Laboratório Crunch: criando dicionários de senhas

Guia do aluno · Parrot Security OS · Atividade complementar às aulas 06 e 07

Objetivos de aprendizagem

Gerar listas pequenas e reproduzíveis; compreender opções de linha de comando; estimar combinações e tamanho; documentar evidências e explicar o impacto da previsibilidade na segurança das senhas.

Uso autorizado: trabalhe exclusivamente com identificadores fictícios, sem listas vazadas ou dados pessoais. O Crunch gera candidatos; não recupera senhas de forma automática.

1. Preparação do ambiente

```
sudo apt update
sudo apt install -y crunch
crunch -h
mkdir -p ~/TSC03-WLAN/crunch
cd ~/TSC03-WLAN/crunch
```

2. Sintaxe e principais opções

```
crunch <min> <max> [caracteres] [opções]
```

Parâmetro	Uso
min / max	Comprimentos mínimo e máximo
conjunto	Caracteres explícitos; ex.: abc ou 0123
-o	Salva a saída em arquivo
-t	Máscara: @ minúsculas; , maiúsculas; % dígitos; ^ símbolos
-p	Permuta itens; deve ser a última opção

3. Primeiro arquivo: 36 combinações

```
crunch 2 3 abc -o exemplo.txt
```

Quantidade: $3^2 + 3^3 = 36$. Os resultados são combinações de dois ou três caracteres, usando somente a, b e c.

4. Conjunto numérico controlado

```
crunch 4 4 0123 -o quatro_digitos.txt
```

Quantidade: $4^4 = 256$ combinações. A lista não representa o comportamento de uma autenticação que bloqueia tentativas.

5. Máscara com prefixo fictício

```
crunch 9 9 -t LAB2026%% -o padrao_lab.txt
```

A máscara produz de LAB202600 até LAB202699: 100 linhas. Textos fixos reduzem o espaço de busca e demonstram o risco de criar senhas previsíveis.

6. Permutação de três termos

```
crunch 1 1 -o permutacoes.txt -p rede teste 2026
```

São 3! = 6 arranjos. A opção -p deve aparecer ao final; os comprimentos numéricos são ignorados nesse modo.

7. Como verificar o resultado

```
wc -l exemplo.txt
head -n 6 exemplo.txt
tail -n 6 exemplo.txt
du -h exemplo.txt
sha256sum exemplo.txt
```

Registre o número de linhas, o início e o final do arquivo, seu tamanho e o hash SHA-256. Se possível, confronte previsão e observação.

8. Estimativa de combinações e armazenamento

Para k caracteres possíveis e comprimento fixo n, o total é k^n. Para um intervalo de comprimentos, some k^n em cada tamanho. A estimativa de bytes considera cada entrada em ASCII mais um byte de quebra de linha LF. Exemplo: $26^8 = 208.827.064.576$ combinações, o que produziria um arquivo extremamente grande.

Verifique a quantidade de combinações e o espaço disponível antes de executar o Crunch. Use Ctrl+C caso um exercício de geração acidental precise ser interrompido.

9. Mão na massa — Centro Horizonte

Arquivo A: gere combinações de três posições usando a, b, c, d. Resultado esperado: $4^3 = 64$ linhas.

```
crunch 3 3 abcd -o desafio_A.txt
```

Arquivo B: gere uma lista com o padrão LAB2026%%. Resultado esperado: 100 linhas.

Entregas: comandos, contagens, observação dos primeiros e últimos registros, hash e um breve parecer explicando por que senhas longas, exclusivas e aleatórias resistem melhor a listas de candidatos.

10. Perguntas para discussão

a) Qual a diferença entre um conjunto pequeno e uma máscara previsível?

b) Por que 26^8 cresce muito mais do que 26^4?

c) Qual a diferença entre gerar uma wordlist e recuperar uma senha?

d) Por que não se deve extrapolar um teste local para a segurança do WPA3-SAE?

Referências de consulta

Debian Manual — crunch(1): https://manpages.debian.org/unstable/crunch/crunch.1.en.html

Kali Linux Tools — Crunch: https://www.kali.org/tools/crunch/

Senac São Paulo — Plano de Curso n.º 316, UC09: Segurança em Redes sem Fio.

Ficha de registro do experimento (para preenchimento)

Verificação	Resultado obtido
Arquivo A: comando e parâmetros	_____
Arquivo A: linhas e SHA-256	_____
Arquivo B: comando e parâmetros	_____
Arquivo B: linhas e SHA-256	_____
Conclusão e recomendação	_____