RELATÓRIO ASIST

Sprint B – G031

24/11/2024

David Sousa – 1220784@isep.ipp.pt

Guilherme Ribeiro – 1220786@isep.ipp.pt

Tiago Carvalho – 1221124@isep.ipp.pt

Conteúdo

Divisão de Tarefas

Duas US por aluno:

- 6.4.1: 1220786;
- 6.4.2: 1220784;
- 6.4.3: 1221124;
- 6.4.4: 1220786;
- 6.4.5: 1220784;
- 6.4.8: 1221124;

US 6.4.1

As system administrator, I want the deployment of one of the RFP modules in a DEI VM to be systematic, validating it on a scheduled basis with the test plan.

Para a realização desta tarefa, foi criada uma máquina virtual, com o sistema operativo Linux na *cloud* do DEI.

O módulo do RFP escolhido pela equipa para ser *deployed* foi o módulo UI.

```
#! /bin/bash

# Paths
repo_path="/root/sarmg031/sem5pi-24-25-g031"
production_path="/root/production"
logs_file="/root/deployScripts/logs.txt"

#Go to repo dir

cd "$repo_path" || exit

log() { local timestamp=$(date + "[%d-%m-%Y %T] - ") echo
        "$timestamp $1" >> echo "$timestamp $1" >> "$logs_file"
}

exec >> "$logs_file" 2>&1

# Check changes

if [ "$(git fetch && git status -uno | grep 'Your branch is behind')" > /dev/null ]; then
    # Changes detected

        log "Changes detected, pulling repo."
        "$(git pull origin main)" > /dev/null
        log "Pull finished."

        ng build
        if [ $? -eq 0 ]; then

                # Tests passed, copying files to production
                log "Build successful, copying files."
                cp -r "$repo_path/" "$production_path"
                log "Files copied."

                APP_PID=$(pgrep -f "ng serve --host 0.0.0.0")
                if [ -n "$APP_PID" ]; then

                        log "Restarting app."
                        kill -9 "$APP_PID"
                fi
```

```
                if [ -n "$APP_PID" ]; then

                        log "Restarting app."
                        kill -9 "$APP_PID"
                fi
                cd
                cd "$production_path/Ui" || exit
                # npm install
                ng serve --host 0.0.0.0 &
                log "App running."
                exit 0
        else
                log "Build failed."
        fi
else
        log "No changes detected."
fi
```

Previamente, os diretórios e ficheiros correspondentes ao `repo_path`, `production_path` e `logs_file` foram criados usando `mkdir` (para os diretórios) e `nano` (para os ficheiros).

Neste código acontece o seguinte:

1. **Definição de Caminhos e Arquivo de *Logs***: O script define as variáveis para o diretório do repositório, o diretório de produção e o arquivo de *logs,* onde as informações serão registadas.
2. **Função *Log***: Cria uma função "*log()*" para registrar mensagens no arquivo de *logs* com especificação de data/hora.
3. **Verificação de Alterações no Repositório**: O script verifica se há alterações no repositório utilizando o comando `git fetch` seguido de `git status`.
4. ***Pull* de Atualizações**: Se alterações forem detetadas, o script faz um `git pull` para atualizar o repositório local.
5. ***Build* da Aplicação Angular**: Após o `git pull`, o *script* executa o comando `ng build` para construir a aplicação Angular.
6. **Cópia dos Arquivos para Produção**: Se o *build* for bem-sucedido, os arquivos gerados são copiados para o diretório de produção.

7. **Reinício da Aplicação**: O script verifica se o processo do servidor Angular (`ng serve`) está em execução e, se estiver, mata-o com `kill -9` para reiniciá-lo.
8. **Instalação de Dependências e Reinício**: O *script* então navega até o diretório de produção, instala as dependências com `npm install` e reinicia a aplicação com `ng serve`.
9. **Encerramento**: O script finaliza com uma mensagem indicando que a aplicação foi iniciada.

Para executar o script de forma automática, utilizamos o agendador de tarefas **cron**. Ao usar o comando `crontab -e`, podemos modificar o arquivo do *cron* e definir uma tarefa que executa o *deploy's script* a cada 5 minutos.

```
*/5 * * * * /root/deployScripts/deploy.sh
```

US 6.4.2

As system administrator, I only want clients on the DEI's internal network (wired or via VPN) to be able to access the solution

O objetivo desta tarefa era controlar e restringir o acesso à nossa máquina virtual Debian 11 Bullseye (base system) - All VNETs. Para atingi-lo, foi necessário criar e editar as **regras de firewall** da máquina virtual. Estas regras de firewall permitem ou bloqueiam o tráfego de rede para a máquina virtual.

