Relatório ASIST SPRINT B

2 DE DEZEMBRO

1211089 – José Gouveia
1211128 – Tiago Oliveira
1211131 – Pedro Pereira
1211151 – Alexandre Geração

isep Instituto Superior de **Engenharia** do Porto

Índice

Conteúdo

Índice de imagens

Divisão de Tarefas

- User Story 650 e 800 – Pedro Pereira (1211131)
- User Story 670 e 830 – Alexandre Geração (1211151)
- User Story 660 e 810 – José Gouveia (1211089)
- User Story 640 e 820 – Tiago Oliveira (1211128)

User Story 640

Como administrador do sistema quero que o deployment de um dos módulos do RFP numa VM do DEI seja sistemático, validando de forma agendada com o plano de testes.

Para a realização desta User Story, 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 de visualização.

```
#! /bin/bash

# Paths
repo_path="/root/RobDroneGo/sem5pi_23_24_g056_Visualization"
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 $1" >> "$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

                cd
                cd "$production_path"/Visualization || 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
```

Figura 1 - Script em bash para deploy automático

Para que o deploy seja sistemático foi criado um script em bash que verifica se existem alterações no repositório do módulo, através do comando **git fetch origin main**

e de um bloco if. Se então existiram alterações o repositório local é atualizado através do comando **git pull origin main**. De seguida é feita a compilação do projeto e apenas se esta for bem-sucedida serão os ficheiros copiados para a pasta de produção. Por fim é feita a procura pelo PID da aplicação que está atualmente a executar para a poder terminar e por fim é executada novamente a aplicação atualizada. Existem também logs para acompanhar o comportamento do script que são definidos pela função "log()" e são registados no ficheiro "logs.txt". Para evitar a sobrecarga no ficheiro logs.txt foi criado também um script para apagar este ficheiro a cada quarenta e cinco minutos.

Para que o script execute sistematicamente é usado o agendamento utilizando cron. Com o comando **crontab -e** editamos o ficheiro e adicionamos uma nova tarefa para executar o script de deploy a cada 5 minutos.

```
*/5 * * * * /root/deployScripts/deploy.sh
*/45 * * * * /root/deployScripts/rm_logs.sh
```

Figura 2 - Definição de crontab para execução de scripts

User Story 650

Como administrador do sistema quero que apenas os clientes da rede interna do DEI (cablada ou via VPN) possam aceder à solução

Esta UserStory tem como objetivo controlar e restringir o acesso à a nossa máquina virtual **"Debian 12 (bookworm) - base system"**. Para isso, será necessário configurar as **regras de firewall** da máquina virtual. As regras de firewall permitem ou bloqueiam o tráfego de rede para a máquina virtual.

Para encontrar a **sequência de IPs** que terá acesso à máquina virtual, basta conectar-se à VPN do DEI e visualizar o **IPconfig**.

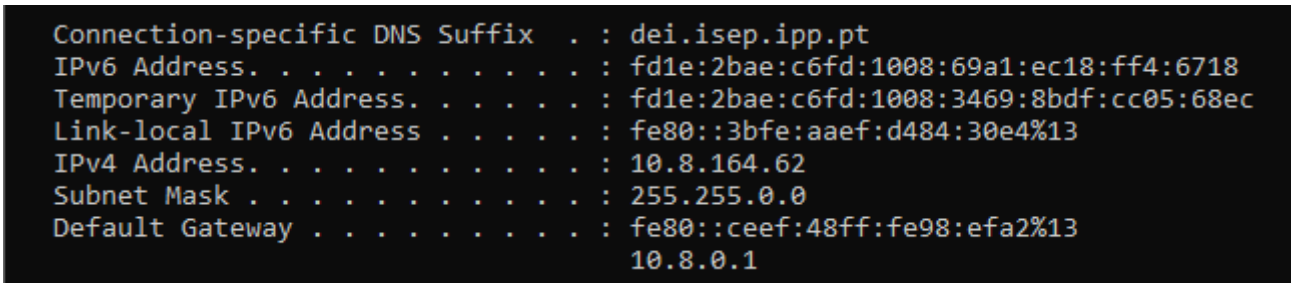


Figura 3 – IPS VPN

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

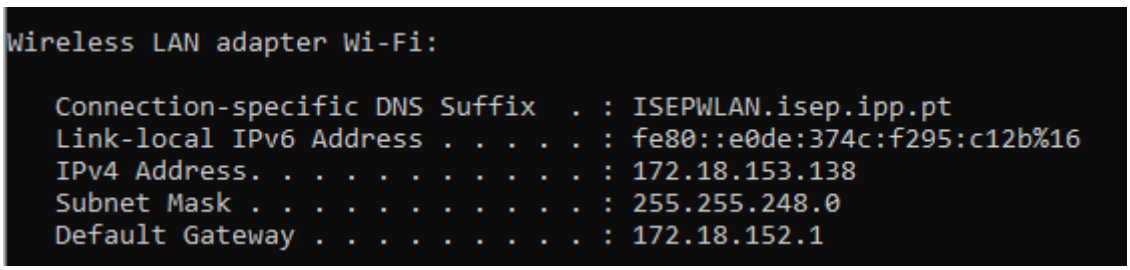


Figura 4 – IPNS WLAN

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

Agora vamos verificar quais **regras de firewall existem** através do comando **"iptables -L"**

```
root@vs1143:~# iptables -L
# Warning: iptables-legacy tables present, use iptables-legacy to see them
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
root@vs1143:~#
```

