

**Relatório de ASIST Sprint 2**

Rute Ferreira, 1220994

Mateus Cabral, 1220704

Rui Soares, 1221283

Sandro Luís, 1221121

Índice

[Divisão de User Stories 3](#_Toc180925740)

[User Story #6.4.1 4](#_Toc180925741)

[User Story #6.4.2 5](#_Toc180925742)

[User Story #6.4.3 6](#_Toc180925743)

[User Story #6.4.4 8](#_Toc180925744)

[User Story #6.4.5 13](#_Toc180925745)

[User Story #6.5.6 15](#_Toc180925746)

[User Story #6.4.7 16](#_Toc180925747)

[User Story #6.4.8 17](#_Toc180925748)

# Divisão de User Stories

User Stories 6.4.1 e 6.4.5 – Mateus Cabral, 1220704

User Stories 6.4.2 e 6.4.6 – Rui Soares, 1221283

User Stories 6.4.3 e 6.4.7 – Sandro Luís, 1221121

User Stories 6.4.4 e 6.4.8 – Rute Ferreira, 1220994

# User Story #6.4.1

**Procedimento**

Para implementar esta *User Story*, será necessário criar um script capaz de realizar de forma sistemática as seguintes tarefas: obter o módulo escolhido, validar sua funcionalidade executando os planos de teste e, por fim, fazer o *deploy* do módulo.

A solução foi implementada por meio de um script que realiza as etapas de obtenção, teste e *deploy* do módulo. Para garantir a execução sistemática do script, utilizou-se o *crontab*.

**Execução**

1. Criar um script com permissões adequadas de execução;
2. Exportar o path para garantir que o script tenha acesso às aplicações e adicionar variáveis e funções locais úteis durante a execução do programa.

Como o ambiente do *crontab* é limitado, é necessário exportar o *path* para que o script tenha acesso às aplicações e ferramentas necessárias. Sem essa linha, o script não conseguiria executar comandos dependentes de pacotes externos.

A screenshot of a computer program

Description automatically generated

1. Criar a lógica responsável por obter a versão mais recente do módulo;

O script deve verificar o repositório ou o local adequado para obter a versão mais recente do módulo. Isso garante que o processo sempre utilize a versão mais atualizada do módulo.



1. Realizar o *build* do módulo para validar seu funcionamento e registar o resultado;

O script deve fazer a compilação do módulo (build) para verificar se esta está funcional. O resultado do processo de compilação deve ser registado num log.

A computer screen with text

Description automatically generated

1. Interromper a execução de uma versão desatualizada do módulo e iniciar a versão mais recente;

Caso o módulo já esteja em execução, o script deve interromper a versão anterior e iniciar a nova versão do módulo. Esse processo também deve ser registado no log, incluindo detalhes sobre o *deploy* da nova versão.

A screenshot of a computer program

Description automatically generated

1. Adicionar o script ao crontab (utilizando crontab -e, a parte > dev/null serve para ignorar o stdout)



**Resultado**

Caso sucesso:

A screen shot of a computer program

Description automatically generated

Caso insucesso:

Para criar um caso de insucesso irei criar um erro num dos ficheiros do modulo escolhido e validar se a log regista esse erro.

A screen shot of a computer program

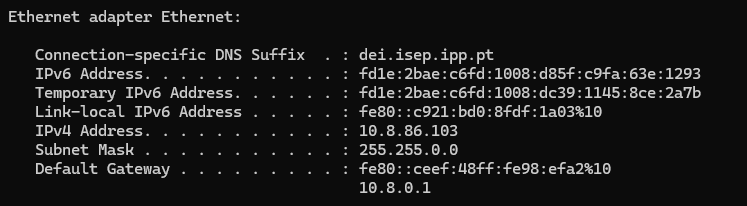
Description automatically generated  
(O erro continua para baixo dando mais informação acerca do erro)

# User Story #6.4.2

**Procedimento:**

Para a realização desta ***User Story***será necessário configurar as regras de ***firewall***. As regras de ***firewall*** vão permitir ou bloquear o tráfego de rede. Neste caso, vou permitir que a rede interna do ***DEI*** aceda à solução, especificando a porta, que é a porta onde a solução será disponibilizada. Todo o tráfego originado de redes não autorizadas para essa porta será bloqueado. Estas regras serão configuradas a nível da ***chain*** ***INPUT***, uma vez que se pretende limitar o acesso, ou seja, o tráfego de entrada.

