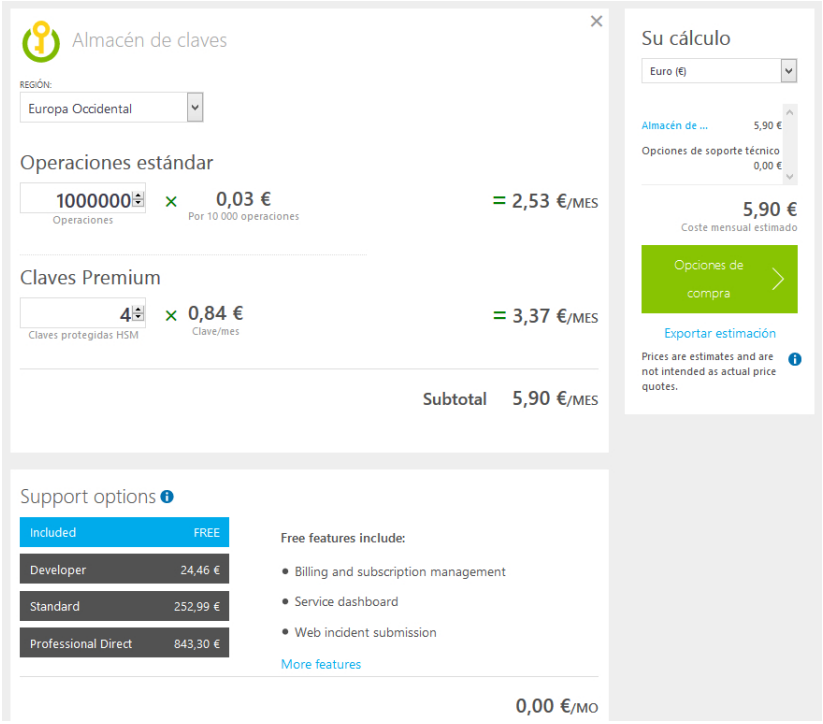

Figura 6.2: Captura de la configuración del coste del almacén de claves de Azure

El valor del número de operaciones se obtiene mediante un sencillo cálculo en función de la previsión del número de usuarios y accesos al almacén de claves por operación. La estimación es la siguiente:

En base al número de usuarios actuales de la aplicación hemos determinado que un valor aproximado pueda ser próximo a 8000. Este valor ha de multiplicarse por el número de operaciones de acceso a las claves del almacén, que es directamente proporcional a las operaciones de pago que se realicen además del cifrado y descifrado de datos. La estimación de operaciones mensuales por usuario es de 60 compras. Por lo tanto la predicción de operaciones en base a un total de 120 operaciones por compras al día asciende a 960000 operaciones, pero para tener cierto margen en caso de aumentar rápidamente el número de usuarios u operaciones esta cifra se redondea a 1000000 operaciones:

$$2(60\ pagos)\ x\ 8000\ usuarios\ = 960000\ op \approx 1000000\ op$$

Además, en la captura también se puede observar la opción de añadir soporte técnico, el cuál no lo consideramos necesario para nuestro desarrollo.

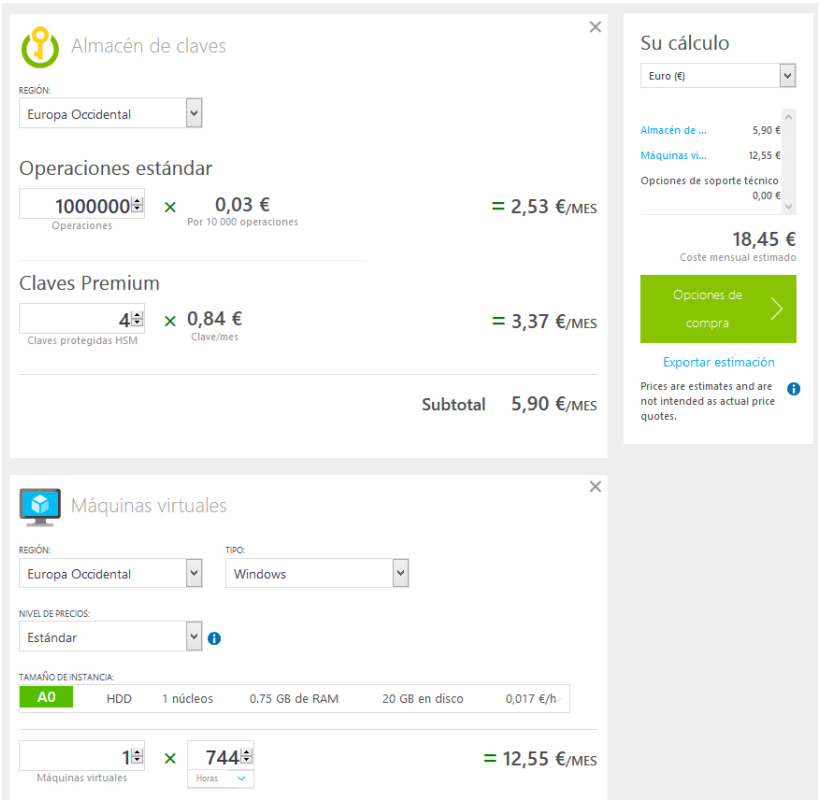

Figura 6.3: Captura de la configuración del coste del almacén de claves de Azure y la máquina virtual

Al precio del almacén de claves se le suma el de la máquina virtual de Windows 10 que se utilizará para la integración de Latch en el desarrollo.

En el caso de la máquina virtual es necesario indicar, además de la región, el tipo de máquina virtual, el tipo de cuenta y el tamaño de la instancia. Para nuestra configuración seleccionamos el tipo Windows, una cuenta estándar y la instancia más económica ya que no se requiere potencia de cálculo, memoria RAM, ni almacenamiento en disco más que el suficiente para hacer funcionar la máquina virtual.

Por lo tanto el coste de esta implementación quedará como se muestra en la siguiente tabla:

Coste		
	Mensual	Anual
Operaciones estándar	2.53 €	30.36 €
2 claves operaciones	1.68 €	20.16 €
2 claves comunicaciones	1.68 €	20.16 €
Máquina virtual	12.55 €	150.6 €
Total:	**18.44 €**	**221.28 €**

Tabla 6.1: Coste de mantenimiento del proyecto mensual y anual

Es necesario puntualizar que este coste solo hace referencia al desarrollo implementado en este TFG, y no a toda la implementación de Pay[In] ya que actualmente se dispone de recursos desplegados en Azure. Sin embargo, esos costes no se incluyen en este proyecto para preservar datos privados de la empresa.

CAPÍTULO 7
Conclusión

7.1 Dificultades encontradas

Durante el desarrollo del proyecto han habido una serie de problemas que se han ido solucionando a medida que se han ido encontrando. A continuación se explican las medidas adoptadas para cada uno de ellos.

- **Guia PCI del cliente de Windows Azure obsoleta**

  El primer hándicap ha sido no disponer de una guía PCI del cliente de Windows Azure actualizada a la versión de la norma en proceso de implementación, por lo que para actualizar los requisitos ha sido necesario analizarlos uno por uno todos ellos en su versión 3.0 respecto a la versión anterior ya que en algunos casos se habían añadido, modificado o eliminado requisitos.

- **Reciente puesta en servicio y modificaciones en la implementación del almacén de claves de Microsoft Azure**

  Otra dificultad encontrada ha sido que el servicio de almacén de claves de Azure ha sido publicado hace relativamente poco tiempo por lo que la documentación existente no es muy extensa. Esto sumado a que constantemente están haciendo mejoras y cambios implica adaptar partes del desarrollo. Un claro ejemplo de esto último se puede observar en los mensajes de advertencia que se muestran en las figuras 5.7 y 5.8. Los mensajes advierten de próximas modificaciones en los parámetros de los *cmdlet*, por lo que será necesario revisar de nuevo los *scripts* para actualizarlos y mantenerlos funcionales.

