Relatório Técnico – Prática 17 – CHMOD E PERMISSÕES DE SERVIDOR

# 1 Identificação

**Disciplina**: Administração de Redes de Computadores

**Professor**: Moisés Andrade

**Aluno**: Anderson de Matos Guimarães

**Data**: 24 de outubro de 2025

**Atividade**: Prática 17 – CHMOD: Gerenciamento de Permissões e Ambiente de Servidor

# 2. Objetivo

Aplicar o comando chmod em diferentes cenários simulando um ambiente de servidor Linux, utilizando notação numérica para definir permissões adequadas a arquivos e diretórios, garantindo segurança, funcionalidade e controle de acesso conforme o papel de cada componente do sistema.

# 3. Materiais e ambiente utilizado

Computador com Linux Mint/Ubuntu instalado

Acesso ao terminal Linux (Ctrl + Alt + T)

Diretório de simulação criado em /tmp/Servidor\_Web

Estrutura de diretórios e arquivos:

* configuracao.conf (arquivo principal de configuração)
* logs/acesso.log (registro de acessos)
* scripts/rotina\_diaria.sh (script de rotina automatizada)
* publico/index.html (arquivo público de página web)

# 4. Procedimentos Executados

## 4.1 Criação da estrutura de diretórios

cd /tmp

mkdir Servidor\_Web

cd Servidor\_Web

mkdir logs scripts publico

touch configuracao.conf

touch logs/acesso.log

touch publico/index.html

touch scripts/rotina\_diaria.sh

## 4.2 Inserção do script

No arquivo scripts/rotina\_diaria.sh, foi inserido o seguinte código:

#!/bin/bash

# Script de Backup

echo "----------------------------" >> /tmp/Servidor\_Web/logs/rotina.log

echo "Rotina executada em: $(date)" >> /tmp/Servidor\_Web/logs/rotina.log

echo "Verificando o estado do serviço..." >> /tmp/Servidor\_Web/logs/rotina.log

/bin/hostname >> /tmp/Servidor\_Web/logs/rotina.log 2>&1

echo "----------------------------" >> /tmp/Servidor\_Web/logs/rotina.log

## 4.3 Aplicação do CHMOD e testes

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Cenário** | **Arquivo/Diretório** | **Permissão Numérica** | **Permissão (rwx)** | **Comando Executado** | **Justificativa de Segurança** |
| 1. Arquivo de configuração | configuracao.conf | **600** | -rw------- | chmod 600 configuracao.conf | Máxima segurança: apenas o dono pode ler e escrever. |
| 2. Log de acesso | logs/acesso.log | **644** | -rw-r--r-- | chmod 644 logs/acesso.log | Leitura para todos, escrita restrita ao sistema. |
| 3. Script executável | scripts/rotina\_diaria.sh | **744** | -rwxr--r-- | chmod 744 scripts/rotina\_diaria.sh | Permite execução pelo dono e leitura por outros usuários. |
| 4. Diretório público | publico/ | **755** | drwxr-xr-x | chmod 755 publico | Padrão de diretório web: acesso de leitura e execução público, modificação apenas pelo dono. |

## 4.4 Teste do script

Após aplicar as permissões, foi executado:

./scripts/rotina\_diaria.sh

Resultado:

* O arquivo logs/rotina.log foi criado com sucesso.
* O script registrou corretamente a data e o nome do host.
* As permissões permitiram execução apenas pelo dono, conforme esperado.

# 5. Resultados

As permissões configuradas foram aplicadas corretamente, refletindo a hierarquia de acesso e segurança adequada para cada tipo de arquivo.

O comando ls -l confirmou as permissões definidas em todos os casos, e o script executou com sucesso, demonstrando o controle de execução e proteção de arquivos sensíveis.

# 6. Dificuldades encontradas

Necessidade de ajustar o caminho de execução (./scripts/rotina\_diaria.sh) para garantir permissão de execução.

Atenção aos modos de permissão quando arquivos são editados com privilégios administrativos (sudo).

Lembrar de usar o parâmetro -d ao verificar permissões de diretórios (ls -ld publico).

# 7. Conclusão

A atividade possibilitou compreender a importância do gerenciamento de permissões no Linux como mecanismo de segurança e organização em servidores.

Os principais aprendizados incluem:

* Interpretação das permissões em notação numérica e simbólica (rwx).
* Definição de políticas de acesso baseadas em segurança e função.
* Criação de um ambiente simulado de servidor para testes práticos.
* Entendimento do papel do CHMOD na proteção de arquivos críticos e scripts automatizados.

# 8 reflexão final

Se o script rotina\_diaria.sh tivesse a permissão 644 (-rw-r--r--), ele não poderia ser executado, pois faltaria o bit “x” (execute) para o usuário proprietário. Assim, o sistema não reconheceria o arquivo como um programa, resultando em erro de permissão durante a execução.