**Execução:**

1. Primeiramente é necessário obter a sequência de ***IPs*** que terá acesso ao sistema. Para isso conectei-me à rede interna do ***DEI*** através de um cabo ***Ethernet***. Em seguida acedi à Linha de Comandos do ***Windows*** e usei o comando ***ipconfig*** para obter as configurações de rede:

#US6.4.2 - Figura : Configurações de Rede

Analisando as configurações, é possível concluir que o IP atualmente configurado pertence à rede **10.8.0.0/16,** que é rede interna do DEI.

1. Em seguida, já na máquina virtual ***Debian***, vou começar por verificar as regras de firewall atualmente ativas, através do comando ***iptables -L***.
2. Uma imagem com texto, Tipo de letra, captura de ecrã, preto

   Descrição gerada automaticamenteDepois vou configurar as regras de ***firewall*** pretendidas:

#US6.4.2 – Figura 3: Comando Para Regra Firewall

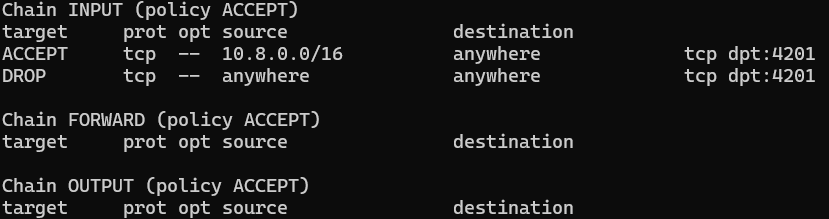
#US6.4.2 - Figura 2: Tabela de Regras de Firewall Antes das Configurações

Esta primeira regra é a responsável por **permitir** que tráfego originado da rede **10.8.0.0/16** consiga aceder à porta ***TCP*** **4201**.

#US6.4.2 - Figura 4: Comando para Regra Firewall

Esta segunda regra tem como função **bloquear** todo o tráfego para a porta ***TCP*** **4201**.

Desta forma, através da combinação destas duas regras, o tráfego originado da rede **10.8.0.0/16** para a porta **4201** será permitido e todo restante tráfego com destino a essa porta será bloqueado.

1. Após a configuração das regras de ***firewall***, utilizo o comando ***iptables-save > /etc/iptables/rules.v4***, que vai guardar as configurações no ficheiro ***rules.v4*** de modo a persistir as configurações, ou seja, em caso de ***reboot*** estas não serão perdidas. O ficheiro ***rules.v4*** é importante porque contém as regras em um formato que pode ser carregado automaticamente durante a inicialização do sistema.
2. Depois, seguido de um **reboot**, através do comando ***iptables -L***, verifica-se que as regras foram guardadas corretamente.

#US6.4.2 - Figura 5: Tabela de Regras de Firewall Depois das Configurações

De notar a ordem com que as regras aparecem, caso estivessem trocadas nenhum tráfego seria permitido, visto que as regras são avaliadas de forma sequencial. Assim que uma regra é correspondida, as regras subsequentes não são verificadas para aquele tráfego.

**Resultado:**

Uma vez que é necessário estar conectado à rede interna do ***DEI*** para conseguir aceder à máquina virtual, para fazer o teste às configurações vou bloquear o ***IP*** da minha conexão ***VPN*** de aceder à porta onde a solução estará disponibilizada, de modo a simular um ***IP*** que não pertence à rede **10.8.0.0/16**. De seguida vou estabelecer outra conexão ***VPN*** de modo a obter um ***IP*** diferente, send este pertencente à rede **10.8.0.0/16**. Neste caso o ***IP*** a ser bloqueado é **10.8.211.3/32**, e então a tabela de regras ***firewall*** (***iptables -L***) ficará assim:

Uma imagem com texto, captura de ecrã, Tipo de letra, preto

Descrição gerada automaticamenteO previsto é que com o ***IP*** bloqueado não consiga aceder à solução e que ao conectar-me novamente à ***VPN*** consiga aceder.