7.2 Ampliaciones y mejoras

Partiendo de que la norma PCI DSS contiene una gran cantidad de requisitos a implementar y solamente se han abarcado tres de ellos en este TFG, una primera ampliación del proyecto es el análisis técnico e implementación de todos y cada uno de los requisitos restantes, así como la actualización y adaptación a las nuevas versiones que se publiquen.

Por otra parte, se puede mejorar la programación de los *scripts* con el fin de aumentar su automatización. Además de la actualización de los *cmdlets* a nuevas versiones.

También existe la opción de integrar Latch en el proyecto .Net y ofrecer el servicio de candado a los propios usuarios de nuestra plataforma. Esto añadiría una capa de seguridad adicional a los usuairos permitiendo bloquear su cuenta y evitando

que pudiesen realizarse pagos estando desactivada. Sin embargo, aunque es una idea interesante, por el momento no es factible de ser implementada ya que al aumentar el número de cuentas en Latch sería necesario contratar otro tipo de suscripción y encareciendo notablemente el coste de la implementación y mantenimiento.

Por último, otra posibilidad de ampliación consiste en adaptar toda la implementación de la infraestructura en la nube de Microsoft a la tecnología del despliegue mediante plantillas del administrador de recursos de Azure[1]. Esta tecnología permite configurar y desplegar los recursos mediante una plantilla en formato JSON. De esta forma se permite desplegar tantos recursos como sean necesarios manteniendo la misma configuración, así como disponer de las distintas versiones de la infraestructura desplegada facilitando su modificación, permitiendo siempre volver a un estado anterior de la configuración. Además, de esta forma se automatizan muchos procesos evitando en mayor medida la intervención humana, disminuyendo por tanto el error humano.

Actualmente la infraestructura en Pay[In] se gestiona a través del portal de Microsoft Azure o mediante *scripts* en PowerShell y la adopción de esta tecnología permitiría gestionar de una forma sencilla los cambios y actualizaciones de versiones de los controladores y el *sofware* ejecutado. Además, este método de despliegue permite mantener un histórico de configuraciones ya que al ser parte del proyecto puede ser gestionado a través de un control de versiones[2], añadiéndose como un fichero más al proyecto y quedando registrados todos los cambios.

7.3 Conclusiones finales

La elaboración de este TFG ha requerido una importante aportación en materia de investigación, ya que se han empleado tecnologías novedosas y poco conocidas que han requerido de un proceso previo de estudio.

En un principio se ha analizado la situación actual respecto al fraude en la banca y lo vulnerables que son todos los sistemas informáticos, en función de lo segurizados que estén. E incluso aún así nunca se puede afirmar que un sistema es 100 % seguro ya que la variabilidad de situaciones posibles y fallos aún no encontrados es incalculable.

Se ha investigado y trabajado sobre la plataforma Azure, esto ha permitido ahondar en sus características y descubrir el infinito mar de posibilidades que ofrece. También se ha analizado la norma PCI DSS de seguridad en el almacenamiento de datos bancarios, permitiendo conocer de primera mano las estándares de seguridad más actuales así como las tecnologías empleadas y los requisitos que han de cumplirse. Ello ha condicionado el uso de prácticas relacionadas con la seguridad informática. Por último, se ha trabajado en la integración de la herramienta Latch para aumentar la seguridad en el control de acceso a las cuentas de administrador de la empresa.

El mundo está en constante evolución, las tecnologías cambian y siempre hay que ser capaz de comprenderlas y saber hacer un buen uso de ellas para poder adaptarlas a las necesidades propias. Por todo ello se puede considerar que el desarrollo de este TFG ha sido completado con éxito, habiéndose alcanzado todos los puntos del mismo y siendo conscientes de que es solo una pequeña parte de la implementación completa de la norma PCI DSS.

[1] Más información sobre el administrador de recursos de Azure en: `https://azure.microsoft.com/es-es/documentation/articles/resource-group-overview/`

[2] El control de versiones es un sistema que registra los cambios realizados sobre uno o varios archivos a lo largo del tiempo.

Bibliografía

[1] "Pay[in], la cartera en tu móvil." http://www.pay-in.es/. Visited on 2016-05-30.

[2] "Latch. El interruptor de seguridad para tu vida digital." https://latch.elevenpaths.com/. Visited on 2016-05-30.

[3] PCI SSC, "Normas de seguridad de datos. requisitos y procedimientos de evaluación de seguridad, versión 3.0." https://es.pcisecuritystandards.org/_onelink_/pcisecurity/en2es/minisite/en/docs/PCI_DSS_v3.pdf, abril 2013. Visited on 2016-05-19.

[4] PCI SSC, "Requirements and Security Assessment Procedures." https://www.pcisecuritystandards.org/documents/PCI_DSS_v3-1.pdf, April 2015. Visited on 2016-06-16.

[5] PCI SSC, "Requirements and security assessment procedures, version 3.2." https://www.pcisecuritystandards.org/documents/PCI_DSS_v3-2.pdf, abril 2016. Visited on 2016-05-21.

[6] Centro Criptológico Nacional(CCN), "CCN-CERT IA-09/16 Ciberamenazas 2015/Tendencias 2016, resumen ejecutivo." https://www.ccn-cert.cni.es/informes/informes-ccn-cert-publicos/1483-ccn-cert-ia-0916-ciberamenazas-2015-tendencias-2016-resumen-ejecutivo/file.html, abril 2016. Visited on 2016-06-3.

[7] adigital, "Informe de medios de pago y fraude online en España 2016." https://info.bbva.com/es/data/8663042016/Adigital-Informe-de-Medios-de-Pago-y-Fraude-Online-en-Espan%CC%83a-2016.pdf, abril 2016. Visited on 2016-05-26.

[8] Departamento de Investigación del IEB, "La transformación digital de la banca española." http://www.ieb.es/wp-content/uploads/2015/11/estudio_banca_2015.pdf, abril 2015. Visited on 2016-05-24.

[9] D. Acosta, "Todo lo que siempre has querido saber acerca de los saq (cuestionarios de auto-evaluación) de pci dss v3.2." http://www.pcihispano.com/todo-lo-que-siempre-ha-querido-saber, may 2016. Visited on 2016-06-11.

[10] D. Acosta, "Controles compensatorios: ¿qué son y cuándo se utilizan?." http://www.pcihispano.com/controles-compensatorios-que-son-y-cuando-se-utilizan/, may 2013. Visited on 2016-06-11.

[11] Microsoft, "What is cloud computing? A beginner's guide." https://azure.microsoft.com/es-es/overview/what-is-cloud-computing/. Visited on 2016-07-8.

[12] O. Mejıa, "Computación en la nube," *ContactoS*, vol. 80, pp. 45–52, 2011. Visited on 2016-07-19.

[13] C. de Wikipedia, "Computación en la nube." https://es.wikipedia.org/wiki/Computaci%C3%B3n_en_la_nube, may 2016. Visited on 2016-06-18.

[14] P. Mell and T. Grance, "The nist definition of cloud computing," 2011. Visited on 2016-07-07.

[15] L. J. Aguilar, "Computación en la nube e innovaciones tecnológicas," *El nuevo paradigma de la Sociedad del Conocimiento*, 2011. visited on 2016-07-08.