Figura 5 – IP Tables -L antes de editar

O próximo passo é alterar as **regras de firewall**, usando as ações **Accept** e **Drop**, para garantir os acessos pretendidos. As ações **Accept** permitem o tráfego, enquanto a ação **Drop** o bloqueia. Para que estas ações sejam guardadas vamos executar o comando **"sudo /sbin/iptables-save"** o arquivo de texto é guradado no diretório **/etc/iptables/**, com o nome **rules.v4** para regras **IPv4** ou **rules.v6** para regras **IPv6**.

A primeira regra será A segunda regra será **"iptables -A INPUT -p tcp --dport 4200 -s 10.8.0.0/16 -j ACCEPT"**, esta regra permitirá conexões TCP da rede do DEI (10.8.0.0/16) para a porta **4200**. Essa é a porta da nossa aplicação que irá correr 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 -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.

No segundo caso será igual, mas para os IPS "**172.18.144.0/21"**.

```
root@vs1143:~# iptables -A INPUT -p tcp --dport 4200 -s 10.8.0.0/16 -j ACCEPT
```

Figura 6 - Comando ACCEPT IP Tables VPN

```
root@vs1143:~# iptables -A INPUT -p tcp --dport 4200 -s 172.18.144.0/21 -j ACCEPT
```

Figura 7 – Comando ACCEPT IP Tables WLAN

A outra regra será **“iptables -A INPUT -p tcp --dport 4200 -j DROP”**, esta regra bloqueará todo o tráfego da porta **4200**. Essa é a porta da nossa aplicação que irá correr 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.

```
root@vs1143:~# iptables -A INPUT -p tcp --dport 4200 -j DROP
```

Figura 8 - Comando DROP IP Tables

Após isto usamos o comando **“sudo /sbin/iptables-save”** e o resultado obtido foi o seguinte:

```
root@vs1143:~# sudo /sbin/iptables-save
# Generated by iptables-save v1.8.9 (nf_tables) on Fri Nov 24 16:46:42 2023
*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 Fri Nov 24 16:46:42 2023
# Generated by iptables-save v1.8.9 (nf_tables) on Fri Nov 24 16:46:42 2023
*nat
:PREROUTING ACCEPT [0:0]
:INPUT ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]
-A PREROUTING -p tcp -m tcp --dport 2222 -j REDIRECT --to-ports 22
COMMIT
# Completed on Fri Nov 24 16:46:42 2023
# Warning: iptables-legacy tables present, use iptables-legacy-save to see them
root@vs1143:~#
```

Figura 9 – IP Tables

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

```
root@vs1143:~# iptables -L
# Warning: iptables-legacy tables present, use iptables-legacy to see them
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
root@vs1143:~#
```

Figura 10 - IP Tables -L depois de editar

Através das seguintes figuras podemos constatar a implementação da solução.

