



MIGRACIÓN A LA NUBE DE LOS SISTEMA DE PRODUCCIÓN Y DESARROLLO DE RDMES TECHNOLOGIES S.L.

MISTIC: MASTER INTERUNIVERSITARIO DE SEGURIDAD
DE LAS TECNOLOGÍAS DE LA INFORMACIÓN Y DE LAS COMUNICACIONES.

Memoria del trabajo final de master para los estudios de Seguridad de las Tecnologías de la Información y de las Comunicaciones presentada por Aitor González Valero y codirigida por María Francisca Hinarejos Campos y Jordi Cadafalch.

Universitat Oberta de Catalunya, RDmes Technologies S.L.
Terrassa, Enero del 2014

Resumen (Abstract)

RDmes es una *spin-off* de la Universitat Politècnica de Catalunya (UPC) fundada el 2006 con el objetivo de desarrollar una versión comercial de un conjunto de herramientas estructuradas entorno a la **Oficina Técnica Virtual**. Estas herramientas permiten el diseño a través de la web de forma automática y centralizada de sistemas y componentes de Mecánica de Fluidos e Ingeniería Térmica, como son: Energía Solar Térmica, Energía Geotérmica, Cargas Térmicas en Edificios, Energía Solar Fotovoltaica, Medidas de Eficiencia Energética en los Edificios, Máquinas y Motores Térmicos, Biomasa , Biocombustibles, etc.

La base tecnológica de la empresa está diseñada desde un punto de vista genérico basado en la implementación de la metodología de **ingeniería moderna** en la que se combinan Física, Modelos matemáticos, Modelos Numéricos Computerizados y Experimentación y Prototipaje.

En la actualidad, parte de estas herramientas ya han sido diseñadas y puestas a disposición del público a través de la plataforma web OmniluS®, la cual está hospedada en un servidor privado en la Universitat Politècnica de Catalunya (UPC). Debido al alto coste computacional y la nula flexibilidad del sistema de producción, se hace patente la necesidad de migrarla hacia otros servicios más adecuados.

Además, el sistema actual para el desarrollo no permite al programador trabajar desde cualquier ubicación, puesto que dicho sistema no cuenta con una infraestructura de red preparada para ello, y por razones de trabajo, se hace necesario contar con esta posibilidad.

Con tal de dar solución a estos problemas la empresa RDmes ha decidido apostar por migrar los sistemas actuales a la nube, ya que considera que también puede aportar un ahorro de energía considerable, así como facilitar el mantenimiento y mejorar la actual política de copias de respaldo y recuperación ante desastres.

Índice

Índice de ilustraciones.....	6
Índice de tablas.....	8
1. Introducción.....	10
1.1 Objetivos.....	10
1.2 Requisitos.....	12
1.3 Planificación.....	14
1.3.1 Fases del proyecto.....	14
1.3.2 Puntos de control.....	14
1.3.3 Recursos necesarios.....	15
1.3.4 Cronograma.....	15
1.3.5 Evaluación de riesgos.....	16
2. Análisis sobre los sistemas actuales.....	17
2.1 Análisis de infraestructura.....	17
2.2 Análisis de seguridad.....	17
2.3 Diagramas.....	19
2.4 Modelo de referencia.....	21
3. Estudio de las soluciones existentes.....	24
3.1 Amazon AWS.....	24
3.2 DigitalOcean.....	27
3.3 DinaHosting.....	29
3.4 Heroku.....	31
3.5 Joyent™.....	31
3.6 Linode.....	33
3.7 OpenShift.....	35
3.8 RackSpace®.....	35
3.9 VMware vCloud®.....	37
3.10 Windows Azure.....	37
3.11 Resumen.....	40
3.12 Conclusiones.....	41
4. Migración de los sistemas actuales.....	42
4.1 Introducción.....	42
4.2 El proceso.....	43
4.3 El resultado.....	44
5. Conclusiones.....	47
5.1 Líneas futuras.....	47
Bibliografía.....	49
Anexo 01 (Guías de migración)	
1. Guía de migración para producción.....	60
1.1 Creación y configuración del entorno.....	60
1.2 Instalación de los paquetes necesarios.....	62

1.3	Instalación de los paquetes de cálculo.....	63
1.4	Instalación de la interfaz web y configuración.....	63
1.5	Instalación de la base de datos en RDS.....	65
1.6	Adaptación de la aplicación a RDS.....	67
1.7	Adaptación de la aplicación a S3.....	68
2.	Guía de migración para desarrollo.....	72
3.	Guía de implementación de copias de respaldo.....	75
3.1	Respaldo de instancias.....	75
3.2	Respaldo del EBS de desarrollo.....	75
3.3	Respaldo de la base de datos.....	76
3.4	Respaldo de los ficheros de S3.....	76
4.	Guía de implementación para la supervisión y el autoescalado.....	77
ANEXO 02 (Manual de uso de Amazon AWS)		
1.	Gestión de cuenta en Amazon AWS.....	81
1.1	Creación de cuenta.....	81
1.2	Configuración de doble autenticación.....	86
1.3	Creación de claves o certificados.....	89
2.	Amazon AWS EC2 (Elastic Cloud).....	91
2.1	Creación de instancias.....	91
2.2	Eliminación de instancias.....	95
2.3	Creación de claves (Key Pairs).....	97
2.4	Eliminación de claves (Key Pairs).....	99
2.5	Creación de imágenes de máquina.....	101
2.6	Eliminación de imágenes de máquina.....	102
2.7	Gestión del acceso (Firewall).....	104
2.8	Gestión de IPs estáticas.....	105
2.9	Ampliación del EBS Ephemeral de instancia por defecto.....	109
2.10	Uso de LoadBalancer.....	110
3.	Amazon RDS.....	114
3.1	Creando una base de datos.....	114
3.2	Cambiando los "Parameter Group"......	118
4.	Amazon S3.....	122
4.1	Creación del servicio.....	122
5.	CloudWatch.....	124

Índice de ilustraciones

Ilustración 1 - Diagrama del entorno de producción actual.....	19
Ilustración 2 - Diagrama del entorno de desarrollo actual	20
Ilustración 3 - Diagrama del ciclo de copias de respaldo actual.....	20
Ilustración 4 - Diagrama del ciclo de desarrollo actual	21
Ilustración 5 - Estructura de referencia para la migración.....	21
Ilustración 6 - Ciclo de desarrollo previsto.....	22
Ilustración 7 - Ciclo de copias de respaldo previsto.....	23
Ilustración 8 - Ciclo de copias de respaldo tras la migración.....	44
Ilustración 9 - Ciclo de desarrollo y testing tras la migración	45
Ilustración 10 - Estructura de red y servicios tras la migración	46
Ilustración 11 - Paso previo a la inscripción 01 de 02: Localizando la opción.	81
Ilustración 12 - Paso previo a la inscripción 02 de 02: Preformulario.....	82
Ilustración 13 - Inscripción paso 01 de 06: Credenciales.....	82
Ilustración 14 - Inscripción paso 02 de 06: Datos de contacto	83
Ilustración 15 - Inscripción paso 03 de 06: Datos de facturación.	84
Ilustración 16 - Inscripción paso 04 de 06: Impuestos.	84
Ilustración 17 - Inscripción paso 05 de 06: Verificación de identidad	85
Ilustración 18 - Inscripción paso 06 de 06: Plan de soporte.....	86
Ilustración 19 - Doble autenticación: Página de inicio	86
Ilustración 20 - Doble autenticación: Localización de la opción en el panel de control	87
Ilustración 21 - Doble autenticación paso 01 de 04: Selección del tipo de dispositivo	87
Ilustración 22 - Doble autenticación paso 02 de 04: Mensaje de instrucción.....	88
Ilustración 23 - Doble autenticación paso 03 de 04: Sincronización de los dispositivos.....	88
Ilustración 24 - Doble autenticación paso 04 de 04: Confirmación	89
Ilustración 25 - Creación de clave/certificados: Inicio	90
Ilustración 26 - Creación de clave/certificados: Sección de configuración	90
Ilustración 27 - EC2 Creación de instancia: Inicio	91
Ilustración 28 - EC2 Creación de instancia: Panel principal AWS Management.	91
Ilustración 29 - EC2 Creación de instancia: Cambio de región	92
Ilustración 30 - EC2 Creación de instancia paso 01 de 06: Iniciando el proceso.....	92
Ilustración 31 - EC2 Creación de instancia paso 02 de 06: Selección de AMI	93
Ilustración 32 - EC2 Creación de instancia paso 03 de 06: Selección de tipo de instancia.....	93
Ilustración 33 - EC2 Creación de instancia paso 04 de 06: Descripción general del sistema a montar	94
Ilustración 34 - EC2 Creación de instancia paso 05 de 06: Configuración key pair	94
Ilustración 35 - EC2 Creación de instancia paso 06 de 06: Confirmación.....	95
Ilustración 36 - EC2 Creación de instancia: Visualización	95
Ilustración 37 - EC2 Eliminación de instancia paso 01 de 02: Selección de instancia.....	96
Ilustración 38 - EC2 Eliminación de instancia paso 02 de 02: Aviso y confirmación	96
Ilustración 39 - EC2 Eliminación de instancia: Comprobación.....	97
Ilustración 40 - EC2 Creación de claves: Inicio	97
Ilustración 41 - EC2 Creación de claves: Interfaz de gestión	98
Ilustración 42 - EC2 Creación de claves: Formulario de creación de Key Pair	98
Ilustración 43 - EC2 Creación de claves: Formulario para la importación de Key Pairs	99
Ilustración 44 - EC2 Eliminación de claves: Inicio.....	100
Ilustración 45 - EC2 Eliminación de claves: Selección y eliminación	100
Ilustración 46 - EC2 Eliminación de claves: Confirmación	100
Ilustración 47 - Creación de AMI: Inicio.....	101
Ilustración 48 - Creación de AMI: Formulario de configuración.....	102
Ilustración 49 - Eliminación de AMI: Inicio	102
Ilustración 50 - Eliminación de AMI: Aviso y confirmación	103
Ilustración 51 - Eliminación de AMI: Borrado de snapshot asociado	103
Ilustración 52 - Eliminación de AMI: Aviso y confirmación del borrado del snapshot asociado.....	104
Ilustración 53 - Gestión del firewall: Interfaz	104
Ilustración 54 - Gestión de IPs estáticas: Inicio de creación	105
Ilustración 55 - Gestión de IPs estáticas: Aviso de creación y confirmación	106
Ilustración 56 - Gestión de IPs estáticas: inicio de asociación	106
Ilustración 57 - Gestión de IPs estáticas: Formulario de asociación	106
Ilustración 58 - Gestión de IPs estáticas: Confirmación de la asociación	107
Ilustración 59 - Gestión de IPs estáticas: Disociación	107

Ilustración 60 - Gestión de IPs estáticas: Confirmación para la disociación.....	108
Ilustración 61 - Gestión de IPs estáticas: Revocación	108
Ilustración 62 - Gestión de IPs estáticas: Confirmación de revocación.....	109
Ilustración 63 - Ampliación del tamaño del volumen por defecto (EBS): Formulario	110
Ilustración 64 - LoadBalancer: Inicio	110
Ilustración 65 - Creación de LoadBalancer paso 01 de 05: Definición	111
Ilustración 66 - Creación de LoadBalancer paso 02 de 05: Configuración de monitorización.....	111
Ilustración 67 - Creación de LoadBalancer paso 03 de 05: Adición de instancias EC2 01	112
Ilustración 68 - Creación de LoadBalancer paso 04 de 05: Adición de instancias EC2 02	112
Ilustración 69 - Creación de LoadBalancer paso 05 de 05: Resumen y confirmación.....	113
Ilustración 70 - RDS Creación de base de datos: Localización	114
Ilustración 71 - RDS Creación de base de datos: Inicio	114
Ilustración 72 - RDS Creación de base de datos paso 01 de 05: Selección de tipo de instancia.....	115
Ilustración 73 - RDS Creación de base de datos paso 02 de 05: Especificación de los detalles de la instancia....	115
Ilustración 74 - RDS Creación de base de datos paso 03 de 05: Configuración adicional de la instancia.....	116
Ilustración 75 - RDS Creación de base de datos paso 04 de 05: Opciones de gestión	116
Ilustración 76 - RDS Creación de base de datos paso 05 de 05: Resumen.....	117
Ilustración 77 - RDS Creación de base de datos: Comprobación de estado	117
Ilustración 78 - RDS Cambio de parámetros: Inicio	118
Ilustración 79 - RDS Cambio de parámetros paso 01 de 06: Creando un grupo de configuración.....	118
Ilustración 80 - RDS Cambio de parámetros paso 02 de 06: Formulario para la creación del grupo	119
Ilustración 81 - RDS Cambio de parámetros paso 03 de 06: Inicio de la edición del grupo de configuración	119
Ilustración 82 - RDS Cambio de parámetros paso 04 de 06: Búsqueda del parámetro a configurar.....	120
Ilustración 83 - RDS Cambio de parámetros paso 05 de 06: Asignación del grupo de configuración 01	120
Ilustración 84 - RDS Cambio de parámetros paso 06 de 06: Asignación del grupo de configuración 02	121
Ilustración 85 - S3: Inicio	122
Ilustración 86 - S3 Creación de bucket paso 01 de 03: Inicio.....	122
Ilustración 87 - S3 Creación de bucket paso 02 de 03: Formulario de creación	123
Ilustración 88 - S3 Creación de bucket paso 03 de 03: Configuración y visualización.....	123
Ilustración 89 - CloudWatch: Activación de la monitorización en una instancia EC2	124

Índice de tablas

Tabla 1 - Clasificación de los objetivos según su priorización.	12
Tabla 2 - Clasificación de los requisitos según su priorización.....	13
Tabla 3 - Fases del proyecto.	14
Tabla 4 - Puntos de control.	14
Tabla 5 - Recursos necesarios.....	15
Tabla 6 - Planificación.	15
Tabla 7 - Cronograma.	16
Tabla 8 - Listado de riesgos.	16
Tabla 9 - Soluciones e impacto de los riesgos presentes.	16
Tabla 10 - Resumen del análisis de las soluciones existentes.....	40

1. Introducción

La empresa RDmes Technologies S.L., promotora de este proyecto y spin-off de la Universitat Politècnica de Catalunya (UPC), se encarga del desarrollo y de la explotación de un conjunto de herramientas estructuradas para el diseño de sistemas y componentes de Mecánica de Fluidos e Ingeniería Térmica.

Con el objetivo de acercar estas herramientas al público y facilitar su uso la empresa desarrolló la plataforma OmniliuS®, un aplicativo web, actualmente accesible desde Internet, con una interfaz amigable enfocada tanto a personas con un perfil técnico alto (licenciados, diplomados, etc.) como a personas con un perfil técnico bajo en el campo de la Ingeniería Mecánica y Térmica.

La plataforma y las herramientas se encuentran hospedadas en un servidor privado de la empresa situado en la Universitat Politècnica de Catalunya (UPC) y están en constante desarrollo en el Institut Politècnic Campus Terrassa (IPCT), lugar donde está situada la oficina técnica de RDmes.

Para el proyecto es importante prestar atención a los entornos que permiten la explotación de la aplicación. Concretamente, si nos fijamos en el párrafo anterior, podemos apreciar que existen dos entornos bien diferenciados: el de producción, que corresponde al servidor privado situado en la Universitat Politècnica de Catalunya (UPC), y el de desarrollo, que corresponde a los sistemas situadas en la oficina técnica de RDmes destinados a este fin, concretamente un servidor para realizar pruebas de funcionamiento y los ordenadores de los empleados implicados en la programación de los aplicativos.

Ambos entornos presentan limitaciones para la explotación de los aplicativos (plataforma y herramientas). En el caso del entorno de producción, debido al coste computacional de las herramientas y la poca flexibilidad del sistema (un servidor no escalable), la explotación se ve limitada a un número reducido de usuarios.

Por otra lado, en el caso del entorno de desarrollo, las limitaciones se derivan de la infraestructura de red: la oficina técnica cuenta con una conexión hacia Internet de un 1Mb/s la cual no puede ampliarse, factor que no permite a los desarrolladores trabajar desde una ubicación distinta a la oficina y hace que la corrección de errores o la actualización de los aplicativos en producción sea lenta, durante el proceso y en el tiempo, debido a que estas acciones se han de realizar desde el servidor de *testing* situado en este entorno.

La empresa considera de vital importancia solventar estas limitaciones para poder realizar una apuesta segura en la plataforma y garantizar un servicio adecuado (que no presente problemas de saturación) a los usuarios. Con esta finalidad ha decidido apostar por migrar los sistemas actuales a la nube, ya que considera que también puede aportar un ahorro económico frente a los gastos actuales, así como facilitar el mantenimiento y abrir las vías para mejorar, de forma general, ambos entornos, tanto a nivel lógico como físico.

Con el fin de recoger todos aquellos factores que han sido determinantes a la hora de tomar una decisión frente a la migración, así como todos aquellos pasos previos y posteriores que han hecho que el proyecto haya sido viable, se presenta este documento. Por lo tanto, podemos encontrar aquí descritos los objetivos que se persiguen, los requisitos a cumplir, la planificación general para llevar a cabo el proyecto en su conjunto, un análisis de los sistemas actuales, un estudio sobre las soluciones existentes, el desarrollo del proceso de migración y las conclusiones finales tras finalizar esta.

Concretamente en este apartado quedan descritos los objetivos, los requisitos y la planificación.

1.1 Objetivos

Los objetivos del proyecto vienen directamente ligados a las limitaciones de los entornos actuales, es decir, con la migración se pretende solucionar aquellas restricciones que en la actualidad nos limitan en nuestro trabajo y en nuestra apuesta por la aplicación.

Los objetivos son distintos para cada entorno, pues cada uno tiene sus propias limitaciones a solventar, de esta forma tenemos que, en producción, se pretende dar solución a:

SPO-01. *Poca flexibilidad.*

Este factor, aunque en la actualidad no es algo determinante, se perfila como un problema a largo plazo.

Seguir con el sistema actual implica que en picos de uso, o en un aumento repentino de los usuarios, el sistema no pueda ofrecer una experiencia satisfactoria y que la empresa, para solventarlo, tenga que comprar otro servidor y situarlo junto al ya existente. Este proceso se puede demorar en el tiempo,

provocando grandes repercusiones para la empresa, como puede ser, la pérdida de confianza de los clientes. Además, si la intención es cubrir picos de uso, se estaría pagando por unos recursos que no se usarían la mayor parte del tiempo.

SPO-02. Alto coste de escalabilidad y mantenimiento.

Comprar un servidor tiene un coste anual alto frente a otras alternativas. Hay que pagar el alquiler del *rack*, amortizarlo, cubrir los gastos de conexión, monitorizarlo, etc.

Un servidor dedicado, virtual o compartido en la nube puede llegar a reducir estos costes de forma considerable, aunque estas soluciones sigan sin ser del todo flexibles. Aun así, existen alternativas que pueden cumplir con la flexibilidad y el escalado a la vez.

SPO-03. Política de copias de respaldo y recuperación de desastres.

La política actual de recuperación de desastres aplicada al servidor de producción, la cual se describe en el apartado 2.2 *Análisis de seguridad*, es realmente pobre e ineficiente, además es totalmente dependiente del sistema de desarrollo.

El cumplimiento de este objetivo es realmente importante, pues la empresa se puede ver comprometida de un momento a otro.

SPO-04. Tiempos de inactividad.

Al estar situado el sistema en la Universitat Politècnica de Catalunya (UPC) este está vinculado a las horas de mantenimiento que dicha universidad hace sobre sus servidores. Esto causa, que fuera de nuestro control, en ocasiones, la empresa se encuentre con el servidor de producción apagado.

Una migración a la nube podría reducir o eliminar estos tiempos de inactividad.

SPO-05. Nula política de actualizaciones.

Gran parte del *software* usado actualmente en los distintos sistemas es anticuado. Migrar favorecerá que la empresa actualice los sistemas actuales mejorando de esta forma la seguridad y la estabilidad de los mismos.

Y en el entorno de desarrollo a:

SDO-01. Poca credibilidad del entorno de testing.

En la actualidad, para probar las modificaciones en la aplicación ésta se empaqueta y se sube a un servidor en el entorno de desarrollo. El problema es que este servidor no tiene las mismas especificaciones ni el mismo *software* que el de producción, lo que provoca que el resultado de las pruebas no sea extrapolable a este último sistema.

Por otro lado, el entorno no nos permite simular usuarios a gran escala, no pudiendo analizar así cómo se comportaría la aplicación ante un gran número de estos.

SDO-02. Poca movilidad de los desarrolladores.

El servidor de desarrollo, debido a su ubicación, tiene limitada la conexión hacia el exterior a 1Mb/s. Esto implica que, por razones de infraestructura, se hace prácticamente imposible para los desarrolladores trabajar desde una ubicación diferente a las oficinas, una funcionalidad que cada día es más necesaria.

También se pretende en un futuro que gente ajena al proyecto pueda colaborar con nosotros, por lo que es indispensable que a la larga se cumpla este punto.

SDO-03. Diversidad de las distintas máquinas.

Cada desarrollador usa su propio entorno de programación a nivel de sistema operativo, lo que entorpece la puesta en producción y el *testing*. Migrar hacia otro servicio u otro sistema es un paso que puede ayudar a fijar los criterios de desarrollo.

A continuación, en la **Tabla 1**, mediante el sombreado gris, se presentan los objetivos clasificados según su relevancia en el proyecto, siendo los críticos aquellos que se han de alcanzar en todo momento y sin los cuales no tendría sentido el proyecto; los prioritarios, aquellos que no tienen por qué cumplirse en la actualidad pero que se tendrían que tener en cuenta para un futuro próximo; y los secundarios, aquellos que se consideran irrelevantes a corto plazo y de los cuales no depende el resultado final. Teniendo en cuenta esto, podemos marcarnos un camino a seguir según la importancia de los mismos, dando máxima prioridad a los críticos, posteriormente a los prioritarios y finalmente, a los secundarios.

Sistema de producción				Sistema de desarrollo			
N. Objetivo	Crítico	Prioritario	Secundario	N. Objetivo	Crítico	Prioritario	Secundario
SPO-01				SDO-01			
SPO-02				SDO-02			
SPO-03				SDO-03			
SPO-04							
SPO-05							

Tabla 1 - Clasificación de los objetivos según su priorización.

1.2 Requisitos

Los requisitos, en este caso, los podemos dividir en dos vertientes: aquellos que busca la empresa en los servicios en la nube así como en su proveedor, y aquellos enfocados en la migración. Estos últimos estrictamente ligados a los objetivos.

Los requisitos que busca la empresa en los servicios en la nube así como en su proveedor son:

SR-01. *El acceso al servicio ha de ser seguro.*

Si hay API's (*Application Programming Interface*) o aplicaciones web para la gestión de los servicios estas han de ser lo suficientemente maduras y seguras como para no suponer un riesgo de seguridad.

SR-02. *El servicio contratado debe tener asociado un sistema de soporte.*

Se busca un sistema de soporte ágil y eficiente, capaz de dar respuesta a un problema o consulta en el mínimo periodo de tiempo posible.

SR-03. *El proveedor ha de ser fiable.*

Este debe de poder demostrar que lo es, ya sea mediante certificados, como el proporcionado por el cumplimiento de la normativa ISO27001 [1], o mediante el prestigio alcanzado. De esta manera, la empresa podría proceder a la migración con cierta tranquilidad.

SR-04. *Privacidad en los datos alojados.*

El servicio ha de garantizar la privacidad del contenido alojado en sus servicios, así como la autoridad de este: no por el hecho de alojar nuestro contenido en su servicio hemos de dejar de ser sus únicos dueños.

SR-05. *Borrado permanente tras nuestra elección.*

Los datos se deberían de eliminar de forma permanente en el momento que se borran, ya sea por motivos de funcionamiento o por abandono del servicio. No deberían quedarse en los sistemas de proveedor de forma oculta.

SR-06. *Libertad de gestión.*

El servicio debe permitirnos, en lo máximo posible, gestionar el servidor con total libertad: instalar nuestro *software*, sistemas de auditoría, etc.

SR-07. *Transparencia.*

Hemos de poder conocer los detalles de la infraestructura del proveedor, así como otros detalles técnicos, que nos puedan ayudar frente a la elaboración de planes de seguridad o frente a problemas legales.

SR-08. *Incorporación de sistemas de alerta y supervisión.*

Un sistema de auditoría que nos permita en todo momento saber el estado actual del sistema, con la funcionalidad de poder programar automatizaciones y notificaciones.

SR-09. *Próximo y motivador.*

Un servicio cercano, en continuo progreso e innovador, que comunique los avances y motive a los usuarios, por ejemplo, con charlas formativas, congresos, etc.

SR-10. *Disponibilidad y replicación de los datos.*

Hemos de disponer de la información alojada sin restricciones y en cualquier momento, pudiéndola replicar en otros soportes fuera de los servicios que nos ofrecen, como un disco duro en local.

SR-11. Cumplimiento de la legalidad.

El proveedor ha de cumplir con las leyes nacionales e internacionales. Es un punto de cara al departamento financiero y la ley de protección de datos.

SR-12. Cancelación sin repercusiones.

No ha de existir un compromiso a largo plazo con el proveedor. Se debe de poder cancelar el servicio de forma rápida y sin repercusión económica.

Y aquellos que busca la empresa para la migración son:

MR-01. Ventaja económica.

Ha de suponer una ventaja económica frente a los sistemas actuales permitiendo a la empresa bajar sus costes a largo o corto plazo.

MR-02. Mejora en la política de copias de respaldo y recuperación de desastres.

Debe de mejorar todo el sistema actual frente a contingencias y copias de respaldo.

MR-03. Mejora de la movilidad.

Como resultado de una migración de éxito ha de garantizar la movilidad a los desarrolladores.

MR-04. Mejora en la fidelidad testing – producción.

La solución escogida ha de permitir recrear el sistema de *testing* de la forma más semejante posible al de producción, de esta manera, las pruebas serán totalmente fiables.

MR-05. Facilidad en la adaptación.

Dentro del cumplimiento de los objetivos y los puntos descritos, se ha de escoger un sistema que nos permita migrar y adaptarnos al nuevo contexto con facilidad.

MR-06. Escalable.

El nuevo sistema se ha de poder adaptar frente a la carga de usuarios de forma casi inmediata sin requerir una gran inversión por parte de la empresa, con tal de garantizar el buen funcionamiento de la plataforma ante eventuales picos de uso.

MR-07. Agilidad.

La migración ha de proporcionar agilidad, permitir pasar la plataforma de desarrollo a producción de forma rápida y sencilla, sin demoras excesivas ni complicaciones.

MR-08. Compatibilidad.

El nuevo sistema ha de funcionar bajo sistemas Linux, sistema operativo para el que están programados los cálculos. La misma lógica se ha de aplicar a la base de datos y otros componentes de *software* dependientes del sistema operativo.

Con tal de marcarnos una hoja de ruta, quedan estos clasificados en la **Tabla 2** según su prioridad.

Requisitos de los proveedores y servicios en la nube				Requisitos enfocados en la migración			
N. Objetivo	Crítico	Prioritario	Secundario	N. Objetivo	Crítico	Prioritario	Secundario
SR-01				MR-01			
SR-02				MR-02			
SR-03				MR-03			
SR-04				MR-04			
SR-05				MR-05			
SR-06				MR-06			
SR-07				MR-07			
SR-08				MR-08			
SR-09							
SR-10							
SR-11							
SR-12							

Tabla 2 - Clasificación de los requisitos según su priorización.

1.3 Planificación

En este apartado se muestra la organización inicial planteada junto a las actividades que han de permitir para llevar a cabo el proyecto.

La planificación consta de las fases del proyecto, los puntos de control, los recursos necesarios, las dependencias, el cronograma y la evaluación de riesgos.

No se incluye el presupuesto, pues la migración es totalmente necesaria y no será un factor determinante en la decisión final, aunque si se apostará por aquellos servicios más económicos que cumplan con el objetivo y los requisitos presentados.

1.3.1 Fases del proyecto

Las fases del proyecto son de vital importancia, pues una mala ejecución de alguna de ellas puede llevarnos a un resultado no deseado. Por ejemplo, si no analizamos bien los objetivos y requisitos, una vez migrado el sistema, nos podemos encontrar con problemas difíciles de solventar, que en el peor de los casos, pueden llevar a una nueva reestructuración de los sistemas.

En la **Tabla 3** se muestran las fases que se han contemplado para este proyecto.

Fase	Descripción
Identificación	En esta fase se determina el problema que se quiere resolver y cuál es la nueva situación que se quiere alcanzar. Esta fase, que es previa a la inicio de la documentación y del proyecto, queda en parte reflejada en el resumen documental y la introducción.
Formulación	Con la formulación del proyecto se especifican los objetivos, requisitos y recursos necesarios para lograr la ejecución de este. Correspondería a gran parte de esta fase: los objetivos, los requisitos, la planificación y el análisis de los sistemas actuales.
Ejecución	Durante esta fase se pretende llevar a cabo lo previsto en las anteriores. Queda reflejada en los apartados: análisis sobre las soluciones existentes y migración de los sistemas actuales.
Seguimiento	Fase que se realiza durante todo el proyecto con la finalidad de ir supervisando la ejecución de este.
Finalización	Es la última fase y marca el fin del proyecto. De ella se extraen las conclusiones de todo el proceso y de los resultados obtenidos. Las conclusiones que quedan detalladas en el último punto del presente documento.

Tabla 3 - Fases del proyecto.

1.3.2 Puntos de control

Los puntos de control forman parte de la fase de seguimiento. Su cumplimiento favorece positivamente al conjunto global del proyecto.

A continuación, en la **Tabla 4**, se muestran los puntos de control junto a una pequeña descripción.

Fecha	Descripción
30/09/2013	Iniciación. Aceptación y definición de la propuesta. Definición de objetivos.
07/10/2013	Presentación y aceptación de la planificación.
23/10/2013	Analizado el sistema actual en detalle.
04/11/2013	Control sobre el desarrollo general del proyecto.
10/11/2013	Analizadas las soluciones existentes.
02/12/2013	Control sobre el desarrollo general del proyecto.
05/12/2013	Adaptación a los nuevos sistemas.
20/12/2013	Migración.
02/01/2013	Revisión de la documentación.
09/01/2013	Validación de la documentación a presentar tras finalizar la revisión.
10/01/2013	Presentación de la documentación del proyecto.
16/01/2013	Validación de la defensa del proyecto.
17/01/2013	Presentación del proyecto.

Tabla 4 - Puntos de control.

1.3.3 Recursos necesarios

Los recursos actuales para el proyecto se especifican en la **Tabla 5** junto a las fases en las que intervienen. Hay que tener en cuenta, que en este caso, todos los recursos humanos recaen sobre la misma persona, el desarrollador de IT, aunque lo más óptimo, tal como se muestra, sería que intervinieran distintos roles.

Grupo	Descripción	Fase
Recursos Humanos	Jefe de proyecto.	En todas las fases.
	Director de proyecto.	Seguimiento y finalización.
	Desarrollador IT.	En todas las fases.
Recursos Virtuales	Software ofimático.	Formulación y finalización.
	Software de planificación de proyectos.	Formulación.
	Software de diagramas.	Formulación.
Recursos físicos	Ordenador.	En todas las fases.
	Conexión a Internet.	En todas las fases.

Tabla 5 - Recursos necesarios.

1.3.4 Cronograma

A nivel general, el proyecto se desarrollará desde septiembre del 2013 hasta principios de enero del 2014 con una dedicación de 7 horas semanales. El total de horas dedicadas serían aproximadamente 211. Se han contabilizado festivos y fines de semana.

La Fecha de inicio es el 30 de septiembre del 2013 y la finalización está prevista para el 17 de enero del 2014. El cronograma queda reflejado en la **Tabla 7** y, en la **Tabla 6**, podemos encontrar la planificación.

Actividad	Días / Total	Inicio	Final
1 Iniciación	1.0 3.0	29-09-13	30-09-13
1.1 Presentación de la propuesta	1.0	30-09-13	30-09-13
1.2 Definición de los objetivos	1.0	30-09-13	30-09-13
2 Planificación	5.0 12.0	01-10-13	07-10-13
2.1 Definición de las fases del proyecto	2.0	01-10-13	02-10-13
2.2 Definición de los puntos de control	2.0	03-10-13	04-10-13
2.3 Análisis de los recursos necesarios	1.0	07-10-13	07-10-13
2.4 Realización del Cronograma y análisis de riesgos	1.0	07-10-13	07-10-13
2.5 Presentación y aceptación de la planificación	1.0	07-10-13	07-10-13
3 Análisis	43.0 87.0	08-10-13	05-12-13
3.1 Documentación sobre el sistema actual.	2.0	08-10-13	09-10-13
3.2 Análisis sobre los cambios en producción	3.0	10-10-13	14-10-13
3.3 Análisis sobre los cambios en desarrollo	7.0	15-10-13	23-10-13
3.4 Debate y documentación sobre los análisis previos	3.0	24-10-13	28-10-13
3.5 Análisis sobre empresas con la misma tipología de negocio	5.0	29-10-13	04-11-13
3.6 Control sobre el desarrollo general del proyecto	4.0	05-11-13	08-11-13
3.7 Búsqueda de soluciones existentes	4.0	11-11-13	14-11-13
3.8 Documentación e información sobre las soluciones existentes	2.0	15-11-13	18-11-13
3.9 Análisis de impacto de cada solución	12.0	19-11-13	04-12-13
3.10 Control sobre el desarrollo general del proyecto	1.0	02-12-13	02-12-13
3.11 Toma de decisión sobre una solución	1.0	05-12-13	05-12-13
4 Implantación	11.0 22.0	06-12-13	20-12-13
4.1 Debate sobre la hoja de ruta a seguir de cara a la migración	2.0	06-12-13	09-12-13
4.2 Documentación sobre la hoja de ruta	4.0	10-12-13	13-12-13
4.3 Migración	4.0	16-12-13	19-12-13
4.4 Análisis de los resultados	1.0	20-12-13	20-12-13
5 Documentación	70.0 140.0	07-10-13	10-01-14
5.1 Recopilación de la documentación	63.0	07-10-13	01-01-14
5.2 Validación de la documentación	6.0	02-01-14	09-01-14
5.3 Presentación de la documentación	1.0	10-01-14	10-01-14
6 Finalización	6.0 12.0	10-01-14	17-01-14
6.1 Inicio de la elaboración de la defensa	2.0	10-01-14	13-01-14
6.2 Validación de la defensa	3.0	14-01-14	16-01-14
6.3 Presentación del proyecto	1.0	17-01-14	17-01-14
	276.0		

Tabla 6 - Planificación.

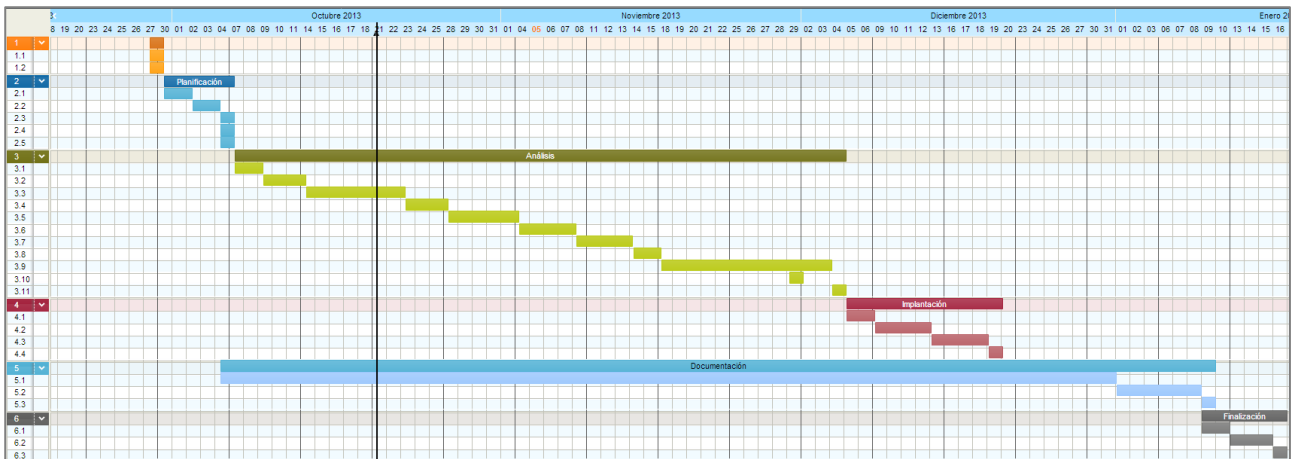


Tabla 7 - Cronograma.

Con tal de cumplir con todos los objetivos presentados con anterioridad, para las fases previas a la migración, se ha optado por seguir un modelo lineal: ninguna fase comienza sin que la anterior haya finalizado. Por lo que respecta a la propia migración, se usará un modelo ágil, donde se irán migrando partes del sistema actual a la vez que se comprueba el correcto funcionamiento de los pasos que se van realizando.

1.3.5 Evaluación de riesgos

Los riesgos que pueden afectar al desarrollo general del proyecto quedan descritos en la **Tabla 8**. En la **Tabla 9**, podemos encontrar estos en relación a su impacto junto a su posible solución.

Lista de riesgos			
ID.	Nombre	Afecta a...	Consecuencias
R01	Planificación temporal optimista	Finalización.	Retrasos. Provoca un aumento de los recursos.
R02	Falta de conocimiento	Formulación. Ejecución. Finalización.	Incumplimiento de objetivos
R03	Cambio de requisitos	Formulación. Ejecución. Finalización.	Retrasos.
R04	Equipo del proyecto reducido	Finalización	Retrasos. Incumplimiento de objetivos.
R05	Herramientas inadecuadas	Formulación. Finalización.	Menor calidad.
R06	Análisis incorrectos	Ejecución. Finalización.	Deficiencias, pérdidas económicas.
R07	Abandono	Finalización.	Pérdidas económicas.

Tabla 8 - Listado de riesgos.

Impacto y posibles soluciones*			
ID.	Probabilidad	Impacto	Posible solución
R01	Media	Crítico	Posponer algunos objetivos.
R02	Alta	Crítico	Revisar el plan de proyecto. Modificar la planificación.
R03	Baja	Crítico	Posponer algunos objetivos. Modificar la planificación.
R04	Media	Crítico	Posponer algunos objetivos. Modificar la planificación.
R05	Baja	Marginal	Introducción de alternativas.
R06	Alta	Catástrofe	Sin solución. El proyecto fracasa.
R07	Baja	Catástrofe	Sin solución. El proyecto fracasa.

Tabla 9 - Soluciones e impacto de los riesgos presentes.

* Rango de valores de más a menos relevancia:
 Probabilidad: Alta – Media – Baja.
 Impacto: Catástrofe – Crítico – Marginal.

2. Análisis sobre los sistemas actuales

A la hora de migrar un sistema es fundamental hacer un análisis profundo de este a todos los niveles, ya que permite determinar aquellos factores que pueden ocasionar un problema en el transcurso de esta.

En el caso que nos atañe se analiza, tanto nivel de seguridad como de infraestructura, el entorno de desarrollo y producción de la empresa RDmes Technologies S.L, ya que ambos se ven implicados en el proceso de migración y son estos dos los que presentan las limitaciones para la explotación de los aplicativos (plataforma y herramientas).

Con el análisis de infraestructura se fijan aquellos factores que se han de mantener tras la migración y, a su vez, los susceptibles a mejorar a nivel de seguridad física y de tipología de red. En el de seguridad, el objetivo es el mismo, pero a nivel de seguridad lógica.

Además, se aprovecha este análisis para definir un modelo de referencia de cara a la migración.

2.1 Análisis de infraestructura

El entorno de producción (ver **ilustración 01**) está situado en la Universitat Politècnica de Catalunya (UPC) y está formado por un solo servidor físico dedicado, el cual va directamente conectado a la red informática mundial (*World Wide Web*) mediante una conexión de fibra óptica de alta capacidad. Este servidor recibe el nombre de *Aire*. Su seguridad está gestionada básicamente a nivel lógico (*software*), aunque cumple con todos los requisitos a nivel físico: acceso con autorización, sistemas antiincendios, temperatura adecuada, etc.

En cuanto al entorno de desarrollo (ver **ilustración 02**), este un poco más complejo, está situado dentro de las oficinas de la empresa en el Institut Politècnic Campus Terrassa (IPCT), a unos 20 minutos en coche del entorno de producción. Este consta de dos servidores físicos: uno para dar servicio a la red interna, llamado *Sol*, y otro como plataforma de *testing*, llamado *Terra*, que es el servidor a migrar.

Terra en la actualidad no consta de ninguna seguridad física, se encuentra al alcance de cualquier persona y a simple vista. Además, hace la función de *bastion host* y está conectado a la red informática mundial (*World Wide Web*) mediante una conexión ADSL de 1Mb/s de capacidad (no aumentable). Este servidor es una pieza fundamental en el ciclo de desarrollo, pues genera los instaladores de la aplicación para su despliegue en *Aire*, lo que implica que cualquier cambio en producción ha de pasar por este previamente (ver **ilustración 04**).

Si juntamos todas las piezas vemos que el principal problema en el entorno de desarrollo es la velocidad de conexión de este con el exterior, ya que esta limita que los desarrolladores puedan trabajar desde una ubicación distinta a las oficinas y, a su vez, ralentiza el despliegue al ser de paso obligatorio para el mismo. El problema, en sí, se puede solventar de forma sencilla: se generan los instaladores en *Terra*, se copian por red local a un portátil, se da dicho portátil a la persona encargada del despliegue y este se dirige hacia un lugar físico donde disponga de una toma a Internet con mayor velocidad, desde donde podrá proceder a actualizar *Aire*. No obstante, no se contempla esta alternativa a largo plazo, ya que sería un protocolo lento e innecesario si se dispusiera de otra infraestructura más acorde a las necesidades.

Por otro lado, aunque no se le haya dado mucha importancia a *Sol*, hay que tener en cuenta que contiene todo el *software* para el control de versiones de la aplicación. Software que en la migración ha de formar parte del entorno de desarrollo en la nube.

Para concluir, como se aprecia a nivel de infraestructura, el entorno de producción cumple con los requisitos deseados (requisitos a mantener tras la migración): topología de red, conexión y seguridad adecuadas; mientras que el entorno de desarrollo presenta ciertos problemas e inconvenientes. Por lo tanto, cualquier migración de este entorno ha de suponer un avance significativo en la situación actual: mejora de la conexión, mejora de la seguridad física y redefinición de los roles que desempeña cada servidor.

2.2 Análisis de seguridad

El entorno de producción cuenta con los siguientes sistemas de seguridad a nivel lógico centralizados en *Aire*:

Copias de respaldo y recuperación de desastre (ver **ilustración 03**).

Se realiza diariamente una copia de respaldo de las bases de datos y de los ficheros dinámicos (XML's de los cálculos y logos). Semanalmente se realiza un duplicado de esta al entorno de desarrollo, concretamente, al servidor *Sol*.

Control del flujo de red.

Se usa el corta fuegos integrado en el sistema operativo (*IPtables* [2]) con política de negación por defecto.

Prevención de intrusos.

Se usa la aplicación *fail2ban* [3] con tal fin.

Configuración mínima de seguridad del servidor web.

El servidor web (*Apache HTTP* [4]), ha sido configurado para que no permita la lectura de directorios mediante la navegación web. También se usan complementos de terceros para fortalecer la seguridad.

Configuración mínima de seguridad respecto a la base de datos.

Aunque las claves de los usuarios se almacenan de forma cifrada y es necesario autenticarse para acceder a ella, mucha de la información sensible relacionada con estos no se cifra.

Conexión SSH (Secure Shell).

No se puede acceder al servidor por SSH (*Secure Shell*) mediante el usuario *root* y hay establecido un método de autenticación mediante clave pública-privada. No obstante, la conexión se hace por el puerto estándar y no se usan certificados para ella.

En cuanto al entorno de desarrollo, semejante al de producción, tiene:

En Terra y Sol:

Conexión SSH (Secure Shell).

No se puede acceder al servidor por SSH (*Secure Shell*) mediante el usuario *root* y hay establecido un método de autenticación mediante clave pública-privada. No obstante, la conexión se hace por el puerto estándar y no se usan certificados para ella.

En Terra:

Prevención de intrusos.

Se usa la aplicación *fail2ban* [3] con tal fin.

Configuración mínima de seguridad del servidor web.

El servidor web (*Apache HTTP* [4]) ha sido configurado para que no permita la lectura de directorios. Se usan complementos de terceros para fortalecer la seguridad.

Configuración mínima de seguridad respecto a la base de datos.

Aunque las claves de los usuarios se almacenan de forma cifrada y es necesario autenticarse para acceder a ella, mucha de la información sensible relacionada con ellos no se cifra.

Control del flujo de red.

Se usa el corta fuegos integrado en el sistema operativo con política de negación por defecto.

En Sol:

Copias de respaldo y recuperación de desastres (ver ilustración 03).

Se realiza diariamente una copia de respaldo tanto de los ficheros de los usuarios (trabajadores) como del directorio de control de versiones. Esta se guarda en un dispositivo externo.

Semanalmente se realiza un duplicado de la copia de respaldo del servidor de producción (*Aire*) mediante SCP (*Secure Copy*) al dispositivo externo.

Sistema de monitorización y avisos.

Contiene un *software* para supervisar intrusiones, verificar la integridad de los ficheros y detectar *rootkits* [5]. Ante una eventualidad genera alertas en tiempo real que se envían por correo a los responsables del sistema.

Por otra parte, *Sol* tiene integrado un DNS (sistema de nombres de dominio) para facilitar el trabajo en la red interna de la empresa así como un sistema de directorio activo.

Al analizar los datos expuestos podemos apreciar que:

Solo se han implementado las medidas de seguridad básicas, en especial en el servidor *Terra* y *Aire*. La integración de sistemas para la detección de intrusos y de monitorización así como fortalecer las medidas de seguridad actuales es un requisito a cumplir.

Muchos de los sistemas de seguridad están duplicados, aumentando el coste y el tiempo de mantenimiento. Por otro lado, aunque no se ha especificado, no están actualizados, por lo que pueden contener vulnerabilidades conocidas.

Se aplica un sistema ineficiente en referencia a las copias de respaldo y la recuperación de desastres debido a la tasa de transferencia del servidor *Sol/Terra* hacia el exterior y la naturaleza de la copia.

Este punto en concreto puede suponer un problema para los objetivos, pues la aplicación no está preparada para guardar los datos dinámicos en una ubicación distinta en la que está funcionando.

En resumen, la migración ha de pasar por mejorar el sistema actual de copias de respaldo y de recuperación de desastres sin afectar gravemente al funcionamiento de la aplicación, y a su vez, permitir aumentar los niveles de seguridad actual. Punto que se puede ver comprometido si el servicio al que migrar no permite una gestión totalmente libre de los recursos que ofrece.

Se ha de tener en cuenta que muchas de los sistemas de seguridad que se aplican pueden ir a cargo del proveedor del servicio escogido y formar parte de la infraestructura. Un ejemplo de ello podría ser un firewall o un sistema de monitorización externo.

La sección **2.3 Diagramas**, puede ayudar a la comprensión de la información presentada en este apartado, debido a que algunos de los diagramas contiene parte del análisis lógico de los sistemas.

2.3 Diagramas

En esta sección se plasma de forma gráfica los análisis realizados en las secciones anteriores con tal de facilitar la comprensión de estos. Se ha optado por ponerlos en una sección a parte para no entorpecer la lectura y conservar una estructura visual del documento amigable.

La **Ilustración 1** presenta un diagrama sobre el entorno de producción actual, donde queda plasmado principalmente la tipología de red y la descripción del servidor que lo compone. De la misma forma sucede en la **Ilustración 2**, pero para el entorno de producción.

Por último se presentan dos diagramas, uno para el ciclo de copias de respaldo, **Ilustración 3**, y otro para el ciclo de desarrollo y despliegue, **Ilustración 4**.

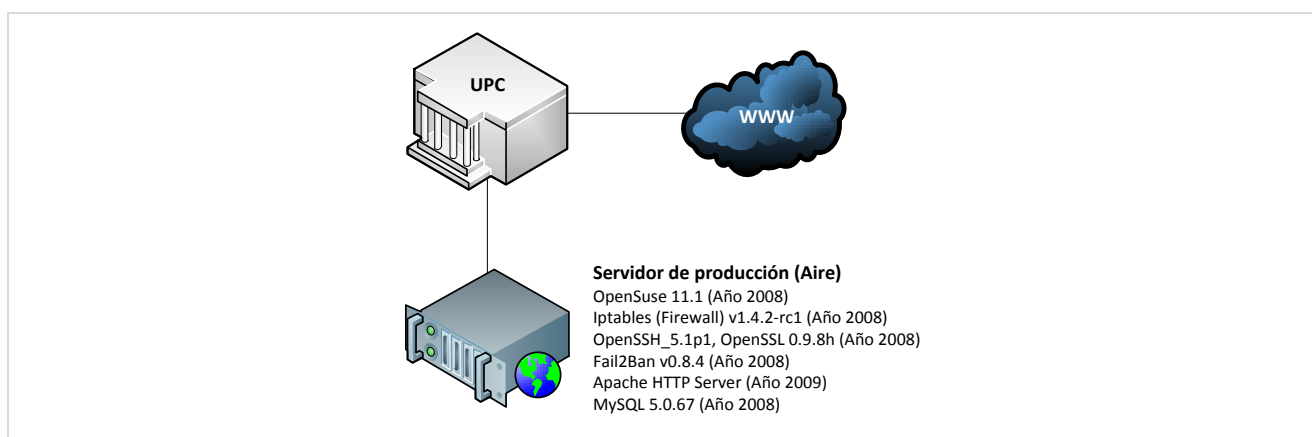


Ilustración 1 - Diagrama del entorno de producción actual

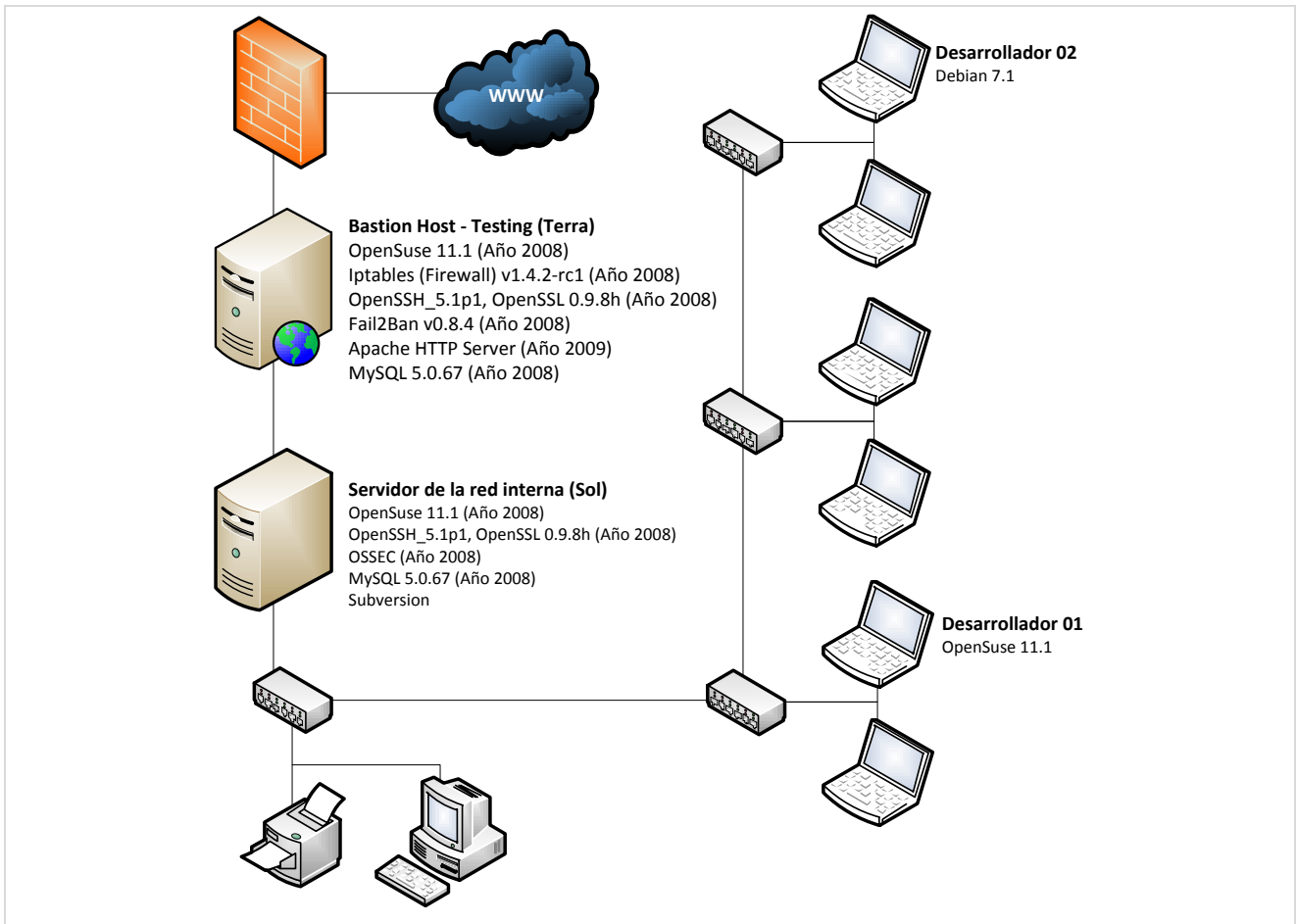


Ilustración 2 - Diagrama del entorno de desarrollo actual

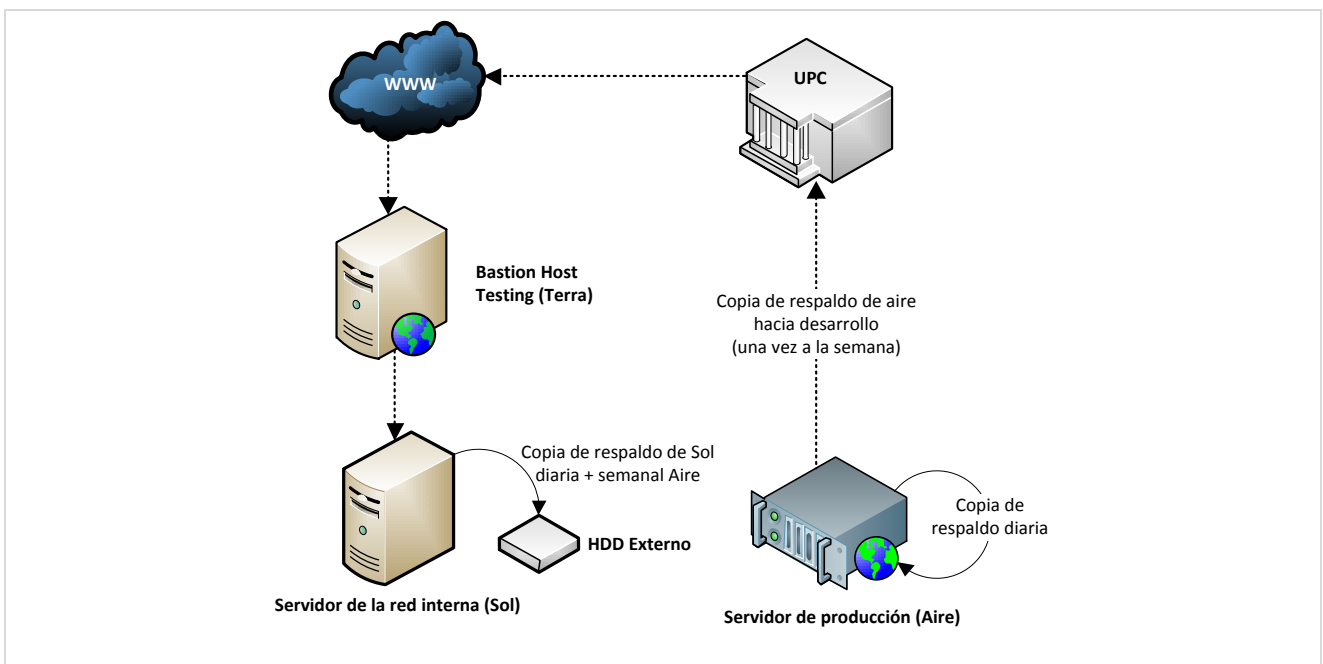


Ilustración 3 - Diagrama del ciclo de copias de respaldo actual

Con la finalidad de que el sistema de producción fuera flexible, lo que permitirá su escalabilidad, se situarían dos nuevos dispositivos junto a este (ver **ilustración 05**): uno destinado a almacenaje de copias de respaldo, de versiones de la aplicación y de ficheros de usuarios (clientes), y otro, con el objetivo de alojar la base de datos.

Mediante estas modificaciones se mejoraría el flujo de desarrollo y despliegue, ya que ambos entornos podrían alternarse el rol de *testing* y desarrollo. Si los dos están conectados a los dispositivos descritos y la aplicación ha sido actualizada y validada en *testing*, solo hace falta indicar en el punto de entrada a la red que redirija las peticiones a la aplicación a *testing*. De esta forma, el entorno de *testing* ahora pasaría a ser el entorno de producción y el de producción el de *testing*. De esta manera, nos permitiría “despliegues” a producción sin interrupción de los servicios (ver **ilustración 06**).

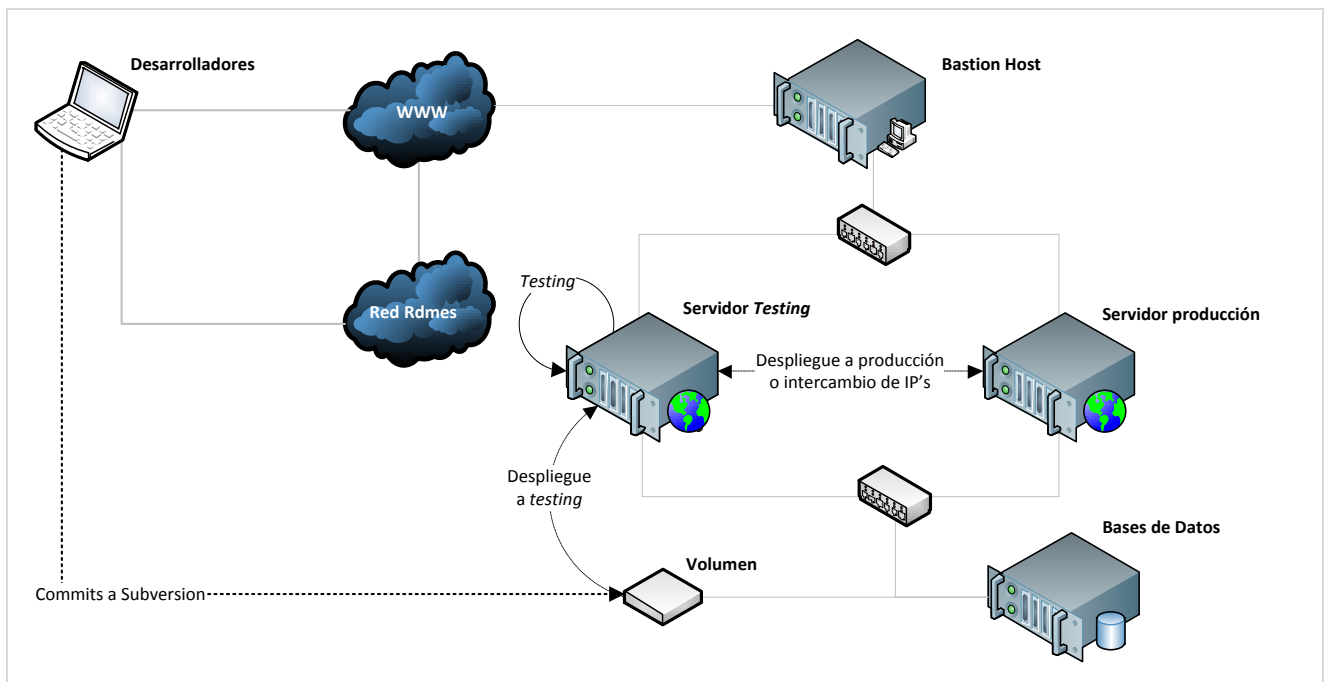


Ilustración 6 - Ciclo de desarrollo previsto

El flujo se complica si se contemplan múltiples servidores concurrentes en el entorno de producción. Si este fuera el caso, habría que crear tantos servidores de *testing* como servidores de producción hubiera, desplegar el *software* y redirigir todas las peticiones de producción hacia *testing*.

Montar un sistema a nivel lógico que permita estos flujos entre desarrollo y producción es realmente complejo y requiere de un tiempo de consolidación. Como actualmente no hay una repercusión importante si durante el despliegue a producción el servicio se interrumpe, se anotará este sistema lógico como un trabajo a futuro y se opta por definir un flujo más clásico, donde para actualizar el *software* se requiere de la interrupción del servicio.

Además de lo mencionado, con estos nuevos dispositivos también ganaríamos en rendimiento puesto que el servidor de producción ya no alberga la base de datos.

En cuanto al entorno de desarrollo en específico, *Terra* y *Sol* dejarían de formar parte de este, ya que sus roles cambian con el nuevo modelo. *Terra* ya no sería un servidor de *testing* ni *Sol* formaría parte de la política de copias de respaldo y recuperación de desastres. Después de la migración, ambos, solo formarían parte del entorno de oficina.

Evidentemente todos los cambios afectan directamente al flujo de copias de respaldo (ver **ilustración 07**). Ahora es suficiente con crear copias de la base de datos y de los ficheros críticos del servidor de producción y almacenarlas en el dispositivo destinado a tal fin sin necesidad de implicar a *Sol* o *Terra*. Como este dispositivo, tal como se ha dicho previamente, también contendría el *software* para la gestión de versiones, este tendría que gestionar automáticamente los *backups* de las versiones de la plataforma y las herramientas.

Es importante tener en cuenta que cuando nos referimos a un servidor no implica que este deba ser una máquina física dedicada, sino que también podría ser un máquina virtual (instancia) o compartida.

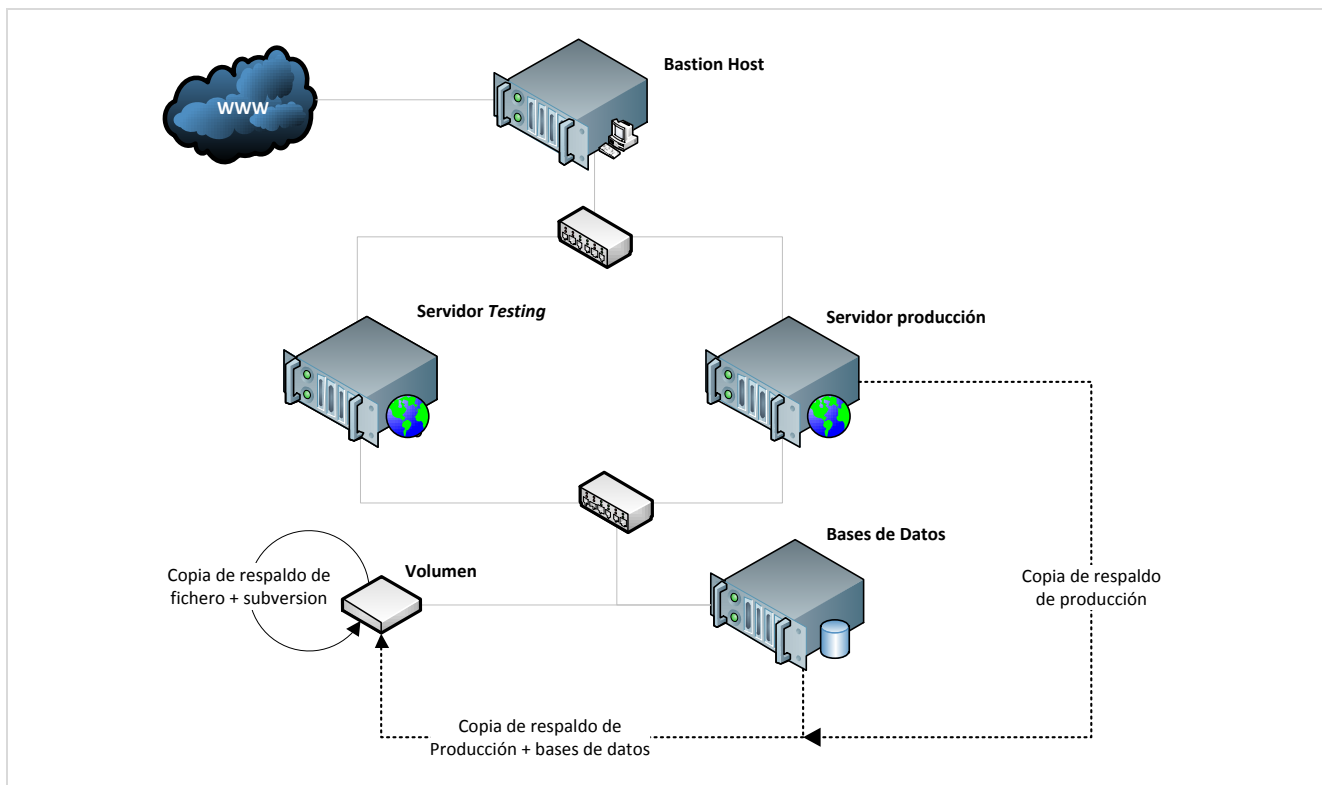


Ilustración 7 - Ciclo de copias de respaldo previsto

A continuación se hace un breve resumen de cómo se cumplen algunos de los objetivos con la implantación de este modelo:

SPO-03. Política de copias de respaldo y recuperación ante desastres.