[16] R. García Santos *et al.*, "Estudio de la tecnología windows azure: aprovisionamiento y escalado automático de aplicaciones multi-tenants," 2013. Visited on 2016-06-25.

[17] C. de Wikipedia, "Microsoft Azure." https://es.wikipedia.org/w/index.php?title=Microsoft_Azure&oldid=91965012, june 2016. Visited on 2016-06-1.

[18] M. A. Murazzo, F. Millán, N. Rodríguez, D. Segura, and D. A. Villafañe, "Desarrollo de aplicaciones para cloud computing," in *XVI Congreso Argentino de Ciencias de la Computación*, 2010. Visited on 2016-06-4.

[19] "2015 Review Shows $110 billion Cloud Market Growing at 28 % Annually." https://www.srgresearch.com/articles/2015-review-shows-110-billion-cloud-market-growing-28-annually, january 2016. Visited on 2016-06-2.

[20] Rob Boucher Jr, "Iintroducción a Microsoft Azure." https://azure.microsoft.com/es-es/documentation/articles/fundamentals-introduction-to-azure/#los-componentes-de-azure, June 2016. Visited on 2016-07-1.

[21] I. Neohapsis, "Windows Azure™ Customer PCI Guide." http://eqinc.com/images/white-papers/azure/windows-azure-pci-guide-january-2014.pdf, January 2014. Visited on 2016-08-3.

[22] PCI SSC, "Resumen de los cambios de la versión 3.1 a la 3.2 de las PCI DSS." https://es.pcisecuritystandards.org/_onelink_/pcisecurity/en2es/minisite/en/docs/PCI_DSS_v3-1_Summary_of_Changes_6Apr2015_es-LA.pdf, April 2015. Visited on 2016-06-16.

[23] C. Bailey, A. Glick, and B. Perler, "Documentación de Almacén de claves." https://azure.microsoft.com/es-es/documentation/services/key-vault/. Visited on 2016-06-4.

[24] C. Bailey, "Introducción al Almacén de claves de Azure." https://azure.microsoft.com/es-es/documentation/articles/key-vault-get-started/, may 2016. Visited on 2016-06-26.

[25] "Scripting con Windows PowerShell." https://technet.microsoft.com/es-es/library/bb978526.aspx, august 2014. Visited on 2016-06-26.

[26] C. Plett, "Cómo instalar y configurar Azure PowerShell." https://azure.microsoft.com/es-es/documentation/articles/powershell-install-configure/, april 2016. Visited on 2016-06-29.

[27] "Cmdlet Overview." https://msdn.microsoft.com/es-es/library/ms714395(v=vs.85).aspx, june 2016. Visited on 2016-06-27.

[28] "Azure Key Vault client samples." https://www.microsoft.com/en-us/download/details.aspx?id=45343, june 2016. Visited on 2016-06-27.

[29] C. Nottingham, "Inicio de sesión en una máquina virtual Windows mediante el Portal de Azure clásico." https://azure.microsoft.com/es-es/documentation/articles/virtual-machines-windows-classic-connect-logon/, may 2016. Visited on 2016-06-29.

[30] Chema Alonso, "Instalar, Configurar y Usar Latch para Windows." http://www.elladodelmal.com/2014/07/instalar-y-configurar-latch-para-windows.html, July 2014. Visited on 2016-08-29.

[31] Microsoft, "Precios de Azure." https://azure.microsoft.com/es-es/pricing/. Visited on 2016-08-12.

[32] PCI SSC, "Resumen de los cambios de la versión 2.0 a la 3.0 de las PCI DSS." https://es.pcisecuritystandards.org/_onelink_/pcisecurity/en2es/minisite/en/docs/PCI_DSS_v3_Summary_of_Changes.pdf, November 2013. Visited on 2016-06-12.

APÉNDICE A

Scripts para el despliegue y configuración del almacén de claves de azure

A.1 *Script* en PowerShell para la creación del almacén de claves

```
# ------------------------------------------------------------------
# This sample PowerShell gets the settings you'll need for the *.cscfg
# files. Modified 21/3/16 by israel.perez@pay-in.es
# ------------------------------------------------------------------

# ------------------------------------------------------------------
# You MUST set the following values before running this script
# ------------------------------------------------------------------
$vaultName            = 'MyVaultName'
$resourceGroupName    = 'MyResourceGroupName'
$applicationName      = 'MyAppName'
$keyName              = ''

# ------------------------------------------------------------------
# You MAY set the following values before running this script
# ------------------------------------------------------------------
$location             = ''# Get-AzureLocation

# ------------------------------------------------------------------
# Should we bounce this script execution?
# ------------------------------------------------------------------
if (($vaultName -eq 'MyVaultName') -or `
    ($resourceGroupName -eq 'MyResourceGroupName') -or `
   ($applicationName -eq 'MyAppName') -or `
   ($keyName -eq ''))
{
  Write-Host 'You must edit the values at the top of this script before
      executing'
          -foregroundcolor Yellow
  exit
}
# ------------------------------------------------------------------
# Add Azure Account
# ------------------------------------------------------------------
$AzureSubscription = Get-AzureSubscription
if(-not $AzureSubscription)
{
```

```
37      Write-Host 'Please add an Azure Subscription'
38      Add-AzureAccount
39  }
40  # ------------------------------------------------------------------
41  # Log into Azure
42  # ------------------------------------------------------------------
43  Write-Host 'Please log into Azure Resource Manager now' -foregroundcolor Green
44  Login-AzureRmAccount
45  $VerbosePreference = "SilentlyContinue"
46
47  # ------------------------------------------------------------------
48  # Create the resource group and vault if needed
49  # ------------------------------------------------------------------
50  $rgExists = Get-AzureRmResourceGroup -Name $resourceGroupName -ErrorAction
        SilentlyContinue
51  if (-not $rgExists)
52  {
53      New-AzureRmResourceGroup -Name $resourceGroupName -Location $location
54  }
55  else
56  {
57      Write-Host "Resource group $resourceGroupName exists!"
58  }
59
60  $vaultExists = Get-AzureRmKeyVault -VaultName $vaultName -ErrorAction
        SilentlyContinue
61  if (-not $vaultExists)
62  {
63      Write-Host "Creating vault $vaultName"
64      $vault = New-AzureRmKeyVault -VaultName $vaultName `
65            -ResourceGroupName $resourceGroupName `
66             -Sku premium `
67             -Location $location
68  }
69  else
70  {
71     Write-Host "The Key Vault $vaultName exists!"
72  }
73  # ------------------------------------------------------------------
74  # Keys creation
75  # ------------------------------------------------------------------
76  $keyExists = Get-AzureKeyVaultKey -Name $keyName -VaultNAme $vaultName
77  if(-not $keyExists) {
78      Write-Host "Setting key $keyName in vault $vaultName using HSM destination"
79      $key = Add-AzureKeyVaultKey `
80            -VaultName $vaultName `
81            -Name $keyName `
82            -Destination HSM
83  }
84  else
85  {
86     Write-Host "The Key $keyName in vault $vaultName exists!"
87  }
88  # ------------------------------------------------------------------
89  # Print the XML settings that should be copied into the CSCFG files
90  # ------------------------------------------------------------------
91  Write-Host "Place the following into both CSCFG files for the project:"
92  -ForegroundColor Cyan
93  '<Appname="KeyUrl" value="' + $key.Id.Substring(0, $key.Id.LastIndexOf('/')) +
94  '" />'
95  Write-Host
96
97  # ------------------------------------------------------------------
98  # Print out in a file
```

```
# ------------------------------------------------------------------
$outputFile =
 '------------------------------------------------------------------' +
 "`r`n" +
 'Time = ' + [System.DateTime]::Now + "`r`n" +
 "Place the following into both CSCFG files for the SampleAzureWebService
 project:" + "`r`n" +
 '<App name="KeyUrl" value="' + $key.Id.Substring(0, $key.Id.LastIndexOf('/'))
     +
 '" />' + "`r`n" +
 '------------------------------------------------------------------'
Out-File -FilePath C:\ConfigurationKeyVaultProject.txt -width 120 -Append
-InputObject $outputFile
```

