

Sprint 2

Administração de Sistemas

Turma 3DG _ Grupo 41

1190679 – Ivo Oliveira

1190384 – André Teixeira

1201199 – Miguel Macedo

1201496 – Mário Cardoso

1201197- João Figueiredo

Data: 02/12/2022

User Story 1

Para o deploy automático para a VM nos servidores do dei, começámos por clonar para a VM o repositório bitbucket que temos o projeto, incluído o módulo SPA que vai ser o módulo a qual vamos dar deploy.

```
sudo apt install git-all
```

```
git clone https://1201199@bitbucket.org/1201197JoaoMiranda/lapr5_41.git
```

Depois, fomos a pasta .ssh através do comando `cd .ssh` onde estão presentes os ficheiros `id_rsa`, e `id_rsa.pub`, que são as chaves SSH privada e pública, respetivamente.

```
authorized_keys id_rsa id_rsa.pub known_hosts
```

Acedi a esse ficheiro usando `cat xxx` e copiei o seu conteúdo (passes SSH) para a aba SSH Keys nas pipelines do nosso repositório no bitbucket.

Joao Pedro / LAPRS / LAPRS_41 / Repository settings

Public key

6zswGbxXaaysEG+NdlMa6l7TLfKuQrwb6c66itRyQ8imhtRGyOwVv1HmVUJvofcxuDME
P6sn9f8nXQgruVBEPc+DOQfrsITyQAAAAABAAEAAAGBAJcNO3CdCyRTp+qnSLHC3T600
6wCPri8xWvcZNWCNqJ146WKe4Y4MAmrygAagQZ+P+na7NDWsygoEzLvJlMel2rw4ebafG
8Rvpf2SjPlg8saqKftg52E44XA60+jkYaxkMpfofloodZbMBpwqwg/TZIKUgJvPPNjz
Tpfz2MFyVvE38ESDrpBuGjs4g6d/+0SWVHqfsO8ayfHlOrY6eYr6DXCQSYdyw2swOggld
8xr/OcNGvJtmcabSTV+TVqQ24TQsyID6AgcYOGH9POxy1yVZPABRyih3OmAJ/HVhffXol
hb4dGwAt6J2rXMa8026/H5uIKKcX8TckZURZClbsCUHG49gNVVpoUQdJcwlw9NZ+cbR7Z
1Yz9Wm+V6b7MUOWsd3rOmJfKcoSdinDKyebVNOAqRC5lwfi2+zsnQJNps0lv20Cx3lfqKz
Vx+oXLPaM9OrgsNyUkbMV91xjldhyQYyt2NJfP7T6vbWfVkJ0St9mQZ7rO1P0muVcRMQAA
AMACi8NjmoiME3hYJZitdAchWkXvBDIzzsdxsIR2YyuX3XJ/H3La4FOUKrCXHojo/G6lPs
PdeKPOAhX7N/850ZyhoBlX9vJUXGnjcJbeCEI1O6PgoZeG1YgGneU7B1+BlvERcl1EA2wq
LPYK7+Nry8y86whrS8AJDT1pfrgOT04tCvvsaeAFQc3f5gf1rxQdVvUln9V1iEPspzJde
LYWJfXwmk2f8l6W+cExkGOzNweeJTLz6LOgviChhGEUAJuD0AAADBANCSgmE2hxUkLMi5
gh00WFJ0BtljJcsdd5TEhQCa1DLpJC2S2E7jmC7KHi0X4qxtq7sfzjrpGXbKMfAm+WfK
Ming5jqXH1xXTJup4TJOINMiefWhpGWdLdf1Cj9yCMSttr7TOro7ARG/nyoomn40n2hm
F3bavvdNB8tqJfGxUVC99UjpfelNV1k9Sh8hJwVu+EP1aRMu9u9qO/8tbkCegU7fRpa461
+Cc4Awbulfm38dawampPgNmruclpQwAAAMEAxeerj6uKX21Rve/lz2QL3avxstWxw6bE
2u1QRK/J2ZvTouICReDhEle59hjtC0ZFwOEC80Ulip1A7sXK8goOlgMzrqaGVuxTKd
fR7e6Yt8Shb2zwxmteEyK2LXf+7pZGJs0YbivEe6H4YKKwhx3zjPF9oWdOv5eFZ1SU7O48
VEaxjsVhOyFwTFn8PmkPr9BMfzRzkO1JgRTxdeiTDUCDhYo+LGHdoce5fZ2AO+zdO3KAQ
I/XZNYB8HstHz7AAADAXJvb3RADnNob3N0MDQ8AgMEBQ==

Copy public key Delete key pair

Known hosts

Add a host address below, then click **Fetch** to see the host's fingerprint. Read about [known hosts](#) for Pipelines.

