Monitoratge i optimització dels sistemes de la plataforma Wattwin allotjats en AWS

Oscar Duran Font

Resum— En aquest projecte s'ha dissenyat i desenvolupat una sèrie de configuracions per a millorar l'estat de la infraestructura de l'aplicació web Wattwin allotjada a Amazon Web Services. Aquestes millores es centren en convertir els sistemes en una arquitectura més estable, eficient, segura i més ben monitoritzada. Per aconseguir tals canvis s'han creat alarmes que prenen accions automàtiques, s'ha redissenyat la xarxa virtual sencera i s'ha automatitzat tota la creació de la infraestructura per a que sigui fàcilment modificable i replicable en cas de necessitat. Amb aquests canvis s'ha aconseguit una millora notable en la maniobrabilitat dels sistemes, s'ha reduït en gran mesura el temps necessàri per detectar un error en l'aplicació web i està més ben preparada devant de canvis futurs.

Paraules clau— Amazon Web Services, Arquitectura de Cloud, Xarxes Prvades del Cloud, Logs, Alarmes, Devops, Fiabilitat, Seguretat, Nginx, Docker, Subxarxes..

Abstract—In this project we design and develop a series of configurations to upgrade the state of the infraestructure of the web aplication Wattwin that is hosted at Amazon Web Services. These upgrades are focused on making the systems of the architecture more reliable, performe better, more secure and to be better monitorized. To achieve said changes in said project we have created alarms that take automatic actions, the virtual private network has been remodeled and all the creation of the infrastructure has been automatized so that the architecture is easily modificable and replicable in case of necessity. With these changes the we have achieved a considerable upgrade in terms of the time needed to find an error in the web application and is the infrastructure of the application is better suited for any future changes.

Index Terms— Amazon Web Services, Cloud Architecture, Virtual Private Cloud, Logs, Alarms, Devops, Reliability, Security, Nginx, Docker, Subnets.

1 Introducció

L'any 2006 Amazon Web Services (AWS) va començar a proporcionar serveis d'infraestructura en el núvol[1], terme més conegut avui en dia com a cloud computing. Des del moment en que es va llençar AWS, milers d'empreses de diferents tamanys l'han utilitzat per a poder satisfer les varies necessitats informàtiques que presentaven cada una d'elles, i l'empresa anomenada Servicios Integrales de Instalaciones Energéticas (SIIE) no va ser una excepció. Fa quatre anys, moment en que es va fundar l'empresa, es va optar per utilitzar AWS per a poder obtenir la infraestructura necessària per desplegar el producte que ven SIIE, una aplicació web anomenada Wattwin.

Aquest treball consisteix en la millora de l'estat actual tant de l'arquitectura general com de l'ús que es fa dels sistemes allotjats a AWS. El projecte busca que l'empresa tingui més control sobre els seus servidors i que sigui capaç de respondre davant de la majoria d'imprevistos. Aquestes millores tenen com a objectiu que els nous sistemes siguin més fiables, segurs, eficients, que estiguin optimitzats en relació amb el preu i que segueixin el model d'excel·lència operacional per respondre millor a les noves necessitats que SIIE pugui tenir.

- *E-mail de contacte: oscar.duranfo@e-campus.uab.cat*
- *Menció realitzada: Tecnologies de la Informació.*
- *Treball tutoritzat per: Sergi Robles Martínez (Departament d'Enginyeria de la Informacio i de les Comunicacions)*
- *Curs 2019/20*

1.2 Motivació

Crear un software és com construir un edifici. Si els ciments no són sòlids els problemes estructurals poden comprometre la funcionalitat del edifici. En el moment de la creació de Wattwin no es varen tenir en compte aquests ciments i, ara que l'empresa està creixent tant en clients com en plantilla, és fa obvia la necessitat de replantejar moltes de les decisions preses en el passat.

Els sistemes no són estables, i una gran quantitat de vegades, quan alguna màquina falla no es pot saber quina ha sigut la causa, ja que no existeix informació que permeti veure l'estat dels sistemes. Aquesta ceguera devant dels problemes de la infraestructura i la poca estabilitat d'aquesta són els motius principals que van fer que aquest projecte és portés a terme.

1.3 Objectius

Quan es dissenya una infraestructura sempre s'han de tenir en compte els cinc pilars sobre els que es cimenta

una bona arquitectura. Essent aquests; eficiència, fiabilitat, cost òptim, seguretat i desenvolupament enfocat d'excel·lència operacional[2]. Al llarg del treball les decisions preses sobre el disseny dels elements dels sistemes s'han fet sempre valorant l'equilibri entre aquests cinc principis.

A continuació es mostren per ordre d'importancia els diferents objectius que conformen el projecte:

- Estudiar de l'estat actual i crear d'una documentació adient als sistemes i serveis dels que disposa l'empresa.
- Separar les diferents bases de dades de cada un dels entorns.
- Crear eines de backup per poder reaccionar en cas d'emergencies.
- Reordanar i revisar els diferents logs que es creen actualment en les diverses màquines, ja que no ofereixen la informació necessària o simplement directament no són accessibles.
- Crear un sistema d'alertes per tal de poder-se anticipar als incidents que puguin afectar als diferents sistemes.
- Utilitzant el servei de CloudFormation crear un script capaç d'aixecar els diferents entorns de l'empresa de forma eficient i homogènia.
- Reestructurar la Virtual Private Cloud (VPC) de l'empresa per millorar-ne la funcionalitat.
- Revisar la gestió de grups de seguretat i crear capes de seguretat addicionals en cas de que es consideri necessari.
- Connectar les alarmes creades en AWS i la pròpia plataforma per tal de poder mostrar el monitoratge a temps real.

1.4 Metodologia i planificació

Deguda a la naturalesa del projecte una metodologia tradicional de desenvolupament en cascada no tenia cap mena de sentit, aquest fet és degut a que el treball requereix d'un estudi inicial en el qual es poden redefinir molts requisits i es poden plantejar diferents objectius dels que es podrien pensar en un moment inicial.

També és important destacar que la majoria d'objectius són independents entre ells i es poden separar en diferents mòduls, que s'aniran assolin en ordre d'importància al llarg del projecte.

Per aquests motius, un cop vist l'abast del projecte s'ha acabat decidint per un model de metodologia iterativa incremental per tal de poder-se adaptar als canvis que arribaran de forma inevitable.

Un cop s'hagi demostrat i testejat que un objectiu s'ha complert es passarà al següent. Com que desplegar l'aplicació en entorns nous pot provocar greus problemes a l'empresa si aquests no han sigut degudament implantats, es farà un extra d'èmfasis en les proves del serveis per assegurar que són correctes abans de migrar qualsevol servei previ.

En el apèndix A1 es pot veure un diagrama de gantt que representa la planificació inicial. Aquesta planificació va haver de ser modificada, com es pot veure en el apèndix A2, degut a que després de realitzar un estudi de l'estat actual dels sistemes es va veure com una gran necessitat el fet de redissenyar i reestructurar la xarxa virtual on residien totes les màquines de l'empresa.

En la Taula 1 es pot veure la distribució d'hores per a cada un dels mòduls que conformen el projecte. Es pot apreciar que la definició de la xarxa virtual i la creació dels logs i de les alarmes són els mòduls principals del treball.