A.2 *Script* en PowerShell para la creación de la aplicación en el directorio activo de Azure

```
# ------------------------------------------------------------------
# This sample PowerShell gets the settings you'll need for the app.config file
# Modified 21/3/16 by israel.perez@pay-in.es
# ------------------------------------------------------------------

# ------------------------------------------------------------------
# You MUST set the following values before running this script
# ------------------------------------------------------------------
$vaultName            = 'MyVaultName'
$resourceGroupName    = 'MyResourceGroupName'
$applicationName      = 'MyAppName'
$administratorID      = '' #Paste ObjectId here

# ------------------------------------------------------------------
# You MAY set the following values before running this script
# ------------------------------------------------------------------
$applicationPassword = '' # If not specified, script will generate a random
                          password during app creation
$location             = '' # Get-AzureLocation

# ------------------------------------------------------------------
# Generates a secure 32-byte symmetric key for authentication
# ------------------------------------------------------------------
Function GenerateSymmetricKey()
{
    $key = New-Object byte[](32)
    $rng = [System.Security.Cryptography.RNGCryptoServiceProvider]::Create()
    $rng.GetBytes($key)
    return [System.Convert]::ToBase64String($key)
}
# ------------------------------------------------------------------
# Should we bounce this script execution?
# ------------------------------------------------------------------
if (($vaultName -eq 'MyVaultName') -or ($resourceGroupName -eq '
    MyResourceGroupName') -or ($applicationName -eq 'MyAppName'))
{
  Write-Host 'You must edit the values at the top of this script before
      executing' -foregroundcolor Yellow
  exit
}
# ------------------------------------------------------------------
# Add Azure Account
# ------------------------------------------------------------------
$AzureSubscription = Get-AzureSubscription
```

```
43 if(-not $AzureSubscription)
44 {
45     Write-Host 'Please add an Azure Subscription'
46     Add-AzureAccount
47 }
48 # ********************************************************************
49 # Log into Azure
50 # ********************************************************************
51 Write-Host 'Please log into Azure Resource Manager now' -foregroundcolor Green
52 Login-AzureRmAccount
53 $VerbosePreference = "SilentlyContinue"
54
55 # ********************************************************************
56 # Create application in AAD if needed
57 # ********************************************************************
58 $SvcPrincipals = (Get-AzureRmADServicePrincipal -SearchString $applicationName)
59 if(-not $SvcPrincipals)
60 {
61   if(-not $applicationPassword)
62   {
63     $applicationPassword = GenerateSymmetricKey
64   }
65     # Create a new AD application if not created before without certificate
66     $identifierUri = "https://webdelservicio/Account/Login"
67     $homePage = "https://webdelservicio.es"
68
69     Write-Host "Creating a new AAD Application"
70     $ADApp = New-AzureRmADApplication `
71       -DisplayName $applicationName `
72       -HomePage $homePage `
73       -IdentifierUris $identifierUri `
74       -Password $applicationPassword `
75
76     Write-Host "Creating a new AAD service principal"
77     $servicePrincipal = New-AzureRmADServicePrincipal `
78     -ApplicationId $ADApp.ApplicationId
79 }
80 else
81 {
82     # Assume that the existing app was created earlier with the right X509
          credentials. We don't modify the existing app to add new credentials
          here.
83
84     Write-Host "WARNING: An application with the specified name (
          $applicationName) already exists." `
85     -ForegroundColor Yellow `
86     -BackgroundColor Black `
87
88     Write-Host "Proceeding with script execution assuming that the existing app
           already has the correct password credentials as specified in
          applicationPassword variable in the script." `
89     -ForegroundColor Yellow `
90     -BackgroundColor Black  `
91
92     Write-Host "If you are not sure about the existing app's credentials,
          choose an app name that doesn't already exist and the script with
          create it and set the specified credentials for you." `
93     -ForegroundColor Yellow `
94     -BackgroundColor Black `
95
96     $servicePrincipal = $SvcPrincipals[0]
97     if (-not $applicationPassword)
98     {
```

```
99          $applicationPassword = "PLEASE FILL THIS IN WITH EXISTING APP'S
                PASSWORD"
100     }
101 }
102
103 # Specify full privileges to the vault for the application
104 Write-Host "Setting access policy"
105 Set-AzureRmKeyVaultAccessPolicy -VaultName $vaultName `
106       ObjectId $servicePrincipal.Id `
107       -PermissionsToKeys all `
108       -PermissionsToSecrets all
109
110 # Specify full privileges to the vault for the Azure administrator
111 if($administratorID)
112 {
113     Write-Host "Setting access policy for Azure administrator"
114     Set-AzureRmKeyVaultAccessPolicy -VaultName $vaultName `
115               -ObjectId $administratorID `
116             -PermissionsToKeys all `
117             -PermissionsToSecrets all
118 }
119 else
120 {
121     Write-Host "administratorId not exists, access policy for Azure
            administrator is not set"
122 }
123 # ********************************************************************
124 # Print the XML settings that should be copied into the web.config file
125 # ********************************************************************
126 Write-Host "Paste the following settings into the web.config file for the
        HelloKeyVault project:" -ForegroundColor Cyan
127 '<add key="HSMKeyVaultUrl" value="' + $vault.VaultUri + '"/>'
128 '<add key="HSMClientId" value="' + $servicePrincipal.ApplicationId + '"/>'
129 '<add key="HSMClientSecret" value="' + $applicationPassword + '"/>'
```