Con el nuevo esquema las copias de respaldo se guardarían en un lugar distinto a los servidores, por lo que sería un sistema totalmente independiente. Además, como el dispositivo destinado a guardar dichas copias se encontraría en la misma red que ambos entornos, los tiempos de recuperación de desastres se verían minimizados.

SPO-05. Nueva política de actualizaciones y SDO-03. Diversidad de las distintas máquinas.

Migrar hacia un modelo de esta índole forzaría a revisar y reasignar los roles de cada servidor, propiciando un momento idóneo para actualizar los entornos.

SDO-01. Poca credibilidad del entorno de testing

Al incorporar dos máquinas o instancias nuevas estas pueden ser idénticas, haciendo que el entorno de testing sea lo más fiel posible al de producción.

SDO-02. Poca movilidad de los desarrolladores

Al situar ambos entornos fuera de la red de oficina quedan solventadas las limitaciones de conectividad que no permitían a los desarrolladores trabajar desde cualquier ubicación distinta a esta.

También cumple aquellos requisitos ligados a los objetivos mencionados:

MR-02. Mejora en la política de copias de respaldo y recuperación de desastres.

En relación al objetivo SPO-03.

MR-03. Mejora de la movilidad.

En relación al objetivo SDO-02.

MR-04. Mejora en la fidelidad testing – producción.

En relación al objetivo SDO-01.

MR-07. Agilidad.

Permite el paso de testing a producción de forma más eficiente que la actual.

El resto de objetivos y requisitos dependerán directamente del servicio y proveedor escogido.

3. Estudio de las soluciones existentes

En este capítulo se analizan los proveedores actuales de servicios en la nube que podrían llegar a albergar el modelo del sistema presentado con anterioridad. Para ello se ha escogido, de entre todos, aquellos que se han forjado poco a poco una reputación, como se puede constatar a través de diferentes medios, como por ejemplo, comparativas, opiniones, conferencias, volumen de usuarios, etc. [6] [7] [8] [9] [10]

No obstante, antes de seleccionar los proveedores, es importante tener en cuenta como se clasifica la computación en la nube en función del modelo de negocio que sigue el proveedor, o dicho de otra forma, en función del servicio que nos ofrece. De esta forma, según este concepto, tenemos tres grupos generales [11] [12]: *software* como servicio (SaaS), plataforma como servicio (PaaS) e infraestructura como servicio (IaaS). Aunque existen de otros más concretos como almacenaje como servicio (StaaS) [13] [14].

Software como servicio [15] [16] es un modelo de distribución de *software* que proporciona a los clientes el acceso al mismo a través de la red. El *software* distribuido mediante este modelo se ejecuta en el sistema del proveedor, de manera que los usuarios de este se ven liberados de su mantenimiento así como de las operaciones técnicas y de soporte. Además, el modelo de licencias tradicional es sustituido por un modelo de pago por uso.

En el caso del modelo de plataforma [17] como servicio se comercializa un conjunto de herramientas para el desarrollo de aplicaciones accesibles desde Internet y alojadas en la infraestructura del proveedor. Brinda la posibilidad de desarrollar *software* sin la necesidad de instalar las herramientas en local, permitiendo después el despliegue de este a producción sin tener conocimientos administrativos especializados para ello.

Por último, el modelo de infraestructura como servicio [18] [19] está enfocado a proporcionar recursos a nivel de hardware a los clientes, generalmente a través de virtualización de sistemas. De esta manera un usuario puede disponer de recursos de almacenaje o de servidores en la nube, entre otros.

Tras analizar las descripciones presentadas para cada tipo de servicio, los únicos proveedores que podrían llegar a cubrir nuestras necesidades son aquellos que siguen un modelo PaaS o IaaS. Teniendo en cuenta este factor, los proveedores seleccionados para el análisis han sido: Amazon AWS, DigitalOcean, DinaHosting, Heroku, Joyent™, Linode, OpenShift, RackSpace®, VMware vCloud® y Windows Azure.

Para cada proveedor se realiza una breve introducción junto a un análisis del servicio que presta frente a los objetivos y requisitos descritos al inicio del documento, marcando mediante un ✓ aquellos objetivos y requisitos que pueden satisfacer, con una ✗ aquellos que no y, con un ?, aquellos que no se han podido comprobar.

Los proveedores escogidos se analizan en orden alfabético. Además, parte de la información presentada, de proveedor a proveedor, puede resultar redundante provocando una lectura “pesada” y repetitiva, debido a que algunos de ellos usan las mismas tecnologías. Este factor, hace que cubran, en parte, los requisitos y objetivos de la misma forma.

Aclarados estos conceptos se presenta el análisis de cada proveedor.

3.1 Amazon AWS



Amazon AWS [20] [21] es en la actualidad un completo conjunto de servicios de informática en la nube propiedad de Amazon.com,inc, empresa fundada en el 1994 con sede en Seattle, Estado de Washington. Estos fueron lanzados en el 2006 con la finalidad de proporcionar una plataforma de infraestructura escalable de alta fiabilidad y de bajo coste, que permitiera replicar de forma casi exacta la estructura de un centro de datos tradicional. Los servicios más destacados son los de informática, bases de datos y almacenamiento, aunque contiene otros igual de interesantes enfocados al desarrollo y la gestión de aplicaciones.

Usa el modelo de negocio de infraestructura como servicio (IaaS) y ofrece una prueba gratuita limitada de un año. Se analizan, a continuación, los objetivos y requisitos respecto a este servicio:

SP0-01. Poca flexibilidad. ✓

Los servicios de Amazon AWS permiten escalar los sistemas de forma sencilla [22]. Para ello se ha de usar EC2 [23] junto a CloudWatch [24]. EC2 [23] nos permitiría virtualizar máquinas de diferente índole (instancias) y CloudWatch [24] monitorizarlas, permitiéndonos a la vez automatizar el cierre o levanta-

tamiento de estas. Mediante ELB (*Elastic Load Balancing*) [25], otro servicio que ofrece, se balancearía la carga entre las distintas instancias.

Las instancias se pueden levantar sincronizadas unas con otras. Es una posibilidad aunque un poco más compleja.

Las instancias creadas que no se usan no se facturan, lo que hace realmente que el servicio sea escalable ya que no supone un gasto adicional para la empresa. Si se cobrasen vendría a ser comparable a la compra de un servidor que no se usase. Además, se pueden crear instancias con CloudWatch [24] a través de AMI's (*Amazon Machine Image*) [26], por lo que realmente tampoco haría falta tenerlas creadas.

SPO-02. Alto coste de escalabilidad y mantenimiento. ✓

Al poder generar máquinas virtuales (instancias) de forma dinámica se reduce drásticamente el coste en infraestructura, ya que no se ha de comprar un servidor para cubrir la demanda de forma permanente o en picos de uso. También hay que tener en cuenta que se paga por uso, de forma que los costes son dinámicos y ajustados.

SPO-03. Política de copias de respaldo y recuperación de desastres. ✓

Mediante los servicios S3 [27] que nos ofrece Amazon AWS podemos mejorar nuestra política de copias de respaldo y recuperación de desastres de forma considerable. S3 [27] es un sistema de almacenaje en la nube que se puede acoplar a los demás servicios. En él podríamos almacenar las AMI's (*Amazon Machine Image*) [26] de las instancias a modo de copia de respaldo. Además, al ser esta una imagen de instancia y estar bajo la misma red que los demás sistemas, la recuperación de desastres sería mucho más rápida, ya que bastaría con levantar una nueva instancia y cargar la AMI (*Amazon Machine Image*) [26]. Proceso automatizable mediante CloudWatch [24].

S3 [27] también se podría usar para guardar los datos dinámicos generados por la aplicación y asociados a los clientes, así como el directorio de control de versiones.

SPO-04. Tiempos de inactividad. ✓

Los tiempos de inactividad por mantenimiento serían eliminados puesto que ofrecen un servicio continuo sin interrupciones. En cuanto a los tiempos de despliegue se podrían ver minimizados o totalmente eliminados según la infraestructura lógica que se implemente.

SPO-05. Nula política de actualizaciones. ✓

Amazon trabaja con las últimas versiones de Software [28] presente en el mercado así que una migración hacia estos servicios comportará forzosamente una actualización de los sistemas.

SDO-01. Poca credibilidad del entorno de testing. ✓

Lo bueno del sistema de instancias es que estas se pueden destruir y crear fácilmente desde una AMI (*Amazon Machine Image*) [26] de otra máquina, por lo tanto, podemos crear una instancia que sea exactamente igual, tanto en prestaciones como en *software*, a la de producción. Esto nos garantiza que las pruebas en *testing* son extrapolables a producción. Pero aún hay más, se podrían crear múltiples instancias que simularan usuarios y hacer desde estas peticiones a la instancia de *testing*, de tal forma que podríamos identificar cómo se comporta esta última ante un volumen de usuarios determinado o incluso, testear el sistema de escalado y notificaciones.

SDO-02. Poca movilidad de los desarrolladores. ✓

Al ser un sistema en la nube este punto queda totalmente resuelto.

SDO-03. Diversidad de las distintas máquinas. ✓

Al migrar y actualizar el sistema de producción, se hace necesaria la actualización de los sistemas de desarrollo.

SR-01. El acceso al servicio ha de ser seguro. ✓

El acceso a las aplicaciones de gestión de los servicios admiten doble autenticación [29] y se realizan por canales de comunicación seguros (SSL - *Secure Sockets Layer*). Por otro lado, todos los accesos para administrar las instancias se realizan por SSH (*Secure Shell*) haciendo uso de un certificado que se genera en el momento de crear esta.

SR-02. El servicio contratado debe tener asociado un sistema de soporte. ✓

Amazon AWS cuenta tanto con guías *online* [30] como con teléfonos de atención al cliente y correos electrónicos de soporte [31]. Tras consultar algunas de las guías he de mencionar que están realmente bien elaboradas. Además, cuenta con una base de datos de *scripts* [32] para implementar de forma sencilla ciertas funcionalidades, como podrían ser la generación de copias de respaldo en S3 [27] [33].

SR-03. El proveedor ha de ser fiable. ✓

Tras buscar información al respecto sobre Amazon AWS solo he leído alabanzas, lo que hace pensar que es un servicio de los más fiables. Investigando un poco sobre este punto vemos datos que corroboran esta premisa [34] [35]: en el Summit del 2013 en Barcelona, Amazon comentó que su servicio S3 [27] tiene una fiabilidad de 99.999999% [36] y que la mayoría de seguridad es mediante hardware y organizada en capas [37]. A su vez, cuenta con el certificado ISO 27001 [1] y ha sido validado con éxito como proveedor de servicio de Nivel 1 [38] conforme al Estándar de seguridad de datos (DSS - *Data Security Standard*) del sector de tarjetas de pago (PCI - *Payment Card Industry*). Cada año se someten a auditorías SOC1 [39], en las cuales obtiene una evaluación satisfactoria en el nivel Moderado correspondiente al sistema del gobierno federal (U.S.A) y en el nivel 2 DIACAP [40] para sistemas del departamento de defensa estadounidense.

SR-04. Privacidad en los datos hospedados. ✓

Según la política de privacidad [41] de Amazon AWS los datos de clientes no son compartidos ni vendidos a terceros. En cuanto a lo almacenado en los servicios, según se explicó en el Summit del 2013 de Barcelona, ningún miembro del equipo tiene acceso. Existe un caso de excepción, y es que bajo ley, por petición de un juez, podrían ofrecer acceso al contenido almacenado.

SR-05. Borrado permanente tras nuestra elección. ?

Es un punto complejo de corroborar. No se tiene constancia de cuál es su política ante esta situación.

SR-06. Libertad de gestión. ✓

Tenemos total libertad tanto a la hora de gestionar la infraestructura como las instancias creadas. A nivel de infraestructura debido a que podemos hacer uso de aquellos servicios que deseamos en el momento deseado llegando a poder montar una red privada, y de cara a las instancias, debido a que podemos instalar el *software* que necesitemos en estas sin restricción.

SR-07. Transparencia. ✓

Amazon AWS es bastante transparente en cuanto a lo que seguridad se refiere. Tanto en la web como en los Summit que organiza no tiene ningún pudor en demostrar que tecnologías usa y que infraestructuras implementa para ofrecer sus servicios [35] [37].

SR-08. Incorporación de sistemas de alerta y análisis. ✓

Cuenta con diversos servicios orientados a monitorización, análisis y alertas, como son CloudWatch [24] o Trusted Advisor [42].

SR-09. Próximo y motivador. ✓

Contiene una sección dedicada a la formación en su portal web junto a casos de éxito. Además organiza conferencias donde, a parte de hablar sobre sus servicios, invita a empresas para que comenten su experiencia durante y tras la migración, así como sus motivaciones para realizarla. También, en estas, permite probar todos los recursos que ofrece y preguntar sobre ellos de forma directa (presencial) a personal cualificado de Amazon AWS.

SR-10. Disponibilidad y replicación de los datos. ✓

Permite obtener en todo momento la información almacenada en la nube, tanto por los usuarios como por los empleados, al no existir un sistema de mantenimiento.

SR-11. Cumplimiento de la legalidad. ✓

Amazon AWS cumple la legalidad vigente en función de la zona donde se esté usando su servicio, por ejemplo, si estas almacenando datos en S3 [27] [43] en la zona de Estados Unidos este cumpliría la leyes de protección de datos de Estados Unidos, de lo contrario, si está en otro zona, por ejemplo, la zona de Irlanda, se cumplen las leyes de protección de datos de la Unión Europea. A la hora de usar los recursos que nos ofrecen hemos de elegir en que zona deseamos usarlos.

SR-12. Cancelación sin repercusiones. ✓

Al ser un servicio bajo demanda y que solo se paga por el uso, el simple hecho de dejar de usarlo ya te exime de cualquier responsabilidad. De todas formas, los servicios de Amazon AWS también se pueden contratar mediante suscripción, aunque tampoco existe ningún tipo de penalización por cancelación.

MR-01. Ventaja económica. ✓

Al ser un servicio de pago por uso el coste es más ajustado a que si la empresa ha de comprar un servidor en previsión al uso que hará de él o, si esta tiene un servidor del cual no usa todos sus recursos. Generalmente, porque al comprar un servidor bajo esas premisas se suelen cometer errores de sobredimensionado, que causan pérdidas económicas, o errores de infravaloración, que causan que a corto plaza haya que comprar uno de nuevo.

A lo ya mencionado, hay que sumar que el coste de una infraestructura es caro, en especial en cuanto a seguridad, monitorización y energía. Si tenemos un sistema sobredimensionado estos costes se disparan.

MR-02. Mejora en la política de copias de respaldo y recuperación de desastres. ✓

Tal como se ha comentado en el punto **SPO-03** este proveedor nos permite mejorar la política actual.

MR-03. Mejora de la movilidad. ✓

Al ser un sistema en la nube solventaría los problemas de infraestructura que no permiten a los desarrolladores trabajar desde otra ubicación que no sea la oficina.

MR-04. Mejora en la fidelidad testing – producción. ✓

Por el mismo factor que el **SDO-01**.

MR-05. Facilidad en la adaptación. ✓

Las interfaces de los servicios son intuitivas y sencillas de usar y el esquema planteado parece fácil de plasmar mediante lo que ofrecen.

MR-06. Escalable. ✓

El servicio EC2 [23] es totalmente escalable. Además se puede configurar para que escale de forma automática [22].

MR-07. Agilidad. ✓

A través de la creación de AMI's (*Amazon Machine Image*) [26] del sistema de *testing* podemos pasar a producción de forma rápida y sencilla. O simplemente, una vez validada la aplicación en *testing* pasar la instancia a producción.

MR-08. Compatibilidad. ✓

Permite una gran variedad [28] de sistemas operativos y *software* en las instancias de EC2 [23]. Entre ellos se encuentra el sistema operativo que necesitamos, que sería, Ubuntu 12.04 o Debian 7. Además, el servicio RDS [44] es compatible con bases de datos MySQL.

Tras el análisis vemos que cumple con todos los objetivos y requisitos, así que es una opción totalmente viable para la migración

3.2 DigitalOcean



DigitalOcean [45] [46] es una compañía de computación en la nube fundada en el 2011 y con sede en el estado de Nueva York. Se ha hecho un nombre con extrema rapidez en el mundo del web hosting. Su estrategia va enfocada a la simplicidad y velocidad de sus servicios junto a precios reducidos. No obstante, tras un simple vistazo vemos que sería complicado adaptar nuestro diseño a sus servicios todo y usar un modelo de negocio IaaS (infraestructura como servicio). A continuación, con el análisis de objetivos y requisitos, vemos porque:

SPO-01. Poca flexibilidad. ✗

DigitalOcean no está enfocada a la escalabilidad en infraestructura, aunque con un poco de ingenio se puede lograr. Bastaría con crear un VPS (*Virtual Private Server*) en el servicio que controlara el servidor de producción, y que ante un determinado factor, crease un nuevo servidor a partir de una copia de respaldo de este. Posteriormente haría falta balancear la carga. [47]

Para el proceso descrito haría falta elaborar *scripts* que interactuaran con la API (*Application Programming Interface*) de DigitalOcean [48]. A priori no es sencillo, por lo que no es una opción cuando buscamos sencillez.

SPO-02. Alto coste de escalabilidad y mantenimiento. ✓

El servicio prestado se paga por horas a un precio realmente económico, e igual que en los demás servicios, permite la creación de servidores en la nube cuando los necesitamos, aunque en este caso, de forma más compleja.

SPO-03. Política de copias de respaldo y recuperación de desastres. ✓

DigitalOcean cuenta con un servicio de auto *backups* [49]. Este lo que hace es crear una imagen de las máquinas en funcionamiento cada cierto tiempo. Ante un desastre, sería tan simple como crear una nueva máquina y cargar la última imagen creada.

SPO-04. Tiempos de inactividad. ✓

Los tiempos de inactividad por mantenimiento serían eliminados, puesto que ofrecen un servicio continuo sin interrupciones [50]. En cuanto a los tiempos de despliegue se podrían ver minimizados o totalmente eliminados según la infraestructura lógica que se implemente.

SPO-05. Nula política de actualizaciones. ✓

DigitalOcean trabaja con las últimas versiones de *software* presente en el mercado así que una migración hacia estos servicios comportará forzosamente una actualización de los sistemas. Se pueden ver los sistemas instalables en sus máquinas a través de la documentación [51].

SDO-01. Poca credibilidad del entorno de testing. ✓

Se podrían crear un VPS (*Virtual Private Server*) igual que el de producción e instalar todo el *software* para hacer *testing*. De hecho, para el mismo fin, se podría clonar el servidor de producción a través de una imagen de este con un mínimo de esfuerzo.

De la misma forma que en los servicios anteriormente presentados, se podrían crear diversos VPS (*Virtual Private Server*) que simularan usuarios para ver cómo se comportaría la aplicación ante un gran volumen de peticiones.

SDO-02. Poca movilidad de los desarrolladores. ✓

Al ser un sistema en la nube este punto queda totalmente resuelto.

SDO-03. Diversidad de las distintas máquinas. ✓

Al migrar y actualizar el sistema de producción se hace necesaria la actualización de los sistemas de desarrollo.

SR-01. El acceso al servicio ha de ser seguro. ✓

El acceso a las aplicaciones de gestión de los servicios admiten doble autenticación [52] y se realizan por canales de comunicación seguros (SSL - *Secure Sockets Layer*). Por otro lado, las instancias se pueden administrar vía interfaz web o por SSH (*Secure Shell*) [53], ya sea mediante usuario-clave o certificados.

SR-02. El servicio contratado debe tener asociado un sistema de soporte. ✓

El servicio de soporte técnico es el adecuado. Ofrece guías bien clasificadas por temáticas [51] y de toda índole, blogs con artículos sobre el servicio [54] e incluso un chat [55] para consultar dudas.

SR-03. El proveedor ha de ser fiable. ?

En este caso no hay información suficiente como para determinar la fiabilidad de los servicios.

SR-04. Privacidad en los datos hospedados. ✓

Según los términos de servicios [56] de DigitalOcean no se revelan los datos de ningún servicio a terceros, a excepción de a los órganos judiciales de cada país si estos los requirieran.

SR-05. Borrado permanente tras nuestra elección. ?

Es un punto complejo de corroborar. No se tiene constancia de cuál es su política ante esta situación.

SR-06. Libertad de gestión. ✓

Tenemos una total libertad a la hora de gestionar los servidores. A nivel de infraestructura es un poco más complejo, pero con el uso de *scripts* se puede llegar a lograr.

SR-07. Transparencia. ✓

A través de la sección *Features* [57] y el apartado *Security* [58] se puede llegar hacer una idea de las tecnologías que implementan y la seguridad del servicio.

SR-08. Incorporación de sistemas de alerta y análisis. ✗

Este servicio no está presente en DigitalOcean. Para cubrir esta necesidad tendríamos que usar un servidor VPS (*Virtual Private Server*) y configurarlo por nuestra cuenta [59].

SR-09. Próximo y motivador. ✓

Cuenta con un amplio calendario de eventos [60] y un chat [55] para consultas. Características que lo hace próximo al cliente.

Por otro lado cuenta con un programa de referidos [61] para obtener ventajas económicas, lo que incentiva a usar sus servicios.

SR-10. Disponibilidad y replicación de los datos. ✓

Permite obtener en todo momento la información almacenada en la nube, tanto por los usuarios como por los empleados, al no existir un sistema de mantenimiento.

SR-11. Cumplimiento de la legalidad. ✓

DigitalOcean cumple especialmente la legalidad vigente en Estados Unidos, aunque sus términos y servicios [56] especifican que cumple con esta en todos aquellos lugares donde está situado su hardware.

SR-12. Cancelación sin repercusiones. ✓

Al ser un servicio bajo demanda y que solo se paga por el uso, el simple hecho de dejar de usarlo ya te exime de cualquier responsabilidad.

MR-01. Ventaja económica. ✓

Al ser un servicio de pago por uso el coste es más ajustado a que si la empresa ha de comprar un servidor en previsión al uso que hará de él. No obstante, implementar el diseño presentado en este servicio quizás no presente un gran margen de ahorro al tener que simular ciertos requisitos mediante el uso de nuevos VPS (*Virtual Private Server*).

MR-02. Mejora en la política de copias de respaldo y recuperación de desastres. ✓

Tal como se ha comentado en el punto **SPO-03** este proveedor nos permite mejorar la política actual.

MR-03. Mejora de la movilidad. ✓

Al ser un sistema en la nube solventaría los problemas de infraestructura que no permiten a los desarrolladores trabajar desde otra ubicación que no sea la oficina.

MR-04. Mejora en la fidelidad testing – producción. ✓

Por el mismo factor que el **SDO-01**.

MR-05. Facilidad en la adaptación. ✗

El servicio en general es sencillo de usar pero no aporta las facilidades suficientes para la migración ya, que en este caso, es complejo implementar un sistema que autoescale.

MR-06. Escalable. ✗

El servicio no es autoescalable de forma sencilla, hay que implementar *scripts* para tal fin y crear servidores que gestionen el escalado [47].

MR-07. Agilidad. ✓

A través de la creación de imágenes y VPS (*Virtual Private Server*) podemos pasar de *testing* a producción de forma rápida y sencilla. De la misma manera a la inversa.

MR-08. Compatibilidad. ✗

Permite una gran variedad de sistemas operativos, no obstante, no cuenta con servicios enfocados a las bases de datos o sistemas de análisis, lo que nos fuerza a usar servicios de terceros o ingeniárnosla con el servicio prestado para lograr los resultados deseados.

Como no queremos tener diversidad de proveedor y queremos un servicio centralizado que cubra todas las necesidades, este no cumple con el requisito de compatibilidad.

Realmente DigitalOcean no es un servicio enfocado a la elasticidad en la nube, como puede ser Amazon AWS, sino más bien enfocado al aprovisionamiento de VPS (*Virtual Private Server*) en ella, y es en este sector donde es realmente competitivo.

3.3 DinaHosting



DinaHosting [62] [63] es una empresa dedicada a proporcionar servicios en la nube de distinta índole, fundada en el 2001 y con sede en Santiago de Compostela, España.

Destaca por su gran atención al cliente y la profesionalidad de sus trabajadores. Cuenta con una demostración interactiva de sus servicios. El modelo de negocio es IaaS (infraestructura como servicio).

En nuestro caso, vemos como se adapta su servicio Real Cloud a nuestros objetivos y requisitos:

SPO-01. Poca flexibilidad. ✗

Tras probar el servicio se ha observado que mediante el panel de control incluido en el administrador de servidores solo se puede programar el escalado del servidor en función del día y la hora. Lamentablemente, para un escalado de infraestructura o basado en los recursos del sistema hay que hacer uso de la API (*Application Programming Interface*) que ofrecen [64]. Como hay otros servicios que ofrecen una completa programación del escalado mediante la interfaz web, para nosotros DinaHosting no cumple con este objetivo.

SPO-02. Alto coste de escalabilidad y mantenimiento. ✓

Frente al sistema tradicional supone una reducción de costes a nivel general de la misma forma que los demás servicios presentados hasta el momento.

SPO-03. Política de copias de respaldo y recuperación de desastres. ✓

El servicio cuenta con un sistema de copias de respaldo automatizable [65] desde la interfaz web de administración. Las copias de respaldo son imágenes completas de máquina, por lo que se pueden usar para crear de nuevas o recuperar las existentes.

SPO-04. Tiempos de inactividad. ✓

Los tiempos de inactividad se ven prácticamente eliminados. DinaHosting asegura que sus máquinas van a funcionar el 99% del tiempo cada mes. En el caso de que no sea así, nos garantiza un porcentaje de devolución de dinero en función del tiempo de inactividad. [66]

SPO-05. Nula política de actualizaciones. ✓

El servicio funciona bajo las últimas versiones de *software* presentes en el mercado, así que el paso hacia este servicio forzaría la actualización del sistema de producción y desarrollo. Este dato se puede corroborar mediante la interfaz de administración de los servidores.

SDO-01. Poca credibilidad del entorno de testing. ✓

Permite la clonación de máquinas tanto a nivel de *software* como de hardware, por lo que se puede crear un sistema de *testing* idéntico al de desarrollo. Este dato se puede corroborar mediante la interfaz de administración de los servidores.

SDO-02. Poca movilidad de los desarrolladores. ✓

Al ser un sistema en la nube este punto queda totalmente resuelto.

SDO-03. Diversidad de las distintas máquinas. ✓

Al migrar y actualizar el sistema de producción se hace necesaria la actualización de los sistemas de desarrollo.

SR-01. El acceso al servicio ha de ser seguro. ✓

El acceso a las aplicaciones de gestión de los servicios admiten se realiza por canales de comunicación seguros (SSL - *Secure Sockets Layer*). Por otro lado, todos los accesos para administrar las instancias se realizan por SSH (*Secure Shell*) [67] desde el panel de control de usuario mediante el uso de un certificado.

SR-02. El servicio contratado debe tener asociado un sistema de soporte. ✓

Si en algo destaca este servicio es en su excelente sistema de soporte. Este es totalmente gratuito y cuenta con [68]: chats, Skype, teléfono y correo electrónico. También cuenta con una sección llamada "central de soporte" [69] donde están las guías para el uso de todos sus servicios.

SR-03. El proveedor ha de ser fiable. ?

Al no conocer de primera mano cuanto es de fiable y al no ser conscientes de su trayectoria profesional no podemos determinar su fiabilidad. No obstante, la gran mayoría de opiniones que podemos encontrar sobre este proveedor en internet son positivas.

SR-04. Privacidad en los datos hospedados. ✓

Según su política de protección de datos [70] cumplen estrictamente con la Ley 15/1999 de Protección de Datos de Carácter Personal: LOPD y lo hospedado en sus servidores es propiedad del contratante.

SR-06. Libertad de gestión. ✓

Tenemos total libertad tanto a la hora de gestionar la infraestructura como las instancias creadas, en especial si el servicio es no administrado.

SR-07. Transparencia. ✓

En la página web del proveedor se puede encontrar toda la información sobre su infraestructura física y lógica [71] [72] [73], así como todas las novedades que van incorporando e incidencias que van sufriendo. Esta última información es visible en la página web a través del monitorizador de red.

SR-08. Incorporación de sistemas de alerta y análisis. ✓

En los servicios administrados consta con un sistema de monitorización y alerta [74].

SR-09. Próximo y motivador. ✓

Es una empresa de ámbito nacional con atención al cliente en español y catalán, lo que lo hace realmente cercana. Además, su predisposición a ayudar, visible a lo largo de la web del proveedor, incentiva el uso de sus servicios.

SR-10. Disponibilidad y replicación de los datos. ✓

Permite obtener en todo momento la información almacenada en la nube, tanto por los usuarios como por los empleados.

SR-11. Cumplimiento de la legalidad. ✓

Cumple con la legalidad vigente a nivel nacional en cuanto a leyes de sociedades y protección de datos. En principio, con el cumplimiento de estas leyes, puede dar servicio al extranjero sin incurrir en ningún delito.

SR-12. Cancelación sin repercusiones. ✓

Al ser un servicio bajo demanda y que solo se paga por el uso, el simple hecho de dejar de usarlo ya te exime de cualquier responsabilidad.

MR-01. Ventaja económica. ✓

Al ser un servicio de pago por uso el coste es más ajustado a que si la empresa ha de comprar un servidor en previsión al uso que hará de él. Esta situación, se da especialmente en sistemas que pueden escalar con facilidad o presentan picos de usuarios en determinados momentos del día.

MR-02. Mejora en la política de copias de respaldo y recuperación de desastres. ✓

Tal como se ha comentado en el punto **SPO-03** este proveedor nos permite mejorar la política actual.

MR-03. Mejora de la movilidad. ✓

Al ser un sistema en la nube solventaría los problemas de infraestructura que no permiten a los desarrolladores trabajar desde otra ubicación que no sea la oficina.

MR-04. Mejora en la fidelidad testing – producción. ✓

Por el mismo factor que el **SDO-01**.

MR-05. Facilidad en la adaptación. ✓

Las interfaces de los servicios son intuitivas y sencillas de usar. El esquema planteado para la migración parece fácil de implementar mediante las herramientas que ponen a disposición de los clientes.

MR-06. Escalable. ✗

El servicio que prestan es totalmente autoescalable a nivel de máquina y no a nivel de infraestructura.

MR-07. Agilidad. ✓

A través de la creación de imágenes de máquina podemos pasar entre producción y *testing* de forma rápida.

MR-08. Compatibilidad. ✓

Permite una gran variedad de sistemas operativos y *software*, así como un servicio dedicado al hospedaje y administración de bases de datos [75].

Aunque el servicio promete nos encontramos ante problemas en la flexibilidad del sistema, ya que solo permite el autoescalado a nivel de máquina y no de infraestructura, por lo que este proveedor queda descartado.

3.4 Heroku



Heroku [76] [77] es un servicio en la nube basado en el modelo PaaS (plataforma como servicio), fue fundada en el 2007 y su sede está en San Francisco, California. Si bien es uno de los mejores servicios para implementar aplicaciones en la nube no es un servicio válido para cubrir nuestras necesidades, principalmente por la incompatibilidad con las tecnologías que usamos para desarrollo y producción (PHP principalmente [78]). Además, en su servicio no integra bases de datos ni permite la generación de ficheros dinámicos. Por lo tanto, en este caso se ha descartado directamente al proveedor sin hacer un análisis más profundo de sus servicios.

3.5 Joyent™



Joyent [79] [80] es una compañía de hardware y *software* especializada en aplicaciones virtuales y en la nube, fue fundada en el 2004 y su sede está en San Francisco, California. Sigue dos modelos de negocio principalmente: IaaS (infraestructura como servicio) y PaaS (plataforma como servicio), los cuales se pueden probar de forma gratuita. De ambos, el modelo que podría cubrir nuestras necesidades es IaaS (infraestructura como servicio), modelo que se analiza a continuación respecto a los requisitos y objetivos:

SP0-01. Poca flexibilidad. ✓

Permite el escalado dinámico del sistema y los servidores en caliente, por lo que cubriría el objetivo de escalabilidad [81].

SPO-02. Alto coste de escalabilidad y mantenimiento. ✓

Al poder crear máquinas virtuales (instancias) de forma dinámica [81] se reduce drásticamente el coste en infraestructura, ya que no se ha de comprar un servidor para cubrir la demanda de forma permanen-

te o en picos de uso. También hay que tener en cuenta que se paga por uso, de forma que los costes son dinámicos y ajustados.

SPO-03. Política de copias de respaldo y recuperación de desastres. ✓

Cuenta con un servicio enfocado al almacenamiento de datos en la nube [82] y un sistema de backups en caliente para bases de datos y servidores [83], lo que mejoraría sustancialmente nuestro sistema actual ante desastre, por la rapidez del proceso y los datos replicados.

SPO-04. Tiempos de inactividad. ✓

Los tiempos de inactividad por mantenimiento serían eliminados, puesto que ofrecen un servicio continuo sin interrupciones [84]. En cuanto a los tiempos de despliegue se podrían ver minimizados o totalmente eliminados según la infraestructura lógica que se implemente.

SPO-05. Nula política de actualizaciones. ✓

Joyent trabaja con las últimas versiones de *software* presente en el mercado así que una migración hacia estos servicios comportará forzosamente una actualización de los sistemas. Se puede corroborar este dato tras probar sus servicios.

SDO-01. Poca credibilidad del entorno de testing. ✓

La mejor característica del sistema de instancias es que estas se pueden destruir y crear fácilmente desde una imagen de otra máquina, por lo tanto, podemos crear una instancia que sea exactamente igual, tanto en prestaciones como en *software*, a la de producción. Esto nos garantiza que las pruebas de *testing* son totalmente fidedignas a la realidad.

SDO-02. Poca movilidad de los desarrolladores. ✓

Al ser un sistema en la nube este punto queda totalmente resuelto.

SDO-03. Diversidad de las distintas máquinas. ✓

Al migrar y actualizar el sistema de producción se hace necesaria la actualización de los sistemas de desarrollo.

SR-01. El acceso al servicio ha de ser seguro. ✓

El acceso a las aplicaciones de gestión de los servicios admiten doble autenticación [85] y se realizan por canales de comunicación seguros (SSL - *Secure Sockets Layer*). Por otro lado, todos los accesos para administrar las instancias se realizan por SSH (*Secure Shell*) junto a un certificado que se genera al crearla, incluso, se puede usar doble autenticación para este protocolo.

SR-02. El servicio contratado debe tener asociado un sistema de soporte. ✗

Cuenta con manuales de iniciación [86] así como una wiki [87] completa donde quedan detallados todas las características de sus servicios junto a manuales de uso. Además se pueden reportar problemas mediante tickets [88]. No obstante, no consideramos que esto sea suficiente para gente que se está iniciando en este mundo y se echa en falta un teléfono u otros medios de comunicación más inmediatos y directos.

SR-03. El proveedor ha de ser fiable. ✓

Joyent dio servicio a THQ y Twitter en sus inicios. Actualmente es el proveedor de LinkedIn, Gilt Groupe and Kabam [79]. Consideremos que si estas empresas han depositado su confianza en él es que es un servicio fiable. Además, cuenta con diversos certificados de seguridad [89].

SR-04. Privacidad en los datos hospedados. ✓

Cumple la directiva Europea 95/46/CE [90] y aplica el programa *Safe Harbor* [91] desarrollado por la UE para la transacción de datos en UE/Suiza y USA. [89]

SR-05. Borrado permanente tras nuestra elección. ✓

Según su política de privacidad [92] y términos de uso [93], al eliminar una instancia se borran de forma permanente todos los datos que contenía, por lo que recomiendan, antes de borrarla, realizar una copia de respaldo en local.

SR-06. Libertad de gestión. ✓

Tenemos total libertad tanto a la hora de gestionar la infraestructura como las instancias creadas.

SR-07. Transparencia. ✓

Aunque no es fácil de encontrar, se puede obtener información sobre que sistemas de seguridad implementan a todos los niveles [89].

SR-09. Próximo y motivador. ✓

Tiene una sección dedicada a casos de éxito [94] así como un calendario con eventos [95] a los que se pueden asistir.

SR-10. Disponibilidad y replicación de los datos. ✓

Permite obtener en todo momento la información almacenada en la nube, tanto por los usuarios como por los empleados, al no existir un sistema de mantenimiento.

SR-11. Cumplimiento de la legalidad. ✓

A través de sus términos de uso [93] y políticas de privacidad [92] se puede observar como cumple la legalidad vigente a nivel internacional. Especialmente a lo que se refiere a protección de datos.

SR-12. Cancelación sin repercusiones. ✓

Al ser un servicio bajo demanda y que solo se paga por uso, el simple hecho de dejar de usarlo ya te exime de cualquier responsabilidad. Es importante tener en cuenta que se estipula que se ha dejado de usar su servicio en el momento que se elimina la última instancia de la nube. Tener una instancia parada puede repercutir en gastos.

MR-01. Ventaja económica. ✓

Al ser un servicio de pago por uso el coste es más ajustado a que si la empresa ha de comprar un servidor en previsión al uso que hará de él.

MR-02. Mejora en la política de copias de respaldo y recuperación de desastres. ✓

Tal como se ha comentado en el punto **SPO-03** este proveedor nos permite mejorar la política actual.

MR-03. Mejora de la movilidad. ✓

Al ser un sistema en la nube solventaría los problemas de infraestructura que no permiten a los desarrolladores trabajar desde otra ubicación que no sea la oficina.

MR-04. Mejora en la fidelidad testing – producción. ✓

Por el mismo factor que el **SDO-01**.

MR-05. Facilidad en la adaptación. ?

Al no cumplir el requisito SR-02 no hemos entrado a analizar este punto en profundidad.

MR-06. Escalable. ✓

Es un servicio autoescalable [81].

MR-07. Agilidad. ✓

A través de la creación de imágenes de instancias podemos pasar a producción de forma rápida y sencilla.

MR-08. Compatibilidad. ✓

Permite una gran variedad de sistemas operativos y *software*. Entre ellos se encuentra el sistema operativo que necesitamos, que sería, Ubuntu 12.04 o Debian 7. Además, tiene un servicio enfocado a bases de datos MySQL [96].

Debido a que no cumple con el requisito de soporte técnico Joyent queda descartado. De cara a un futuro, con más experiencia en la nube, podría ser una posibilidad.

3.6 Linode



linode.com

Linode [97] [98] es una compañía de hardware en la nube basada en el modelo de negocio IaaS (infraestructura como servicio), fue fundada en el 2003 y la localización de su sede es Galloway, New Jersey. Concretamente es un proveedor de VPS

(Virtual Private Server) con sistemas operativos Linux.

A continuación se analiza el proveedor y el servicio respecto a los requisitos y objetivos que ha de cumplir:

SPO-01. Poca flexibilidad. ✘

Es un sistema escalable, pero al igual que DigitalOcean es más complejo que sus competidores, ya que hay que hacer uso de la API (*Application Programming Interface*) [99] que proporcionan y del servicio NodeBalancer [100] [101]. Se puede escalar un servidor o clonarlo [102] a través de *scritps*. Al buscar sencillez, este punto no se cumple.

SPO-02. Alto coste de escalabilidad y mantenimiento. ✓

Al poder generar máquinas virtuales (instancias) de forma dinámica se reduce drásticamente el coste en infraestructura, ya que no se ha de comprar un servidor para cubrir la demanda de forma permanente o en picos de uso.

SPO-03. Política de copias de respaldo y recuperación de desastres. ✓

Tiene un servicio enfocado a la creación de copias de respaldo a través de imágenes de máquina, por lo que mejora sustancialmente nuestro sistema actual. [103]

SPO-04. Tiempos de inactividad. ✓

Los tiempos de inactividad por mantenimiento serían eliminados, puesto que ofrecen un servicio continuo sin interrupciones [104]. En cuanto a los tiempos de despliegue se podrían ver minimizados o totalmente eliminados según la infraestructura lógica que se implemente.

SPO-05. Nula política de actualizaciones. ✓

Linode trabaja con las últimas versiones de Software presente en el mercado así que una migración hacia estos servicios comportará forzosamente una actualización de los sistemas. [105]

SDO-01. Poca credibilidad del entorno de testing. ✓

Cabe destacar del sistema de instancias la facilidad de destrucción y creación de estas desde una imagen de otra máquina, por lo tanto, podemos crear una instancia que sea exactamente igual, tanto en prestaciones como en *software*, a la de producción. Esto nos garantiza que las pruebas de *testing* son totalmente fidedignas a la realidad.

SDO-02. Poca movilidad de los desarrolladores. ✓

Al ser un sistema en la nube este punto queda totalmente resuelto.

SDO-03. Diversidad de las distintas máquinas. ✓

Al migrar y actualizar el sistema de producción se hace necesaria la actualización de los sistemas de desarrollo.

SR-01. El acceso al servicio ha de ser seguro. ✓

El acceso a las aplicaciones de gestión de los servicios admiten doble autenticación [106] y se realizan por canales de comunicación seguros (SSL - *Secure Sockets Layer*). Por otro lado, todos los accesos para administrar las instancias se realizan por SSH (*Secure Shell*) junto a un certificado.

SR-02. El servicio contratado debe tener asociado un sistema de soporte. ✓

Cuenta con una wiki repleta de guías para principiantes y avanzados [107], un IRC Chat [108], un foro [109], un teléfono de contacto [110] y un sistema de tickets para este fin [110].

SR-03. El proveedor ha de ser fiable. ?

Se desconoce la fiabilidad de este proveedor, aunque por las opiniones de sus clientes parece ofrecer un buen servicio. Por otra parte, parece que fue vulnerado en el 2012 causando un robo masivo de Bitcoins y cuentas [111].

SR-04. Privacidad en los datos hospedados. ✓

Según su política de privacidad solo utiliza de los datos privados para proporcionarnos sus servicios y no accede a los datos hospedados excepto por requerimiento legal. Por otro lado, al igual que Joyent, cumple con la directiva Europa ante privacidad [90] y con el protocolo *Safe Harbor* [91] para el intercambio de datos entre EU y USA. [112]

SR-05. Borrado permanente tras nuestra elección. ?

Es un punto complejo de corroborar. No se tiene constancia de cuál es su política ante esta situación.

SR-06. Libertad de gestión. ✓

Tenemos total libertad tanto a la hora de gestionar la infraestructura como los VPS (*Virtual Private Server*) creados. A nivel de infraestructura debido a que podemos crearnos la nuestra propia mediante distintos VPS (*Virtual Private Server*), y de cara a los VPS (*Virtual Private Server*), debido a que podemos instalar el *software* que necesitemos.

SR-07. Transparencia. ✓

A través de la wiki del proveedor [107] y la página web [98] podemos obtener información referente a la tecnología usada tanto en infraestructura como en seguridad.

SR-08. Incorporación de sistemas de alerta y análisis. ✓

A través del servicio NodeBalancer [101] y la interfaz web del servicio se pueden gestionar alertas y análisis de los VPS (*Virtual Private Server*). De la misma forma a través de *scripts* que hagan uso de su API (*Application Programming Interface*) [99]. [113]

SR-09. Próximo y motivador. ✘

A pasear de disponer de un blog [114] donde van poniendo sus actualizaciones y progresos se desconocen eventos a los que asista o casos de éxito.

SR-10. Disponibilidad y replicación de los datos. ✓

Permite obtener en todo momento la información almacenada en la nube, tanto por los usuarios como por los empleados, al no existir un sistema de mantenimiento.

SR-11. Cumplimiento de la legalidad. ✓

A través de la lectura de sus términos y servicios [115] como de la política de privacidad [112] vemos que cumple con la legalidad vigente a nivel internacional.

SR-12. Cancelación sin repercusiones. ✗

No es un servicio bajo demanda, así que se paga mes a mes, por lo que si se cancela el contrato a mediados de mes se pierde el dinero correspondiente a los días restantes de uso del servidor.

MR-01. Ventaja económica. ✓

Todo y no ser un servicio de pago bajo demanda el coste es más ajustado a que si la empresa ha de comprar un servidor en previsión al uso que hará de él.

MR-02. Mejora en la política de copias de respaldo y recuperación de desastres. ✓

Tal como se ha comentado en el punto **SPO-03** este proveedor nos permite mejorar la política actual.

MR-03. Mejora de la movilidad. ✓

Al ser un sistema en la nube solventaría los problemas de infraestructura que no permiten a los desarrolladores trabajar desde otra ubicación que no sea la oficina.

MR-04. Mejora en la fidelidad testing – producción. ✓

Por el mismo factor que el **SDO-01**.

MR-05. Facilidad en la adaptación. ✗

Las interfaces de los servicios son intuitivas y sencillas de usar, no obstante, la necesidad de crear *scripts* para implementar algunas funcionalidades hacen que la adaptación sea más compleja frente a otros servicios.

MR-06. Escalable. ✗

El servicio no es autoescalable de forma sencilla, hay que implementar *scripts* para tal fin.

MR-07. Agilidad. ✓

A través de la creación de imágenes de máquina podemos pasar entre producción y *testing* de forma rápida.

MR-08. Compatibilidad. ✗

Cuenta con los sistemas operativos necesarios para correr nuestra aplicación. No obstante, no cuenta con un servicio de bases de datos, lo que no nos permitiría adaptarnos al esquema propuesta, por lo que ese requisito no lo cumple.

Debido al gran número de requisitos que no cumple este proveedor queda descartado.

3.7 OpenShift



OpenShift [116] [117] es un producto de computación en la nube que sigue un modelo de comercialización PaaS (Plataforma como servicio), se lanzó en 2011 y pertenece a RedHat.

En nuestro caso, aunque sería un proveedor capaz de cubrir la mayoría de los requisitos y objetivos presentados, lo hemos tenido que descartar debido a que sus sistemas solo funcionan bajo el sistema operativo RedHat.

3.8 RackSpace®



Rackspace [118] [119] es una compañía de hospedaje en la nube que sigue un modelo de negocio IaaS (infraestructura como servicio), se fundó en 1998 y su sede está situada en City of Windcrest, San Antonio. Es una de las empresas más veteranas en el sector y junto a Amazon una de las que más apuestan por la innovación.

A continuación se analiza el proveedor y el servicio de *cloud* híbrida respecto a los requisitos y objetivos que ha de cumplir:

SPO-01. Poca flexibilidad. ✓

El sistema es escalable a través de imágenes de máquina/instancia [120]. Permite la distribución de carga entre ellos mediante un servicio de balanceo dedicado [121].

SPO-02. Alto coste de escalabilidad y mantenimiento. ✓

Al poder generar instancias de forma dinámica se reduce drásticamente el coste en infraestructura, ya que no se ha de comprar un servidor para cubrir la demanda de forma permanente o en picos de uso. También hay que tener en cuenta que se paga por uso, de forma que los costes son dinámicos y ajustados. Quizás, el único inconveniente en este caso, es que es un poco más caro que la media, por lo que un posible ahorro no sería tan elevado.

SPO-03. Política de copias de respaldo y recuperación de desastres. ✓

Ofrece un servicio de almacenaje en la nube [122] así como otro enfocado a la creación de copias de respaldo [123]. Por lo tanto, se podría llegar a mejorar sustancialmente nuestro sistema actual.

SPO-04. Tiempos de inactividad. ✓

Los tiempos de inactividad por mantenimiento serían eliminados, puesto que ofrecen un servicio continuado, sin interrupciones [124]. En cuanto a los tiempos de despliegue se podrían ver minimizados o totalmente eliminados según la infraestructura lógica que se implemente.

SPO-05. Nula política de actualizaciones. ✓

Trabaja con las últimas versiones de Software [125] presente en el mercado así que una migración hacia estos servicios comportará forzosamente una actualización de los sistemas.

SDO-01. Poca credibilidad del entorno de testing. ✓

Lo bueno del sistema de instancias es que estas se pueden destruir y crear fácilmente desde una imagen de otra máquina, por lo tanto, podemos crear una instancia que sea exactamente igual, tanto en prestaciones como en *software*, a la de producción. Esto nos garantiza que las pruebas de *testing* son totalmente fidedignas a la realidad.

SDO-02. Poca movilidad de los desarrolladores. ✓

Al ser un sistema en la nube este punto queda totalmente resuelto.

SDO-03. Diversidad de las distintas máquinas. ✓

Al migrar y actualizar el sistema de producción se hace necesaria la actualización de los sistemas de desarrollo.

SR-01. El acceso al servicio ha de ser seguro. ✓

El acceso a las aplicaciones de gestión de los servicios se realiza por canales de comunicación seguros (SSL - *Secure Sockets Layer*). Por otro lado, todos los accesos para administrar las instancias se realizan por SSH (*Secure Shell*) mediante certificado o usuario-clave [125].

SR-02. El servicio contratado debe tener asociado un sistema de soporte. ✓

Cuenta con un servicio técnico bastante completo: chats, tickets, teléfonos, correo electrónico, foros, manuales y servicios de asistencia. Para una asistencia continuada, con cortos plazos de respuesta y enfocada a ciertos temas hay que contratar servicios adicionales con un coste mensual asociado. [126] [127]

SR-03. El proveedor ha de ser fiable. ✓

Su veteranía en el mundo del *cloud computing* junto al renombre de sus clientes (Mazda, Dominos Pizza, VEVO, etc) [128] nos ha llevado a determinar que es un proveedor fiable. Por otro lado, cumple que la normativa ISO27001 [1] [129].

SR-04. Privacidad en los datos hospedados. ✓

Según la declaración de privacidad [130] del proveedor los datos recopilados no son cedidos a terceros, siendo además nosotros, los únicos responsables de los datos almacenados en los servicios. Por otro lado, cumple la política Safe Harbor [91] para el intercambio de datos entre Estados Unidos y la Unión Europea/Suiza. Solo se contempla el acceso a nuestros datos bajo procesos judiciales.

SR-05. Borrado permanente tras nuestra elección. ?

Es un punto complejo de corroborar. No se tiene constancia de cuál es su política ante esta situación.

SR-06. Libertad de gestión. ✓

Tenemos total libertad tanto a la hora de gestionar la infraestructura como las instancias creadas. A nivel de infraestructura debido a que podemos hacer uso solo de aquellos servicios que necesitemos usar en el momento que deseemos pudiendo llegar a montar nuestra propia red privada, y de cara a las instancias, debido a que podemos instalar el *software* que necesitemos.

SR-07. *Transparencia.* ✓

Mediante su página web se puede consultar tanto las tecnologías que emplean como su infraestructura. Así como leyes que cumplen, certificados, etc. [131]

SR-08. *Incorporación de sistemas de alerta y análisis.* ✓

Cuenta con una herramienta para el monitoreo y análisis de los sistemas en la nube.

SR-09. *Próximo y motivador.* ✓

Organiza y acude a eventos [132], contiene un blog [133] donde cuelga noticias del sector y sus avances, una sección dedicada a casos de éxito [134], etc.

SR-10. *Disponibilidad y replicación de los datos.* ✓

Permite obtener en todo momento la información almacenada en la nube, tanto por los usuarios como por los empleados.

SR-11. *Cumplimiento de la legalidad.* ✓

Según sus términos y servicios [135] así como su política de privacidad [130] cumple con las leyes internacionales del sector.

SR-12. *Cancelación sin repercusiones.* ✓

Al ser un servicio bajo demanda no hay repercusiones asociadas al abandono de este.

MR-01. *Ventaja económica.* ✓

Al ser un servicio de pago por uso el coste es más ajustado a que si la empresa ha de comprar un servidor en previsión al uso que hará de él.

MR-02. *Mejora en la política de copias de respaldo y recuperación de desastres.* ✓

Tal como se ha comentado en el punto **SPO-03** este proveedor nos permite mejorar la política actual.

MR-03. *Mejora de la movilidad.* ✓

Al ser un sistema en la nube solventaría los problemas de infraestructura que no permiten a los desarrolladores trabajar desde otra ubicación que no sea la oficina.

MR-04. *Mejora en la fidelidad testing – producción.* ✓

Por el mismo factor que el **SDO-01**.

MR-05. *Facilidad en la adaptación.* ✓

Las interfaces de los servicios son intuitivas y sencillas de usar y el esquema planteado parece fácil de plasmar mediante lo que ofrecen.

MR-06. *Escalable.* ✓

El servicio se puede configurar sin complicaciones para que autoescale.

MR-07. *Agilidad.* ✓

A través de la creación de imágenes de máquina podemos pasar entre *testing* y producción de forma sencilla.

MR-08. *Compatibilidad.* ✓

Permite una gran variedad de sistemas operativos y *software* para las instancias en la nube. Además, cuenta con un servicio dedicado para el almacenaje en bases de datos. [125] [136]

Rackspace parece ser una buena alternativa para realizar la migración, aunque da la sensación que va enfocado a un público experimentado.

3.9 VMware vCloud®.



VMware vCloud [137] [138] es un servicio de computación en la nube que se comercializa siguiendo el modelo IaaS (infraestructura como servicio), lanzado en el 2009 y perteneciente a la empresa VMware.

En este caso no ha hecho falta hacer un análisis profundo para descartarlo, ya que por ahora, este servicio solo está presente en Estados Unidos y tiene prevista su internacionalización durante el 2014. [139]

3.10 Windows Azure



Windows Azure [140] [141] engloba un conjunto de servicios en la nube ofrecidos por Microsoft y lanzados al mercado en el 2010. Nació bajo el modelo de negocio PaaS (plataforma como servicio), servicio por el que se le conoce, aunque desde del 2012 se adentró en el modelo de negocio IaaS (infraestructura como servicio). Actualmente ofrece una prueba gratuita de sus servicios.

En este caso analizaremos los servicios asociados al modelo IaaS (infraestructura como servicio), los cuales son similares a los ofrecidos por Amazon AWS. Los resultados del análisis han sido los siguientes:

SPO-01. Poca flexibilidad. ✓

Los servicios de Azure permiten el autoescalado del sistema de tal forma que se adapte a la demanda. Se puede configurar esta opción de forma sencilla desde el panel de configuración de las máquinas virtuales. El escalado se puede automatizar por el uso de CPU o por el número de mensajes en cola de entrada al servidor. [142]

Para hacer el balanceo del tráfico entre las distintas máquinas hay que usar Azure Load Balancer. [143]

SPO-02. Alto coste de escalabilidad y mantenimiento. ✓

A pesar de no ser un servicio de los más económicos sigue siendo rentable para la empresa, ya que permite escalar el sistema de forma sencilla sin la necesidad de inversión por parte de la empresa en infraestructura.

En este caso el coste es por uso y aunque se especifique en horas, cobra por minutos de uso.

SPO-03. Política de copias de respaldo y recuperación de desastres. ✓

Azure ofrece un servicio de almacenamiento en la nube que nos puede servir para almacenar las copias de seguridad de nuestros datos de tal forma que podemos mejorar nuestra política de copias de respaldo y recuperación de desastres de forma considerable. [144]

En él podríamos guardar las imágenes virtuales de las máquinas como copias de seguridad. Además, al ser esta una imagen de máquina no haría falta ante un desastre reinstalar todo el *software*, y al estar todo bajo la misma red, se acelera el proceso tanto en la creación como la recuperación de la copia.

SPO-04. Tiempos de inactividad. ✓

Los tiempos de inactividad por mantenimiento serían eliminados puesto que ofrecen un servicio continuo sin interrupciones [145]. En cuanto a los tiempos de despliegue se podrían ver minimizados o totalmente eliminados según la infraestructura lógica que se implemente.

SPO-05. Nula política de actualizaciones. ✓

Azure trabaja con las últimas versiones de Software presente en el mercado así que una migración hacia estos servicios comportará forzosamente una actualización de los sistemas.

SDO-01. Poca credibilidad del entorno de testing. ✓

Al tratarse de máquinas virtuales podemos crear una máquina virtual exactamente igual al sistema de producción para usarla de *testing*. Además, una vez realizado el *testing* podríamos dejar la máquina de *testing* como producción, cerrar la máquina de producción y crear una imagen virtual de la de *testing* como copia de respaldo.

Como se aprecia, el *testing* es totalmente fidedigno a lo que será el sistema de producción. Además, como en Amazon, podemos crear varias máquinas virtuales que hagan peticiones a la de *testing* para ver como esta se comportaría ante un gran número de estas.

SDO-02. Poca movilidad de los desarrolladores. ✓

Al ser un sistema en la nube este punto queda totalmente resuelto.

SDO-03. Diversidad de las distintas máquinas. ✓

Al migrar y actualizar el sistema de producción se hace necesaria la actualización de los sistemas de desarrollo.

SR-01. El acceso al servicio ha de ser seguro. ✓

El acceso al administrador de los servicios permite autenticación de doble factor [146] y se realizan por canales de comunicación seguros (SSL - *Secure Sockets Layer*). Por otro lado, todos los accesos a las máquinas virtuales permiten el uso de SSH (*Secure Shell*) junto a un certificado [147], aunque, para instancias rápidas, permite configurar el acceso mediante clave y usuario.

SR-02. El servicio contratado debe tener asociado un sistema de soporte. ✗

Azure cuenta con manuales de soporte *online* [148], así como con foros [149] o *emails* de contacto. Las guías son fáciles de seguir y efectivas. Lamentablemente si se quieren respuestas rápidas o atención telefónica hay que pagar un precio elevado [150], y para gente sin experiencia, esto es un impedimento.

SR-03. El proveedor ha de ser fiable. ✓

Microsoft es uno de los gigantes de la informática y cuenta con los recursos necesarios para garantizar la fiabilidad de sus servicios. Así que este punto lo contabilizamos como un voto de confianza.

SR-04. Privacidad en los datos hospedados. ✓

La política de privacidad [151] de Azure especifica que los datos de cliente, de cualquier nivel, no se revelan a terceros a no ser que sea por imperativo legal o cesión de servicios, si este lo requiere. En el momento de los cobros podría llegar a ceder los datos personales para luchar contra el fraude económico.

SR-05. Borrado permanente tras nuestra elección. ?

Es un punto complejo de corroborar. No se tiene constancia de cuál es su política ante esta situación.

SR-06. Libertad de gestión. ✓

Tenemos total libertad tanto a la hora de gestionar la infraestructura como las instancias creadas. A nivel de infraestructura debido a que podemos hacer uso solo de aquellos servicios que necesitemos, y de cara a las máquinas virtuales, debido a que podemos instalar el *software* deseado.

SR-07. Transparencia. ✓

Aunque no es fácil de encontrar, se puede obtener información sobre que sistemas de seguridad implementan a todos los niveles. [152] [153] [154]

SR-08. Incorporación de sistemas de alerta y análisis. ✓

En el panel de administración se pueden configurar sistemas de alerta y análisis, y a través de estos, automatizar acciones. [155]

SR-09. Próximo y motivador. ✓

Contiene una sección dedicada a la formación [156] en su portal web junto a casos de éxito [157], realizan videos explicativos y de interés sobre sus servicios [158] además de eventos para los clientes [159].

SR-10. Disponibilidad y replicación de los datos. ✓

Permite obtener en todo momento la información almacenada en la nube, tanto por los usuarios como por los empleados, al no existir un sistema de mantenimiento.

SR-11. Cumplimiento de la legalidad. ✓

Al igual que Amazon AWS, Microsoft Azure cumple la legalidad vigente en función de la zona donde se esté usando su servicio.

SR-12. Cancelación sin repercusiones. ✓

Al ser un servicio que se paga por uso no hay ninguna repercusión al dejar de usarlo.

MR-01. Ventaja económica. ✓

Al ser un servicio de pago por uso el coste es más ajustado a que si la empresa ha de comprar un servidor en previsión al uso que hará de él. Generalmente, porque al comprar un servidor bajo esas premisas se suelen cometer errores de sobredimensionado, que causan pérdidas económicas, o errores de infra-valoración, que causan que a corto plaza haya que adquirir uno de nuevo.

A lo ya mencionado, hay que sumar que el coste de una infraestructura es caro, en especial en cuanto a seguridad, monitorización y energía. Si tenemos un sistema sobredimensionado estos costes se disparan.

MR-02. Mejora en la política de copias de respaldo y recuperación de desastres. ✓

Tal como se ha comentado en el punto **SPO-03** este proveedor nos permite mejorar la política actual.

MR-03. Mejora de la movilidad. ✓

Al ser un sistema en la nube solventaría los problemas de infraestructura que no permiten a los desarrolladores trabajar desde otra ubicación que no sea la oficina.

MR-04. Mejora en la fidelidad testing – producción. ✓

Por el mismo factor que el **SDO-01**.

MR-05. Facilidad en la adaptación. ✓

Las interfaces de los servicios son aún más intuitivas y sencillas de usar que en Amazon, el esquema planteado, a priori, parece fácil de plasmar mediante lo que ofrecen.

MR-06. Escalable. ✓

El servicio prestado es totalmente escalable y automatizable

MR-07. Agilidad. ✓

A través de la creación de imágenes de máquinas virtuales podemos pasar del sistema de *testing* al de producción de forma rápida y sencilla. O simplemente, una vez validada la aplicación en *testing* pasar la máquina a producción.

MR-08. Compatibilidad. ✗

Aunque permite la instalación del sistema operativo que necesitamos no proporciona un servicio para bases de datos relacionales que no sean Microsoft SQL Server. Se podría solventar instalando MySQL en la misma máquina de producción pero entonces ya no nos adaptaríamos al diseño previsto.

Como queremos un servicio centralizado que cubra todas las necesidades, Azure, para nosotros, no cumple con el requisito de compatibilidad.

Mediante el análisis podemos observar que hay dos requisitos que no cumple, el de compatibilidad y el de soporte técnico. Al ser estos requisitos indispensables para la migración este proveedor que descartado.

3.11 Resumen

En este apartado se muestra un resumen, representado en una tabla (**Tabla 10**), del análisis realizado. Este también contiene un pequeño análisis sobre el coste anual que supondría implementar el modelo de referencia en aquellos proveedores que, en un principio, cumplen los objetivos y requisitos del proyecto. Hay que tener en cuenta que el coste total nunca será el real, puesto que son servicios de pago por uso. Si bien esto supone una ventaja en cuanto a costes finales, ya que se paga por recurso consumido, supone una desventaja en cuanto a previsión de costes.

		Proveedores de servicios en la nube						
		Amazon AWS	DigitalOcean	DinaHosting RealCloud	Joyent	Linode	RackSapce	Windows Azure
Objetivos	SPO-01	✓	✗	✗	✓	✗	✓	✓
	SPO-02	✓	✓	✓	✓	✓	✓	✓
	SPO-03	✓	✓	✓	✓	✓	✓	✓
	SPO-04	✓	✓	✓	✓	✓	✓	✓
	SPO-05	✓	✓	✓	✓	✓	✓	✓
	SDO-01	✓	✓	✓	✓	✓	✓	✓
	SDO-02	✓	✓	✓	✓	✓	✓	✓
SDO-03	✓	✓	✓	✓	✓	✓	✓	
Requisitos en el servicio a contratar	SR-01	✓	✓	✓	✓	✓	✓	✓
	SR-02	✓	✓	✓	✗	✓	✓	✗
	SR-03	✓	?	?	✓	?	✓	✓
	SR-04	✓	✓	✓	✓	✓	✓	✓
	SR-05	?	?	?	✓	?	?	?
	SR-06	✓	✓	✓	✓	✓	✓	✓
	SR-07	✓	✓	✓	✓	✓	✓	✓
	SR-08	✓	✗	✓	✓	✓	✓	✓
	SR-09	✓	✓	✓	✓	✗	✓	✓
	SR-10	✓	✓	✓	✓	✓	✓	✓
	SR-11	✓	✓	✓	✓	✓	✓	✓
	SR-12	✓	✓	✓	✓	✗	✓	✓
Requisitos de la migración	MR-01	✓	✓	✓	✓	✓	✓	✓
	MR-02	✓	✓	✓	✓	✓	✓	✓
	MR-03	✓	✓	✓	✓	✓	✓	✓
	MR-04	✓	✓	✓	✓	✓	✓	✓
	MR-05	✓	✗	✓	?	✗	✓	✓
	MR-06	✓	✗	✗	✓	✗	✓	✓
	MR-07	✓	✓	✓	✓	✓	✓	✓
	MR-08	✓	✗	✓	✓	✗	✓	✗
Valoración final		✓	✗	✗	✗	✗	✓	✗
Coste por servicio que cumple con las expectativas								
Coste por servicio*	Instancia ¹	570'96 €	-	-	-	-	516'47 €	-
	Datos ²	6'45 €	-	-	-	-	132'79 €	-
	Tráfico ³	6'96 €	-	-	-	-	5'17 €	-
	BB.DD ⁴	121'17 €	-	-	-	-	258'21 €	-
	Balanceador ⁵	1'68 €	-	-	-	-	193'32 €	-
	Supervisión	18'00 €	-	-	-	-	13'27 €	-
	Coste total	725.32 €	-	-	-	-	1.119'23 €	-

Tabla 10 - Resumen del análisis de las soluciones existentes.

