Relatório de Análise: Ataques Off-line com John The Ripper

Aluno: Martinho Prata Dos Santos

1. Introdução

Este documento apresenta um resumo técnico dos procedimentos e resultados obtidos em um teste prático de quebra de senhas utilizando a ferramenta John The Ripper. Foram comparados dois métodos principais: o ataque de **força bruta (modo incremental)** e o **ataque por lista de palavras (wordlist)**. A análise visa demonstrar como a complexidade da senha e as configurações de hash (especificamente o uso de *salts*) influenciam a segurança.

2. Materiais e Metodologia

- **Ferramenta Principal:** John The Ripper (JTR)
- **Arquivos de Hashes:** parte1.txt, parte1a.txt, parte2.txt
- **Algoritmo de Hash:** MD5Crypt
- **Lista de Palavras:** merged_wordlist.txt

3. Procedimentos Executados

Parte 1: Ataque de Força Bruta (Incremental)

Nesta fase, o objetivo foi testar a capacidade do JTR de descobrir senhas testando todas as combinações possíveis de caracteres.

- Comando Utilizado:
  time john --format=md5crypt --incremental [arquivo_de_hash]
- **Cenários Testados:**
  1. parte1.txt: Continha 5 hashes, cada um com um *salt* (sal) diferente.
  2. parte1a.txt: Continha os mesmos 5 hashes, mas todos com o mesmo *salt*.

Parte 2: Testes com Limite de Tempo (300 Segundos)

Esta fase comparou a eficácia da força bruta contra o ataque de wordlist em um cenário com tempo limitado, usando um arquivo com 40 hashes (parte2.txt).

- Comando de Força Bruta:
  timeout 300 john --format=md5crypt --incremental parte2.txt
- Comando de Wordlist com Regras:
  timeout 300 john --wordlist=merged_wordlist.txt --rules parte2.txt

4. Resultados Obtidos

Resultados da Parte 1

Arquivo	Sais (Salts	Senhas Quebradas	Tempo de Execução
parte1.txt	5 (diferentes)	5 de 5	44m 42s
parte1a.txt	1 (iguais)	5 de 5	32m 37s

Observação: O uso de salts iguais em parte1a.txt resultou em uma redução de tempo de aproximadamente 27% (~12 minutos).

Resultados da Parte 2 (Comparativo em 300s)

Método de Ataque	Hashes Alvo	Senhas Descobertas
Força Bruta (Incremental)	40	0
Wordlist + Regras	40	7

As senhas descobertas pelo método de wordlist foram: qwerty, password, secret, Ceres, dictionary, a, e uma senha em branco.

5. Análise e Interpretação

5.1. O Impacto dos Salts na Velocidade do Ataque

A principal razão pela qual o teste com parte1a.txt foi significativamente mais rápido é que, com um *salt* único, o John The Ripper pode otimizar os cálculos de hash. Em vez de recalcular cada tentativa para cada um dos 5 *salts* diferentes (como em parte1.txt), ele realiza o cálculo uma vez e o compara com todos os hashes que compartilham aquele *salt*. Isso demonstra que a utilização de **um *salt* único por senha é uma medida de segurança crucial**, pois aumenta exponencialmente o trabalho exigido de um atacante.

5.2. A Eficiência da Wordlist vs. Força Bruta

Os resultados da Parte 2 mostram de forma clara que, para senhas comuns, um ataque direcionado é muito mais eficaz do que a força bruta.

- **A Força Bruta** falhou porque o universo de combinações possíveis é vasto demais para ser explorado em um curto período.

- **A Wordlist** obteve sucesso porque se concentrou em senhas altamente prováveis: palavras comuns, sequências de teclado e nomes próprios. As "regras" do JTR ainda ampliaram o alcance, testando variações (como maiúsculas e números), o que foi suficiente para quebrar 7 senhas em menos de 5 minutos.

6. Conclusões e Recomendações de Segurança

1. **Senhas Fracas São o Maior Risco:** A eficácia do ataque de wordlist prova que senhas baseadas em palavras simples são o principal ponto de falha na segurança de contas.
    - **Recomendação:** Promover o uso de **passphrases** (frases secretas) longas e memoráveis em vez de senhas curtas e complexas.
2. **Salts Únicos São Mandatórios:** A prática de usar um *salt* diferente para cada senha armazenada é uma defesa fundamental contra ataques off-line, aumentando o custo computacional para o atacante.
3. **Use Algoritmos Lentos:** O MD5Crypt é um algoritmo rápido, o que o torna inseguro para os padrões atuais.
    - **Recomendação:** Implementar algoritmos de hash modernos e lentos, como **Argon2**, **scrypt** ou **bcrypt**, projetados especificamente para dificultar a quebra de senhas.
4. **Implemente Defesa em Camadas:** A segurança de uma conta não deve depender apenas da senha.
    - **Recomendação:** Adotar políticas de senha rigorosas, limitar as tentativas de login e habilitar a **Autenticação de Dois Fatores (2FA)** sempre que possível.