UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DIRETORIA DE PESQUISA E PÓS-GRADUAÇÃO
DEPARTAMENTO ACADÊMICO DE INFORMÁTICA
CURSO DE ESPECIALIZAÇÃO EM TECNOLOGIA E SOFTWARE LIVRE

PRISCILA VRIESMAN ARAUJO

DETECÇÃO DE ANOMALIAS EM PROCESSOS DO SISTEMA OPERACIONAL LINUX

MONOGRAFIA DE ESPECIALIZAÇÃO

CURITIBA
2017

PRISCILA VRIESMAN ARAUJO

DETECÇÃO DE ANOMALIAS EM PROCESSOS DO SISTEMA OPERACIONAL LINUX

Monografia de Especialização, apresentado ao Curso de Especialização em Tecnologia e Software Livre, do Departamento Acadêmico de Informática, da Universidade Tecnológica Federal do Paraná – UTFPR, como requisito parcial para obtenção do título de Especialista.

Orientador: Prof. Fabiano Kuss

CURITIBA
2017

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Curitiba
Diretoria de Pesquisa e Pós-Graduação

TERMO DE APROVAÇÃO

DETECÇÃO DE ANOMALIAS EM PROCESSOS DO SISTEMA OPERACIONAL LINUX

por

Priscila Vriesman Araujo

Esta monografia foi apresentada às 19 horas do dia 19 de abril de 2017 como requisito parcial para a obtenção do título de ESPECIALISTA EM TECNOLOGIA E SOFTWARE LIVRE, do Programa de Pós-Graduação da Universidade Tecnológica Federal do Paraná. O candidato foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após a deliberação, a Banca Examinadora considerou o trabalho aprovado.

_____
Msc Christian Carlos Souza Mendes
UTFPR

_____
Msc Leandro Batista de Almeida
UTFPR

_____
Msc Fabiano Kuss
SERPRO

_____
Prof. Msc Lincoln Herbert Teixeira

Dedico esse trabalho como contribuição de mais um passo de minha pesquisa no Sistema Operacional Linux, fruto de um trabalho de mestrado, com o objetivo de colaborar para projetos de Software Livre e para sociedade.

AGRADECIMENTO(S)

Agradeço a todos os professores do Curso de Especialização em Tecnologia e Software Livre da UTFPR, pela atenção e os ensinamentos que colaboraram muito para o meu aperfeiçoamento técnico.

There's innovation in Linux. There are some really good technical features that I'm proud of. There are capabilities in Linux that aren't in other operating systems.
(Linus Torvalds)

RESUMO

ARAUJO, Priscila Vriesman. **Detecção de anomalias em processos do Sistema Operacional Linux.** 2017. 41 f. Monografia (Curso de Especialização em Tecnologia e Software Livre), Departamento Acadêmico de Informática, Universidade Tecnológica Federal do Paraná. Curitiba, 2017.

Os processos provêm todas as informações de aplicações, programas auxiliares e serviços *batches (*redes, *daemons* diversos), gerenciando tudo que é executado num Sistema Operacional. Muitos processos são demandados sem o conhecimento prévio de que possam gerar algum dano ao funcionamento do sistema. Manter integridade e a garantia de desempenho é fundamental para um bom gerenciamento dos Sistemas Operacionais. Em mineração de dados, considera-se a técnica de detecção de anomalias a identificação de uma ação diferente do comportamento normal. Assim, este trabalho apresenta um modelo de detecção de anomalias com base no comportamento dos processos do Sistema Operacional Linux.

Palavras chave: Detecção de Anomalias. Mineração de Dados. Sistemas Operacionais.

ABSTRACT

ARAUJO, Priscila Vriesman. **Anomaly Detection in processes of the Linux Operating System**. 2017. 41 f. Monografia (Curso de Especialização em Tecnologia em Software Livre), Departamento Acadêmico de Informática, Universidade Tecnológica Federal do Paraná. Curitiba, 2017.