Para identificar a **sequência de IPs** que poderão conectar a máquina virtual, é necessário conectar-se à VPN do DEI e verificar as informações exibidas pelo comando `ipconfig`.

```
Unknown adapter OpenVPN TAP-Windows6:

   Connection-specific DNS Suffix  . : dei.isep.ipp.pt
   Link-local IPv6 Address . . . . . : fe80::97f6:6cc3:b5dd:d85b%15
   IPv4 Address. . . . . . . . . . . : 10.8.214.121
   Subnet Mask . . . . . . . . . . . : 255.255.255.252
   Default Gateway . . . . . . . . . :
```

Concluímos que o **intervalo de IPs** pretendido é: **10.8.0.0/16**.

```
Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . : ISEPWLAN.isep.ipp.pt
   Link-local IPv6 Address . . . . . : fe80::f0ba:4f19:3feb:1747%18
   IPv4 Address. . . . . . . . . . . : 172.18.155.60
   Subnet Mask . . . . . . . . . . . : 255.255.248.0
   Default Gateway . . . . . . . . . : 172.18.152.1
```

Concluímos também que o intervalo de IPs "**172.18.144.0/21**" é o pretendido.

Necessitamos agora de verificar quais são as **regras de firewall** que existem através do comando `iptables -L`.

```
root@debian:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

O próximo passo é alterar as **regras de firewall**, através de ações **Accept** e **Drop**.

- **Ações Accept** - permite garantir tráfego;
- **Ações Drop -** bloqueia os acessos.

Para guardar estas ações devemos executar o comando `sudo /etc/correct-ips`, assim, o arquivo de texto, é guardado no diretório `/etc/iptables/`, com o nome **rules.v4** para regras **IPv4** ou **rules.v6** para regras **IPv6**.

```
root@debian:~# iptables -A INPUT -p tcp --dport 4200 -s 10.8.0.0/16 -j ACCEPT
root@debian:~# iptables -A INPUT -p tcp --dport 4200 -s 172.18.144.0/21 -j ACCEPT
root@debian:~# iptables -A INPUT -p tcp --dport 4200 -j DROP
```

A primeira regra é do tipo: `iptables -A INPUT -p tcp --dport 4200 -s 10.8.0.0/16 -j ACCEPT`. O parâmetro **-A** é usado para adicionar uma nova regra à tabela de regras **INPUT**. O parâmetro **INPUT** especifica a tabela de regras à qual a nova regra será adicionada. O parâmetro `-p` especifica o tipo de protocolo, neste caso será **TCP**. O parâmetro `--dport` especifica a porta de destino (*4200*). O parâmetro `-s` especifica a origem da conexão que será **10.8.0.0/16**. O parâmetro `-j` especifica a ação a ser tomada neste caso a ação **ACCEPT** aceita a conexão.

A segunda regra é do tipo: `iptables -A INPUT -p tcp --dport 4200 -j DROP`, que tem a função de bloquear todo o tráfego direcionado à porta **4200**. Esta é a porta utilizada pela aplicação, a qual estará em execução nesta máquina. O parâmetro `-A` é usado para adicionar uma nova

regra à tabela de regras **INPUT**. O parâmetro **INPUT** especifica a tabela de regras à qual a nova regra será adicionada. O `parâmetro -p` especifica o tipo de protocolo, neste caso será **TCP**. O parâmetro `--dport` especifica a porta de destino (**4200**). O parâmetro `-j` especifica a ação a ser tomada neste caso a ação **DROP** que descarta a conexão.

De seguida, podemos usar o comando `sudo sbin/iptables-save` e obter o seguinte:

```
root@debian:~# /sbin/iptables-save
# Generated by iptables-save v1.8.7 on Thu Nov 21 16:17:27 2024
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -s 10.8.0.0/16 -p tcp -m tcp --dport 4200 -j ACCEPT
-A INPUT -s 172.18.144.0/21 -p tcp -m tcp --dport 4200 -j ACCEPT
-A INPUT -p tcp -m tcp --dport 4200 -j DROP
COMMIT
# Completed on Thu Nov 21 16:17:27 2024
```

Agora voltamos a verificar quais **regras de firewall** existem através do comando `iptables -L`.

s

```
root@debian:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  10.8.0.0/16         anywhere             tcp dpt:4200
ACCEPT     tcp  --  172.18.144.0/21     anywhere             tcp dpt:4200
DROP       tcp  --  anywhere            anywhere             tcp dpt:4200

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

US 6.4.3

As system administrator, I want the clients listed in the requirement 6.3.2 to be able to be defined by simply changing a text file.

Para realizar esta User Story criamos um ficheiro `nano /etc/correct-ips` onde estarão os IPs dos utilizadores que pretendemos que tenham acesso ao nosso sistema.

Depois de ter os utilizadores definidos vamos criar um script que atualiza as iptables conforme os IPs no ficheiro correct-ips. Para criar o script fazemos `nano /update-iptables.sh`

```
#!/bin/bash

while IPS= read -r ip;
do
        iptables -A INPUT -p tcp --dport 4200 -s $ip/32 -j ACCEPT
        echo "$ip is now able to Connect to App"

done < correct-ips

/sbin/iptables-save
```

Ao executar este script vai acrescentar as iptables definidas nas US anteriores os IPs que estejam no ficheiro correct-ips.