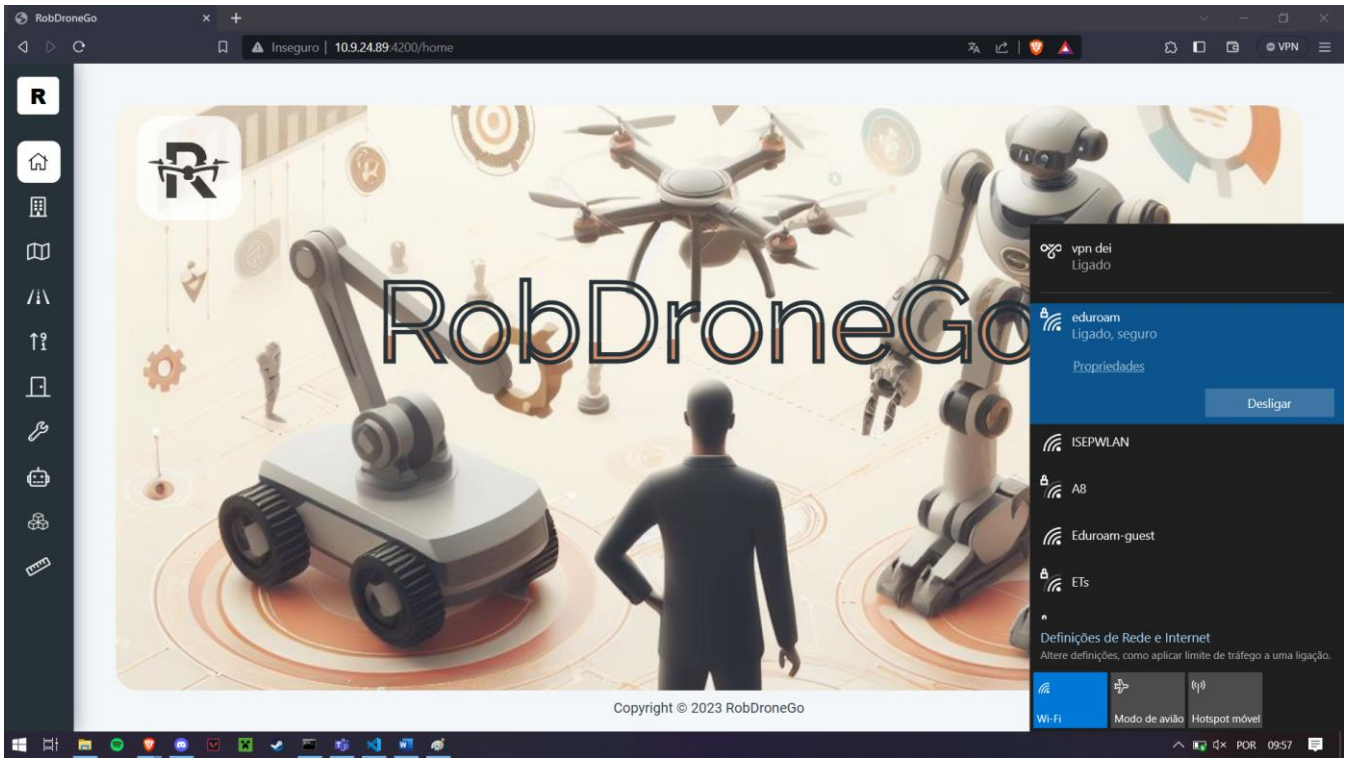


Figura 11 – App com VPN ligada

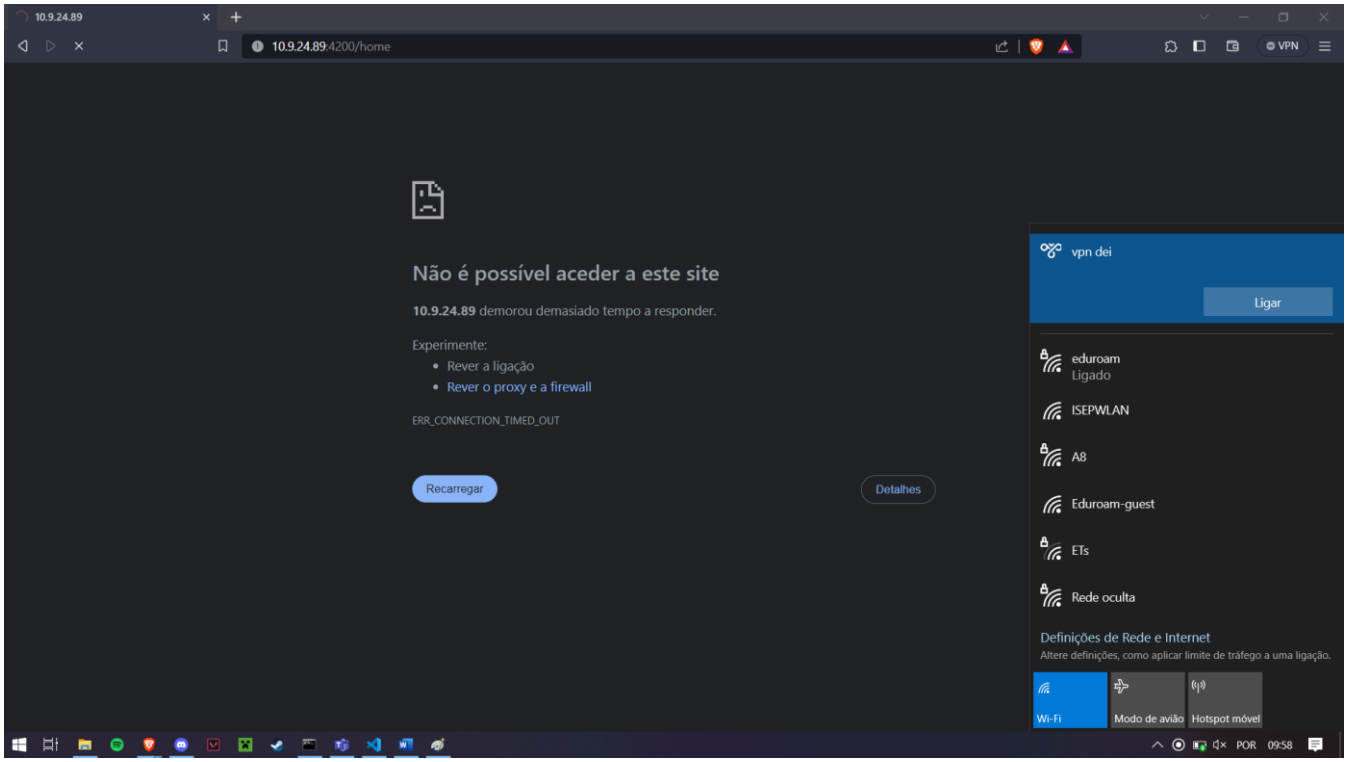


Figura 12 – App sem VPN ligada

User Story 660

Como administrador do sistema quero que os clientes indicados na user story 2 possam ser definidos pela simples alteração de um ficheiro de texto.

