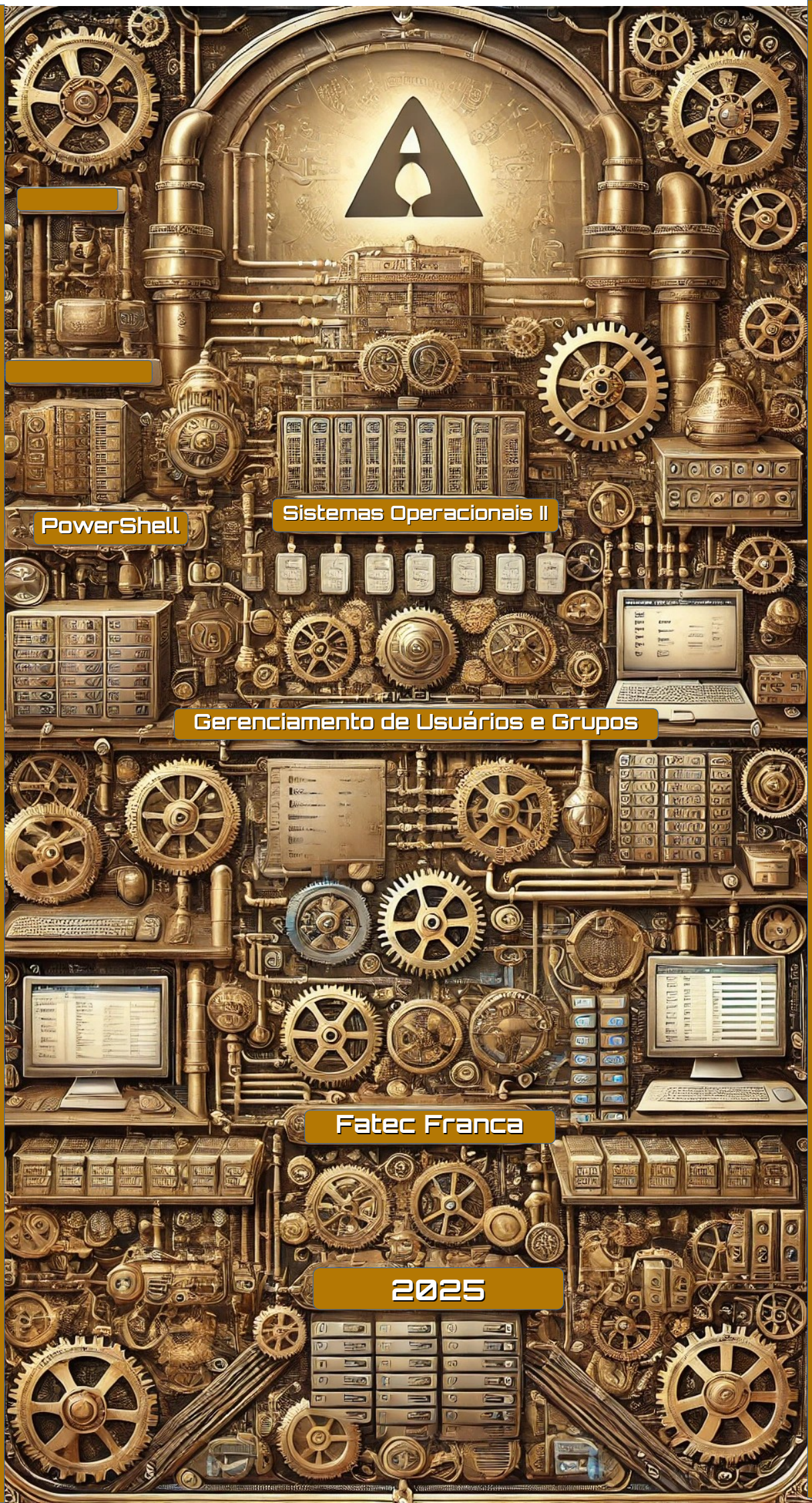
PowerShell
Sistemas Operacionais II
Gerenciamento de Usuários e Grupos
Fatec Franca
2025

PowerShell

Introdução

O PowerShell é um ambiente de automação e gerenciamento de configuração desenvolvido pela Microsoft, que combina uma interface de linha de comando interativa (CLI) com uma linguagem de script robusta baseada em .NET. Ele foi projetado para facilitar a administração de sistemas, a automação de tarefas repetitivas e a integração com diversos serviços e aplicações.