The processes provide information about applications, auxiliary and service programs such as network and daemons, managing that is executed in the Operating System. Many processes are required without the prior knowledge that managers can test the system operation. Maintaining integrity, good performance and security assurance are fundamental to a good management of Operating Systems. In data mining, it is considered an anomaly detection technique to identify an action different from normal behavior. Thus, this work presents an anomaly detection model based on the behavior of the Linux Operating System processes.

Keywords: Detection Anomaly. Data mining. Operating Systems.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS, SIGLAS E ACRÔNIMOS

API	*Application Programming Interface*
ARFF	*Attribute Relation File Format*
CFS	*Correlation-based Feature Selection*
CPU	*Central Processig Unit*
GDM	*GNOME Display Manager*
GNU	*Gnu's Not Unix*
GTK	*Gimp Tolkit*
GVFs	*GNOME Virtual file system*
LOF	*Llocal Outlier Factor*
PAM	*Pluggable Authentication Module*
SSH	*Secure Shell*
VFS	*Virtual File System*
VNC	*Virtual Network Computing*
WEKA	*Waikato Environment for Knowledge Analysis*

SUMÁRIO

1 INTRODUÇÃO

Desde a criação dos primeiros computadores, a computação evoluiu rapidamente, sendo a década de 70 caracterizada por um melhor aperfeiçoamento da tecnologia através da otimização dos problemas de usuário, garantia da confiabilidade e desempenho.

Este avanço tecnológico, fez com que a computação se tornasse onipresente, os usuários passaram a realizar diversas tarefas como a navegar na Internet, jogar, editar textos e planilhas, entre outras, provocando um aumento da diversidade de aplicações e a complexidade das aplicações de uso geral.

De acordo com (TANENBAUM, 2001), em decorrência destas mudanças é comum encontrar problemas de desempenho relativos ao alto consumo de memória e tempo excessivo de processamento, onde o sistema operacional é o responsável por realizar todo o gerenciamento, com o controle das tarefas para proporcionar um ambiente interativo ao usuário.

Nesse contexto, cada aplicação, ou programa, é representado por um ou mais processos, que possuem diversas características durante seu ciclo de vida (TANENBAUM, 2001). Com a análise das características dos processos, é possível identificar os processos que não se adequam ao comportamento normal do sistema, podendo ser considerados como anomalias**.**

Este trabalho propõe um estudo de detecção de anomalias em processos do Sistema Operacional Linux utilizando técnicas de mineração de dados, com o objetivo de identificar comportamentos irregulares nos processos que possam prejudicar o desempenho do Sistema Operacional.

1.1 Problema

O caso de estudo será realizado na identificação de anomalias em processos do Sistema Operacional Linux, observados em relação das 108 características que contém as informações dos programas executados de um servidor de terminais remotos utilizados por alunos de graduação de uma instituição de ensino.

Geralmente os problemas que degradam o desempenho do sistema operacional são programas com alto consumo de processamento e memória, que

podem provocar a interrupção das aplicações ou até mesmo travamento em todo o sistema operacional. Encontrar as causas desses problemas, usualmente, envolvem a análise de logs e métricas de sistemas, e requerem uma administração manual.

Através da técnica de detecção de anomalias, em mineração de dados, pode-se identificar padrões de comportamentos não normais dos processos, para que o gerenciamento do Sistema Operacional haja de forma preventiva, podendo identificar e bloquear os processos que estejam prejudicando o seu desempenho.

1.2 Objetivos

Nesta seção são apresentados os objetivos geral e específico do trabalho, relativos ao problema anteriormente apresentado.

1.2.1 Objetivo Geral

Avaliar o comportamento de processos do Sistema Operacional Linux para a identificação de anomalias nos processos utilizando técnicas de mineração de dados.

1.2.2 Objetivos Específicos

- Selecionar as características mais importantes dos processos do sistema operacional Linux para a criação do modelo de detecção de anomalias;
- Definir as principais características dos processos que possam ser indicadas como anomalias;
- Quantificar e identificar os processos do Linux, que possam indicar alguma anomalia no contexto do trabalho proposto.