A.3 *Scripts* en PowerShell para importar clave pública de cifrado móvil al almacén de claves de Azure

```
1
2 # ***********************
3 # Mobile key import
4 # ***********************
5
6
7 # ********************************************************************************
8 # Add Azure Account
9 # ********************************************************************************
10 $AzureSubscription = Get-AzureSubscription
11 if(-not $AzureSubscription)
12 {
13     Write-Host 'Please add an Azure Subscription'
14     Add-AzureAccount
15 }
16
17 # ********************************************************************************
18 # Log into Azure
19 # ********************************************************************************
20 Write-Host 'Please log into Azure Resource Manager now'
21   -foregroundcolor Green
22
23 Login-AzureRmAccount
```

```
24
25
26 # **********************
27 # Values
28 # **********************
29
30 $vaultName       = 'PayInKeyVault'
31 $mobileKeyName   = ''
32 $mobileKeyFPath  = ''
33 $password        = '' # Password of the import key file
34 $outputFilePath  = ''
35
36 if (($vaultName -eq '') -or `
37     ($mobileKeyName -eq '') -or `
38   ($mobileKeyFPath -eq '') -or `
39   ($password -eq '') -or `
40     ($outputFilePath -eq ''))
41
42 {
43   Write-Host 'You must edit the values at the top of this script before
       executing'
44     -foregroundcolor Yellow
45   exit
46 }
47
48
49 # **********************
50 # Key creation
51 # **********************
52
53 $keyVaultExists = Get-AzureRmKeyVault -VaultName $vaultName
54 Write-Host "Verifying key Vault"
55
56 if($keyVaultExists)
57 {
58     Write-Host "Key Vault $keyVaultExists exists, creating keys"
59       -ForegroundColor Green
60
61     $old_ErrorActionPreference = $ErrorActionPreference
62     $ErrorActionPreference = "SilentlyContinue"
63
64     Write-Host "Verifying if mobile Key exists"
65     $mobileKeyExists = Get-AzureKeyVaultKey -Name $mobileKeyName -VaultName
         $vaultName
66
67     if((!$mobileKeyExists) -and $mobileKeyFPath -and $mobileKeyName)
68     {
69           Write-Host "Mobile Key don't exists, setting key2 $mobileKeyName in
               vault $vaultName using file path" -ForegroundColor Green
70
71           $fileExists = Test-Path $mobileKeyFPath
72           if(!$fileExists)
73           {
74              Write-Host "File isn't in the file path!" -ForegroundColor Yellow
75              exit
76           }
77
78           $passwordSecured = ConvertTo-SecureString
79                     -String $password
80                     -AsPlainText
81                     -Force
82           $mobilekey = Add-AzureKeyVaultKey
83                    -VaultName $vaultName
84                    -Name $mobileKeyName
```

```
85                  -KeyFilePath $mobileKeyFPath
86                  -KeyFilePassword $passwordSecured
87      if($mobilekey)
88      {
89        Write-Host "Import task is done!" -ForegroundColor Green
90            Write-Host $mobilekey.Id.Substring(0, $mobilekey.Id.LastIndexOf('/')
                  )
91              -foregroundcolor Green
92
93            $outputFile = $mobilekey.Id.Substring(0, $mobilekey.Id.LastIndexOf('
                  /')) + "`r`n"
94                Out-File   -FilePath $outputFilePath
95                     -width 120
96                     -Append
97                     -InputObject $outputFile
98      }
99          $ErrorActionPreference = $old_ErrorActionPreference
100     }
101     else
102     {
103         Write-Host "Error verifiyng mobile key!" -ForegroundColor Red
104         exit
105     }
106 }
107 else
108 {
109     Write-Host "Error, Key vault don't exists!" -ForegroundColor Red
110     exit
111 }
```

APÉNDICE B

Requisitos de la norma PCI DSS a implementar por Pay[In]

B.1 Revisión de la norma hasta la versión 3.0

Los requisitos que aquí se indican son los que han de ser gestionados por la empresa, siendo obviados todos aquellos ya cumplidos por Microsoft Azure. Están actualizados hasta la versión 3.0 de la norma y se basan en los documentos[3, 32, 21] así como en el informe confidencial previamente mencionado.

Requisito 1: Instale y mantenga una configuración de *firewalls* para proteger los datos de los titulares de las tarjetas

- **1.1.2** Diagrama de red actual que identifica todas las conexiones entre el entorno de datos de titulares de tarjetas y otras redes, incluso cualquier red inalámbrica.
- **1.1.3** El diagrama actual que muestra todos los flujos de datos de titulares de tarjetas entre los sistemas y las redes.
    - Requisito añadido en la versión 3.0 de la norma.
- **1.1.6** Documentación y justificación de negocio para el uso de todos los servicios, protocolos y puertos permitidos, incluida la documentación de las funciones de seguridad implementadas en aquellos protocolos que se consideran inseguros.
    - Requisito 1.1.5 en la versión 2.0 de la norma.
- **1.2.1** Restrinja el tráfico entrante y saliente a la cantidad necesaria para el entorno de datos de los titulares de tarjetas y niegue específicamente el tráfico restante.
- **1.3.5** No permita que el tráfico saliente no autorizado proveniente del entorno de datos del titular de la tarjeta ingrese en Internet.
- **1.4** Instale software de *firewall* personal en todos los dispositivos móviles o de propiedad de los trabajadores que tengan conexión a Internet cuando están fuera de la red (por ejemplo, computadoras portátiles que usan los trabajadores), y que también se usan para acceder a la red.
- **1.5** Asegúrese de que las políticas de seguridad y los procedimientos operativos para administrar los *firewalls* estén documentados, implementados y que sean de conocimiento para todas las partes afectadas.
    - Requisito añadido en la versión 3.0 de la norma.

Requisito 2: No utilizar contraseñas de sistemas y otros parámetros de seguridad provistos por los proveedores

- **2.2.1** Implemente sólo una función principal por servidor a fin de evitar que coexistan funciones que requieren diferentes niveles de seguridad en el mismo servidor.
- **2.4** Lleve un inventario de los componentes del sistema que están dentro del alcance de las PCI DSS.
    - Requisito añadido en la versión 3.0 de la norma.
- **2.5** Asegúrese de que las políticas de seguridad y los procedimientos operativos para administrar los parámetros predeterminados del proveedor y otros parámetros de seguridad estén documentados, implementados y que sean de conocimiento para todas las partes afectadas.
    - Requisito añadido en la versión 3.0 de la norma.

Requisito 3: Proteja los datos del titular de la tarjeta que fueron almacenados

- **3.1** Almacene la menor cantidad posible de datos del titular de la tarjeta implementando políticas, procedimientos y procesos de retención y eliminación de datos.
    - Combinación de los requisitos 3.1 y 3.1.1 de la versión 2.0.
- **3.2** No almacene datos confidenciales de autenticación después de recibir la autorización (aun cuando estén cifrados). Si se reciben datos de autenticación confidenciales, convierta todos los datos en irrecuperables al finalizar el proceso de autorización.
- **3.2.1** No almacene contenido completo de ninguna pista de la banda magnética (ubicada en el reverso de la tarjeta, datos equivalentes que están en un chip o en cualquier otro dispositivo).
- **3.2.2** No almacene el valor ni el código de validación de tarjetas (número de tres o cuatro dígitos impreso en el anverso o reverso de una tarjeta de pago) que se utiliza para verificar las transacciones de tarjetas ausentes.
- **3.2.3** No almacene el número de identificación personal (PIN) ni el bloqueo del PIN cifrado.
- **3.3** Oculte el PAN (número de cuenta principal) cuando aparezca (los primeros seis y los últimos cuatro dígitos es la cantidad máxima de dígitos que aparecerá), de modo que solo el personal con una necesidad comercial legítima pueda ver el PAN (número de cuenta principal) completo.
- **3.4** Convierta el PAN (número de cuenta principal) en ilegible en cualquier lugar donde se almacene (incluidos los datos que se almacenen en medios digitales portátiles, en medios de copia de seguridad y en registros).
- **3.5** Documente e implemente procedimientos que protejan las claves utilizadas para proteger los datos del titular de la tarjeta almacenados contra su posible divulgación o uso indebido.
- **3.5.1** Restrinja el acceso a las claves criptográficas a la menor cantidad de custodios necesarios.

- **3.5.2** Siempre guarde las claves secretas y privadas utilizadas para cifrar/descifrar los datos del titular de la tarjeta de una o más formas.
- **3.5.3** Guarde las claves criptográficas en la menor cantidad de ubicaciones posibles.
    - Requisito añadido en la versión 3.0, siendo el resultado de la subdivisión del requisito 3.5.2 en la versión 2.0.
- **3.6** Documente por completo e implemente todos los procesos y procedimientos de ad ministración de claves de las claves criptográficas que se utilizan para el cifrado de datos del titular de la tarjeta.
- **3.6.1** Generación de claves de cifrado sólido.
- **3.6.2** Distribución segura de claves de cifrado.
- **3.6.3** Almacenamiento seguro de claves de cifrado.
- **3.6.4** La clave criptográfica cambia en el caso de las claves que han llegado al final de su período de cifrado, según lo defina el proveedor de la aplicación relacionada o el responsable de las claves, y basándose en las mejores prácticas y recomendaciones de la industria.
- **3.6.5** Retiro o reemplazo de claves según se considere necesario cuando se haya debilitado la integridad de la clave o cuando se sospeche que las claves están en riesgo.
- **3.6.6** Si se usan operaciones manuales de administración de claves criptográficas de texto claro, se deben realizar con control doble y conocimiento dividido.
- **3.6.7** Prevención de sustitución no autorizada de claves criptográficas.
- **3.6.8** Requisito para que los custodios de claves criptográficas declaren, formalmente, que comprenden y aceptan su responsabilidad como custodios de claves.
- **3.7** Asegúrese de que las políticas de seguridad y los procedimientos operativos para proteger los datos del titular de la tarjeta almacenados estén documentados, implementados y que sean de conocimiento para todas las partes afectadas.
    - Requisito añadido en la versión 3.0 de la norma.