Para realizar esta User Story vamos criar um ficheiro /etc/good-guys onde ficarão os IPs dos utilizadores que pretendemos que tenham acesso ao nosso sistema.

Figura 13 /ect/good-guys

Depois de ter os utilizadores definidos vamos criar um script que atualiza as iptables conforme os IPs no ficheiro good-guys

```
#!/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 < /etc/good-guys

/sbin/iptables-save



~
~
~
~
~
~
update_permissions.sh [+]                                    15,0-1          All
```

Figura 14 update_permissions script

Ao executar este script vai acrescentar as iptables definidas na US anterior os IPs que estejam no ficheiro good-guys.

```
root@vs1113:/etc/permissions# iptables -L
# Warning: iptables-legacy tables present, use iptables-legacy to see them
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       tcp  --  anywhere             anywhere             tcp dpt:4200
ACCEPT     tcp  --  10.8.0.0/16          anywhere             tcp dpt:4200
ACCEPT     tcp  --  172.18.144.0/21      anywhere             tcp dpt:4200
ACCEPT     tcp  --  10.8.178.143         anywhere             tcp dpt:4200

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@vs1113:/etc/permissions#
```

Figura 15 resultado da execução do script

Desta forma só permitimos os utilizadores listados no ficheiro possam aceder a aplicação. É importante remover as duas regras definidas anteriormente, pois estas permitem o acesso a qualquer utilizador sempre que esteja na rede do DEI.

User Story 670

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

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

		Impacto		
Probabilidade		1	2	3
	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%)	-	-	-

Através da análise da matriz, é possível concluir que a elevada dependência do servidor do DEI, faz com que um elevado investimento em garantias da segurança e funcionamento deste sejam recomendadas. Para além disso, o aumento de checks na aplicação para apanhar erros na implementação podem ser vantajosas para diminuir as possibilidades de ter consequência elevadas no uso da aplicação.

User Story 800

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

O MBCO (Minimum Business Continuity Objective), é tal como o nome indica o nível mínimo de serviços/produtos que uma organização deve ser capaz de manter durante e após um desastre ou uma interrupção significativa nos seus negócios.

A nossa organização tem como objetivo encontra uma solução para a gestão de uma frota de drones e robots que executam tarefas no interior de um campus, por isso consideramos que os serviços essenciais da nossa aplicação são:

- Criação de rotas de entrega
- Visualização dos dados de robots em tempo real
- Visualização dos dados de edifícios em tempo real

Havendo algum problema crítico com a base de dados, uma avaria por exemplo, haveria um grande problema pois sem os seus dados seria impossível criar os melhores percursos. Para tal mal isto ocorra uma equipa de Resposta a incidentes (ERI) composta por 4 pessoas será notificada através de um sistema de alerta automático. O sistema irá enviar um e-mail a todos os membros da ERI, com informações 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:

- Restauração dos dados de backup para a base de dados
- Reparação/Substituição da infraestrutura necessária
- Reconfiguração da aplicação

Os dados de backup, que são gerados pela base de dados periodicamente serão utilizados neste caso, estes dados estão guardados em duas bases de dados separadas já pensadas para este tipo de situação. A equipa trabalhará em turnos de 4 horas para restaurar os serviços essenciais o mais rapidamente possível.

A organização irá realizar testes e exercícios de recuperação periodicamente para garantir que eficácia dos planos de recuperação. Os testes irão avaliar a capacidade de notificação, avaliação e recuperação da ERI. Os exercícios simularam uma interrupção em tempo real e avaliarão a eficácia do plano de recuperação desenvolvido.

Módulos como o de visualização 3D, que permitem uma visualização gráfica do programa serão deixados de parte, uma vez que consideramos que têm pouco impacto no funcionamento do sistema.

A manutenção normal da aplicação não será mantida enquanto a situação não se encontrar regularizada, todos os esforços serão redirecionados para recuperar o normal funcionamento da aplicação.

User Story 810

Como administrador do sistema quero que seja proposta, justificada e implementada uma estratégia de cópia de segurança que minimize o RPO (Recovery Point Objective) e o WRT (Work Recovery Time)

O RPO (Recovery Point Objective) pode ser definido como a máxima quantidade de informação que é admissível perder após uma falha no sistema que involucre a perda dos dados. O WRT (Work Recovery Time) é o tempo necessário para colocar o sistema de volta em funcionamento após a falha critica.