1.3 Justificativa

Analisando o grau de importância do gerenciamento de processos, muito se tem pesquisado na área em busca da otimização do escalonador de processos,

conforme (LOVE, 2003), onde o sistema operacional deve fornecer um tempo rápido de resposta aos processos para as aplicações de interação com o usuário.

Além disso, outros tipos de processos não interativos podem consumir muitos recursos do sistema, o que podem prejudicar os processos interativos ou o sistema operacional como um todo.

A garantia da confiabilidade e a qualidade do sistema operacional estão diretamente relacionadas ao monitoramento dos processos, no que diz respeito ao tempo de CPU e consumo de memória.

Existem diversas ferramentas de monitoramento de desempenho no Linux, porém esse estudo visa a identificação de anomalias com base nas informações dos processos dentro de grupos de processos, utilizando técnicas de mineração de dados.

1.4 Estrutura do trabalho

O presente trabalho apresenta-se em capítulos, a seguir descritos:

Capítulo 1 - Introdução: justificativa do trabalho proposto, bem com os objetivos, estrutura do documento e as limitações deste trabalho.

Capítulo 2 – Detecção de anomalias: conceito e técnicas de detecção de anomalias.

Capítulo 3 – Mineração de dados: revisão da literatura sobre mineração de dados, especialmente para detecção de *outliers.*

Capítulo 4 – Metodologia: neste capítulo serão descritos a metodologia para o desenvolvimento da pesquisa e os resultados obtidos.

Capítulo 5 – Considerações finais: apresenta as considerações finais, se os objetivos foram atingidos e as recomendações aos trabalhos futuros.

2 DETECÇÃO DE ANOMALIAS

Este capítulo apresenta os principais conceitos sobre detecção de anomalias em mineração de dados.

2.1 Introdução

De acordo com (HAN e KAMBER, 2000), a mineração de dados consiste num processo de descoberta do conhecimento com a utilização de técnicas e métodos que possibilitem realizar a análise em grandes quantidades de dados num conjunto de dados para a extração de uma informação previamente não conhecida.

No processo de mineração de dados é possível identificar o tipo de tarefa aplicado à aprendizagem do conhecimento, de modo que seja utilizado o algoritmo corretamente. Uma classificação das tarefas de mineração de dados é dada por (CHEN et. al, 1996): regras de associação, padrões sequenciais, classificação ou predição, análise de agrupamento (*clustering*), e análise de *outliers* (detecção de anomalias).

A análise de *outliers*, foco deste trabalho, é uma tarefa comumente utilizada em diversas áreas e aplicações, como sistemas de saúde, fraudes, detecção de intrusos, e sistemas de segurança. O objetivo da detecção de anomalias é a busca de padrões de dados não conformes com o comportamento esperado (CHANDOLA et. al., 2009). Essas não conformidades podem ser informações discordantes, exceções ou peculiaridades.

Definir uma região normal engloba muitas possibilidades de um comportamento entre uma anomalia normal e anormal não ser preciso, podendo identificar dados incorretamente como anômalos. Além disso, os dados podem conter ruídos similares a uma anomalia e por isso são mais difíceis de distinguir.

Outro fator importante é que a identificação da anomalia pode se tornar mais difícil quando uma se trata de uma ação maliciosa, as informações maliciosas se adaptam entre elas e fazem que observações anormais se apresentem normais. (CHANDOLA et. al., 2009).

Desta forma, existem várias técnicas para detecção de anomalias, que são induzidas dependendo de seu tipo, da disponibilidade da classificação e o contexto da detecção a ser proposto. A Tabela 1 representa vantagens e desvantagens da detecção de anomalias.

Tabela 1. Características da detecção de anomalias

Vantagens	• Detecta comportamento não usuais, logo possui a capacidade de detectar sintomas de ataques sem conhecimento prévio. • Produz informações úteis que podem ser utilizadas no gerenciamento do sistema operacional.
Desvantagens	• Pode identificar incorretamente as anomalias, devido ao comportamento imprevisível de usuários e sistemas. • Requer muitas sessões para coleta de amostra de dados a modo de caracterizar os padrões como comportamentos normais.