Desde sua criação, o PowerShell passou por uma evolução significativa, tornando-se uma ferramenta essencial para administradores de sistemas, engenheiros de software e especialistas em segurança.

1. Criação e Primeiras Versões

O PowerShell foi desenvolvido como uma resposta às limitações do tradicional Prompt de Comando (CMD) e do Windows Script Host (WSH), que ofereciam poucas funcionalidades para automação e administração avançada do sistema.

A Microsoft iniciou o projeto sob o codinome Monad no início dos anos 2000, liderado por Jeffrey Snover.

Em 2006, a primeira versão oficial, PowerShell 1.0, foi lançada como um componente separado para o Windows XP, Windows Server 2003 e Windows Vista. O principal diferencial do PowerShell foi sua base no .NET Framework, permitindo que os comandos fossem tratados como objetos, em vez de simples texto.

O PowerShell 1.0 já trazia conceitos inovadores, como:

- **Cmdlets** (pequenos comandos especializados).
- **Pipeline** de objetos, permitindo o encadeamento de comandos.
- **Automação avançada** através da execução de scripts.

2. Evolução e Melhorias

Com o sucesso inicial, a Microsoft continuou expandindo o PowerShell, introduzindo melhorias em novas versões:

PowerShell 2.0 (2009)

- ✔ Lançado junto com o Windows 7 e Windows Server 2008 R2.
- ✔ Introduziu Remoting, permitindo execução remota de comandos em outros sistemas.
- ✔ Suporte aprimorado para scripting modular com módulos reutilizáveis.
- ✔ Melhorias na depuração e tratamento de erros.

PowerShell 3.0 (2012)

- ✔ Lançado com o Windows 8 e Windows Server 2012.
- ✔ Introdução do Workflows, permitindo execução de tarefas assíncronas e paralelismo.
- ✔ Expansão da biblioteca de cmdlets.
- ✔ Melhorias na integração com o Windows Management Instrumentation (WMI) e CIM.

PowerShell 4.0 (2013)

- ✔ Lançado junto com o Windows 8.1 e Windows Server 2012 R2.
- ✔ Introdução do Desired State Configuration (DSC), um sistema de gerenciamento de configuração declarativa.

- ✔ Melhorias no suporte a scripts avançados.

PowerShell 5.0 (2016)

- ✔ Lançado com o Windows 10 e Windows Server 2016.
- ✔ Introdução do PowerShellGet, facilitando a instalação de módulos via PowerShell Gallery.
- ✔ Melhorias de segurança e criptografia.
- ✔ Suporte aprimorado para classes e tipos personalizados.

3. A Revolução: PowerShell Core e Open Source

Até a versão 5.x, o PowerShell era baseado exclusivamente no .NET Framework, o que limitava sua compatibilidade com outros sistemas operacionais. A grande mudança ocorreu em 2016, quando a Microsoft anunciou que o PowerShell se tornaria open source e multiplataforma.

PowerShell Core (2018 - Versão 6.0)

- ✔ Primeira versão de código aberto, disponível no GitHub.
- ✔ Baseada no .NET Core, tornando-a compatível com Windows, macOS e Linux.
- ✔ Introdução do modelo de suporte modular, reduzindo a dependência de recursos do Windows.

A mudança para o código aberto fortaleceu a adoção do PowerShell em ambientes corporativos e em nuvem, permitindo maior integração com Azure, AWS, Google Cloud, entre outros.

Status Atual: PowerShell 7 e Futuro

A versão atual do PowerShell é o PowerShell 7, lançado em 2020, baseado no .NET 7.

Principais Características do PowerShell 7:

- ✔ Retorno ao modelo de numeração tradicional, unificando o nome entre Windows e outras plataformas.
- ✔ Melhor desempenho e otimização de comandos.
- ✔ Suporte aprimorado para paralelismo e concorrência.
- ✔ Maior compatibilidade com scripts legados do PowerShell 5.x.
- ✔ Expansão do uso de Inteligência Artificial e automação em nuvem.