Host address

Click 'Fetch' to see the host's fingerprint

Fetch

Host address	Fingerprint
vsgate-ssh.dei.isep.ipp.pt:10189	31:9b:53:31:0c:66:a0:7c:07:5f:b6:1a:c8:3c:32:71

Em hostname meti o hostname da VM no servidor DEI e adicionei esse address à pipeline.

Criei por último a pipeline

USER STORY 2

O primeiro passo para utilizar as iptables é fazer a instalação do package iptables-persistent

através do comando:

```
$ sudo apt install iptables-persistent
```

Para podermos criar regras e guardar as mesmas regras da firewall precisamos de fazer o comando

```
netfilter-persistent save
```

Decidimos implementar loopback- interfaces referidas como (-lo) para permitir conexões dentro da própria máquina.

1. `sudo iptables -A INPUT -i lo -j ACCEPT`
- 2.
3. `sudo iptables -A OUTPUT -o lo -j ACCEPT`

Por exemplo se dermos o comando `ping local host` o servidor vai responder utilizando o loopback

```
root@vs189:~# ping localhost
PING localhost (localhost (:::1)) 56 data bytes
64 bytes from localhost (:::1): icmp_seq=1 ttl=64 time=0.031 ms
64 bytes from localhost (:::1): icmp_seq=2 ttl=64 time=0.041 ms
64 bytes from localhost (:::1): icmp_seq=3 ttl=64 time=0.040 ms
64 bytes from localhost (:::1): icmp_seq=4 ttl=64 time=0.040 ms
64 bytes from localhost (:::1): icmp_seq=5 ttl=64 time=0.042 ms
64 bytes from localhost (:::1): icmp_seq=6 ttl=64 time=0.039 ms
```

Como o nosso objetivo é apenas permitir conexões da rede interna do dei

Vamos permitir conexões ssh específicas de uma subnet, neste caso a fonte específica é a do DEI . Ou seja, neste caso queremos permitir todos os 10.8.0.1/16 da subnet correndo os seguintes comandos

```

root@vs189:~# iptables -A INPUT -p tcp -s 10.8.0.1/16 --dport 3000 -m conntrack --ctstate
NEW,ESTABLISHED -j ACCEPT
root@vs189:~# iptables -A OUTPUT -p tcp --sport 3000 -m conntrack --ctstate ESTABLISHED -j ACCEPT

```

Através destes comandos limitamos a rede para a subnet com a porta 3000

User Story 4

Como administrador quero identificar e quantificar os riscos envolvidos na solução preconizada.

Módulo	Localização	Serviços Dependentes	Prioridade
Módulo Armazéns	MariaDB Server – DEI Cloud	Módulo de Logística Planeamento	2
Módulo Logística	MongoDB Server – DEI Cloud	Planeamento	2
Planeamento	SWI-Prolog – DEI Cloud		3
Visualização		Todos	1

Avaliação de Riscos

De modo a avaliar os possíveis acontecimentos que possam pôr em causa o bom funcionamento da aplicação e de todos os componentes, desenvolvemos uma matriz de risco, com o objetivo de prever todas as possíveis futuras falhas de programa. A matriz relaciona a probabilidade de acontecimento com o impacto que resulta da falha.