TAULA 1: DISTRIBUCIÓ D'HORES PER TASCA

Etapa	Tasca	Hores estimades
Planificació	Definició del Projecte	10 h
	Formació Sistemes	50 h
	Formació Xarxa Virtual	30 h
Disseny	Disseny logs i alarmes	15 h
	Disseny Xarxa Virtual	15 h
Desenvolupament	Creació de la Documentació	15h
	Separació de les bases de dades	10 h
	Creació de Logs	35 h
	Creació d'alarmes	30 h
	Reestructuració Xarxa Virtual	40 h
	Grups de Seguretat	5 h
	Script de CloudFormation	10 h

2 ESTAT DE L'ART

En aquest apartat es posarà en context AWS dins del marcat fent una petita menció a les altres opcions disponibles quan es busca contractar infraestructura com a servei.

Abans de l'aparició dels serveis de cloud, quan una empresa volia desplegar una aplicació web, volia disposar d'una base de dades pròpia o havia d'utilitzar un software intern es trobava amb el problema de que també havia de disposar de la infraestructura necessària per a poder suportar tals serveis.

Això acostumava a significar que per a empreses petites el cost de construir la infraestructura era més gran que els beneficis que podrien aportar els serveis que es desitjaven, mentre que per les grans empreses sempre comportava una gran despesa per tal d'establir tals serveis.

Aquest paradigma va canviar radicalment quan es va començar a oferir solucions de cloud computing. Infraestructura com a servei disponible per a tothom a un preu molt més accessible que la tradicional opció de construir de zero l'arquitectura.

Actualment existeixen tres grans proveïdors de cloud computing; Amazon Web Services, Microsoft Azure i Google Cloud Computing [3].

Cada un dels diferents entorns ofereix unes característiques que poden interessar més o menys segos el tipus d'empresa. Per exemple, mentre que Microsoft Azure està més encarat a grans empreses que encara puguin treballar amb Datacenters i fa la connexió amb altres tecnologies més accessible, Google Cloud Computing està més encarat a algoritmes de machine learning i d'anàlisi de grans quantitats de dades.

El que és indiscutible és que AWS ofereix més diversitat de serveis que cap altre dels seus competidors fet que el col·loca com a servei més utilitzat actualment.

L'empresa SIIE utilitza AWS des de el seu inici per a poder desplegar la seva aplicació de forma eficient i segura, i aprofita els serveis que ofereix per diverses funcionalitats, com podria ser el control de logs o la connexió amb les diferents bases de dades.

A continuació es fa un petit resum dels sistemes més utilitzats de AWS per l'empresa SIIE i de les diferents màquines i entorns que conformen Wattwin.

2.1 Serveis AWS

AWS ofereix centenars de serveis, des de servei de DNS (Route53) fins a servei de gestió de cues (Simple Queue Service) passant per bases de dades relacionals (Relational Database Service) i no relacionals (Amazon DocumentDB)[4][5], a continuació es fa una breu descripció dels serveis més importants que necessita Wattwin per funcionar.

CloudFormation: CloudFormation et permet crear infraestructura de AWS de forma previsible i periòdica. Graciès a aquest servei es pot definir una plantilla en la qual s'especifica quins serveis vols implementar i de quina manera. A Wattwin s'utilitza una plantilla d'aquest servei per a generar tota la infraestructura a excepció de les bases de dades, que es generen de forma manual o se troben fora de AWS. Un dels objectius principals del projecte és el de millorar aquesta plantilla per a que no sigui única per a tots els entorns.

ElasticBeanstalk: El servei d'ElasticBeanstalk és el principal motor de gestió dels sistemes de Wattwin. ElasticBeanstalk permet implementar aplicacions al núvol sense haver-se de preocupar de molts dels aspectes de la configuració, com podrien ser permisos d'usuaris o relacions de dependències entre diferents serveis d'AWS. Aquest servei però, té una gran desventatja, com més automàtica és la creació dels entorns menys control tens sobre aquests. Al llarg del projecte s'ha anat assumint part de la feina que feia ElasticBeanstalk per a tenir més control sobre el resultat final. Això no vol dir que la gestió hagi passat d'automàtica a manual sinó que s'ha utilitzant CloudFormation o altres vies de configuració per a atenuar l'afecte negatiu d'ElasticBeanstalk sobre la infreastructura final.

Amazon EC2: Amazon Elastic Compute Cloud (Amazon EC2) és el nucli de qualsevol infraestructura de AWS, ja que és un servei web que ofereix capacitat de còmput adaptable que es pot utilitzar per compilar i servir el teu software. Wattwin utilitza 16 màquines EC2 per poder funcionar. Entre aquestes màquines es troben les API que serveixen el back-end de l'aplicació, les Batch que és on s'executa el còmput més pesat i dos serveis externs; el Deepstream, un servei que facilita el realTime en l'aplicació a través de webSockets[6] i el Camunda, una plataforma de flux de treball i automatització de decisions de codi obert que s'utilitza per crear flux de treball i models de decisió, operar models desplegats en producció i permetre als usuaris executar tasques de flux de treball assignades a ells[7].

S3: Amazon Simple Storage Service (S3) és un servei d'emmagatzemament en el cloud, que es pot utilitzar per guardar i recuperar qualsevol quantitat de dades en qualsevol moment. Wattwin utilitza aquest servei per diverses funcions però la més important és la de servir el front-end de l'aplicació web.

RDS: Relational Database Service és un servei que facilita les tasques de configuració, utilització i escalat d'una base de dades relacional en el núvol. Wattwin en si no utilitza cap base de dades relacional però el servei de Camunda en necessita una per a poder funcionar. Les dades de Wattwin es troben fora de AWS, gestionades directament per cloudmongoDB. Aquesta separació es va fer pensant en que tot el que fossin dades persistents era millor tenir-les emmagatzemades fora de AWS per així poder configurar i experimentar amb AWS amb el menor risc possible estant segur de que les dades es trobaven aïllades en cloudMongoDB.

CloudWatch: CloudWatch proporciona una solució de monitoratge escalable i flexible. Gran part d'aques projecte es centra en treure el màxim partit a aquest servei. En un inici aques servei només s'utilitzava per rebre uns logs de la API i de la Batch en un document de texte sense cap mena de format, però CloudWatch ofereix moltíssimes possibilitats com per exemple la implementació d'alarmes, la parametrització de logs per a poder realitzar consultes o el recull de mètriques per valorar l'estat dels sistemes.

VPC: Virtual Private Cloud (VPC) permet llençar els serveis mencionats anteriorment dins d'una xarxa virutal aïllada de la resta d'Internet. La xarxa actual de Wattwin

necessita un redisseny ja que s'ha anat creant de forma manual al llarg dels anys, afegint quan calia nous elements sense seguir una metodologia adient. Aquesta redisseny ha acabat sent un dels objectius principals del projecte.

CloudFront: Amazon CloudFront agilitza la distribució de contingut web estàtic i dinàmic, com podrien ser arxius .html, .css, .php, imatges i arxius multimèdia. Quan els usuaris sol·liciten contingut, CloudFront ho entrega a través d'una xarxa mundial d'ubicacions buscant minimitzar la latència i maximitzar el rendiment. Wattwin ha incorporat aquest servei recentment per a poder oferir més ràpidament la seva aplicació web. Degut a que els servidors d'amazon es troben a Irlanda i existeix una distribució de CloudFront a Madrid la millora de temps de resposta és d'una importància considerable.

Lambda: Lambda permet executar codi sense haver de provisionar ni administrar servidors. A Wattwin s'utilitza aquest servei per afegir les capçaleres necessàries al servei de CloudFormation i per gestionar els logs que CloudFormation genera.