O PowerShell 7 continua sendo amplamente utilizado em diversos setores, incluindo administração de servidores, gerenciamento de configurações e automação de infraestrutura em nuvem.

4. Perspectivas Futuras

A Microsoft continua investindo no PowerShell como uma ferramenta essencial para DevOps, gerenciamento de TI e automação. Algumas tendências futuras incluem:

- ✔ Integração com IA e aprendizado de máquina para automação avançada.
- ✔ Expansão de funcionalidades em nuvem para administração remota.
- ✔ Aprimoramento da segurança, com suporte a autenticação moderna e criptografia avançada.

5. A Filosofia do PowerShell

O PowerShell foi projetado com uma abordagem revolucionária para automação e gerenciamento de sistemas, adotando princípios que o diferenciam de outras linguagens de script e interfaces de linha de comando. Sua filosofia é baseada em conceitos fundamentais que garantem flexibilidade, eficiência e integração com diversas tecnologias.

5.1. Tudo é um Objeto

Uma das diferenças mais marcantes do PowerShell em relação a outras ferramentas de automação é sua abordagem baseada em objetos. Em vez de tratar a saída de comandos como simples texto, como ocorre no Prompt de Comando (CMD) ou no Bash, o PowerShell trabalha com objetos do .NET.

Isso significa que os dados gerados pelos comandos podem ser manipulados diretamente, sem a necessidade de parsing manual. Esse princípio permite maior precisão e eficiência ao processar informações.

5.2. Pipeline Baseado em Objetos

Outro conceito central do PowerShell é o pipeline, que permite encadear comandos para que a saída de um seja usada como entrada do próximo. No PowerShell, diferentemente de outras shells, esse pipeline passa objetos estruturados, e não apenas texto.

Esse modelo simplifica operações complexas e melhora a reutilização de dados sem a necessidade de manipulação intermediária, reduzindo erros e otimizando o processamento.

5.3. Cmdlets: Comandos Pequenos e Poderosos

O PowerShell introduziu o conceito de cmdlets, que são comandos pequenos, modulares e reutilizáveis, projetados para realizar tarefas específicas de administração de sistemas.

Os cmdlets seguem a convenção de verbo-substantivo (por exemplo, Get-Process, Set-Item, Remove-User), garantindo uma estrutura previsível e intuitiva. Além disso, o PowerShell permite a criação de cmdlets personalizados, promovendo flexibilidade e extensibilidade.

5.4. Simplicidade e Consistência

Diferentemente de linguagens de script tradicionais, que podem ter sintaxes inconsistentes, o PowerShell foi projetado para ser uniforme e previsível. Seus comandos seguem padrões lógicos, e os parâmetros são estruturados de forma padronizada, tornando o aprendizado mais acessível para administradores de sistemas.

A consistência da sintaxe também facilita a depuração e manutenção de scripts, reduzindo a curva de aprendizado para novos usuários.

5.5. Interoperabilidade e Extensibilidade

O PowerShell é altamente extensível, permitindo a criação de módulos personalizados e integração com APIs externas. Além disso, ele suporta chamadas para:

- ✔ Bibliotecas do .NET
- ✔ APIs REST e SOAP
- ✔ Componentes COM e WMI
- ✔ Ferramentas de terceiros como AWS CLI e Google Cloud SDK

Essa capacidade de interoperabilidade torna o PowerShell uma ferramenta essencial para DevOps, administração de servidores e automação de infraestrutura.

5.6. Automação em Larga Escala

O PowerShell foi projetado para permitir automação em grande escala, minimizando a necessidade de intervenções manuais. Seus recursos permitem:

- ✔ Execução de scripts complexos para gerenciamento de sistemas inteiros.
- ✔ Administração remota, facilitando o controle de múltiplos servidores e dispositivos.
- ✔ Gerenciamento de configuração declarativa, como visto no Desired State Configuration (DSC).

Essa filosofia faz do PowerShell uma ferramenta essencial para ambientes corporativos e operações em nuvem.

5.7. Segurança e Controle

O PowerShell foi desenvolvido com um forte foco em segurança, adotando políticas como:

- ✔ Execução restrita de scripts, exigindo assinaturas digitais para prevenir códigos maliciosos.
- ✔ Gerenciamento granular de permissões, utilizando o Role-Based Access Control (RBAC).
- ✔ Autenticação segura e suporte a protocolos modernos, como OAuth e Kerberos.