2.2 Tipos de anomalias

Em relação aos tipos de anomalias, de acordo com (CHANDOLA et. al, 2009), estas anomalias podem ser classificadas nas seguintes categorias:

- Anomalias ponto: se um dado individual de uma instância pode ser considerado como anormal, então a instância é denominada como um ponto anormal.

- Anomalias contextuais: se uma instância é anormal num contexto específico, de atributos contextuais ou comportamentais. Os atributos contextuais são usados para determinar o contexto para a instância. Já os atributos comportamentais definem as características não contextualizadas para a instância.

- Anomalias coletivas: se uma parte dos dados da instância são anormais, todo o resto é anormal. Os dados individuais da instância anormal podem não ser anomalias, mas juntos se tornam anormais.

2.3 Técnicas de detecção de anomalias

A identificação de uma instância se é anormal ou não, pode requer muitas vezes um esforço manual, para se obter o conjunto de treinamento, a fim de cobrir todo o tipo possível de comportamento, porém muitas vezes podem surgir outros tipos de anomalias que não são identificadas previamente, e uma instância normal pode estar classificada incorretamente.

Assim sendo, as técnicas de detecção de anomalias podem operar nos seguintes modos (CHANDOLA et. al, 2009):

- Detecção de anomalias supervisionada: o treinamento supervisionado, utilizando uma base de dados com os dados rotulados par as instâncias normais e anormais.
- Detecção de anomalias semi-supervisionadas: assume os dados de treinamento, rotulando as instâncias somente para a classe normal.
- Detecção de anomalias não supervisionadas: não requer dados de treinamento, e são amplamente aplicáveis, pois assumem que as instâncias normais sejam mais frequentes que as anomalias nos dados de testes.

2.4 Algoritmos de detecção de anomalias

Os algoritmos de detecção de anomalias inicialmente, tinham como objetivo encontrar anomalias num contexto global, onde todas as instâncias eram tratadas da mesma forma, independente da região do espaço em que os atributos se encontram, o que poderiam gerar problemas de *outliers* em regiões de espaços diferentes. Assim, foram desenvolvidos outros algoritmos com base a detecção local, que consideram que não há um número exato de mecanismos que geram os dados, e também analisam o comportamento da vizinhança ao redor.

O algoritmo LOF de (BREUNIG et al, 2000) é um algoritmo de detecção local que busca por *outliers* considerando a densidade do ponto ***p*** e a densidade de seus k-vizinhos. Este método compara a densidade local de uma instância com a

densidade dos seus vizinhos, dado que a densidade é inversamente proporcional a média das distâncias dos k-vizinhos mais próximos.

A Figura 1, representa a distância de alcance, no caso o ponto p_1 estaria dentro da distância, já o ponto p_2 não seria um k vizinho mais próximo.

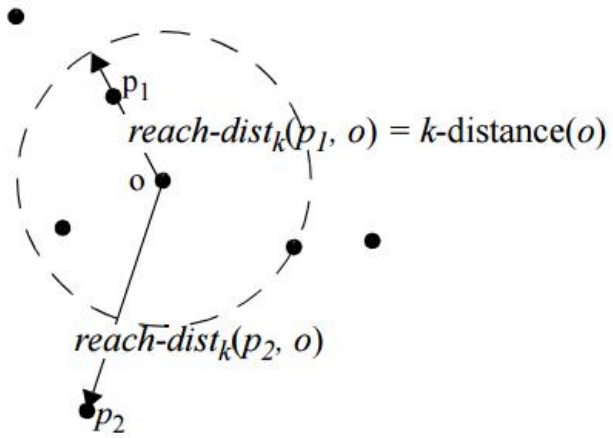

Figura 1. Distância de alcance LOF

O cálculo a densidade da vizinhança de um ponto ***p*** é representada pela Equação 2.4.1, definida pelo inverso da média das distâncias de alcançabilidade entre ***p*** e seus vizinhos. Quanto maior o valor de $lrd_k(p)$ mais densa é a região que se encontra ***p***. O *score* final é representado pela Equação 2.4.2, que calcula a média das razões entre a densidade de cada vizinho mais próximo de ***p*** em relação a densidade de ***p***.

$$lrd_k(p) = 1/\left(\frac{\sum_{o \in N_{k\_dist}(p)} reach\_dist_k(p,o)}{|N_{k\_dist}(p)|}\right). \quad (2.4.1)$$

$$LOF_k(p) = \frac{\sum_{o \in N_{k\_dist}(p)} \frac{lrd_k(o)}{lrd_k(p)}}{|N_{k\_dist}(p)|}. \quad (2.4.2)$$

O resultado do *score* demonstra que as observações normais têm um valor LOF próximo a 1, e as anomalias um valor maior que 1.