**Los costes presentados son anuales. Para su cálculo se han contabilizado los servicios más básicos de cada proveedor, por lo que el coste final puede incrementar sustancialmente.*

¹ *Para las instancias se ha tenido en cuenta la instancia de menor capacidad, puesto que el sistema ha de ser escalable. Además, se ha tenido en cuenta que esta estaría funcionando 24 horas diarias para producción y 48 horas al mes para testing.*

² *El coste de datos tiene en cuenta las solicitudes al sistema de almacenamiento, así como el coste asociado a las transacciones con el servicio. Se ha calculado para un almacenamiento de 2Gb y cinco transacciones diarias.*

³ *Precio de tráfico calculado para 5Gb mensuales. Que estipulamos que es el mínimo generado entre producción, desarrollo y testing.*

⁴ *Para una base de datos MySQL de 20 GB y rendimiento medio.*

⁵ *Para un balanceador con un tráfico de 5Gb.*

Si bien lo más sensato sería presentar una estimación de posibles costes en función de la demanda actual y la prevista de los servicios ofrecidos por RDmes, la empresa ha decidido, por ahora, no realizar las estimaciones puesto que considera la migración un paso obligado para la explotación de la plataforma, pasando generalmente el coste a un segundo plano. Además, esta se ofrece gratuitamente a los usuarios quedando aún por definir un plan para su rentabilización. Por otro lado, la plataforma es joven y no se tiene constancia del volumen de usuarios a lo largo del tiempo ni una estimación de su posible crecimiento.

3.12 Conclusiones

Después del análisis y de probar diversos proveedores (interfaz, facilidad de uso, etc.), vemos que solo dos de estos podrían llegar a cumplir todas nuestras expectativas. Estos proveedores son Amazon AWS y RackSpace.

Como ambas alternativas tienen un gran renombre y el servicio que prestan es de calidad, no queda otra que basarnos en el precio para decantarnos por una de ellas, aunque como ya se ha mencionado previamente, no es un baremo aceptable.

En la actualidad el coste total en infraestructura asciende a unos 1.400 euros, entre alquiler del rack, el coste eléctrico, la amortización y otros factores. Frente a este valor, es Amazon AWS quien en un principio nos proporciona un ahorro mayor, y por lo tanto, es el proveedor escogido. Además, este ofrece un gran abanico de servicios que un futuro nos podrían ser útiles.

La empresa RDmes no tiene asociado ningún coste por mantenimiento o gestión de los servidores, ya que son los propios desarrolladores de los aplicativos los que se encargan de ello, por lo que no existe un ahorro asociado a estos conceptos.

4. Migración de los sistemas actuales

En este capítulo se documenta de forma general el proceso de migración y su resultado final a través de una breve introducción, una descripción del proceso y un análisis de los resultados en comparación al sistema inicial y al modelo de referencia.

Se puede consultar una versión detallada a todos los niveles de la migración en el **Anexo 01**: descripción de los procesos, instalación del software, configuración de las máquinas, etc. Esta información se ha descrito en el anexo y no en la memoria debido a que pretender ser una guía completa para la migración de los entornos y no una simple descripción de los pasos realizados.

4.1 Introducción

Durante la migración de los sistemas a Amazon AWS se han tenido que utilizar diversos servicios ofrecidos por el proveedor. Con la finalidad de poner en contexto al lector se definen dichos servicios y su utilización a lo largo de esta:

Amazon Elastic Compute Cloud (EC2).

“Amazon EC2 presenta un auténtico servicio de alojamiento en la nube, que permite utilizar interfaces de servicio web para lanzar instancias con distintos sistemas operativos, cargarlas con su entorno de aplicaciones personalizado, gestionar sus permisos de acceso a la red y ejecutar su imagen utilizando los sistemas que desee”. [23]

A lo largo del proyecto se ha utilizado para la creación y gestión de instancias y volúmenes (EBS) [160], así como para la implementación de la política de copias de respaldo a nivel de volúmenes e instancias.

Amazon Relational Database Service (RDS).

“Servicio web que facilita las tareas de configuración, operación y escalado de una base de datos relacional en la nube. Proporciona capacidad rentable y de tamaño modificable y, al mismo tiempo, gestiona las tediosas tareas de administración de la base de datos”. [44]

Se utiliza para hospedar y gestionar la base de datos de producción. También se ha implementado en el servicio la política de copias de respaldo de esta.

Amazon Simple Storage Service (S3).

“Proporciona una sencilla interfaz de servicios web que puede utilizarse para almacenar y recuperar la cantidad de datos que desee, cuando desee y desde cualquier parte de la Web”. [27]

Se utiliza como sistema de almacenamiento en la nube para los ficheros dinámicos generados por la aplicación: XML's y logos de clientes.

AutoScaling.

“Permite escalar automáticamente la capacidad de Amazon EC2, para aumentarla o reducirla, de acuerdo con las condiciones que defina.” [22]

Se usa principalmente para hacer el sistema autoescalable. A través de él se gestionan las políticas de autoescalado .

Elastic Load Balancer.

“Distribuye automáticamente el tráfico entrante de las aplicaciones entre varias instancias de Amazon EC2”.

Se usa junto a *autoescalado* [22]. Nos permite distribuir las peticiones a producción entre las instancias dinámicas de este entorno.

Amazon Simple Notification Service (SNS).

“Servicio de mensajería push rápido, flexible y completamente gestionado”.

Se usa como complemento de CloudWatch [24]. Con este servicio, ante ciertos eventos, podemos enviar notificaciones por correo electrónico.

CloudWatch.

“Proporciona la supervisión de los recursos de la nube de AWS y de las aplicaciones que los clientes ejecutan en AWS”. [24]

Nos permite supervisar las instancias y gestionar notificaciones ante eventos definidos. También se puede usar para la supervisión de Elastic Load Balancer [25] o RDS [44], aunque en este caso no se ha utilizado con tal finalidad.

Elastic IP.

Elastic IP es un servicio dentro de EC2 [23] que nos permite reservar diversas direcciones IP fijas para las instancias creadas. Posteriormente estas las podemos asociar a instancias de EC2 [23] o RDS [44]. En nuestro caso se utilizó en un principio para facilitar la puesta en marcha del sistema, pero finalmente se ha optado por acceder a las diversas instancias por su *endpoint*.

Cada servicio aquí presentado tiene un coste asociado por uso y lamentablemente, todos ellos han sido imprescindibles a la hora de cumplir todos los requisitos.

4.2 El proceso

El proceso realizado se puede estructurar en un conjunto de pasos, tal como se ha definido en el **Anexo 01**. Si los siguiéramos tendríamos que llegar al mismo resultado final, por lo tanto, siguiendo esta lógica y en orden secuencial, se presentan las acciones realizadas hasta completar la migración con éxito:

1. *Creación de la cuenta y configuración de la doble autenticación.*
2. *Creación de una instancia para producción en EC2 [23], a partir de una AMI (Amazon Machine Image) [26] de Debian 7.1. Una vez creada se ha procedido a realizar las configuraciones de seguridad pertinentes, asociarle una IP estática media Elastic IP [161], instalarle el aplicativo de RDmes y comprobar su funcionamiento, tanto a través de la web como en la misma instancia.*
3. *Creación de una instancia para desarrollo en EC2 [23], a partir de una AMI (Amazon Machine Image) [26] de Debian 7.1. Tras su creación se le ha asociado un EBS [160] persistente y se ha configurado esta para que este esté disponible al inicio.*

Se ha procedido a instalar el Subversion y se ha configurado para que guarde los repositorios en el EBS [160] persistente. Además, se ha sincronizado con el que hay en el servidor *Sol* de RDmes y se ha configurado para que sea accesible desde Internet, siempre y cuando se tengan las credenciales necesarias.

Igual que en el punto anterior, se han realizado las configuraciones de seguridad pertinentes, se ha asociado una IP estática a la instancia y se ha comprobado su correcto funcionamiento.

4. *Creación de una base de datos en RDS [44] y adaptación de la aplicación de RDmes, situada en la instancia de producción, para que haga uso de esta. Se han tenido que realizar algunas configuraciones en el servicio para poder realizar algunas acciones restringidas.*
5. *Creación de un bucket (contenedor) en S3 [27] para almacenar los datos dinámicos generados por la aplicación de RDmes. Se ha tenido que adaptar esta para que obtenga y guarde los ficheros de este servicio. Para ello se ha tenido que usar el SDK (Software Development Kit) [162] de Amazon.*
6. *Implementación de la política de copias de respaldo:* para los ficheros de S3 [27] no se ha implementado ninguna política en especial, pues el servicio cuenta con control de versiones y en principio es totalmente fiable.

Respecto a las instancias se procede a generar una AMI (Amazon Machine Image) [26] una vez están totalmente configuradas. Las AMI (Amazon Machine Image) [26] se guardan en S3 [27], así que para estas no se contempla ninguna acción especial.

Por otro lado, para los EBS [160] y la base de datos en RDS [44], se ha creado un cron en desarrollo que a partir de la API (Application Programming Interface) [163] de Amazon genera cada día un *snapshot* [164] de cada servicio. Estos se pueden administrar y ver desde la interfaz web de Amazon en el panel de control de cada uno de ellos.

No se descarta en un futuro generar otro cron para la descarga de una copia semanal de la base de datos. No se contempla para el EBS [160], puesto que su contenido es una réplica de la información contenida en el servidor *Sol* de RDmes, ni tampoco del S3 [27], tanto por las medidas de seguridad que ya lleva implementadas como por el volumen de datos que puede llegar a albergar en un futuro.

7. *Implantación del autoescalado*: para ellos se ha creado un balanceador mediante Elastic Load Balancer y a través del API (*Application Programming Interface*) [163] de Amazon se han definido las políticas para este fin. También ha sido necesario definir la configuración de inicio de cada nueva instancia.

Mediante la configuración del servicio se especifica que todas las instancias creadas han de ir tras el balanceador iniciado previamente, de tal forma que este pueda gestionar la carga que recibe cada una de ellas.

Además, se han tenido que hacer modificaciones en el servidor web de la instancia de producción para que todas las peticiones vayan dirigidas al balanceador.

8. *Implementación de las notificaciones*: mediante CloudWatch [24] se han definido notificaciones por mail para el sistema de autoescalado y la supervisión de las máquinas.

Cada punto de estos queda explicado de forma técnica y extendida en el **Anexo 01**.

4.3 El resultado

Todo el proceso realizado nos ha llevado a una nueva definición de los entornos tanto a nivel de infraestructura como de flujos de trabajo.

Si previamente las copias de seguridad de producción se generaban de forma diaria en *Aire* y una vez a la semana se copiaban a *Sol*, con el riesgo que esto comportaba, ahora basta con generar una AMI (*Amazon Machine Image*) [26] de la instancia de producción ya configurada y un *snapshot* [164] de la base de datos de forma diaria.

Respecto a la recuperación ante el desastre, si antes una falla en *Aire* suponía investigar y arreglar la causa, pudiendo suponer la restauración de los datos desde *Sol*, tanto de bases de datos como de ficheros, en la actualidad basta con cerrar la instancia que ha fallado e iniciar una nueva desde la AMI (*Amazon Machine Image*) [26] guardada. Los datos no se verían afectados como pasaba antes, puesto que estos ahora se encuentran en S3 [27](con redundancia y control de versiones) y en RDS [44]. Si fallase la base de datos con el nuevo sistema solo tendríamos que restaurar el *snapshot* [164] más reciente.

Como vemos, el sistema de copias de respaldo y recuperación ante desastres se simplifica y se hace más efectivo, pudiendo reducir los tiempos de recuperación (ver **ilustración 08**).

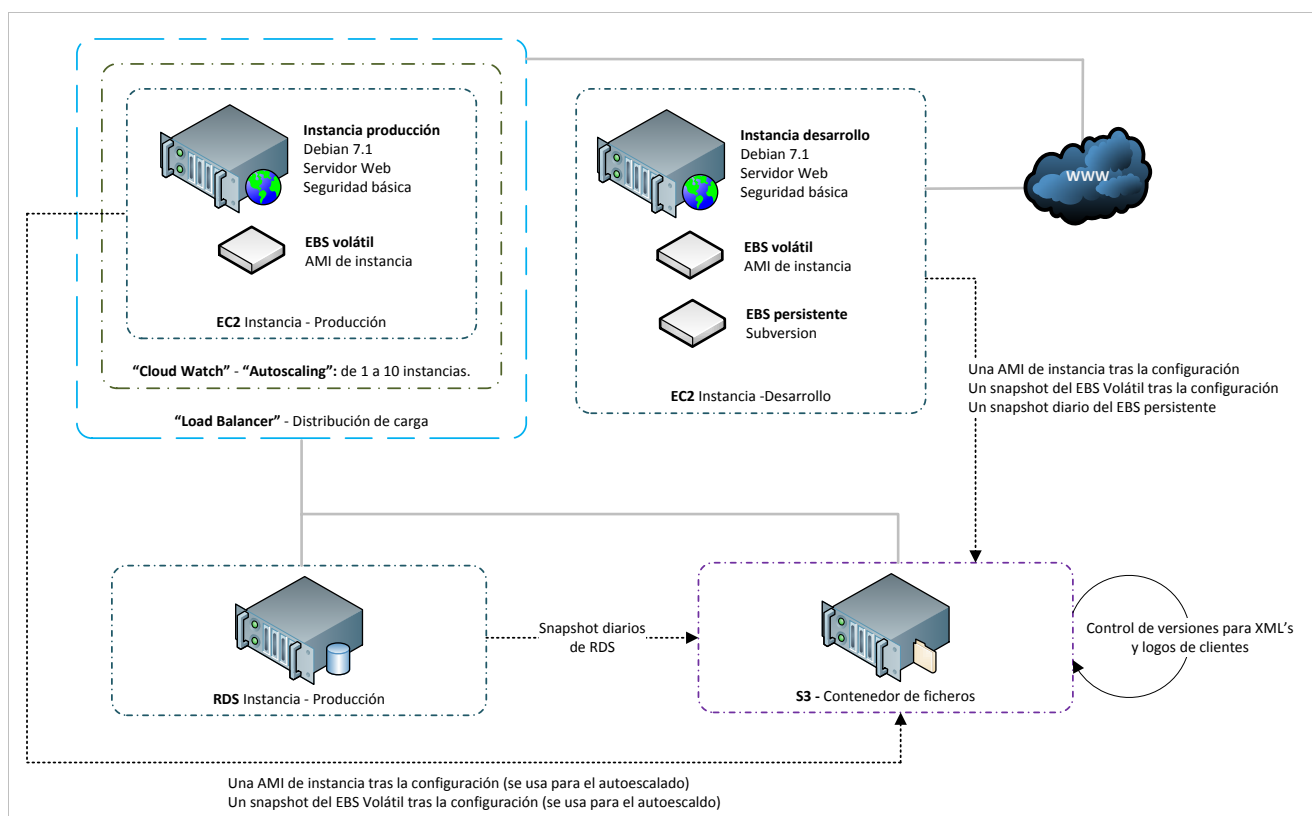


Ilustración 8 - Ciclo de copias de respaldo tras la migración

En referente al sistema de desarrollo (ver **ilustración 09**) nos encontramos ante una situación muy similar. Como el Subversion del EBS [160] de la instancia de desarrollo en la nube esta sincronizado con el del *Sol*, ambos, siempre tendrán una copia de respaldo, y aún más, como del EBS [160] se genera un *snapshot* [164] diario, en caso de fallo de ambos sistemas, siempre podremos recuperar los datos montando un nuevo volumen a través del *snapshot* [164].

En cuanto al flujo de trabajo es similar al actual, pero ahora los trabajadores pueden trabajar desde cualquier ubicación si trabajan contra el servidor de desarrollo en la nube.

Para el *testing* (ver **ilustración 09**) las cosas también han mejorado, si antes hacía falta tener una máquina sobre la que ir aplicando las actualizaciones y luego replicar estas en producción una vez validadas, ahora basta con levantar una nueva instancia desde la AMI (*Amazon Machine Image*) [26] de producción y actualizarla. Si la actualización pasa la validación se sustituye la máquina de producción actual por la nueva actualizada, se crea una AMI (*Amazon Machine Image*) [26] de esta y se actualiza la configuración del autoescalado.

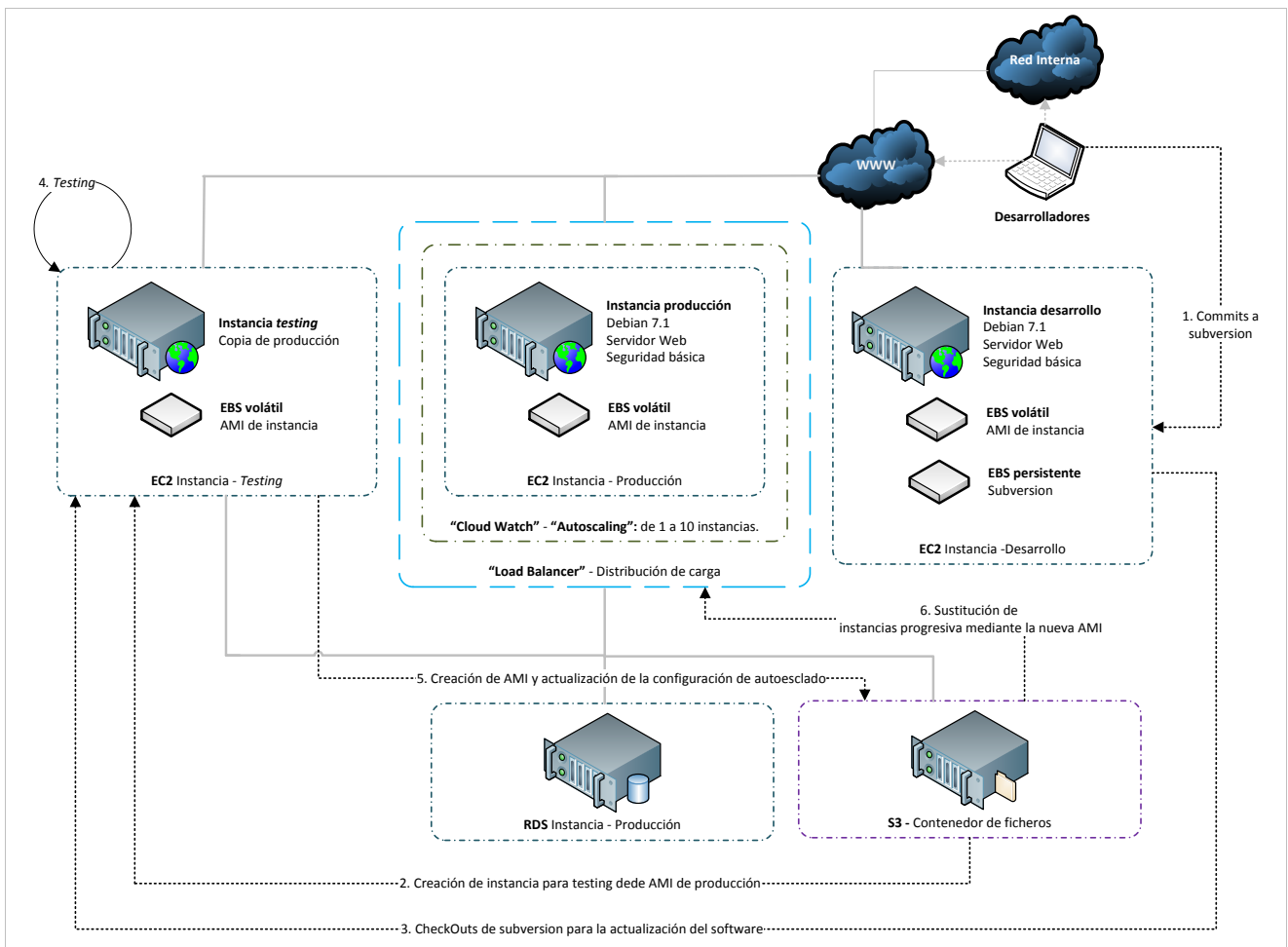


Ilustración 9 - Ciclo de desarrollo y testing tras la migración

Además, todo el proceso ha comportado una actualización de todos los sistemas de producción y desarrollo, por lo que supone un avance en estandarización y seguridad. A parte de que escala de forma totalmente autónoma, algo impensable hasta ahora.

Podemos decir, que en su conjunto, nos hemos podido adaptar en gran parte al modelo de referencia (ver **ilustración 10**), aunque se han producido algunas modificaciones respecto a este:

- Se ha eliminado el servidor de *testing*. Este solo se crea en el momento que sea necesario.
- Se ha eliminado el bastión host. En un futuro se podría implementar.
- Se ha añadido un balanceador de carga.
- Se ha podido implementar la alternancia entre *testing* y producción sin interrupción del servicio.

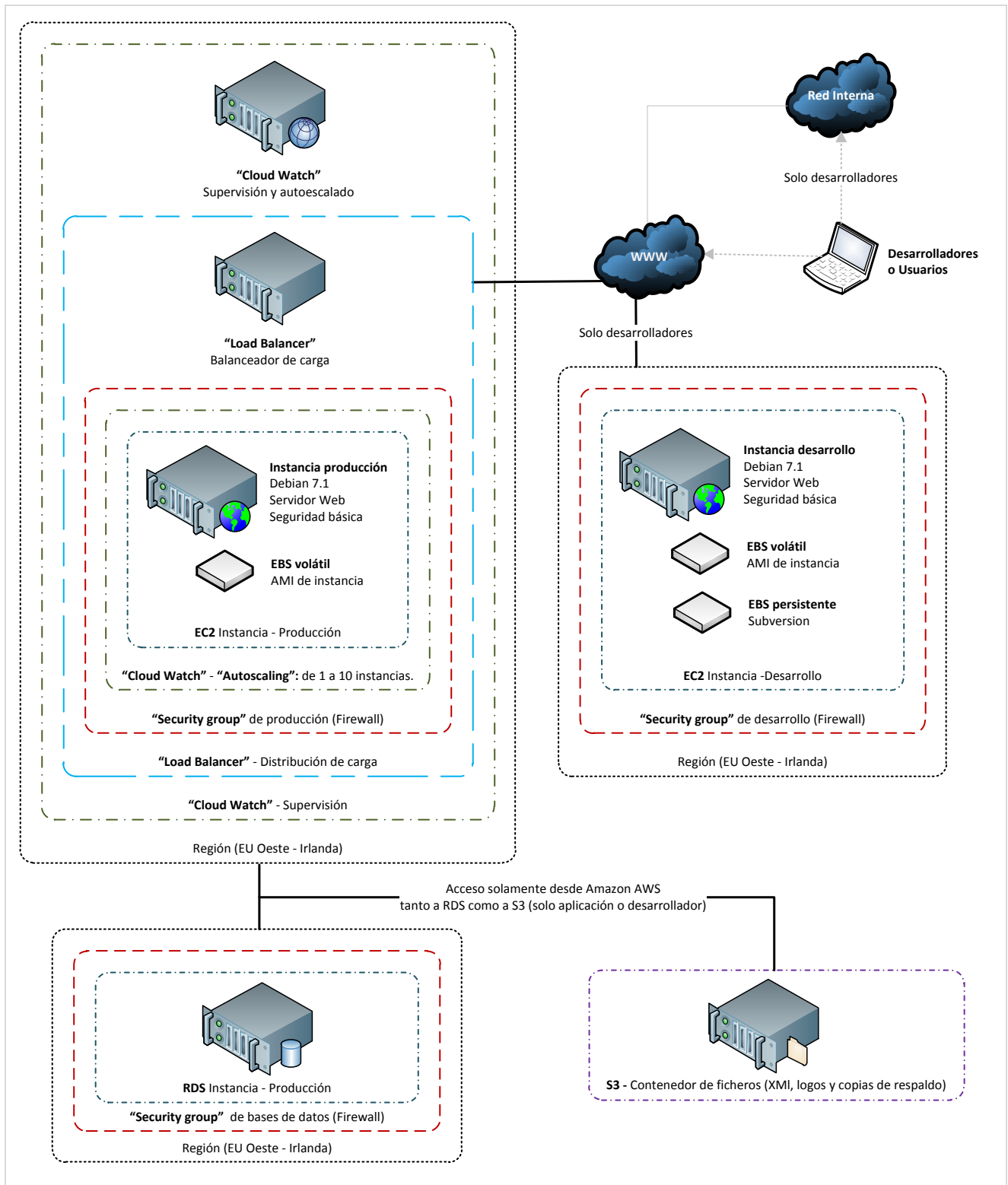


Ilustración 10 - Estructura de red y servicios tras la migración

Los diagramas de esta sección pretenden ser lo más esclarecedores posible, no obstante, debido la complejidad del resultado final pueden resultar difíciles de comprender. Con tal de facilitar su interpretación hay que analizar el conjunto como si fuera una cebolla, donde cada servicio representa una capa de esta que actúa sobre las capas inferiores, que también son otros servicios. De esta forma tenemos que por ejemplo, Cloud-Watch [24] sería la capa superior, que es la que interactúa sobre los demás servicios controlándolos y supervisándolos, y así, hasta llegar a la última capa, que serían las instancias de EC2 [23]. Siendo esta última la única imprescindible sobre la que giran los demás servicios.

5. Conclusiones

Si bien el resultado general de la migración y el proveedor escogido cubren todas nuestras necesidades, la sensación después de migrar todo el sistema es un poco desoladora, ya que pese a ser un sistema autoescalable, una sola instancia que fuera capaz de cubrir la petición de un solo cálculo dispararía el coste de la solución final por encima de los costes actuales. Esto nos lleva a considerar el uso en un futuro de otros servicios más económicos y más potentes junto con algunos de los que ofrece Amazon. Quizá Linode junto con RDS [44] sería una alternativa, aunque hubiera que renunciar al escalado del sistema y potenciar el de instancia.

Resulta difícil de entender como una instancia en Amazon AWS de mejores características (físicas y lógicas) que el servidor privado de producción de RDmes ofrece un rendimiento más pobre que este último, y más teniendo en cuenta que la base de datos y los ficheros dinámicos ya no se encuentran en la propia instancia, factor que debería mejorar el rendimiento de esta. Debido a la limitación de tiempo para el desarrollo del proyecto no se ha podido indagar más en este aspecto y se deja como un trabajo futuro enfocando el problema desde un punto de vista de mala configuración tanto del sistema operativo como del *software* de RDmes.

Otro elemento negativo que se suma al anterior es que, para lograr una implementación que cubriera todos los requisitos hemos tenido que recurrir a servicios que no creíamos que fueran necesarios, como SNS, aumentando de esta manera aún más el coste final. Además, hemos tenido que usar las API's (*Application Programming Interface*) [163] de Amazon con la dificultad que comporta. Algo que no habíamos previsto, ya que se creía que a través de la interfaz web se podrían realizar todas las acciones.

Por otro lado, la migración ha sido algo traumática. Nos hemos encontrado en muchos momentos desamparados sin saber hacia dónde ir o qué hacer, especialmente en el momento de la configuración del autoescalado y las alarmas. Aunque bien es cierto que la documentación es detallada, es fácil perderse entre ella o encontrar referencias a soluciones ya no aplicables. En cuanto al servicio técnico fue una pequeña decepción, ya que cuando tuvimos que recurrir a él, recibimos la siguiente respuesta: "No damos soporte para la migración".

No obstante, todo este proceso nos ha ayudado a ver dónde tenemos nuestras limitaciones, en especial a nivel de aplicación. Este factor nos ayudará de cara a futuras metas y la implantación de una solución más factible.

Por último, en cuando al desarrollo general del proyecto, y por razones ajenas a este, no se ha podido seguir la planificación establecida, lo que ha llevado a posponer algunas tareas previstas en la migración para más adelante. Estas quedan reflejadas en las líneas futuras.

5.1 Líneas futuras

Si bien la línea futura más inmediata es buscar una nueva configuración para la instancia creada que mejore el rendimiento sin aumentar los costes por encima de los actuales o, directamente buscar una nueva solución para la migración, a continuación se presentan algunas mejoras aplicables al resultado final, de las cuales algunas, se tendrían que haber aplicado durante el proceso:

Para la mejora del rendimiento:

Sustitución de Apache HTTP por Nginx:

Nginx es un servidor web que utiliza menos recursos que Apache HTTP para realizar las mismas tareas. Además, las realiza más rápido y está preparado para un mayor número simultáneo de conexiones. No usa *mods* como Apache HTTP para el procesamiento de *scripts*. [165] [166] [167] [168] [169]

Optimización en los cálculos de la aplicación:

Las herramientas para los cálculos de ingeniería fueron programadas hace más de 4 años. Muchas de ellas utilizan funciones y módulos obsoletos, incluso algunas de las aplicaciones no usan *threats* para los cálculos. Además el uso de tablas para extrapolaciones de los resultados podría mejorar sustancialmente el rendimiento.

Uso de dynamoDB para almacenar los XMLs en vez de S3 [27]:

DynamoDB [170] es un servicio de Amazon AWS para el hospedaje de base de datos NoSQL. Con el uso de este servicio no haría falta la creación de ficheros XML's ni implementar toda su gestión, algo compleja.

Implementar una base de datos con este fin simplificaría y reduciría los procesos.

Ampliación de las políticas de autoescalado:

Enfocar el *autoescalado* a RDS [44] y al balanceador de carga, así como también a la propia instancia, puede mejorar el rendimiento del entorno de producción ante picos de uso, ya que si por ejemplo, la infraestructura

tura escala pero el servidor de base de datos se satura ante ciertas peticiones y este no evoluciona, el sistema en general se verá repercutido.

Para la mejora de la seguridad:

Implementación de SSL (Secure Sockets Layer) para producción y desarrollo:

De esta manera se mejoraría la confidencialidad de los datos entre cliente y servidor, no viéndose estos comprometidos ante ataques del tipo *man in the middle*.

Realización de pentesting:

Con tal de garantizar la seguridad ante intrusiones de los sistemas migrados.

Para el supervisado y monitorización:

Creación de nuevas alarmas (notificaciones):

Crear alarmas para RDS [44], Load Balancer [25] y EBS (*Elastic Block Store*) [160] con la finalidad de supervisar la latencia y el rendimiento de los servicios. De esta manera, la empresa tendrá en todo momento constancia del funcionamiento del sistema en los casos especificados.

Creación de alertas para las aplicaciones de monitorización en las propias instancias:

Generar alertas por correo electrónico desde las aplicaciones de supervisión en las instancias, como fail2ban [3], ayudará a la empresa a estar informado en todo momento sobre el funcionamiento de los entornos y posibles ataques contra este.

Bibliografía

- [1] Carlos Manuel Fernández. (2012, Septiembre) Asociación española para la calidad. «*La Norma ISO 27001 del sistema de Gestión de la garantía de la Seguridad de la información*».
[Online]. Último acceso: 09 de Enero de 2014.
http://www.aec.es/c/document_library/get_file?uuid=a89e72de-d92b-47cf-ba5e-5ea421fcbeb4&groupId=10128
- [2] (2010) Netfilter. «*Documentation about the netfilter/iptables project*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.netfilter.org/documentation/>
- [3] (2001, Noviembre) Fail2Ban Wiki. «*FAQ Español*».
[Online]. Último acceso: 09 de Enero de 2014.
http://www.fail2ban.org/wiki/index.php/FAQ_spanish
- [4] (2013, Marzo) Apache HTTPD Wiki. «*FAQ*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://wiki.apache.org/httpd/FAQ>
- [5] Marcelo Rivero and Rivero Marcelo. (2009, Marzo) InfoSpyware. «*¿Qué son los Rootkits?*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.infospyware.com/articulos/que-son-los-rootkits/>
- [6] Franco Bocchio. (2013, Julio) UTN.BA Escuela de posgrado de Buenos Aires - Universidad Tecnología Nacional. «*Estudio comparativo de plataformas cloud computing para arquitecturas SOA*».
[Online]. Último acceso: 09 de Enero de 2014.
http://posgrado.frba.utn.edu.ar/investigacion/especialidades/Bocchio-2013_tf_esp.pdf
- [7] Christine Burns. (2012, Abril) NetworkWorld. «*10 most powerful IaaS companies*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.networkworld.com/supp/2012/enterprise2/040912-ecs-iaas-companies-257611.html?page=1>
- [8] Thoran Rodrigues. (2012, Agosto) TechRepublic U.S. «*Side-by-side comparisons of IaaS service providers*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.techrepublic.com/blog/the-enterprise-cloud/side-by-side-comparisons-of-iaas-service-providers/5717/>
- [9] Stephenie James. (2012, Mayo) Cloud Reviews. «*IaaS Cloud Providers | Best Infrastructure as a Service Providers*».
[Online]. Último acceso: 01 de Enero de 2014.
<http://www.cloudreviews.com/blog/best-iaas-cloud-service-providers>
- [10] Joe Panettieri. (2013, Julio) TalkinCloud. «*Top 100 Cloud Services Providers List 2013: Ranked 10 to 1*».
[Online]. Último acceso: 01 de Enero de 2014.
<http://talkincloud.com/talkin039-cloud-top-100-cloud-services-providers/top-100-cloud-services-providers-list-2013-ranked-0>
- [11] Txema Rodríguez. (2012, Agosto) GenBetaDev. «*Entendiendo la nube: el significado de SaaS, PaaS y IaaS*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.genbetadev.com/programacion-en-la-nube/entendiendo-la-nube-el-significado-de-saas-paas-y-iaas>
- [12] Andrés Hevia. (2011, Noviembre) Xataka On. «*Cuando hablamos de la nube: ¿qué es IaaS, PaaS, SaaS?*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.xatakaon.com/almacenamiento-en-la-nube/cuando-hablamos-de-la-nube-que-es-iaas-paas-saas>
- [13] (2013, Septiembre) ODM (Microsoft Partner Cloud). «*Qué es STaaS, el almacenamiento como servicio*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.odm.es/noticia.asp?p=que-es-staas-el-almacenamiento-como-servicio>
- [14] Jesus Diaz Ruiz. (2013, Septiembre) Revista Cloud Computing. «*STaaS, el almacenamiento como servicio*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.revistacloudcomputing.com/2013/09/staas-el-almacenamiento-como-servicio/>
- [15] Juan Carlos González Martín. (2007, Octubre) Geeks. «*Software As a Service (SaaS): ¿Qué es?*».
[Online]. Último acceso: 04 de Enero de 2014.
<http://geeks.ms/blogs/ciin/archive/2007/10/05/software-as-a-service-sas-191-qu-233-es.aspx>

- [16] Luis Castro. About.com Aprender Internet. «¿Qué es SaaS o software como servicio?». [Online]. Último acceso: 09 de Enero de 2014. <http://aprenderinternet.about.com/od/Glosario/g/Software-como-servicio.htm>
- [17] Luis Castro. About.com Aprender Internet. «¿Qué es PaaS o plataforma como servicio?». [Online]. Último acceso: 09 de Enero de 2014. <http://aprenderinternet.about.com/od/Glosario/g/Plataforma-como-servicio.htm>
- [18] Antonio Ortiz. (2009, Mayo) Error 500. «Infraestructura como servicio (IAAS) en el Cloud computing». [Online]. Último acceso: 09 de Enero de 2014. <http://www.error500.net/infraestructura-como-servicio-iaas-cloud-computing/>
- [19] Margaret Rouse. (2010, Agosto) SearchCloudComputing. «Infrastructure as a Service (IaaS)». [Online]. Último acceso: 09 de Enero de 2014. <http://searchcloudcomputing.techtarget.com/definition/Infrastructure-as-a-Service-IaaS>
- [20] (2013, Septiembre) Wikipedia España. «Amazon Web Services». [Online]. Último acceso: 01 de Enero de 2014. https://es.wikipedia.org/wiki/Amazon_Web_Services
- [21] Amazon Web Services. [Online]. Último acceso: 09 de Enero de 2014. <https://aws.amazon.com/es/>
- [22] AWS Amazon. «Auto Scaling». [Online]. Último acceso: 09 de Enero de 2014. <https://aws.amazon.com/es/autoscaling/>
- [23] AWS Amazon. «Amazon Elastic Compute Cloud (Amazon EC2)». [Online]. Último acceso: 09 de Enero de 2014. <https://aws.amazon.com/es/ec2/>
- [24] AWS Amazon. «Amazon CloudWatch». [Online]. Último acceso: 09 de Enero de 2014. <https://aws.amazon.com/es/cloudwatch/>
- [25] AWS Amazon. «Elastic Load Balancing». [Online]. Último acceso: 01 de Enero de 2014. <https://aws.amazon.com/es/elasticloadbalancing/>
- [26] (2013, Octubre) AWS Amazon documentation. «Amazon Machine Images (AMI)». [Online]. Último acceso: 09 de Enero de 2014. <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AMIs.html>
- [27] AWS Amazon. «Amazon Simple Storage Service (Amazon S3)». [Online]. Último acceso: 09 de Enero de 2014. <https://aws.amazon.com/es/s3/>
- [28] AWS Amazon. «MarketPlace». [Online]. Último acceso: 09 de Enero de 2014. <https://aws.amazon.com/marketplace>
- [29] AWS Amazon. «AWS Multi-Factor Authentication». [Online]. Último acceso: 09 de Enero de 2014. <https://aws.amazon.com/es/mfa/>
- [30] AWS Amazon. «Documentación». [Online]. Último acceso: 09 de Enero de 2014. <https://aws.amazon.com/es/documentation/>
- [31] AWS Amazon. «Contacto». [Online]. Último acceso: 09 de Enero de 2014. <https://aws.amazon.com/es/contact-us/>
- [32] AWS Amazon. «Sample Code & Libraries». [Online]. Último acceso: 09 de Enero de 2014. <https://aws.amazon.com/code>
- [33] AWS Amazon. «Standalone S3 PHP class». [Online]. Último acceso: 09 de Enero de 2014. <https://aws.amazon.com/code/1448>
- [34] AWS Amazon. «AWS Compliance (Certificados)». [Online]. Último acceso: 09 de Enero de 2014. <https://aws.amazon.com/es/compliance/>

- [35] AWS Amazon. «*Centro de seguridad de AWS*». [Online]. Último acceso: 09 de Enero de 2014. <https://aws.amazon.com/es/security/>
- [36] AWS Amazon. «*Media sharing*». [Online]. Último acceso: 09 de Enero de 2014. <https://aws.amazon.com/es/media-sharing/>
- [37] (2013, Octubre) SlideShare AWS Summit 2013. «*AWS Cloud Security*». [Online]. Último acceso: 09 de Enero de 2014. <http://www.slideshare.net/AmazonWebServices/aws-summit-barcelona-security-keynote>
- [38] AWS Amazon. «*AWS PCI DSS Level 1 FAQs*». [Online]. Último acceso: 09 de Enero de 2014. <https://aws.amazon.com/es/compliance/pci-dss-level-1-faqs/>
- [39] AICPA - American Institute for CPAs. «*SOC 1*». [Online]. Último acceso: 09 de Enero de 2014. <http://www.aicpa.org/InterestAreas/FRC/AssuranceAdvisoryServices/Pages/AICPASOC1Report.aspx>
- [40] Secure State. «*DIACAP / DoD 8500*». [Online]. Último acceso: 09 de Enero de 2014. <http://www.securestate.com/Federal/Certification%20and%20%20Accreditation/Pages/DIACAP-DOD8500.aspx>
- [41] Amazon. «*Amazon.com Privacy Notice (Iguales para Amazon AWS)*». [Online]. Último acceso: 09 de Enero de 2014. <http://www.amazon.com/gp/help/customer/display.html?nodeId=468496>
- [42] AWS Amazon. «*AWS Trusted Advisor*». [Online]. Último acceso: 09 de Enero de 2014. <https://aws.amazon.com/es/premiumsupport/trustedadvisor/>
- [43] AWS Amazon. «*FAQ S3 - How do I decide which Region to store my data in?*». [Online]. Último acceso: 09 de Enero de 2014. https://aws.amazon.com/es/s3/faqs/#How_do_I_decide_which_Region_to_store_my_data_in
- [44] AWS Amazon. «*Amazon Relational Database Service (Amazon RDS)*». [Online]. Último acceso: 09 de Enero de 2014. <https://aws.amazon.com/es/rds/>
- [45] (2013, Diciembre) Wikipedia inglesa. «*DigitalOcean*». [Online]. Último acceso: 09 de Enero de 2014. <https://en.wikipedia.org/wiki/DigitalOcean>
- [46] DigitalOcean. «*Simple Cloud Hosting*». [Online]. Último acceso: 09 de Enero de 2014. <https://www.digitalocean.com/>
- [47] (2013, Febrero) DigitalOcean. «*How To Scale Your Infrastructure with DigitalOcean*». [Online]. Último acceso: 09 de Enero de 2014. <https://www.digitalocean.com/community/articles/how-to-scale-your-infrastructure-with-digitalocean>
- [48] DigitalOcean. «*API*». [Online]. Último acceso: 09 de Enero de 2014. <https://developers.digitalocean.com/>
- [49] (2012, Septiembre) DigitalOcean. «*DigitalOcean Backups and Snapshots Explained*». [Online]. Último acceso: 01 de Enero de 2014. <https://www.digitalocean.com/community/articles/digitalocean-backups-and-snapshots-explained>
- [50] DigitalOcean. «*What is your SLA?*». [Online]. Último acceso: 09 de Enero de 2014. <https://www.digitalocean.com/faq>
- [51] DigitalOcean. «*Help & Community*». [Online]. Último acceso: 09 de Enero de 2014. <https://www.digitalocean.com/community>
- [52] (2013, Mayo) DigitalOcean. «*Introducing Two-Factor Authentication*». [Online]. Último acceso: 01 de Enero de 2014. https://www.digitalocean.com/blog_posts/introducing-two-factor-authentication

- [53] (2013, Enero) DigitalOcean. «*How To Use SSH Keys with DigitalOcean Droplets*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.digitalocean.com/community/articles/how-to-use-ssh-keys-with-digitalocean-droplets>
- [54] Digital Ocean. «*Blog*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.digitalocean.com/blog>
- [55] DigitalOcean. «*Chat IRC*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://webchat.freenode.net/?channels=digitalocean&uiou=d4>
- [56] (2013, Mayo) DigitalOcean. «*Terms Of Service*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.digitalocean.com/tos>
- [57] DigitalOcean. «*Features*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.digitalocean.com/features>
- [58] DigitalOcean. «*Security*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.digitalocean.com/security>
- [59] (2013, Abril) DigitalOcean. «*How To Setup Monitoring For Your Droplets*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.digitalocean.com/community/articles/how-to-setup-monitoring-for-your-droplets>
- [60] DigitalOcean. «*Events Calendar*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.digitalocean.com/events-calendar>
- [61] DigitalOcean. «*Referral Program*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.digitalocean.com/referral-program>
- [62] (2013, Noviembre) Wikipedia Gallega. «*Dinahosting*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://gl.wikipedia.org/wiki/Dinahosting>
- [63] DinaHosting. «*Hosting i allotjament web*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://ca.dinahosting.com/>
- [64] DinaHosting. «*API*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://ca.dinahosting.com/api/documentation>
- [65] DinaHosting. «*Els teus backups a dinahosting*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://ca.dinahosting.com/mes-serveis/backups>
- [66] DinaHosting. «*Service Level Agreement (SLA)*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://ca.dinahosting.com/cloud-hosting/sla>
- [67] DinaHosting. «*Hosting Linux*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://ca.dinahosting.com/hosting/hosting-linux>
- [68] DinaHosting. «*Contacto*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://ca.dinahosting.com/sobre-dinahosting/contacte>
- [69] DinaHosting. «*Central de soporte*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.centraldesoporte.com/>
- [70] DinaHosting. «*Protección de Datos Personales*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://ca.dinahosting.com/legal/legal-proteccion-de-datos>
- [71] DinaHosting. «*Centre de dades*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://ca.dinahosting.com/cloud-hosting/centre-de-dades>

- [72] DinaHosting. «*Xarxa*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://ca.dinahosting.com/cloud-hosting/xarxa>
- [73] DinaHosting. «*Hardware que fem servir a RealCloud*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://ca.dinahosting.com/cloud-hosting/hardware>
- [74] DinaHosting. «*Cloud Hosting*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://ca.dinahosting.com/cloud-hosting>
- [75] DinaHosting. «*Sistemes de bases de dades disponibles*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://ca.dinahosting.com/hosting/bases-de-datos>
- [76] (2014, Enero) Wikipedia Español. «*Heroku*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://es.wikipedia.org/wiki/Heroku>
- [77] Heroku. «*Cloud Application Platform*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.heroku.com/>
- [78] Heroku. «*Features*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.heroku.com/features>
- [79] (2013, Octubre) Wikipedia inglesa. «*Joyent*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://en.wikipedia.org/wiki/Joyent>
- [80] Joyent. «*High-Performance Cloud Infrastructure*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.joyent.com/>
- [81] Joyent. «*Features*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.joyent.com/products/private-cloud/features>
- [82] Joyent. «*Manta Storage*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.joyent.com/products/manta>
- [83] Joyent. «*Back-up*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.joyent.com/developers/back-up/>
- [84] Joyent. «*Cloud Hosting Service Level Agreement*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.joyent.com/company/policies/cloud-hosting-service-level-agreement>
- [85] Joyent. «*Duo Security*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.joyent.com/partners/duo-security>
- [86] Joyent. «*Getting Started with Joyent*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.joyent.com/developers/getting-started>
- [87] Joyent. «*Wiki*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://wiki.joyent.com/wiki/display/jpc2/JoyentCloud+Home>
- [88] Joyent. «*Support Tickets - Submit a request*».
[Online]. Último acceso: 09 de Enero de 2014.
https://help.joyent.com/anonymous_requests/new
- [89] Joyent. «*Security and Compliance*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.joyent.com/company/security-and-compliance>
- [90] (2011, Febrero) Europa.eu. «*Síntesis de la legislación de la UE - Protección de los datos personales*».
[Online]. Último acceso: 09 de Enero de 2014.
http://europa.eu/legislation_summaries/information_society/data_protection/l14012_es.htm

- [91] (2013, Enero) Export.gov. «*Welcome to the U.S.-EU & U.S.-Swiss Safe Harbor Frameworks*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://export.gov/safeharbor/>
- [92] Joyent. «*Privacy Policy*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.joyent.com/company/policies/privacy-policy>
- [93] (2013, Octubre) Joyent. «*Terms of Service*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.joyent.com/company/policies/terms-of-service>
- [94] Joyent. «*Customers*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.joyent.com/company/customers>
- [95] Joyent. «*Events*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.joyent.com/company/events>
- [96] Joyent. «*Setting up a MySQL Database*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://wiki.joyent.com/wiki/display/jpc2/Setting+up+a+MySQL+Database>
- [97] Wikipedia inglesa. «*Linode*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://en.wikipedia.org/wiki/Linode>
- [98] Linode. «*Deploy and Manage Linux Virtual Servers in the Linode Cloud*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.linode.com/>
- [99] Linode. «*API*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.linode.com/api/>
- [100] Linode. «*Node Balancer API*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.linode.com/api/nodebalancer>
- [101] Linode. «*NodeBalancer*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.linode.com/nodebalancers/>
- [102] (2012, Junio) Linode. «*Linode clone server call*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://blog.linode.com/2012/06/15/linode-clone-api-call/>
- [103] Linode. «*Backup Service*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.linode.com/backups/>
- [104] Linode. «*What is your SLA?*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.linode.com/faq.cfm#what-is-your-sla>
- [105] Linode. «*Which distributions do you offer?*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.linode.com/faq.cfm#which-distributions-do-you-offer>
- [106] (2013, Mayo) Linode. «*Linode Manager Two-Step Authentication*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://blog.linode.com/2013/05/02/linode-manager-two-step-auth/>
- [107] Linode. «*Wiki*».
[Online]. Último acceso: 09 de Enero de 2014.
https://www.linode.com/wiki/index.php/Main_Page
- [108] Linode. «*IRC Chat*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.linode.com/irc/>
- [109] Linode. «*Forum*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://forum.linode.com/>

- [110] Linode. «*About Us*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.linode.com/about/>
- [111] Dan Goodin. (2012, Marzo) ArsTechnica. «*Bitcoins worth \$228,000 stolen from customers of hacked Webhost*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://arstechnica.com/business/2012/03/bitcoins-worth-228000-stolen-from-customers-of-hacked-webhost/>
- [112] Linode. «*Privacy Policy*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.linode.com/privacy.cfm>
- [113] Matthew Cone. (2012, Agosto) Linode Library. «*Monitoring and Maintaining Your Server*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://library.linode.com/monitoring-and-maintaining>
- [114] Linode. «*Blog*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://blog.linode.com/>
- [115] Linode. «*Terms of Service*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.linode.com/tos.cfm>
- [116] (2013, Mayo) Wikipedia española. «*OpenShift*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://es.wikipedia.org/wiki/OpenShift>
- [117] OpenShift. «*The Open Hybrid Cloud Application Platform by Red Hat*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://www.openshift.com/>
- [118] Wikipedia inglesa. «*Rackspace*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://en.wikipedia.org/wiki/Rackspace>
- [119] Rackspace. «*The open cloud company*».
[Online]. Último acceso: 09 de Enero de 2014.
http://www.rackspace.com/es/?CMP=GEO_US
- [120] Sanjay Sohoni. (2013, Noviembre) Rackspace. «*Easily Scale Your Cloud With Rackspace Auto Scale*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.rackspace.com/blog/easily-scale-your-cloud-with-rackspace-auto-scale/>
- [121] Rackspace. «*Load Balancers*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.rackspace.com/es/cloud/load-balancing/>
- [122] Rackspace. «*Block Storage*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.rackspace.com/es/cloud/block-storage/>
- [123] Rackspace. «*Cloud Backup*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.rackspace.com/es/cloud/backup/>
- [124] Rackspace. «*Información SLA para los servicios*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.rackspace.com/es/information/legal/cloud/sla>
- [125] Rackspace. «*Features*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.rackspace.com/cloud/servers/features/>
- [126] Rackspace. «*Support*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://support.rackspace.com/>
- [127] Rackspace. «*Help & Support links*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.rackspace.com/support/>

- [128] Rackspace. «*Rack Stories - Customers*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://stories.rackspace.com/customers>
- [129] Rackspace. «*Security - Security is a partnership*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.rackspace.com/security/>
- [130] Rackspace. «*Declaración de privacidad*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.rackspace.com/es/information/legal/privacystatement>
- [131] Rackspace. «*Rackspace security management*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.rackspace.com/security/management/>
- [132] Rackspace. «*Show Upcoming Events*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.rackspace.com/events/>
- [133] Rackspace. «*Blog*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.rackspace.com/blog/>
- [134] Rackspace. «*Stories*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://stories.rackspace.com/stories>
- [135] Rackspace. «*Cloud Terms of Service*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.rackspace.com/information/legal/cloud/tos>
- [136] Rackspace. «*The high-performance MySQL database on the cloud*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.rackspace.com/cloud/databases/>
- [137] (2013, Diciembre) Wikipedia española. «*VMware*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://es.wikipedia.org/wiki/VMware>
- [138] VMware. «*vCloud*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.vmware.com/products/vcloud-suite/>
- [139] VMware. «*Noticias de VMware*».
[Online]. Último acceso: 09 de Enero de 2014.
http://www.vmware.com/es/company/news/releases/NP_vCloud_Hybrid_Service_ES.html
- [140] (2013, Diciembre) Wikipedia española. «*Windows Azure*».
[Online]. Último acceso: 09 de Enero de 2014.
https://es.wikipedia.org/wiki/Windows_Azure
- [141] Windows Azure. «*Plataforma en la nube de Microsoft*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.windowsazure.com/es-es/>
- [142] Windows Azure. «*How to Scale an Application*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.windowsazure.com/en-us/documentation/articles/cloud-services-how-to-scale/>
- [143] Windows Azure. «*Load Balancing Virtual Machines*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.windowsazure.com/en-us/documentation/articles/load-balance-virtual-machines/>
- [144] Windows Azure. «*Almacenamiento, copia de seguridad y recuperación*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.windowsazure.com/es-es/solutions/storage-backup-recovery/>
- [145] Windows Azure. «*Servicios en la nube*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.windowsazure.com/es-es/services/cloud-services/>
- [146] Windows Azure. «*Multi-Factor Authentication*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.windowsazure.com/en-us/documentation/services/multi-factor-authentication/?fb=es-es>

- [147] Windows Azure. «*How to Use SSH with Linux on Windows Azure*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.windowsazure.com/en-us/documentation/articles/linux-use-ssh-key/?fb=es-es>
- [148] Windows Azure. «*Documentation Center*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.windowsazure.com/en-us/documentation/?fb=es-es>
- [149] Windows Azure. «*Foros*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.windowsazure.com/es-es/support/forums/>
- [150] Windows Azure. «*Planes de soporte técnico de Windows Azure*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.windowsazure.com/es-es/support/plans/>
- [151] (2013, Marzo) Windows Azure. «*Declaración de privacidad de Windows Azure*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.windowsazure.com/es-es/support/legal/privacy-statement/>
- [152] (2013, Octubre) Windows Azure. «*Trust Center - Security*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.windowsazure.com/en-us/support/trust-center/security/>
- [153] Windows Azure. «*Technical Overview of the Security Features in the Windows Azure Platform*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.windowsazure.com/en-us/support/legal/security-overview/>
- [154] Windows Azure. «*Compliance*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.windowsazure.com/en-us/support/trust-center/compliance/>
- [155] Windows Azure. «*How to Monitor Cloud Services*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.windowsazure.com/en-us/documentation/articles/cloud-services-how-to-monitor/>
- [156] Windows Azure. «*Educators*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.windowsazure.com/en-us/community/education/?fb=es-es>
- [157] Windows Azure. «*Case Studies*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.windowsazure.com/en-us/overview/case-studies/?fb=es-es>
- [158] Channel9 MSDN. «*Windows Azure Media Services Tutorials*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://channel9.msdn.com/Series/Windows-Azure-Media-Services-Tutorials>
- [159] Windows Azure. «*Eventos*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://www.windowsazure.com/es-es/community/events/>
- [160] AWS Amazon. «*Amazon Elastic Block Store (Amazon EBS)*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://aws.amazon.com/es/ebs/>
- [161] AWS Amazon. «*Elastic IP Addresses (EIP)*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/elastic-ip-addresses-eip.html>
- [162] AWS Amazon. «*AWS SDK para PHP*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://aws.amazon.com/es/sdkforphp/>
- [163] (2013, Septiembre) AWS Amazon. «*Amazon EC2 API Tools*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://aws.amazon.com/developertools/351>
- [164] (2013, Diciembre) Wikipedia inglesa. «*Snapshot (computer storage)*».
[Online]. Último acceso: 09 de Enero de 2014.
[https://en.wikipedia.org/wiki/Snapshot_\(computer_storage\)](https://en.wikipedia.org/wiki/Snapshot_(computer_storage))
- [165] Rodrigo Zamora. (2012, Septiembre) Web++. «*Apache 2.4 vs Nginx en un entorno mas real*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://webplusplus.blogspot.com.es/2012/09/apache-24-vs-nginx-en-un-entorno-mas.html>

- [166] (2013, Noviembre) Lite Speed. «*New Benchmarks! LiteSpeed vs. Apache vs. nginx for Static Content*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://blog.litespeedtech.com/2013/11/12/new-benchmarks-litespeed-vs-apache-vs-nginx-for-static-content/>
- [167] (2013, Febrero) StackOverflow. «*PHP processing speed apache 2.4 mpm-prefork mod_php 5.4 vs nginx 1.2.x PHP-FPM 5.4*».
[Online]. Último acceso: 09 de Enero de 2014.
<https://stackoverflow.com/questions/14983276/php-processing-speed-apache-2-4-mpm-prefork-mod-php-5-4-vs-nginx-1-2-x-php-fpm-5>
- [168] Armin Čoralić. (2013, Mayo) Coralic Blog. «*Apache 2.4 vs Nginx 1.4*».
[Online]. Último acceso: 09 de Enero de 2014.
<http://blog.coralic.nl/2013/05/12/apache-2-4-vs-nginx-1-4/>
- [169] Kevin Schroeder. (2013, Enero) ESchrade. «*Why is FastCGI /w Nginx so much faster than Apache /w mod_php?*».
[Online]. Último acceso: 09 de Enero de 2014.
http://www.eschrade.com/page/why-is-fastcgi-w-nginx-so-much-faster-than-apache-w-mod_php/
- [170] AWS Amazon. «*Amazon DynamoDB*».
[Online]. Último acceso: 2014 de Enero de 2014.
<https://aws.amazon.com/es/dynamodb/>

ANEXO 01

GUÍAS DE MIGRACIÓN

1. Guía de migración para producción

En esta sección se detallan los pasos llevados a cabo para la puesta en marcha de la plataforma Omnibus® en Amazon AWS teniendo en cuenta que la persona conoce los servicios prestados por el proveedor, si este no fuera el caso, es recomendable realizar primero una lectura del **Anexo 01**, ya que este contiene un conjunto de manuales sobre el uso básico de los servicios que ofrece el proveedor.

Las guías aquí presentes abarcan desde la creación de los servidores hasta la instalación y configuración del *software* necesario.

1.1 Creación y configuración del entorno

Para la creación del entorno seguimos los siguientes pasos:

1. En Amazon AWS creamos una instancia m1.small junto a su clave a partir de una AMI (*Amazon Machine Image*) de Debian 7 (ami-954559e1).
2. Asociamos una IP estática a la instancia mediante la interfaz de EC2.
3. En “*Security Groups*” seleccionamos el grupo asociado a la instancia y en la pestaña de “*outbound*” eliminamos la regla que viene por defecto y añadimos el puerto HTTP 80.
4. Ahora, en nuestro ordenador local procederemos de la siguiente forma:

```
# Creamos la carpeta .ssh en nuestro directorio local si esta no existe
$ mkdir ~/.ssh
# Copiamos la clave descargada asociada a la instancia a la carpeta .ssh de nuestro usuario
$ cp -r ~/Downloads/miclave.pem ~/.ssh
# Le cambiamos los permisos a solo lectura
$ sudo chmod 400 miclave.pem
# Si el usuario dueño es root lo cambiamos por el nuestro
$ sudo chown [mi_usuario]:[mi_grupo] miclave.pem
# creamos un fichero de configuración para facilitar la conexión
$ sudo nano .ssh/config
# En el introducimos:
# -- Host aws.host.production
# -- Hostname [IP_estatica_asociada_a_la_instancia]
# -- User admin //Por defecto en la AMI de Debian 7.1
# -- IdentityFile "~/ssh/[nombre_del_fichero].pem"
# Procedemos a conectarnos
$ ssh aws.host.production
```

5. Una vez conectados procedemos a:

```
# Actualizar el sistema
$ sudo apt-get update
$ sudo apt-get dist-upgrade

# Configurar el ssh
# -- Nos añadimos como usuario al sistema para la conexión ssh
$ adduser [usuario]
# -- Nos añadimos al sudoers: [usuario] ALL=(ALL) ALL
$ sudo nano /etc/sudoers
# Nos copiamos la clave del usuario admin
$ sudo cp ~/.ssh /home/[usuario]/
# Modificamos los permisos
$ sudo chown -R [usuario]:[usuario]/home/[usuario]/.ssh/
# Nos salimos
$ exit
```

6. Una vez fuera comprobamos si funciona:

```
# Modificamos el fichero config de .ssh:
$ sudo nano .ssh/config
# En el introducimos:
# -- Host aws.host.production.[usuario]
# -- Hostname [IP_estatica_asociada_a_la_instancia]
# -- User [usuario]
# -- IdentityFile "~/ssh/[nombre_del_fichero].pem"
# Probamos de acceder:
$ ssh aws.host.production.[usuario]
# Si nuestro usuario puede acceder al sistema y usar el comando sudo todo ha ido
```

```
# Correctamente y podemos seguir con los puntos marcados. Si este no fuera el caso
# hay que lograr cambiar el usuario por defecto para ingresar por ssh a la máquina
# por uno de nuestro control.
```

7. Si los pasos anteriores han funcionado realizamos las siguientes acciones:

```
# Quitamos el acceso por ssh al usuario por defecto admin.
$ sudo rm -rf /home/admin/.ssh
# Miramos que no se pueda acceder con usuario root a la máquina
$ sudo nano /etc/ssh/sshd_config
# Miramos que la línea sigueinte este tal cual se describe aquí:
# -- PermitRootLogin no
# Aprovechamos y cambiamos el puerto por defecto. En el mismo fichero:
# -- Port [nuevo_puerto]
# Reiniciamos el servicio para que tenga efecto
$ sudo /etc/init.d/ssh restart
```
8. Sin salirnos de la instancia nos dirigimos a Amazon AWS y en el “Security Gropus” de la instancia tanto en “Inbound” como en “Outbound” añadimos el nuevo puerto para SSH.
9. Mediante una nueva consola probamos si podemos acceder. Para ello:

```
# Modificamos el fichero config de .ssh:
$ sudo nano .ssh/config
# En el introducimos:
# -- Host aws.host.production.[usuario].port
# -- Hostname [IP_estatica_asociada_a_la_instancia]
# -- User [usuario]
# -- IdentityFile "~/ssh/[nombre_del_fichero].pem"
# -- Port [nuevo_puerto]
# Probamos de acceder:
$ ssh aws.host.production.[usuario].port
#Si accedemos continuamos con el proceso, sino hay que investigar porque no funciona el cambio de puerto. En este paso tambien es recomendable ver si con el usuario admin aún se puede acceder, y de la misma forma, con el usuario root.
```
10. Si todo ha ido correctamente podemos cerrar la conexión abierta previamente y en el “Security Gropus” de nuestra instancia en EC2 eliminar de “Inbound” el puerto SSH (Secure Shell) 22.

Resumen: En este punto nos hemos de encontrar con una conexión SSH abierta con la instancia, realizada a partir de un usuario y puerto personalizado: con cuatro reglas en el grupo de seguridad de EC2: puerto personalizado para SSH + HTTP80 tanto en “inbound” como “outbound”; con que ni el usuario por defecto admin ni root puede acceder al servicio; y con que el sistema está totalmente actualizado.

11. Con la conexión abierta procedemos a configurar el fail2ban:

```
# Instalamos la aplicación
$ sudo install fail2ban
# Realizamos la configuración. Para ello:
# -- Replicamos el fichero de configuración
$ sudo cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.local
# -- Modificamos la configuración
# -- -- backend = auto por backend = polling
$ sudo nano /etc/fail2ban/jail.conf
# -- -- backend = auto por backend = polling
$ sudo nano /etc/fail2ban/jail.local
# -- -- backend = auto por backend = polling
# -- -- en [ssh] port = ssh por port = [nuevo_puerto]
#reiniziamos fail2ban para que tengan efecto los cambios
$ sudo /etc/init.d/fail2ban restart
# Llegados este punto la ideal sería probar que funciona, para ello en el fichero jail.local ponemos en el tiempo de baneo (bantime) un tiempo bajo y reiniciamos el servicio. En nuestro PC modificamos el usuario de nuestro fichero config de ssh y probamos a conectarnos reiteradas veces. En caso de que nos bane todo funciona correctamente, así que esperamos el tiempo de ban y volvemos a poner el fichero config tal como estaba. Tras pasar el tiempo entramos a la instancia, reconfiguramos el fail2ban con un tiempo para ban más acorde y reiniciamos el servicio. En caso de que no funcione sería cuestión de mirar que está pasando.
```
12. Introducimos las variables de entorno necesarias para la instalación de los paquetes de Omnibus®.

```
# Modificamos el inicio del shell
$ sudo nano /etc/bash.bashrc
# Añadimos las siguientes líneas
```

```

# -- export RD_USER_INSTALL_ROOT=/home/${LOGNAME}/INSTALL
# #definition of the rd install path for user installations
# -- export PKG_CONFIG_PATH=${PKG_CONFIG_PATH}:/usr/lib/pkgconfig:$RD_USER_INSTALL_ROOT/lib/pkgconfig
# #pkgconfig path for user installations
# -- export PATH=$PATH:${RD_USER_INSTALL_ROOT}/bin
# #bin path for user installations

```

Estos serían todos los pasos que se han de realizar para preparar el entorno si hemos escogido la AMI (*Amazon Machine Image*) correcta mencionada. Los demás sistemas de seguridad en gran parte vienen proporcionados por Amazon AWS a través de sus servicios.