#US6.4.2 - Figura 6: Tabela de Regras Firewall Para o Teste de Configurações

Uma imagem com texto, captura de ecrã, software, Ícone de computador

Descrição gerada automaticamenteCom a conexão via ***VPN*** utilizando o ***IP*** bloqueado **10.8.211.3/32,** simulando que não pertence à rede **10.8.0.0/16**:

#US6.4.2 - Figura 7: Resultado Obtido Para Um IP Não Pertencente à Rede Interna do DEI

Com a conexão via ***VPN*** utilizando um ***IP*** não bloqueado pertencente à rede **10.8.0.0/16**:

Uma imagem com captura de ecrã, texto, software, ecrã

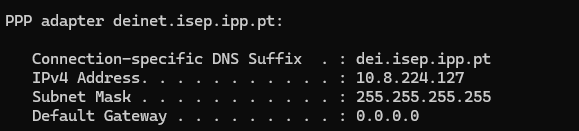
Descrição gerada automaticamenteDesta forma é possível concluir que as configurações de ***firewall*** estão então a funcionar.

#US6.4.2 - Figura 8: Resultado Obtido Para Um IP Pertencente à Rede Interna do DEI

# User Story #6.4.3

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

Primeiro começamos por cria um ficheiro de texto que conterá os ip´s dos clientes permitidos.





*#US6.4.3 - Figura1 ip da vpn do isep*

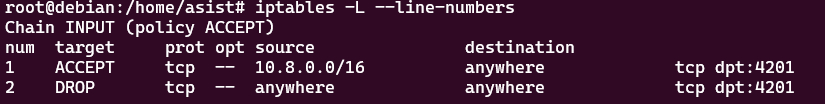
*#US6.4.3 - Figura2 ficheiro com utilizadores permitidos /etc/authorized-clients*

Agora passo a criação do script que é uma solução simples e eficaz para gerenciar permissões de acesso a um serviço específico em um servidor Linux. Ele utiliza o iptables para permitir conexões à porta TCP 4201,22,80 e 443 apenas para endereços IP listados no ficheiro /etc/authorized-clients. A automação reduz a possibilidade de erros e torna o gerenciamento de regras de firewall mais eficiente.

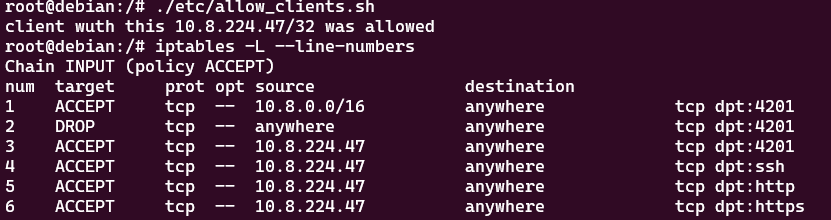


*#US6.4.3 - Figura3 script /etc/allow\_clients.sh*

* Porta 22: Permite conexões SSH.
* Porta 80: Permite conexões HTTP.
* Porta 443: Permite conexões HTTPS.
* Porta 4201: Permite o tráfego na porta 4201 usada na user story 2.



*#US6.4.3 - Figura3 iptables antes de Executar o script*



*#US6.4.3 - Figura4 iptables depois de executar ./etc/allow\_clients.sh*

O objetivo do script é garantir que apenas os usuários cujos endereços IP estão listados no arquivo possam aceder a aplicação.

# User Story #6.4.4

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

**Procedimento:** Para realizar esta *user story*, comecei por identificar os riscos que possam comprometer a solução preconizada. Para quantificá-los elaborei uma matriz de risco que permite o cálculo do valor de cada risco através do produto entre a probabilidade e impacto.