Wattwin utilitza altres serveis, com el de DNS o el de gestió de cues però els mencionats anteriorment són els principals i els que més es veuen afectats per aquest treball de fi de grau. Un cop vistos els serveis utilitzats en la següent secció es descriurà com aquests serveis estan distribuïts dins de Wattwin.

2.2 Entorns de Wattwin

Wattwin està dividit en tres entorns; integració, test i producció. Cada un d'aquests entorns compleix una funció diferent i per tant haurà de ser configurat de forma adient. L'entorn d'integració és l'entorn on treballen els enginyers directament, on es fan les proves més volàtils i on mai hi hauria d'entrar un client. En l'entorn de test, com el seu nom indica, és on es validen noves funcionalitats abans de passar-les a l'entorn de producció, que és el conjunt de màquines que mantenen l'aplicació web real que utilitzen els clients de Wattwin.

L'únic entorn que està sempre en funcionament és el de producció i també és l'únic que disposa dels serveis replicats per a que en cas d'una error inesperat es pugui seguir servint als clients.

En el moment en que es va començar a fer aquest projecte es va crear un quart entorn anomenat Development per poder fer proves de sistemes sense molestar a la resta de treballadors de l'empresa. Aquest entorn no es manté molt de temps en funcionament ja que es fa servir de forma esporàdica per a poder fer els tests dels canvis que més podrien afectar als altres entorns.

Ara que ja s'ha descrit l'estat actual de l'empresa es passa a descriure la realització del treball en les següents seccions.

3 ANÀLISI DEL PROJECTE

En aquest apartat es descriuen els requisits generals del projecte dividits en funcionals i no funcionals.

Un cop es tenen els objectius i l'estat actual del sistema ha sigut possible especificar els requisits del projecte. Aquests requisits són fruit de l'estudi inicial de l'estat de la infraestructura, la posterior creació de la documentació i de varies reunions amb el tutor de l'empresa que en el context del projecte actua com a client. A continuació es mostren els requisits globals separats en funcionals i no funcionals.

Funcionals:

- Sistema d'alarmes que notifiqui a l'empresa en cas de que alguna mètrica compleixi una condició específica.
- Generació d'informació útil per a l'empresa a través de Logs en els diferents entorns dels que disposa.
- Generar un script que permeti a l'empresa la inicialització de diferents entorns de forma dinàmica.

No Funcionals

- Separació de les bases de dades per a una millora en la fiabilitat, seguretat i eficiència dels sistemes
- Crear un entorn de proves aïllat del sistema.
- Generar una documentació adient tant a les millores realitzades com a la infraestructura ja existent.
- Reestructuració de la xarxa de l'empresa per tal d'evitar IP's públiques en màquines internes, ports oberts innecessaris i millorar la fiabilitat dels sistemes ubicant les màquines en diferents zones geogràfiques.
- Crear imatges de backup per tenir opció a recuperar-se d'un error en cas de que falli alguna configuració del sistema.

Un cop definits els requisits es va començar amb el desenvolupament dels diferents mòduls que constitueixen el projecte en si.

4 DESENVOLUPAMENT DELS MÒDULS

En aquest apartat es fa un recull del nucli del projecte, el desenvolupament dels diferents mòduls que el conformen. Es comença descrivint la realització dels dos objectius de menor importància i pes (la migració de la base de dades i la creació d'una imatge de backup), després es

mostra de forma més extensiva el nucli del projecte (els logs, les alarmes i la xarxa virtual) i s'acaba amb el script de CloudFormation ja que acaba sent el resultat final que engloba els anteriors.

4.1 Migració de la base de dades

Un dels objectius inicials del projecte era el de la separació de cada una de les bases de dades de cada entorn. Amb aquesta separació es busca poder millorar la seguretat, l'estabilitat, la gestió de permisos i poder oferir la possibilitat de ser capaços de testejar coses noves sense comprometre l'estat de l'entorn de producció.

Per tal d'aconseguir aquests objectiu es va utilitzar el servei d'amazon Relational Database Service (RDS). Tot i que la distribució perfecte seria la de poder tenir un RDS per a cada entorn de l'empresa el cost de mantenir tres bases de dades era massa elevat. Per sort, al tractar-se d'una empresa petita, es va veure que el nombre de connexions simultànies i el tant per cent d'ús de la CPU de les instàncies estava molt per sota del que realment poden aguantar. Així que no hi havia cap problema per compartir una instància entre dos entorns. Es va optar per aïllar l'entorn d'integració ja que es tracta del més volàtil dels tres i d'aquesta manera les condicions de test i producció es mantenien similars entre elles, un fet que beneficia a l'empresa ja que sempre es busca la màxima similutid entre l'entorn de test al de producció..

Utilitzant RDS servei es va crear una nova instancia de base de dades que tenia les mateixes característiques que l'original. Un cop creada, es va decidir que en la instancia original (*int-rds*) només romandria la base de dades de l'entorn d'integració, mentre que en la nova instància (*prod-rds*) s'implementarien les dades dels entorns de test i producció. Degut a que l'entorn d'integració és el més inestable es va optar per separar aquest i gràcies a això

Amb la rèplica ja creada es va migrar en primera instància les dades de l'entorn de test per assegurar el correcte funcionament de la migració. Un cop realitzada aquesta migració es va dur a terme la migració de l'entorn de producció. Com que aquesta migració afectava a la disponibilitat de la pàgina web es va dur a terme durant la nit per reduir l'impacte en la usabilitat dels clients.

Gracies a la migració de les instàncies es va veure que el certificat SSL que utilitzaven s'estava apunt d'exaurir, així que es va aprofitar per canviar-lo per un de nou. Amb aquests canvis es van assolir els requisits marcats en l'anàlisi del projecte.

4.2 Imatges de backup

Durant el desenvolupament d'aquest treball l'empresa es va veure amb la necessitat de disposar d'un servidor proxy per poder debugar una comunicació que fallava entre el servei on es troba emmagatzemada l'aplicació web S3 i el servei de cache d'amazon CloudFront.

La informació i els logs proporcionats pels serveis d'amazon no van resultar suficients com per poder solucionar el problema, així que es va optar per replicar el que feien els serveis de amazon per poder estudiar les traces de comunicació entre aquests. Es volia replicar el servei de CloudFront imitant el seu comportament de caché, per així poder veure com accedia al sistema d'emmegatzematge S3.

Per seguir amb la metodologia establerta, primer es va crear un servidor de prova que es connectava amb l'entorn d'integració. Aquesta instància proxy contenia un nginx que redirigia el tràfic cap a la pàgina de test de S3, i per reproduir les condicions que presenta CloudFront es van afegir opcions de cache i de https amb els mateixos certificats SSL utilitzats en els altres entorns. Aquest proxy es va implementar en una màquina del servei EC2 per tal de poder gestionar-la fàcilment i es va configurar un servidor nginx amb els paràmetres adients.

Gràcies al servidor Proxy es va poder determinar que el problema residia en el servei de CloudFront, ja que si accedies al servei de S3 a través del nou Proxy l'aplicació no presentava cap error. Es va reconfigurar el servei de CloudFront fins a arreglar el problema, que al final es va veure que procedia de les capçaleres que s'afegien al passar per aquest servei de cache.

Un cop es va veure que tot funcionava correctament es va procedir a crear una imatge de la nova màquina. Al tenir una imatge es va poder replicar la màquina per l'entorn de producció, però el benefici més important és el de que al disposar d'una imatge es pot aconseguir aixecar un servidor personalitzat en qüestió de pocs minuts.

