Relatório do projeto 1.1 e 1.2 de SO/PSO

Alunos: Giovanne da Silva Santos, Kevin Wallacy de Souza Maciel.

Parnamirim, 11 de março de 2019

Introdução

Neste relatório vamos detalhar o passo a passo da construção do anti fork bomb, assim como a construção da árvore hierárquica de processos. Detalhando o pensamento por trás da abordagem utilizada bem como o caminho para chegarmos a tal conclusão.

Implementação do anti fork bomb

Para implementar o anti fork bomb, pensamos em duas abordagens:

1. **Identificar e matar o processo causador do bomb.**
2. **Limitar o número máximo de processos da aplicação.**

1° estratégia, identificar e matar o processo malicioso.

Essa estratégia consiste em usar o comando **“ps -e -o ppid,user| sort | uniq –c"** para gerar uma lista com todos os processos rodando no sistema.

Com essa lista em mãos, analisamos se o usuário padrão (usuário atual da máquina) possui algum processo que tenha mais filhos do que um certo limite (definido pelo usuário ao rodar a aplicação anti fork bomb), caso haja um processo que exceda esse limiar, mandamos o comando **“kill -9 pid”** (onde pid é o id do processo suspeito) que irá matar o processo e assim parar o fork bomb.

Porém, essa estratégia acarretou alguns problemas de execução, como não conseguir executar o comando **“kill -9 pid”** enquanto o fork bomb estava em execução, pois o kill também é um processo e o sistema operacional não estava deixando-o ser executado.

Outro problema com essa implementação fora que ao encerrar o fork bomb manualmente (via Ctrl + C) o anti fork conseguia executar o kil e matava o sistema operacional, o que reiniciava a máquina e tornava inútil o tratamento (como o fork bomb pode ser tratado apenas desligando o sistema, não faz sentido o tratamento também desligar o sistema no processo).

Foi por esses dois problemas que resolvemos abandonar essa estratégia e tentar limitar a quantidade máxima de processos por usuário.

2° estratégia, limitar o número máximo processos de um usuário

Como a 1° estratégia estava causando muitos problemas, optamos por seguir esse caminho, que consiste em limitar a quantidade de processos do usuário padrão.

Esse método consiste em executar o fork bomb e o seu tratamento num mesmo código, previamente mudando (no mesmo código) o limite de processos que um usuário pode executar (utilizando a função setrlimit()).

Após limitar a quantidade de processos que o usuário pode executar (esse limite é definido pelo próprio usuário ao executar o código) entramos no laço infinito do fork bomb, neste laço uma variável ficará responsável por receber o retorno da função **fork()**, caso o fork retorne –1 significa que houve um erro e ele não foi criado, após isso verificamos se o erro fora devido ao usuário ultrapassar o limite, caso tenha sido, mandamos uma mensagem de que o tratamento foi executado com sucesso e o usuário já pode fechar o programa.

Segunda parte do trabalho : utilização da pasta /proc.

A segunda parte do trabalho,primeiramente, utilizou-se os comandos : “ps aux | wc -l” para monitorar os processos sendo realizados na máquina ; e o comando “ps -e -o user | sort | uniq -c” para também monitorar os processor porém organizados pelos usuários. Além disso, foi usado uma funação chama “exec()” que serve para converter uma string e um comando para o terminal.

E para a impressão da árvore , foi usado o comando “ps tree” para imprenssão dela a partir do pid desejado e exportado no formato .txt .