Sería conveniente que de cara a los siguientes pasos la instancia tenga un tamaño mínimo de EBS de 30GB.

1.2 Instalación de los paquetes necesarios

Los paquetes necesarios que hay instalar para el funcionamiento correcto de la web y los cálculos son los siguientes:

```

$ sudo apt-get install automake
$ sudo apt-get install make
$ sudo apt-get install libtool (Suele instalar el gcc tambien)
$ sudo apt-get install gcc
$ sudo apt-get install python-dev python-all-dev python-numpy
$ sudo apt-get install python3 python3-dev python3-all-dev python3-numpy
$ sudo apt-get install zlib1g zlib1g-dev
$ sudo apt-get install libffi-dev
$ sudo apt-get install glade libglade2-dev
$ sudo apt-get install doxygen
$ sudo apt-get install python-lxml python3-lxml python-xlwt
$ sudo apt-get install libatk1.0-0 libatk1.0-dev
$ sudo apt-get install libatspi2.0-dev
$ sudo apt-get install libcairo2-dev
$ sudo apt-get install libpango1.0-dev
$ sudo apt-get install libgdk-pixbuf2.0-dev
$ sudo apt-get install libgsl0-dev
$ sudo apt-get install gfortran gobjc++ gobjc
$ sudo apt-get install libpopt0 libpopt-dev
$ sudo apt-get install swig
$ sudo apt-get install lam-dev
$ sudo apt-get install libxml2 libxslt1.1 libxslt1-dev python-libxslt1
$ sudo apt-get install gnuplot gnuplot-doc
$ sudo apt-get install libcppunit-doc libcppunit-dev
$ sudo apt-get install libglib2.0-0 libglib2.0-dev
$ sudo apt-get install libgtk2.0-0 libgtk2.0-dev
$ sudo apt-get install texlive texlive-latex3
$ sudo apt-get install php5 php5-gmp php5-gd php5-mysql libapache2-mod-php5 php-gettext
$ sudo apt-get install mysql-client mysql-server python-mysql.connector libmysqlcppconn-dev
$ sudo apt-get install libmysqlcppconn5
$ sudo apt-get install gle-graphics
$ sudo apt-get install python-gnuplot
$ sudo apt-get install liblog4cplus-dev liblog4cplus-dbg liblog4cplus-1.0-4
$ sudo apt-get install python-gtk2-dev
$ sudo apt-get install libmysql++-dev python-mysqldb
$ sudo apt-get install xsltproc
$ sudo apt-get install gtk-doc-tools doxygen-doc
$ sudo apt-get install latex2html doxygen-doc texlive-full graphviz dot2tex biblatex biblatex-dw
$ sudo apt-get install bibtool latexml bibtex2html latex2rtf
$ sudo apt-get install cabextract
$ sudo apt-get install gtk2-engines-murrine gtk2-engines-pixbuf

```

También hay que instalar el pyXML (<http://pyxml.sourceforge.net/topics/download.html>):

```

# Descargar
$ wget [ultima_version]
# Descomprimir
$ tar xvf [file]
# Instalar
$ cd [file]
$ sudo python setup.py install

```

```
# Editar el fichero ParsedAbbreviatedAbsolutePath.py para que funcione
$ sudo nano /usr/local/lib/python2.7/dist-packages/_xmlplus/xpath/ParsedAbbreviatedAbsolutePath.py
# -- Renombrar as por as_

Instalar las fuentes de Microsoft (http://packages.debian.org/wheezy/ttf-mscorefonts-installer)
# Descarga
$ wget http://ftp.de.debian.org/debian/pool/contrib/m/msttcorefonts/ttf-mscorefonts-
  installer_3.4+nmu1_all.deb
# Instalación
$ sudo dpkg -i ttf-mscorefonts-installer_3.4+nmu1_all.deb
# Cambios de permisos
$ sudo chmod 0777 /usr/share/fonts/truetype/msttcorefonts/Arial.ttf
# Accedemos a la carpeta
$ cd /usr/share/fonts/truetype/
# Creamos un enlace simbolico
$ sudo ln -s msttcorefonts/Arial.ttf arial.ttf
```

Configurar el locale:

```
# Reconfigure
$ sudo dpkg-reconfigure locales
# Presinamos espacio en all, posteriormente presionamos enter en all y seguimos las instrucciones.
# El local por defecto ha de ser en_US.utf8.
```

1.3 Instalación de los paquetes de cálculo

Para ello primero hemos de subir los paquetes mediante SCP al servidor:

```
$ scp packets.tar aws.host.production:/tmp
```

O mediante un *checkout* del Subversion de producción:

```
$ svn co http://[IP]/svn/software/branches/2.6.0/ .
```

Una vez en el servidor procedemos a instalar los paquetes de la siguiente forma:

```
$ ./autogen.sh
$ make;
$ sudo make install;
```

En el siguiente **orden**: `rdutils rdtex rddbase rdclib-<VERSION> rddataprocs rdsolsys rdlib-<VERSION> rdsqldb rdsoldes rdsmi ; -<VERSION>`. *Se le ha de añadir al nombre de la carpeta antes de instalar, ya que si no puede dar error durante la instalación. ej: rdclib-1.0.25*

Una vez instalado el `rdclib` y el `rdlib` se ha de realizar una prueba de funcionamiento para ver si las instalaciones han ido correctamente:

```
$ rdexamples -S
$ rdexamples -t --add-list=<NUMERO>
```

1.4 Instalación de la interfaz web y configuración

Para su instalación procederemos de la siguiente forma:

```
# Nos situamos en la carpeta del servidor web
$ cd /srv/www
# Creamos la carpeta que contendrá los ficheros relacionados con la aplicación
# Accedemos a ella
$ sudo mkdir omnibus
$ cd omnibus
# Creamos la carpeta que contendrá la aplicación web
# Accedemos a ella
$ sudo mkdir htdocs
$ cd htdocs
# Descargamos desde el svn la aplicación. También podemos realizar un scp o un mv desde
# otra ubicación.
$ svn co http://sol/svn/software/branches/2.6.0/solarium/ .
# Nos situamos en la carpeta omnibus
$ cd ../htdocs
# Creamos las carpetas para los ficheros dinámicos y de configuración
$ sudo mkdir profiles
$ sudo mkdir logos //Dentro hay que situar default.jpg -> en solarium/extras
$ sudo mkdir logs
```

```

$ sudo mkdir xmls
# Cambiamos los permisos
$ sudo chown -R www-data:www-data logos logs xmls
# En profiles hay que copiar el fichero omnibus.xml que se encuentra en solarium/extras
$ cd profiles
# Cambiamos los permisos
$ sudo chown -R www-data:www-data omnibus.xml
# Nos situamos en htdocs
$ cd ../htdocs
# Cambiamos los permisos de la wiki
$ sudo chown -R www-data:www-data wiki/
# Si no nos deja porque no está en el grupo www-data
# (para ver en qué grupo esta sudo groups www-data)
# (para cambiar grupo y usuario: nano /etc/apache2/envvars)
$ sudo adduser www-data www-data

```

De esta manera quedaría configurada la estructura de la aplicación web. No obstante, para una instalación de producción en htdocs realizar:

```

$ sudo make build
$ sudo make dist
$ sudo tar xvf omnibus.tar .
$ sudo rm -rf omnibus.tar
$ sudo chown -R www-data:www-data wiki/

```

Ahora configuramos el PHP para que interprete los *shortTags*. Para ello:

```

# Editamos el fichero php.ini
$ sudo nano /etc/php5/apache2/php.ini
# -- Modificamos short_open_tag = off por short_open_tag = On

```

Por último nos quedaría configurar el servidor web:

```

# Añadimos un nuevo sitio web
$ sudo touch /etc/apache2/sites-available/omnibus
# Editamos el sitio web
$ sudo nano /etc/apache2/sites-available/omnibus
# Para cada virtual host: www, es, en, ca.
# -- <VirtualHost *:80>
# --     ServerName [es | www | ca | en].[IP | dominio]
# --     ServerAlias [es | null | ca | en].[IP | dominio]
# --     HostnameLookups Off
# --     UseCanonicalName On
# --     ServerSignature On
# --     DocumentRoot /srv/www/omnibus/htdocs
# --     <Directory "/srv/www/omnibus/htdocs">
# --         Options -Indexes
# --         AllowOverride All
# --         SetEnv ENVIRONMENT "PRODUCTION"
# --         Order allow,deny
# --         Allow from all
# --     </Directory>
# -- </VirtualHost>
# Habilitamos el sitio
$ cd /etc/apache2/sites-enabled
$ sudo ln -s ../sites-available/omnibus ./omnibus

```

Ahora habilitamos los *mods* necesarios:

```

$ cd /etc/apache2/mods-enabled
$ sudo ln -s ../mods-available/headers.load ./headers.load
$ sudo ln -s ../mods-available/rewrite.load ./rewrite.load

```

Añadimos los *handlers* para los *.js y los *.inc:

```

$ sudo nano /etc/apache2/mods-available/php5.conf
# -- <IfModule mod_php5.c>
# --     # If php is turned on, we respect .php and .phps files.
# --     AddHandler application/x-httpd-php .js
# --     AddHandler application/x-httpd-php .inc
# --     # Since most users will want index.php to work we
# --     # also automatically enable index.php

```



```
# --      <IfModule mod_dir.c>
# --          DirectoryIndex index.html index.php
# --      </IfModule>
# -- </IfModule>
```

Hacemos que el /tmp del sistema sea un “saco roto” (*Sticky bit*):

```
$ sudo chown -R root:root /tmp
$ sudo chmod -R 1777 /tmp
```

Reiniciamos Apache:

```
$ sudo /etc/init.d/apache2 restart
# Si al reiniciar nos da el siguiente error:
# -- Stopping web server apache2
# -- apache2: Could not reliably determine the server's fully qualified domain name, using
# -- 127.0.1.1 ServerName ... waiting
# Ejecutamos:
$ sudo sh -c 'echo "ServerName localhost" >> /etc/apache2/conf.d/name'
$ sudo service apache2 restart
```

Soluciones a errores comunes

/usr/include/c++/4.7/cstdlib:112:11: error: ':: <function C>' has not been declared

Poner #include <cstdlib> en los ficheros que da el error.

SWIG Versions: 1.30.222 > 2.XX.XX

Modificar el configure.ac en la carpeta root del paquete donde da el error:

Sustituir AC_PROG_SWIG(1.30.222) por AC_PROG_SWIG

/usr/bin/ld: cannot find -llammpi++

Añadir en el autogen.sh: export LDFLAGS="-L/usr/lib/lam/lib"

Python no encuentra el lam

En <paquete>/include/common.mk: Añadir -I/usr/include/mpi

Guía de cosas a validar

1. Se accede correctamente a la interfaz.
2. Los cálculos, los *plots* y los *reports* funcionan.
3. Se ven los *captchas*.
4. Se envían correctamente los correos electrónicos de soporte.
5. Funcionan los logos.
6. Funcionan las traducciones.
7. Que no se visualizan los directorios al acceder por la barra del navegador.
8. Que no se muestre el contenido de los ficheros *include* o *js*.
9. Funcionan los subdominios en, ca y es.
10. Funciona la wiki.

1.5 Instalación de la base de datos en RDS

Para instalar las bases de datos en el servicio de Amazon RDS hemos de proceder de la siguiente forma:

1. Crear la base de datos en el servicio de Amazon RDS con la última versión de MySQL y procurando que sea *multiAZ* y privada (no se pueda acceder desde Internet).
2. Abrimos el puerto para la base de datos en el grupo de seguridad de esta y en nuestra instancia. Es recomendable en este instante meterla en un DNS, puesto que cada vez que la reiniciamos cambia de dirección.
3. Una vez creada comprobamos que funcione desde de la instancia donde está la aplicación:

```
# Conectamos a nuestra instancia
$ ssh [IP_instancia]
# Comprobamos que entra al mysql
$ mysql -h [endpoint_adress] -u[user-creación] -p[password-creación]
```
4. Una vez dentro procedemos a:

```
# Crear el usuario para la aplicación
mysql > CREATE USER '[user-omnilus]'@'%' IDENTIFIED BY '[password-omnilus]';
mysql > FLUSH PRIVILEGES;
# Nos salimos de la base de datos
mysql > Exit
# Comprobamos que funcione
```

```
$ mysql -h [endpoint_adress] -u[user-omnilus] -p[password-omnilus]
# Nos salimos de la base de datos
mysql > Exit
```

- Entramos a la interfaz de RDS en Amazon y seguimos los siguientes pasos con la finalidad de evitar el siguiente error:

```
#1419 – You do not have the SUPER privilege and binary logging is enabled (you *might* want to use the less safe log_bin_trust_function_creators variable
```

- Abrimos en la parte izquierda de la interfaz de RDS la opción “Parameter Groups”.
- Creamos uno nuevo. En el dialogo que se abre seleccionamos la familia de MySQL con la que creamos la instancia, introducimos el nombre y confirmamos.
- Seleccionamos el “Parameter Group” creado y le damos a “Edit Parameters”.
- En el buscado de la lista de parámetro introducimos ‘log_bin_trust_function_creators’. Se nos mostrará la opción y en “value” ponemos 1.
- Guardamos los cambios.
- Nos dirigimos a la base de datos creada en la interfaz de Amazon y le clicamos derecho “Modify”.
- Le asociamos en “Parameter Group” el grupo creado, habilitamos “Apply Immediately” y clicamos en continuar y confirmar los cambios.
- Nos fijamos en “Configuration Details” de la instancia se ha aplicado el nuevo grupo.
- Si es así y pone *reboot*, volvemos a clicar derecho en la instancia y presionamos sobre la opción “reboot”.
- Tras el reinicio el problema ya no debería de existir.

- Ahora realizamos los siguientes cambios en el paquete rdsqsql para poder instalar la base de datos:

```
# Nos situamos en el path de los scripts sql
$ cd rdsqsql/share
# Editamos el Makefile
$ nano Makefile
# En la opción :install modificamos la sentencia mysql añadiendo
# -- -h [endpoint_adress]
# Debería quedar @mysql -h [endpoint_adress] -u[user-creación] -p[password-creación]
# < ./utils/mysql_users.sql
# Editamos el Makefile.inc
$ nano Makefile.inc
# Modificamos la variable MYSQL_CMD de tal forma que nos quede:
# MYSQL_CMD = mysql -h [endpoint_adress] -u[user-creación] -p[password-creación]
```

- A partir de aquí podemos intentar ya realizar un `make install` e ir solucionando los pequeños errores que vayan saliendo. Los más comunes quedan documentados tras este apartado.

```
# Realizamos la instalación
$ make install
# Cualquier error consultar el apartado Errores comunes y soluciones de esta sección
```

En este punto tendríamos la base de datos montada en RDS. Para una migración de un sistema ya existente bastaría con realizar un *dump* de la base de datos y realizar el siguiente comando:

```
$ mysql -h [endpoint_adress] -u[user-creación] -p[password-creación] < [dump].sql
```

Errores comunes y soluciones

ERROR 1227 (42000) at line --- : Access denied; you need (at least one of) the SUPER privilege(s) for this operation o Access denied; you need (at least one of) the SUPER privilege(s) for this operation.

Este problema surge cuando los tigger para la base de datos están mal definidos o no se adaptan a la nueva situación. Basta con ir a la línea que nos indica el error y sustituir `DEFINER=[user]@[host]` por `DEFINER=[correctUser]@[correctIP]`.

Lo más probable es que este problema se de en la creación de las vistas definidas en el *scheme* de solarium. Para solucionarlo haremos lo siguiente:

```
# Editamos el schema
$ nano rdsqsql/share/solarium/schema.sql
# Nos digirimos a las dos ultims consultas
# -- Sustituimos en ambas el DEFINER=[]@localhost por DEFINER=`[user-creación]`@`%`
```

ERROR de mysql syntax en el momento de crear los usuarios y permisos

En este caso el error viene dado por la especificación del GRANT. Bastara con realizar los siguientes pasos:

```
# Editamos el fichero que contiene los permisos
$ nano rdsqsql/share/utils/mysql_users.sql
# Sustituimos `localhost` por `%` en todas las sentencias. Posteriormente, sustituimos
```

```
# *.* por `%`.* .
```

ERROR: Invalid default value for 'date'

Esto error sucede si no se ha definido un valor por defecto para los campos `date not null` de una sentencia `create table`. Para solucionarlo bastará con ir al script que provoca el error y especificarle el valor por defecto.

Lo más común, en nuestro caso, es que este error se produzca debido a la definición de las sentencias para la creación de tablas de la base de datos oTracker. Los pasos a seguir serían:

```
# Editamos el fichero scheme.sql de oTracker.
$ nano rdsq1/share/oTracker/schema.sql
# Sustituimos en todas las sentencias que haga falta `date` TIMESTAMP NOT NULL , por
# `date` TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ,.
```

1.6 Adaptación de la aplicación a RDS

Los primeros pasos es indicarle a la interfaz web donde ha de ir a buscar los datos. Para ello seguimos los siguientes pasos:

```
# Nos situamos en el directorio web
$ cd /srv/www/omnilus/
# Editamos el perfil de conexión a la base de datos
$ sudo nano profiles/omnilus.xml
# Realizamos los siguiente cambios
# -- <host>actual_host</host> por <host>[endpoint_adress]</host>
# -- <user>actual_user</user> por <user>[user-omnilus]</user >
# -- <pswd>actual_pswd</pswd> por <pswd>[user-password]</pswd >
# Ahora editamos el fichero que proporciona la configuración de usuario
$ sudo nano htdocs/u/db/main/getConfiguration.php
# En la conexión PDO sustituimos:
# -- host=actual_host; por host=[endpoint_adress];
# -- 'actual_user','actual_pswd'); por '[user-omnilus]', '[user-password]');
# Editamos el fichero de registro
$ sudo nano htdocs/base/php/scripts/register.php
# En la función create_bbdd modificar el GRANT:
# -- Poner $bbdd_name.*
# -- Sutiluir localhost por %
# -- Quitar sprintf
```

Llegados esta punto se recomienda buscar las conexiones al `localhost` no contraladas en la interfaz web que vayan destinadas a la base de datos y modificarlas. Una manera de hacerlo podría ser la siguiente:

```
# Nos situamos en el directorio correcto
$ cd /srv/www/omnilus/htdocs
# Creamos un fichero donde guardaremos los resultados de la búsqueda
$ sudo touch search
# Le cambiamos los permisos para que se pueda escribir en el
$ sudo chmod 777 search
# Iniciamos una búsqueda con grep
$ grep -r "localhost" . > search
# Cuando termine sería cuestión de mirar el fichero search e ir modificando las conexiones
# que hagan falta.
```

Hay que tener en cuenta que algunos paquetes de cálculos también hacen conexiones a la base de datos. La forma de proceder para cambiar la dirección de conexión a este sería igual que con la de la interfaz web. Con tal de facilitar ambos procesos, se detallan a continuación una lista de ficheros a modificar en ambos lugares:

```
# En la aplicación web
$ ./htdocs/u/db/tools/crudHelp.php
$ ./htdocs/u/db/cpanel/getApps.php
$ ./htdocs/u/db/cpanel/setApps.php
$ ./htdocs/u/db/projects/loadExample.php
$ ./htdocs/u/db/projects/loadProject.php
# -- Query GRAND: localhost por %.
$ ./htdocs/base/php/scripts/register.php
# -- Sustituimos en dos lugares: DEFINER=[]@localhost por DEFINER=`[user-creación]`@`%`
$ ./htdocs/base/php/scripts/sql/schema.sql

# En los cálculos
# -- Añadir en if bundle[1][0] != '127.0.0.1': la nueva conexión.
$ ./rdutils/src/python/seelogserver.py
# -- Sustituimos *.* por `%`.* .
```

```
# -- Sustituimos '$db-user'@'localhost' por '$db-user'@'%'.
# -- Añadimos a las consultas -h [endpoint_adress]
$ ./rdutils/src/shell/dbbuild.sh
$ ./rdutils/lib/python/getexpeller.py
$ ./rdutils/lib/python/getcustomer.py
$ ./rdslib/lib/python/MySQLDB.py
$ ./rdslib/share/utils/read_str_field.py
$ ./rdslib/share/utils/generate_new_strings
$ ./rdslib-2.0.2/lib/lib1/dataBase/DataBaseMySQL.cc
```

Una vez modificados los paquetes de cálculo hay que recompilarlos e instalarlos. Tras ello solo haría falta probar que todo funciona, y si es el caso, eliminar la base de datos de la instancia donde se encuentra la aplicación.

```
$ sudo apt-get remove mysql-server
$ sudo apt-get purge mysql-server
$ sudo apt-get autoremove
```

Evidentemente esta es una solución que ha de ser temporal y se ha de aplicar con la mentalidad de centralizar los datos de conexión a la base de datos. Además, hay que crear una variable local para poder desarrollar de tal forma que si esta está activa se trabaje contra la base de datos local.

1.7 Adaptación de la aplicación a S3

Antes de poder usar Amazon S3 desde la aplicación web hay que crear una clave de identificación para el servicio. Para ello nos dirigiremos a nuestra cuenta en Amazon AWS, y en la parte izquierda clicaremos en credenciales de seguridad. Una vez realizada la acción nos situaremos en el centro de la página y bajaremos hasta la sección “Credenciales de acceso” donde tendremos que generar una clave. Mediante estas claves ya podemos proceder a usar Amazon S3 en la aplicación web.

Por otra parte, para usar el servicio mediante PHP, que es nuestro caso, tenemos varias alternativas: SDK (*Software Development Kit*) oficial de Amazon, Zen framework, Symfony, etc. En esta implementación se ha usado el SDK (*Software Development Kit*) oficial. También es importante tener en cuenta que S3 trabaja sobre SSL (*Secure Sockets Layer*), por lo que hay que abrir el puerto 443 en los *Grupos de Seguridad* de nuestras instancias.

Ahora que ya nos hemos puesto en contexto, procederemos a:

```
# Instalar el pear
$ sudo apt-get install php-pear
# Instalar las curl
$ sudo apt-get install php5-curl
# Instalar la sdk
$ pear -D auto_discover=1 install pear.amazonwebservices.com/sdk
# Métodos de instalación alternativos en:
# http://docs.aws.amazon.com/aws-sdk-php/guide/latest/installation.html
```

Ya en la aplicación:

```
# Editamos el fichero omnibus.xml
$ sudo nano /srv/www/omnilus/profiles/omnilus.xml
# Añadimos las claves generadas entre los tags <profile>
# -- <awsKey>
# --     <id>[id_clave]</id>
# --     <value>[valor_secreto]</value>
# -- </awsKey>
# Editamos el fichero profiles profile.inc
$ sudo nano /srv/www/omnilus/htdocs/base/php/include/profile.inc
# Añadimos dos constantes para tener a mano el fichero con las claves
# -- define("_AWS_KEYS_FILENAME","omnilus.xml");
# -- define("_AWS_KEYS_FILE ","_PROFILE_PATH."/". _AWS_KEYS_FILENAME);
# Añadimos el fichero awsClass.inc a /srv/www/omnilus/htdocs/base/php/classes/
$ sudo cp [path]/awsClass.inc /srv/www/omnilus/htdocs/base/php/classes/awsClass.inc
# El contenido del fichero awsClass.inc queda más para adelante documentado en este apartado
# así como ejemplos para usar el SDK.
```

Ahora solo nos faltaría modificar aquellos *scripts* que usan ficheros dinámicos para que hagan uso de Amazon S3. Estos *scripts* son los siguientes:

```
$ /srv/www/omnilus/htdocs/u/db/apps/classes/XML.inc
$ /srv/www/omnilus/htdocs/u/db/cpanel/crudLogo.php
```

Para XML.inc hay que especificar que cuando un usuario guarde un cálculo su XML se guarde en el S3 y no en la carpeta de XMLs del servidor. De la misma forma hay que actuar con los logos.

También hay que tener en cuenta, que los cálculos y los *reports* usan los XMLs y los logos, así que al abrir un cálculo su XML se ha de copiar en el temporal desde S3, y al identificarse un usuario en la aplicación, copiar su logo a su carpeta de logos si este no existe.

A continuación se describen los cambios a realizar:

```
# Editamos CRUDXML.php
$ sudo nano /htdocs/u/db/apps/common/CRUDXML.php
# Añadimos:
#     Despues del requiere profile.inc:
# -- require_once $_SERVER["DOCUMENT_ROOT"].'/base/php/classes/awsClass.inc';
#     Antes de instanciar la clase xml
# -- $aws = amazonAWS::getInstance(_AWS_KEYS_FILE);
#     En el bloque del SAVE, UPDATE y READ:
# -- client = $aws->s3Client();
# -- $xml->setAWSClient($client);
# Editamos XML.inc y añadimos:
$ sudo nano /htdocs/u/db/apps/classes/XML.inc
# Añadimos:
#     La siguiente variable privada:
# -- private $s3;
# -- private $s3Path;
#     La siguiente función:
# -- public function setAWSClient($cl) {
# --     $cl->registerStreamWrapper();
# --     $this->s3Path = 's3://[bucket]/[folders]/'.$_SESSION['db'];
# --     $this->s3 = $cl;
# -- }
# En la función load:
#     Comentamos:
# -- //exec('cp '.$xmlFile.' '.$fileName);
#     Añadimos:
# -- if($stream = fopen($this->s3Path.'/'.$this->id.'.xml', 'r')){
# --     file_put_contents($fileName, $stream);
# --     fclose($stream);
# -- }
# En la función doQuery:
#     Comentamos:
# -- //exec('cp -f '.$this->tmp_dir.'/'.$this->tmp_id.'.xml '.$xmlFile);
#     Añadimos:
# -- $s3file = $this->s3Path.'/'.$this->id.'.xml';
# -- if($stream = fopen($this->tmp_dir.'/'.$this->tmp_id.'.xml', 'r')){
# --     file_put_contents($s3file, $stream);
# --     fclose($stream);
# -- }
#     Modificamos:
# -- if(file_exists($xmlFile)) { por if(file_exists($s3file)) {
```

De cara a los logos:

```
$ sudo nano base/php/scripts/access.php
# Añadimos:
# -- require_once $_SERVER["DOCUMENT_ROOT"].'/base/php/include/profile.inc';
# -- require_once $_SERVER["DOCUMENT_ROOT"].'/base/php/classes/awsClass.inc';
#     En setUserLogged una vez se ha autenticado:
# -- $aws = amazonAWS::getInstance(_AWS_KEYS_FILE);
# -- $client = $aws->s3Client();
# -- $client ->registerStreamWrapper();
# -- $defLogo = $_SESSION['db'].'/logo.jpg';
# -- $s3LogoPath = 's3://rdomn/logos/'.$defLogo;
# -- $localLogo = _LOGO_PATH.'/'.$defLogo;
# -- if(!file_exists($localLogo) && file_exists($s3LogoPath)) {
# --     if($stream = fopen($s3LogoPath, 'r')){
# --         file_put_contents($localLogo, $stream);
# --         fclose($stream);
# --     }
# -- }
```

```

# -- }
$ sudo nano u/db/cpanel/crudLogo.php
# Añadimos:
# -- require_once $_SERVER["DOCUMENT_ROOT"].'/base/php/classes/awsClass.inc';
# En el bloque UPDATE previamente al success true:
# -- $aws = amazonAWS::getInstance(_AWS_KEYS_FILE);
# -- $client = $aws->s3Client();
# -- $client ->registerStreamWrapper();
# -- if($stream = fopen(LOGO_PATH.'/logo.jpg', 'r')){
# -- file_put_contents('s3://rdomn/logos/'._SESSION['db'].'/logo.jpg', $stream);
# -- fclose($stream);
# -- }

```

De la misma forma que en la migración de la base de datos, hace falta definir una variable local de entorno que indique si es el sistema está en producción o desarrollo, de tal forma que durante producción se trabaje con los directorios locales.

Por otro lado se recomienda añadir a la clase awsClass.inc dos funciones, una para subir los fichero y otros para bajarlos, de tal forma que no haya que ir repitiendo el mismo bloque de código destinado a estas funciones en tantos ficheros.

Ejemplo de descarga y subida mediante la SDK

Como ejemplo se muestra a continuación el código necesario tanto para subir como para bajar un fichero del servicio S3 de Amazon.

```

<?
//Cargamos los includes de respaldo
// -- profile.inc :: variables de uso general
// -- awsClass.inc :: interfaz para interactuar con las clases de Amazon.
//Ambos ficheros quedan documentados en este documento
require_once $_SERVER["DOCUMENT_ROOT"].'/base/php/include/profile.inc';
require_once $_SERVER["DOCUMENT_ROOT"].'/base/php/classes/awsClass.inc';

//Instanciamos la clase pasándole el path donde se encuentran nuestras llaves
//de autenticación de amazon almacenadas tal como se ha documentado previamente
$aws = amazonAWS::getInstance(_AWS_KEYS_FILE);

//Iniciamos un cliente con S3
$client = $aws->s3Client();
//Indicamos la aplicación de wrapper sobre el stream de PHP
$client->registerStreamWrapper();

//SUBIDA DE UN FICHERO
// -- Cargamos en el steam el fichero a subir en modo lectura
if($stream = fopen('[path_del_fichero]/[fichero].[extension]', 'r')){
    // -- Le indicamos que lo guarde en S3: si no estan creadas las carpetas que
    // -- especificamos estas se crearán automáticamente
    file_put_contents('s3://[bucket]/[folders]/[nombre_es_S3].[extension_en_S3]',
        $stream);
    fclose($stream);
}

//DESCARGA DE UN FICHERO
// -- Indicamos como fichero abrir una dirección de un fichero valido en S3.
if($stream = fopen('s3://[bucket]/[folders]/[nombre_es_S3].[extension_en_S3]', 'r')){
    // -- Indicamos donde queremos guardar el fichero
    file_put_contents('[path_del_fichero]/[fichero].[extension]', $stream);
    fclose($stream);
}
?>

```

Para otro ejemplos visitar la URL <http://docs.aws.amazon.com/aws-sdk-php/guide/latest/service-s3.html> y consultar el apartado “Amazon S3 stream wrapper”.

Fichero awsClass.inc

Queda escrito el código para el fichero awsClass.inc más básico. Fichero sobre el cual sería conveniente escribir los métodos para acceso a S3 u otros servicios de Amazon AWS, así como las funciones para hacer usos de estos.

```

<?php
require 'AWSSDKforPHP/aws.phar';
use Aws\S3\S3Client;

/* Clase encargada de gestionar las conexiones a la base de datos */
class amazonAWS {

    private $id_key;
    private $vl_key;
    private $s3Client;

    static $_instance;

    private $noClientError = "No client";

    /*La función construct es privada para evitar que el objeto pueda ser creado mediante
    new*/
    private function __construct($xml_file){
        $this->setKeys($xml_file);
    }

    /*Método para establecer los parámetros de la conexión*/
    private function setKeys($xml_file){
        // Enable user handling errors
        libxml_use_internal_errors(true);
        if($xml = simplexml_load_file($xml_file)) {
            $this->id_key = $xml->awsKey->id;
            $this->vl_key = $xml->awsKey->value;
        }
    }

    /*Evitamos el clonaje del objeto. Patrón Singleton*/
    private function __clone(){ }

    /*Función encargada de crear, si es necesario, el objeto. Esta es la función que debemos
    llamar desde fuera de la clase para instanciar el objeto, y así, poder utilizar sus méto-
    dos*/
    public static function getInstance($xml_file){
        if (!(self::$_instance instanceof self)){
            self::$_instance = new self($xml_file);
        }

        return self::$_instance;
    }

    public function s3Client() {
        $this->s3Client = S3Client::factory(array(
            'key' => $this->id_key,
            'secret' => $this->vl_key
        ));

        return $this->s3Client;
    }

    public function listBuckets() {
        $client = $this->s3Client;

        if($this->isClient($client)) {
            $result = $client->listBuckets();
            return $result;
        } else {
            return $noClientError;
        }
    }

    public function isClient($cl) {
        if(isset($cl) && !empty($cl)) {
            return true;
        } else {
            return false;
        }
    }
}
} ?>

```

2. Guía de migración para desarrollo

La idea principal de esta fase es migrar el Subversion a la nube, de esta forma, cualquier trabajador o colaborador podrá participar en el proyecto desde cualquier lugar. Además, facilitará el *testing* y la puesta en producción.

Para ello hemos de seguir los siguientes pasos:

1. En Amazon AWS creamos una instancia t1.micro junto a su clave a partir de una AMI (*Amazon Machine Image*) de Debian 7 (ami-954559e1). Además, durante la creación le añadimos un EBS persistente que es donde montaremos el Subversion.
2. Asociamos una IP estática a la instancia mediante la interfaz de EC2.
3. En “*Security Groups*” seleccionamos el grupo asociado a la instancia y en la pestaña de *outbound* eliminamos la regla que viene por defecto y añadimos el puerto HTTP 80 y HTTP443.

4. Ahora, en nuestro ordenador local procederemos de la siguiente forma:

```
# Copiamos la clave descargada asociada a la instancia a la carpeta .ssh de nuestro usuario
$ cp ~/Downloads/miclave.pem ~/.ssh
# Le cambiamos los permisos a solo lectura
$ sudo chmod 400 miclave.pem
# Si el usuario dueño es root lo cambiamos por el nuestro
$ sudo chown [mi_usuario]:[mi_grupo] miclave.pem
# creamos un fichero de configuración para facilitar la conexión
$ sudo nano .ssh/config
# En el añadimos:
# -- Host aws.host.development
# -- Hostname [IP_estatica_asociada_a_la_instancia]
# -- User admin //Por defecto en la AMI de Debian 7.2
# -- IdentityFile "~/ssh/[nombre_del_fichero].pem"
# Procedemos a conectarnos
$ ssh aws.host.development
```

5. Una vez conectados procedemos a:

```
# Actualizar el sistema
$ sudo apt-get update
$ sudo apt-get dist-upgrade

# Configurar el ssh
# -- Nos añadimos como usuario al sistema para la conexión ssh
$ adduser [usuario]
# -- Nos añadimos al sudoers: [usuario] ALL=(ALL) ALL
$ sudo nano /etc/sudoers
# Nos copiamos la clave del usuario admin
$ sudo cp -r ~/.ssh /home/[usuario]/
# Modificamos los permisos
$ sudo chown -R [usuario]:[usuario]/home/[usuario]/.ssh/
# Nos salimos
$ exit
```

6. Una vez fuera comprobamos si funciona:

```
# Modificamos el fichero config de .ssh:
$ sudo nano .ssh/config
# En el introducimos:
# -- Host aws.host.production.[usuario]
# -- Hostname [IP_estatica_asociada_a_la_instancia]
# -- User [usuario]
# -- IdentityFile "~/ssh/[nombre_del_fichero].pem"
# Probamos de acceder:
$ ssh aws.host.production.[usuario]
# Si nuestro usuario puede acceder al sistema y usar el comando sudo todo ha ido
# Correctamente y podemos seguir con los puntos marcados. Si este no fuera el caso
# hay que lograr cambiar el usuario por defecto para ingresar por ssh a la máquina
# por uno de nuestro control.
```

7. Si los pasos anteriores han funcionado realizamos las siguientes acciones:

```
# Quitamos el acceso por ssh al usuario por defecto admin.
$ sudo rm -rf /home/admin/.ssh
# Miramos que no se pueda acceder con usuario root a la máquina
```



```

$ sudo nano /etc/ssh/sshd_config
# Miramos que la linea siguiente este tal cual se describe aquí:
# -- PermitRootLogin no
# Aprovechamos y cambiamos el puerto por defecto. En el mismo fichero:
# -- Port [nuevo_puerto]
# Reiniciamos el serviio para que tenga efecto
$ sudo /etc/init.d/ssh restart

```

8. Sin salirnos de la instancia nos dirigimos a Amazon AWS y en el “Security Gropus” de la instancia tanto en “Inbound” como en “Outbound” añadimos el nuevo puerto para SSH.

9. Mediante una nueva consola probamos si podemos acceder. Para ello:

```

# Modificamos el fichero config de .ssh:
$ sudo nano .ssh/config
# En el introducimos:
# -- Host aws.host.production.[usuario].port
# -- Hostname [IP_estatica_asociada_a_la_instancia]
# -- User [usuario]
# -- IdentityFile "~/ssh/[nombre_del_fichero].pem"
# -- Port [nuevo_puerto]
# Probamos de acceder:
$ ssh aws.host.production.[usuario].port
#Si accedemos continuamos con el proceso, sino hay que investigar porque no funciona el cambio de puerto. En este paso tambien es recomendable ver si con el usuario admin aún se puede acceder, y de la misma forma, con el usuario root.

```

10. Si todo ha ido correctamente podemos cerrar la conexión abierta previamente y en el “Security Gropus” de nuestra instancia en EC2 eliminar de “Inbound” el puerto ssh 22.

Resumen: En este punto nos hemos de encontrar con una conexión SSH abierta con la instancia, realizada a partir de un usuario y puerto personalizado: con cuatro reglas en el grupo de seguridad de EC2: puerto personalizado para SSH + HTTP80 tanto en “inbound” como “outbound”; con que ni el usuario por defecto admin ni root puede acceder al servicio; y con que el sistema esta totalmente actualizado.

11. Con la conexión abierta procedemos a configurar el fail2ban:

```

# Instalamos la aplicación
$ sudo install fail2ban
# Realizamos la configuración. Para ello:
# -- Replicamos el fichero de configuración
$ sudo cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.local
# -- Modificamos la configuración
# -- -- backend = auto por backend = polling
$ sudo nano /etc/fail2ban/jail.conf
# -- -- backend = auto por backend = polling
$ sudo nano /etc/fail2ban/jail.local
# -- -- backend = auto por backend = polling
# -- -- en [ssh] port = ssh por port = [nuevo_puerto]
# Reiniciamos fail2ban para que tengan efecto los cambios
$ sudo /etc/init.d/fail2ban restart
# Llegados este punto la ideal sería probar que funciona, para ello en el fichero jail.local ponemos en el tiempo de baneo (bantime) un tiempo bajo y reiniciamos el servicio. En nuestro PC modificamos el usuario de nuestro fichero config de SSH y probamos a conectarnos reiteradas veces. En caso de que nos bane todo funciona correctamente, así que esperamos el tiempo de ban y volvemos a poner el fichero config tal como estaba. Tras pasar el tiempo entramos a la instancia, reconfiguramos el fail2ban con un tiempo para ban más acorde y reiniciamos el servicio. En caso de que no funcione sería cuestión de mirar que está pasando.

```

12. Configuramos los locales:

```

# Reconfigure
$ sudo dpkg-reconfigure locales
# Presinamos espacio en all, posteriormente presionamos enter en all y seguimos las instrucciones. El local por defecto ha de ser en_US.utf8.

```

13. Montamos el EBS y creamos el directorio para el Subversion:

```

# Para saber el nombre de la unidad ebs a montar miráremos en la interfaz de Amazon EC2
# el nombre de los volúmenes asignados a la instancia > Seleccionamos la instancia, en la
# pestaña “Detail”, al final de la columna izquierda cualquiera de las unidades que
# no son root.
# Si el nombre es del estilo /dev/sdb en la máquina tendrá el siguiente nombre /dev/xvdb
# (s = xv)

```

```

# Comprobamos su existencia
$ ls /dev/xvdb
# Le damos formatos
$ sudo mkfs.ext4 /dev/xvdb
# Creamos la carpeta donde montaremos la unidad
$ sudo mkdir /media/ebs
# Montamos la unidad
$ sudo mount -t ext4 /dev/xvdb /media/ebs/
# Creamos la carpeta donde situaremos los repositorios
$ sudo mkdir /media/ebs/svn
# Le asignamos permisos de apache y de subversion
$ sudo chown -R www-data:subversion /media/ebs/svn/
$ sudo chmod -R 770 /media/ebs/svn/
# Añadimos para que se monte al inicio del sistema
Sudo nano /etc/fstab
# -- /dev/xvdb /media/ebs/ ext4 defaults 1 1

```

14. Procedemos con el Subversion:

```

# Instalamos el subversios y sus tools.
$ sudo apt-get install subversion subversion-tools
# Instalamos el apache y el mod necesario para poder acceder a él desde Internet.
$ sudo apt-get install apache2 libapache2-svn
# Añadimos un grupo para los usuarios del subversion
$ sudo groupadd subversion
# Despeus de añadir los usuarios para este mediante adduser los añadimos
# al grupo del subversion
$ sudo usermod -G subversion sol
# Los añadimos a la configuración del mod de apache:
# -- Para el primer usuario
$ sudo htpasswd -c /etc/apache2/dav_svn.passwd myuser1
# -- Para los demas usuarios
# Configuramos el mod de apache para el svn
$ sudo nano /etc/apache2/mods-available/dav_svn.conf
# -- Descomentamos el bloque locations.
# -- Modificamos <Location /svn> a <Location /omnilus>
# -- Descomentamos DAV svn
# -- Descomentamos SVNPath y le cambiamos el path por SVNPath /media/ebs/svn/omnilus
# -- Descomentamos AuthType Basic
# -- Descomentamos AuthName "Subversion Repository"
# -- Descomentamos AuthUserFile /etc/apache2/dav_svn.passwd
# -- Añadimos Require valid-user
# Creamos la carpeta para el repositorio
$ sudo mkdir -p /media/ebs/svn/omnilus
# Le indicamos al svn donde se encuentra
$ sudo svnadmin create /media/ebs/svn/omnilus
# Creamos los directorios básicos en el repositoriols
$ sudo svn mkdir --message="Setting up the directories..." \
  file:///media/ebs/svn/omnilus/trunk \
  file:///media/ebs/svn/omnilus/tags \
  file:///media/ebs/svn/omnilus/branches \
#Cambiamos los permiso
$ sudo chown -R www-data.www-data media/ebs/svn
# Eliminamos la página por defecto de apache
$ sudo rm -r /var/www index.html
# Configuramos en sites-enabled a nuestro gusto.
# Reiniciamos Apache
$ sudo /etc/init.d/apache2 restart

```

A partir de aquí solo hace falta comprobar que funciona mediante *shell* (svn ls http://localhost/omnilus), sincronizar los repositorios (local – EBS; svnsync + cron), implementar SSL (*Secure Sockets Layer*) y aplicar las medidas de seguridad necesarias tanto en apache como en *fail2ban*.

3. Guía de implementación de copias de respaldo

Las copias de respaldo se harán principalmente sobre el contenido del EBS de desarrollo y la base de datos, los demás servicios ya cuentan con redundancia o políticas automáticas de respaldo. De todas formas se documenta a continuación la implementación de forma general en todos los servicios para tener una noción de cómo funcionan.

3.1 Respaldo de instancias

Cuando se genera una imagen (AMI - *Amazon Machine Image*) de instancia desde la interfaz web de Amazon EC2 esta automáticamente se hospeda en S3 y genera un snapshot de los discos asociados, de tal forma, que no hace falta implementar nada en especial en este caso. Bastaría con generar una AMI (*Amazon Machine Image*) para las instancias base una vez montado todo el sistema.

En caso de fallo de una instancia solo haría falta crear una de nueva a partir de una AMI (*Amazon Machine Image*) de respaldo ya creada.

3.2 Respaldo del EBS de desarrollo

En este caso procederemos a crear *snapshots* a partir de la API (*Application Programming Interface*) de Amazon AWS. Para ello, creamos un cron que ejecute el siguiente *scripts*:

```
<?
# Current time
$time = date(DATE_RFC2822);
# Save response from create snapshot
$resp = array();
# Save response from describe snapshot
$snap = array();

# Get current snapshots
//exec("ec2-describe-snapshots --filter \"description=*EBS DEV*\" --region eu-west-1",
    $snap);

# Create snapshot
exec("ec2-create-snapshot -d 'PHP Script - EBS DEV Daily - $time' vol-6604bc33 --region
eu-
    west-1", $resp);

$delControl = true;
# Delete current snapshots
/*if(isset($snap) && !empty($snap)) {
    for($i = 0; $i < sizeof($snap); $i++) {
        $aux = preg_split('/\s+/', $snap[$i]);
        exec("ec2-delete-snapshot $aux[1] --region eu-west-1", $ok);
        if(!isset($ok) || empty($ok)) {
            $delControl = false;
        }
    }
} else {
    $delControl = false;
}

if(!$delControl) {
    sendMail('noDeleteSnap');
}
*/

if(!isset($resp) || empty($resp)) {
    sendMail('failSnap');
}
?>
```

Para que este funcione primero hemos de realizar los siguientes pasos:

```
# Instalar las tools de Amazon AWS:
# descarga: https://aws.amazon.com/developertools/351
$ scp ec2-api-tools.zip aws.host.development:/tmp
```

```

$ sudo mkdir /usr/local/ec2
$ unzip /tmp/ec2-api-tools.zip
$ sudo cp -r ec2-api-tools/* /usr/local/ec2
# Instalamos java
$ sudo apt-get install openjdk-7-jre
# Configuramos las variables de entorno necesarias
$ sudo nano /etc/profile
# -- Añadimos:
# -- export EC2_HOME=/usr/local/ec2
# -- export PATH=$PATH:$EC2_HOME/bin
# -- export JAVA_HOME=/usr
# -- export AWS_ACCESS_KEY=your-aws-access-key-id
# -- export AWS_SECRET_KEY=your-aws-secret-key
# Aplicamos los cambios
$ source /etc/profile
# Instalamos PHP sino está instalado
$ sudo apt-get install php5

```

Por último hay que tener en cuenta que los *snapshot* se guardan de forma predeterminada en S3, así que un principio, la solución es fiable.

3.3 Respaldo de la base de datos

En este caso, al igual que el anterior, hemos procedido a usar las herramientas que nos proporciona Amazon para crear *snapshots* sobre la base de datos, así que la solución pasa por añadir al script del apartado anterior las siguientes líneas de código:

```

exec("rds-create-db-snapshot --db-instance-identifier ommicrodb -s PHP Script DB Daily", $db);

if(!isset($db) || empty($db)) {
    sendMail('db span no created', $db);
}

```

Para que funcione previamente hemos de:

```

# Descárganos los comandos de RDS para el shell
$ wget http://s3.amazonaws.com/rds-downloads/RDSCli.zip
# Instalar los binarios
$ unzip /tmp/RDSCli.zip
$ sudo cp -r /tmp/RDSCli/ /usr/local/rds
# Configuramos las variables de entorno
$ sudo nano /etc/profile
# -- Añadimos:
# -- export AWS_RDS_HOME=/usr/local/rds
# -- export PATH=$PATH:$AWS_RDS_HOME/bin
# -- export EC2_REGION=us-west-1
# -- export AWS_CREDENTIAL_FILE=${AWS_RDS_HOME}/credential-file-path.template
# Replicamos los cambios
$ source /etc/profile
# Configuramos las credenciales
$ sudo nano ${AWS_RDS_HOME}/credential-file-path.template
# -- Quitamos los comentarios
# -- Modificamos:
# -- AWSAccessKeyId=<Write your AWS access ID>
# -- AWSSecretKey=<Write your AWS secret key>
# Cambios los permisos al fichero para evitar errores
$ sudo chmod 600 ${AWS_RDS_HOME}/credential-file-path.template

```

Se contempla la idea de crear copias a local cada semana o dos mediante un *mysqldump* en un *cron*.

3.4 Respaldo de los ficheros de S3

S3 ya cuenta con redundancia y control de versiones, además se pueden definir ciclos de vida para los ficheros, de tal forma que se puede determinar la duración de su existencia o rutinas de copias a Glacier.

Se podría implementar un pequeño script para descargar datos para tener una copia en local o meterlos en el EBS y que al ejecutarse el script de creación de *snapshot* estos se guardaran, pero carece de sentido conforme el tamaño de los datos aumenta.

Así que en este punto se ha optado por usar el control de versiones.

4. Guía de implementación para la supervisión y el autoescalado

Nota: Desde el 11 de diciembre del 2013, fecha posterior al redactado de este apartado, el autoescalado y la supervisión se puede hacer íntegramente desde la interfaz web.

Para la supervisión y el autoescalado del sistema se han de usar cuatro servicios distintos de Amazon junto a EC2: Cloud Watch, Auto Scaling, Load Balancer y SNS.

Lo primero que haremos será configurar SNS, que es el servicio que nos permite recibir notificaciones por correo electrónico de los demás servicios. Para ello procederemos de la siguiente forma:

1. Panel de control de Amazon AWS > SNS (Tercera columna) > "Create Topic".
2. Rellenamos el formulario > "Create".
3. Presionamos en "Subscription" > Rellenamos el formulario (protocolo: mail; endpoint: dirección de correo electrónico). > Clicamos en "Suscribe".

Ahora que ya tenemos el servicio de notificaciones activo procedemos a configurar el sistema de forma general:

1. Quitamos la Elastic IP de la instancia de producción si tenemos una asociada a esta. Esto implica que para las conexiones SSH (*Secure Shell*), si usábamos dicha IP ahora tendremos que usar el endpoint de la instancia (DNS público).
2. Creamos un "Load balancer" mediante la interfaz de EC2 presionando en la correspondiente opción en el menú de la parte izquierda. Durante la creación indicaremos que máquinas ha de controlar, en que intervalo de tiempo y contra que página lo ha de hacer. En nuestro caso indicaremos la instancia de producción, el puerto 80 y al fichero index.php.
A partir de ahora podremos acceder a la aplicación mediante el *endpoint* del balanceador.
3. Accedemos a la máquina de producción a través de su *endpoint* y ponemos que arranque el apache en el momento del inicio del sistema: `$ sudo update-rc.d apache2 defaults`
4. Creamos una AMI (*Amazon Machine Image*) de la máquina de producción, que será que se usará para escalar el sistema.

En este punto tenemos la estructura básica para comenzar hacer un sistema escalable de forma manual, así que ahora procedemos a automatizar los procesos de incorporación o eliminación de instancias. Y a cada una de estas tareas su correspondiente notificación.

5. Entramos a la instancia de desarrollo y realizamos las siguientes tareas:

```
# Nos descargamos las herramientas de shell para el "autoscaling"
# https://aws.amazon.com/developertools/2535?encoding=UTF8&jiveRedirect=1
# http://docs.aws.amazon.com/AutoScaling/latest/DeveloperGuide/US_BasicSetup.html
$ cd /tmp/
$ wget http://ec2-downloads.s3.amazonaws.com/AutoScaling-2011-01-01.zip
# Instalamos los binarios
$ unzip AutoScaling-2011-01-01.zip
$ sudo cp -r AutoScaling-1.0.61.3/ /usr/local/autoScaling
# En este instante si hemos realizado los pasos para la instalación de RDS solo tendremos que
indicar
# las variables de entorno, sino fuera el caso habria que instalar el JAVA, indicar la region
por defecto y
# el fichero de credenciales tal como se describe en la guia para la migración de base de da-
tos
$ sudo nano /etc/profile
# -- Añadimos:
# -- export AWS_AUTO_SCALING_URL=https://autoscaling.eu-west-1.amazonaws.com
# -- export AWS_AUTO_SCALING_HOME = /usr/local/autoScaling
# -- export PATH=$PATH:$AWS_AUTO_SCALING_HOME/bin
# Replicamos los cambios.
$ source /etc/profile
# Probamos que funcione
$ as-cmd
# Creamos una configuración de arranque para el autoescalado
$ as-create-launch-config [name_config] --image-id [ami_production] --instance-type m1.small
# Notificación
# -- OK-Created launch config
# Creamos un grupo de escalado
```

```

$ as-create-auto-scaling-group [name_group] --launch-configuration [name_config]
--availability-zones eu-west-1a,eu-west-1c,eu-west-1b --load-balancers [name_balancer]
--max-size [max_instances] --min-size [min_instances]
# Notificación
# -- OK-Created AutoScalingGroup
# Comprobamos que funcionan correctamente
$ as-describe-auto-scaling-groups [name_group] -headers
# Notificación
# -- AUTO-SCALING-GROUP GROUP-NAME LAUNCH-CONFIG AVAILABILITY-ZONES
# -- LOAD-BALANCERS MIN-SIZE MAX-SIZE DESIRED-CAPACITY TERMINATION-POLICIES
# -- AUTO-SCALING-GROUP omn-group-scaling omn-launch-config eu-west-1a,eu-west-1b,eu-
west-
# 1c omnBalancer 1 10 1 Default
# -- INSTANCE INSTANCE-ID AVAILABILITY-ZONE STATE STATUS LAUNCH-CONFIG
# -- INSTANCE i-a40f2de8 eu-west-1c InService Healthy omn-launch-config

```

En este punto, mediante la interfaz EC2, veremos cómo se crea una nueva instancia que ya forma parte del balanceador. Respecto a la instancia inicial de producción podemos quitarla del balanceador o apagarla sino la vamos usar. Ahora se trata de añadir políticas de autoescalado, que serán las que determinarán cuando se inicia una nueva instancia o se elimina.

```

# Añadimos una política de autoescalado al grupo (auto-scaling-group) creado anteriormente para que
# añada una (adjustment=1) instancia (ChangeInCapacity)
$ as-put-scaling-policy [name_add] --adjustment=1 --auto-scaling-group=[name_group]
--type ChangeInCapacity
# Notificación: Se ha de guardar para la generación de alarmas y definición de las reglas
# -- [arn_add_policy]
# Añadimos una política de autoescalado al grupo (auto-scaling-group) creado anteriormente para que
# elimine una (adjustment=1) instancia (ChangeInCapacity)
$ as-put-scaling-policy [name_remove] --auto-scaling-group=[name_group] --adjustment=-1
--type ChangeInCapacity
# Notificación: Se ha de guardar para la generación de alarmas y definición de las reglas
# -- [arn_remove_policy]
# Podemos comprobar su creación
$ as-describe-policies --auto-scaling-group omn-group-scaling -headers

```

Ahora tenemos la configuración básica para el autoescalado. Los siguientes pasos será determinar bajo qué condiciones ha de escalar el sistema.

```

# Nos descargamos las herramientas para CloudWatch
$ wget http://ec2-downloads.s3.amazonaws.com/CloudWatch-2010-08-01.zip
$ Unzip CloudWatch-2010-08-01.zip
# Instalamos los binarios
$ sudo cp -r CloudWatch-1.0.13.4/ /usr/local/cloudWatch
# Configuramos las variables de entorno
$ sudo nano /etc/profile
# Añadimos:
# -- export AWS_CLOUDWATCH_HOME=/usr/local/cloudWatch
# -- export PATH=$PATH:$AWS_CLOUDWATCH_HOME/bin
# -- export AWS_CLOUDWATCH_URL=http://monitoring.eu-west-1.amazonaws.com/
# -- source /etc/profile
# Comprobamos que funcione.
$ mon-cmd
# Configuramos las alarmas: en este caso se levantarán instancia o eliminar en función del
# uso de CPU, tal como se indica en --metric-name. --threshold es el valor límite y --period
# indica en el periodo de tiempo que se ha de producir.
$ mon-put-metric-alarm --alarm-name [name01] --metric-name CPUUtilization --namespace AWS/EC2
--statistic Average --period 60 --threshold 70
--comparison-operator GreaterThanOrEqualToThreshold
--dimensions AutoScalingGroupName=[name_group] --evaluation-periods 1
--alarm-actions [arn_add_policy]
# Notificación:
# -- OK-Created Alarm
$ mon-put-metric-alarm --alarm-name [name02] --metric-name CPUUtilization --namespace AWS/EC2
--statistic Average --period 120 --threshold 40
--comparison-operator LessThanOrEqualToThreshold
--dimensions AutoScalingGroupName=[name_group] --evaluation-periods 2
--alarm-actions [arn_remove_policy]
# Notificación:

```

```
# -- OK-Created Alarm
# Podemos comprobar que se han creado
$ mon-describe-alarms --headers
```

En este punto ya hemos terminado todo el proceso y el sistema es autoescalable. Para las notificaciones tendremos que dirigirnos a CloudWatch mediante el panel de control de Amazon AWS y allí, editar las alarmas que hemos creado.

Ahora que se ha terminado el proceso, si alguna vez queremos modificar la configuración de inicio del grupo de autoescalado primero tendremos que crear una nueva mediante `as-create-launch-config` y posteriormente actualizar el grupo mediante `as-update-auto-scaling-group`. Por ejemplo:

```
$ as-create-launch-config [new_cfg] --image-id [new_ami] --instance-type [new_type]
$ as-update-auto-scaling-group [name_group] --launch-configuration [new_cfg]
# Notificación: OK-Updated AutoScalingGroup
```

Para eliminar la antigua basta con `as-delete-launch-config`.

Sería importante aplicar este concepto a la base de datos y/o parámetros como la latencia. Punto que queda a desarrollar a futuro.

Para más información:

http://docs.aws.amazon.com/AutoScaling/latest/DeveloperGuide/US_BasicSetup.html

<http://docs.aws.amazon.com/AutoScaling/latest/DeveloperGuide/as-scale-based-on-demand.html>

ANEXO 02

MANUAL DE USO DE AMAZON AWS

1. Gestión de cuenta en Amazon AWS

En esta sección del anexo se documentan aquellos manuales enfocados a la gestión de cuenta. Los manuales presentados son: creación de cuenta, configuración de doble autenticación y creación de claves o certificados.

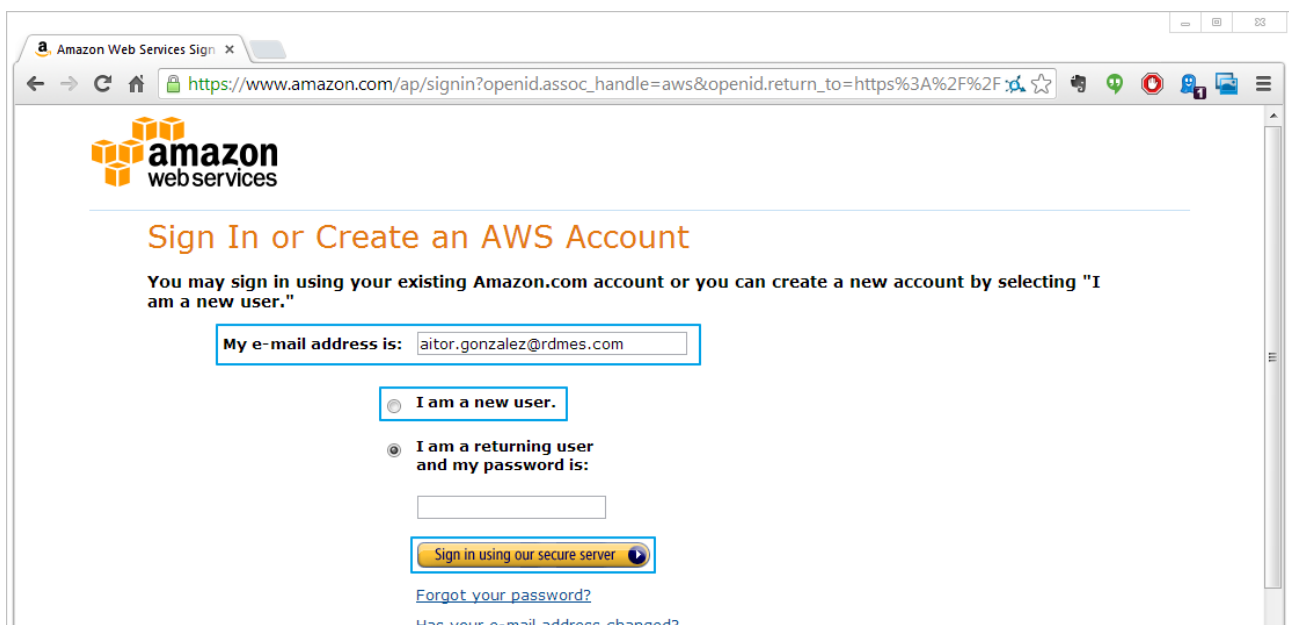
1.1 Creación de cuenta

Para crear una cuenta en Amazon AWS, paso indispensable para hacer uso de sus servicios, nos hemos de dirigir a la siguiente dirección web <https://aws.amazon.com/es/> y posteriormente clicar en el botón "Inscribirse" (Ilustración 11).



Ilustración 11 - Paso previo a la inscripción 01 de 02: Localizando la opción.

Una vez clicado el botón nos aparecerá un formulario, visible en la **Ilustración 12**, donde introduciremos en "My e-mail address" la dirección de correo electrónico con la que queremos acceder al servicio, y, posteriormente, seleccionaremos "I am a new user". Una vez hecho estos pasos, clicaremos en el botón "Sign in using our secure server".



Learn more about [AWS Identity and Access Management](#) and [AWS Multi-Factor Authentication](#), features that provide additional security for your AWS Account.

Ilustración 12 - Paso previo a la inscripción 02 de 02: Preformulario.

Se nos mostrará el formulario como el de la **Ilustración 13** para introducir las credenciales de acceso al servicio.

amazon
webservices

Login Credentials

Use the form below to create login credentials that can be used for AWS as well as Amazon.com.

My name is:

My e-mail address is:

Type it again:

note: this is the e-mail address that we will use to contact you about your account

Enter a new password:

Type it again:

About Amazon.com Sign In

Amazon Web Services uses information from your Amazon.com account to identify you and allow access to Amazon Web Services. Your use of this site is governed by our Terms of Use and Privacy Policy linked below.

Ilustración 13 - Inscripción paso 01 de 06: Credenciales.

Completaremos el formulario de la siguiente forma:

- En el campo “*My name is*” introduciremos nuestro nombre.
- En el campo “*My e-mail address is*” nuestra dirección de correo electrónico, que será la que usaremos posteriormente para acceder al servicio y a través de la cual se podrá poner en contacto con nosotros el proveedor. En “*Type it again*” reescribimos la dirección de correo.
- En el campo “*Enter a new password*” la clave con la que quereos identificarnos ante el servicio. Posteriormente la reescribimos en “*Type it again*”.

Una vez introducidos todos los campos presionamos el botón “*Continue*”. Todo seguido nos saldrá otro formulario para rellenar (**Ilustración 14**).

amazon
webservices


Suscripción a Amazon Web Services

Datos de contacto

* campos obligatorios

Nombre completo*:
Nombre de la empresa:
País*:
Línea 1 de dirección*:
Dirección postal, apartado de correos, nombre de la compañía, A/A
Línea 2 de dirección:
Apartamento, suite, unidad, edificio, piso, etc.
Ciudad*:
Provincia o Región*:
Código postal*:
Número de teléfono*:

Security Check

Imagen:
Try a different image  ¿Por qué te pedimos que escribas estos caracteres?

Escribe los caracteres de la imagen*:
¿Problemas? Ponte en contacto con nosotros.

Contrato de cliente de AWS

Marca esta opción para indicar que has leído y aceptas las condiciones generales del Contrato de cliente de Amazon Web Services.

Ilustración 14 - Inscripción paso 02 de 06: Datos de contacto

En este caso procederemos de la siguiente forma:

- En *Nombre completo* introduciremos nuestro nombre y apellidos.
- En *Nombre de la empresa*, si lo deseamos ya que es opcional, el nombre de la empresa a la que pertenecemos.
- En *País* seleccionamos un país de nuestra conveniencia.
- En *Línea 1 de dirección* una dirección de contacto compuesta por calle y número.
- En *Ciudad* la población donde de la dirección especificada.
- En *Provincia o Región* la provincia de la dirección especificada.
- En *Código Postal* el código postal de la dirección especificada.
- En *Número de teléfono* un número de contacto.

Posteriormente procederemos a introducir las letras de la imagen que se nos muestra en *Escribe los caracteres de la imagen*, seleccionaremos el *checkbox* marcado en azul conforme aceptamos las condiciones generales del servicio y para finalizar presionaremos el botón *Crear cuenta y continuar*. Pasaremos inmediatamente a otro formulario (**Ilustración 15**) donde tendremos que introducir nuestros datos para el cobro del servicio.



amazon web services™ Suscripción a Amazon Web Services

CREATE ACCOUNT **PAYMENT METHOD** IDENTITY VERIFICATION SUPPORT PLAN CONFIRMATION

Se han creado las credenciales de su cuenta de AWS, pero para poder empezar a usar los servicios, necesita indicar sus datos de pago y continuar. No hay tarifa de inscripción, solamente se paga el consumo realizado.