D'aquesta manera es disposa d'un servei de backup en cas de que es torni a trobar un problema amb el servei de CloudFront.

4.3 Reconfiguració dels logs

En aquesta secció es descriu el primer dels objectius principals del projecte, permetre poder obtenir informació de forma senzilla de l'estat de les màquines que formen la infraestructura de l'empresa. Per aconseguir satisfer aquest objectiu s'ha utilitzat el servei d'amazon de Cloudwatch, que permet rebre corrents de logs i visualitzar-los de forma còmode.

Per tal de determinar quins paràmetres es volien monitoritzar es van fer una sèrie de reunions tant amb el tutor del projecte en l'empresa com amb els enginyers que treballen allà. Aquestes reunions van servir per detectar que era el que realment necessitaven veure els enginyers, que al cap i a la fi, són les persones que acabaran utilitzant els logs. Un cop finalitzades les reunions es va veure que es volien logs dels accessos al nginx de les diferents instàncies, els errors de node que es llençaven des del codi de back-end, la pròpia consola de node i les peticions i respostes generades en el CloudFront. Com a últims tipus de logs es van afegir dos grups per ajudar a gestionar els sitemes: el AWS.log, un corrent de logs que informa de com s'estan gestionant la resta de corrents de logs i el eb-docker, que dona informació relacionada amb el funcionament de les aplicacions que es despleguen mitjançant docker (Camunda i Deepstream).

Per assolir els requisits s'ha fet ús de tres eines d'AWS, Lambda, CloudFormation i les ebExtensions.

EC2 permet una personalització de les instàncies addi-

cional en forma de fitxers de configuració anomenats ebExtensions. Amb aquests fitxers es poden personalitzar les màquines d'EC2 de mil maneres diferents i en aquest mòdul del treball s'han utilitzat per especificar quina informació havia d'enviar cada màquina i a on. Els fitxers ebExtensions es troben en el codi back-end de l'aplicació, el que implica que cada vegada que es faci un desplegament de codi nou la configuració s'actualitzarà de forma adient. Aquests fitxers de configuració s'han creat de forma que tenen un comportament dinàmic, és a dir, que el mateix fitxer pot servir per a diferents màquines (com podrien ser la API o la Batch) i ell mateix s'adapta a cada situació utilitzant les variables d'entorn. Aquestes variables, que es troben en cada una de les instàncies, venen definides per la plantilla de CloudFormation utilitzada en el moment de la seva creació, així que si mai es realitza una modificació en les màquines, les ebExtensions es podran adaptar de forma automàtica, aconseguint així que la configuració de logs segueixi el principi d'excel·lència operacional mencionat a l'inici de l'article.

També es van utilitzar els fitxers ebextensions per reconfigurar el servidor nginx de totes les màquines amb l'objectiu de poder afegir informació sobre el estat de la màquina o per afegir capçaleres a les respostes http (com podria ser el temps de resposta). Gràcies a aquest enriquiment després es poden realitzar consultes més complexes per aconseguir, per exemple, un llistat de les crides que tinguessin un temps de resposta més alt.

És tant important especificar quines dades s'envien com a on s'envien aquestes dades. CloudWatch permet agrupar els logs que li arriben en LogGroups i subdividir cada un d'aquests grups en LogStreams. Utilitzant ebExtensions cada instancia envia els logs que genera a un grup diferent que va diferenciat per quin tipus d'instancia és (API, Batch, Camunda o Deepstream) i per a quin entorn de SIIE pertany (integració, test o producció). Com que un entorn pot tenir més d'una instancia en cada LogGroup es divideix en un LogStream per a cada instancia utilitzant la identificació única de la que disposa cada una d'elles.

S'ha utilitzat CloudFormation per dos objectius; configurar els permisos de les màquines i per la generació dels grups de logs de forma independent.

Per tal de poder enviar logs cap a Cloudwatch s'han de configurar una sèrie de paràmetres en les màquines EC2 de l'empresa. Primer de tot, la màquina necessita permisos per a poder escriure aquests logs en aquest nou servei. Es va optar per crear un perfil nou d'instància que unifiqués tots els permisos que les màquines de EC2 podrien compartir per tal de mantenir una cohesió jeràrquica en l'arquitectura. D'aquesta manera es podrà gestionar el conjunt de màquines de forma eficient i senzilla, ja que s'obté una visió més grupal que no la que hi havia implementada fins el moment, que requeria anar configurant cada una d'elles de forma manual.

La segona funció que ha complert CloudFormation ha sigut la creació dels LogGroups de CloudWatch on acabarà tota la informació necessària per monitoritzar els sistemes. Aquests LogGroups han de coincidir amb els que s'especifiquen des de les ebExtensions que s'han comentat anteriorment.

En la figura 1 es pot veure un diagrama de seqüència que il·lustra com des de CloudFormation es generen els logGroups i després cada vegada que es realitzi una pujada de codi a les màquines de EC2 les instàncies es configuren per enviar la informació necessària al LogGroup adient.

Per últim falta monitoritzar el servei de CloudFront. Al ser un servei extern no es poden utilitzar les ebExtensions per configurar-ne la interacció amb CloudWatch, així que es va utilitzar el servei Lambda com a solució. CloudFront genera logs de tota la seva activitat, però els envia directament a S3 sense parametritzar-los ni ordenar-los de cap manera per tant en el moment de la veritat no resulten molt útils. Devant d'aquesta problemàtica es va implementar una funció Lambda que s'activés cada vegada que CloudFront volgués enviar dades a S3 i redirigís aquest tràfic cap a CloudWatch. Redirigir la informació no era suficient perquè seguia sense estar formatejada, així que la funció també parametritza les traces per una visualització molt més còmode. Gràcies a aquests canvis ara es poden aplicar consultes a CloudFront com si es tractés d'un altre servei qualsevol.

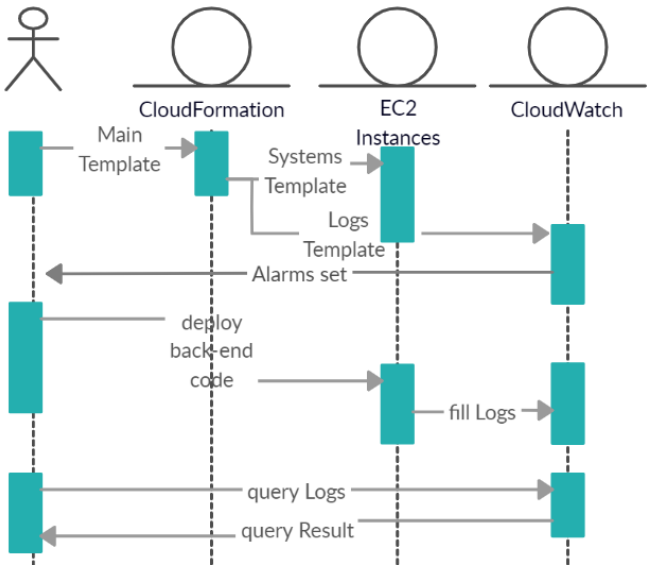


Fig 1: Diagrama de seqüència de la relació per la configuració i posterior consulta de logs i alarmes

Altra informació que s'ha recol·lectat per a l'empresa ha sigut qüestions relacionades amb l'estat de les màquines, com podrien ser la memòria de la que disposen o el tant per cent de CPU que estan utilitzant. Amb aquesta informació es pot treballar de forma molt més eficient, localitzar problemes amb molt menys temps i fer avaluacions de en quin estat general es troba la infraestructura i el codi de l'empresa

4.4 Implementació d'un sistema d'alarmes

Amb els nous logs l'eina Cloudwatch ens permet la creació d'alarmes. Aquestes seran alarmes que detectaran

errors web o una utilització per sobre de l'esperat de la CPU.