De acordo com o MBCO definido na user story anterior sabemos que os serviços indispensáveis para a aplicação são:

- Criação de rotas de entrega
- Visualização dos dados de robots em tempo real
- Visualização dos dados de edifícios em tempo real

Portanto os dados mais importantes de manter disponíveis são:

- Os mapas dos pisos e edifícios
- Informação sobre os robots é as tarefas que estão a realizar
- Informação sobre os edifícios e os seus respetivos pisos

Tendo em conta estas informações a estratégia de copia de segurança que iremos aplicar será a seguinte: todos os domingos à meia-noite iremos fazer uma cópia de segurança total da base de dados do sistema, cada dia à meia-noite (excetuando o domingo) serão realizadas cópias de segurança incrementais. Considerando esta solução o RPO seria de 24h.

Serão criados dois scripts para criar os backups totais e incrementais:

```
#!/bin/bash

mongodump --uri=mongodb://mongoadmin:4a5627f241f602c27f1b4f7d@vsgate-s1.dei.isep.ipp.pt:11098

date=$(date '+%Y-%m-%d_%H-%M-%S')

dir_name="/backup/database/$(date '+%Y-%V')"

mkdir -p $dir_name

tar -c -z -f ${dir_name}/backup.${date}.total.tar -p ./dump

rm -fr ./dump

exit 0
```

Figura 16 total backup

```
#!/bin/bash

mongodump --uri=mongodb://mongoadmin:4a5627f241f602c27f1b4f7d@vsgate-s1.dei.isep.ipp.pt:11098

date=$(date '+%Y-%m-%d_%H-%M-%S')

dir_name="/backup/database/$(date '+%Y-%V')"

tar -c -z -f $dir_name/backup.${date}.incremental.tar -p -g /backup/database/backup.snap ./dump

rm -fr ./dump

exit 0
```

Figura 17 incremental backup

Estes scripts criam um ficheiro comprimido com o dia e hora do backup como nome, também indicando se é um backup incremental ou total.

Ainda vamos criar um script para restaurar os dados extraídos da base de dados.

```
#!/bin/bash

backup_dir="/backup/database/$(date '+%Y-%V')"

destination="/restorations"

total_backup=($ls ${backup_dir}/backup.*.total.tar)
tar -xf $total_backup -p -C ${destination}

incremental_backup=$(ls ${backup_dir}/backup.*.increment.tar 2> /dev/null)

if [ -z "${incremental_backup}" ]
then
        echo "No incremental backups found"
else

        for backup in $incremental_backup
        do
                tar -xf ${backup} -p -G -C ${destination}
        done

fi

mv "${destination}/dump" "${destination}/$(date '+%Y-%m-%d')_restoraion"

echo "Backup restoration complete"

exit 0
```

Figura 18 restore script

Este script vai buscar no diretório dos backups os backups da semana e vai restaurar os ficheiros dentro do /restorations com a data e hora da restauração no nome do ficheiro. Este ficheiro pode ser exportado a uma base de dados alternativa enquanto a principal ainda estiver inoperável.

Finalmente os scripts que criam os backups serão agendados usando o crontab, os backups totais serão realizados ao domingo e os incrementais nos restantes dias.

```
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
*/1 * * * * /deploySem5PI/auto_pull_repo.sh

0 0 * * 1-6 /backup_scripts/download_database_increment.sh

0 0 * * 0 /backup_scripts/download_database_total.sh
```

Figura 19 crontab file

Também foi criado um script que ira guardar tanto o frontend e o backend, este backup será guardado num outro servidor, o qual está numa localização física diferente. Este script será executado todos os dias a meia noite.

```
#!/bin/bash

tar -c -z -f /backup/backup.total.tar -p /root/production/ /root/RobDroneGo/

rsync -a /backup/backup.total.$(date '+%Y-%m-%d').tar root@vs1155.dei.isep.ipp.pt:/backups/program

rm -fr /backup/*

exit 0
~
```

Figura 20 backup frontend and backend

User Story 820

Como administrador do sistema quero definir uma pasta pública para todos os utilizadores registados no sistema

Para a resolução desta User Story é necessário primeiro definir qual será a utilidade e função da pasta para o cliente. Posto isto, a equipa definiu que a pasta será utilizada para passar informação dos Managers (Campus, Task, Fleet) para os restantes utilizadores do sistema.