Esses mecanismos garantem que administradores tenham controle total sobre o que pode ser executado no sistema.

5.8. Multiplataforma e Open Source

Originalmente, o PowerShell era exclusivo do Windows, mas a Microsoft adotou uma filosofia open source e multiplataforma, tornando-o compatível com Linux e macOS. Essa mudança permitiu sua adoção em ambientes híbridos e reforçou seu papel como ferramenta universal de automação.

A seguir pode-se ver uma tabela de comparação entre os comando básicos mais utilizados do linux (bash) do cmd (windows) e PowerShell.

Ação	Linux (Bash)	CMD (Windows)	PowerShell
Listar arquivos e diretórios	ls	dir	Get-ChildItem
Mudar diretório	cd	cd	Set-Location
Criar diretório	mkdir	mkdir	New-Item -ItemType Directory
Remover diretório	rmdir	rmdir	Remove-Item -Recurse
Criar arquivo	touch arquivo.txt	echo. > arquivo.txt	New-Item -ItemType File
Remover arquivo	rm arquivo.txt	del arquivo.txt	Remove-Item arquivo.txt
Copiar arquivo	cp arquivo1 arquivo2	copy arquivo1 arquivo2	Copy-Item arquivo1 arquivo2
Mover arquivo	mv arquivo1 destino/	move arquivo1 destino	Move-Item arquivo1 destino
Renomear arquivo	mv arquivo1 novo_nome	rename arquivo1 novo_nome	Rename-Item arquivo1 novo_nome
Exibir conteúdo de um arquivo	cat arquivo.txt	type arquivo.txt	Get-Content arquivo.txt
Exibir caminho atual	pwd	cd	Get-Location

Ação	Linux (Bash)	CMD (Windows)	PowerShell
Exibir processos em execução	ps aux	tasklist	Get-Process
Finalizar processo	kill <PID>	taskkill /PID <PID>	Stop-Process -Id <PID>
Ver endereço IP	ip a	ipconfig	Get-NetIPAddress
Apagar tela do terminal	clear	cls	Clear-Host
Criar usuário	sudo useradd <nome>	net user <nome> /add	New-LocalUser -Name <nome>
Excluir usuário	sudo userdel <nome>	net user <nome> /delete	Remove-LocalUser -Name <nome>
Alterar senha	passwd <nome>	net user <nome> *	Set-LocalUser -Name <nome> -Password (ConvertTo-SecureString 'senha' -AsPlainText -Force)
Desligar/ Reiniciar sistema	shutdown -h now / reboot	shutdown /s /t 0	Stop-Computer / Restart-Computer
Listar usuários do sistema	cat /etc/passwd	net user	Get-LocalUser
Ver data e hora	date	time & date	Get-Date

Essa tabela compara os comandos básicos mais utilizados em **Bash (Linux)**, **CMD (Windows)** e **PowerShell**.

6. Manual de Referência - Comandos Básicos do PowerShell

6.1. Gerenciamento de Arquivos e Diretórios

Listar arquivos e diretórios

- **Comando:**
  Get-ChildItem
- **Descrição:** Lista arquivos e pastas no diretório atual.
- **Exemplo:** Listar todos os arquivos e subdiretórios com detalhes:
  Get-ChildItem -Path C:\Users -Recurse

Mudar diretório

- **Comando:**
  Set-Location
- **Descrição:** Muda para outro diretório.
- **Exemplo:**
  Set-Location C:\Users

Criar diretório

- **Comando:**
  New-Item -ItemType Directory

- **Descrição:** Cria um novo diretório.
- **Exemplo:** Criar a pasta "Projetos":
  New-Item -Path C:\Projetos -ItemType Directory

Remover diretório

- **Comando:**
  Remove-Item -Recurse
- **Descrição:** Remove um diretório e seu conteúdo.
- **Exemplo:**
  Remove-Item -Path C:\Projetos -Recurse -Force

Criar arquivo

- **Comando:**
  New-Item -ItemType File
- **Descrição:** Cria um novo arquivo.
- **Exemplo:**
  New-Item -Path C:\Projetos\notas.txt -ItemType File