Les alarmes no relacionades amb errors web s'han pogut crear amb una combinació CloudFormation i ElasticBeanstalk sense configuració addicional. L'alarma més important és la que en funció del tant per cent d'utilització de la CPU de cada màquina es crea una copia per reduir la càrrega de treball o s'apaguen per tal d'estalviar costos a l'empresa. Per a aquestes últimes no han calgut logs nous i ha sigut suficient utilitzar una propietat dels entorns d'ElasticBeanstalk que permet la creació de autoScalingGroups que gestionen aquestes accions de replicar màquines o apagar-les. Gràcies a aquesta millora els sistemes són capaços de reaccionar per si mateixos devant d'una alta quantitat de peticions o es poden recuperar si alguna de les màquines no funciona correctament.

Per les alarmes relacionades amb els errors web primer s'ha de mesurar d'alguna manera un grup de logs per poder convertir un corrent d'informació en quelcom quantificable i mesurable. Utilitzant l'eina de Cloudformation i fitxers de configuració ebextensions s'han creat les mètriques que utilitzen els grups de logs creats anteriorment, específicament el de les peticions que es fan al nginx de la API. S'han creat dues alarmes, una que quantifiqui les peticions en les que el status_code comença amb un quatre i una que quantifiqui les que comença amb un cinc. Si es conten més de dos unitats en menys de dos minuts l'alarma salta i es genera una acció. De moment aquesta acció és la de notificar via correu electrònic al treballador de l'empresa que es desitgi. Igual que en el cas anterior de la creació de logs totes les alarmes es generen de forma dinàmica en funció de l'instància o entorn en la que es trobin associades.

A l'estar estretament relacionats els logs i les alarmes s'ha optat per a que comparteixin una plantilla de CloudFormation per tenir tota la gestió de la monitorització separada de la resta de sistemes. En la figura 1 es pot veure que en el diagrama de seqüència el "Logs Temlate" també genera les alarmes de CloudWatch.

4.5 Reestructuració de la xarxa virtual

En aquesta secció es descriu com es va dur a terme l'últim gran mòdul del projecte, la reestructuració de la xarxa virtual de l'empresa. Aquest mòdul neix principalment devant de la necessitat d'evitar que totes les màquines tinguin una IP pública, ja que a part de ser una mala praxis, això provocava que constantment es patissin atacs que intentaven buscar algún forat en la seguretat de la pàgina web. Aquests atacs mai aconseguien accedir a informació confidencial però si que provocaven que els servidors caiessin davant del gran número de peticions que es rebien en poc tems (com si es tractés t'un atac de denegació de servei). Els atacs no anaven dirigits directament a Wattwin, sinó que al tenir totes les màquines amb IP públiques els atacs indiscriminats que es fan per internet entre grans rangs de IPs acababen enganxant a Wattwin com a víctima de forma colateral.

El primer que es va fer va ser realitzar un estudi de l'estat actual de la xarxa, per tal de poder veure les mancances d'aquesta i que es podia millorar. L'apèndix A3 és una representació visual de l'estat en el que es trobava la xarxa. A part del conegut problema sobre les IPs públiques es van descobrir també els següents aspectes a millorar:

- Tot i tenir només 17 instàncies de màquines existeixen 23 subxarxes a la xarxa.
- Els entorns no estan separats.
- Balancejadors de càrrega no actualitzats.
- Serveis antics que ja no s'utilitzen.
- La creació de la infraestructura no s'ha realitzat utilitzant CloudFormation, per tant, qualsevol canvi en la xarxa s'ha de fer de forma manual.
- Més d'una Virtual Private Cloud (VPC) que no s'utilitza.
- 48 grups de seguretat

En el apèndix A4 es pot veure el disseny final de la VPC de SIIE. En aquest disseny es pot apreciar que el nombre de subxarxes s'ha reduït dràsticament, passant de 23 a 11. Aquestes noves subxarxes divideixen cada un dels entorns d'integració, test i producció i alhora divideixen cada un d'aquests entorns en la seva subxarxa pública i privada. Com que la base de dades és comuna entre test i producció i el servei d'ElasticSearch és comú pels tres entorns es van crear dues subxarxes (amb les seves corresponents rèpliques en altres zones geogràfiques) separades i que són comunes per els tres entorns. Un cop es va tenir l'estructura base de la xarxa clara es va començar a decidir on es residiria cada una de les instàncies.

Després d'un estudi es va determinar que només la API i el Deepstream necessiten una comunicació bidireccional amb Internet. Tota la resta de màquines només utilitzaven la IP pública per a poder descarregar software necessari per l'aplicació, com podria ser la instal·lació de llibreries externes o les diferents actualitzacions del programari utilitzat.

Per tal de solucionar la problemàtica de les IPs públiques per les màquines internes es va optar per implementar una porta d'accés NAT. AWS ofereix una instancia especialitzada que s'encarrega de la traducció de IPs de les màquines que vulguin sortir de la xarxa virtual a buscar el software necessari per funcionar. Amb aquesta opció s'aconsegueix que les màquines internes dels tres entorns (integració, test i producció) puguin accedir a Internet però que des d'Internet no es pugui accedir a elles. En el fet d'implementar una NAT implica que tot el tràfic que vulguin fer les instàncies privades passarà per la NAT, encara que aquest tràfic vagi dirigit a la base de dades de mongo externa de l'empresa o al sistema d'emmagatzematge S3 propi de SIIE. Això no seria un

problema si no fos perquè el servei de NAT de AWS cobra pel tràfic que processa sense distingir el destí final. Com a solució s'han creat un endpoints per accedir a S3 i un VPC Peering (interconnexió entre diferents VPC) per accedir a la xarxa on resideix la base de dades de mongo. Un cop s'han creat aquests accessos s'ha editat la taula d'encaminament del router intern per dirigir el tràfic amb destí a la base de dades de mongo i a S3 directament a través dels endpoints, sense haver de passar per la NAT i haver de pagar pel tràfic ocasionat. Després de revisar el tràfic generat per les instàncies privades s'ha vist que amb una instància NAT ja era suficient per a tota l'arquitectura de Wattwin i, donat que el entorn de producció és el que està configurat amb repliques per oferir una millor resiliència davant d'errors, s'ha decidit que la instància de NAT residirà en la subxarxa pública de producció.

Com s'ha dit anteriorment la API necessita una IP pública per a poder connectar-se amb els diferents navegadors web que accedeixen a l'aplicació Wattwin, però tenir totes les instàncies de forma pública comporta que les màquines són més vulnerables a diferents tipus d'atacs. Com a solució s'ha optat per a fer que les instàncies de la API siguin privades i que sigui el balencejador de càrrega que distribueix el tràfic entre aquestes l'element que tingui una IP pública. Per tant, aquest balancejador de càrrega es troba en la subxarxa pública de cada un dels entorns, en aquesta subxarxa es troba també la màquina de Deepstream. Cada entorn de l'empresa disposa només d'una instancia de Deepstream, així que l'opció d'aplicar la solució del balancejador en aquest cas concret no tenia molt de sentit.