Introduzca sus datos de pago a continuación

No se le cobrará nada en su tarjeta de crédito hasta que no empiece a utilizar AWS, y gran parte de sus aplicaciones y uso de AWS pueden funcionar dentro del nivel de uso gratuito de AWS. Si su consumo mensual supera el nivel gratuito de AWS, se le facturarán servicios en la tarjeta de crédito que indique a continuación. [Ver precios del servicio en detalle](#)

* campos obligatorios

Tarjeta de crédito*:

Número de tarjeta*:

Titular de la tarjeta*:

Fecha de vencimiento*:

Introduce tu dirección de facturación

Seleccione la dirección de facturación asociada a su tarjeta de crédito.

Utilizar mi dirección de contacto como dirección de facturación
(Brutau 26, Sabadell, Barcelona 08203, ES, 97206934)

Introduce una nueva dirección

Ilustración 15 - Inscripción paso 03 de 06: Datos de facturación.

Tras rellenar los datos, siguiendo la lógica de los formularios anteriores, y presionar el botón “Continuar” se mostrará una página como la visible en la **(Ilustración 16)**.

Servicios Web de Amazon

<https://portal.aws.amazon.com/gp/aws/developer/registration/index.html>

amazon web services™ Suscripción a Amazon Web Services

CREATE ACCOUNT **PAYMENT METHOD** IDENTITY VERIFICATION SUPPORT PLAN CONFIRMATION

Datos del impuesto sobre el valor añadido

Hemos advertido que tu dirección de facturación está en un país que aplica el impuesto sobre el valor añadido (IVA). Marca la casilla correspondiente para indicar si está exento del IVA y, si procede, indícanos tu número de registro de IVA para evitar el cobro del IVA en su factura mensual.

Mi empresa está aplica el IVA y esta es mi identificación fiscal a efectos de IVA:

No aplico el IVA, o indicaré mi identificación fiscal a efectos de IVA en otro momento
Puedes introducir un número de registro de IVA cuando quieras desde la página de actividad de tu cuenta de AWS.

Ilustración 16 - Inscripción paso 04 de 06: Impuestos.

En esta hemos de especificar si se aplicará el IVA (Impuesto sobre el Valor Añadido) en los cobros del servicio. En caso afirmativo, tendremos que seleccionar la primera opción que se nos presenta e introducir el número de registro del IVA (Impuesto sobre el Valor Añadido). Si no fuera el caso bastaría con seleccionar la segunda opción. Una vez hayamos seleccionado una opción proseguiremos presionando el botón continuar el cual nos redireccionará hacia una página donde tendremos que validar nuestra identificación tal como se aprecia en la **Ilustración 17**.

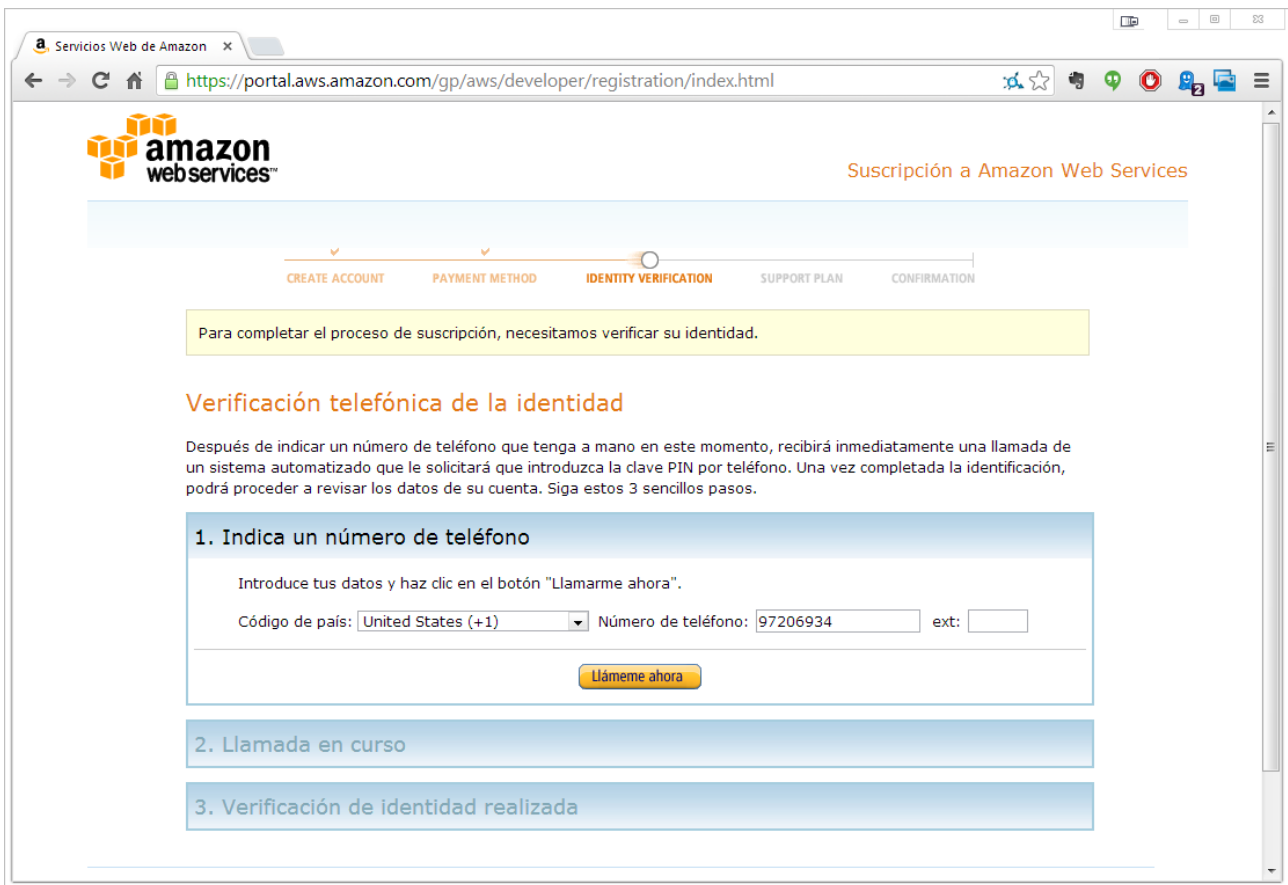


Ilustración 17 - Inscripción paso 05 de 06: Verificación de identidad

En este caso seguimos los pasos que nos van marcando, que básicamente consisten en introducir un número de teléfono al cual nos llamarán, durante la llama introducir un número que nos saldrá en la página y posteriormente presionar el botón “Continuar”. De esta forma llegaremos a la página para la selección de soporte técnico (Ilustración 18).



respuesta en 15 minutos e incluye el servicio de un gestor de cuentas y un experto en su caso, gestión de prioridad que avisa a su técnico de cuenta y al equipo de ingeniería cuando surgen problemas críticos.

Continuar

Ilustración 18 - Inscripción paso 06 de 06: Plan de soporte

Donde escogeremos si deseamos soporte para el servicio. Tras realizar la acción y presionar el botón continuar habremos finalizado el registro.

Ahora para acceder a la cuenta bastará con volver a la dirección web inicial (<https://aws.amazon.com/es/> - **Ilustración 11**) y clicar en el botón “Inscribirse” de nuevo.

En este caso seleccionaremos “I am a returning user and my password is” en vez de seleccionar “I am a new user” e introduciremos la clave y el correo electrónico especificados durante el registro. De esta forma, al darle al botón “Sign in using our secure server” accederemos a la cuenta.

Una vez autenticados se nos redirigirá a la página de gestión de cuenta, donde podemos configurar los métodos de pagos, los datos personales, los servicios a los que estamos suscritos, borrar la cuenta, entre otras opciones.

1.2 Configuración de doble autenticación

Para activar la autenticación de doble factor en Amazon AWS nos hemos de dirigir a la página: <https://aws.amazon.com/es/mfa/> y presionar el botón “Habilitar el dispositivo AWS MFA” tal como se muestra en la **Ilustración 19**.

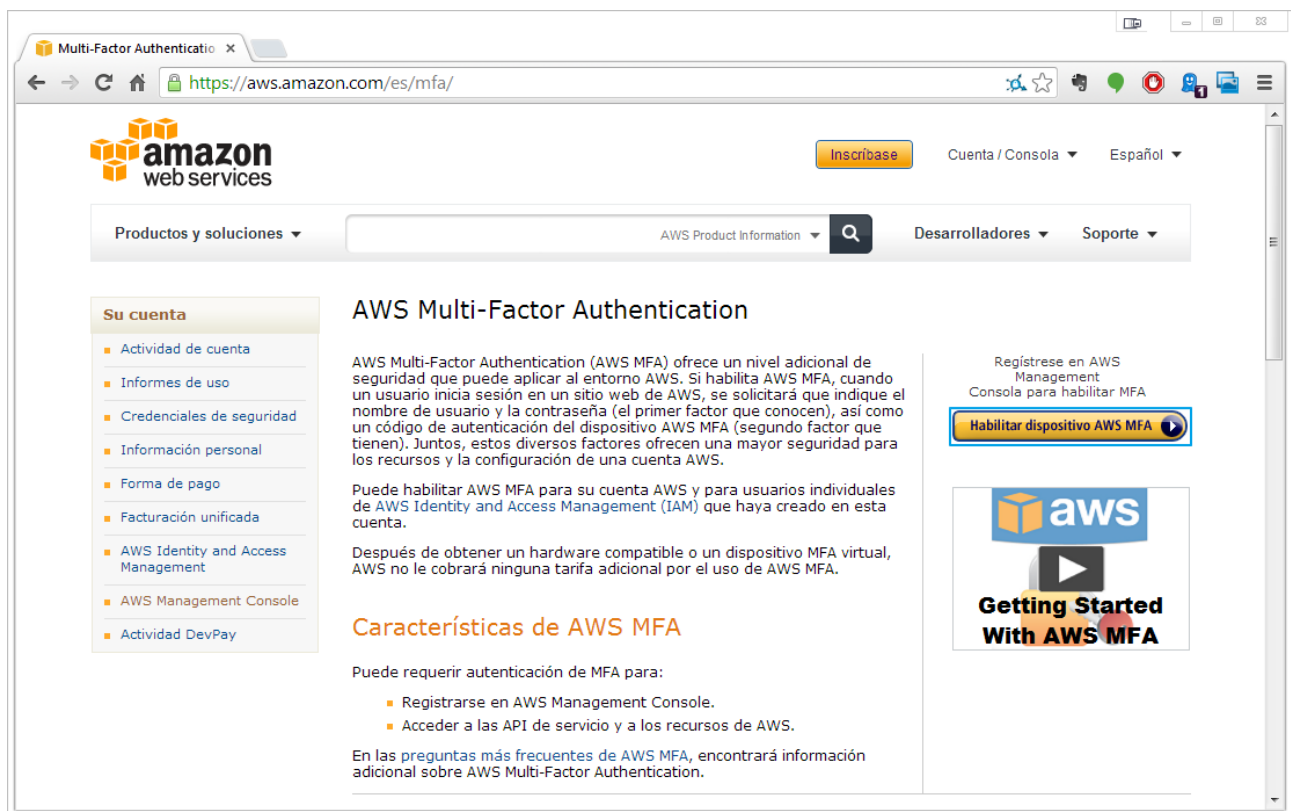


Ilustración 19 - Doble autenticación: Página de inicio

Tras presionar el botón nos pedirá que nos autentiquemos. Una vez autenticados nos redirigirá al panel de administración de los servicios. Allí, presionamos en el botón “Manage MFA Device” en el apartado “Security Status”.

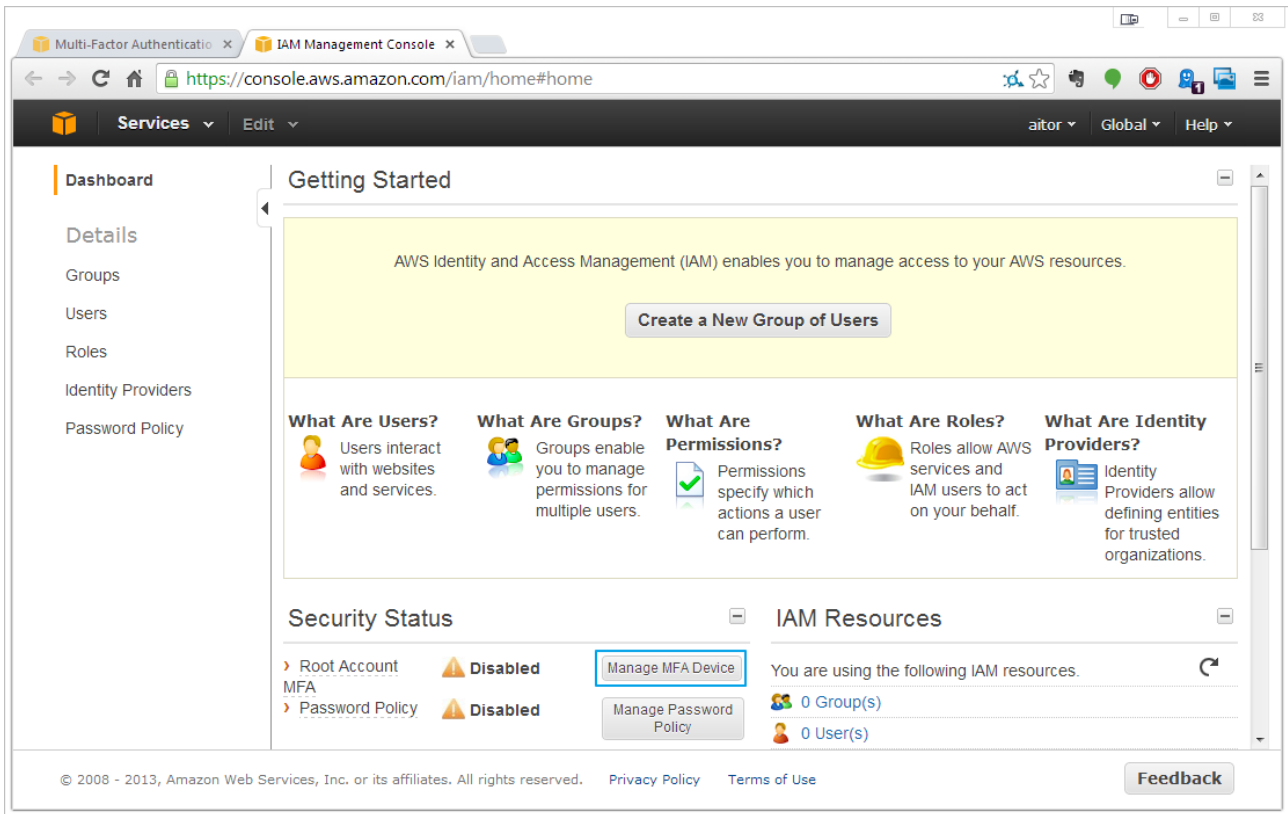


Ilustración 20 - Doble autenticación: Localización de la opción en el panel de control

Al presionar el botón se nos mostrará un pequeña ventana para que escojamos el tipo de dispositivo MFA que deseamos usar, tal como se muestra en la **Ilustración 21**.

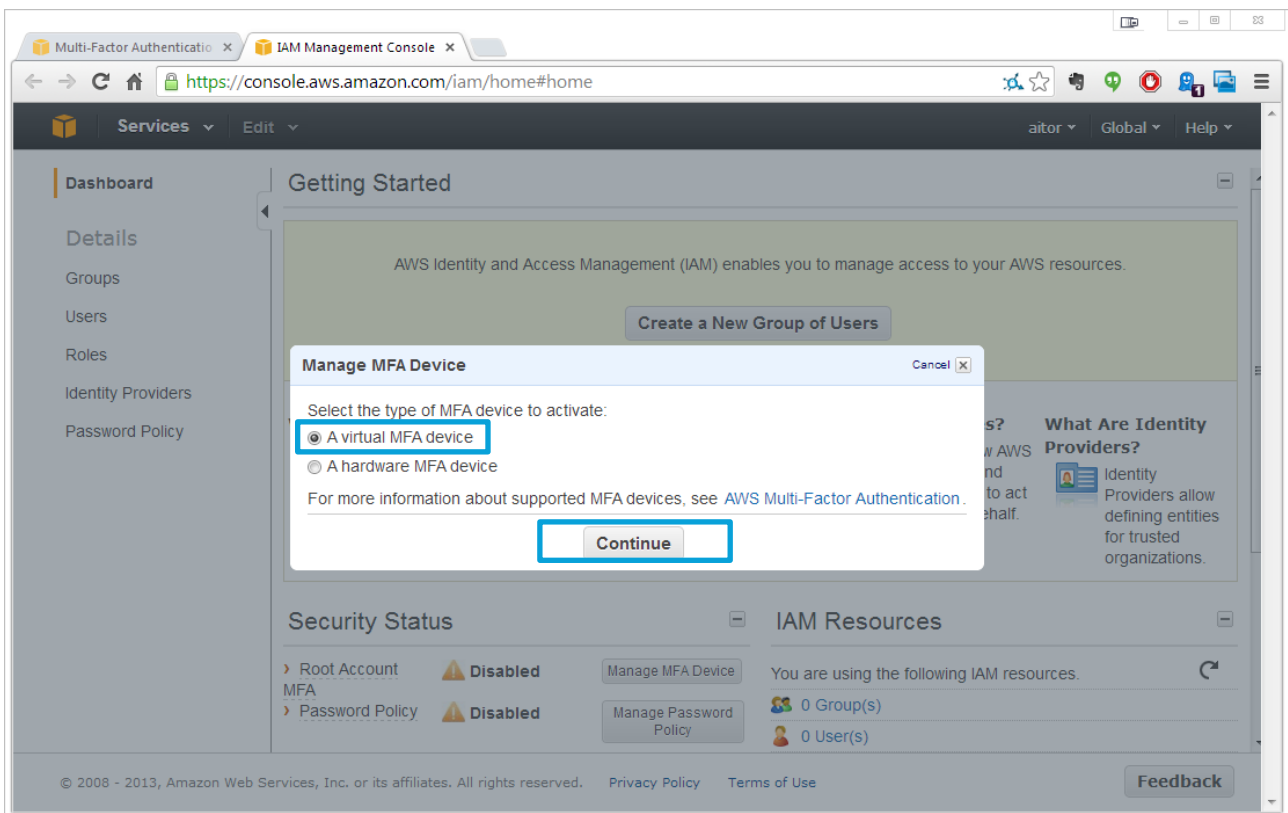


Ilustración 21 - Doble autenticación paso 01 de 04: Selección del tipo de dispositivo

En este caso optaremos por seleccionar la opción “A virtual MFA device” y presionar al botón “Continue”. Se nos mostrará un pequeño mensaje de advertencia (**Ilustración 22**). Tras leerlo presionamos continuar nuevamente.

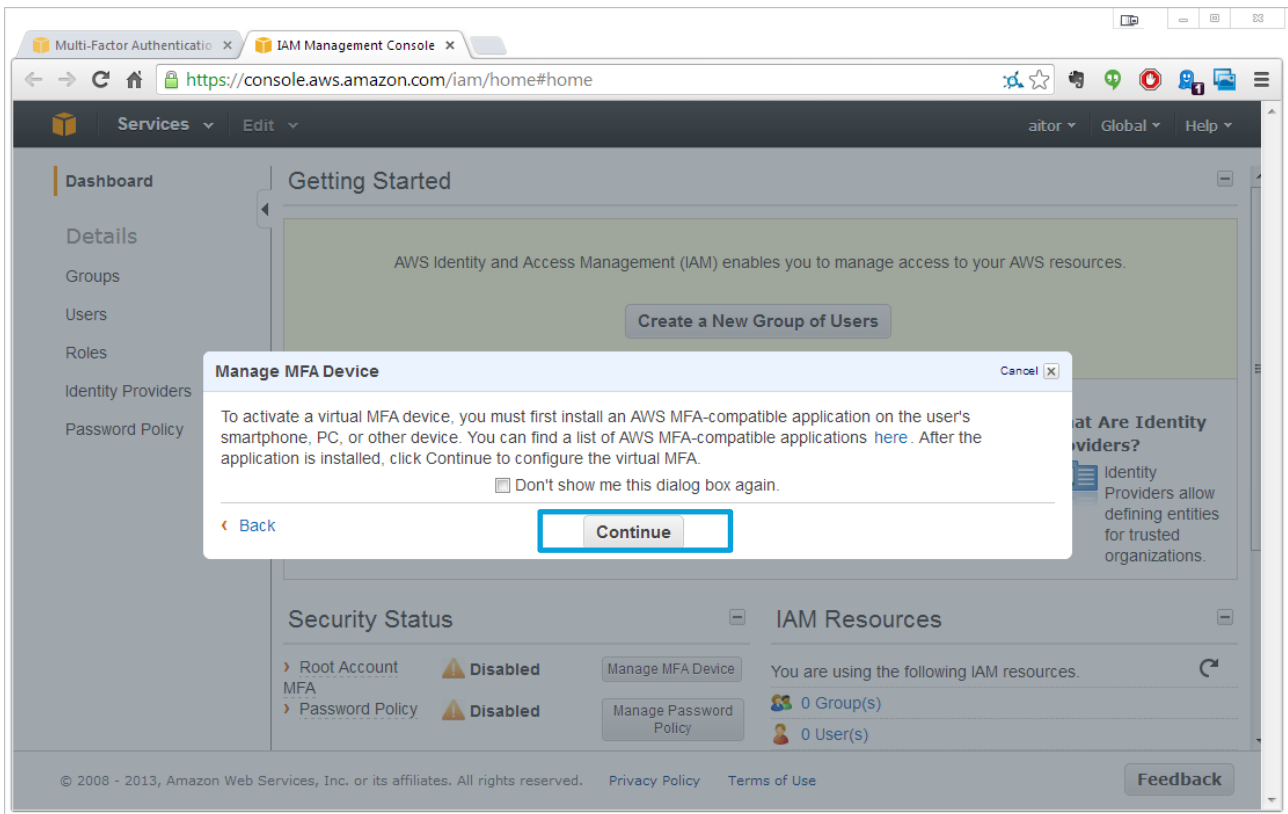


Ilustración 22 - Doble autenticación paso 02 de 04: Mensaje de instrucción

Ahora la ventana habrá cambiado y nos mostrará un código QR junto a dos campos vacíos: “Authentication Code 1” y “Authentication Code 2”. Podemos ver esta nueva ventana en la **Ilustración 23**.

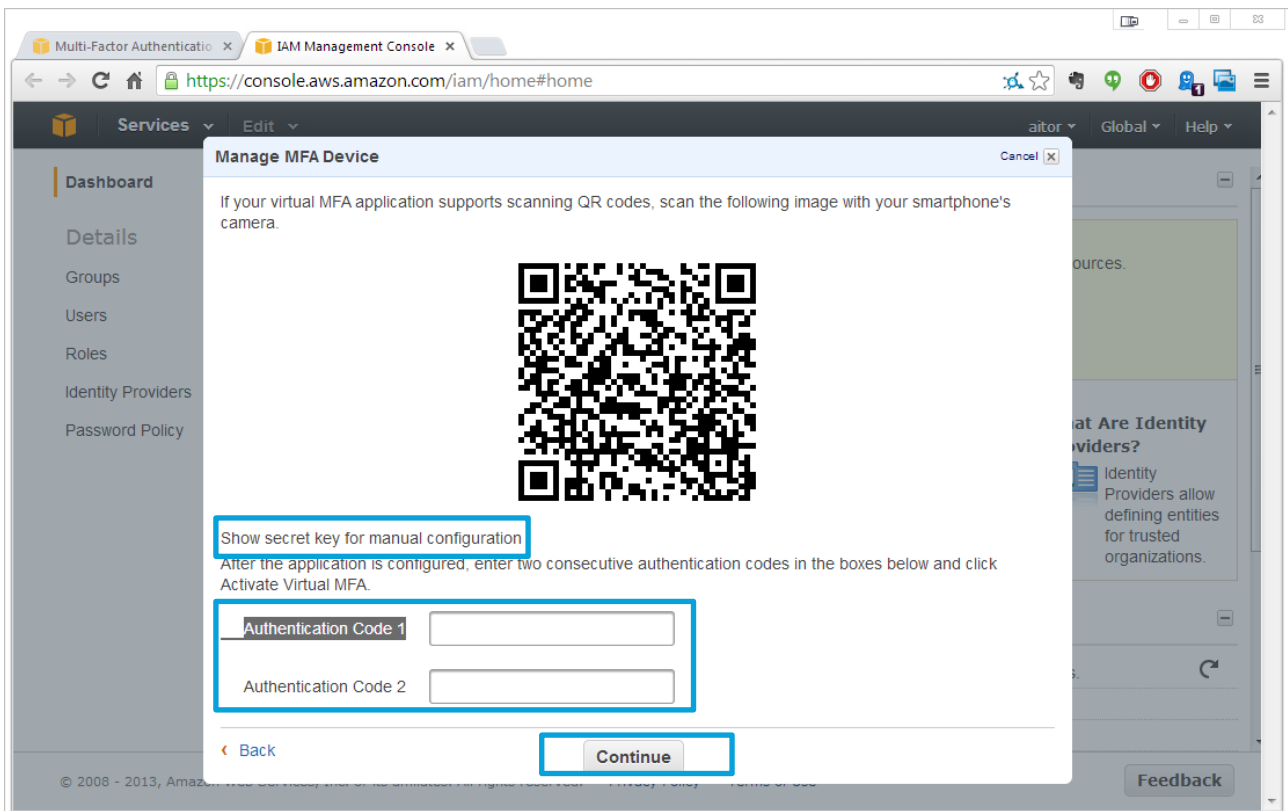


Ilustración 23 - Doble autenticación paso 03 de 04: Sincronización de los dispositivos

Aquí tenemos dos formas de proceder:

Leemos el código QR mediante la aplicación que usaremos para la doble autenticación o, le introducimos el código que se muestra al presionar “*Show secret key for manual configuration*” de forma manual. De esta manera la aplicación quedará configurada y podemos proceder a introducir los códigos en el formulario y continuar con el proceso al presionar “*Continue*”. Si todo ha ido correctamente saldrá un mensaje conforme todo ha salido correctamente (**Ilustración 24**).

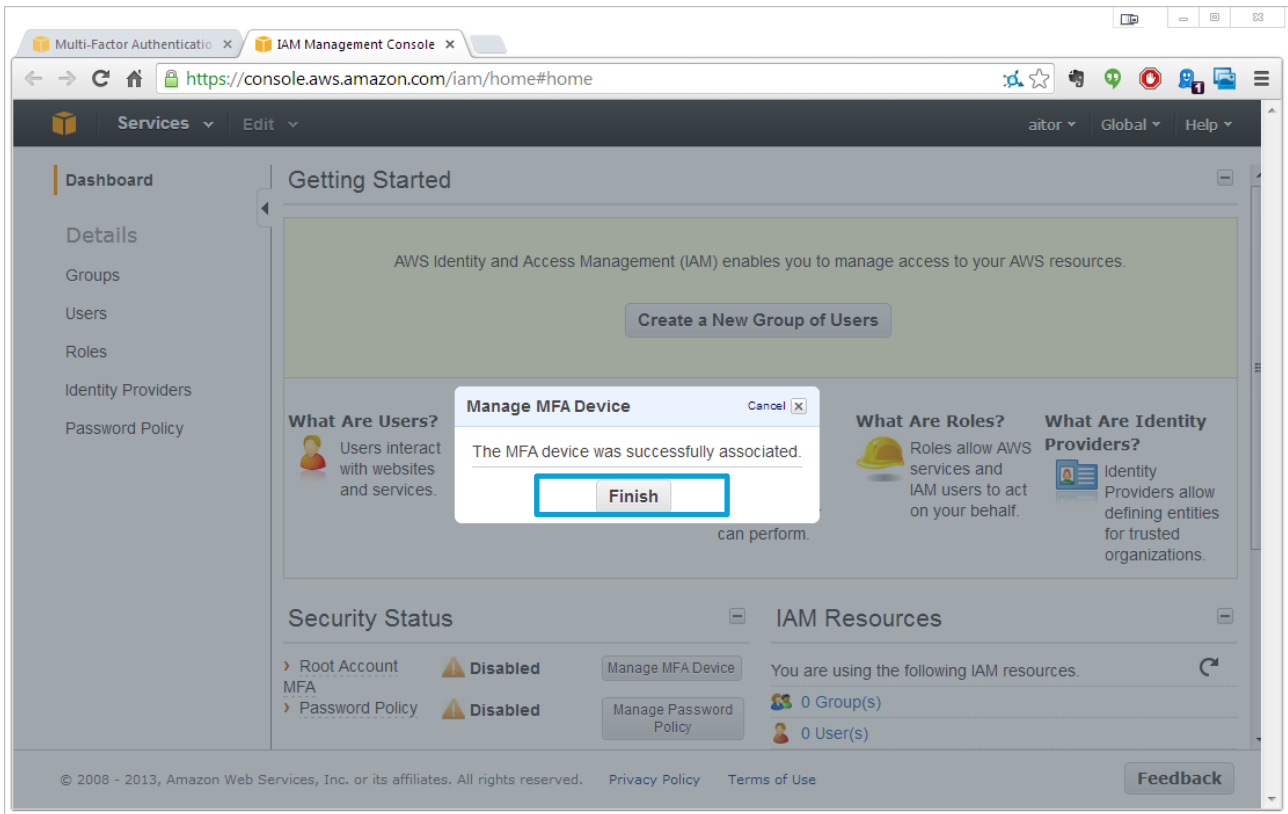


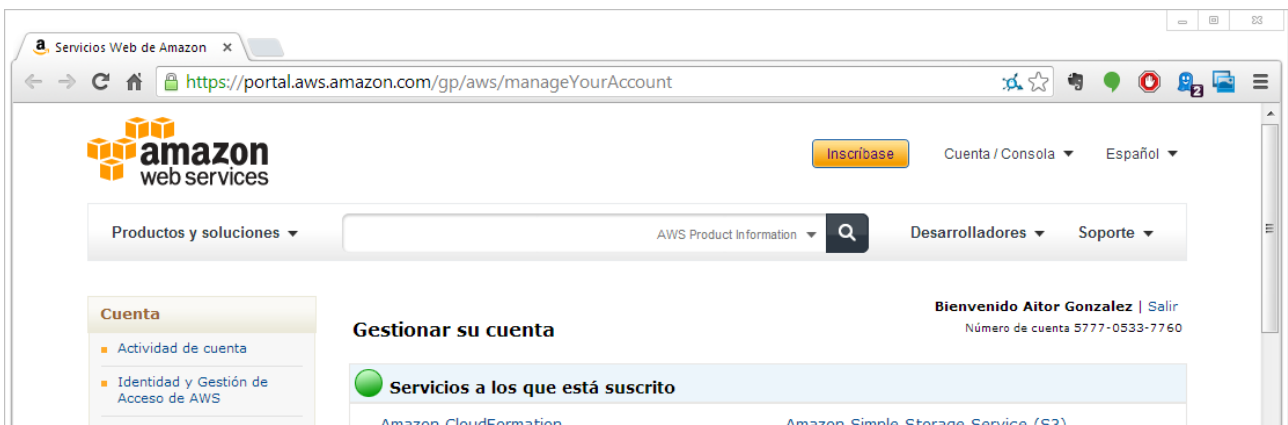
Ilustración 24 - Doble autenticación paso 04 de 04: Confirmación

Presionamos finalizar y ya dispondremos de doble autenticación para nuestra cuenta de Amazon AWS.

1.3 Creación de claves o certificados

Algunos servicios requieren de claves de acceso o certificados para su utilización desde *scripts*. Para crear estos procederemos de la siguiente forma:

1. Nos dirigiremos a la página de Amazon AWS y nos autenticaremos.
2. En el menú izquierdo presionaremos sobre “*Credenciales de seguridad*” tal como se muestra en la **Ilustración 25**.



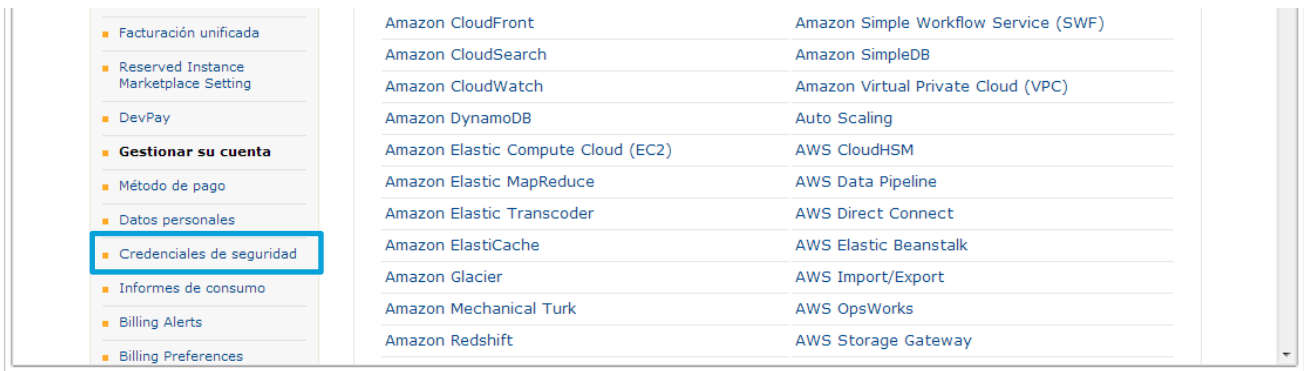


Ilustración 25 - Creación de clave/certificados: Inicio

- Ahora en la parte central de la página, nos situaremos en la sección credenciales de seguridad y según lo que necesitemos procederemos a crearlas (**Ilustración 26**).

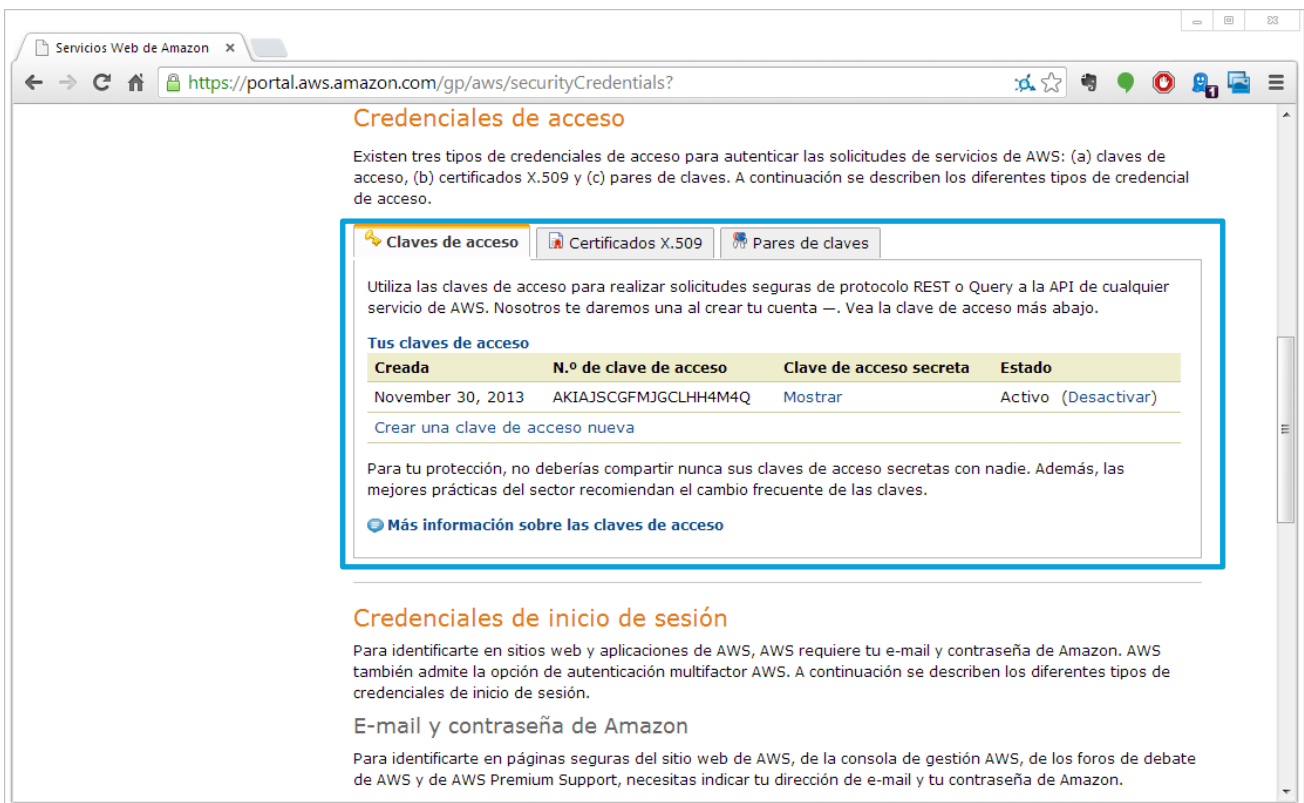


Ilustración 26 - Creación de clave/certificados: Sección de configuración

Una vez creadas es importante guardarlas en un lugar seguro, puesto que una vez consultadas el sistema no nos las volverá a mostrar.

2. Amazon AWS EC2 (Elastic Cloud)

EC2 es el servicio para la creación y gestión de instancias en la nube de Amazon AWS. En este apartado se describen aquellos pasos esenciales para comenzar con el servicio mediante los siguientes manuales: creación y eliminación instancias, creación e eliminación de claves, creación y eliminación de imágenes de máquina, gestión del acceso, gestión de IPs estáticas, y, por último, ampliación del EBS por defecto.

No se contemplan lo referente a creación y eliminación de volúmenes EBS ya que para el proyecto presentada suponen una pequeña restricción para el escalado.

2.1 Creación de instancias

Para crear una instancia nos hemos de autenticar en los servicios de Amazon AWS. Una vez autenticados y en la página de gestión de cuenta presionaremos en el botón “Cuenta/Consola” resaltado en la **Ilustración 27**.

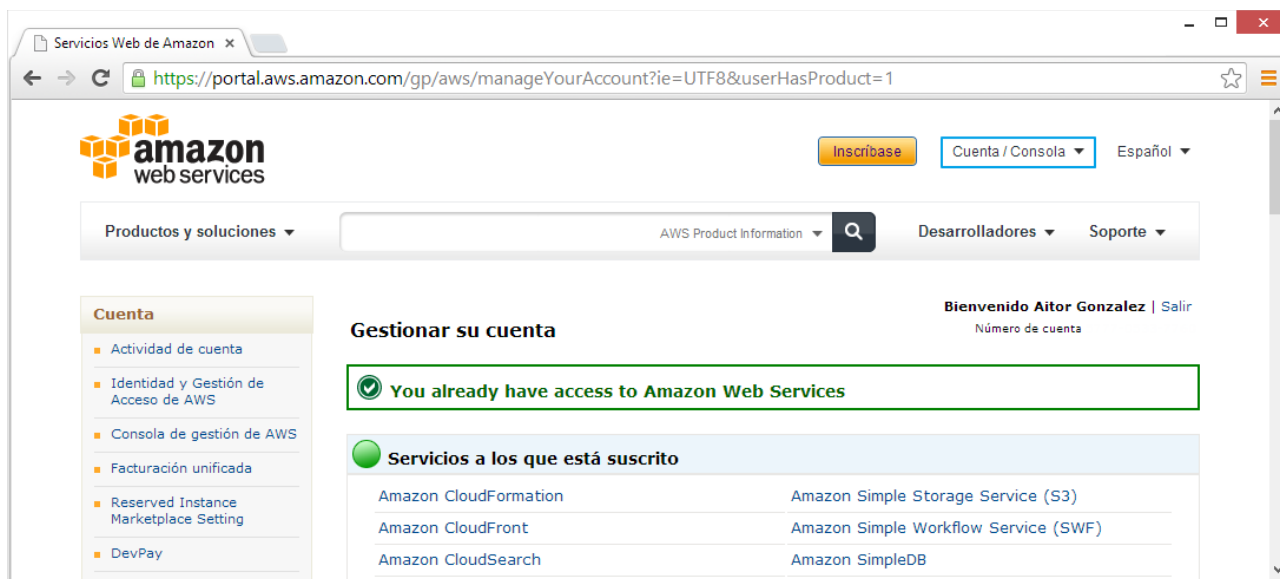


Ilustración 27 - EC2 Creación de instancia: Inicio

Tras presionar en el botón nos saldrá un menú desplegable en el cual tendremos que seleccionar “AWS Management Console”. De esta manera el sistema nos redirigirá a la página de administración de los servicios (**Ilustración 28**). Allí solo hemos de seleccionar aquel servicio que deseemos configurar, en este caso, EC2.

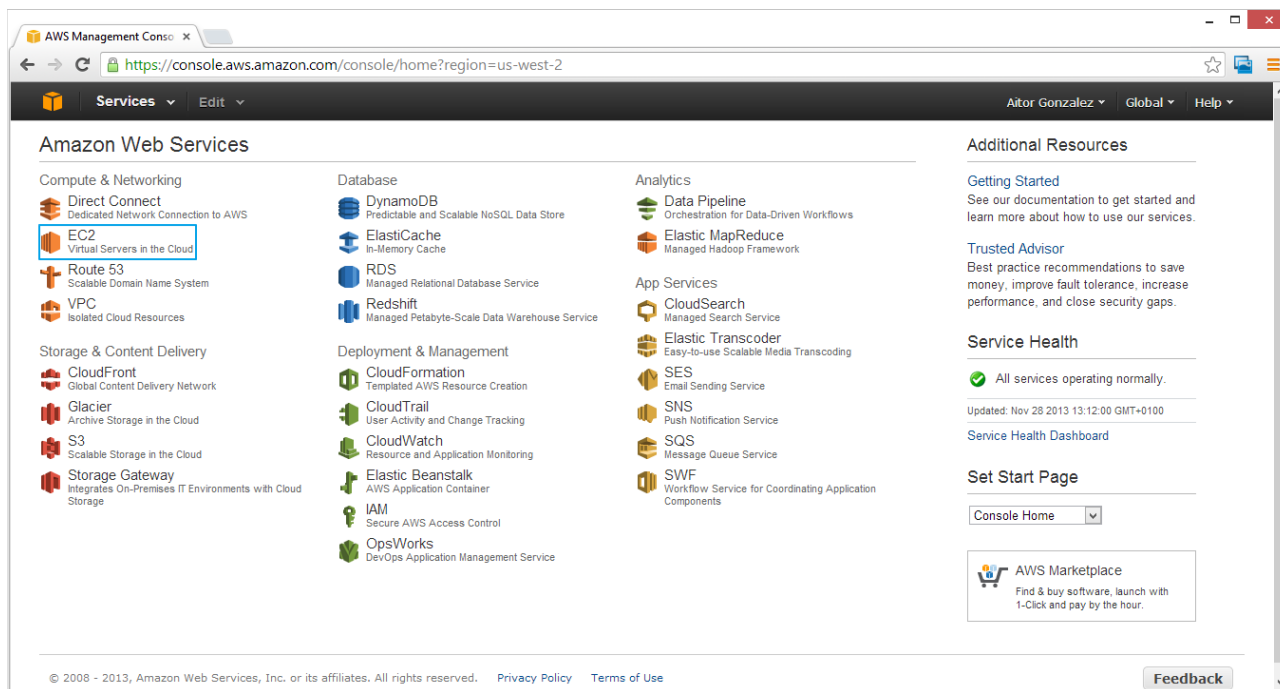


Ilustración 28 - EC2 Creación de instancia: Panel principal AWS Management.

Al presionar en EC2 iremos directamente a la página de inicio de dicho servicio, donde se nos presenta un resumen general del estado de este y los recursos que estamos usando de él. Es importante en este instante cambiar la región donde en un futuro se crearán las instancias, puesto que a posterior no podemos mover una instancia de región. Además, cuanto más cercana la creamos a nuestra ubicación o clientes menor será el coste del servicio y mayor su calidad.

Para cambiar la región basta con que pulsemos en la esquina superior derecha donde pone “Origen” y seleccionar la que más nos convenga, tal como se muestra en la **Ilustración 29**.

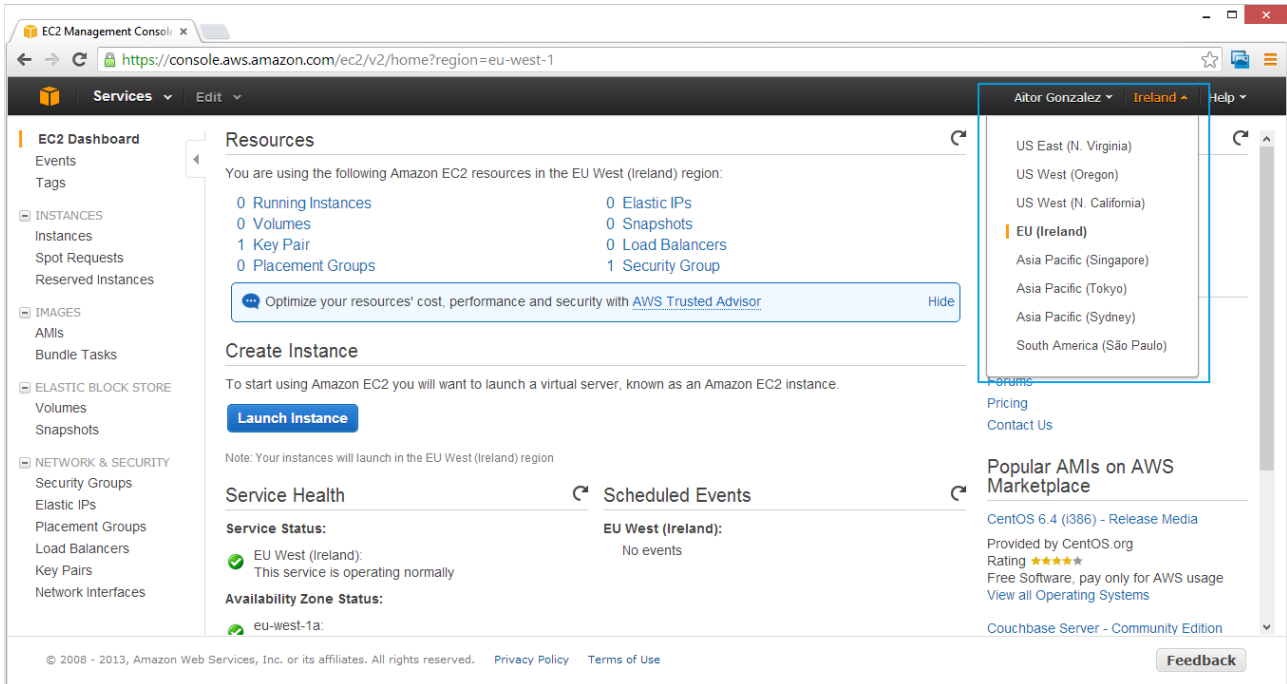


Ilustración 29 - EC2 Creación de instancia: Cambio de región

Una vez seleccionada la región presionaremos en el botón “Launch Instance” (**Ilustración 30**) en el centro de la página para iniciar una nueva instancia.

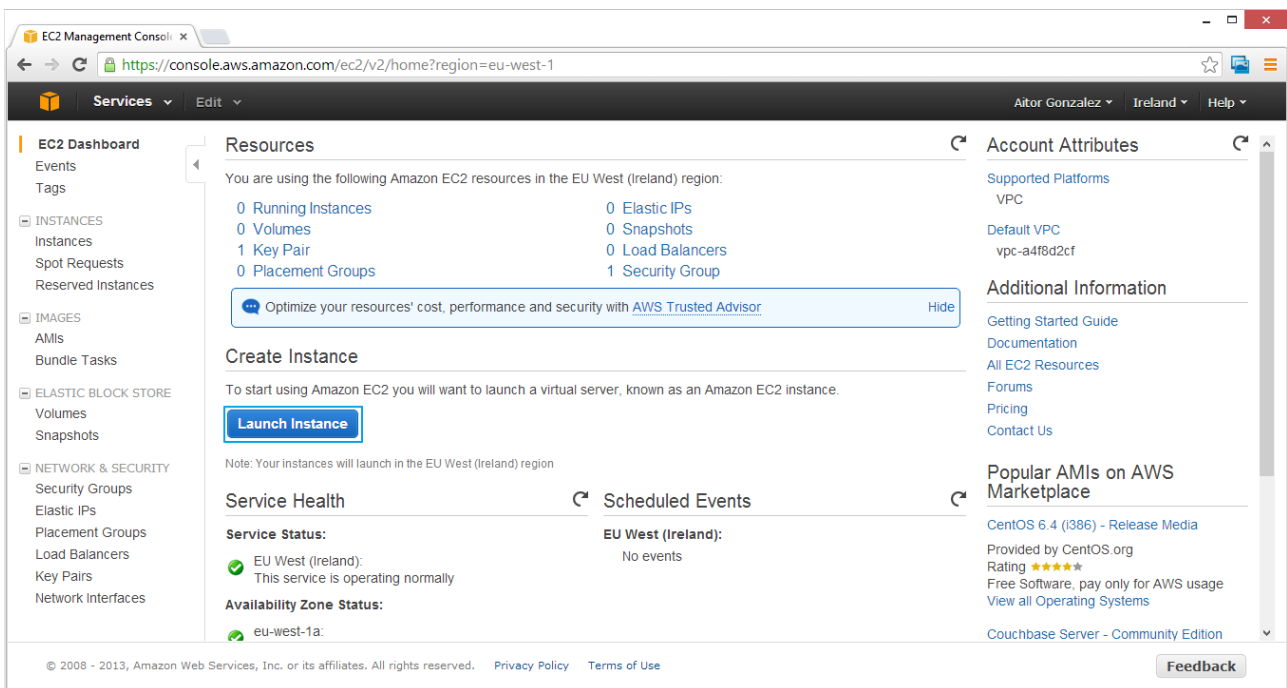


Ilustración 30 - EC2 Creación de instancia paso 01 de 06: Iniciando el proceso

A partir de aquí solo hace falta seguir los pasos que nos marca Amazon. Estos pasos son:

1. **Selección de la AMI (Amazon Machine Image - Ilustración 31):** hemos de seleccionar la imagen de máquina que más se adapta a nuestras necesidades. Podemos escoger entre imágenes de sistemas operativos completamente limpias o con *software* preinstalado. Al darle a “Select” pasaremos directamente a la siguiente etapa del proceso.

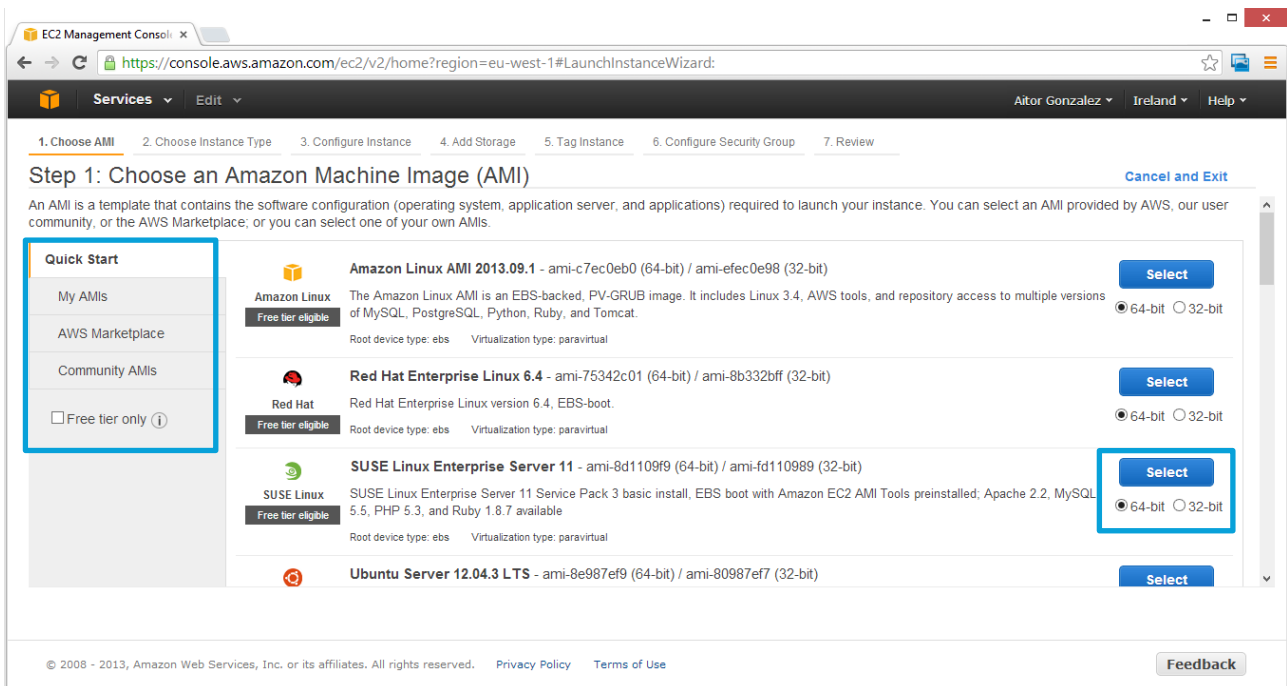


Ilustración 31 - EC2 Creación de instancia paso 02 de 06: Selección de AMI

2. **Selección del tipo de instancia (Ilustración 32):** tal como indica el nombre, en este caso, hemos de seleccionar la instancia que más se adapte a nuestros propósitos. Bastará con presionar en el menú izquierdo la opción deseada para que se nos muestre en el centro de la pantalla una tabla con el tipo de instancias disponibles. Tras seleccionar en la tabla la instancia y posteriormente presionar el botón “Review and Launch” pasaremos a la siguiente fase.

Si presionáramos en “Next: configuration detail” podremos configurar más parámetros de la instancia. En esta sección no se describe la configuración de estos, puesto que hay otros manuales en el anexo que cubren esta posibilidad una vez creada esta.

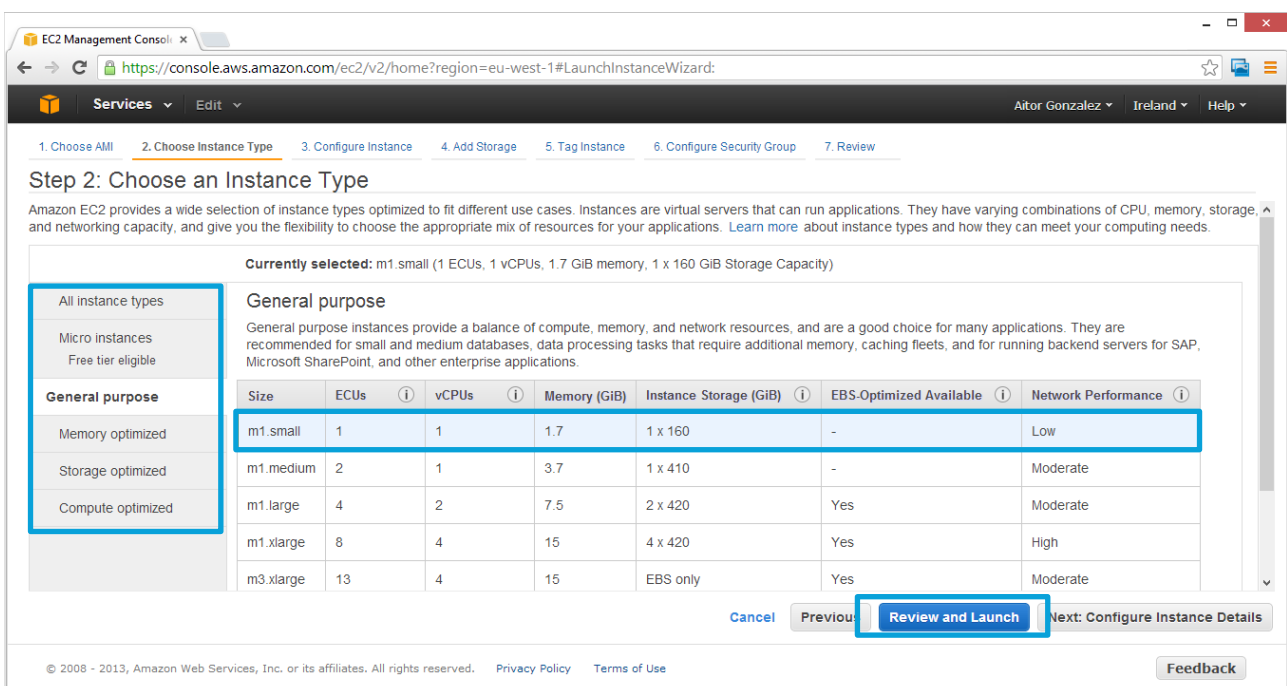


Ilustración 32 - EC2 Creación de instancia paso 03 de 06: Selección de tipo de instancia

3. **Review (Ilustración 33):** se nos muestra información general sobre la configuración de la instancia. Tras revisar que todo está correcto tendremos que presionar “Launch”.

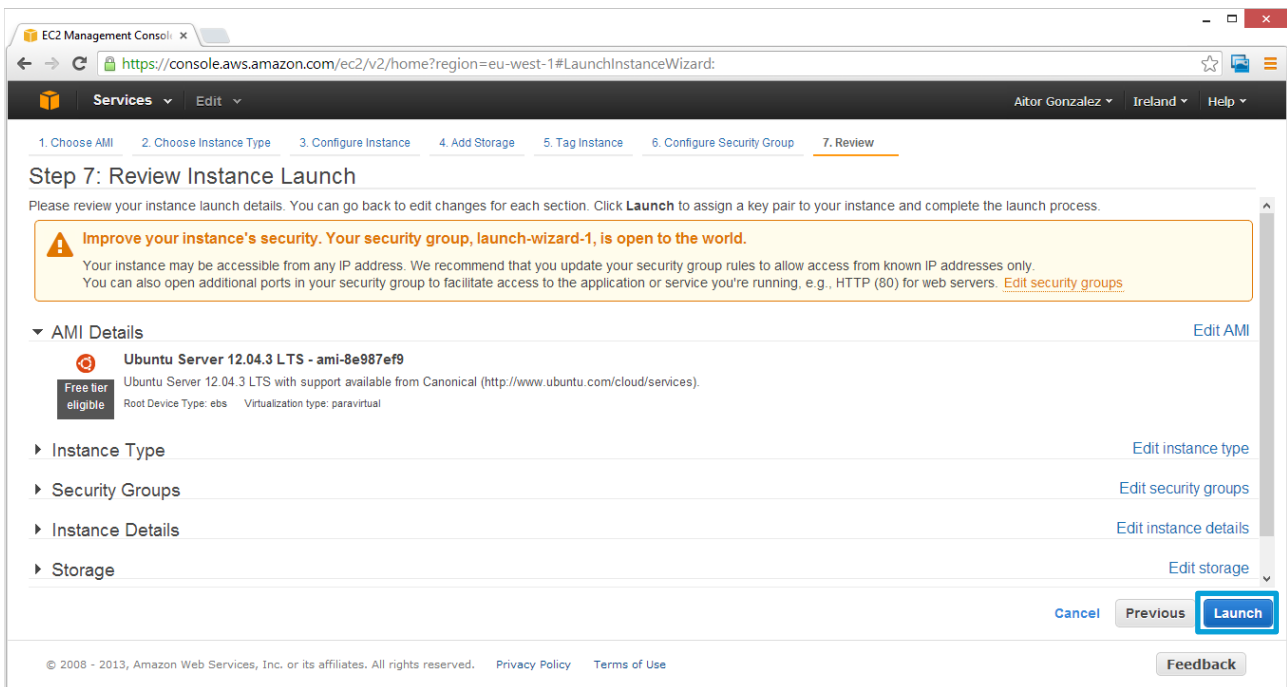


Ilustración 33 - EC2 Creación de instancia paso 04 de 06: Descripción general del sistema a montar

Al presionarlo se nos mostrará una ventana para que seleccionemos o creamos una “Key Pair”, la cual será usada para acceder mediante SSH (*Secure Shell*) a la instancia creada (**Ilustración 34**).

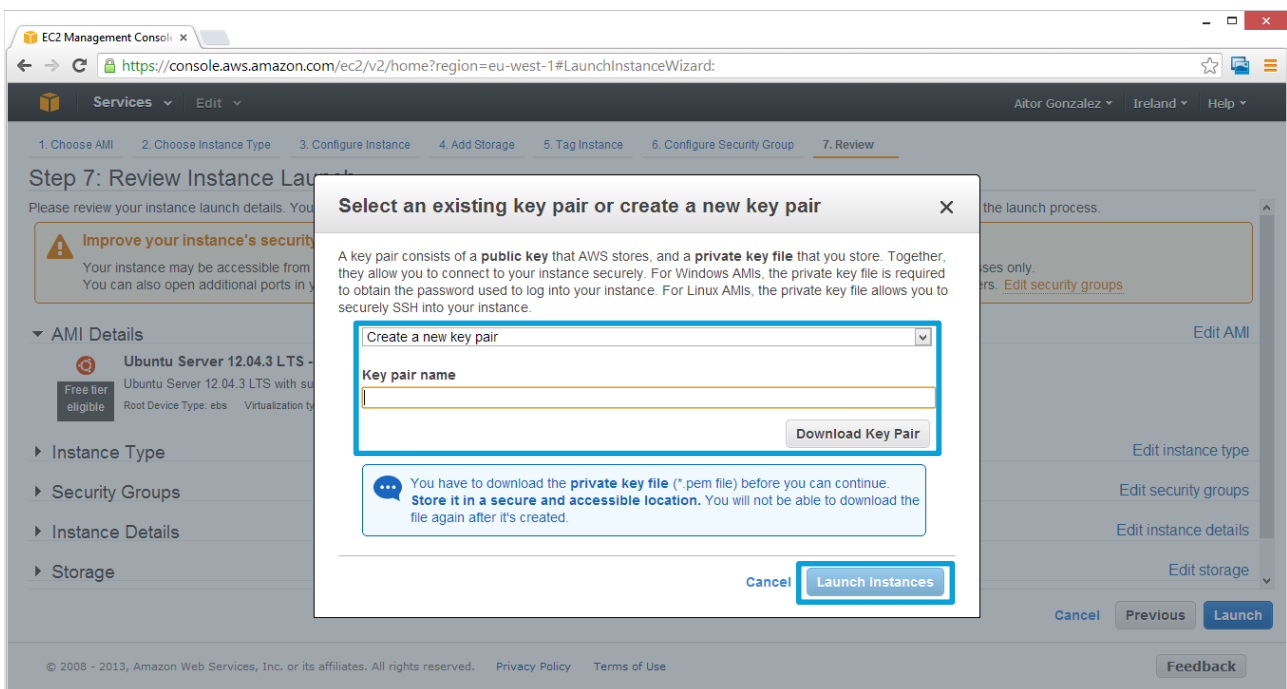


Ilustración 34 - EC2 Creación de instancia paso 05 de 06: Configuración key pair

Una vez seleccionada la una “Key Pair” existente o la creada, presionaremos “Launch Instances” y la instancia será creada. Se nos mostrará una confirmación conforme todo el proceso ha sido un éxito tal como se muestra en la **Ilustración 35**.