Remover arquivo

- **Comando:**
  Remove-Item
- **Descrição:** Apaga um arquivo.
- **Exemplo:**
  Remove-Item -Path C:\Projetos\notas.txt

Copiar arquivo

- **Comando:**
  Copy-Item
- **Descrição:** Copia um arquivo de um local para outro.
- **Exemplo:**
  Copy-Item -Path C:\Projetos\notas.txt -Destination C:\Backup\

Mover arquivo

- **Comando:**
  Move-Item
- **Descrição:** Move um arquivo para outro diretório.
- **Exemplo:**
  Move-Item -Path C:\Projetos\notas.txt -Destination C:\Backup\

Renomear arquivo

- **Comando:**
  Rename-Item
- **Descrição:** Altera o nome de um arquivo ou pasta.
- **Exemplo:**
  Rename-Item -Path C:\Projetos\notas.txt -NewName notas_importantes.txt

6.2. Manipulação de Sistema e Processos

Exibir conteúdo de um arquivo

- **Comando:**
  Get-Content

- **Descrição:** Exibe o conteúdo de um arquivo de texto.
- **Exemplo:**
- Get-Content -Path C:\Projetos\notas.txt

Exibir caminho atual

- **Comando:**
  Get-Location
- **Descrição:** Exibe o diretório atual.
- **Exemplo:**
  Get-Location

Exibir processos em execução

- **Comando:**
  Get-Process
- **Descrição:** Lista todos os processos em execução no sistema.
- **Exemplo:**
  Get-Process | Sort-Object -Property CPU -Descending

Finalizar processo

- **Comando:**
  Stop-Process -Id <PID>
- **Descrição:** Finaliza um processo pelo ID.
- **Exemplo:**
  Stop-Process -Id 1234

Ver endereço IP

- **Comando:**
  Get-NetIPAddress
- **Descrição:** Exibe informações sobre os adaptadores de rede e seus IPs.
- **Exemplo:**
  Get-NetIPAddress | Select-Object -Property InterfaceAlias, IPAddress

Apagar tela do terminal

- **Comando:**
  Clear-Host
- **Descrição:** Limpa a tela do terminal.
- **Exemplo:**
  Clear-Host

6.3. Gerenciamento de Usuários

Criar usuário

- **Comando:**
  New-LocalUser -Name <nome>
- **Descrição:** Cria um novo usuário local.
- **Exemplo:**
  New-LocalUser -Name "joao" -Password (ConvertTo-SecureString "Senha123" -AsPlainText -Force)

Excluir usuário

- **Comando:**
  Remove-LocalUser -Name <nome>

- **Descrição:** Remove um usuário local.
- **Exemplo:**
  Remove-LocalUser -Name "joao"

Alterar senha

- **Comando:**
  Set-LocalUser -Name <nome> -Password <nova_senha>
- **Descrição:** Altera a senha de um usuário local.
- **Exemplo:**
  Set-LocalUser -Name "joao" -Password (ConvertTo-SecureString "NovaSenha456" -AsPlainText -Force)

6.4. Controle do Sistema

Desligar ou Reiniciar o Sistema

- **Comando:**
  Stop-Computer
- **Descrição:** Desliga o computador.
- **Exemplo:**
  Stop-Computer -Force
- **Reiniciar:**
  Restart-Computer -Force

Listar usuários do sistema

- **Comando:**
  Get-LocalUser
- **Descrição:** Lista todas as contas de usuário locais.
- **Exemplo:**
  Get-LocalUser

Ver data e hora

- **Comando:**
  Get-Date
- **Descrição:** Exibe a data e hora atuais do sistema.
- **Exemplo:**
  Get-Date -Format "dd/MM/yyyy HH:mm:ss"

Conclusão

O PowerShell evoluiu de uma simples ferramenta de automação para uma plataforma robusta e multiplataforma, essencial para administradores de sistemas e profissionais de TI. Sua flexibilidade, capacidade de automação e integração com serviços em nuvem garantem que ele continue sendo uma peça-chave na administração de ambientes Windows, Linux e híbridos.