Assim sendo, foram criados dois grupos no servidor, um chamado “managers” e outro chamado “otherUsers”.

```
root@vs1143:/# groupadd -g 6000 managers
root@vs1143:/# groupadd -g 6001 otherUsers
root@vs1143:/#
```

Figura 21 - Criação de grupos de utilizadores

Foram também criados dois utilizadores, um manager e um outro que faz parte do “otherUsers” e adicionámos o utilizador root ao grupo “managers”.

```
root@vs1143:/# useradd -g managers -m campusManager1
root@vs1143:/# useradd -g otherUsers -m luser1
root@vs1143:/# passwd campusManager1
New password:
Retype new password:
passwd: password updated successfully
root@vs1143:/# passwd luser1
New password:
Retype new password:
passwd: password updated successfully
```

Figura 22 - Criação de utilizadores

Foi criado um diretório chamado “sharedFolder” que é a pasta pública.

```
root@vs1143:/# ls
backup  bin  boot  dev  etc  home  lib  lib32  lib64  libx32  media  mnt  opt  proc  root  run  sbin  scripts
sharedFolder  srv  sys  tmp  usr  var
```

Figura 23 - Criação da pasta pública

Mudámos o grupo da pasta para “managers” com o comando **chown :managers /sharedFolder/** e alterámos as permissões do diretório para apenas o dono (root) e membros do grupo do dono (“managers”) terem permissão de escrita, leitura e execução e os restantes apenas de leitura e execução com o comando **chmod 775 sharedFolder/**.

Para que a partilha seja possível foi instalado o “samba” com o comando **apt install samba** e para configurar este acedemos ao ficheiro **etc/samba/smb.conf.**

```
[sharedFolder]
   path = /sharedFolder
   read only = yes
   guest ok = no
   browseable = yes
   create mask = 775
   directory mask 775
   write list = @managers
   force group = managers
   valid users = @managers, @otherUsers
```

Figura 24 - Configuração do ficheiro smb.conf

Nesta configuração definimos o caminho do diretório, definimos para que apenas seja possível leitura, não permitimos o acesso a utilizadores não registados, permitimos a navegação, definimos as permissões de criação de ficheiros e diretórios dentro da "sharedFolder", definimos que o grupo "managers" tem permissão de escrita, forçamos que o grupo dono de tudo aquilo que seja criado seja "managers" e damos acesso aos utilizadores dos grupos "managers" e "otherUsers".

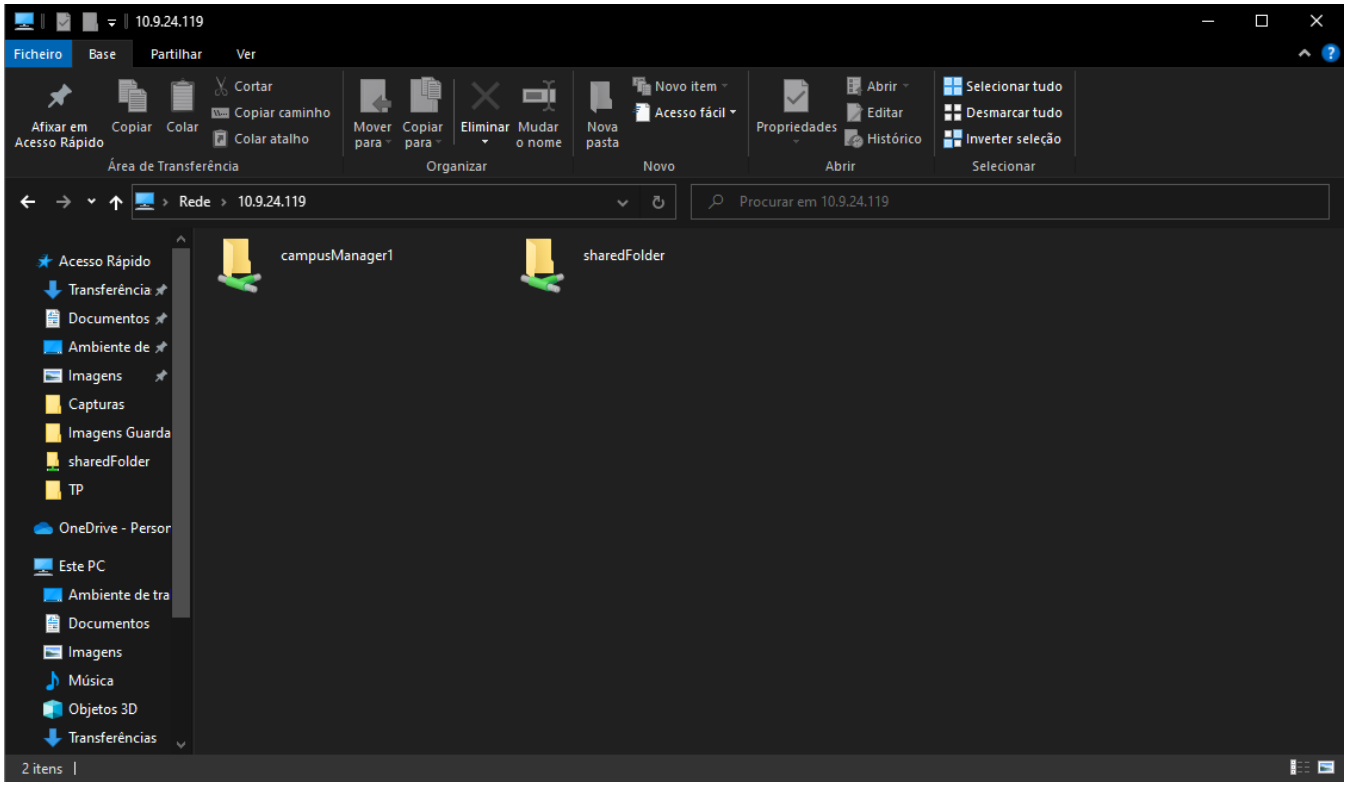
Adicionamos também os utilizadores ao samba.

```
root@vs1143:/# smbpasswd -a campusManager1
New SMB password:
Retype new SMB password:
Added user campusManager1.
root@vs1143:/# smbpasswd -a root
New SMB password:
Retype new SMB password:
Added user root.
root@vs1143:/# smbpasswd -a luser1
New SMB password:
Retype new SMB password:
Added user luser1.
```

