MANUAL DE JOHN THE RIPPER

Auditoria defensiva de hashes, politica de senhas e exercicios de laboratorio

Elemento	Descricao
Nivel	Intermediario a avancado, com base defensiva
Foco	Uso etico, autorizado e documentado em laboratorio ou ambiente proprio
Formato	Manual pratico com comandos seguros, checklists e exercicios
Publico	Estudantes, tecnicos de TI, administradores, analistas SOC e gestores de seguranca

Regra de autorizacao: Todos os exemplos devem ser executados apenas em localhost, maquinas virtuais, laboratorios controlados ou ativos formalmente autorizados. Nao aplique estes procedimentos contra sistemas de terceiros.

Indice do manual

1. Finalidade do John the Ripper

Este manual apresenta a ferramenta com foco em aprendizagem profissional, auditoria autorizada, validacao defensiva e documentacao tecnica. O objetivo nao e promover intrusao, mas capacitar o estudante a compreender riscos, validar controlos e produzir evidencias tecnicas responsaveis.

Limite operacional: Use apenas ativos proprios, maquinas virtuais, laboratorios, ambientes internos autorizados ou alvos explicitamente definidos em contrato/escopo. Qualquer teste fora desse limite pode ser ilegal e causar danos.

John the Ripper e uma ferramenta de auditoria de senhas e hashes. O objetivo defensivo e identificar senhas fracas, validar politicas de complexidade e demonstrar a importancia de hashing adequado, salting, MFA e gestao segura de credenciais.

Uso defensivo	Risco se usado incorretamente
Auditar hashes de teste ou hashes autorizados.	Violacao de privacidade e acesso indevido.
Avaliar qualidade de politicas de senhas.	Exposicao de credenciais reais.
Treinar equipas sobre armazenamento seguro.	Uso de senhas obtidas sem permissao.

2. Instalacao e verificacao

```
sudo apt update
sudo apt install john
john --list=formats | head
john --help
```

A versao jumbo possui suporte a muitos formatos. Em ambiente profissional, documente sempre formato, origem autorizada do hash e objetivo do teste.

3. Conceitos essenciais

Conceito	Descricao
Hash	Representacao nao reversivel de dados, usada para verificar senhas.
Salt	Valor aleatorio que dificulta ataques por tabelas pre-computadas.
Wordlist	Lista de palavras usada para teste controlado de senhas fracas.
Rules	Transformacoes aplicadas a palavras para simular variacoes humanas.
Format	Tipo de hash que John deve interpretar.
john.pot	Ficheiro local onde John guarda resultados descobertos.

4. Exercicio guiado - Hash de laboratorio

Este exercicio usa uma senha criada pelo proprio estudante apenas para demonstracao. Nao utilize hashes reais de pessoas sem autorizacao formal.

```
# Criar uma senha de laboratorio e gerar hash SHA-512 crypt
openssl passwd -6 -salt labsalt SenhaLab2026! > hash_lab.txt

# Criar uma wordlist pequena e controlada
echo SenhaErrada1 > wordlist_lab.txt
echo SenhaLab2026! >> wordlist_lab.txt
echo OutraSenha2 >> wordlist_lab.txt

# Executar John em modo wordlist
john --wordlist=wordlist_lab.txt hash_lab.txt
john --show hash_lab.txt
```

Questões: A senha foi encontrada? Por que uma wordlist pequena conseguiu descobrir a senha? Que politicas poderiam impedir esse resultado?

5. Modos de uso defensivo

Modo	Uso defensivo
Wordlist	Validar senhas comuns ou previsiveis em contas de teste.
Incremental	Demonstrar risco de senhas curtas; usar com cautela por custo computacional.
Rules	Avaliar padroes humanos como trocar letras por numeros.
Show	Gerar evidencia controlada de resultado em laboratorio.
Status	Acompanhar progresso de auditoria autorizada.

```
john --status
john --show hash_lab.txt
john --wordlist=wordlist_lab.txt --rules hash_lab.txt
```

6. Exercicios adicionais

1. Gerar tres hashes de laboratorio com senhas de diferentes comprimentos e comparar resultados.
2. Criar uma wordlist pequena contendo apenas senhas ficticias e documentar o tempo de descoberta.
3. Criar uma politica de senha baseada em frases longas e explicar por que ela e melhor que complexidade artificial.
4. Descrever a diferenca entre hash, criptografia e codificacao.
5. Criar uma recomendacao para armazenamento seguro de senhas em uma aplicacao web.

7. Boas praticas defensivas

- Nunca guardar hashes e resultados em locais sem criptografia.
- Nao auditar hashes de utilizadores sem autorizacao, escopo e justificacao.
- Apagar dados de teste apos a avaliacao quando permitido pela politica.
- Usar algoritmos modernos de armazenamento de senhas, como Argon2, bcrypt ou scrypt quando aplicavel.
- Ativar MFA para reduzir impacto de senhas comprometidas.

- Bloquear senhas comuns e reutilizadas.

8. Projeto pratico final

Criar uma avaliacao de politica de senhas para uma organizacao ficticia. Use apenas hashes de laboratorio criados por si. Entregue uma matriz com senha, comprimento, previsibilidade, resultado, risco e recomendacao.

Criterio	Descricao
Amostra controlada	Somente senhas ficticias e hashes gerados no laboratorio.
Analise	Tempo aproximado, padroes fracos e impacto.
Recomendacao	Frases-passe, MFA, bloqueio de senhas comuns e educacao.

Referencias oficiais e leitura recomendada

Openwall John the Ripper Documentation: https://www.openwall.com/john/doc/

Openwall John the Ripper Usage Examples: https://www.openwall.com/john/doc/EXAMPLES.shtml

Kali Linux Tools - john: https://www.kali.org/tools/john/

Kali Linux Documentation e Kali Tools: https://www.kali.org/docs/ e https://www.kali.org/tools/

NIST Cybersecurity Framework 2.0: https://www.nist.gov/cyberframework

CIS Critical Security Controls: https://www.cisecurity.org/controls