Requisito 4: Cifrar la transmisión de los datos del titular de la tarjeta en las redes públicas abiertas

- **4.1** Utilice cifrado sólido y protocolos de seguridad para proteger los datos confidenciales del titular de la tarjeta durante la transmisión por redes públicas abiertas.
- **4.1.1** Asegúrese de que las redes inalámbricas que transmiten los datos del titular de la tarjeta o que están conectadas al entorno de datos del titular de la tarjeta utilicen las mejores prácticas de la industria a fin de implementar un cifrado sólido para transmitir y autenticar.
- **4.2** Nunca debe enviar PAN no cifrados por medio de tecnologías de mensajería de usuario final.

- **4.3** Asegúrese de que las políticas de seguridad y los procedimientos operativos para cifrar las transmisiones de los datos del titular de la tarjeta estén documentados, implementados y que sean de conocimiento para todas las partes afectadas.
    - Requisito añadido en la versión 3.0 de la norma.

Requisito 5: Proteger todos los sistemas contra *malware* y actualizar los programas o software antivirus regularmente

A pesar de que este requisito lo cubre Microsoft Azure la empresa dispone de equipos de trabajo que deben ser segurizados debidamente.

- **5.1** Implemente un software antivirus en todos los sistemas que, generalmente, se ven afectados por software malicioso (en especial, computadoras personales y servidores).
- **5.1.1** Asegúrese de que los programas de antivirus puedan detectar y eliminar todos los tipos de software malicioso conocidos y proteger a los sistemas contra estos.
- **5.1.2** Para aquellos sistemas que no suelen verse afectados por *software* maliciosos, lleve a cabo evaluaciones periódicas para identificar y evaluar las amenazas de *malware* que pueden aparecer a fin de determinar si es necesario o no implementar un *software* antivirus en dichos sistemas.
- **5.2** Asegúrese de que los mecanismos de antivirus estén actualizados, ejecuten análisis periódicos y generen registros de auditoría que se guarden de conformidad con el Requisito 10.7 de las PCI DSS.
- **5.3** Asegúrese de que los mecanismos de antivirus funcionen activamente y que los usuarios no puedan deshabilitarlos ni alterarlos, salvo que estén específicamente autorizados por la gerencia en casos particulares y durante un período limitado.
    - Requisito añadido en la versión 3.0 de la norma.
- **5.4** Asegúrese de que las políticas de seguridad y los procedimientos operativos que protegen los sistemas estén documentados, implementados y que sean de conocimiento para todas las partes afectadas.
    - Requisito añadido en la versión 3.0 de la norma.

Requisito 6: Desarrolle y mantenga sistemas y aplicaciones seguras

- **6.1** Establezca un proceso para identificar las vulnerabilidades de seguridad por medio de fuentes externas conocidas para obtener información sobre las vulnerabilidades de seguridad, y asigne una clasificación de riesgo a las vulnerabilidades de seguridad recientemente descubiertas.
    - Cambio de orden con el requisito 6.1 en la versión 3.0 de la norma.
- **6.2** Asegúrese de que todos los *software* y componentes del sistema tengan instalados parches de seguridad proporcionados por los proveedores que ofrecen protección contra vulnerabilidades conocidas. Instale los parches importantes de seguridad dentro de un plazo de un mes de su lanzamiento.
    - Cambio de orden con el requisito 6.2 en la versión 3.0 de la norma.

- **6.3** Desarrolle aplicaciones de *software* internas y externas (incluso acceso administrativo a aplicaciones basado en web) de manera segura.
- **6.3.1** Elimine las cuentas de desarrollo, de prueba y de aplicaciones personalizadas, las ID de usuario y las contraseñas antes de que las aplicaciones se activen o se pongan a disposición de los clientes.
- **6.3.2** Revise el código personalizado antes de enviarlo a producción o de ponerlo a disposición de los clientes a fin de identificar posibles vulnerabilidades en la codificación (mediante procesos manuales o automáticos).
- **6.4** Siga los procesos y procedimientos de control de todos los cambios en los componentes del sistema.
- **6.4.1** Separe los entornos de desarrollo/prueba de los entornos de producción y refuerce la separación con controles de acceso.
- **6.4.2** Separación de funciones entre desarrollo/prueba y entornos de producción.
- **6.4.3** Los datos de producción (PAN activos) no se utilizan para las pruebas ni para el desarrollo.
- **6.4.4** Eliminación de datos y cuentas de prueba antes de que se activen los sistemas de producción.
- **6.4.5** Los procedimientos de control de cambios para implementar los parches de seguridad y las modificaciones en los software deben incluir los siguientes subíndices:
- **6.4.5.1** Documentación de incidencia.
- **6.4.5.2** Aprobación de cambio documentada por las partes autorizadas.
- **6.4.5.3** Verifique que se hayan realizado las pruebas de funcionalidad y que el cambio no impacte negativamente en la seguridad del sistema.
- **6.4.5.4** Procedimientos de desinstalación.
- **6.5** Aborde las vulnerabilidades de codificación comunes en los procesos de desarrollo de *software*.
- **6.5.1** Errores de inyección, en especial, errores de inyección SQL. También considere los errores de inyección de comandos de OS, LDAP y Xpath, así como otros errores de inyección.
- **6.5.2** Desbordamiento de *buffer*.
- **6.5.3** Almacenamiento cifrado inseguro.
- **6.5.4** Comunicaciones inseguras.
- **6.5.5** Manejo inadecuado de errores.
- **6.5.6** odas las vulnerabilidades de "alto riesgo" detectadas en el proceso de identificación de vulnerabilidades (según lo definido en el Requisito 6.1 de las PCI DSS).
- **6.5.7** Lenguaje de comandos entre distintos sitios (XSS).

- **6.5.8** Control de acceso inapropiado (como referencias no seguras a objetos directos, no restricción de acceso a URL y exposición completa de los directorios, y la no restricción de acceso a las funciones por parte de los usuarios).
- **6.5.9** Falsificación de solicitudes entre distintos sitios (CSRF)
- **6.5.10** Autenticación y administración de sesión interrumpidas.
  - Requisito añadido en la versión 3.0 de la norma.
- **6.6** En el caso de aplicaciones web públicas, trate las nuevas amenazas y vulnerabilidades continuamente y asegúrese de que estas aplicaciones se protejan contra ataques conocidos.
- **6.5.7** Asegúrese de que las políticas de seguridad y los procedimientos operativos . para desarrollar y mantener seguros los sistemas y las aplicaciones estén documentados, implementados y que sean de conocimiento para todas las partes afectadas.
  - Requisito añadido en la versión 3.0de la norma.