A decisão da Microsoft de torná-lo open source e compatível com múltiplas plataformas consolidou seu papel como um padrão para automação e gerenciamento de sistemas modernos.

A filosofia do PowerShell combina objetos, automação, segurança e interoperabilidade, criando uma ferramenta poderosa para administração de sistemas e infraestrutura. Sua abordagem baseada em objetos e pipelines oferece eficiência e flexibilidade, enquanto sua consistência e extensibilidade garantem que ele continue sendo um padrão para automação de TI e DevOps.

O PowerShell não é apenas uma linguagem de scripting, mas sim uma plataforma completa de automação que continua evoluindo para atender às necessidades modernas de administração e gerenciamento de sistemas.

O manual simplificado apresentado aqui cobre os comandos mais essenciais do **PowerShell** para administração de arquivos, processos e usuários no sistema. O PowerShell é uma ferramenta poderosa para **automação** e **gerenciamento**, e seu uso pode ser expandido para tarefas mais avançadas, como **gerenciamento de servidores, redes e segurança**. Caso necessite de mais detalhes ou queira expandir com outros comandos, consulte o material da referencia

Referencias

a. PowerShell

Documentação Oficial da Microsoft

- **Introdução ao PowerShell:** https://learn.microsoft.com/pt-br/powershell/scripting/overview
- **Comandos básicos do PowerShell:** https://learn.microsoft.com/pt-br/powershell/scripting/learn/deep-dives/essential-commands
- **Cmdlets do PowerShell:** https://learn.microsoft.com/pt-br/powershell/scripting/samples/sample-scripts
- **História e evolução do PowerShell:** https://devblogs.microsoft.com/powershell/announcing-powershell-7-0/

b. Windows – Gerenciamento de Usuários e Grupos

Documentação Oficial da Microsoft

- **Gerenciamento de Contas no Windows:** https://learn.microsoft.com/pt-br/windows-server/identity/ad-ds/manage/windows-server-active-directory
- **Gerenciamento de usuários e grupos no Windows:** https://learn.microsoft.com/pt-br/windows/security/identity-protection/access-control/user-account-management
- **Comandos net user e net localgroup no Windows:** https://learn.microsoft.com/en-us/windows-server/administration/windows-commands/net-user https://learn.microsoft.com/en-us/windows-server/administration/windows-commands/net-localgroup

c. Linux – Gerenciamento de Usuários e Grupos

Documentação Oficial e Manuais

- **Comandos useradd, passwd, usermod, groupadd e deluser:** https://linux.die.net/man/8/useradd https://linux.die.net/man/1/passwd https://linux.die.net/man/8/usermod https://linux.die.net/man/8/groupadd https://linux.die.net/man/8/deluser
- **Gerenciamento de usuários e grupos no Linux:** https://ubuntu.com/server/docs/security-users
- **Explicação sobre /etc/passwd e /etc/shadow:** https://www.cyberciti.biz/faq/understanding-etcpasswd-file-format/

d. Active Directory (AD) e LDAP

- **Visão geral do Active Directory:** https://learn.microsoft.com/pt-br/windows-server/identity/ad-ds/active-directory-domain-services-overview
- **LDAP e gerenciamento de diretórios:** https://ldap.com/
- **Introdução ao LDAP para administração de usuários:** https://www.redhat.com/sysadmin/linux-ldap-basics

e. Comparação entre Bash, CMD e PowerShell

- **Documentação Oficial do Bash:** https://www.gnu.org/software/bash/manual/bash.html
- **Guia de comandos do CMD (Prompt de Comando do Windows):** https://learn.microsoft.com/en-us/windows-server/administration/windows-commands/windows-commands
- **PowerShell vs CMD vs Bash – Comparação técnica:** https://adamtheautomator.com/powershell-vs-cmd-vs-bash/

f. Segurança e Gerenciamento de Credenciais

- **Linux PAM (Pluggable Authentication Modules):** https://linux-pam.org/
- **Gerenciamento de Senhas no Windows:** https://learn.microsoft.com/pt-br/windows/security/threat-protection/security-policy-settings/store-passwords-using-reversible-encryption
- **Autenticação e autorização no Windows:** https://learn.microsoft.com/pt-br/windows/security/identity-protection/access-control/authentication-overview