	Muito baixo	Baixo	Moderado	Elevado	Muito elevado
Muito provável (90%)					Falha na Autenticação do User
Provável (70%)				(MDA e MDL) Falha da VM do DEI	(Visualização) Falha da VM do Dei
Ocasional (50%)					Falhas na Internet

Improvável (30%)				Ataques à VM do DEI	Leaks de Informação das Bases de Dados
Muito improvável (10%)					Falhas na Eletricidade

Matriz de Risco

Posteriormente, a esta avaliação, verificamos que existem alguns pontos a melhorar no que toca ao suporte da aplicação. De modo, a diminuir os riscos, deveria alojar-se os componentes da aplicação em serviços Cloud, em que o servidor contasse com o suporte de uma bateria UPS, que garantisse os serviços de emergência no caso de uma falha de energia. De maneira, a melhorar a proteção de dados, as mensagens entre os módulos de armazéns e de logística, com o módulo de visualização deveriam ser encriptadas para despistar a manipulação de dados. Para prevenir agentes maliciosos que se podem aproveitar do uso de credenciais roubadas e senhas fracas, deverão ser atribuídos tokens únicos e “two-step-verification”.

User story 5

Como administrador do sistema quero que seja definido o MBCO (Minimum Business Continuity Objective) a propor aos stakeholders.

Tendo em consideração as especificações do cliente que pretende que o sistema esteja operacional o máximo de tempo possível, sendo aceites pequenos períodos de indisponibilidade inferiores a 1 hora. Preferencialmente o sistema deve ser resiliente o suficiente para suportar funcionamento parcial (apenas alguns módulos estão disponíveis). E que se pretende apenas a definição do MBCO (Minimum Business Continuity Objective, Objetivo de continuidade de negócios mínimo) face à arquitetura idealizada neste momento.

Base de dados dos Armazéns - nível 0

Este servidor de base de dados contém todas as informações das entidades armazéns e entregas pertencentes ao modelo operacional da ElectricGo, logo este serviço não pode estar inoperacional, é um serviço crítico. O MTD (“Maximum Tolerable Downtime”, o tempo máximo de inoperacionalidade) terá de ser inexistente e o MTPD (“Maximum Tolerable Period of Disruption”, o tempo máximo de desempenho inferior aos requisitos) também.

Base de dados da Logística - nível 0

Este servidor contém todos os dados das entidades camiões, empacotamentos e percursos necessários para a o sistema ElectricGo, neste caso assim como no primeiro é um serviço critico. Assim sendo também apresenta um MTD de inexistente assim como o MTPD.

Modulo de Armazéns - nível 1

Esta modulo da aplicação serve para criar, listar, editar e apagar armazéns e entregas, logo o seu serviço não é critico ou tão critico como os anteriores, pois não existe a perda de dados, no entanto ele é necessário para o modulo de SPA (front-end), e para o modulo de logística poder registar percursos, pois ele é que conhece os armazéns para poderem ser criados os percursos.

Para este modulo sugerimos um MTD baixo de 10 a 15 min e um MTPD também baixo.

Modulo de Logística - nível 1

Este modulo da aplicação serve para criar, listar, editar e apagar camiões, empacotamentos e percursos, este serviço apresenta priorização, assim como o modulo anterior, não tão critica. Apesar de a informação necessária para a criação de percursos seja um serviço que necessite da aplicação do modulo de armazéns e entregas e da base de dados continua a ter grande prioridade, porque pode na mesma criar, listar, editar e eliminar os camiões.

Apesar de ter um serviço que necessita de outro (secundário) sugerimos um MTD baixo 10-15 minutos, mas o MTPD pode ser um pouco superior mais 5-10 min até ao modulo de Armazéns esteja completamente operacional para poder executar a parte da criação dos percursos.

Modulo de planeamento - nível 2

Este modulo serve para gerar todas as trajetórias possíveis e sequencias de armazéns onde deverão ser as entregas a realizar, assim ele tem dependência do modulo de logística (percursos). Por isso sugerimos para este modulo poder ter um MTD superior aos anteriores, 25-30 minutos e um MTPD 35 -40 minutos.

Modulo de SPA (front-end) – nível 3

Por fim o modulo de SPA, é modulo “front-end” que apresenta a interface com o utilizador, necessita dos restantes módulos e serviços operacionais para poderem estar no seu completo potencial, logo não é um serviço crítico por isso terá o MTD mais alto 45-50 minutos e um MTPD de 55-60 minutos.

Com estas sugestões de sequência e prioridades, MTD, MTPD conseguimos propor um MBCO com as especificações pretendidas pelo cliente, tendo em consideração os módulos já disponíveis e tempo de indisponibilidade abaixo da 1 hora.