Al revisar el comportament dels balancejadors es va veure que les instàncies de Camunda utilitzen un Elastic Load Balancer (ELB) en comptes de un Aplication Load Balancer (ALB). Els ALB són més novedosos i més complexes i permeten balancejar el tràfic a nivell d'aplicació, però per les necessitats de l'empresa es pensava que amb els ELB ja era suficient i no s'havia plantejat passar a ALB. Això va canviar gràcies als logs dels que ara disposava l'empresa on es va poder veure que Camunda funcionaria millor amb Http/2, un protocol que només suporten els ALB. L'únic problema que es podria oposar a aquesta millora era que el cost dels ALB fos molt més elevat que el dels ELB, però després de revisar el tràfic que generaven les diferents instàncies de Camunda es va poder calcular el preu total de la millora i, per sorpresa de tots els implicats, eren més barats els ALB que els ELB. Això és degut a que el que cobra el ALB per tràfic és menys que el que cobrava el ELB per hora d'ús. Així doncs es va passar d'utilitzar els antics ELB pels nous ALB millorant el rendiment de Camunda.

Un dels altres aspectes que s'ha aprofitat per millorar en la reestructuració de la xarxa virtual ha sigut el dels grups de seguretat. Dins d'AWS, un grup de seguretat es defineix per una sèrie de normes que indiquen quin rang d'adreces IP poden comunicar-se amb les diferents instàncies i a través de quins ports. Resulta evident que 48 grups de seguretat són masses grups per un total de 16 instàncies de les que disposa l'empresa. Al revisar els grups de seguretat s'ha vist que la majoria d'ells eren autogenerats per ElasticBeanstalk i estaven repetits o en desús. Al tenir una bona organització de les subxarxes ara es pot utilitzar una altra característica de la VPC de AWS, les Access Control Lists (ACL). Les ACLS són molt similars als grups de seguretat ja que s'utilitzen per controlar l'accés a les instàncies mitjançant rangs de IPs i ports però en comptes d'estar associades a una instància s'associen a una subxarxa sencera. D'aquesta manera es pot assignar una ACL a les subxarxes privades dels tres entorns i a les dels serveis interns comuns que previngui tot el tràfic que no vingui de dins de la VPC, i assignar una altra ACL a les subxarxes públiques per a que el permetin. En el cas de que sorgeixi la necessitat de personalitzar l'accés a alguna de les instàncies es podrà tornar a utilitzar els grups de seguretat per poder-ho resoldre.

Degut a que és una migració molt important i a que afecta a tots els sistemes de l'aplicació de moment només s'ha migrat el entorn d'integració. Després d'un període de prova es passarà a migrar l'entorn de test a la xarxa i, si no sorgeix cap problema finalment es migrarà el entorn de producció.

Hi ha mil maneres d'estructurar la VPC d'una empresa i apart d'aquest disseny se'n van explorar d'altres (com per exemple separar cada entorn en la seva pròpia VPC), però es va acabar optant per aquest perquè permet una millora important de l'estat actual sense haver d'invertir molt capital en AWS, ja que només la instància NAT suposarà un cost nou per a l'empresa.

4.6 Script de CloudFormation

Un dels objectius és el de crear un script de Cloudformation que es comporti d'una manera molt més dinàmica que del que es disposa actualment. Cloudformation genera diferents instàncies de serveis en funció d'una plantilla que especifiquès com a usuari. En l'inici d'aquest projecte aquesta plantilla era d'un sol fitxer en el que s'indicava la infraestructura de l'empresa sencera.

Després d'un estudi i un disseny inicial s'ha acabat optant per crear un script que agrupi a diverses plantilles. Una plantilla pels entorns que ja existien però actualitzats, una plantilla per totes les qüestions relacionades amb els logs i alarmes i una última plantilla dedicada solament a la xarxa de l'empresa.

Amb aquest plantejament és guanya molta flexibilitat en el moment de recrear un entorn o sistema, ja que al ser mòduls independents no és necessari esborrar l'arquitectura sencera quan es vulgui fer un canvi en només una de les parts.

A la figura 2 es mostra, mitjançant un diagrama de seqüència, com es relacionen les diferents plantilles. Cal notar que la plantilla de sistemes (Systems Template) necessita una sèrie de valors que li proporciona la plantilla de la xarxa virual (VPC Template), aquests valors són les diferents identificacions de xarxa i subxarxa, els diferents grups de seguretat, els ACL, els endpoints i tot el necessari per a que ElasticBeanstalk pugui crear les instàncies dins de la xarxa de la forma en que s'ha especificat anteriorment.

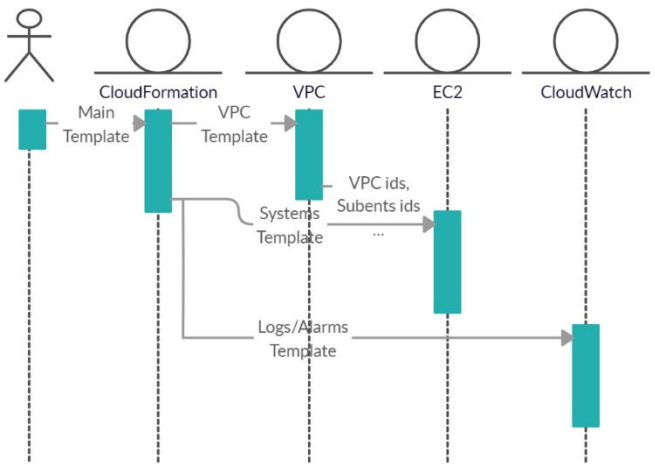


Fig 2: Diagrama de seqüència de la relació entre els diferents scipts de CloudFormation.

5 RESULTATS

En aquest apartat es mostraran els resultats dels diferents mòduls del projecte. Deguda a la naturalesa del treball els resultats obtinguts són més aviat de caràcter qualitatiu que quantitatiu.

El requisit del projecte referents als logs era que es generés informació útil per a l'empresa a través de Logs en els diferents entorns dels que disposa.

Els nous LogGrous de CloudWatch permeten veure els errors generats en les màquines, traces d'accés o d'error web, gestió interna de la pròpia màquina, comandes del docker si la màquina es de deepstream o de camunda i fins i tot, logs de com s'estan gestionant els logs. Gràcies a que la informació que ens arriba als logs és exactament la que volem també podem realitzar operacions com la que es mostra en la figura 3, on s'ordenen les peticions que es fan a la nostre API per ordre de temps de resposta. Aquesta informació pot resultar molt útil a l'empresa per a poder conèixer quins punts són els que s'han de millorar i amb quin nivell d'importància.

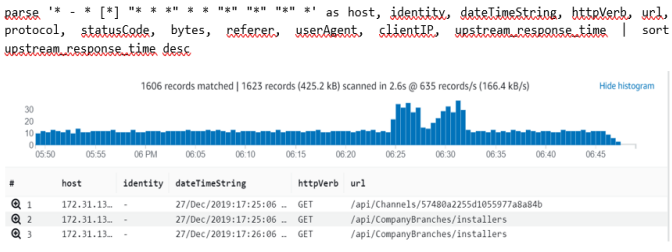


Fig 3: Exemple de consulta a CloudWatch.