Figura 25 - Adição de utilizadores ao samba

Por fim reiniciamos o serviço samba com o comando **systemctl restart smbd**.

Para aceder à pasta vamos ao explorador de ficheiros do Windows e na barra de endereço colocamos o endereço IP da máquina e entrar na “sharedFolder” teremos de fazer log in com os dados de um dos utilizadores definidos previamente.



User Story 830

Como administrador do sistema quero obter os utilizadores com mais do que 3 acessos incorretos

Para fazer esta US é necessário primeiro instalar o módulo que irá guardar as autenticações dos acessos à máquina virtual. Para isso é necessário executar o comando "**sudo apt -y install -y rsyslog**".

Após este módulo estar instalado, irá ser criado um ficheiro automaticamente no diretório **var/log**, com o nome **auth.log**. Este será o lugar que irão ficar gravadas as tentativas de acesso de qualquer utilizador na máquina.

```
2023-11-25T16:41:52.381375+00:00 vs1143 sshd[1336]: Failed password for root from 10.8.1.1 port 60003 ssh2
2023-11-25T16:41:58.614325+00:00 vs1143 sshd[1336]: Failed password for root from 10.8.1.1 port 60003 ssh2
2023-11-25T16:42:03.935496+00:00 vs1143 sshd[1336]: Failed password for root from 10.8.1.1 port 60003 ssh2
2023-11-25T16:42:05.179757+00:00 vs1143 sshd[1336]: Connection reset by authenticating user root 10.8.1.1 port 60003 [preauth]
2023-11-25T16:42:05.179928+00:00 vs1143 sshd[1336]: PAM 2 more authentication failures; logname= uid=0 euid=0 tty=ssh ruser= rhost=10.8.1.1  user=root
2023-11-25T16:42:21.009225+00:00 vs1143 sshd[1339]: Accepted password for root from 10.8.1.1 port 60006 ssh2
2023-11-25T16:42:21.009493+00:00 vs1143 sshd[1339]: pam_unix(sshd:session): session opened for user root(uid=0) by (uid=0)
2023-11-25T16:42:21.047965+00:00 vs1143 systemd-logind[149]: New session 36139 of user root.
2023-11-25T16:42:21.137294+00:00 vs1143 (systemd): pam_unix(systemd-user:session): session opened for user root(uid=0) by (uid=0)
2023-11-25T16:42:21.662436+00:00 vs1143 sshd[1339]: pam_env(sshd:session): deprecated reading of user environment enabled
2023-11-25T16:42:52.638401+00:00 vs1143 chpasswd[1382]: pam_unix(chpasswd:chauthtok): password changed for root
2023-11-25T17:17:01.607435+00:00 vs1143 CRON[1396]: pam_unix(cron:session): session opened for user root(uid=0) by (uid=0)
2023-11-25T17:17:02.220970+00:00 vs1143 CRON[1396]: pam_unix(cron:session): session closed for user root
2023-11-25T18:14:51.247904+00:00 vs1143 sshd[1339]: Received disconnect from 10.8.1.1 port 60006:11: disconnected by user
2023-11-25T18:14:55.708778+00:00 vs1143 sshd[1339]: Disconnected from user root 10.8.1.1 port 60006
2023-11-25T18:14:55.709090+00:00 vs1143 systemd-logind[149]: Session 36139 logged out. Waiting for processes to exit.
2023-11-25T18:14:56.001585+00:00 vs1143 sshd[1339]: pam_unix(sshd:session): session closed for user root
2023-11-25T18:14:56.490378+00:00 vs1143 systemd-logind[149]: Removed session 36139.
2023-11-25T18:17:01.667524+00:00 vs1143 CRON[1413]: pam_unix(cron:session): session opened for user root(uid=0) by (uid=0)
2023-11-25T18:17:02.781598+00:00 vs1143 CRON[1413]: pam_unix(cron:session): session closed for user root
2023-11-25T18:20:16.126830+00:00 vs1143 sshd[1417]: Accepted password for root from 10.8.189.92 port 63002 ssh2
2023-11-25T18:20:16.870238+00:00 vs1143 sshd[1417]: pam_unix(sshd:session): session opened for user root(uid=0) by (uid=0)
2023-11-25T18:20:16.870665+00:00 vs1143 (systemd): pam_unix(systemd-user:session): session opened for user root(uid=0) by (uid=0)
2023-11-25T18:20:16.871088+00:00 vs1143 systemd-logind[149]: New session 36477 of user root.