**Execução:**

**1º Passo:** Identificar dos riscos

* Falhas no desempenho da VPN que podem levar à indisponibilidade de acesso à aplicação;
* Falhas de segurança na VPN que podem causar acessos não autorizados;
* Ataques informáticos;
* Falta de monitoramento adequado;
* Perdas financeiras associadas a falhas, tais como interrupções prolongadas da aplicação;
* Erros dos utilizadores como a inserção incorreta de dados;
* Escassez de pessoas para resolver problemas técnicos rapidamente;
* Divulgação indevida de dados sensíveis dos utilizadores.

Uma imagem com texto, captura de ecrã, Tipo de letra, número

Descrição gerada automaticamente**2º Passo:** Classificar e avaliar os riscos

*#US 6.4.4 – Tabela de riscos, probabilidades de ocorrência e impacto*

**3º Passo:** Elaborar a matriz de risco

Através da matriz de risco, é possível visualizar com mais clareza e organização a probabilidade e o impacto de cada risco.

Uma imagem com texto, captura de ecrã, quadrado, Tipo de letra

Descrição gerada automaticamente

*#US 6.4.4 – Matriz de riscos*

**4º Passo:** Quantificar os riscos

Após realizar uma análise da matriz, é possível quantificar os riscos, ou seja, atribuir-lhes um valor que representa a gravidade de cada risco, permitindo a sua comparação e atribuição de prioridade. O valor do risco é calculado através do produto entre a probabilidade e o impacto.

* High – 20 (4 x 5):
* A **indisponibilidade da VPN, falhas da segurança, divulgação de dados sensíveis** possuem uma prioridade máxima para mitigação.
* High – 16 (4 x 4):
* Os **ataques informáticos e as perdas financeiras** representam riscos graves que podem causar danos operacionais e financeiros.
* Serious – 8 (4 x 4):
* Os **erros de utilizadores e a falta de monitoramento** têm um efeito considerável, mas não menos grave do que os anteriores.
* Medium – 4 (2 x 4):
* A **escassez de pessoas** é uma situação que pode ser monitorada com o objetivo que seja evitada.

**Resultado:**

****

*#US 6.4.4 – Tabela da quantificação e identificação dos riscos*

# User Story #6.4.5

O *Minimum Business Continuity Objective* (MBCO) corresponde ao nível mínimo de serviços ou produtos que é aceitável para a empresa durante uma interrupção das operações, garantindo que os seus objetivos essenciais possam ser atingidos.

A nossa aplicação tem como objetivo gerir a utilização de um bloco operatório de um hospital e possui os seguintes módulos:

**SPA (Single Page Application)**: Este módulo estabelece a ligação entre o utilizador e os restantes módulos;

**3D-Visualization-Module**: Responsável pela criação de representações gráficas 3D do bloco operatório do hospital;

**Backend**: Atua como intermediário entre a aplicação e a base de dados;

**Planning Module**: Gere o agendamento de horários para a utilização das salas de operações, assim como a alocação de funcionários;

**Database**: Armazena todos os dados da aplicação.

**Identificação de módulos essenciais**

Entre os cinco módulos descritos, apenas o 3D-Visualization-Module não é considerado essencial, pois a sua funcionalidade centra-se exclusivamente na representação gráfica, que não é crítica para a continuidade das operações da aplicação. No entanto, os outros quatro módulos são extremamente importantes para o funcionamento do sistema, sendo que base de dados assume uma importância ligeiramente superior. A razão para esta distinção deve-se ao facto de que todos os dados críticos da organização estão armazenados nesta, tornando-a assim no núcleo do sistema.