Gràcies als nous logs el temps que s'inverteix en detectar on es troba qualsevol problema que tingui l'aplicació s'ha reduït dràsticament. Abans del projecte només es podia accedir a un document de text dels últims 100 logs de les instàncies i si es volien més de 100 s'havia d'anar a S3 a buscar el fitxer original que era d'un tamany que el convertia en impossible d'utilitzar eficientment.

El requisit del projecte referent a alarmes especificava que s'havia de muntar un sistema d'alarmes que notifiqui a l'empresa en cas de que alguna mètrica compleixi una condició específica.

S'han creat dues alarmes de control d'errors web i quatre de control de l'ús de la CPU. Abans de la creació d'aquestes alarmes la majoria de vegades els clients de Wattwin s'asebentaven abans que l'empresa de que la plataforma es trobava fora de servei. Gràcies a les alarmes ara es pot respondre de manera instantània devant d'errors web i de forma automàtica devant d'errors de CPU. En la figura 4 es pot veure un exemple de l'alarma d'errors web que es troba en estat d'alarma el que significa que acaba d'enviar un correu per notificar de la incidència al responsable de sistemes de l'empresa. Gràcies a aquesta millora el responsable pot resoldre el problema abans de que afecti a la funcionalitat de l'aplicació web.

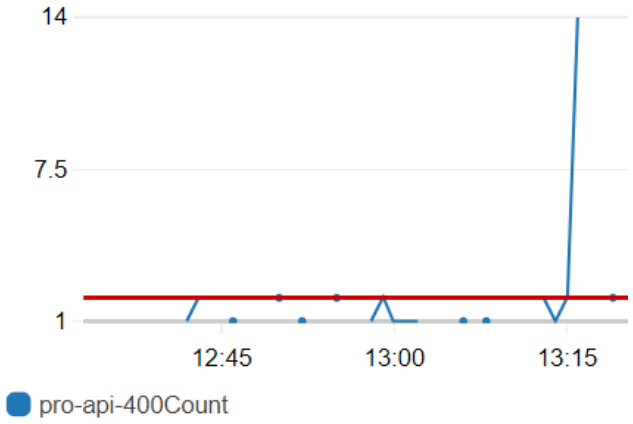


Fig 4: Alarma d'errors 400 originats en la API

En la figura 5 es mostra un exemple de l'alarma de utilització de CPU que es troba en estat de OK. Si en algun moment l'ús de la CPU sobrepassa el llindar del 60% se n'aixecarà una rèplica per tal de compensar la sobrecàrrega de feina. Gràcies a aquesta millora els sistemes reaccionen de forma automàtica devant de les necessitats de l'aplicació Wattwin.

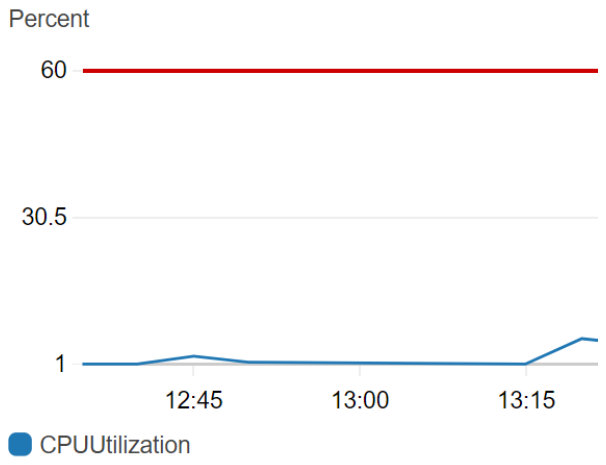


Fig 5: Alarma de la utilització de la CPU de la API

El disseny final de la VPC de l'empresa es pot apreciar en el apèndix A4. En el moment de la finalització d'aquest treball la migració de l'entorn d'integració ja s'ha dut a terme i ha donat els resultats esperats. Els beneficis que més destaquen són el fet de que al tenir totes les màquines en privat el filtratge i gestió de tràfic es pot centralitzar en el ALB de la part de front de la infraestructura i el fet de que ara tots els entorns están separats per diferents subxarxes. Amb la VPC final s'han eliminat moltissims elements que no s'utilitzaven i s'han actualitzat els que sí, aconseguint que sigui molt més eficient de gestionar i mantenir. La nova VPC també facilita que es pugui filtrar l'accés a cada una de les màquines utilitzant la subxarxes a la que pertany.

La migració de la base de dades i la creació de la imatge de backup contribueixen en aportar una millor funcionalitat en l'infraestructura final. Amb la nova base de dades es poden separar els permisos d'accès, no hi ha risc de corrompre les dades reals de producció i ara es possible fer proves més volàtils sense posar en risc l'estabilitat de l'aplicació que utilitzen els clients. Amb la creació de la imatge de backup es disposa d'una forma ràpida i eficient de replicar el servei de CloudFront per tal de poder debugar amb profunditat aquest servei i, en cas de necessitat, poder aixecar una màquina que en compleixi la seva funcionalitat.

Gràcies als nous scripts de CloudFormation els logs, les alarmes, la xarxa i els propis sistemes es poden gestionar de forma automàtica. Aquesta millora facilita enormement la feina que requereix mantenir els entorns, els torna replicables i ofereix la possibilitat de canviar la configuració d'un sense interrompre els altres.

6 Conclusió

En aquest apartat es mostren els objectius assolits, un petit resum dels resultats i línies futures del projecte.

La gran majoria d'objectius s'han assolit de forma satisfactòria. L'empresa compte ara amb un sistema de logs per poder veure que passa en les seves màquines, un sistema d'alarmes que fa diverses accions automàtiques davant de problemes imprevistos, les bases de dades s'han separat per millorar-ne l'estat, es disposa d'imatges preparades en cas de que torni ha haver-hi problemes amb CloudFront o altres serveis, la xarxa virtual ha sigut millorada i està en procés de ser migrada completament i es disposa d'una documentació que permet una visió més general de l'arquitectura de l'empresa. Amb totes aquestes millores s'ha aconseguit que la infraestructura sigui més estable, fiable, segura, eficient i que es gestioni d'una manera més automàtica. Com a línies futures del treball queda pendent la total migració dels elements del sistema a la nova xarxa vritual. També s'haurien de tenir en compte els objectius els quals van ser substituïts per la creació de la xarxa virtual; el monitoratge a temps real integrat amb l'aplicació Wattwin i la definició de grups de seguretat d'usuaris. El monitoratge podria permetre que els usuaris de l'aplicació poguessin veure a temps real l'estat d'aquesta sense haver d'accedir a AWS i la definició de grups podria resultar molt profitosa en cas de que l'empresa segueixi creixent en treballadors i es vulgui dividir la plantilla en departaments. Durant la realització del projecte també han aparegut noves idees que es podrien dur a terme com podria ser replicar la migració de la base del servei d'ElasticSearch o passar a fer tota la gestió de codi i de deploy amb el servei d'amazon CodeDeploy.

Agraïments

Agraïr a l'empresa SIIE per donar-me l'oportunitat de realitzar el meu treball de fi de grau amb ells i especialment al meu tutor en l'empresa Jordi Alborch per sempre estar disposat a donar-me un cop de mà quan era necessari. Important agraïr també als meus companys de la feina per l'ajuda que m'han proporcionat sense la qual no hagués sigut possible realitzar el treball.

Bibliografia

[1] Mathew, Sajee, and J. Varia. "Overview of amazon web services." Amazon Whitepapers (2014).

[2] Ben Piper, David Clinton, ed SYBEX. A"WS Certified Solutions Architect Study Guide", 2019, Second Edition.