Requisito 7: Restrinja el acceso a los datos del titular de la tarjeta según la necesidad de saber que tenga la empresa

- **7.1** Limite el acceso a los componentes del sistema y a los datos del titular de la tarjeta a aquellos individuos cuyas tareas necesitan de ese acceso.
- **7.1.1** Defina las necesidades de acceso de cada función.
  - Requisito añadido en la versión 3.0 de la norma.
- **7.1.2** Limite el acceso de usuarios con ID privilegiadas a la menor cantidad de privilegios necesarios para llevar a cabo las responsabilidades del trabajo.
  - Cambio de requisito 7.1.1 a 7.1.2 en la versión 3.0 de la norma.
- **7.1.3** Asigne el acceso según la tarea, la clasificación y la función del personal.
  - Cambio de requisito 7.1.2 a 7.1.3 en la versión 3.0 de la norma.
- **7.1.4** Solicite la aprobación documentada de las partes autorizadas en la que se especifiquen los privilegios necesarios.
  - El requisito 7.1.4 de la versión 2.0 ha sido eliminado por cumplirse en el requisito 7.2 de la nueva versión. Este punto es nuevo en la versión 3.0.
- **7.2** Establezca un sistema de control de acceso para los componentes del sistema que restrinja el acceso según la necesidad del usuario de conocer y que se configure para “negar todo”, salvo que se permita específicamente.
- **7.2.1** Cobertura de todos los componentes del sistema.
- **7.2.2** La asignación de privilegios a una persona se basa en la clasificación del trabajo y su función.
- **7.2.3** Configuración predeterminada de “negar todos”.
- **7.3** Asegúrese de que las políticas de seguridad y los procedimientos operativos para restringir el acceso a los datos del titular de la tarjeta estén documentados, implementados y que sean de conocimiento para todas las partes afectadas.
  - Requisito añadido en la versión 3.0.

Requisito 8: Identificar y autenticar el acceso a los componentes del sistema

- **8.1** Defina e implemente políticas y procedimientos para garantizar la correcta administración de la identificación de usuarios para usuarios no consumidores y administradores en todos los componentes del sistema.
  - Requisito actualizado en la versión 3.0 para distinguir entre la identificación y la autenticación del usuario.
- **8.1.1** Asigne a todos los usuarios una ID exclusiva antes de permitirles acceder a los componentes del sistema o a los datos del titular de la tarjeta.
  - Requisito 8.1 en la versión 2.0 de la norma.
- **8.1.2** Controle la incorporación, la eliminación y la modificación de las ID de usuario, las credenciales y otros objetos de identificación.
  - Requisito 8.5.1 en la versión 2.0 de la norma.
- **8.1.3** Cancele de inmediato el acceso a cualquier usuario cesante.
  - Requisito 8.5.4 en la versión 2.0 de la norma.
- **8.1.4** Elimine o inhabilite las cuentas de usuario inactivas, al menos, cada 90 días.
  - Requisito 8.5.5 en la versión 2.0 de la norma. A pesar de que la responsabilidad de la implementación recae sobre Microsoft Azure, se ha optado por incluirla como propia, para aumentar la seguridad en las cuentas de acceso a los componentes del sistema.
- **8.1.6** Limite los intentos de acceso repetidos mediante el bloqueo de la ID de usuario después de más de seis intentos.
  - Requisito 8.5.13 en la versión 2.0 de la norma.
- **8.1.7** Establezca la duración del bloqueo a un mínimo de 30 minutos o hasta que el administrador habilite la ID del usuario.
  - Requisito 8.5.14 en la versión 2.0 de la norma.
- **8.1.8** Si alguna sesión estuvo inactiva durante más de 15 minutos, solicite al usuario que vuelva a escribir la contraseña para activar la terminal o la sesión nuevamente.
  - Requisito 8.5.15 en la versión 2.0 de la norma.
- **8.2.1** Deje ilegibles todas las credenciales de autenticación durante la transmisión y el almacenamiento en todos los componentes del sistema mediante una criptografía sólida.
  - Requisito 8.4 en la versión 2.0 de la norma.
- **8.2.3** Las contraseñas/frases deben tener una longitud mínima de siete caracteres y estar compuesta por una combinación de caracteres numéricos y alfabéticos.
  - Requisitos 8.5.10 y 8.5.11 en la versión 2.0 de la norma.
- **8.2.4** Cambie la contraseña/frase de usuario, al menos, cada 90 días.
  - Requisito 8.5.9 en la versión 2.0 de la norma.

- **8.2.5** No permita que una persona envíe una contraseña/frase nueva que sea igual a cualquiera de las últimas cuatro contraseñas/frases utilizadas.
  - Requisito 8.5.12 en la versión 2.0 de la norma.
- **8.2.6** Configure la primera contraseña/frase y las restablecidas en un valor único para cada usuario y cámbiela de inmediato después del primer uso.
  - Requisito 8.5.3 en la versión 2.0 de la norma.
- **8.4** Documente y comunique los procedimientos y las políticas de autenticación a todos los usuarios.
  - Requisito 8.5.7 en la versión 2.0 de la norma.
- **8.5** No use ID ni contraseñas de grupo, compartidas ni genéricas, ni otros métodos de autenticación.
  - Requisito 8.5.9 en la versión 2.0 de la norma.
- **8.6** Si se utilizan otros mecanismos de autenticación se deben asignar a una sola cuenta e implementar controles físicos y lógicos para garantizar que solo la cuenta deseada usa esos mecanismos para acceder.
  - Requisito añadido en la versión 3.0 de la norma.
- **8.7** Se restringen todos los accesos a cualquier base de datos que contenga datos del titular de la tarjeta.
  - Requisito 8.5.16 en la versión 2.0 de la norma.
- **8.8** Asegúrese de que las políticas de seguridad y los procedimientos operativos de identificación y autenticación estén documentados, implementados y que sean de conocimiento para todas las partes afectadas.
  - Requisito añadido en la versión 3.0 de la norma.

Requisito 9: Restringir el acceso físico a los datos del titular de la tarjeta

Dada la infraestructura de Pay[In] en la que no se mantiene ni almacena ningún tipo de información a nivel local, los controles de acceso físicos se delegan en la plataforma de Azure, cumpliendo ya todos los requisitos del apartado noveno.

Requisito 10: Rastree y supervise todos los accesos a los recursos de red y a los datos de los titulares de las tarjetas

- **10.2.1** Implemente pistas de auditoría automáticas en todos los componentes del sistema a fin de reconstruir todo acceso por parte de usuarios a los datos del titular de la tarjeta.
- **10.5.1** Limite la visualización de las pistas de auditoría a quienes lo necesiten por motivos laborales.
- **10.8** Asegúrese de que las políticas de seguridad y los procedimientos operativos para monitorizar todos los accesos a los recursos de la red y a los datos del titular de la tarjeta estén documentados, implementados y que sean de conocimiento para todas las partes afectadas.

