Instituto Superior de Engenharia de Coimbra	**Sistemas Operativos** **2012/13** **DEIS – Engenharia Informática**	**Prova escrita** **Parte prática - API** **Época normal** **5 / Julho / 2013**

Pretende-se um sistema informático para testar a segurança de passwords. O sistema funciona da seguinte forma: existe um programa chamado "cracker" que é lançado indicando-lhe a password encriptada a descobrir por argumento de linha de comandos. O programa vai então tentar descobrir a password original. Quando a descobrir apresenta-a no ecran. Também lhe é dado na linha de comandos o número de máximo de segundos para a descoberta da password. Passado esse tempo o programa desiste e termina.

Para aproveitar a capacidade dos processadores multi-core, o programa cracker vai lançar 27 filhos formando com eles um cenário cliente-servidor através de named pipes. Cada filho vai testar uma gama de combinações de letras (o primeiro testa passwords começadas por "a", o segundo começadas por "b", etc. Quando um dos filhos encontra a password original, informa o pai, que por sua vez informa todos os filhos que devem terminar. O programa cracker também informa os filhos quando passa o timeout.

O método de descobrir a password original baseia-se em método de força bruta e não é relevante para este enunciado nem sequer vai ser pedido para implementar esse algoritmo (apenas a matéria de SO).

Todas as alíneas são independentes. Não se pretende ter os programas completos. Apenas partes pontuais de funcionalidade indicadas em cada alínea. Não deve perder tempo com #includes. As escritas em ficheiros ou pipes binários são obrigatoriamente feitas com as funções sistema e não com as de biblioteca.

a) (15%) Implemente a parte do programa cracker que trata do início da sua execução, e que faz:
- **Verifica se tem argumentos de linha de comandos válido, que são dois: o primeiro é a password a descobrir e o segundo é um número que é o timeout. Se não tiver, sai logo.**
- **Liberta a consola imediatamente (método semelhante ao do trabalho prático).**
- **Se já estiver a correr, termina logo com a mensagem "favor esperar – hacking a decorrer" (este aspecto ´pode ser feito de diversas formas, use mesma que usou no trabalho prático).**

b) (20%) Implemente a funcionalidade do cracker correspondente à criação e preparação das tarefas dos filhos:
- **Prepara um named pipe cujo objectivo é o de posteriormente receber informação dos filhos.**
- **Lança 27 filhos. Cada filho vai executar o programa "testa" (note: programa independente).**
- **Envia para um named pipe criado por cada programa testa a informação constituída por uma estrutura contendo uma letra (indica ao filho quais as passwords que deve procurar) e uma matriz de 10 caracteres que é a password encriptada.**
- **Aguarda que um filho lhe diga que descobriu a password (como exactamente = parte da questão).**

c) (15%) Escreva a parte do programa cracker que recebe a password descoberta, que envolve:
- **O programa cracker vai receber no pipe ("qual?" = parte da questão) uma estrutura com 10 caracteres que é a password descoberta.**
- **Indica a todos os filhos que devem terminar enviando o sinal SIGUSR1 a cada um.**
- **Escreve a password no ecran, elimina o seu pipe.**

d) (15%) O programa cracker desiste de procurar password passado um certo número de segundos, indicado na linha de comandos. Passado esse tempo o programa desiste de procurar, envia o sina SIGUSR1 aos filhos, escreve no ecran "timeout", elimina o seu pipe e termina. Escreva a parte do programa cracker que prepara o mecanismo temporizador (que mecanismo é esse = parte da questão) e atende o "evento". Há pelo menos duas formas de fazer isto. Aquela que prepara um mecanismo de alarme, deixando o programa continuar a sua execução é a mais simples e é a que se sugere. Se fez a alínea anterior, na detecção do fim de evento pode

apenas escrever "timeout" no ecran, uma vez que o aviso aos filhos e eliminação do pipe já foi coberto nessa alínea anterior.

e) (20%) Escreva a parte do programa "testa" (filho do cracker) que lê os dados de trabalho enviado pelo cracker, tenta descobrir a password (simulação apenas) e envia o resultado ao cracker. Esta sequência de tarefas é muito simples:

- **Cria um pipe para receber as ordens do cracker.**
- **Lê a estrutura com os dados para trabalhar.**
- **Invoca uma função chamada char * descobre(char * letraInicial).**
- **Se o resultado da função descobre for NULL, a password não começa pela letra que este filho recebeu e o programa termina. Se for diferente de NULL, foi descoberta a password e o programa envia essa password ao cracker e termina. O programa elimina sempre o seu pipe.**

f) (15%) O programa "testa" termina ordeiramente (eliminado o seu pipe) quando recebe o sinal SIGUSR1. Escreva a parte do programa que prepara a recepção do sinal e o código que atende esse sinal.