[3] Qian, Ling, et al. "Cloud computing: An overview." IEEE International Conference on Cloud Computing. Springer, Berlin, Heidelberg, 2009.

[4] Murty, James. Programming amazon web services: S3, EC2, SQS, FPS, and SimpleDB. " O'Reilly Media, Inc.", 2008.

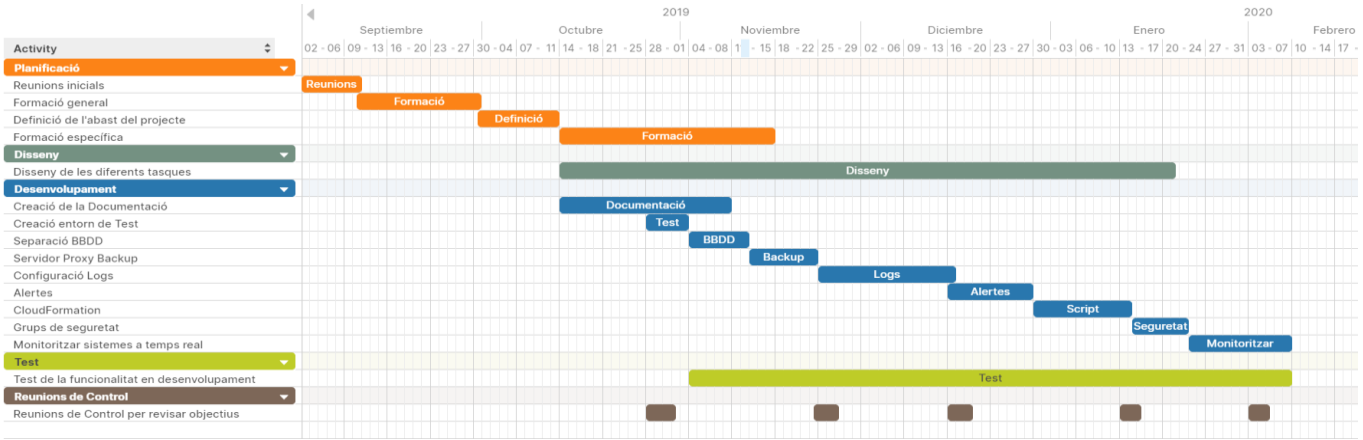
[5] Srinivasan, Latha, and Jem Treadwell. "An overview of service-oriented architecture, web services and grid computing." HP Software Global Business Unit 2 (2005): 1-13.

[6] "Documentació de Deepstream". [En línea] Disponible a: https://deepstream.io/docs/server/configuration/

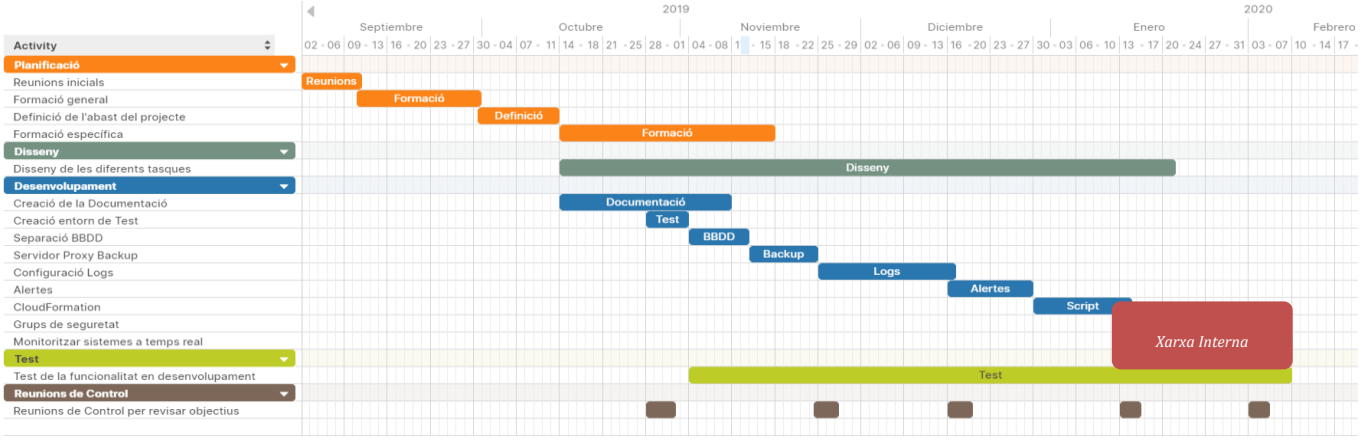
[7] "Documentació de Camunda". [En línea] Disponible a: https://docs.camunda.org/manual/7.11/?__hstc=252030934.92a5c92ea65345344f1c9cc1c70770d4.1573513408550.1573513408550.1573513408550.1&__hssc=252030934.2.1573513408550&__hsfp=405395720

APÈNDIX

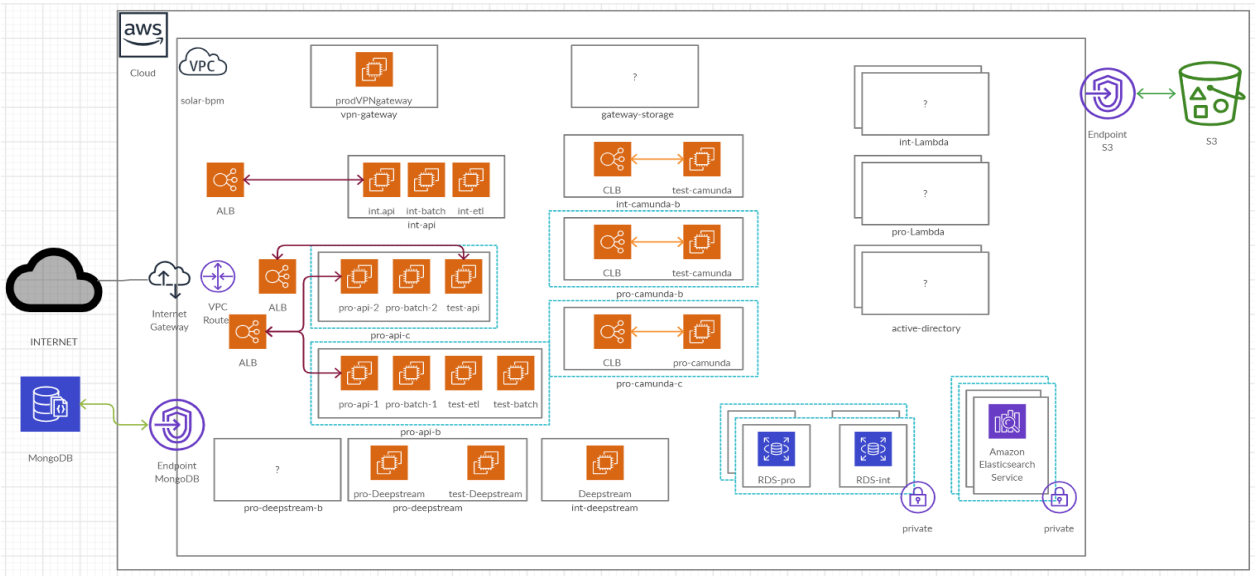
A1. PLANIFICACIÓ INICIAL DEL PROJECTE



A2. PLANIFICACIÓ FINAL DEL PROJECTE



A3 VPC INICIAL



A4 VPC FINAL