`/etc/update-iptables.sh`

```
root@vs497:/etc# /etc/update-iptables.sh
10.9.10.31 is now able to Connect to App
# Generated by iptables-save v1.8.7 on Thu Nov 21 17:13:20 2024
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -s 10.8.0.0/16 -p tcp -m tcp --dport 4200 -j ACCEPT
-A INPUT -s 172.18.144.0/21 -p tcp -m tcp --dport 4200 -j ACCEPT
-A INPUT -p tcp -m tcp --dport 4200 -j DROP
-A INPUT -s 10.9.10.31/32 -p tcp -m tcp --dport 4200 -j ACCEPT
COMMIT
# Completed on Thu Nov 21 17:13:20 2024
```

Assim, garantimos que apenas os utilizadores especificados no ficheiro terão acesso à aplicação. É fundamental eliminar as duas regras configuradas anteriormente, pois estas concedem acesso a qualquer utilizador que esteja conectado à rede do DEI.

US 6.4.4

As an administrator, I want to identify and quantify the risks involved in the recommended solution.

Para ser possível analisar os riscos envolvidos na solução recomendada, foi feita uma matriz de risco para termos uma melhor análise dos dados.

		Impacto		
		1	2	3
Probabilidade	1 (0%-30%)	-	-Problemas de concorrência	- Problemas do lado do servidor do DEI -Ataques informáticos
	2 (31%-60%)	-Má utilização da solução por parte dos utilizadores	-	-Má implementação de código que provoque crash da aplicação
	3 (61%-90%)	-	-	-

Com base na análise na tabela, conclui-se que a alta dependência do servidor do DEI torna essencial um investimento significativo em medidas para garantir sua segurança e funcionamento. Além disso, aumentar o número de verificações na aplicação para identificar erros na implementação pode ser uma estratégia eficaz para reduzir o risco de consequências graves no uso da mesma.

US 6.4.5

As system administrator, I want to define the MBCO (Minimum Business Continuity Objective) to propose to stakeholders.

O MBCO (Minimum Business Continuity Objective) refere-se ao nível mínimo de serviços ou produtos que uma organização precisa garantir durante e após um desastre ou uma interrupção significativa nas suas operações, como o próprio nome sugere.

O objetivo da nossa organização é desenvolver uma solução personalizada para a gestão hospitalar. Nesse contexto, identificamos como serviços essenciais da aplicação:

- Planeamento e criação de rotas de entrega.
- Monitorização em tempo real dos dados dos robôs.
- Monitorização em tempo real dos dados dos edifícios.
- Planeamento e criação de rotas de entrega (reiterado como prioridade).

Em caso de um problema crítico na base de dados, como uma avaria, surgiria uma grande dificuldade, pois, sem os dados, seria inviável criar os melhores percursos. Para lidar com essa situação, assim que o problema for identificado, uma Equipa de Resposta a Incidentes (ERI), composta por quatro membros, será notificada por um sistema de alerta automático. Esse sistema enviará um e-mail a todos os integrantes da equipa, contendo informações detalhadas sobre a natureza da interrupção.

A equipa chamada irá avaliar os danos e implementar um plano de recuperação para recuperar os serviços essenciais o mais rapidamente possível. A recuperação irá incluir as seguintes ações:

- Pedidos de cirurgia
- Marcações de cirurgia
- Gestão de Recursos

Os dados de backup, gerados periodicamente pela base de dados, serão utilizados neste tipo de situação. Esses backups estão armazenados em duas bases de dados separadas, projetadas especificamente para

cenários de emergência. A equipa atuará em turnos de 4 horas para restaurar os serviços essenciais no menor tempo possível.

A organização realizará testes e exercícios de recuperação regularmente para assegurar a eficácia dos planos de recuperação. Esses testes vão avaliar a capacidade da ERI em notificar, avaliar e restaurar os serviços. Os exercícios irão simular interrupções em tempo real, permitindo analisar a eficiência do plano de recuperação implementado.

Funcionalidades secundárias, como o módulo de visualização 3D, que fornece uma representação gráfica do sistema, serão temporariamente descartadas, pois são consideradas de baixo impacto no funcionamento geral.

Durante o período de recuperação, a manutenção rotineira da aplicação será suspensa, com todos os esforços concentrados na restauração do pleno funcionamento do sistema.

US 6.4.8

As system administrator I want to get users with more than 3 incorrect accesses attempts

Para implementar esta User Story, é preciso, inicialmente, instalar um módulo que registre as autenticações dos acessos à máquina virtual. Isso pode ser feito utilizando o comando:

```
sudo apt -y install rsyslog
```

Depois que o módulo for instalado, um arquivo chamado `auth.log` será criado automaticamente no diretório `/var/log`. Esse arquivo será responsável por armazenar todas as tentativas de acesso feitas por utilizadores na máquina.

Para agora filtrar os resultados para apenas ficarem os utilizadores com mais do que três acessos incorretos, executa-se o comando:
`grep "Failed password" /var/log/auth.log | awk '{print $(NF-5)}' | sort | uniq -c | awk '$1 > 3 {print $2}'`
Agora os resultados vão aparecer no terminal.