2023-11-25T18:20:21.425011+00:00 vs1143 sshd[1417]: pam_env(sshd:session): deprecated reading of user environment enabled
2023-11-25T18:40:21.454153+00:00 vs1143 sshd[1699]: Accepted password for root from 10.8.189.166 port 61731 ssh2
2023-11-25T18:40:24.380870+00:00 vs1143 sshd[1699]: pam_unix(sshd:session): session opened for user root(uid=0) by (uid=0)
2023-11-25T18:40:24.381129+00:00 vs1143 systemd-logind[149]: New session 36534 of user root.
2023-11-25T18:40:24.381199+00:00 vs1143 sshd[1699]: pam_env(sshd:session): deprecated reading of user environment enabled
2023-11-25T18:41:48.753735+00:00 vs1143 sudo:     root : TTY=pts/1 ; PWD=/root ; USER=root ; COMMAND=/sbin/iptables-save
2023-11-25T18:41:48.754469+00:00 vs1143 sudo: pam_limits(sudo:session): Could not set limit for 'core' to soft=0, hard=-1: Operation not permitted; uid=0,euid=0
2023-11-25T18:41:48.754521+00:00 vs1143 sudo: pam_unix(sudo:session): session opened for user root(uid=0) by root(uid=0)
2023-11-25T18:41:48.756505+00:00 vs1143 sudo: pam_unix(sudo:session): session closed for user root
2023-11-25T18:43:33.578899+00:00 vs1143 sshd[1699]: Received disconnect from 10.8.189.166 port 61731:11: disconnected by user
2023-11-25T18:43:33.579849+00:00 vs1143 sshd[1699]: Disconnected from user root 10.8.189.166 port 61731
2023-11-25T18:43:33.580338+00:00 vs1143 sshd[1699]: pam_unix(sshd:session): session closed for user root
2023-11-25T18:43:33.588174+00:00 vs1143 systemd-logind[149]: Session 36534 logged out. Waiting for processes to exit.
2023-11-25T18:43:33.589102+00:00 vs1143 systemd-logind[149]: Removed session 36534.
2023-11-25T19:17:02.076392+00:00 vs1143 CRON[2021]: pam_unix(cron:session): session opened for user root(uid=0) by (uid=0)
2023-11-25T19:17:08.164351+00:00 vs1143 CRON[2021]: pam_unix(cron:session): session closed for user root
2023-11-25T19:31:14.505139+00:00 vs1143 sshd[2027]: Accepted password for root from 10.8.4.1 port 53113 ssh2
2023-11-25T19:31:14.707284+00:00 vs1143 sshd[2027]: pam_unix(sshd:session): session opened for user root(uid=0) by (uid=0)
2023-11-25T19:31:14.707570+00:00 vs1143 systemd-logind[149]: New session 36701 of user root.
2023-11-25T19:31:14.992880+00:00 vs1143 sshd[2027]: pam_env(sshd:session): deprecated reading of user environment enabled
2023-11-25T19:36:45.598785+00:00 vs1143 sshd[2027]: Received disconnect from 10.8.4.1 port 53113:11: disconnected by user
2023-11-25T19:36:45.797813+00:00 vs1143 sshd[2027]: Disconnected from user root 10.8.4.1 port 53113
2023-11-25T19:36:45.798827+00:00 vs1143 systemd-logind[149]: Session 36701 logged out. Waiting for processes to exit.
2023-11-25T19:36:45.799158+00:00 vs1143 sshd[2027]: pam_unix(sshd:session): session closed for user root
2023-11-25T19:36:45.800138+00:00 vs1143 systemd-logind[149]: Removed session 36701.
2023-11-25T19:36:49.377161+00:00 vs1143 sshd[2064]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=ssh ruser= rhost=10.8.4.1  user=root
2023-11-25T19:36:51.517508+00:00 vs1143 sshd[2064]: Failed password for root from 10.8.4.1 port 54787 ssh2
2023-11-25T19:36:57.074279+00:00 vs1143 sshd[2064]: Failed password for root from 10.8.4.1 port 54787 ssh2
```

Figura 26 – ficheiro auth.log

Para agora filtrar os resultados para apenas ficarem os utilizadores com mais do que 3 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}'**. Estes irão aparecer no terminal.

```
root@vs1143:~# grep "Failed password" /var/log/auth.log | awk '{print $(NF-5)}' | sort | uniq -c | awk '$1 > 3 {print $2}'
root
root@vs1143:~#
```

Figura 27 – exemplo do comando a rodar