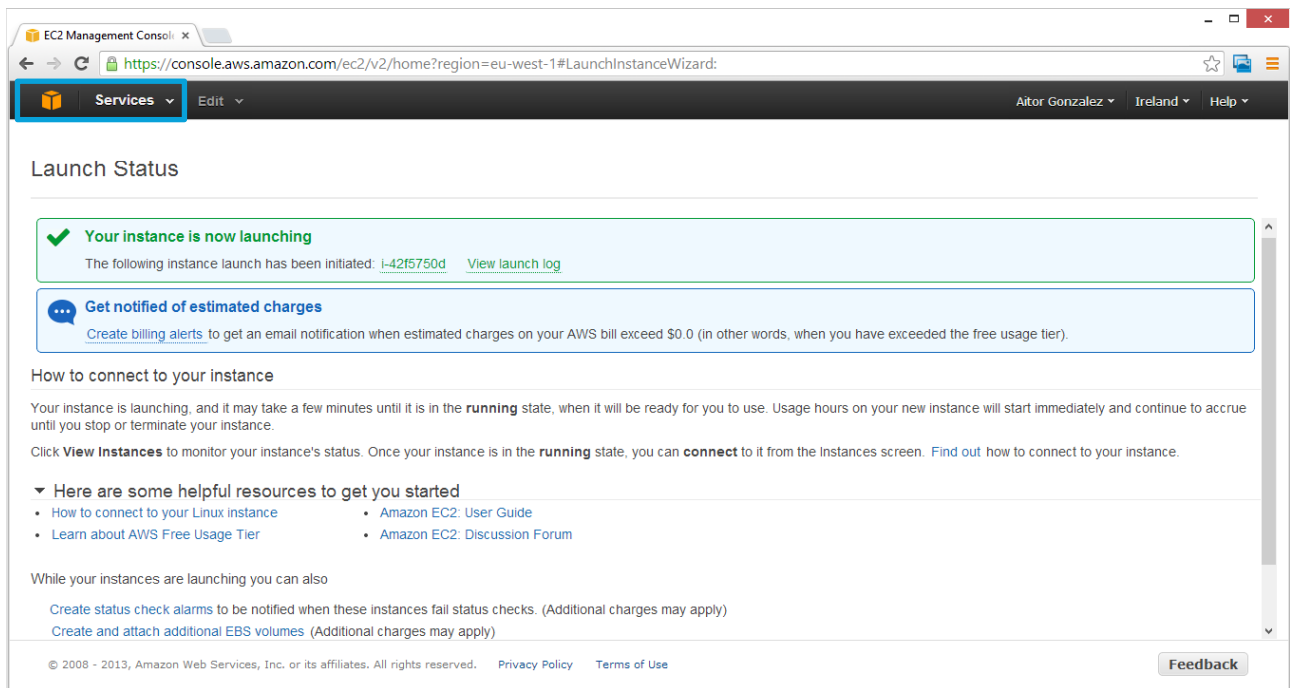


Ilustración 35 - EC2 Creación de instancia paso 06 de 06: Confirmación

Ahora solo falta dirigirnos al servicio mediante el menú “Services” (**Ilustración 35**) de la parte superior izquierda y presionar en el menú lateral “Instances” para ver que esta se ha creado correctamente y está en funcionamiento. En la Ilustración 36 se muestra un ejemplo de cómo se deberían visualizar.

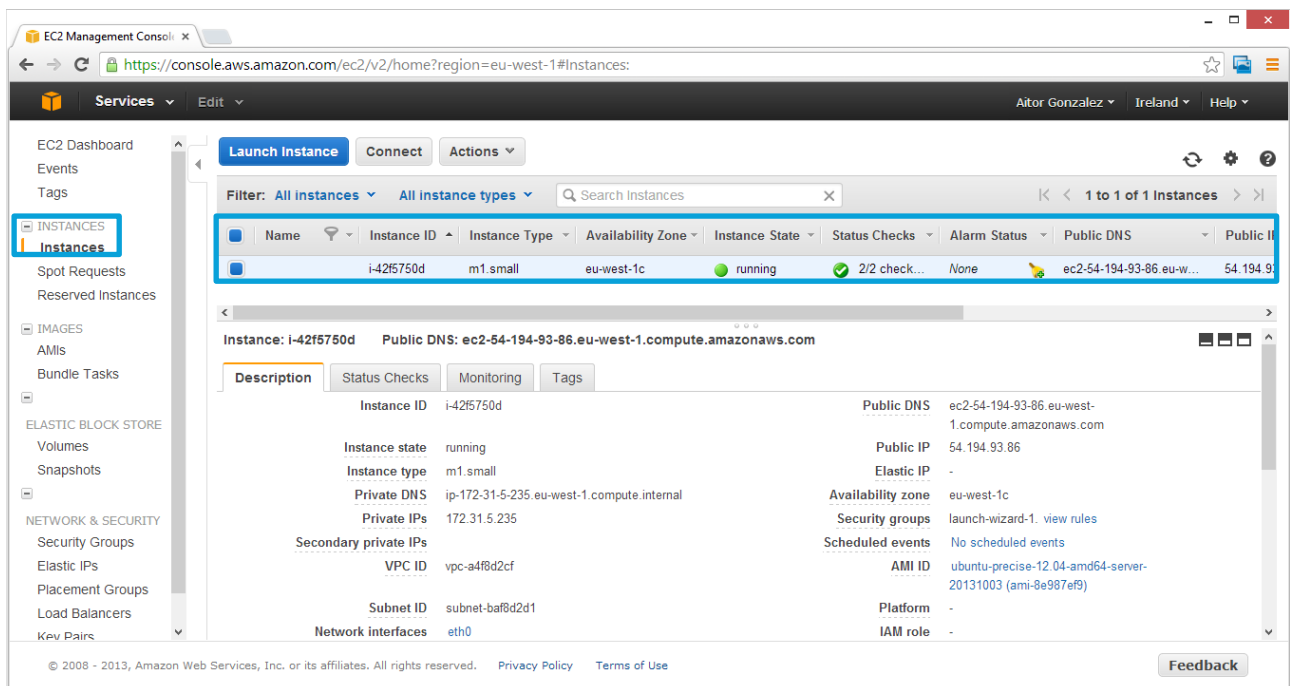


Ilustración 36 - EC2 Creación de instancia: Visualización

2.2 Eliminación de instancias

Para eliminar una instancia bastará con que nos dirijamos al panel de instancias, cliquemos botón derecho en la instancia que deseamos eliminar y presionemos “Terminate” tal como se muestra en la **Ilustración 37**.

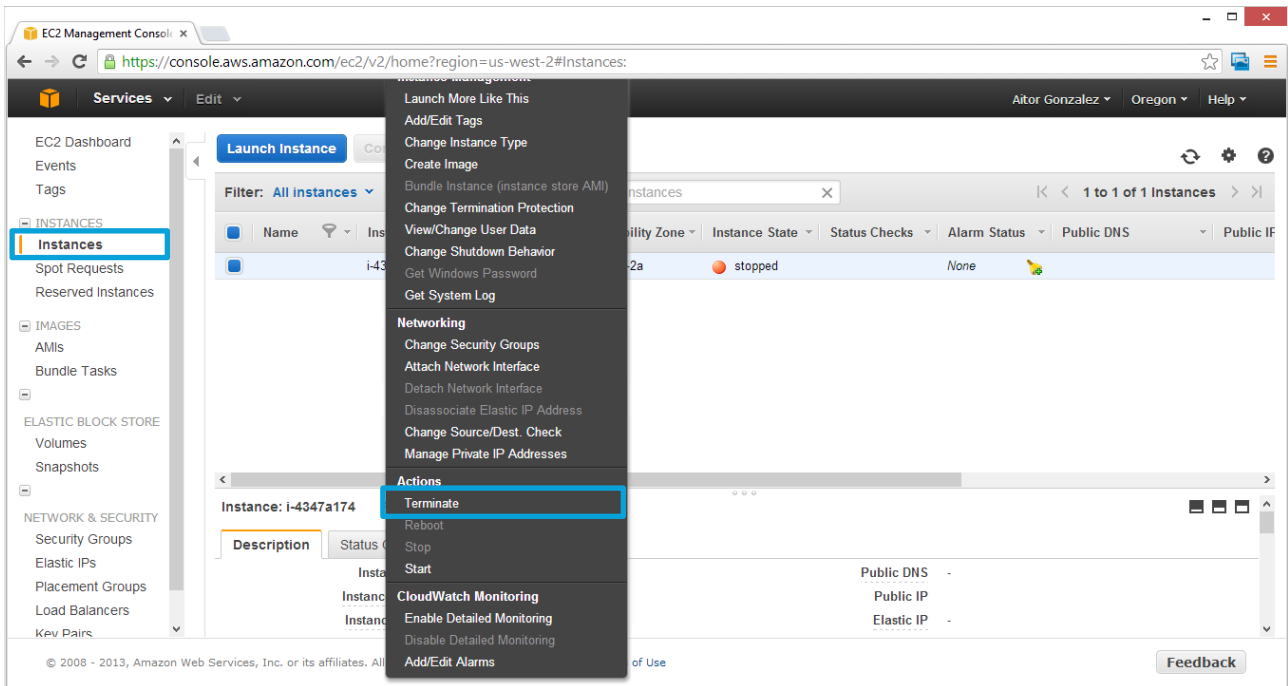


Ilustración 37 - EC2 Eliminación de instancia paso 01 de 02: Selección de instancia

Tras presionar el botón nos saldrá un mensaje de aviso (**Ilustración 38**) que nos informa que si eliminamos la instancia se eliminarán aquellos volúmenes EBS asociados como *root* a esta. Si estamos de acuerdo presionamos “Yes, Terminate”, en caso contrario presionaremos “Cancel” y proceso quedará cancelado quedando todo como estaba.

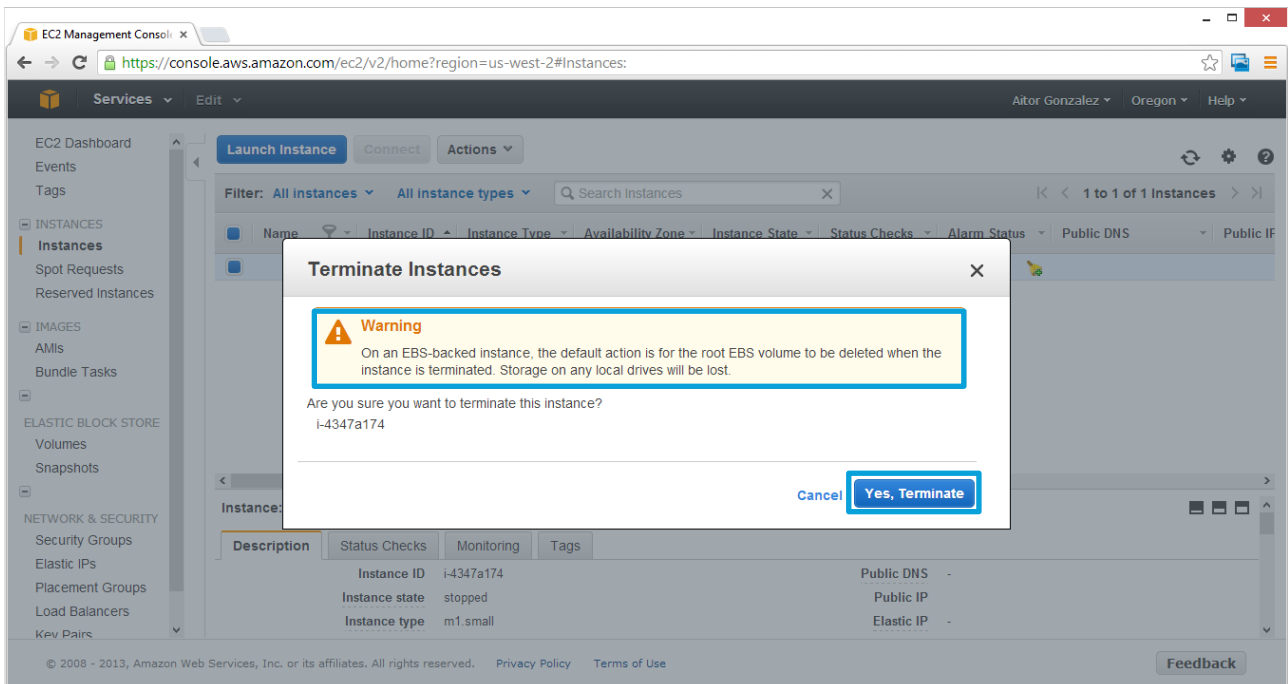


Ilustración 38 - EC2 Eliminación de instancia paso 02 de 02: Aviso y confirmación

Al presionar “Yes, Terminate” esta se pondrá en “Instance State: terminated” y con el tiempo acabará desapareciendo de la tabla. La **Ilustración 39** muestra el resultado final de todo el proceso.

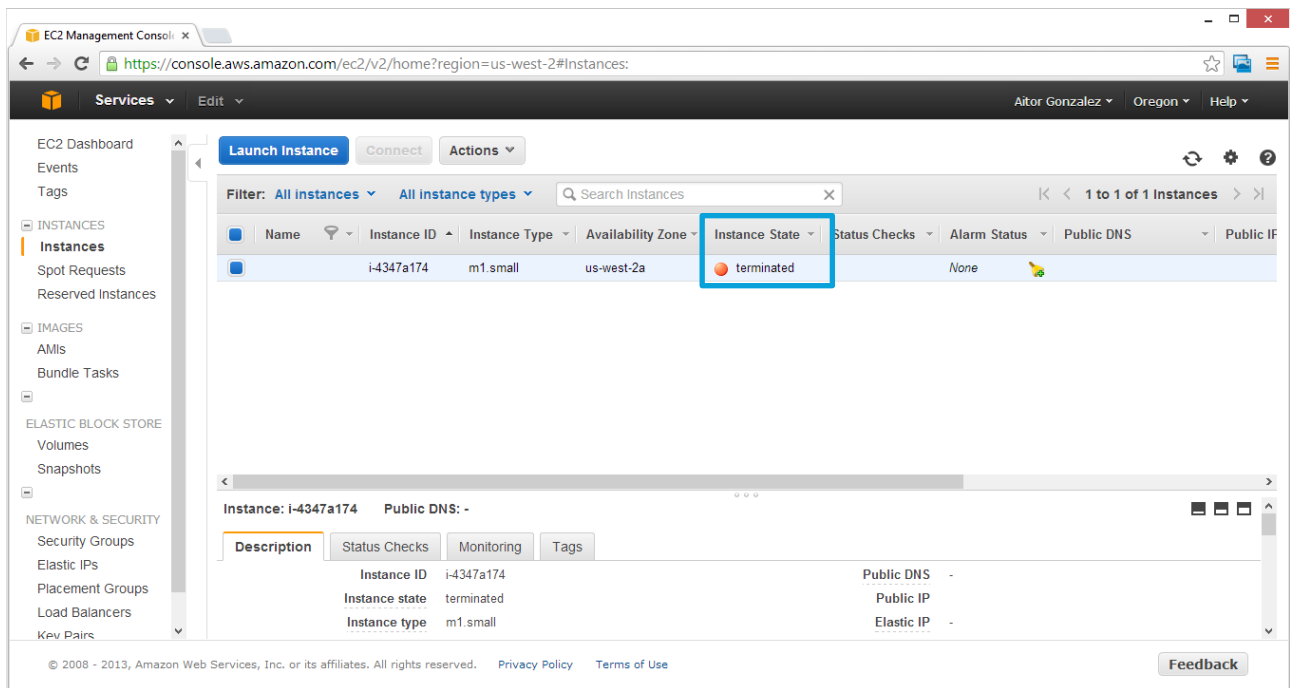


Ilustración 39 - EC2 Eliminación de instancia: Comprobación

2.3 Creación de claves (Key Pairs)

La Creación de una “Key Pair” se puede hacer de dos formas distintas: importando una existente o creando una directamente en el sistema. La creación de una nueva “Key Pair”, sea por el método que sea, solo nos será útil en el momento de crear nuevas instancias. Además, las creadas no se pueden descargar.

Para crear una desde el servicio bastará con dirigirnos al menú izquierdo y presionar sobre la opción “Key Pairs” en la sección “Network and Security” (**Ilustración 40**).

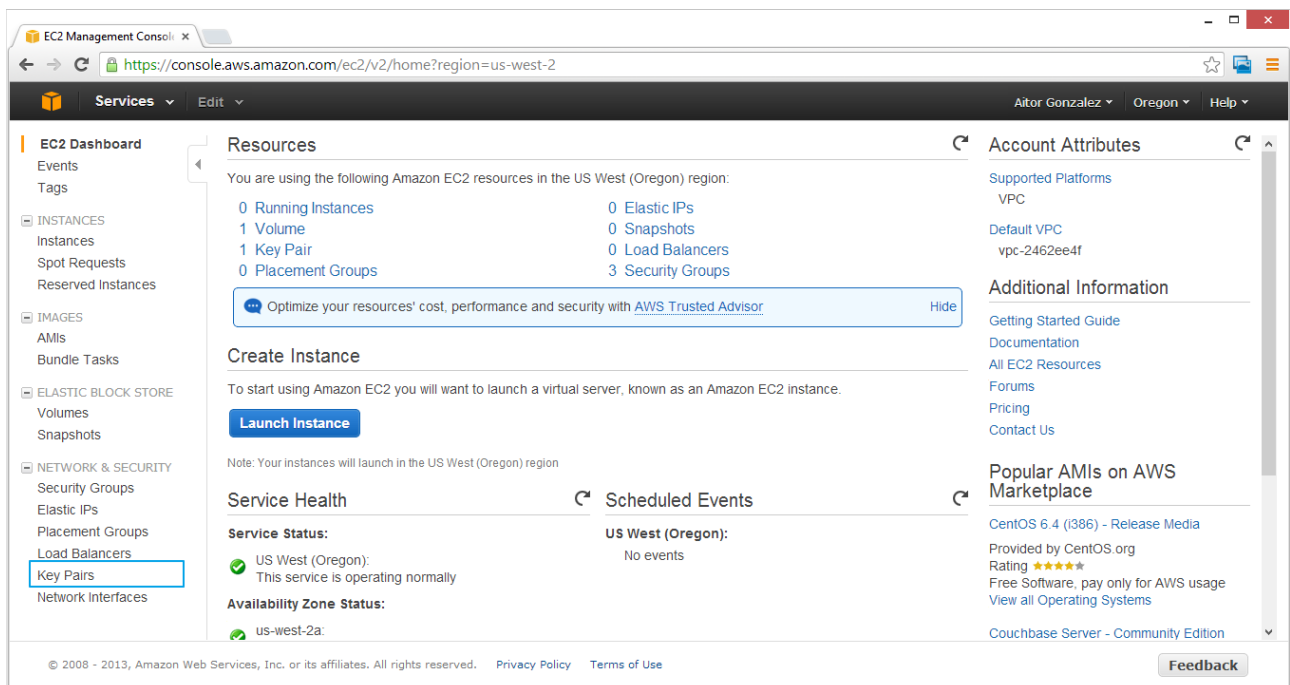


Ilustración 40 - EC2 Creación de claves: Inicio

Una vez seleccionada dicha opción y cargada la interfaz de gestión de claves escogeremos la opción que más nos conviene presionando en el botón “Create Key Pair” o “Import Key Pair” (**Ilustración 41**).

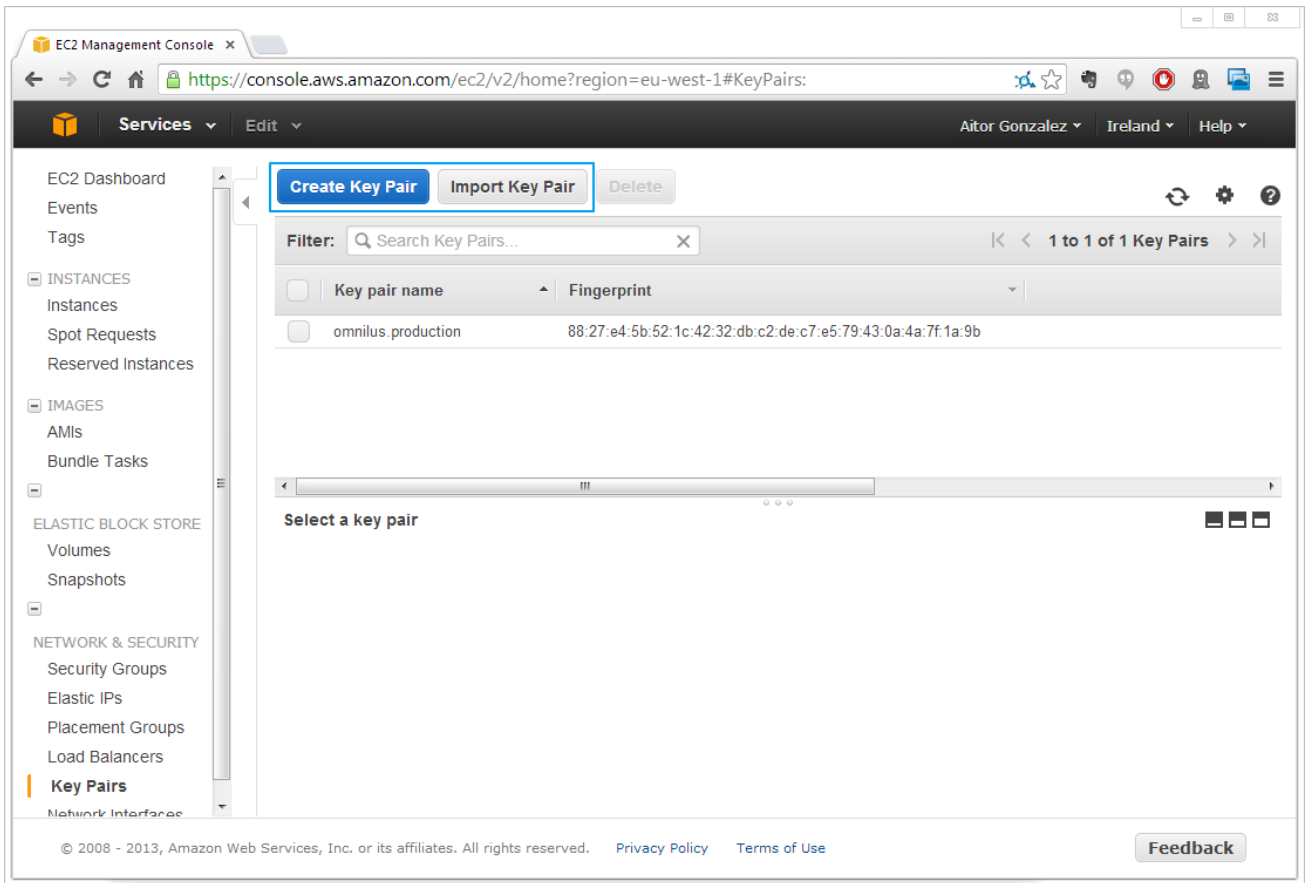


Ilustración 41 - EC2 Creación de claves: Interfaz de gestión

En el caso de que escojamos “Create Key Pair” nos mostrará un formulario (Ilustración 42) para que introduzcamos el nombre de la nueva llave.

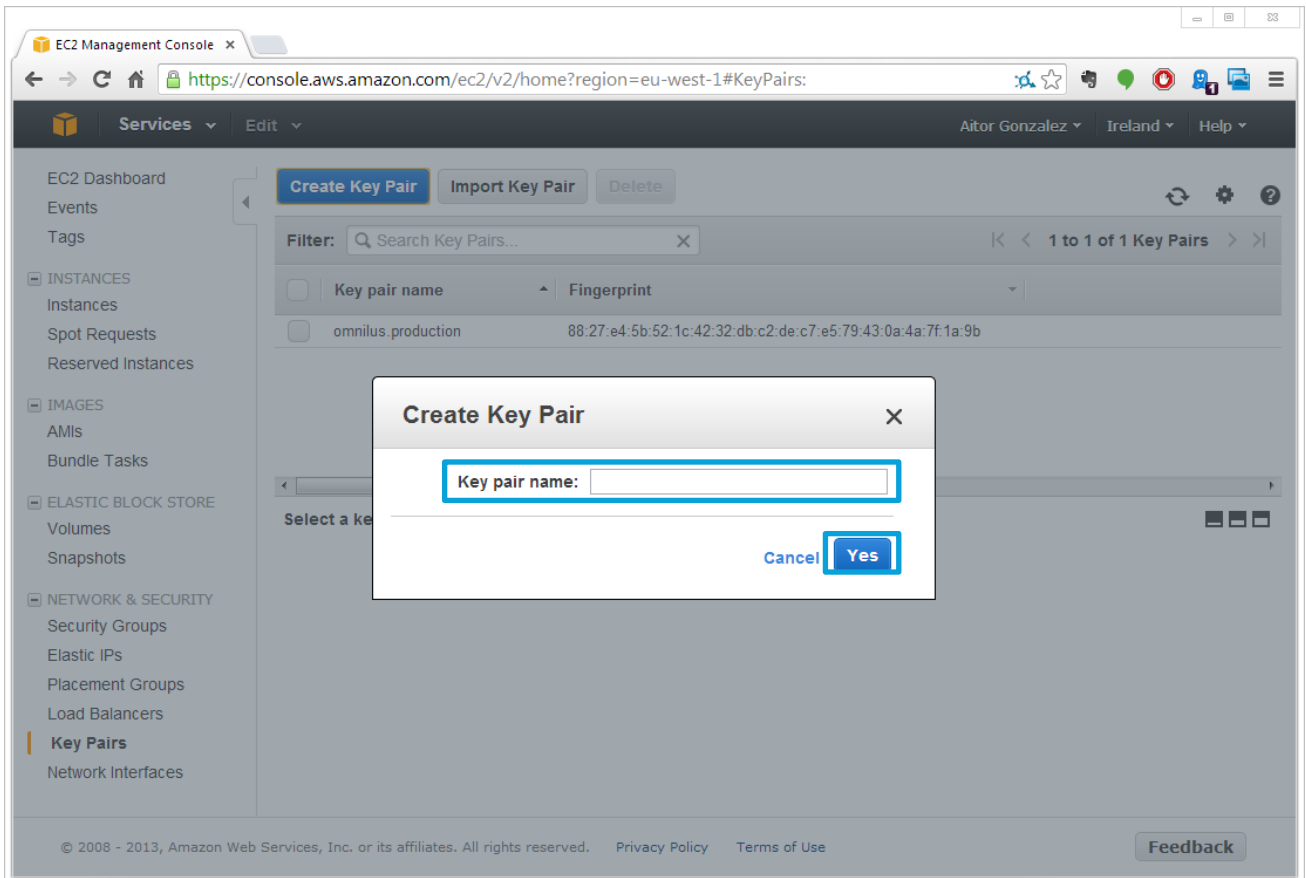


Ilustración 42 - EC2 Creación de claves: Formulario de creación de Key Pair

Al introducir el nombre y presionar “Yes” automáticamente se creará y se iniciará su descarga en el ordenador.

Si la opción escogida es importar una llave, se nos mostrará otro formulario (**Ilustración 43**) para la importación. Tras importarla y presionar “Import” esta será visible en la tabla de claves existentes.

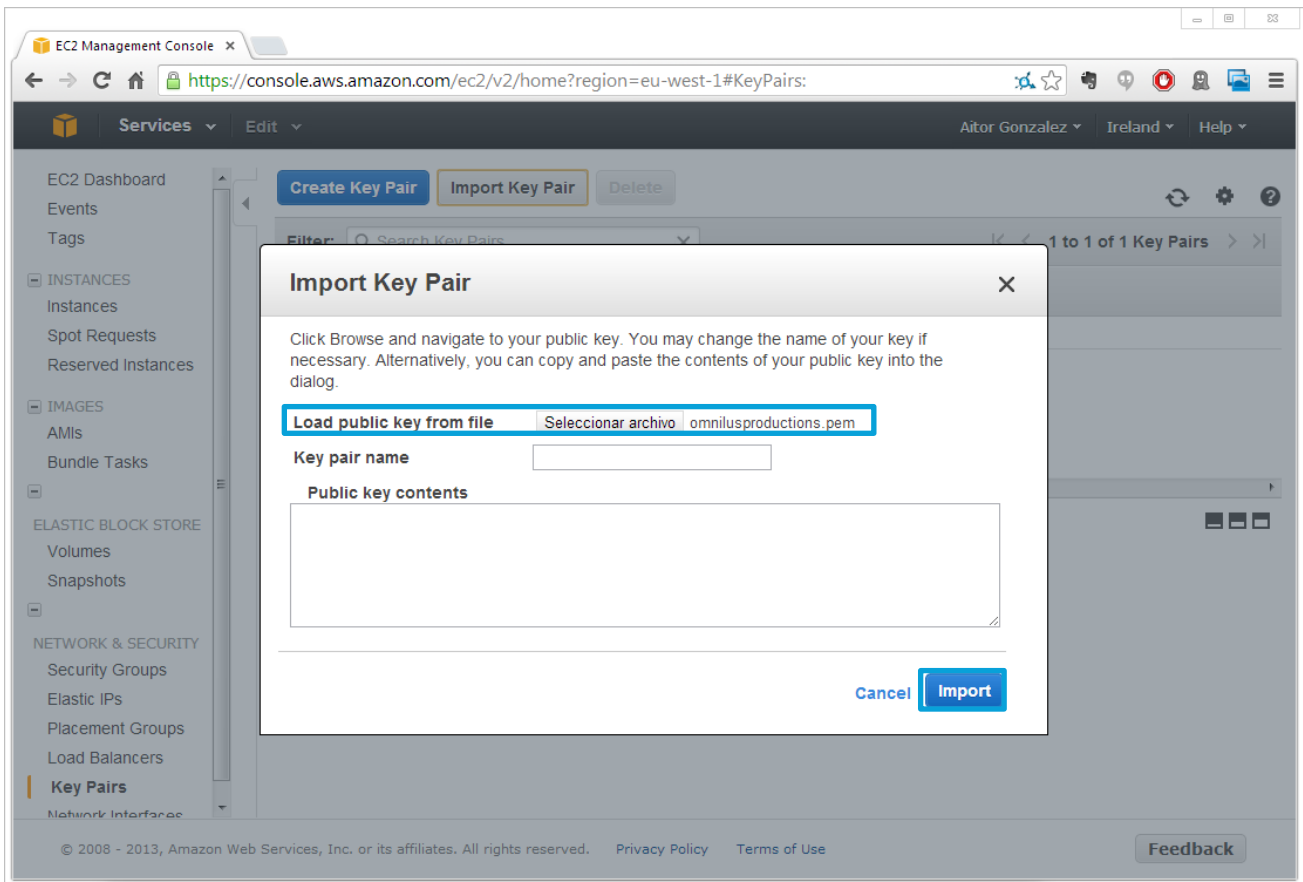
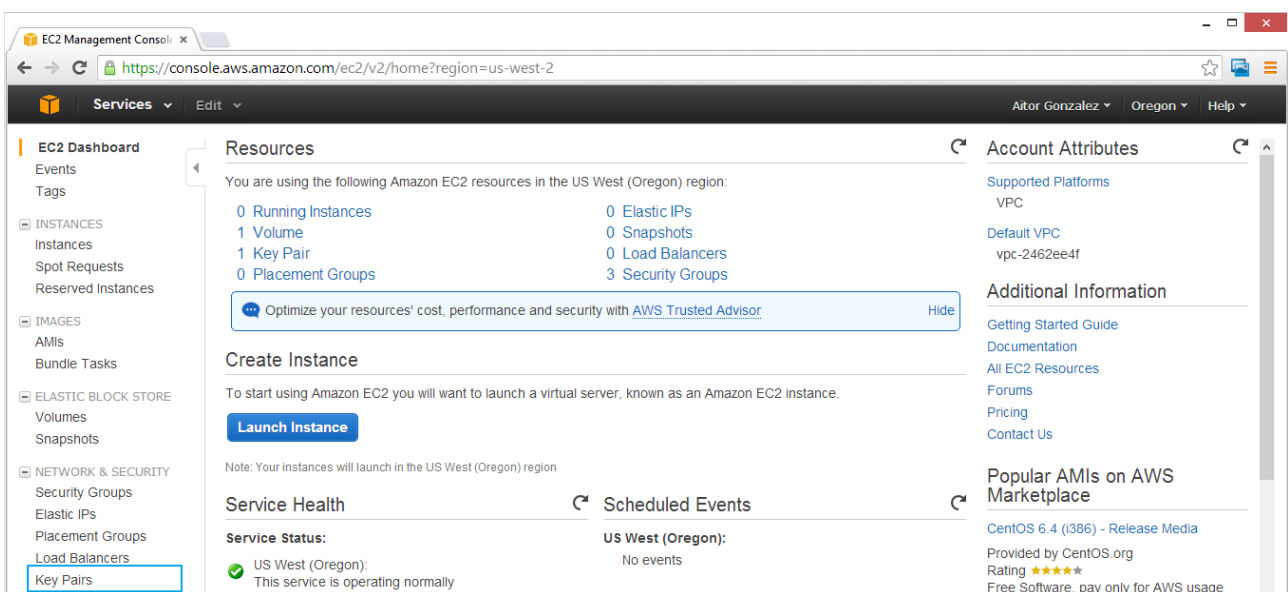


Ilustración 43 - EC2 Creación de claves: Formulario para la importación de Key Pairs

2.4 Eliminación de claves (Key Pairs)

Si se desea eliminar una *key pair* asociada a una instancia presionaremos en el menú izquierdo a la opción “Key Pairs” en la sección “Network and Security” (**Ilustración 44**). La eliminación de la “Key Pair” no tiene ninguna repercusión sobre las instancias creadas.



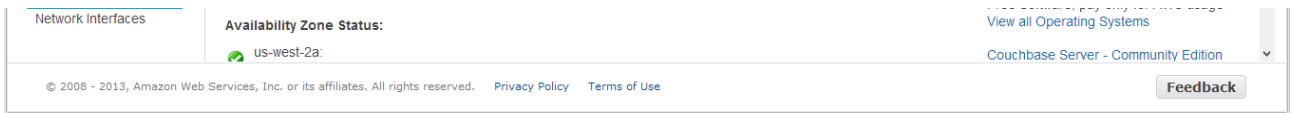


Ilustración 44 - EC2 Eliminación de claves: Inicio

Una vez dentro de dicha sección presionaremos con el botón derecho sobre la “Key Pair” que deseamos eliminar y clicaremos en la opción “Delete”. También la podemos seleccionar y presionar en el botón superior “Delete.” Se puede contemplar ambas alternativas en la **Ilustración 45**.

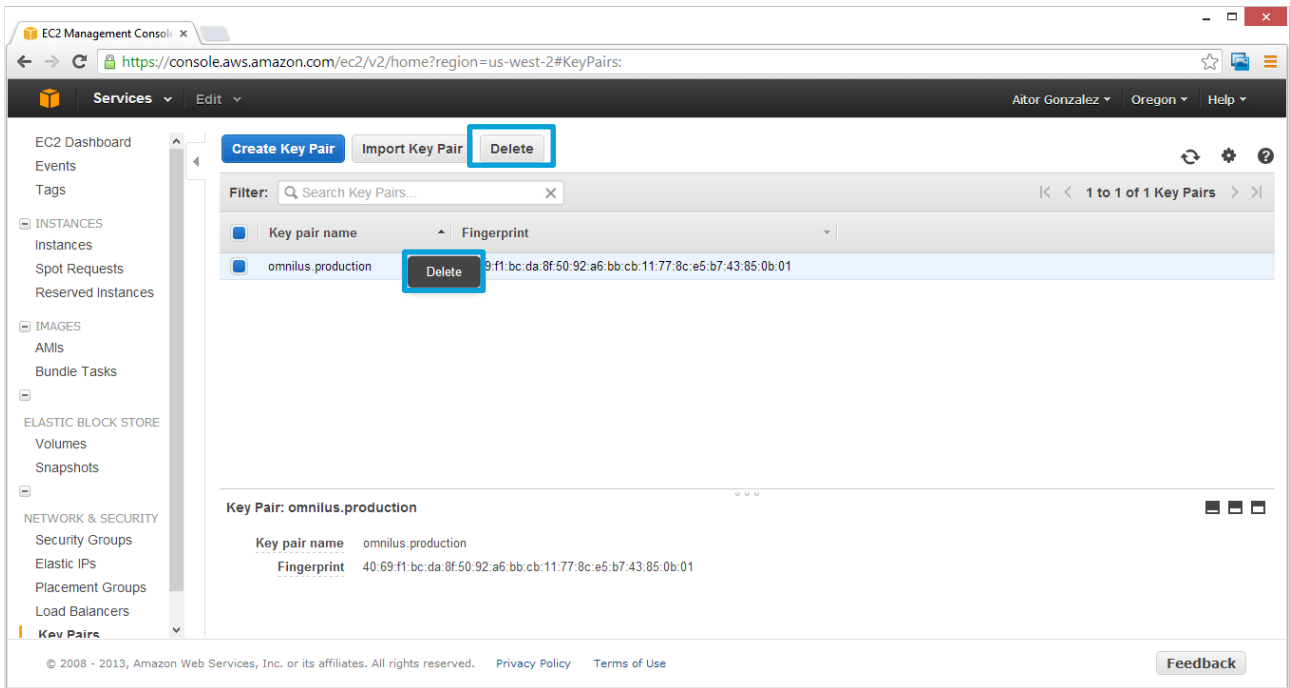


Ilustración 45 - EC2 Eliminación de claves: Selección y eliminación

Tras presionar la opción nos saldrá una ventana de confirmación, tal como se muestra en la **Ilustración 46**. Si estamos de acuerdo procedemos a clicar en “Yes” y esta se eliminará. En caso contrario, presionamos “Cancel” y todo quedará como estaba.

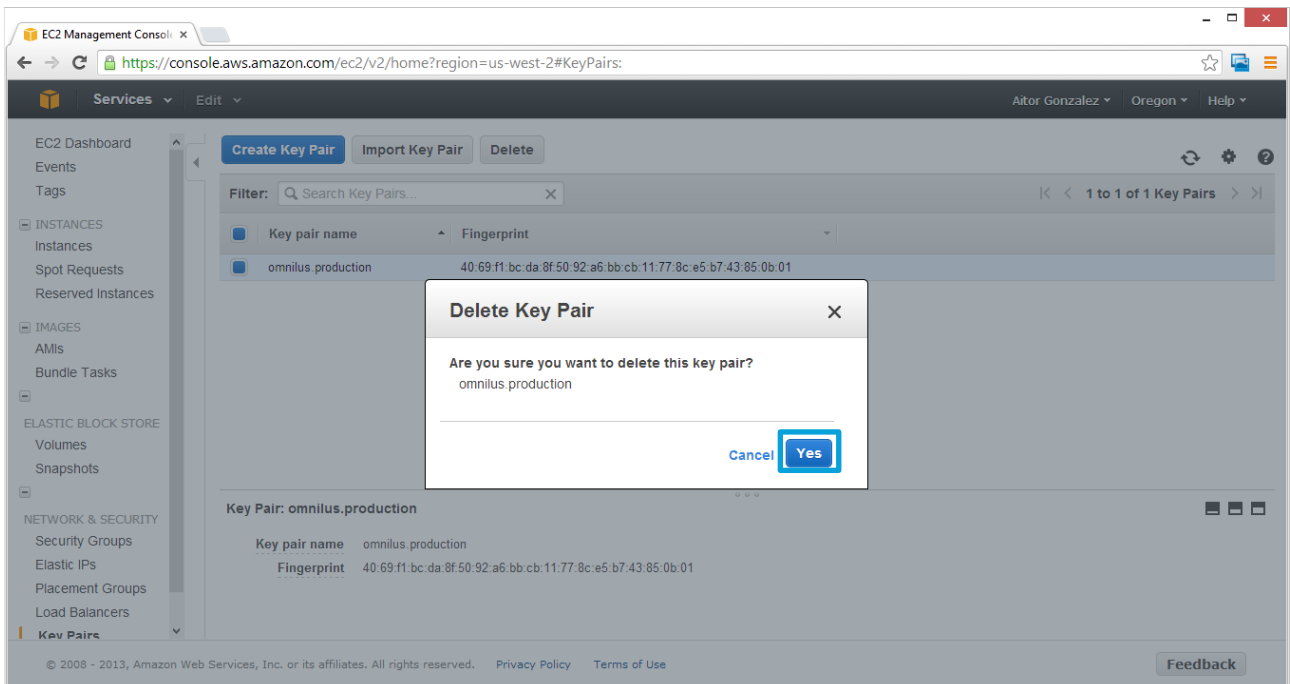


Ilustración 46 - EC2 Eliminación de claves: Confirmación

2.5 Creación de imágenes de máquina

Para crear una AMI (*Amazon Machine Image*) de una máquina nos hemos de dirigir al servicio de EC2, dirigiéndonos a la ventana de instancias, hacer clic derecho sobre una instancia y presionamos la opción “*create image*” (**Ilustración 47**).

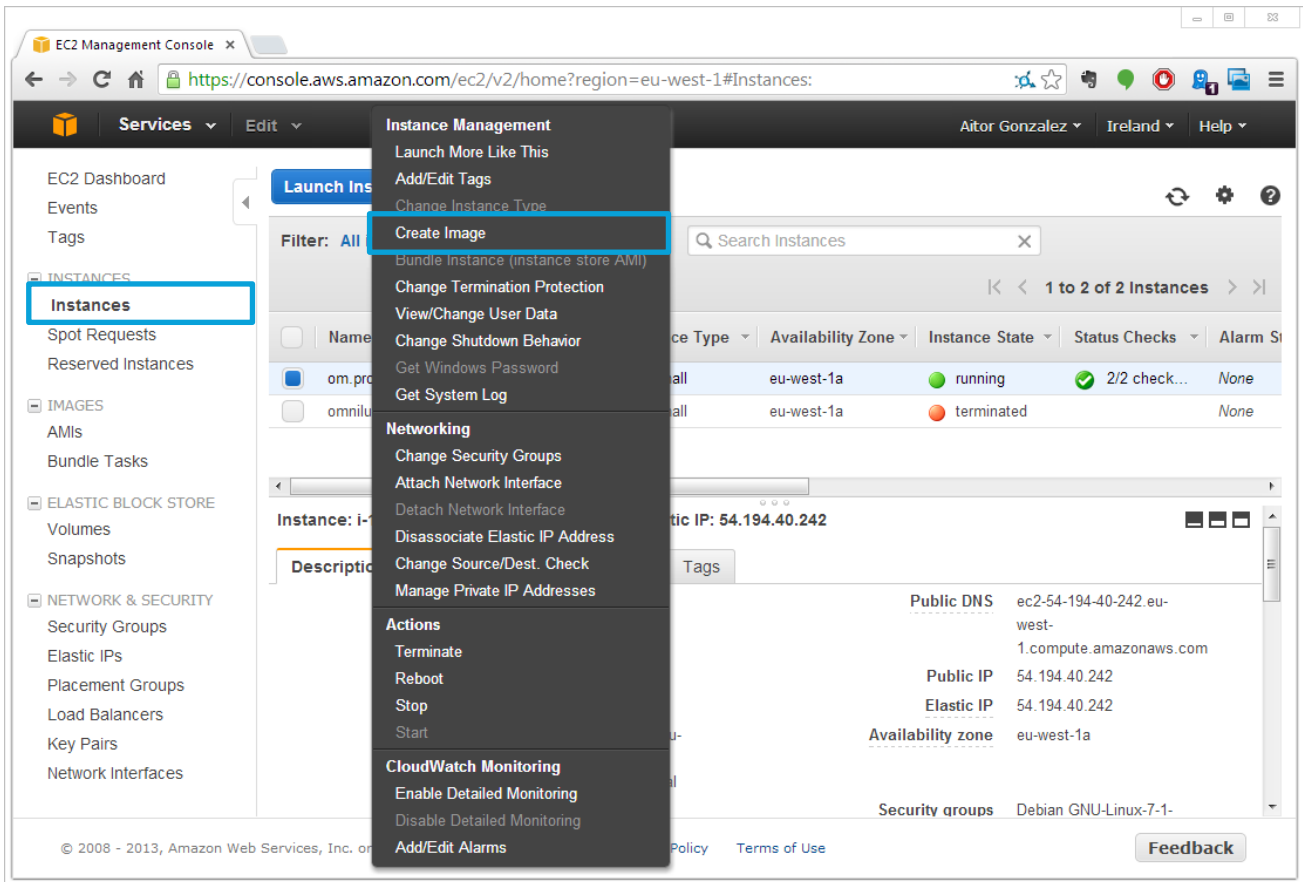
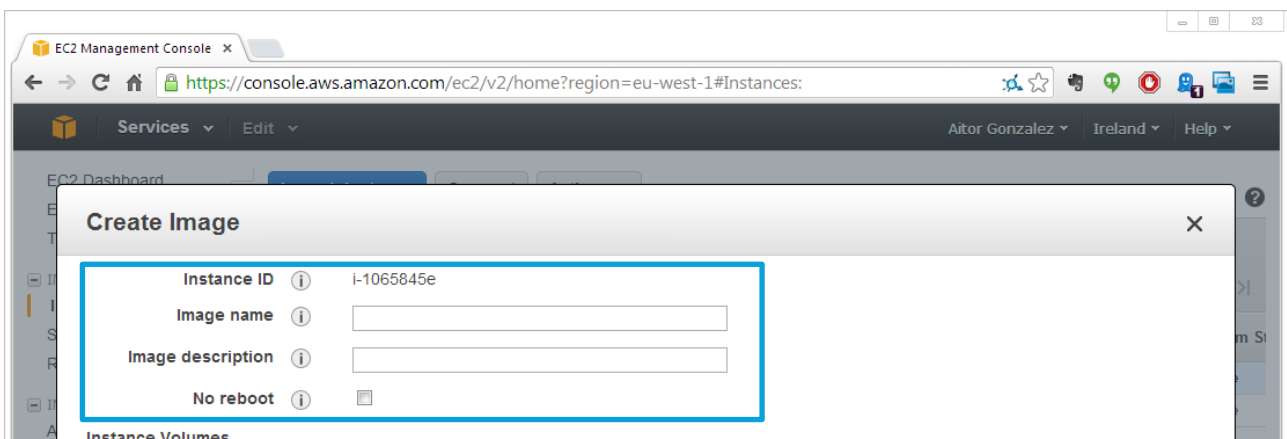


Ilustración 47 - Creación de AMI: Inicio

Una vez presionada en la opción mencionada procedemos a rellenar el formulario que se nos presenta (**Ilustración 48**). Dos puntos a destacar en el proceso son:

1. El check “*delete on termination*”: cuando se crea una AMI (*Amazon Machine Image*) de una máquina automáticamente se crea un *snapshot* de su EBS temporal, de esta manera, al iniciar una nueva instancia mediante la imagen creada automáticamente se le asociara una nueva unidad EBS creada a partir del *snapshot*. Si mantenemos presionado el *check*, esta nueva unidad será temporal y se eliminará tras eliminar la instancia al que está asociada, por lo contrario, si este no está marcado, este será permanente y continuará existiendo tras la eliminación de la instancia.
2. Opción “*Add new volumen*”: se pueden definir nuevos volúmenes de almacenaje (EBS) asociadas a la AMI (*Amazon Machine Image*) para que se monten tras crear una nueva instancia desde esta.



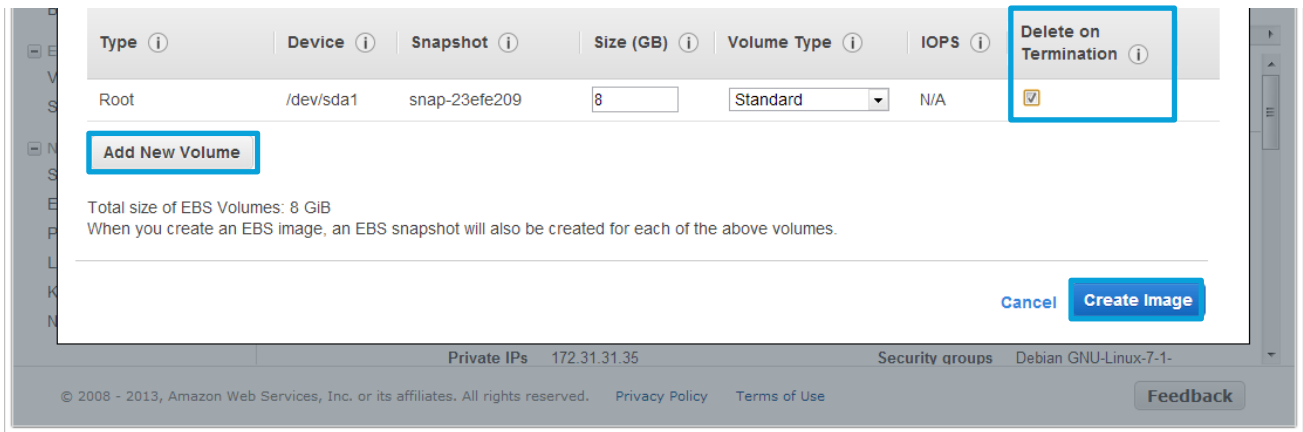


Ilustración 48 - Creación de AMI: Formulario de configuración

Una vez rellenados los campos y presionado en “Create Image” se creará la AMI (Amazon Machine Image) en la sección “AMIs” y un *snapshot* del volumen de la instancia en “Snapshots”.

2.6 Eliminación de imágenes de máquina

El proceso de eliminación es sencillo, basta con ir a la sección de “AMIs”, clicar botón derecho sobre la que se desea eliminar y presionar en “Deregister” (**Ilustración 49**).

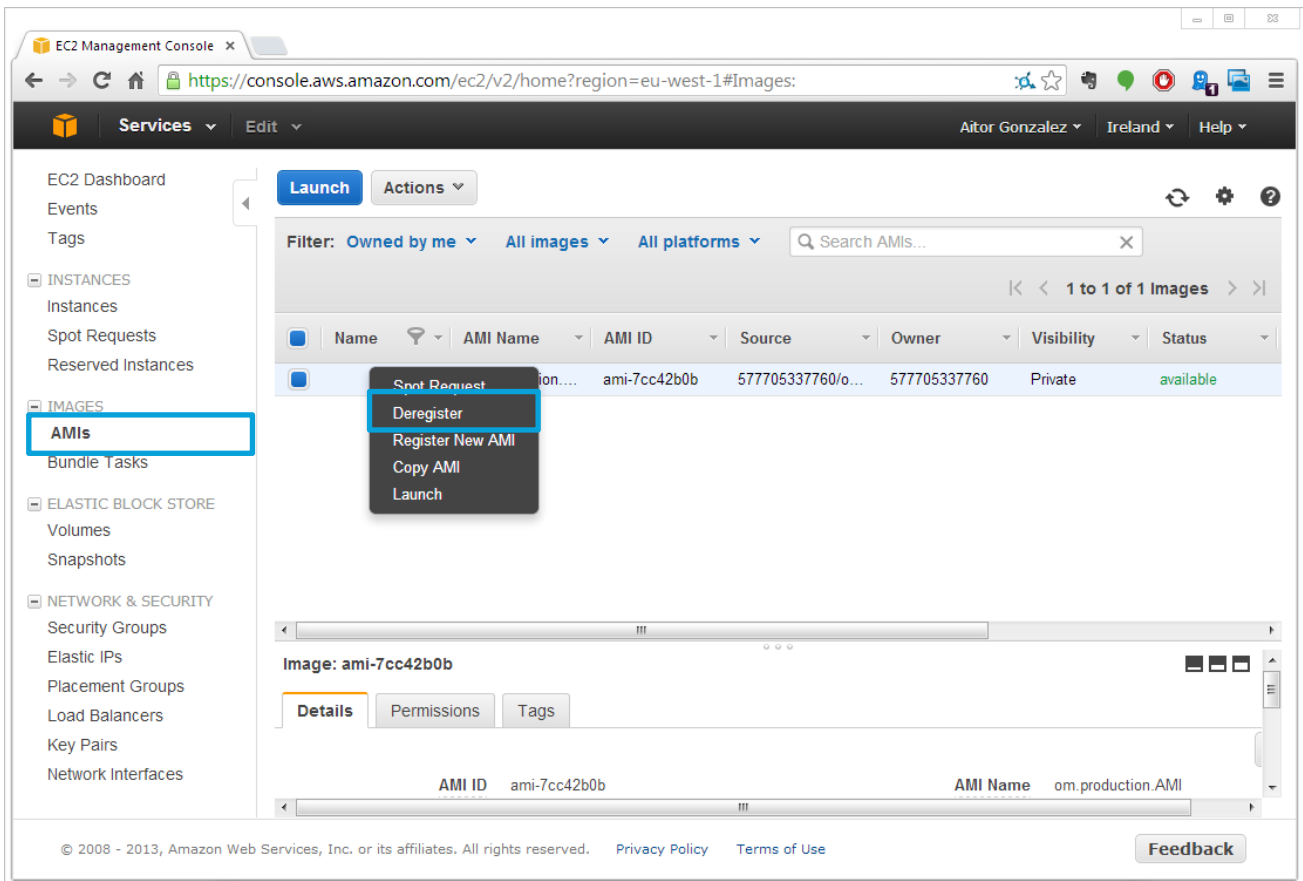


Ilustración 49 - Eliminación de AMI: Inicio

Automáticamente no saldrá una ventana para la confirmación tal como se muestra en la **Ilustración 50**. Presionamos en “Continue” y el proceso habrá terminado.

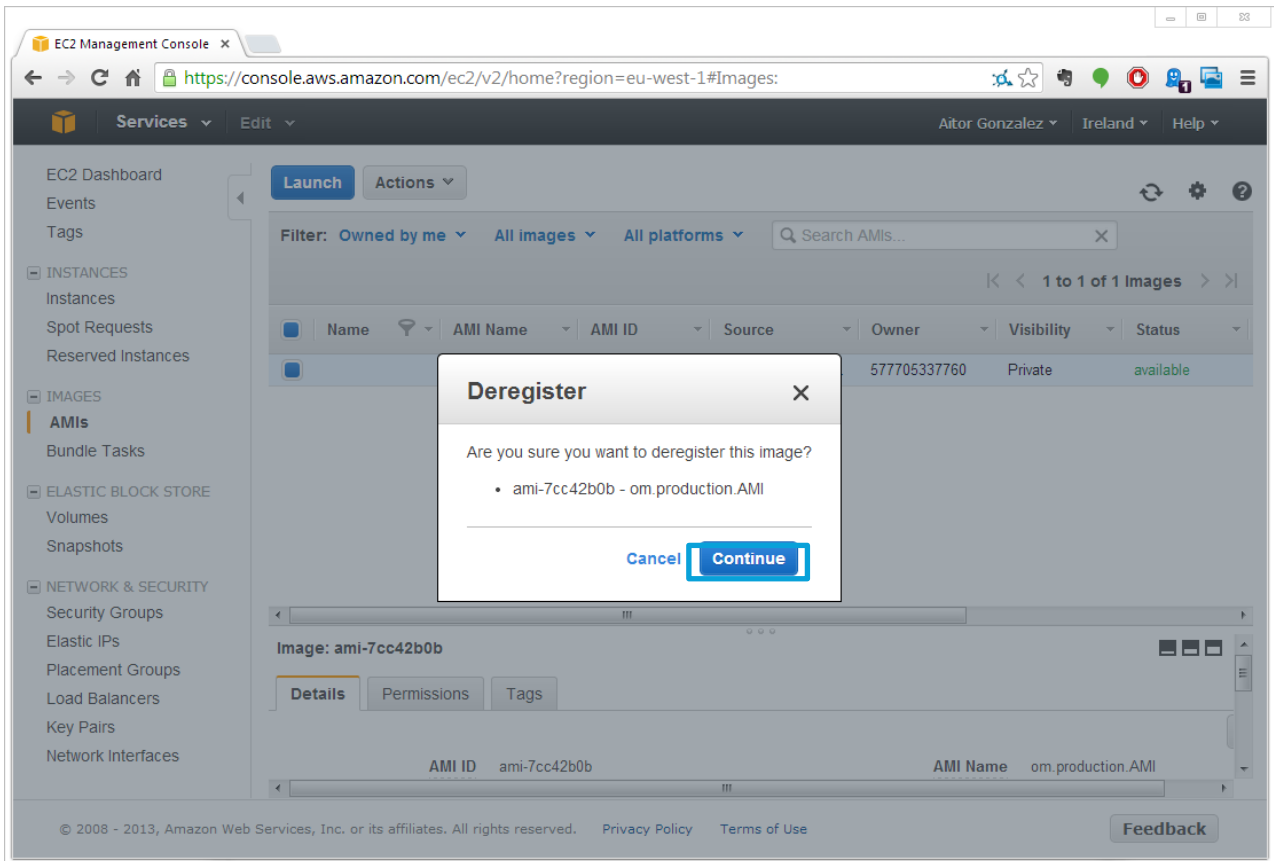


Ilustración 50 - Eliminación de AMI: Aviso y confirmación

No obstante es recomendable eliminar el “*snapshot*” asociado a la AMI (*Amazon Machine Image*) para abaratar costes. Para ello nos dirigimos a la sección de “*snapshots*”, presionamos botón derecho sobre el que deseamos eliminar y clicamos en “*Delete Snapshot*” (**Ilustración 51**).

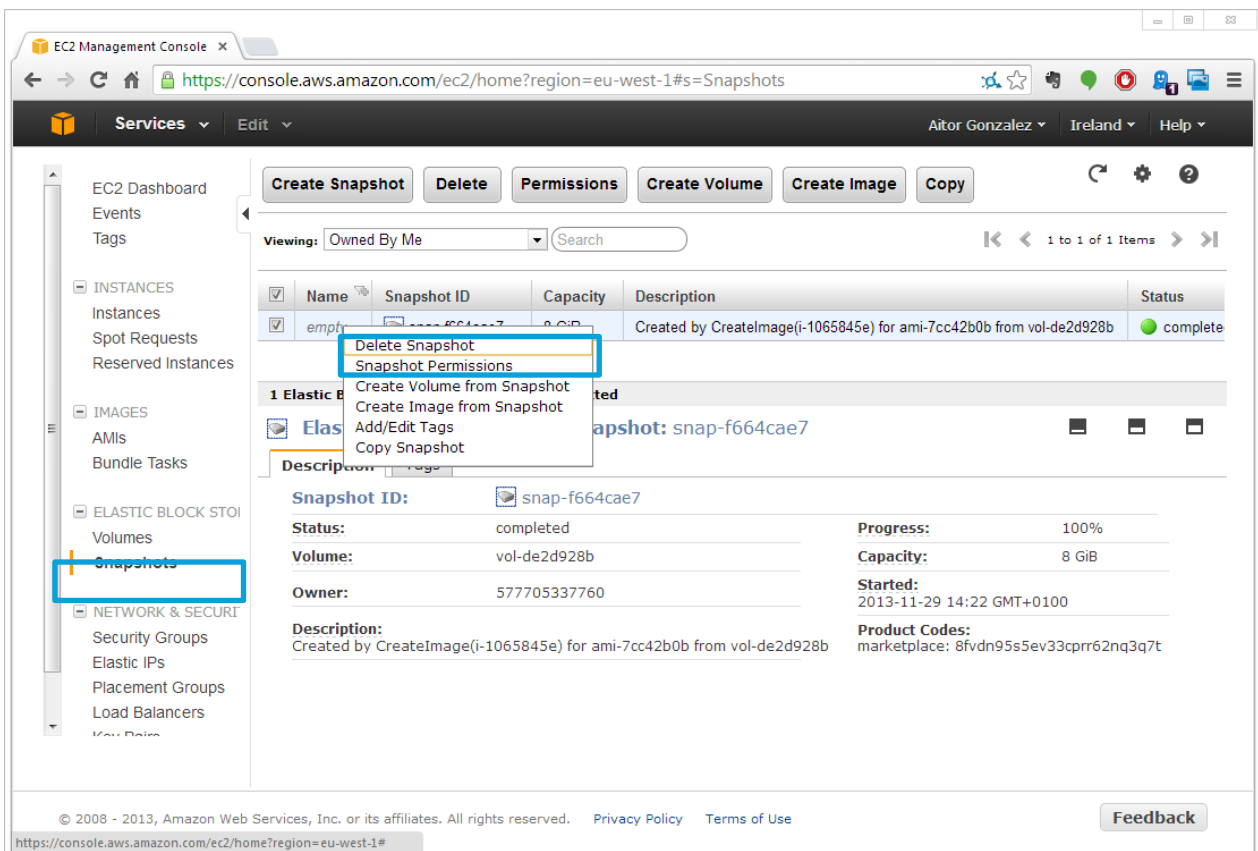


Ilustración 51 - Eliminación de AMI: Borrado de snapshot asociado

De la misma forma que en el caso anterior nos saldrá una ventana para la confirmación (**Ilustración 52**), clicamos en “Yes, Delete” y este será eliminado.

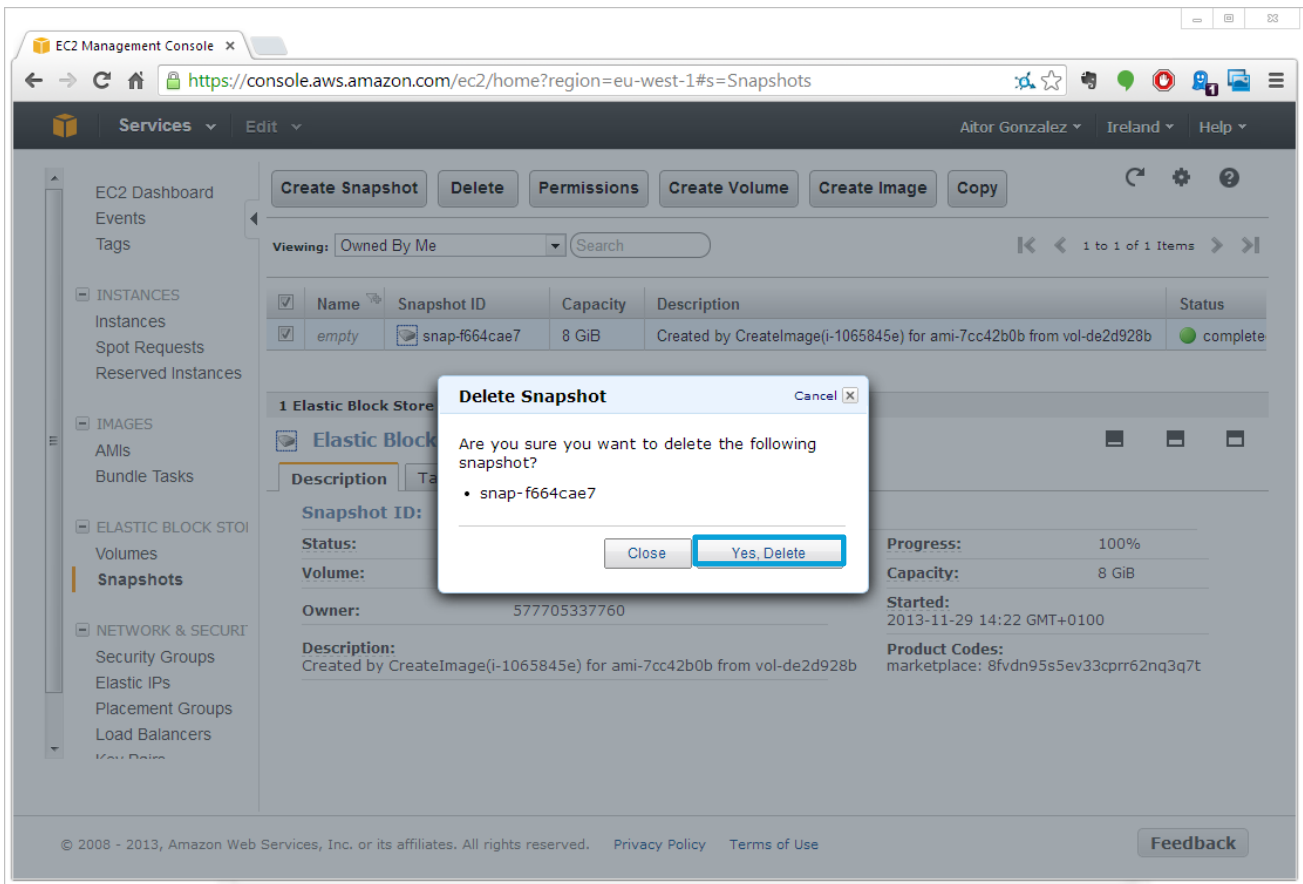


Ilustración 52 - Eliminación de AMI: Aviso y confirmación del borrado del snapshot asociado

2.7 Gestión del acceso (Firewall)

EC2 lleva incorporado un firewall gestionable desde la web. Para gestionarlo basta con presionar en “Security Groups” en la sección “Network & Security”. Al presionar se mostrará la interfaz de gestión (**Ilustración 53**).

