Trabalho teórico e prático em dupla

Disciplina: Sistemas Operacionais Distribuídos
Professor: Rafael José de Alencar Almeida
Valor: 20 pontos
Data de entrega: 17/12/2012 (data limite para *upload* no *Moodle*)

As senhas dos usuários de sistemas baseados em Unix costumam ficar armazenadas no arquivo */etc/shadow*. Para garantir a segurança das mesmas, além da permissão de acesso restrito do arquivo, as senhas ficam criptografadas, o que garante, dentre outras coisas, que o usuário *root* não possa ler a senha dos demais usuários.

As entradas no arquivo */etc/shadow* ficam no formato:

usuario:algoritmo_de_hashsalhash_da_senha:outras_informacoes...

Como funções de *hash* são de mão única (irreversíveis), a única maneira *garantida* de se descobrir a senha que, associada ao "sal", produziu determinado *hash*, é por meio do uso de força bruta, ou seja, gerando todas as combinações de senha possíveis em um determinado intervalo, e comparando o *hash* gerado com o *hash* armazenado no arquivo.

Esta tarefa é relativamente trivial, podendo ser implementada facilmente na linguagem Python, com base no exemplo de código abaixo:

```
import crypt, itertools, string

# Calcula todas as senhas de 6 dígitos baseadas na combinação de letras minúsculas
senhas = itertools.product(string.ascii_lowercase, repeat=6)

# Gera o hash de uma senha arbitrária no formato do arquivo /etc/shadow
crypt.crypt('S3NH4', '$algoritmo_hash$sal$')
```

Apesar disso, a segurança das senhas armazenadas ainda é razoavelmente elevada, uma vez que o custo computacional para produzir e checar todas as combinações de letras, números e caracteres especiais de uma senha forte e extensa é muito alto. Caso não haja falhas (e nem *backdoors*) no algoritmo de encriptação, a única alternativa para se quebrar senhas e mensagens criptografadas é através do uso de supercomputadores – técnica amplamente utilizada por agências de inteligência, como a NSA.

Atividade

- Implemente um algoritmo de força bruta para quebrar uma senha extraída do arquivo */etc/shadow*. O programa deve utilizar técnicas de multiprocessamento para acelerar a execução da tarefa.

- Utilize o *cluster* de alto desempenho baseado em Linux implementado nas últimas aulas para testar a execução do programa de forma paralela, em vários nós.

- Escreva um relatório, baseado no formato definido pela SBC <http://www.sbc.org.br/index.php?option=com_jdownloads&Itemid=195&task=view.download&catid=32&cid=38> com os seguintes tópicos (além do título, autores, resumo, conclusão e referências):

  - Introdução teórica sobre *clusters* de alto desempenho e sobre o *cluster* utilizado nas bancadas;
  - Descrição comentada (trecho a trecho) do código implementado;
  - Análise de desempenho da execução do programa, baseada em um gráfico de tempo de execução, utilizando 1 (processamento apenas na máquina *master*), 2, 3 e 4 nós do *cluster*.