**Estratégias para continuidade e recuperação**

**Base de Dados**

A base de dados é o modulo mais importante e, por isso, requer um plano robusto de continuidade. As seguintes estratégias serão implementadas:

* **Backups regulares**: A base de dados será sujeita a cópias de segurança frequentes, garantindo que os dados estão sempre protegidos e disponíveis para recuperação em caso de falha.
* **Testes periódicos**: Realizar testes de recuperação para assegurar que os backups são funcionais e podem ser restaurados rapidamente.
* **Replicação de dados**: Os backups serão enviados para um local externo ao qual tanto a base de dados principal como uma base de dados alternativa tenham acesso, permitindo uma rápida transição para o sistema alternativo em caso de falha grave.
* **Substituição contingencial**: Caso a base de dados principal deixe de estar operacional, uma base de dados alternativa poderá ser incorporada desde que esteja devidamente sincronizada e atualizada com a principal.

A recuperação da base de dados deve ser feita o mais depressa possível, pois todo o sistema depende do seu funcionamento para operar.

**Restantes Módulos Essenciais**

Os módulos **SPA**, **Backend** e **Planning Module** são também críticos para a continuidade da aplicação. Em caso de falha de qualquer um destes módulos, o plano de recuperação incluirá:

* **Restauro do estado da máquina**: Tentativa inicial de restaurar a funcionalidade do módulo na máquina original.
* **Substituição por máquina alternativa**: Caso o restauro não seja viável, uma máquina alternativa será utilizada para executar o módulo. Neste cenário, será necessário notificar os restantes módulos que comunicam com este sobre a mudança, para que possam redirecionar as interações corretamente.

É também importante notar que a recuperação destes módulos deve também ser feita o mais depressa possível pois o bom funcionamento do hospital não é possível sem eles.

# User Story #6.4.6

**Procedimento:**

Como administrador de sistemas responsável pela implementação de uma estratégia de backup para uma aplicação de um hospital, a necessidade de minimizar o ***RPO (Recovery Point Objective)*** e o ***RTO (Recovery Time Objective)*** é essencial para garantir a continuidade dos serviços, a integridade dos dados dos pacientes e a conformidade com as exigências de disponibilidade dos sistemas médicos. Desta forma, a estratégia que considerei mais eficaz é realizar ***Backups*** **Totais** todos os domingos pelas 00:00 horas, onde existirá menos atividade na aplicação, uma vez que o tempo necessário para completar o ***Backup*** **Total** pode ser longo e pode afetar a performance durante o processo. Nos restantes dias da semana vão ser realizados ***Backups*** **Incrementais** de hora em hora. Estes ***Backups*** de horaem hora permitem reduzir o ***RPO*** para uma hora, o que significa que em caso de falha a perda de dados seria limitada à última hora de alterações. Para um hospital, onde a precisão dos dados e a continuidade do tratamento são vitais, esse ***RPO*** mais baixo é crucial.

A estratégia de ***Backups* Totais semanais** com **Incrementais horários** reduz o ***RTO*,** pois a cada semana é iniciado um novo **Ciclo de *Backups***, garantindo que o ***Backup*** **Total**, a partir do qual os ***Backups*** **Incrementais** vão ser feitos, esteja sempre relativamente recente. Caso contrário, se fosse feito apenas um único ***Backup*** **Total** no início e ***Backups*** **Incrementais** fossem realizados por um longo período (como um mês ou mais, por exemplo), a recuperação exigiria a restauração do ***Backup*** **Total** seguido por todos os **Incrementais** acumulados durante esse período, o que aumentaria significativamente o volume de dados a ser restaurado e, consequentemente, o tempo de recuperação necessário (***RTO***).

**Execução:**

1. Uma imagem com texto, captura de ecrã, Tipo de letra

   Descrição gerada automaticamentePrimeiro comecei por criar um ***Script*** responsável por fazer o ***Backup*** **Total** da Base de Dados, ***Backup*** este que ficará compactado no formato ***.tar.gz***:
2. De seguida criei um Script que tem como função realizar o ***Backup*** **Incremental**:

Uma imagem com texto, captura de ecrã, Tipo de letra

Descrição gerada automaticamente

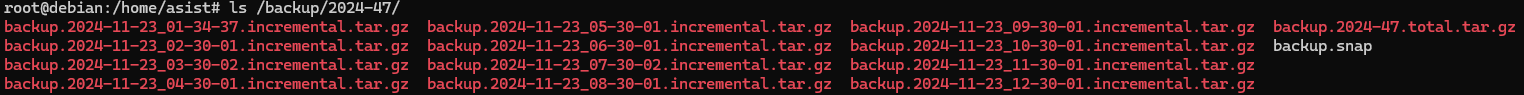
Este ***Script*** exporta o conteúdo da Base de Dados e através do comando ***tar*** compacta-o no formato ***.tar.gz***. O parâmetro ***-g $dir\_name/backup.snap*** especifica o uso do arquivo ***snapshot*** para rastrear as mudanças desde o último ***Backup***. No primeiro ***Backup*** é criado o arquivo ***backup.snap*** que contém uma lista de dados incluídos no ***Backup***. Nos ***Backups*** subsequentes o comando ***tar*** verifica o arquivo ***backup.snap*** e só adiciona ao ***Backup*** os arquivos que foram mudados ou criados desde o último ***Backup***. O ***backup.snap*** é essencial para o ***Backup*** **Incremental**, pois permite que o ***Script*** rastreie apenas as alterações nos dados.

1. Uma imagem com texto, captura de ecrã, Tipo de letra

   Descrição gerada automaticamenteDepois, editei o ficheiro ***crontab*** através do comando ***crontab -e*** de forma a executar o Script de ***Backup*** **Total** à meia-noite de domingo para segunda-feira e o ***Script*** de ***Backup*** **Incremental** de hora em hora, começando às 00:30 horas.

**Resultado*:***

Os ***Backups*** vão ficar organizados por semana, ou seja, sempre que é corrido o ***Script*** de ***Backup*** **Total** é criada uma pasta relativa à semana correspondente, onde esse mesmo ***Backup*** **Total** e os ***Backups*** **Incrementais** associados vão ficar armazenados. O nome dos ***Backups* Totais** será relativo à semana em questão, o nome dos ***Backups* Incrementais** será relativo à data e hora em que forem executados.



Criei também um ***Script*** responsável por restaurar os ***Backups*** automaticamente. Este ***Script*** começa por verificar qual a semana mais recente e de seguida restaura os ***Backups*** pela ordem correta. Começa pelo ***Backup*** **Total**, de seguida organiza os incrementais por ordem crescente da data e depois restaura-os seguindo essa mesma ordem.

Uma imagem com texto, captura de ecrã, Tipo de letra

Descrição gerada automaticamente

# User Story #6.4.7

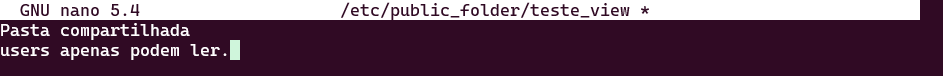
**Como administrador do sistema quero definir uma pasta pública para todos os utilizadores registados no sistema, onde podem ler tudo o que lá for colocado.**

Começamos pela criação de um novo diretório ***/etc/public\_folder*** que será a pasta partilhada e passo por dar apenas permissão de leitura aos utilizadores registados no sistema.


*#US6.4.7 - Figura 1: Criação do diretório partilhado e definição de permissões*

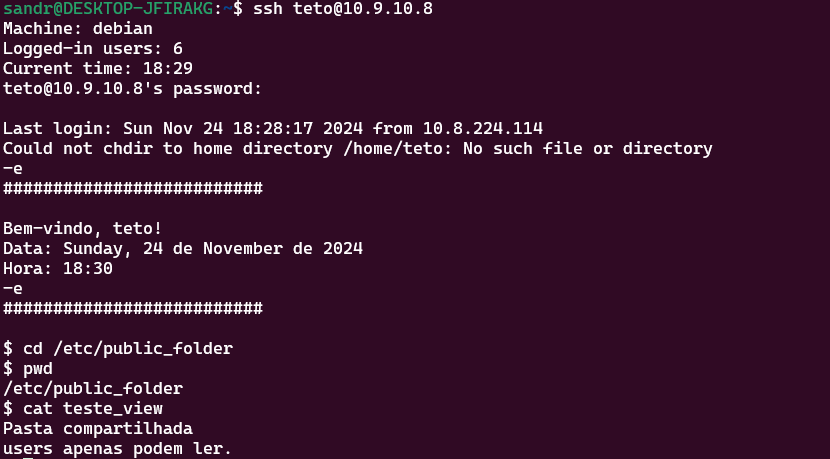
Uso **chmod 750** para dar as seguintes permissões:

* 7 para o proprietário: Permissões de leitura (r), escrita (w) e execução (x).
* 5 para o grupo: Permissões de leitura (r) e execução (x), mas sem permissão de escrita.
* 5 para outros usuários: Permissões de leitura (r) e execução (x), mas sem permissão de escrita.

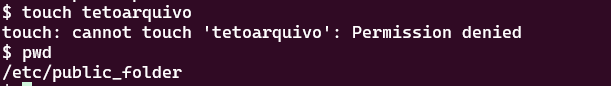
*#US6.4.7 - Figura 2 : criação do ficheiro teste\_view como owner*

Testando a ligação como um utilizador do sistema:

Para este caso usamos o user teto após o login tento aceder o ficheiro ***teste\_view***.  
Como resultado consigo aceder a pasta e ler o conteúdo do ficheiro:



*#US6.4.7 - Figura 4: Leitura de um ficheiro na pasta partilhado como um utilizador do sistema*



*#US6.4.7 - Figura 3: Tentativa de criação de um ficheiro na pasta partilhada*

# User Story #6.4.8

**Como administrador do sistema quero obter os utilizadores com mais de 3 tentativas de acesso incorretas.**

**Procedimento:** Para resolver esta *user* *story*, criei um ficheiro que guardasse apenas as mensagens *logs* vindas do *sshd* executei um comando *grep* para filtrar esse ficheiro e obter os utilizadores com mais de 3 tentativas falhadas.

**Execução:**

**1º Passo:** Criar o ficheiro */etc/rsyslog.d/sshd.conf* com o seguinte conteúdo

Uma imagem com texto, Tipo de letra, captura de ecrã, número

Descrição gerada automaticamente

*#US6.4.8- Conteúdo do ficheiro /etc/rsyslog.d/sshd.conf*

As mensagens de *logs* vindas do *ssh* ficarão guardadas no ficheiro */var/log/sshd.log*.

**2º Passo:** Executar o comando *sudo service rsyslog restart* para reiniciar o *rsyslog*.

Uma imagem com texto, captura de ecrã, menu, preto e branco

Descrição gerada automaticamente

*#US6.4.8- Conteúdo do ficheiro ssh.log*

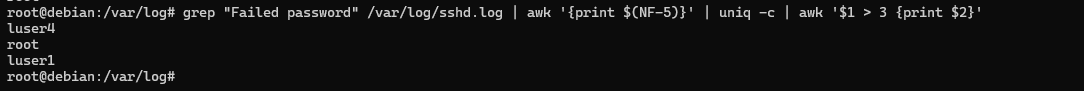
**3º Passo:** Executar o comando ***grep "Failed password" /var/log/sshd.log | awk '{print $(NF-5)}' | uniq -c | awk '$1 > 3 {print $2}'***de modo a obter os utilizadores com mais de 3 tentativas de acesso falhadas.

***grep "Failed password" /var/log/sshd.log***: filtra o ficheiro para encontrar as linhas com o texto “Failed password” que representam tentativas de acesso falhadas.

***awk '{print $(NF-5)}'*** : extrai a quinta palavra a partir do final (NF-5) de cada linha filtrada anteriormente. Geralmente, essa posição corresponde ao endereço *IP* de quem tentou a autenticação falhada.

***uniq -c*** : conta o número de ocorrências de cada endereço *IP*, ou seja, cada tentativa falhada por cada endereço de IP.

***awk '$1 > 3 {print $2}'*** : filtra apenas os *IPs* com mais de 3 falhas de login.

**Resultado:**

*#US6.4.8- Utilizadores com mais de 3 tentativas de acesso falhadas*