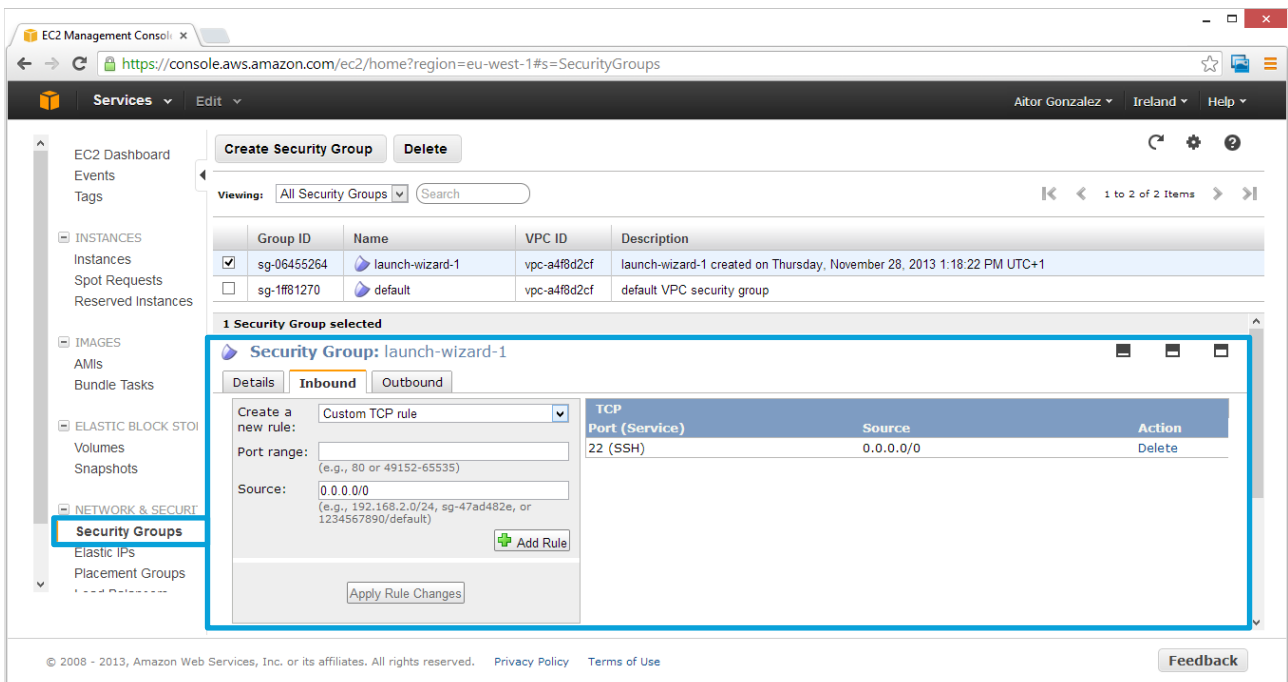


Ilustración 53 - Gestión del firewall: Interfaz

Cada máquina puede tener asociado un grupo de seguridad distinto, por lo tanto, hemos de seleccionar aquel grupo de seguridad que está asociado a nuestra instancia. Tras seleccionarlo se nos mostrarán las opciones de configuración.

La interfaz es totalmente intuitiva, basta con navegar por las pestañas o eliminando los puertos que deseamos.

Si lo deseamos podemos crear un nuevo grupo desde el botón “*Create Security Group*” y asociarlo a una interfaz de red. Nos hemos de asegurar que esta es la de la instancia que deseamos controlar. Una vez creado procederíamos tal como hemos explicado con anterioridad.

Desde el mismo entorno se pueden eliminar grupos ya creados.

2.8 Gestión de IPs estáticas

Para asociar una IP estática a una instancia creada nos seleccionaremos la opción “*Elastic IPs*” en la sección “*Network & Security*” en el menú izquierdo de la interfaz de gestión de EC2. Posteriormente presionaremos en “*Allocate new Address*” (**Ilustración 54**).

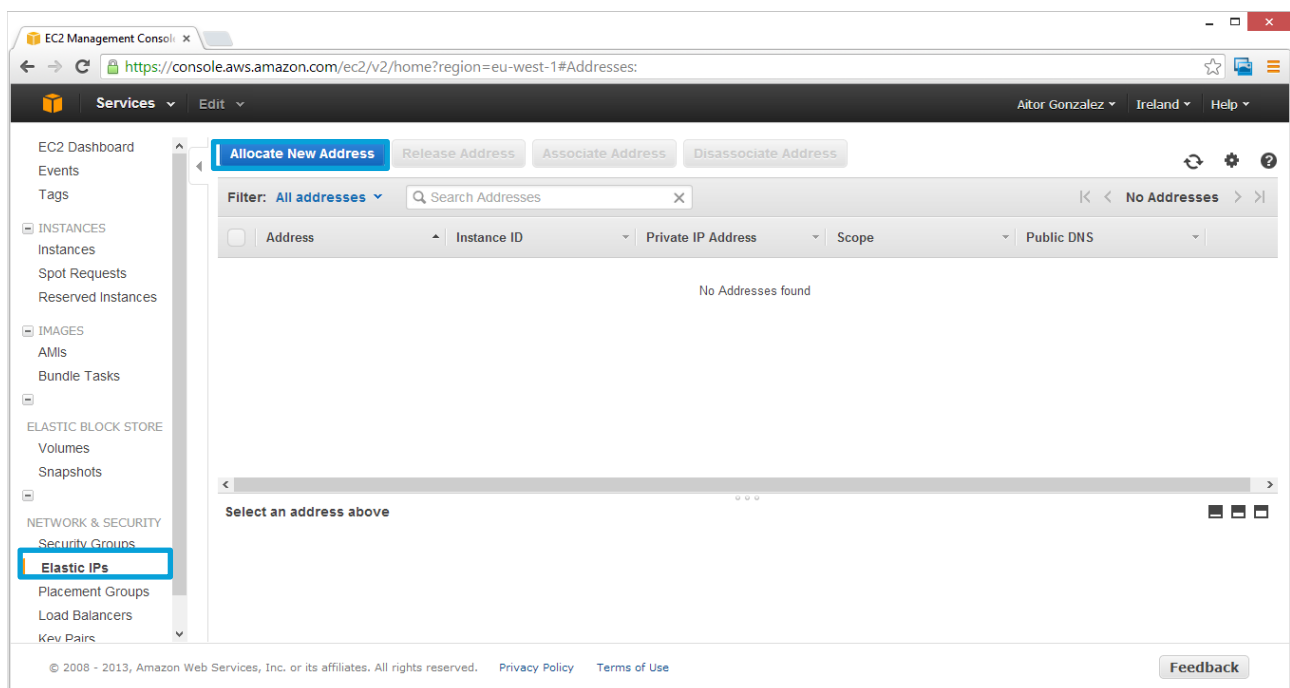
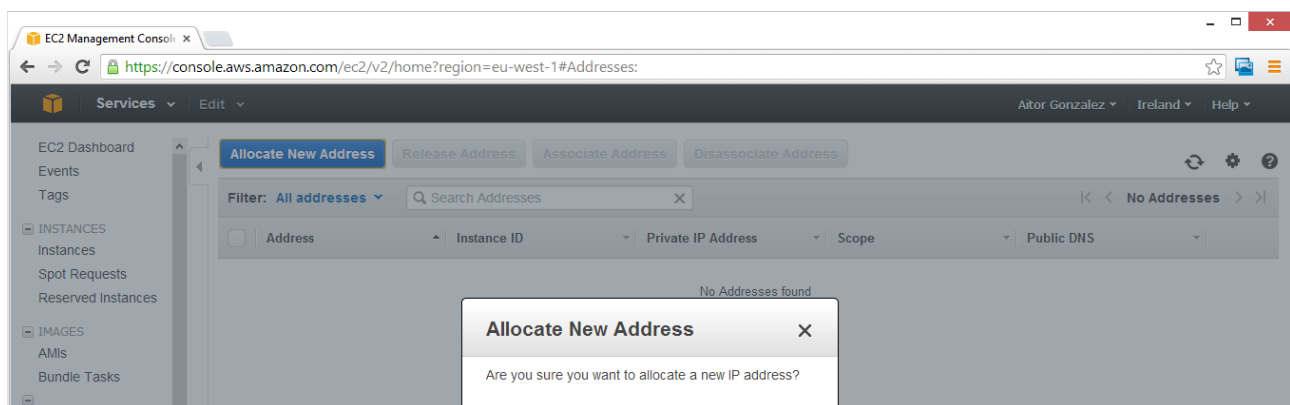


Ilustración 54 - Gestión de IPs estáticas: Inicio de creación

Tras presionar en “*Allocation New Adress*” no saldrá una ventana emergente (**Ilustración 55**). Basta con que presionemos en “*Yes, Allocate*” para que el sistema nos dé una IP estática.



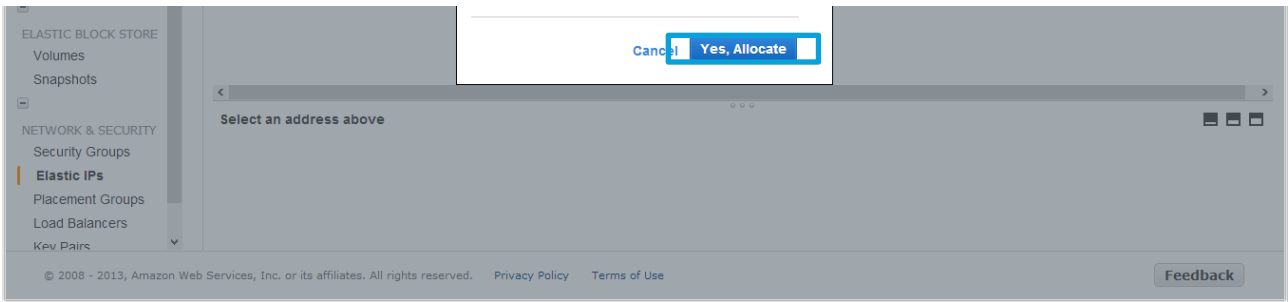


Ilustración 55 - Gestión de IPs estáticas: Aviso de creación y confirmación

De esta manera en la tabla inicial veremos la IP que se no ha concedido. Sobre esta IP hacemos *click* derecho y presionamos en “Associate Address” (**Ilustración 56**).

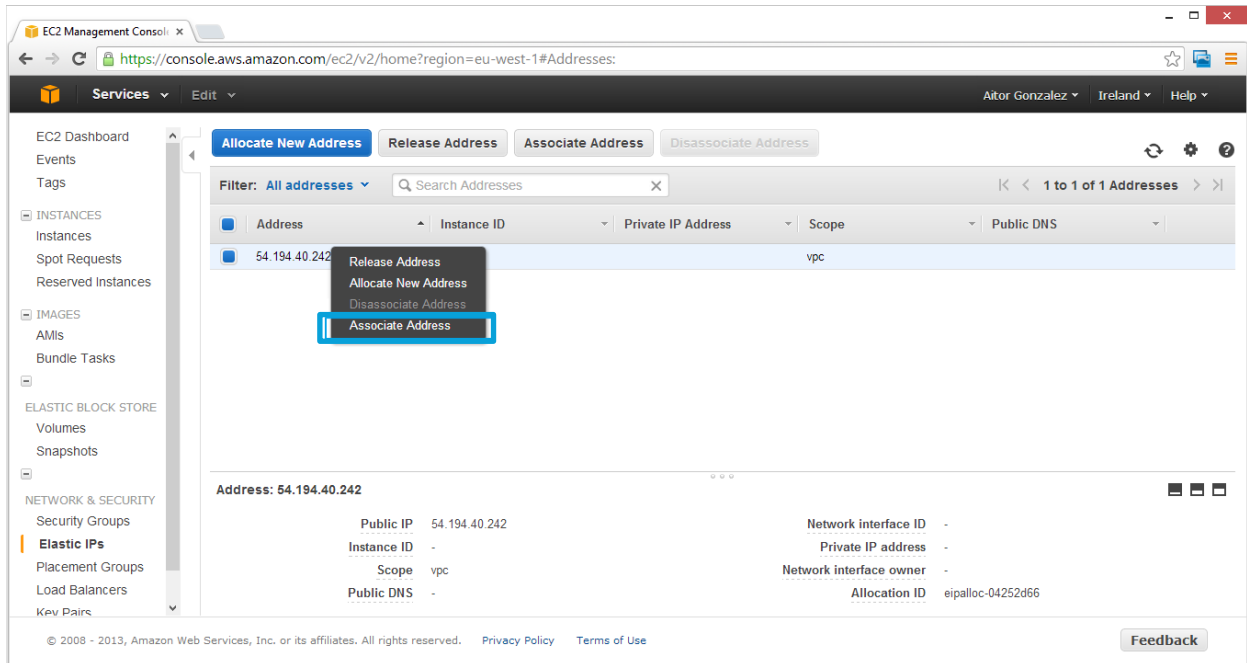


Ilustración 56 - Gestión de IPs estáticas: inicio de asociación

Nos saldrá una nueva ventana emergente (**Ilustración 57**) donde tendremos que rellenar un formulario. En él es importante seleccionar a la instancia que queremos asociar la IP. Posteriormente clicamos en “Associate”.

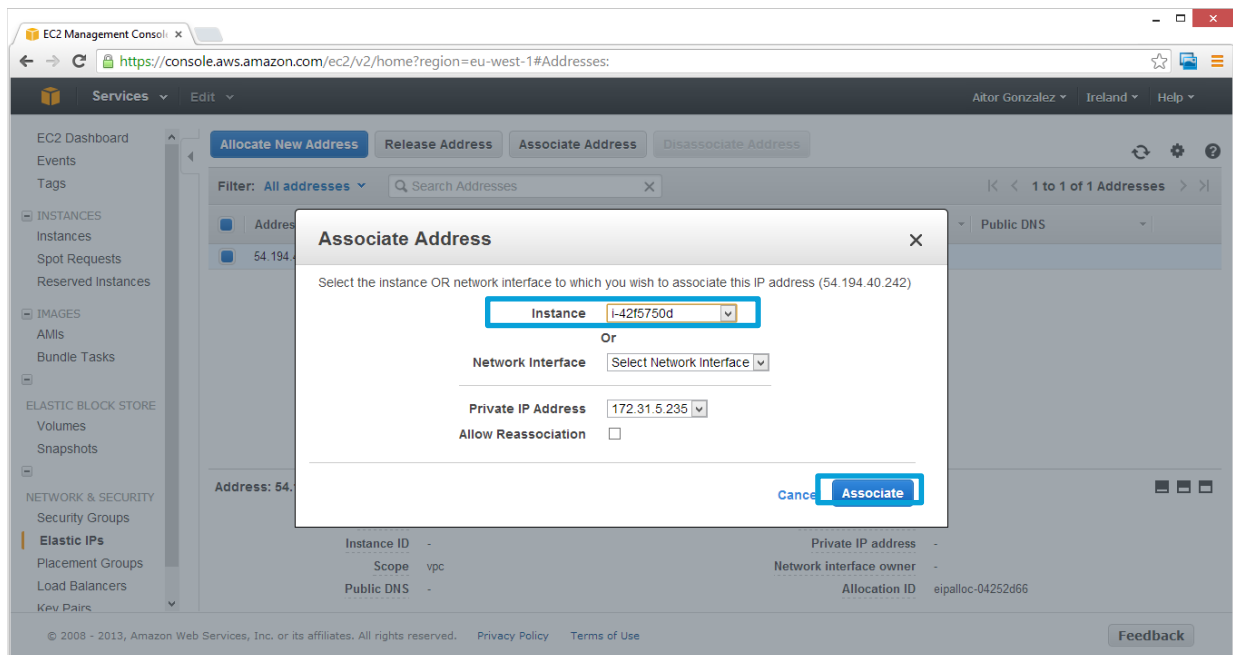


Ilustración 57 - Gestión de IPs estáticas: Formulario de asociación

Ahora ya tenemos la IP estática asociada a la instancia tal como podemos observar en la **Ilustración 58**.

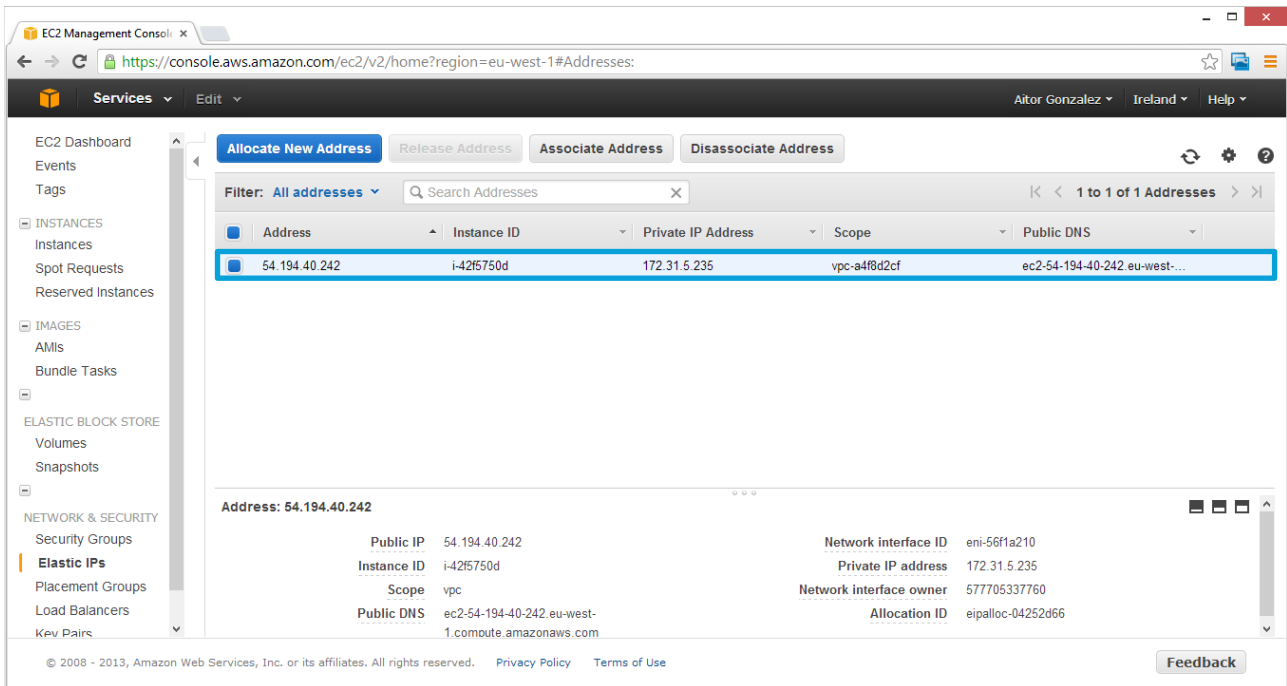


Ilustración 58 - Gestión de IPs estáticas: Confirmación de la asociación

En caso de que queramos desasociar una IP de una instancia será suficiente con seleccionar en la tabla la IP a desasociar, presionar el botón derecho del ratón y seleccionar “Disassociate Address” (**Ilustración 59**). También podemos realizar la acción mediante los botones superiores a la tabla.

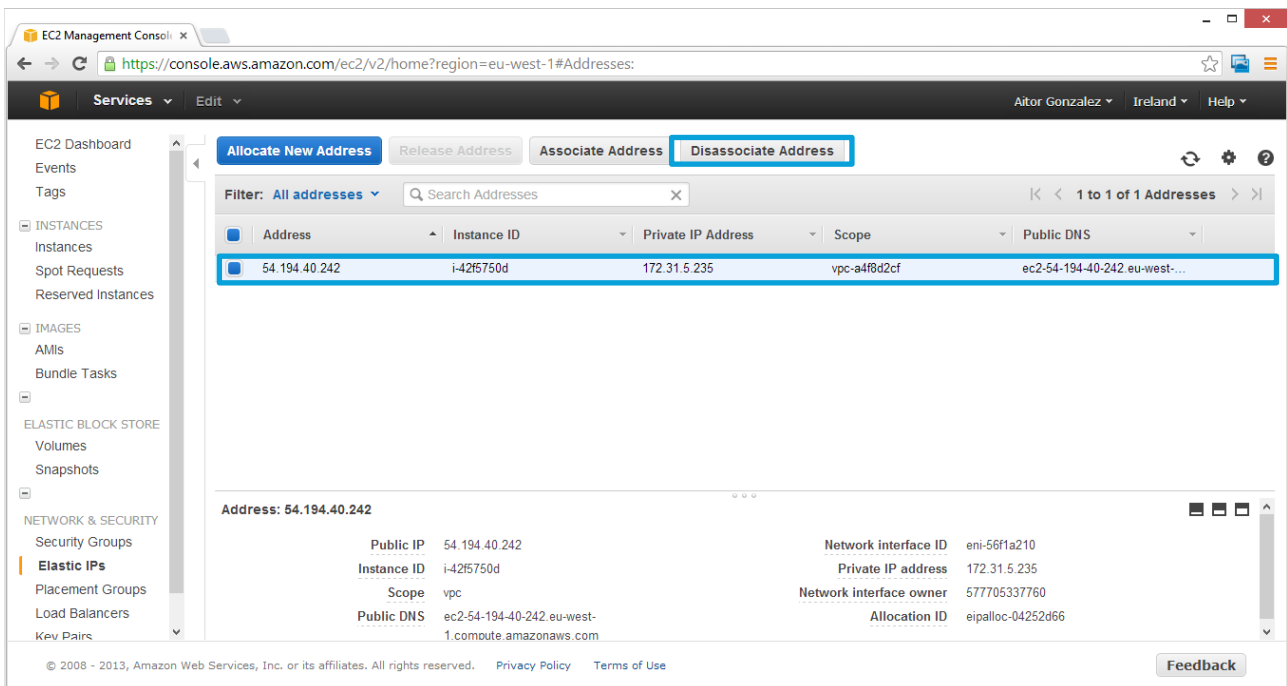


Ilustración 59 - Gestión de IPs estáticas: Disociación

Una vez seleccionada la opción o presionado el botón nos saldrá una ventana emergente (**Ilustración 60**) pidiendo la confirmación. Clicamos en “Yes, Disassociate” y la IP quedará desasociada de la instancia.

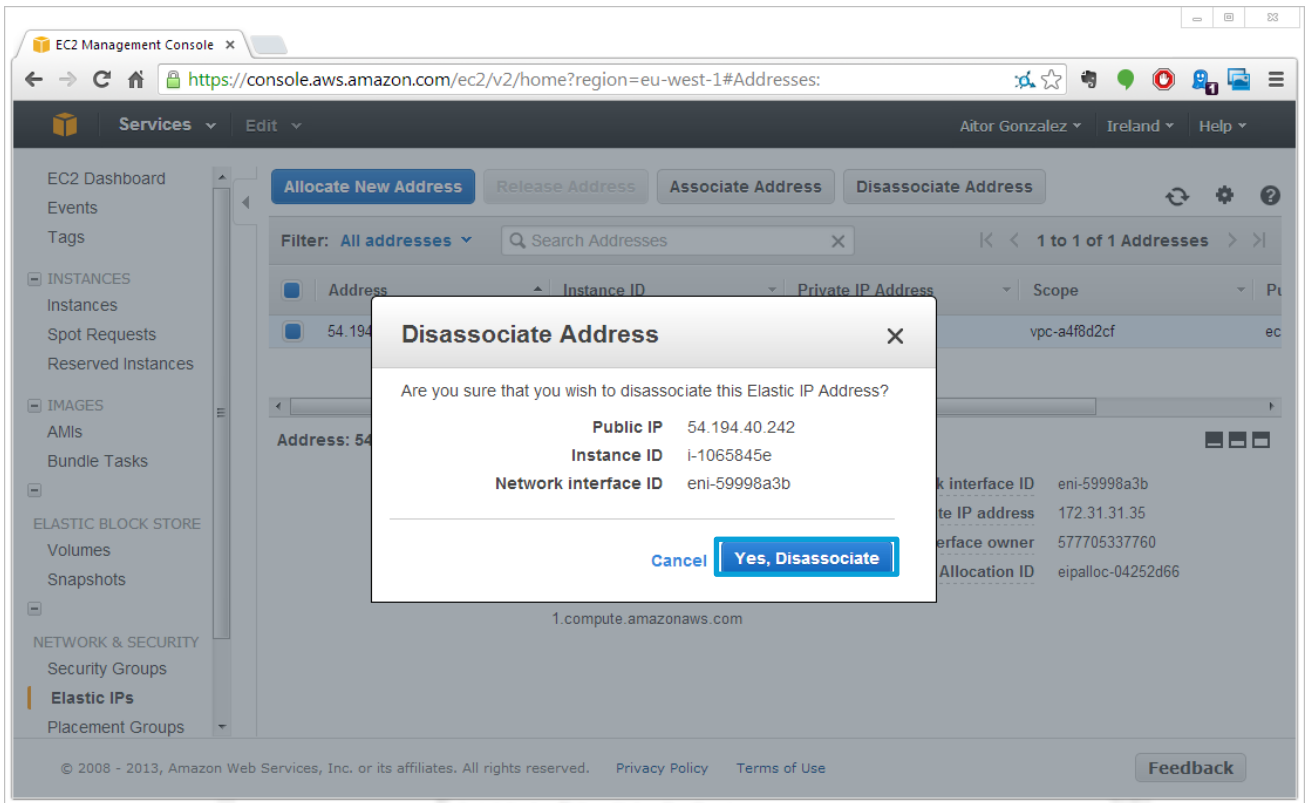


Ilustración 60 - Gestión de IPs estáticas: Confirmación para la disociación

Por finalizar, si deseamos revocar la adquisición de una IP estática bastará con desasociarla primero y a posterior clicar en "Release Address" (Ilustración 61).

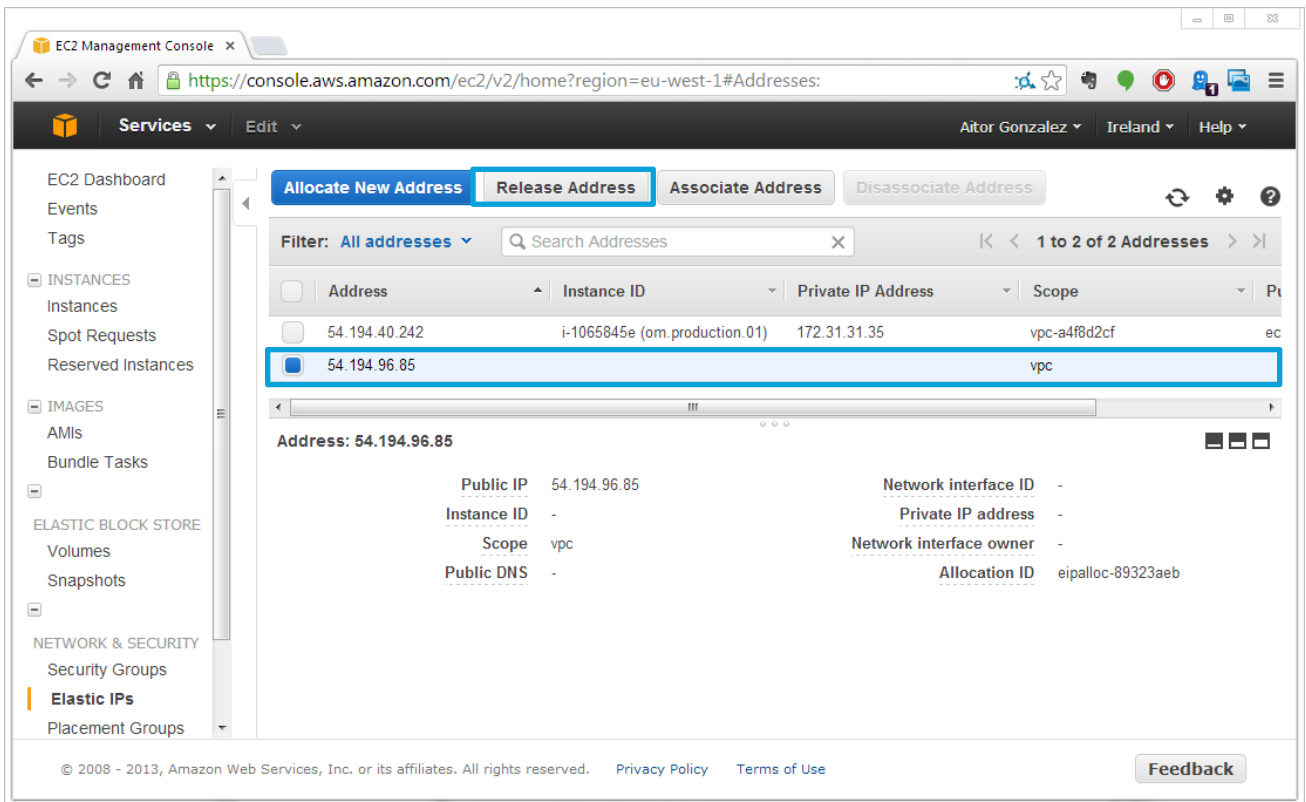


Ilustración 61 - Gestión de IPs estáticas: Revocación

Nos saldrá una ventana emergente (Ilustración 62) donde lo único que tendremos que hacer es clicar en "Yes, Release".

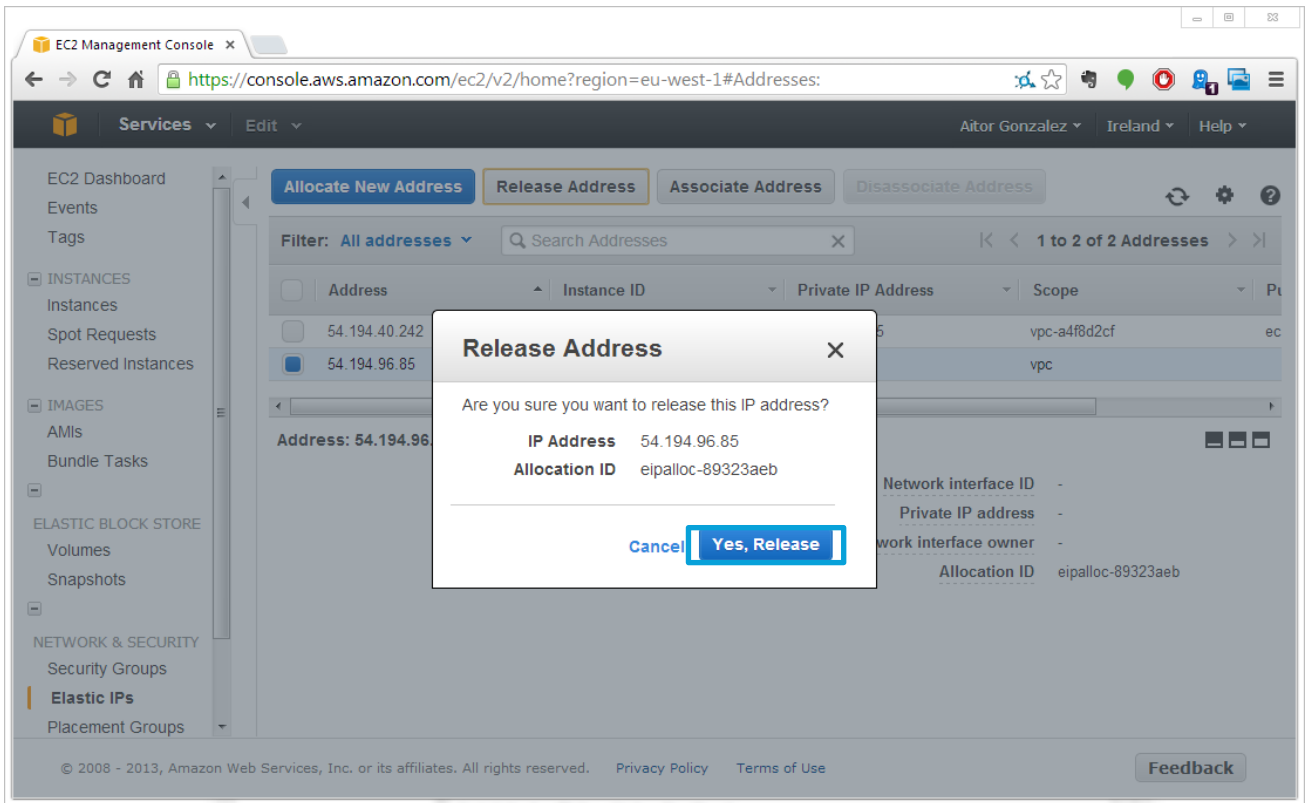
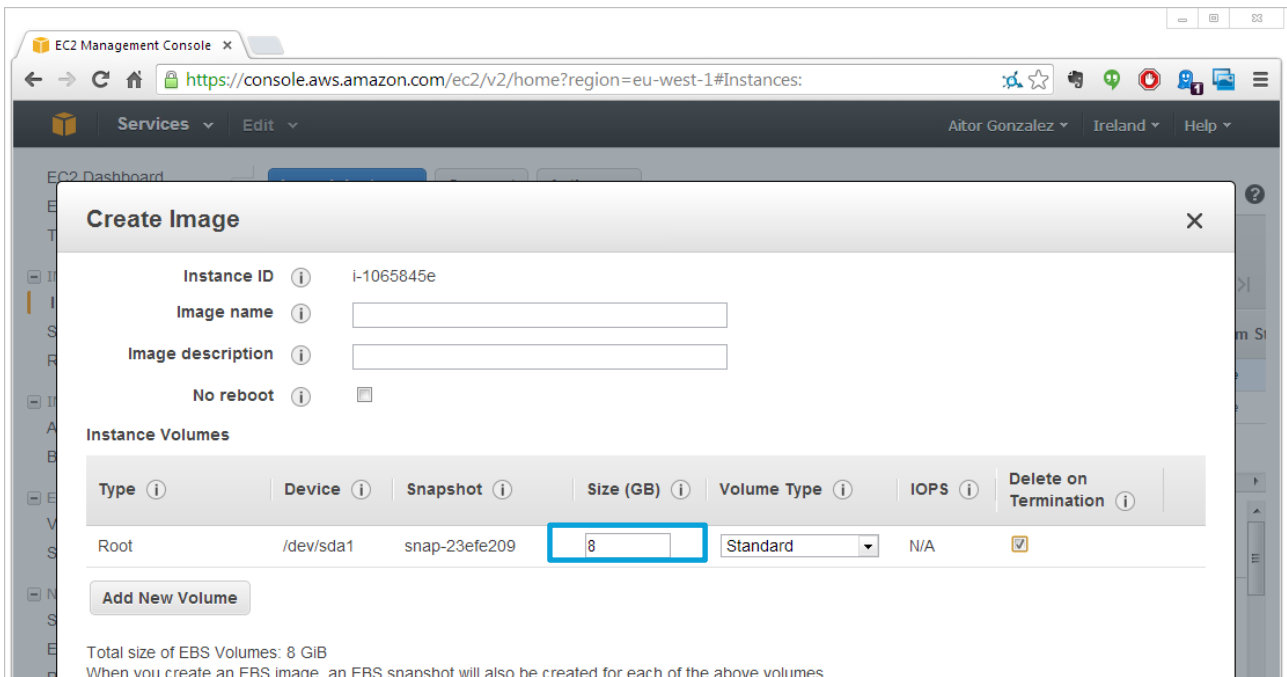


Ilustración 62 - Gestión de IPs estáticas: Confirmación de revocación

De esta manera la IP estática quedará devuelta al sistema.

2.9 Ampliación del EBS Ephimeral de instancia por defecto

Mientras trabajamos nos podemos dar cuenta de que el tamaño de disco por defecto (EBS) de la instancia no dispone de la capacidad deseada para nuestro propósito. En este caso, lo primero que hemos de hacer, es crear una AMI (*Amazon Machine Image*) de la máquina y especificar un tamaño de *snapshot* adecuado, tal como se muestra en la **Ilustración 63**.



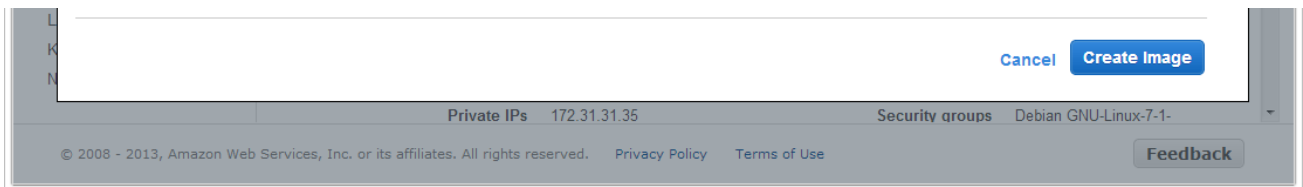


Ilustración 63 - Ampliación del tamaño del volumen por defecto (EBS): Formulario

Una vez creada la AMI (*Amazon Machine Image*) hemos de levantar otra instancia mediante esta, y automáticamente se le asociará un volumen del tamaño especificado con todos los datos y configuraciones de la instancia usada para el proceso.

2.10 Uso de LoadBalancer

LoadBalancer es un servicio dentro de EC2 que nos permite distribuir las cargas de las instancias en función del tráfico de red.

Para activar el servicio realizaremos los siguientes pasos:

1. **(Ilustración 64)** En la interfaz de gestión de EC2 clicamos sobre "LoadBalancers", situado en el grupo "Network & Security" en el menú izquierdo. Después presionamos en "Create Load Balancer".

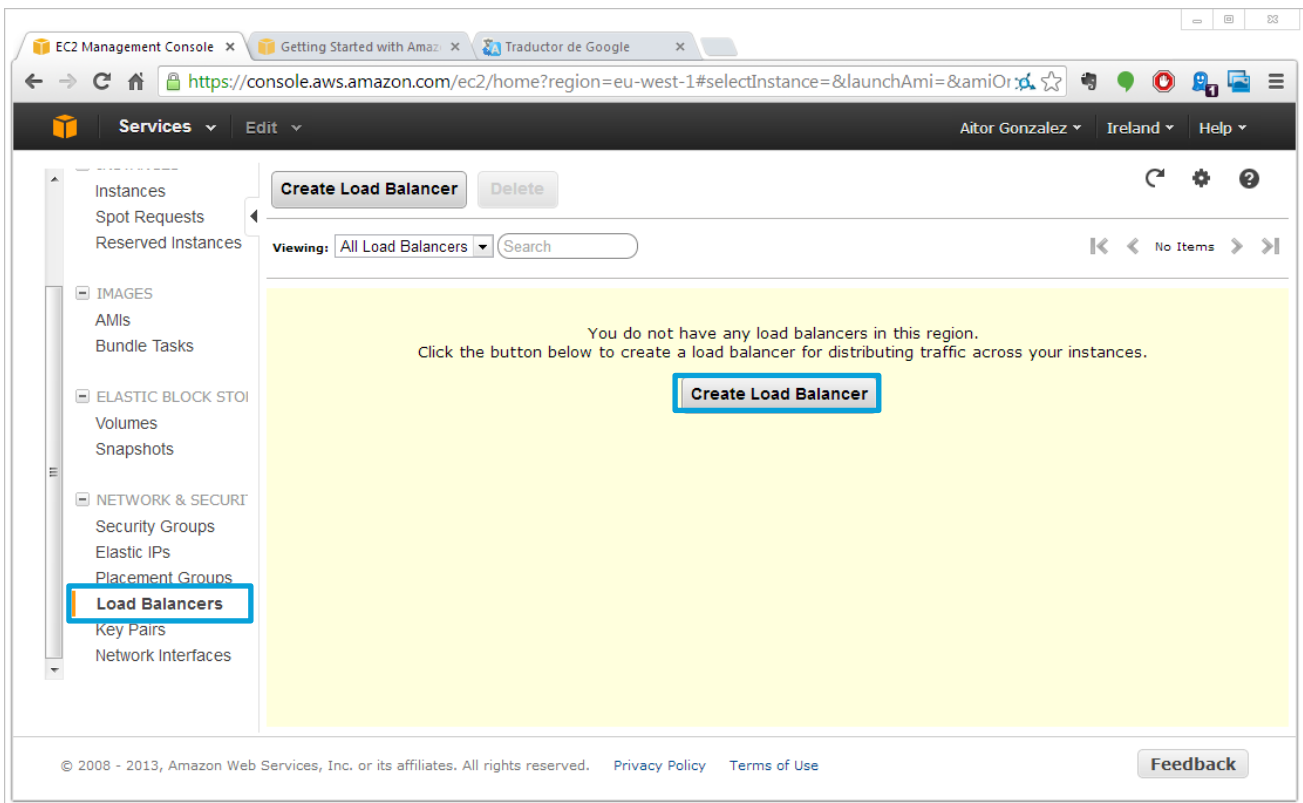
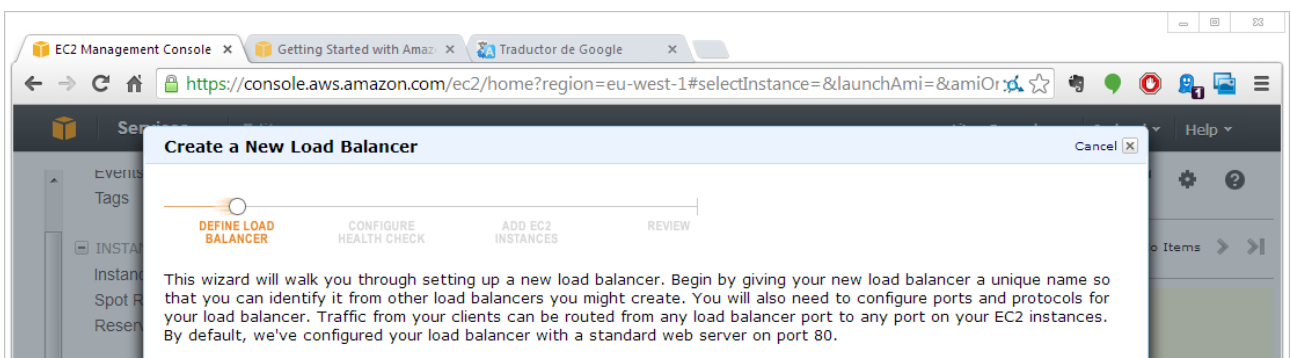


Ilustración 64 - LoadBalancer: Inicio

2. **(Ilustración 65)** Rellenamos el formulario según nuestras preferencias y según lo que nos indique. Tras ello presionamos en "Continue".



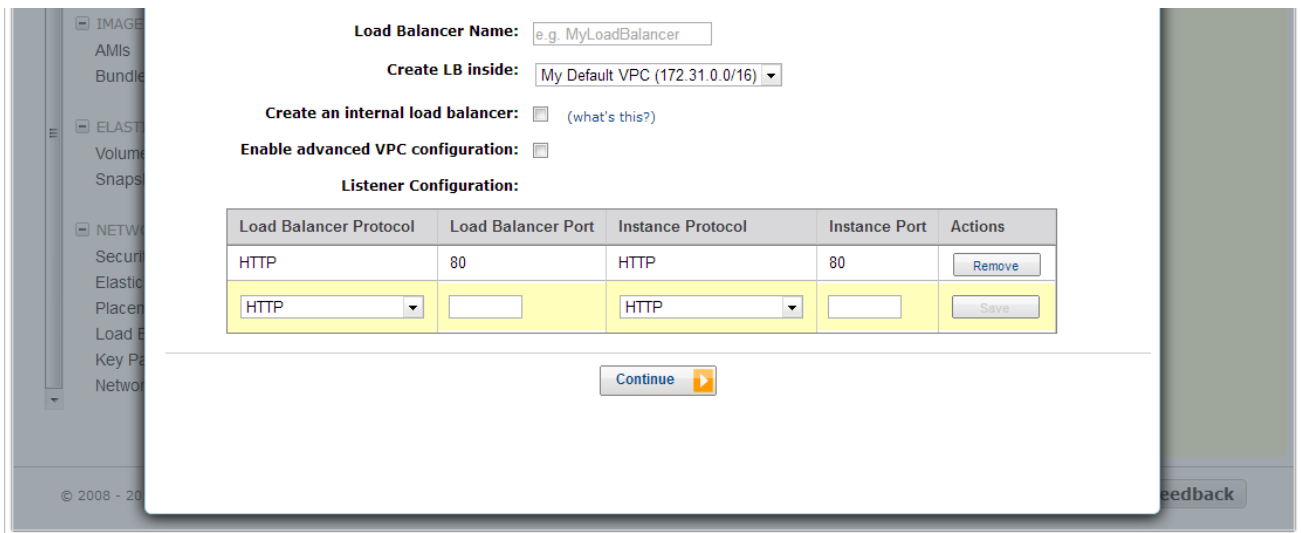


Ilustración 65 - Creación de LoadBalancer paso 01 de 05: Definición

El punto más destacable de este es la configuración de los puertos y la posibilidad de crear un balanceador interno para la comunicación entre instancias.

3. **(Ilustración 66)** Actuamos de la misma forma que en el formulario anterior, prestando especial atención a la sección "Advanced Options".

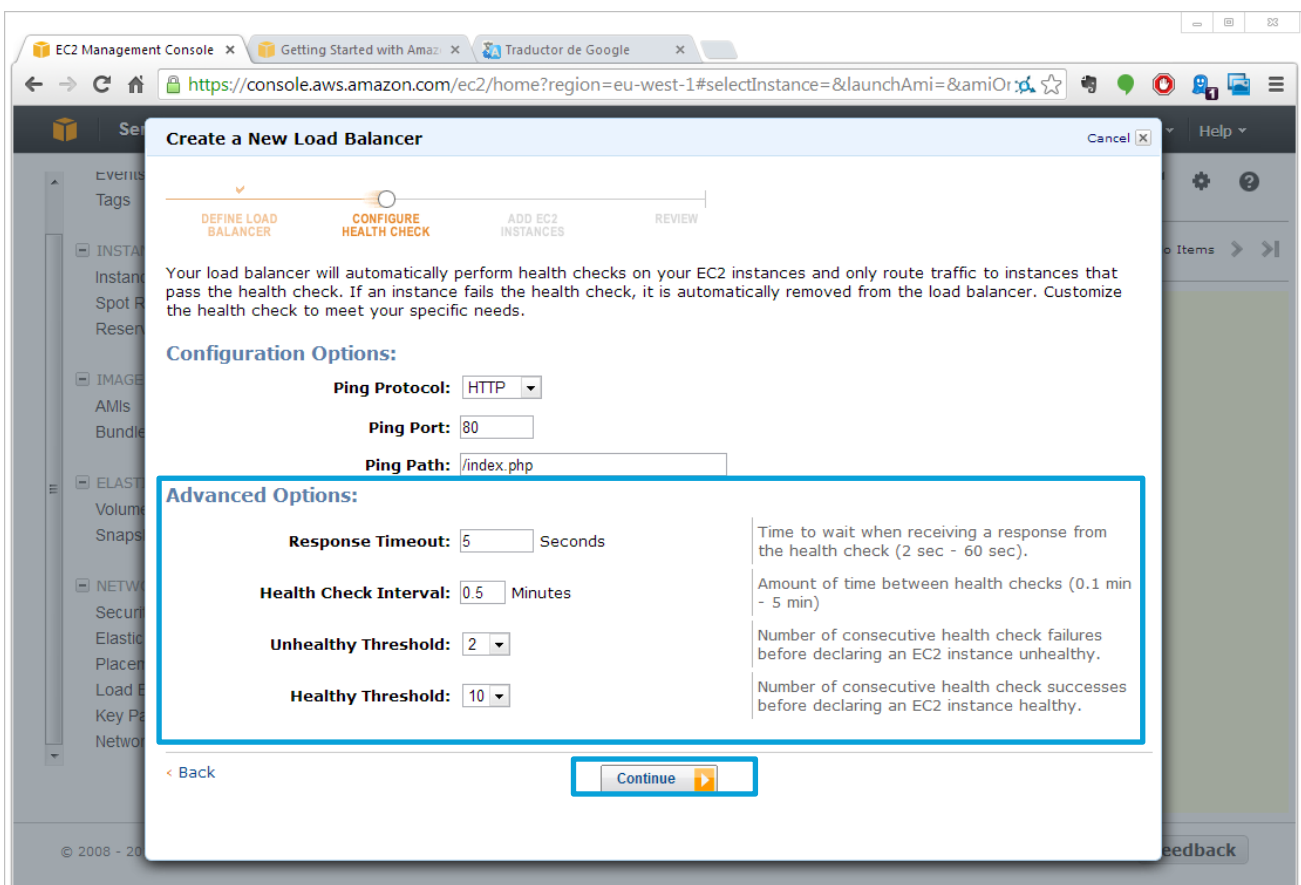


Ilustración 66 - Creación de LoadBalancer paso 02 de 05: Configuración de monitorización

4. **(Ilustración 67)** En este punto seleccionamos el grupo que de seguridad que le queremos asignar y presionamos en "Continue".

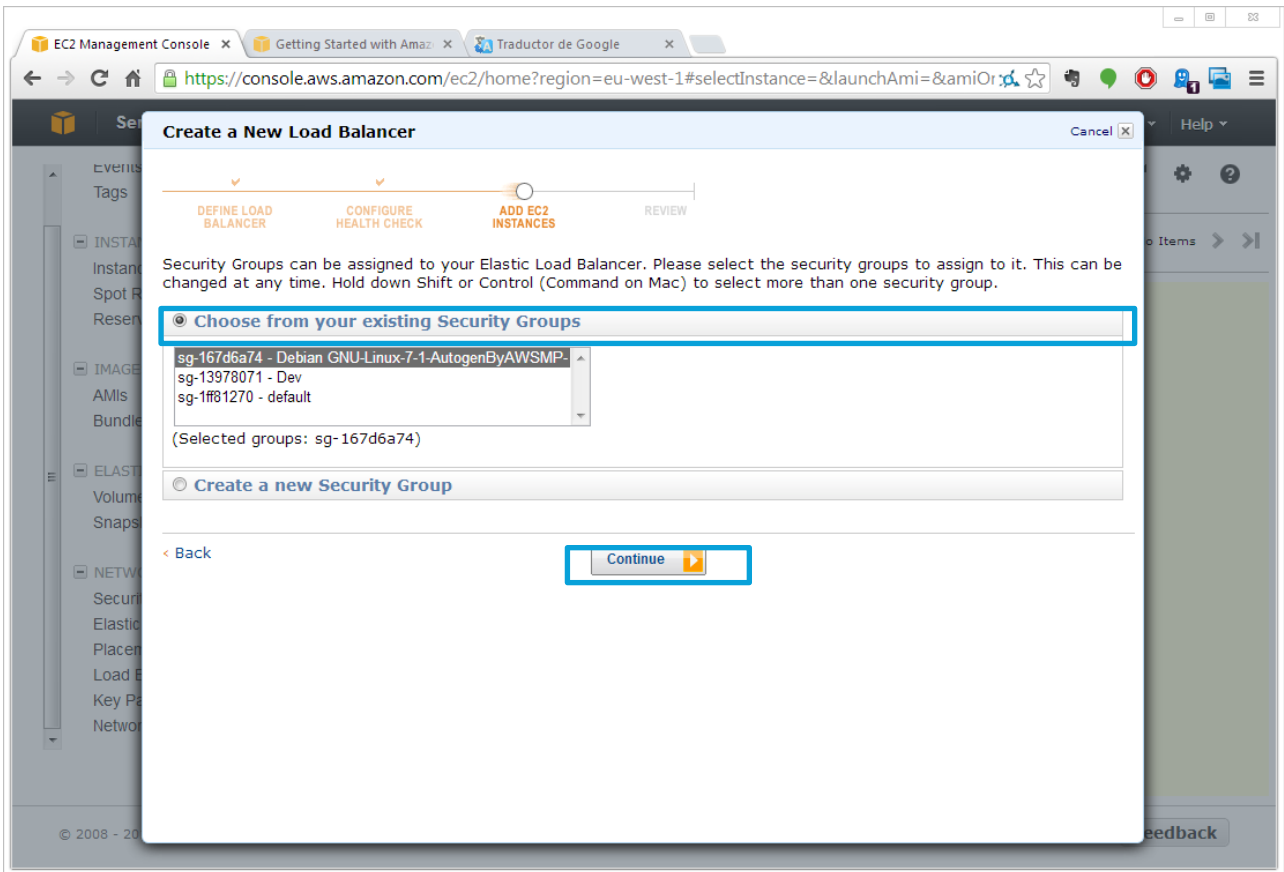


Ilustración 67 - Creación de LoadBalancer paso 03 de 05: Adición de instancias EC2 01

5. **(Ilustración 68)** En este punto seleccionamos las instancias que queremos controlar y entre las cuales queremos distribuir el tráfico. A continuación pulsaremos en "Continue".

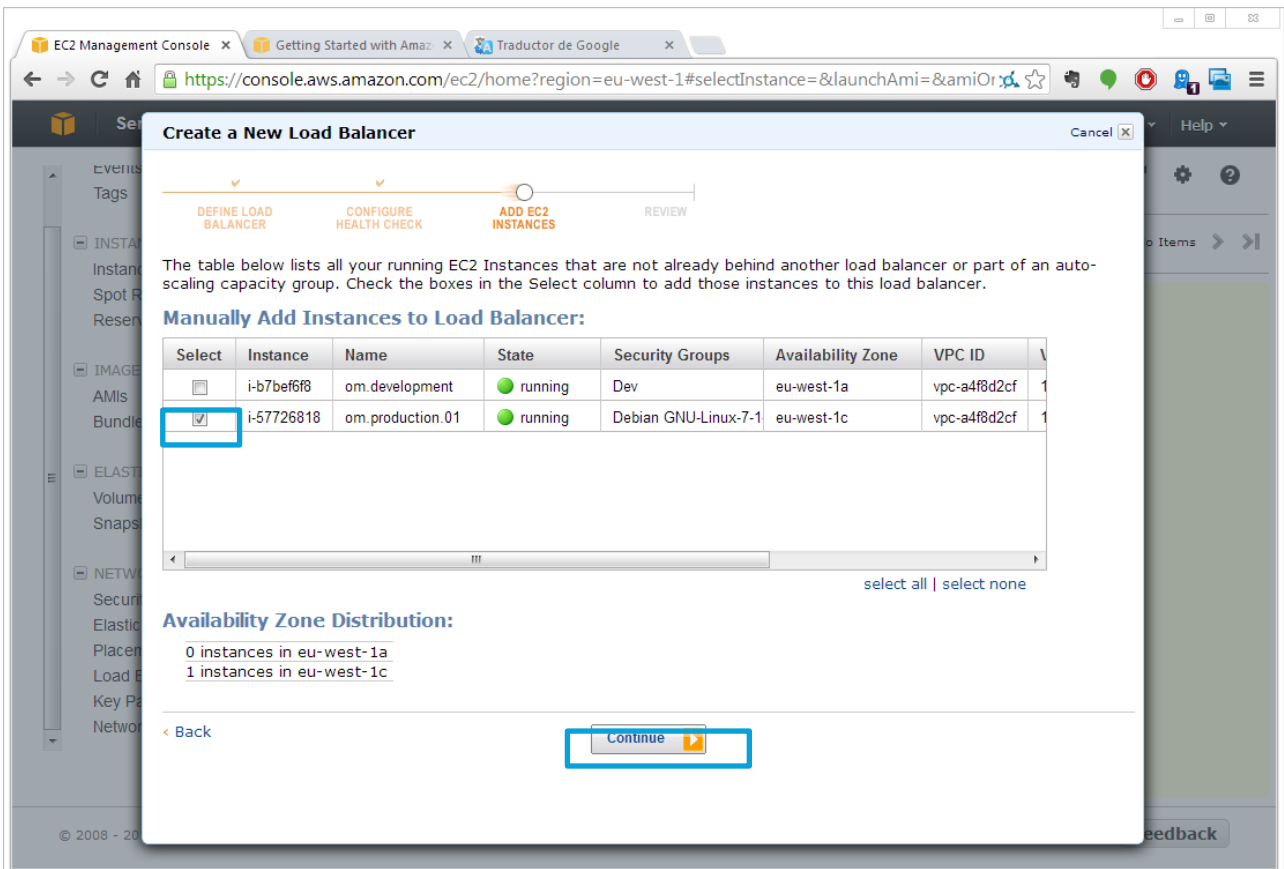


Ilustración 68 - Creación de LoadBalancer paso 04 de 05: Adición de instancias EC2 02

6. **(Ilustración 69)** Por último nos mostrará un resumen de la configuración que hemos establecido y clicaremos en "Create".

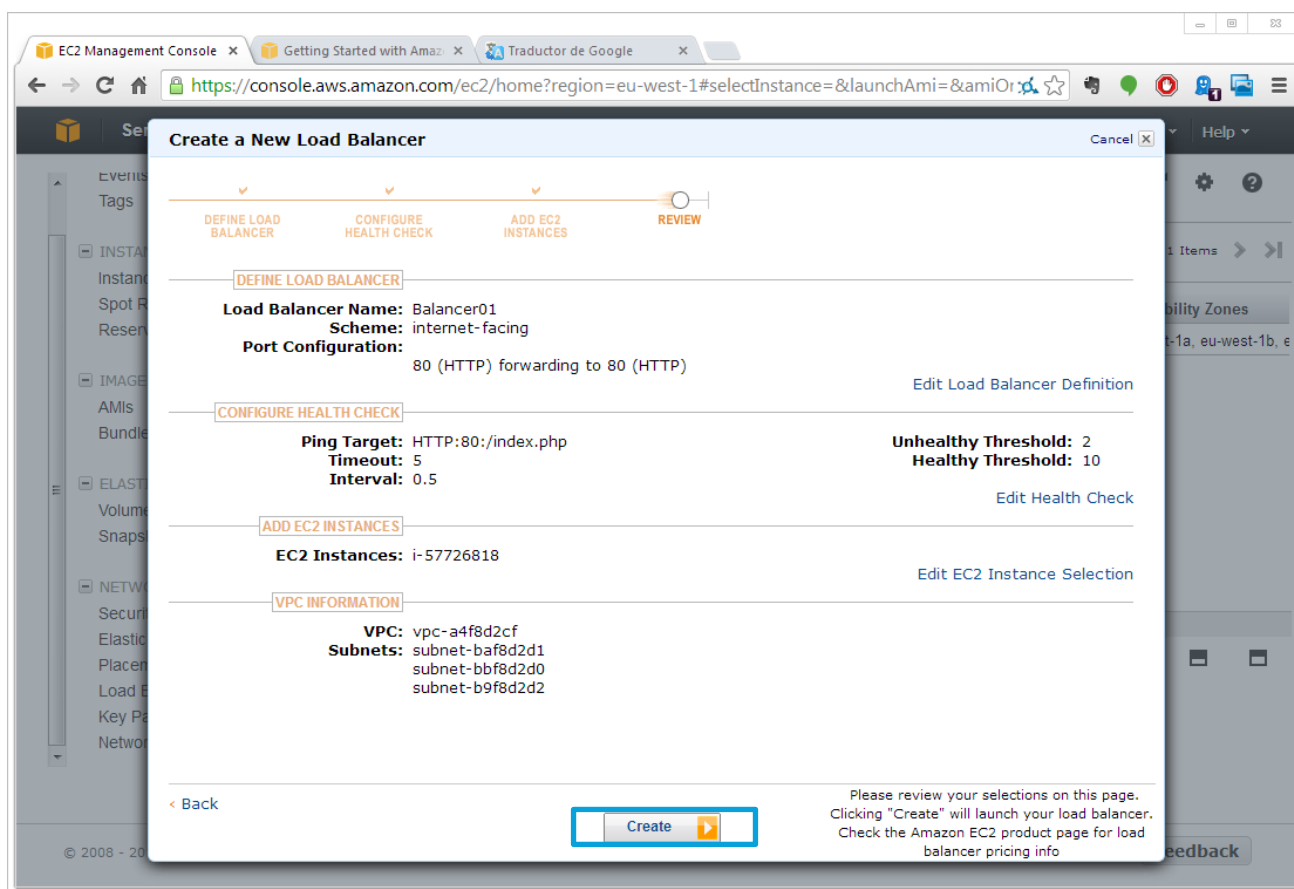


Ilustración 69 - Creación de LoadBalancer paso 05 de 05: Resumen y confirmación

7. Tras un periodo breve de tiempo nos mostrar un mensaje conforme se ha creado. Procedemos a cerrarlo.

3. Amazon RDS

RDS es el servicio de Amazon destinado al hospedaje de base de datos. En este apartado se documentan los pasos más básicos para iniciarse en él: creación de una base de datos y cambio de los parámetros por defecto.

3.1 Creando una base de datos

Para crear una base de datos nos autenticaremos en Amazon AWS y nos dirigiremos a nuestro panel de control. Una vez allí presionaremos sobre RDS en la sección “Databases” (Ilustración 70).

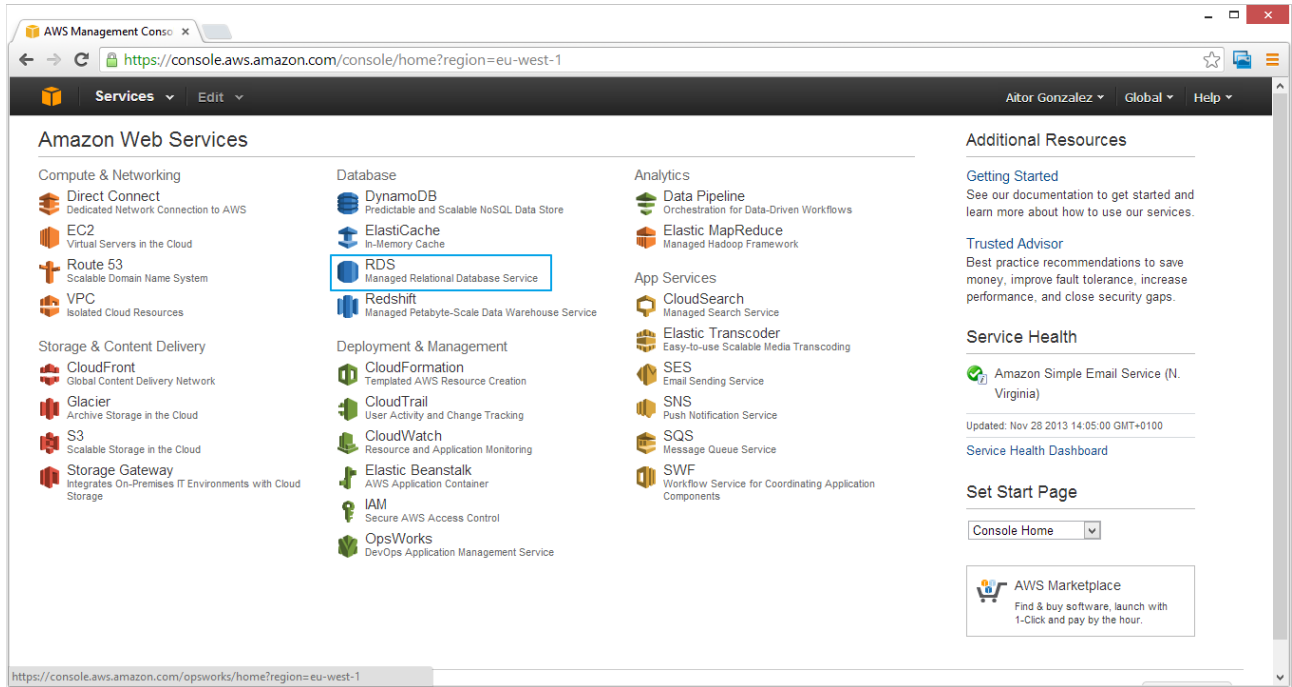


Ilustración 70 - RDS Creación de base de datos: Localización

En seguida iremos a parar a la consola de control del servicio donde seguidamente clicaremos en el botón “Launch a DB instance” (Ilustración 71).