Requisito 11: Pruebe con regularidad los sistemas y procesos de seguridad

- **11.2.1** Lleve a cabo análisis trimestrales de las vulnerabilidades internas y vuelva a repetir el análisis cuantas veces sea necesario hasta corregir todas las vulnerabilidades de "alto riesgo" (según lo estipulado en el Requisito 6.1). Los análisis deben estar a cargo de personal calificado.
- **11.2.2** Los análisis trimestrales de vulnerabilidades externas deben estar a cargo de un ASV (proveedor aprobado de escaneo) que esté certificado por el PCI SSC (PCI Security Standards Council). Vuelva a realizar los análisis cuantas veces sea necesario hasta que todos los análisis estén aprobados.
- **11.2.3** Lleve a cabo análisis internos y externos, y repítalos, según sea necesario, después de realizar un cambio significativo. Los análisis deben estar a cargo de personal calificado.
- **11.3** Implemente una metodología para las pruebas de penetración.
  - Requisito añadido en la versión 3.0 de la norma.
- **11.3.1** Lleve a cabo pruebas de penetración externas, al menos, una vez al año y después de implementar una actualización o modificación significativa en las infraestructuras o aplicaciones.
  - Requisito 11.3 dividido en en la versión 2.0 de la norma. Este ha sido dividido en dos, 11.3.1 y 11.3.2.
- **11.3.2** Lleve a cabo pruebas de penetración internas, al menos, una vez al año y después de implementar una actualización o modificación significativa en las infraestructuras o aplicaciones.
  - Requisito 11.3 en en la versión 2.0 de la norma. Este ha sido dividido en dos, 11.3.1 y 11.3.2.
- **11.3.3** Las vulnerabilidades de seguridad detectadas en las pruebas de penetración se corrigen, y las pruebas se repiten para verificar las correcciones.
  - Requisito añadido en la versión 3.0 a partir del requisito 11.3.b en la versión 2.0 de la norma.
- **11.6** Asegúrese de que las políticas de seguridad y los procedimientos operativos para monitorizar y comprobar la seguridad estén documentados, implementados y que sean de conocimiento para todas las partes afectadas.
  - Requisito añadido en la versión 3.0 de la norma.

Requisito 12: Mantener una política que aborde la seguridad de la información de todo el personal

Los requisitos 12.1.1 y 12.2 han sido recombinados y añadidos enotros puntos de la norma según su finalidad. Los requisitos resultantes son: 1.5, 2.5, 3.7, 4.3, 5.4, 6.7, 7.3, 8.8, 9.10, 10.8 y 11.6.

- **12.1** Establezca, publique, mantenga y distribuya una política de seguridad.
  - Requisito añadido en la versión 3.0 de la norma.

- **12.1.1** Revise la política de seguridad, al menos, una vez al año y actualícela cuando se realicen cambios en el entorno.
    - Requisito 12.1.3 en la versión 2.0 de la norma.
- **12.2** Implemente un proceso de evaluación de riesgos.
    - Requisito 12.1.2 en la versión 2.0 de la norma.
- **12.3** Desarrolle políticas de uso para las tecnologías críticas y defina cómo usarlas correctamente.
- **12.3.1** Aprobación explícita de las partes autorizadas.
- **12.3.2** Autenticación para el uso de la tecnología.
- **12.3.3** Lista de todos los dispositivos y el personal que tenga acceso.
- **12.3.4** Método para determinar, con exactitud y rapidez, el propietario, la información de contacto y el objetivo.
- **12.3.5** Usos aceptables de la tecnología.
- **12.3.6** Ubicaciones aceptables de las tecnologías en la red.
- **12.3.7** Lista de productos aprobados por la empresa.
- **12.3.8** Desconexión automática de sesiones para tecnologías de acceso remoto después de un período específico de inactividad.
- **12.3.9** Activación de las tecnologías de acceso remoto para proveedores y socios de negocio sólo cuando sea necesario, con desactivación inmediata después de su uso.
- **12.3.10** En el caso del personal que tiene acceso a los datos del titular de la tarjeta mediante tecnologías de acceso remoto, prohíba copiar, mover y almacenar los datos del titular de la tarjeta en unidades de disco locales y en dispositivos electrónicos extraíbles, a menos que sea autorizado explícitamente para una necesidad comercial definida.
- **12.4** Asegúrese de que las políticas y los procedimientos de seguridad definan, claramente, las responsabilidades de seguridad de la información de todo el personal.
- **12.5** Asigne a una persona o a un equipo responsabilidades de administración de seguridad de la información.
- **12.5.1** Establezca, documente y distribuya las políticas y los procedimientos de seguridad.
- **12.5.2** Monitorice y analice las alertas y la información de seguridad y comuníquelas al personal correspondiente.
- **12.5.3** Establezca, documente y distribuya los procedimientos de escalamiento y respuesta ante incidentes de seguridad para garantizar un manejo oportuno y efectivo de todas las situaciones.
- **12.5.4** Administre las cuentas de usuario, incluso las incorporaciones, eliminaciones y modificaciones.
- **12.5.5** Monitorice y controle todo acceso a los datos.

- **12.6** Implemente un programa formal de concienciación sobre seguridad para que todo el personal tome conciencia de la importancia de la seguridad de los datos del titular de la tarjeta.
- **12.6.1** Capacite al personal inmediatamente después de contratarlo y, al menos, una vez al año.
- **12.6.2** Exija al personal que realice, al menos, una vez al año, una declaración de que leyeron y entendieron la política y los procedimientos de seguridad de la empresa.
- **12.7** Examine al personal potencial antes de contratarlo a fin de minimizar el riesgo de ataques desde fuentes internas.
- **12.8** Mantenga e implemente políticas y procedimientos para administrar los proveedores de servicios con quienes se compartirán datos del titular de la tarjeta, o que podrían afectar la seguridad de los datos del titular de la tarjeta.
- **12.8.1** Mantenga una lista de proveedores de servicios.
- **12.8.2** Mantenga un acuerdo por escrito en el que los proveedores de servicios aceptan responsabilizarse de la seguridad de los datos del titular de la tarjeta que ellos poseen, almacenan, procesan o transmiten en nombre del cliente, o en la medida en que puedan afectar la seguridad del entorno de datos del titular de la tarjeta del cliente.
- **12.8.3** Asegúrese de que exista un proceso establecido para comprometer a los proveedores de servicios, que incluya una auditoría adecuada previa al compromiso.
- **12.8.4** Mantenga un programa para monitorear el estado de cumplimiento de las PCI DSS por parte del proveedor de servicios.
- **12.8.5** Conserve información sobre cuáles son los requisitos de las PCI DSS que administra cada proveedor de servicios y cuáles administra la entidad.
  - Requisito añadido en la versión 3.0 de la norma.
- **12.9** Los proveedores de ser vicios aceptan, por escrito y ante los clientes, responsabilizarse de la seguridad de los datos del titular de la tarjeta que ellos poseen, almacenan, procesan o transmiten en nombre del cliente, o en la medida en que puedan afectar la seguridad del entorno de datos del titular de la tarjeta del cliente.
  - Requisito añadido en la versión 3.0 de la norma.
- **12.10** Implemente un plan de respuesta ante incidentes. Prepárese para responder de inmediato ante un fallo en el sistema.
- **12.10.1** Desarrolle el plan de respuesta ante incidentes que se implementará en caso de que ocurra una falla del sistema.
- **12.2** Implemente un proceso de evaluación de riesgos.
  - Requisito 12.9.1 en la versión 2.0 de la norma.
- **12.10.2** Pruebe el plan, al menos, una vez al año.
  - Requisito 12.9.2 en la versión 2.0 de la norma.

- **12.10.3** Designe a personal específico para que esté disponible las 24 horas al día, los 7 días de la semana para responder a las alertas.
  - Requisito 12.9.3 en la versión 2.0 de la norma.
- **12.10.4** Capacite adecuadamente al personal sobre las responsabilidades de respuesta ante fallas de seguridad.
  - Requisito 12.9.4 en la versión 2.0 de la norma.
- **12.10.5** Incluya alertas de los sistemas de monitorización de seguridad, que incluye, entre otros, sistemas de intrusión-detección y de intrusión-prevención, *firewalls* y sistemas de monitorización de integridad de archivos.
  - Requisito 12.9.5 en la versión 2.0 de la norma.
- **12.10.6** Elabore un proceso para modificar y desarrollar el plan de respuesta ante incidentes según las lecciones aprendidas e incorporar los desarrollos de la industria.
  - Requisito 12.9.6 en la versión 2.0 de la norma.