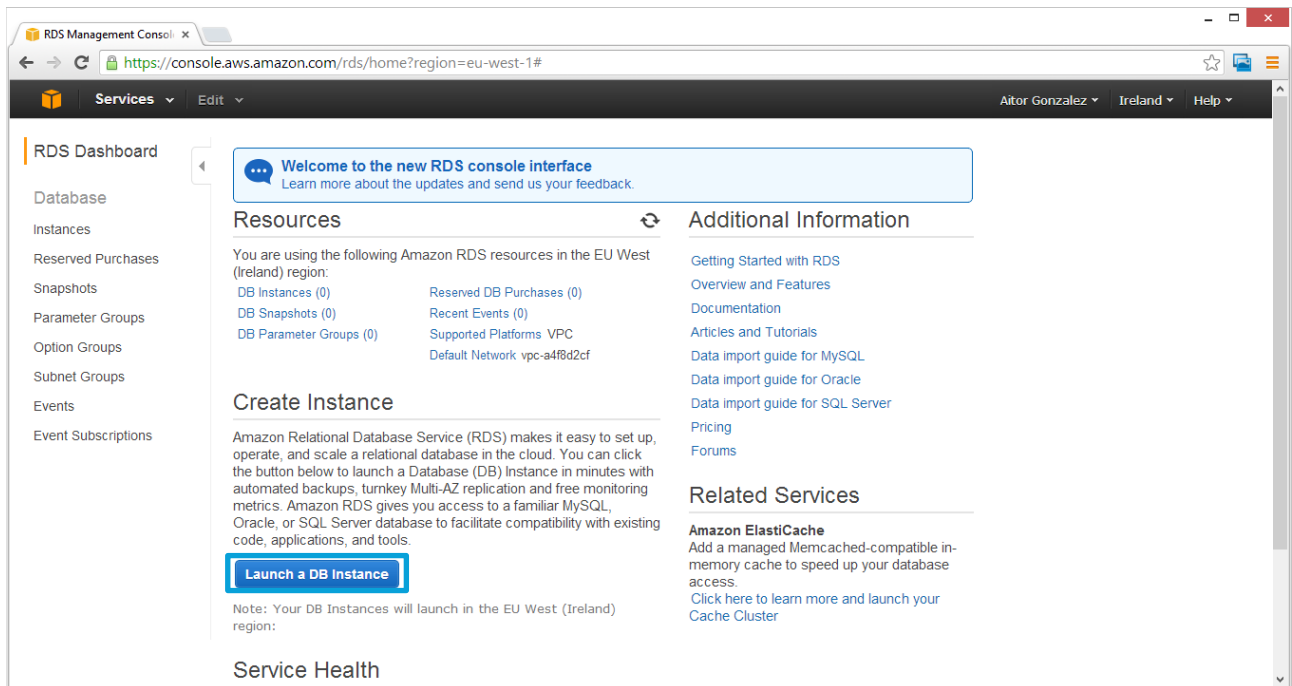
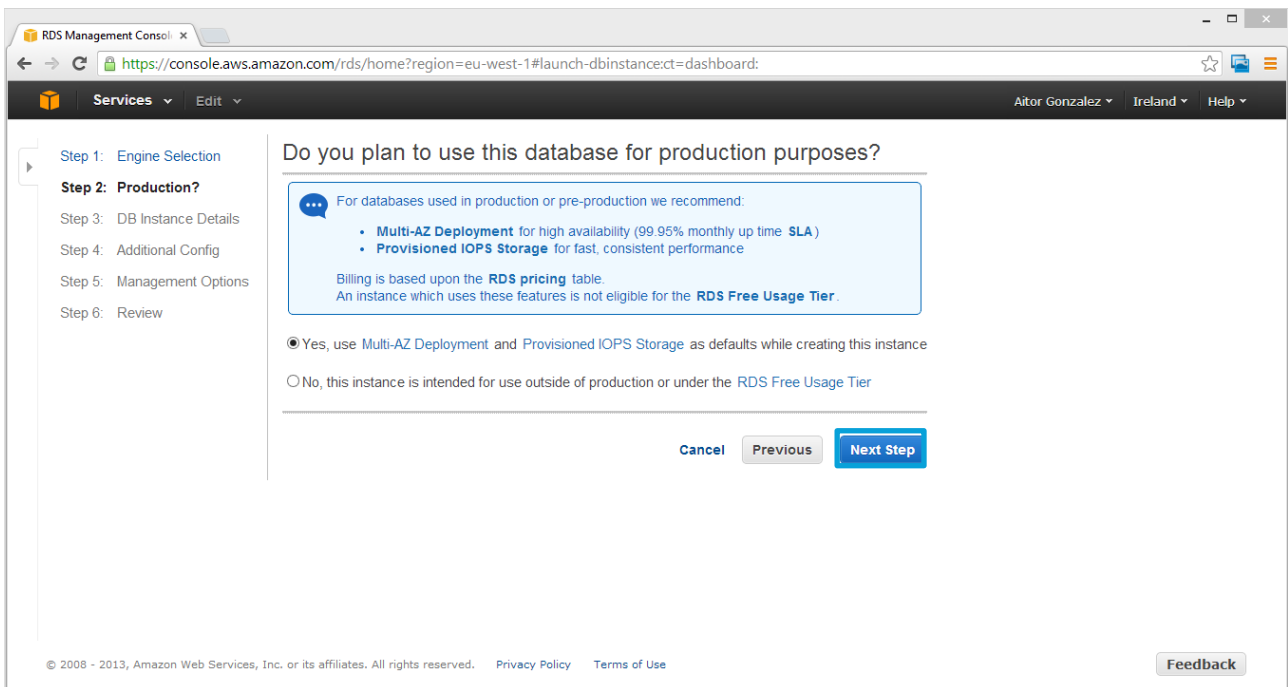


Ilustración 71 - RDS Creación de base de datos: Inicio

Nos aparecerá un primer formulario (**Ilustración 72**) donde tendremos que escoger entre dos modalidades de base de datos: una destinada a la alta disponibilidad y rendimiento y otra más enfocada a desarrollo o *testing*. Escogemos la que más nos convenga y presionamos en “Next Step”.

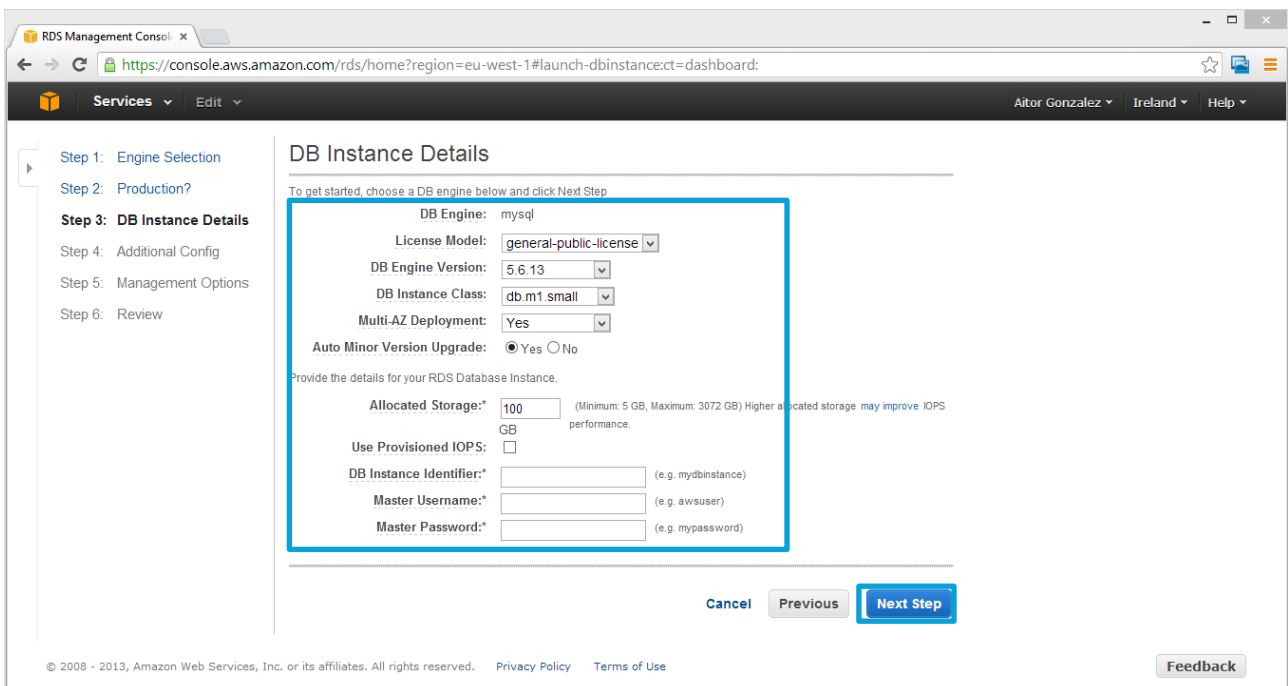
Un punto a tener en cuenta durante este proceso, es que la modalidad *MultiAZ Deployment* no sufre cortes por mantenimiento y además pueden ser usadas entre regiones, mientras que la otra modalidad no.



The screenshot shows the AWS RDS console interface. The browser address bar displays `https://console.aws.amazon.com/rds/home?region=eu-west-1#launch-dbinstance:ct=dashboard:`. The page title is "Do you plan to use this database for production purposes?". On the left, a sidebar lists steps: Step 1: Engine Selection, Step 2: Production? (active), Step 3: DB Instance Details, Step 4: Additional Config, Step 5: Management Options, and Step 6: Review. The main content area features a blue callout box with a speech bubble icon, stating: "For databases used in production or pre-production we recommend: • Multi-AZ Deployment for high availability (99.95% monthly up time SLA) • Provisioned IOPS Storage for fast, consistent performance". Below this, it notes: "Billing is based upon the RDS pricing table. An instance which uses these features is not eligible for the RDS Free Usage Tier." There are two radio button options: "Yes, use Multi-AZ Deployment and Provisioned IOPS Storage as defaults while creating this instance" (selected) and "No, this instance is intended for use outside of production or under the RDS Free Usage Tier". At the bottom right, there are three buttons: "Cancel", "Previous", and "Next Step" (highlighted in blue). The footer contains copyright information and a "Feedback" button.

Ilustración 72 - RDS Creación de base de datos paso 01 de 05: Selección de tipo de instancia

Al presionar “Next Step” nos encontraremos con el segundo formulario (**Ilustración 73**) de configuración de la base de datos. Este es el más sencillo y es autoexplicativo, así que procedemos a rellenarlo y presionamos de nuevo en “Next Step”.



The screenshot shows the AWS RDS console interface for the "DB Instance Details" step. The browser address bar displays `https://console.aws.amazon.com/rds/home?region=eu-west-1#launch-dbinstance:ct=dashboard:`. The page title is "DB Instance Details". On the left, a sidebar lists steps: Step 1: Engine Selection, Step 2: Production?, Step 3: DB Instance Details (active), Step 4: Additional Config, Step 5: Management Options, and Step 6: Review. The main content area features a blue callout box with the text: "To get started, choose a DB engine below and click Next Step". Below this, there are several configuration options: "DB Engine: mysql", "License Model: general-public-license", "DB Engine Version: 5.6.13", "DB Instance Class: db.m1.small", "Multi-AZ Deployment: Yes", and "Auto Minor Version Upgrade: Yes" (selected). Below these, there is a section for "Provide the details for your RDS Database Instance." with fields for "Allocated Storage: 100 GB" (with a note: "(Minimum: 5 GB, Maximum: 3072 GB) Higher allocated storage may improve IOPS performance."), "Use Provisioned IOPS: [checkbox]", "DB Instance Identifier: [text input] (e.g. mydbinstance)", "Master Username: [text input] (e.g. awsuser)", and "Master Password: [text input] (e.g. mypassword)". At the bottom right, there are three buttons: "Cancel", "Previous", and "Next Step" (highlighted in blue). The footer contains copyright information and a "Feedback" button.

Ilustración 73 - RDS Creación de base de datos paso 02 de 05: Especificación de los detalles de la instancia

En este punto nos encontramos ante el penúltimo formulario (**Ilustración 74**). Igual que en el caso anterior es bastante sencillo, aunque merece la pena prestar especial atención a los siguientes campos:

- *Publicly Accessible*: determina si la base de datos es accesible desde Internet o solo desde la red interna de Amazon AWS.
- *Parameter Group*: es una lista de parámetros por defecto para la base de datos. Esta se puede crear previamente a la creación de la instancia.
- *VPC Security Group*: cumple el mismo rol que los grupos de seguridad de las instancias EC2.

Tras rellenar el formulario y revisar que todo es correcto clicamos en “Next Step”.

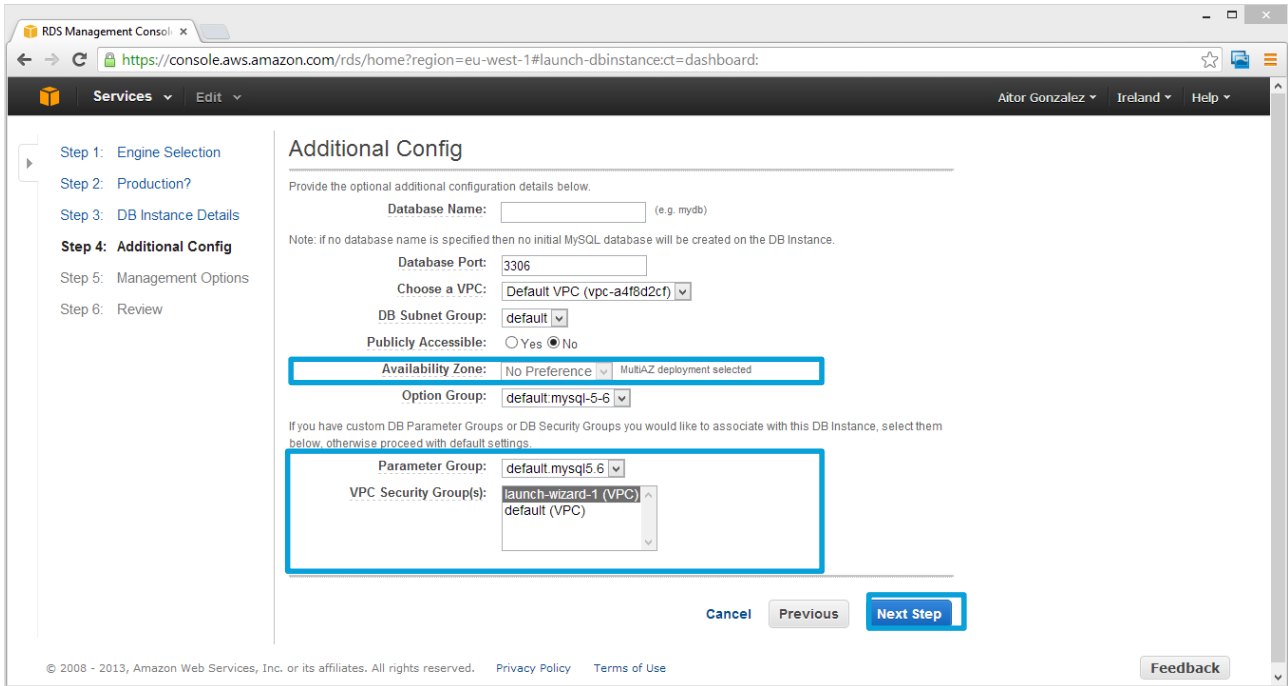


Ilustración 74 - RDS Creación de base de datos paso 03 de 05: Configuración adicional de la instancia

Ahora sí, ya nos encontramos en el último formulario (**Ilustración 75**), en el hemos de definir durante cuánto tiempo se guardan las copias de seguridad de la base de datos, si deseamos que estas se hagan de forma automática, y, quizás lo más complejo, el “*backup windows*” y la “*maintenance windows*”.

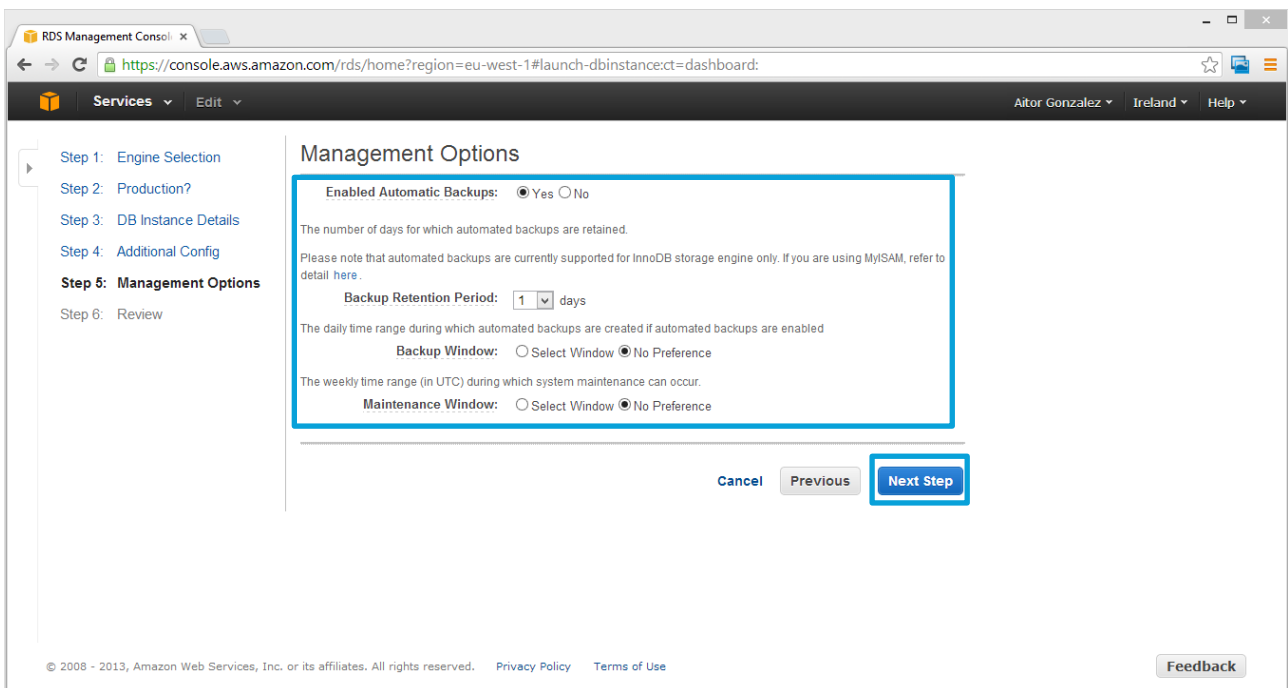


Ilustración 75 - RDS Creación de base de datos paso 04 de 05: Opciones de gestión

El “*backup windows*” determina durante qué periodo de tiempo se desea que se realicen las copia de respaldo. Hay que tener en cuenta que ese proceso, en una no *MultiAZ*, puede provocar que la base de datos no este accesible durante un breve periodo de tiempo.

El “*maintenance windows*” determina en qué periodo de tiempo deseamos que Amazon realice el mantenimiento de la base de datos. Al igual que en el “*backup windows*”, si no es una *MultiAZ*, podría encontrarse no disponible durante el tiempo especificado.

Una vez definidos los parámetros clicamos a “*Next Step*”.

Ahora solo nos queda comprobar ante el resumen (**Ilustración 76**) que se nos presenta que todos los datos son correctos y presionar en “*Launch DB Instance*”.

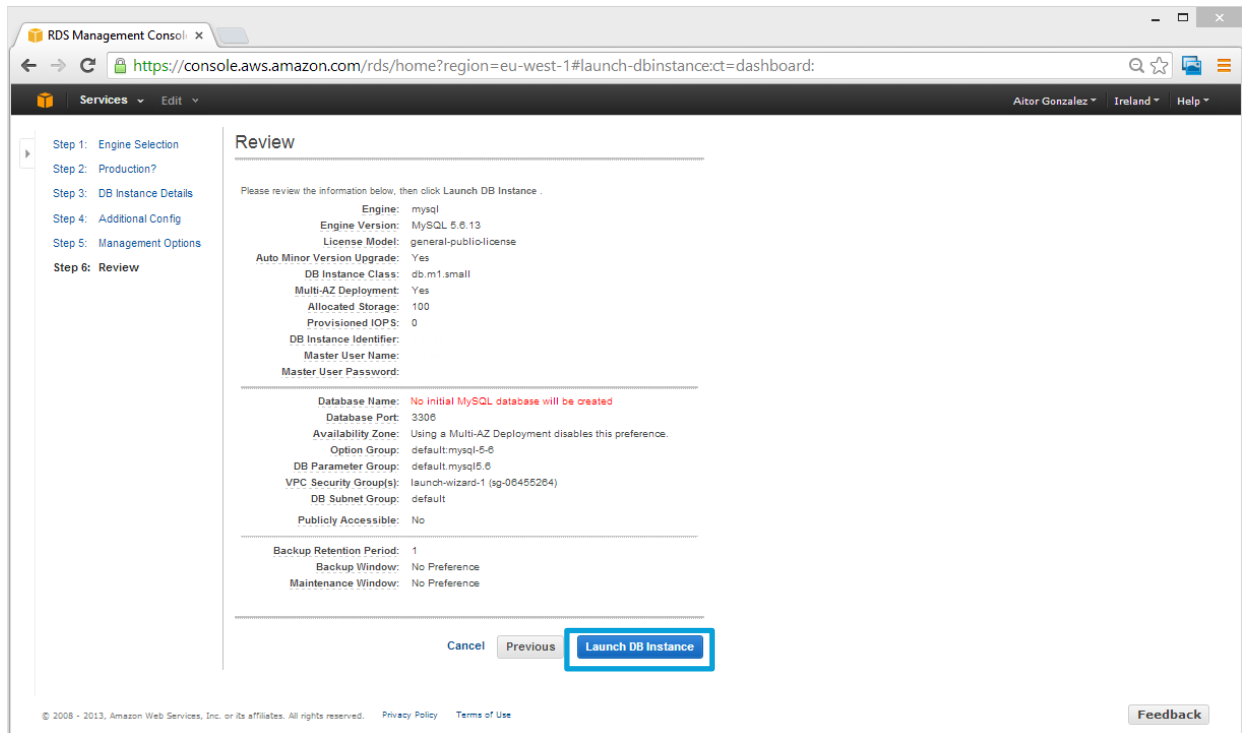


Ilustración 76 - RDS Creación de base de datos paso 05 de 05: Resumen

La base de datos se creará y podremos observar su estado al clicar en “*Instances*” en el menú lateral izquierdo.

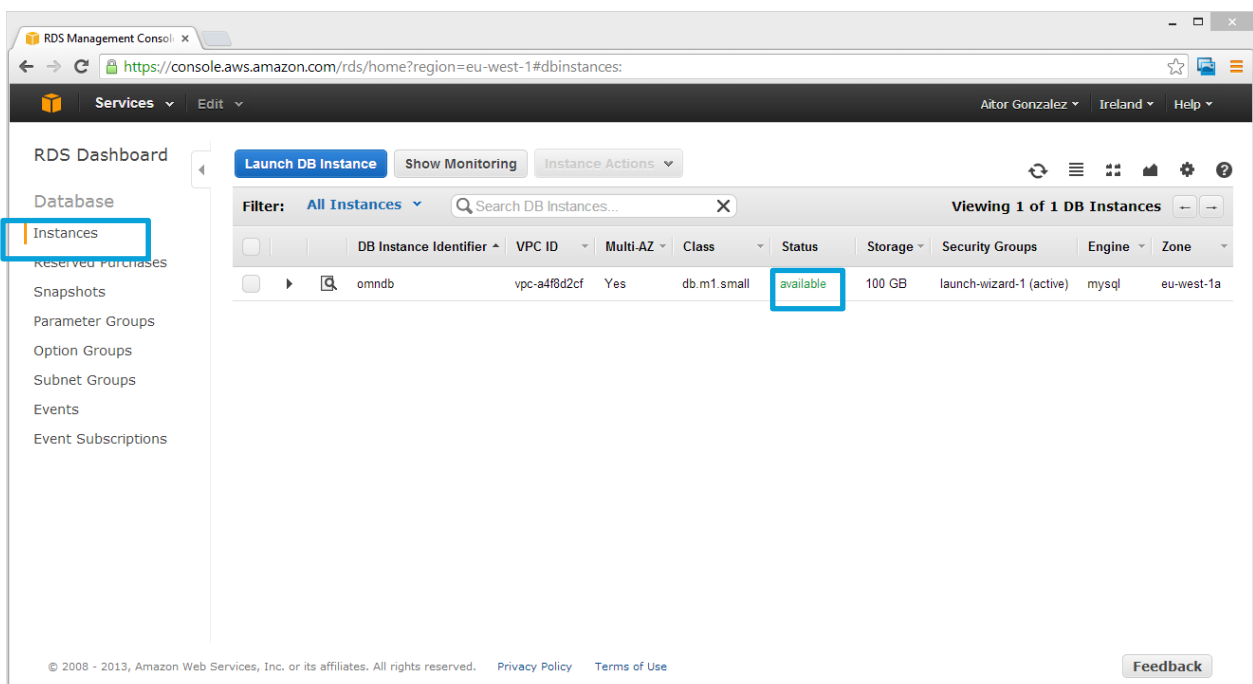


Ilustración 77 - RDS Creación de base de datos: Comprobación de estado

3.2 Cambiando los "Parameter Group".

Los "Parameter Group" son un conjunto de valores de configuración asociados a la instancia que requerirían de acceso al servidor para configurarlos. Como en RDS no tenemos acceso a este, se han de modificar mediante esta opción. Para modificarlos seguiremos los siguientes pasos:

1. **(Ilustración 78)** Accederemos al panel de control de Amazon RDS y presionaremos en el menú lateral izquierdo en el enlace "Parameter Groups".

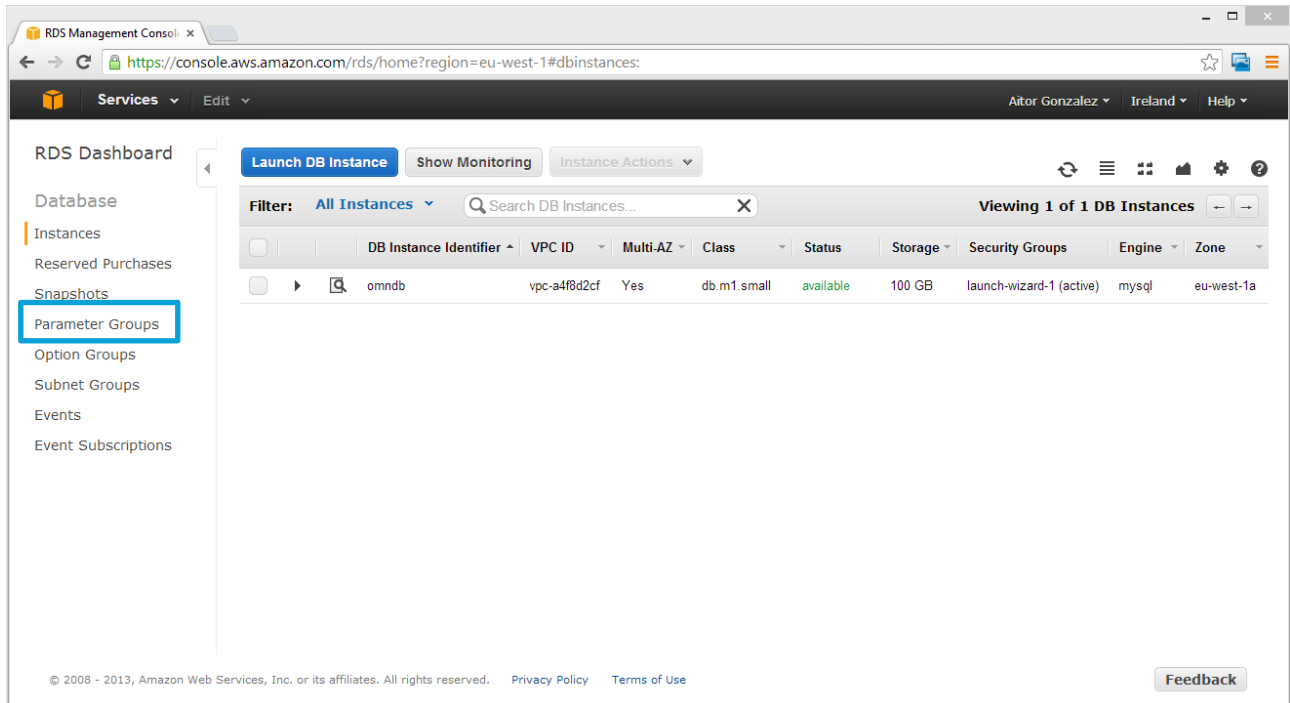


Ilustración 78 - RDS Cambio de parámetros: Inicio

2. **(Ilustración 79)** Clicamos en "Create DB Parameter Group".

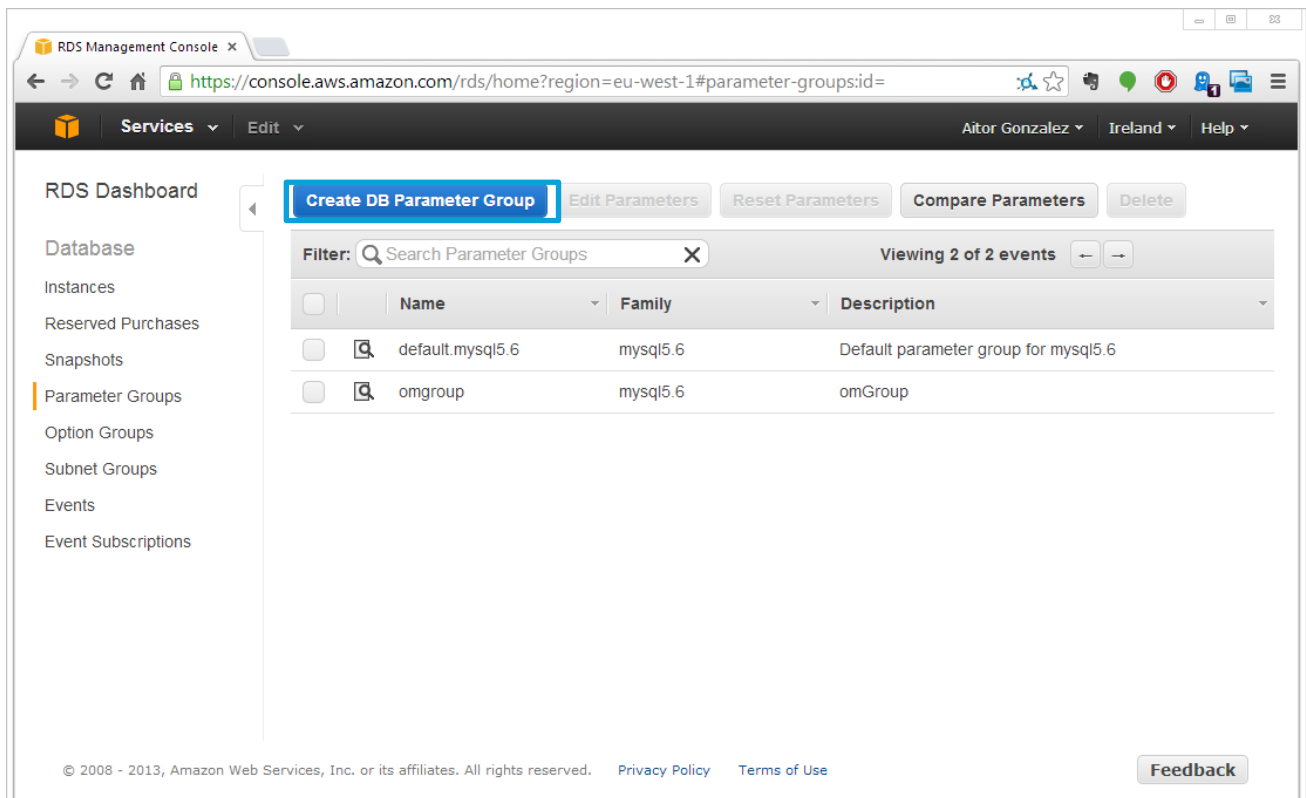
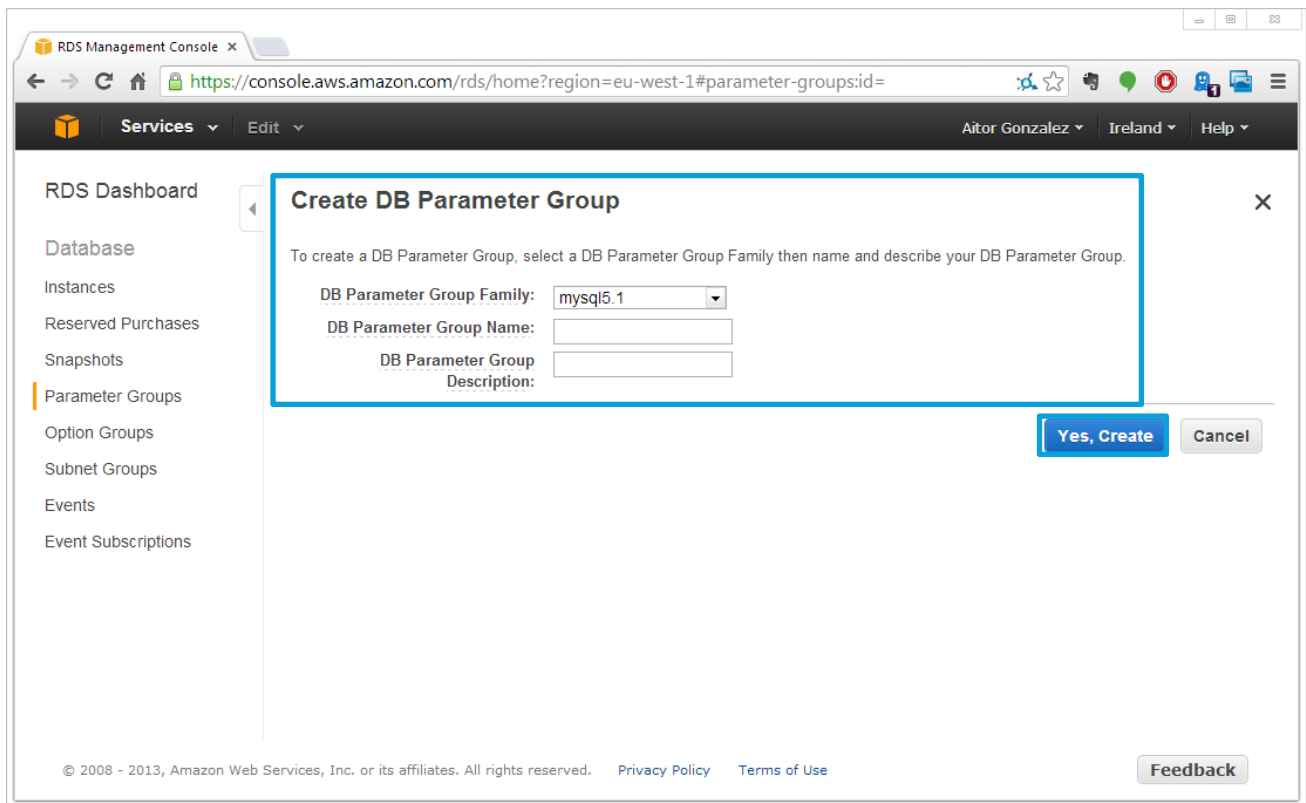


Ilustración 79 - RDS Cambio de parámetros paso 01 de 06: Creando un grupo de configuración

3. **(Ilustración 80)** Nos saldrá un formulario, lo rellenamos según nuestras preferencias y presionamos en “Yes, Create”.



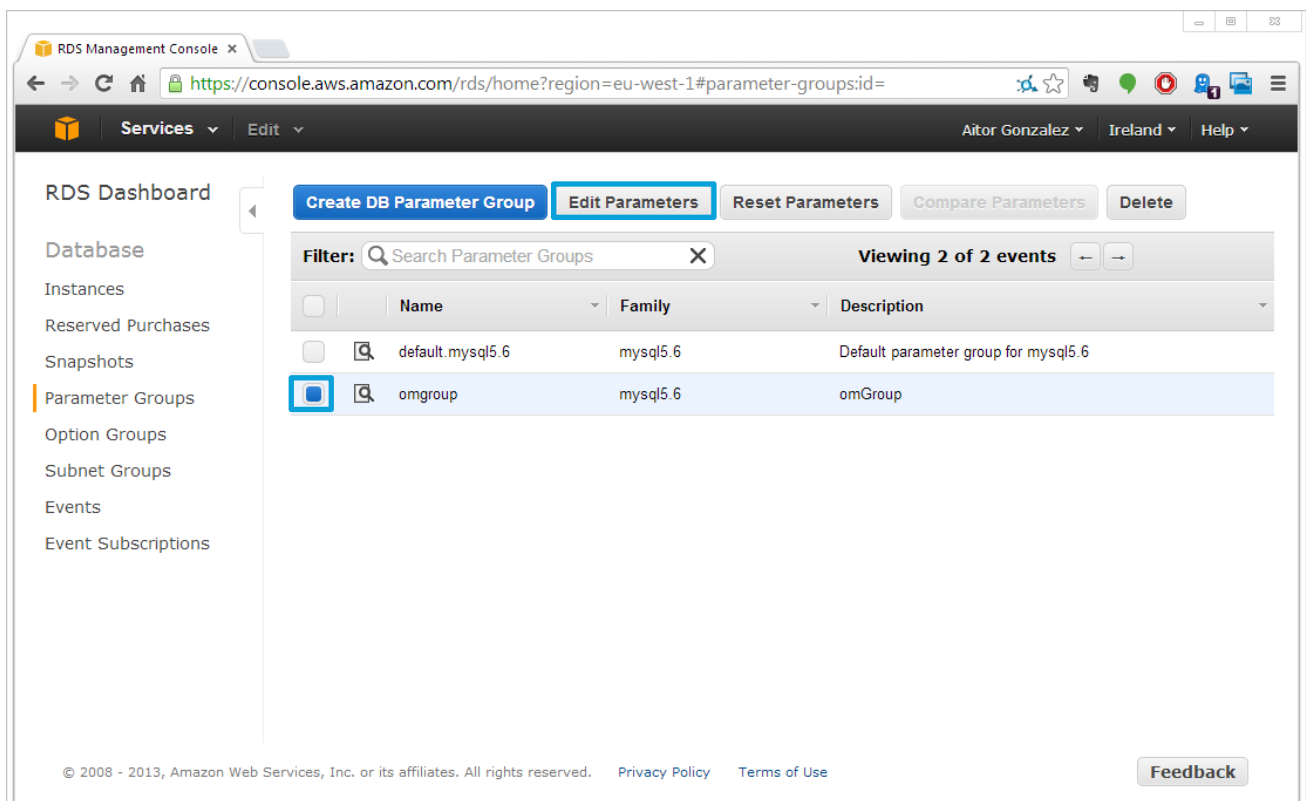
The screenshot shows the AWS RDS Management Console interface. On the left is a navigation menu with 'Parameter Groups' selected. The main content area displays a modal window titled 'Create DB Parameter Group'. The modal contains the following fields and controls:

- DB Parameter Group Family:
- DB Parameter Group Name:
- DB Parameter Group Description:
- Buttons: 'Yes, Create' (highlighted in blue) and 'Cancel'.

At the bottom of the console, there is a footer with copyright information and a 'Feedback' button.

Ilustración 80 - RDS Cambio de parámetros paso 02 de 06: Formulario para la creación del grupo

4. **(Ilustración 81)** Una vez creado nos aparecer junto a los ya existentes. Lo seleccionamos y presionamos en editar.



The screenshot shows the AWS RDS Management Console interface. The 'Parameter Groups' section is active, displaying a list of parameter groups. The 'Edit Parameters' button is highlighted in blue. The list contains the following entries:

	Name	Family	Description
<input type="checkbox"/>	default.mysql5.6	mysql5.6	Default parameter group for mysql5.6
<input checked="" type="checkbox"/>	omgroup	mysql5.6	omGroup

At the bottom of the console, there is a footer with copyright information and a 'Feedback' button.

Ilustración 81 - RDS Cambio de parámetros paso 03 de 06: Inicio de la edición del grupo de configuración

- (Ilustración 82) Buscamos los parámetros que deseamos modificar mediante el buscador, los editamos y finalmente presionamos en "Save Changes".

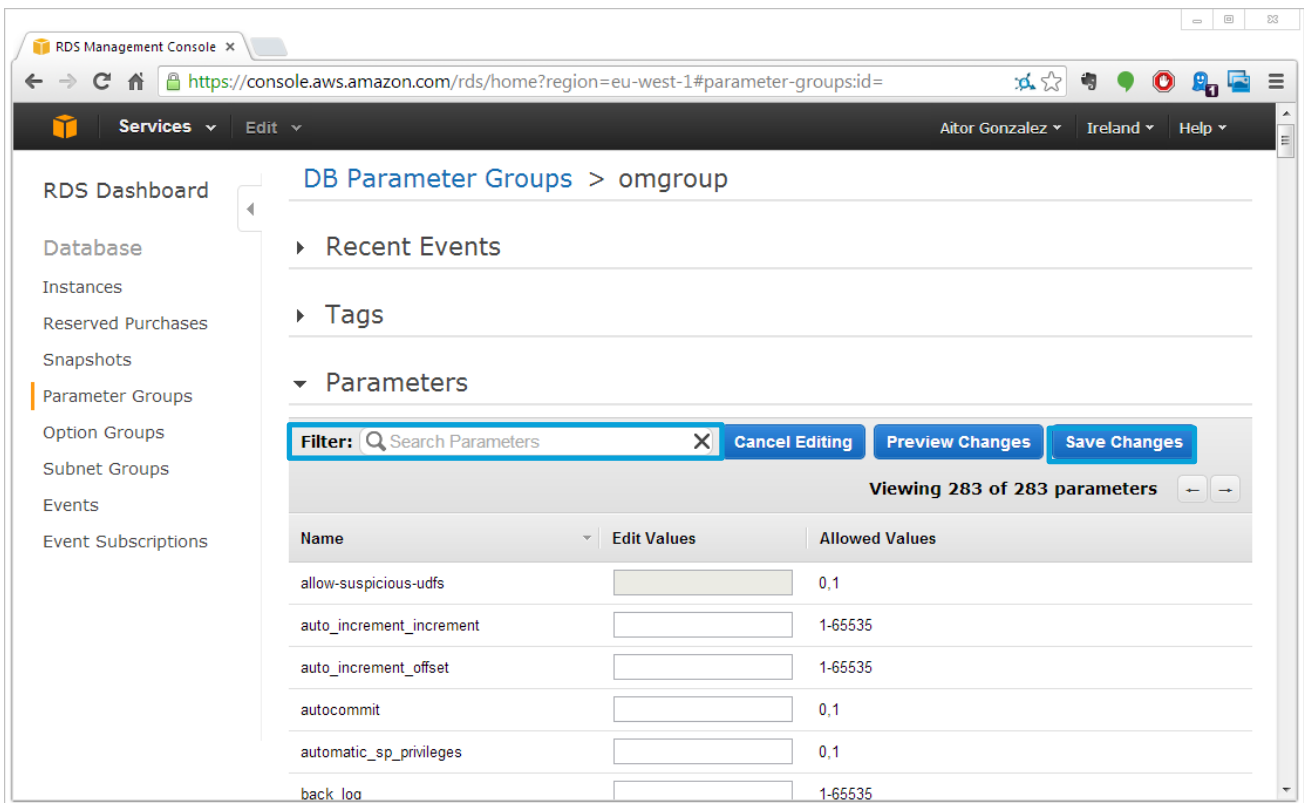


Ilustración 82 - RDS Cambio de parámetros paso 04 de 06: Búsqueda del parámetro a configurar

- (Ilustración 83) Ahora nos dirigimos a las instancias de bases de datos, seleccionamos a la que deseamos cambiarla los parámetros de configuración y presionamos en "Instance Actions > Modify".

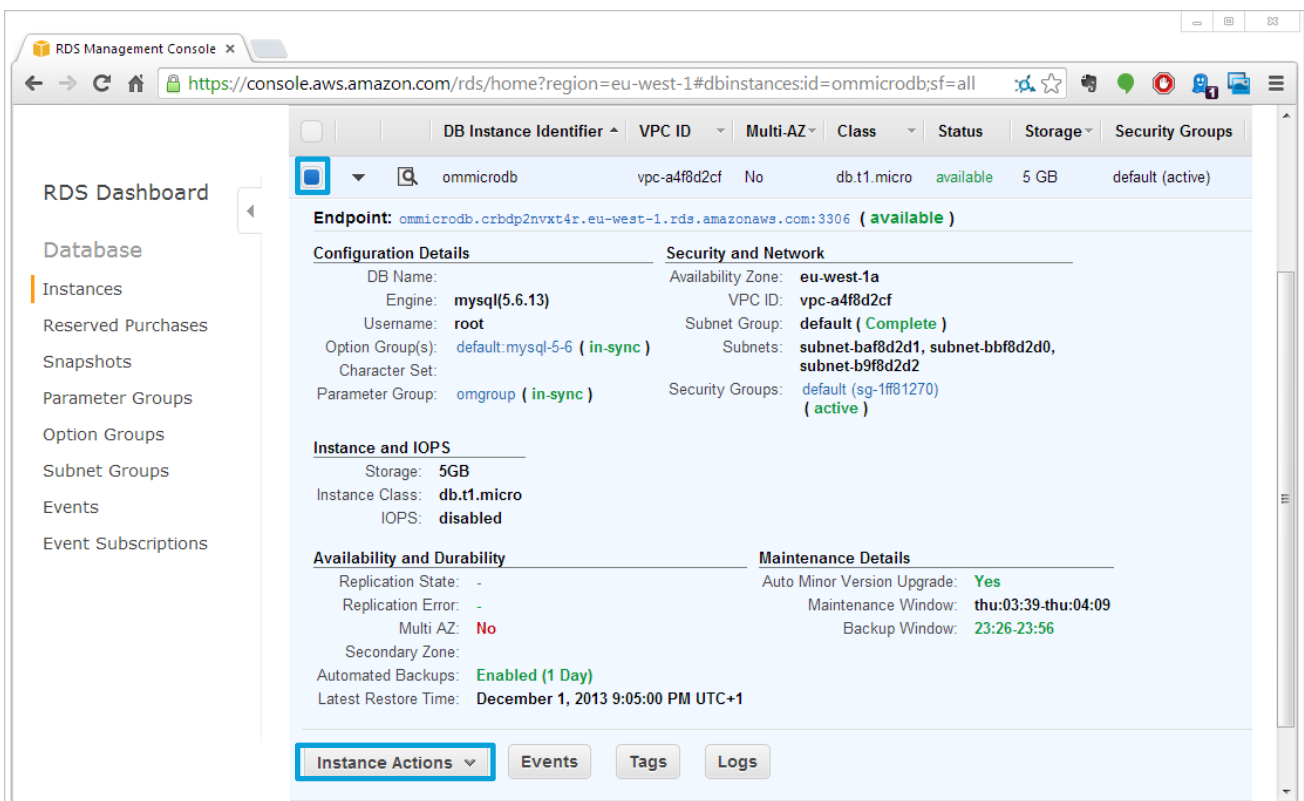


Ilustración 83 - RDS Cambio de parámetros paso 05 de 06: Asignación del grupo de configuración 01

7. **(Ilustración 84)** En el formulario que nos saldrá modificamos el “*Parameter Group*”, seleccionamos el *check* “*Apply Immediately*” y presionamos en el botón “*Continue*”. Terminamos el proceso.

The screenshot shows the AWS RDS Management Console configuration page for a new database instance. The browser address bar shows the URL: <https://console.aws.amazon.com/rds/home?region=eu-west-1#dbinstancesid=ommicrodb;sf=all>. The configuration form includes the following fields:

- DB Instance Identifier:** ommicrodb
- DB Engine Version:** MySQL 5.6.13 (default)
- DB Instance Class:** db.t1.micro
- Multi-AZ Deployment:** No
- Auto Minor Version Upgrade:** Yes (selected)
- Allocated Storage:** 5 GB (Minimum: 5 GB, Maximum: 3072 GB) Higher allocated storage may improve IOPS performance.
- Use Provisioned IOPS:**
- Provisioned IOPS:** RDS MySQL supports IOPS / GB ratios between 3 and 10
- Parameter Group:** ommicrodb (highlighted with a blue box)
- Security Group:** Debian GNU-Linux-7-1-AutogenByAWSMP- (VPC) default (VPC)
- Option Group:** default.mysql-5-6
- New Master Password:**
- Backup Retention Period:** 1 days
- Backup Window:** Start Time: 23 : 26 UTC, Duration: 0.5 hours
- Maintenance Window:** Start Time: Thursday 03 : 39 UTC, Duration: 0.5 hours
- Apply Immediately:** (highlighted with a blue box)
- Continue:** (highlighted with a blue box)

Ilustración 84 - RDS Cambio de parámetros paso 06 de 06: Asignación del grupo de configuración 02

8. Ahora en la sección de instancias de bases de datos seleccionamos la instancia a la que le hemos cambiado el “*Parameter Group*”, clicaremos en “*Instance Actions*” como hicimos anteriormente y seleccionaremos “*Re-boot*” para que se apliquen los cambios.

Una vez se haya reiniciado la instancia la nueva configuración quedara establecida.

4. Amazon S3

S3 es el sistema de almacenaje de ficheros de Amazon en la nube. Es sencillo y fácil de usar, tanto desde la web como desde *scripts*. En este caso solo se documenta el uso desde la interfaz web. Para ejemplos de cómo usarlo desde *scripts* recomiendo leer el Anexo 2 adjuntos a este proyecto.

4.1 Creación del servicio

Para crear el servicio nos dirigiremos a nuestro panel de control de los servicios de Amazon AWS y seleccionaremos sobre el enlace de S3 en la pantalla de inicio (**Ilustración 85**).

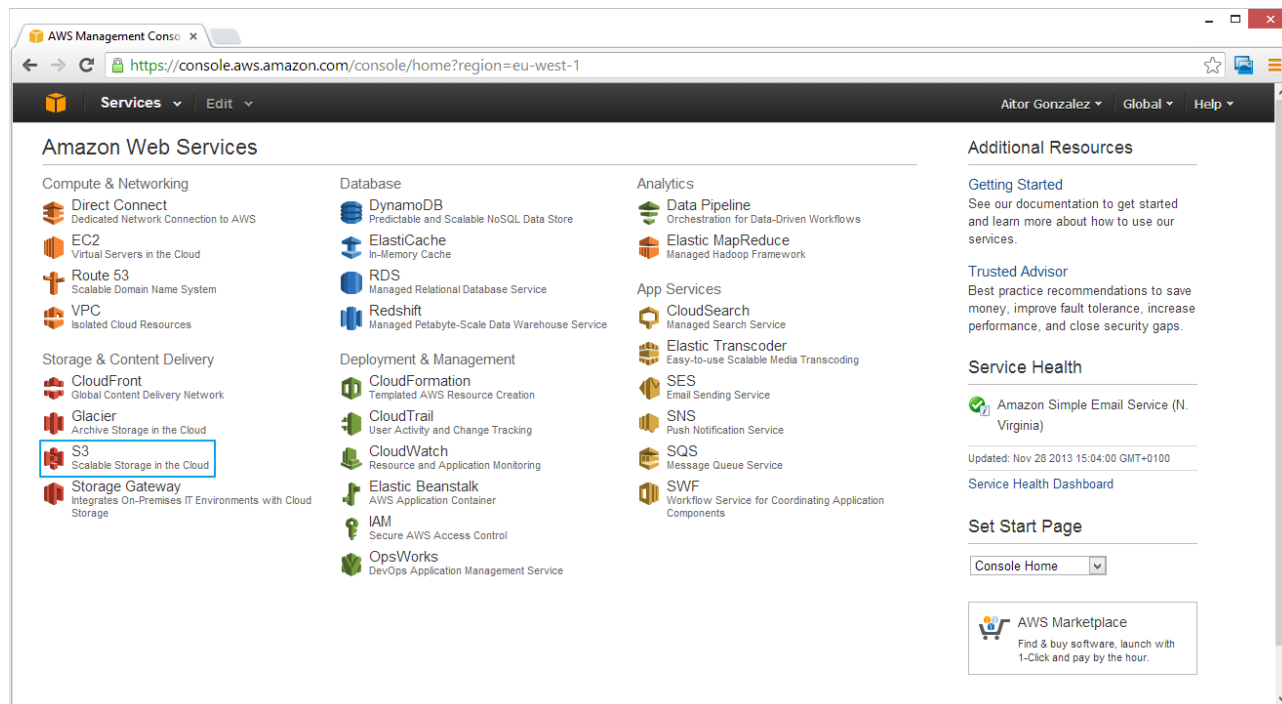


Ilustración 85 - S3: Inicio

Una vez clicado el enlace simplemente tendremos que crear un nuevo *bucket* mediante el botón “*Create Bucket*” visible en el centro de la página (**Ilustración 86**).

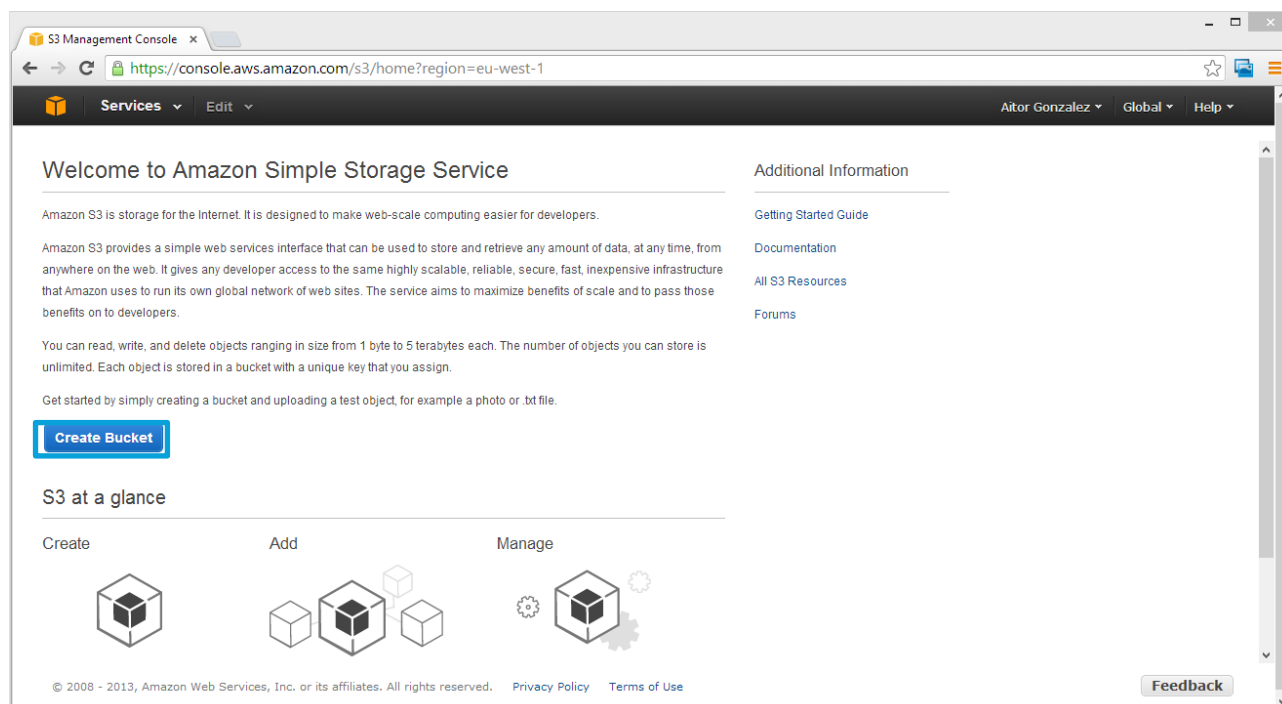


Ilustración 86 - S3 Creación de bucket paso 01 de 03: Inicio

Para crear el *bucket* tendremos que rellenar un formulario (**Ilustración 87**) donde especificaremos el nombre de este y en la región que deseemos que este accesible.

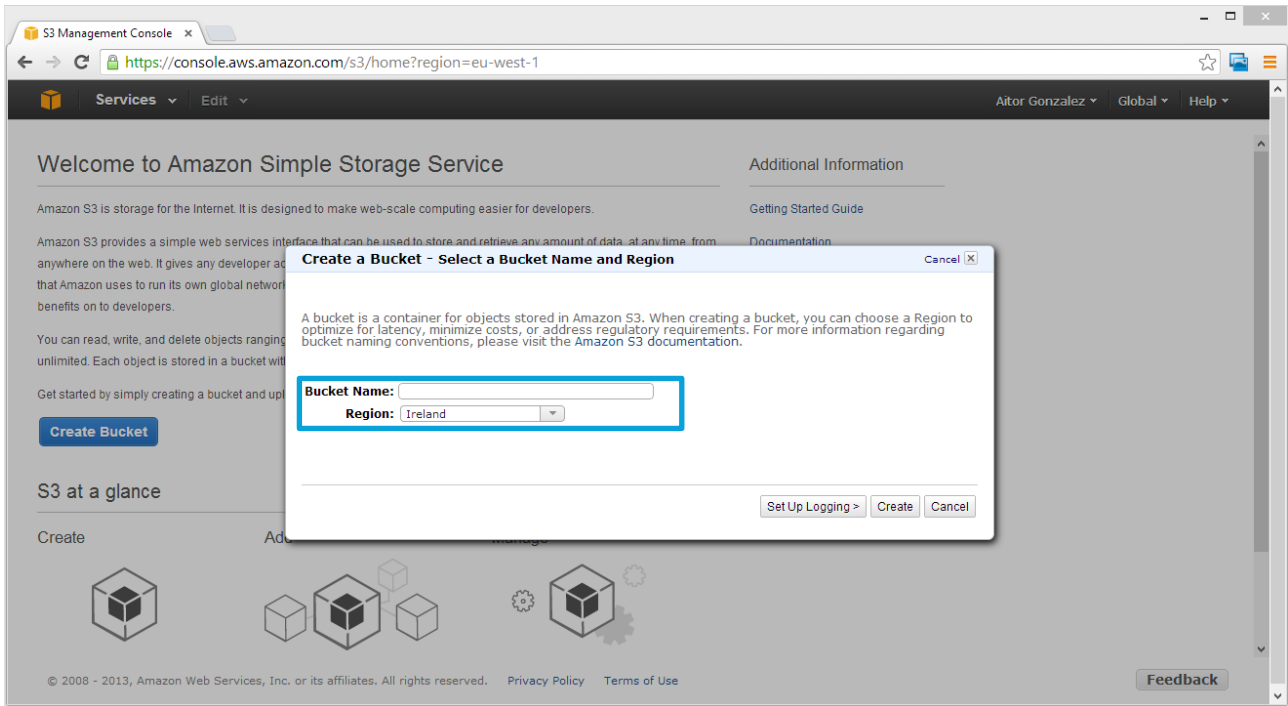


Ilustración 87 - S3 Creación de bucket paso 02 de 03: Formulario de creación

Ya creado lo podremos administrar desde la interfaz. También podremos crear de nuevos o subir/crear ficheros en estos mediante el menú "Actions", tal como se muestra en la **Ilustración 88**.

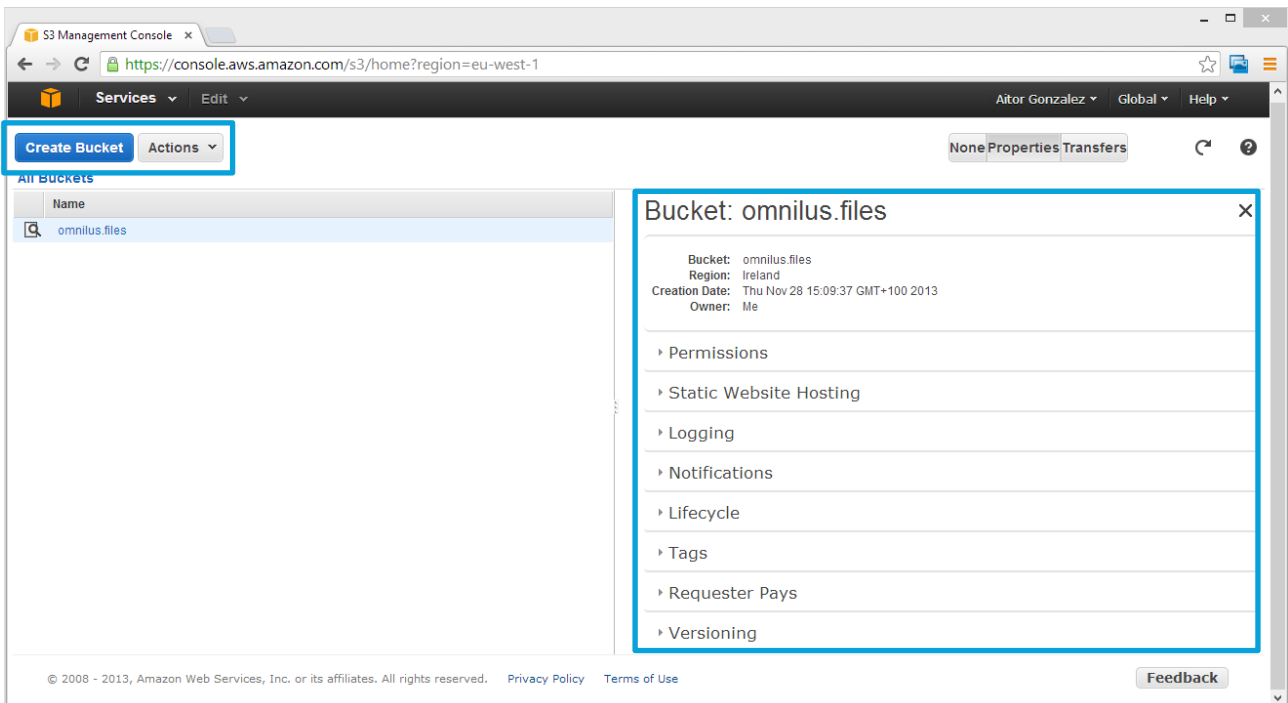


Ilustración 88 - S3 Creación de bucket paso 03 de 03: Configuración y visualización

5. CloudWatch

Cloud Watch es un servicio de Amazon AWS que nos permite monitorizar nuestras instancias. Para activarlo basta con seleccionar una instancia en la interfaz de EC2, clicar en el botón derecho del ratón y presionar en “Enable Detail Monitoring” (**Ilustración 89**).

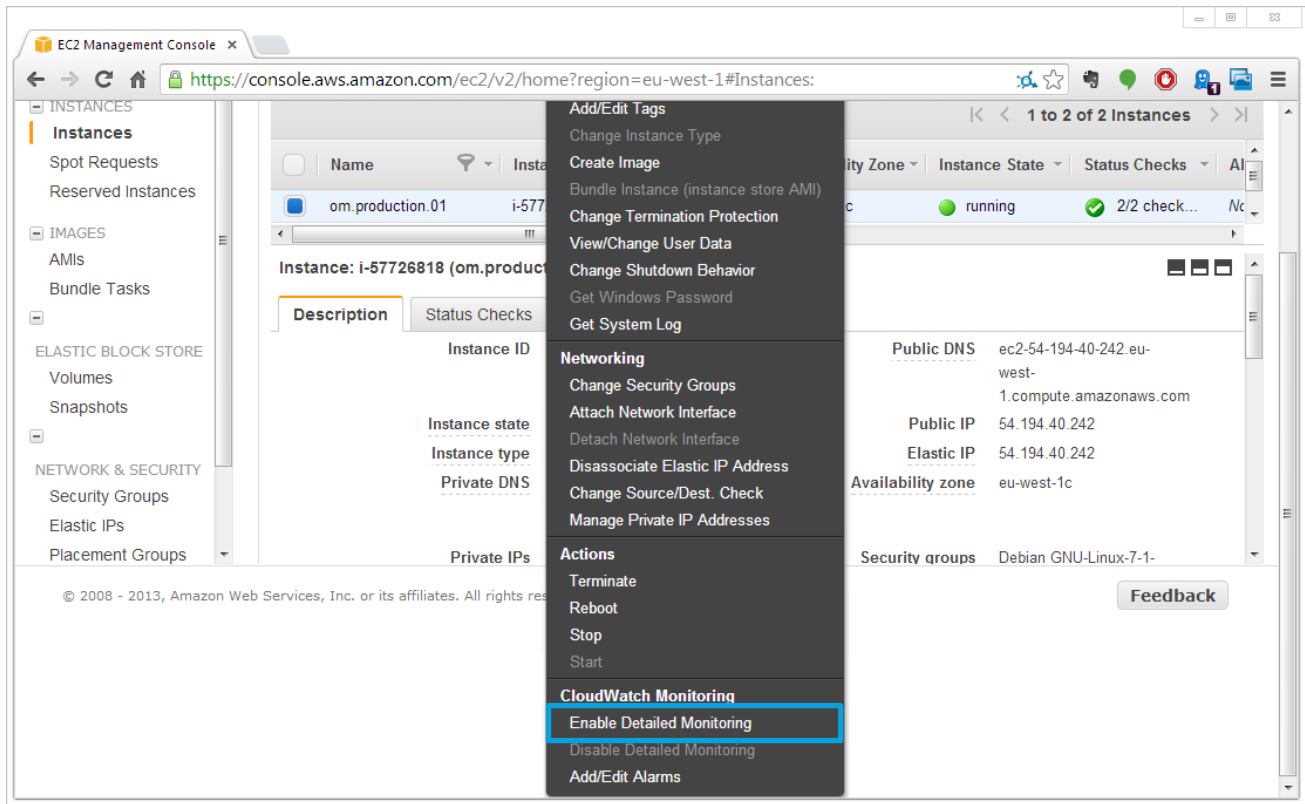


Ilustración 89 - CloudWatch: Activación de la monitorización en una instancia EC2

Tras dos ventanas emergentes sin gran misterio nos informará de que el servicio ha sido habilitado. Para gestionar las alarmas o supervisar la instancia bastará con dirigirnos al servicio desde el enlace en nuestra consola de Amazon AWS, tal como se ha indicado para los demás.

A partir de aquí es experimentar un poco con el servicio. Los puntos más interesantes que nos aporta son la implementación de políticas de autoescalado y la previsión de costes del sistema, puntos totalmente detallados en la documentación de Amazon sobre este servicio, que podemos encontrar en la siguiente dirección web: <http://docs.aws.amazon.com/AmazonCloudWatch/latest/DeveloperGuide/GettingStarted.html>