2.5 Conclusão

A detecção de anomalias com o algoritmo LOF está sendo utilizada amplamente em diversas pesquisas, como na detecção de intrusão de redes, detecção de *outliers* em dados geográficos, mostrando um ótimo desempenho.

O conceito sobre a detecção de anomalias, seus tipos e técnicas são fundamentais para o entendimento do trabalho proposto. Visto que os sistemas operacionais possuem uma grande quantidade de informações que podem ser trabalhadas junto à mineração de dados, com o objetivo de contribuir para o seu gerenciamento.

Os detectores baseados em anomalias identificam comportamentos não usuais, podendo ser aplicados em estudos em sistemas operacionais. Pode ser visto de duas formas, na identificação de pressupostos que ataques, ou de processos que podem danificar o comportamento do sistema.

3 PROCESSOS

Este capítulo apresenta sobre o conceito e características de processos do Sistema Operacional Linux.

3.1 Introdução

Um processo é definido com a abstração de um programa, ou seja, é a unidade que representa um programa no qual o sistema operacional trabalha para realizar o gerenciamento de tarefas (TANEMBAUM, 2001).

Os processos possuem um ciclo de vida, e para que possam ser executados de forma eficiente, o escalonador de processos é o responsável pelo seu gerenciamento, utilizando um conjunto de regras para identificar a ordem em que os processos sejam executados pelo núcleo do sistema operacional.

Os processos são separados de acordo sua responsabilidade, pois caso um processo falhe não causará danos aos outros processos. Cada processo possui seu próprio endereço virtual, e é executado individualmente, não sendo capaz de interagir diretamente com outro processo, exceto por mecanismos do núcleo do sistema operacional.

3.2 Estados dos processos

Durante o ciclo de vida dos processos, estes podem estar em constante mudança, permitindo permanecer em vários estados durante a execução de um programa, sendo que cada processo pode encontrar-se em apenas um estado num determinado momento.

Caso um processo esteja em execução, e o processador necessite executar um processo de maior prioridade, o que está executando é suspenso temporariamente, e depois retorna à execução após um processo prioritário. A Figura 2 (BOVETI e CESATI, 2005) apresenta os estados dos processos no Linux:

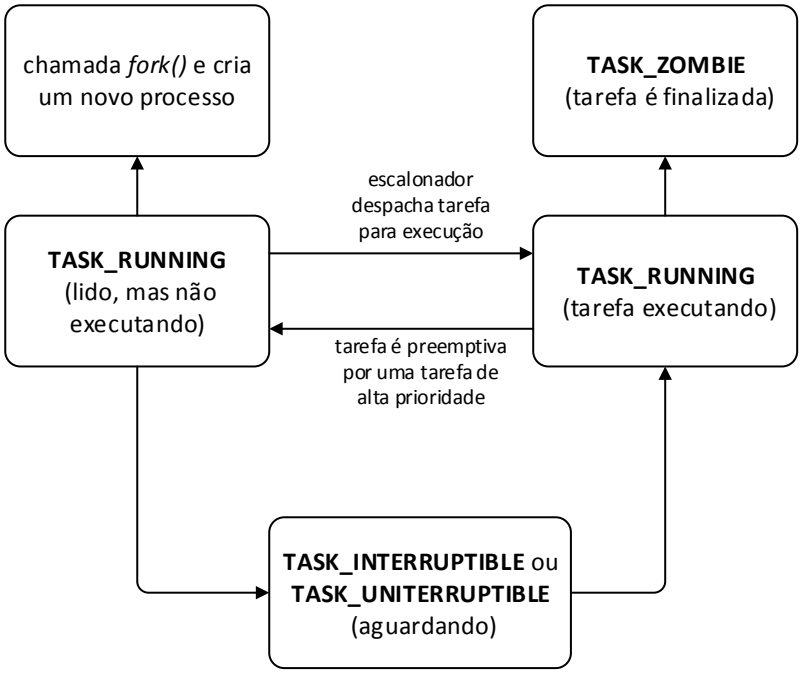

Figura 2. Diagrama de estados dos processos

3.3 Classificação de processos

Os processos são classificados de acordo com sua necessidade de processamento. Alguns processos gastam mais tempo esperando operações de entrada/saída *(I/O bound),* e outros necessitam de um longo tempo de CPU *(CPU bound)*. Nesse sentido, os processos são classificados em três classes (TANEMBAUM, 2001):

- Processos batch: os processos *batch* não necessitam de intervenções com os usuários, e muitas vezes podem estar executando em segundo plano. Um programa possui um conjunto de dados de entrada, é realizado todo o processamento e após produz um conjunto de dados de saída, como exemplo programas de cálculos numéricos, complicações, backups, etc.
- Processos interativos: são tarefas orientadas a entrada/saída que interagem diretamente com o usuário, geralmente possuem uma interface gráfica, e precisam de um tempo de resposta rápida do sistema operacional. Exemplos: editores de texto, navegadores Web, clientes de e-mail, entre outros.

- Processos *daemons*: são processos inicializados no *boot* do sistema, que executam continuamente enquanto o sistema estiver ativo, e esperam em segundo plano até que um serviço seja requerido.

Com o objetivo de auxiliar o escalonador de processos na definição das prioridades de execução dos processos e melhorar o tempo de respostas às tarefas interativas, proporcionou uma outra abordagem de classificação de processos (ARAUJO et. al, 2011), conforme:

- A (Aplicações interativas): todos os tipos de processos interativos, como os editores de texto, navegadores Web, clientes de e-mail, e outros *(a.out, chrome, codeblocks, evince, fileroller, firefox, gcalctool, geany, gedit, gnome-terminal, gnotravex, mousepad, nautilus, npviewer, pcmanfm, scalc, soffice, swritter, thunar, tomboy, wireshark, xterm).*
- D (Daemons): processos que executam em segundo plano e estão prontos para receber instruções *(atd, avahi-daemon, console-kit-daemon, crond, cupsd, dbus-daemon, dbus-launch, devkit-disks-daemon, devkit-power-daemon, evolution-data-daemon, failban-server, gam-server, gconfd, gnome-keyring-daemon, gnome-settings-daemon, gvfsd, hald, ibus-daemon, ibus-gconf, ibus-x, im-settings-daemon, in.talkd, init, irqbalance, lxde-settings-daemon, menu-cached, mrtg, notification-daemon, pcscd, polkitd, rsyslogd, sendmail, udevd, xfconfd, xfdesktop, xfsettingsd, xinetd).*
- F (Funcionalidades de desktop): processos que realizam tarefas de apoio ao ambiente de desktop gráfico, como processos de configuração ou framework de componentes *(bonobo-activati, canberra-gtk-pl, clock-applet, exo-helper, fih-applet, gdm-binary, gmd-sessionwork, gdm-simple-greeter, gdm-simple-slave, gnome-panel, gnome-screensaver, gnome-session, gweather-applet, lxpanel, lxsession, multiload-applet, notification-area, openbox, trashapplet, wnck-applet, xfce-panel, xfce-session, xfce-menu-plug, xfwm, xorg, xvnc).*
- N (network): processos envolvidos com a comunicação de rede *(httpd, sftp-server, snmpd, sshd).*
- C (Comandos de texto): processos de comandos simples de terminal em modo texto *(awk, bash, cb_console_runn, ck-xinit-session, consolerhelper, man, mingetty, ping, python, runmozilla, screen,sh, ssh, ssh-agent, tail, talk, top, uml_mconsole, uml_switch, vim, wish, java, sudo).*
- K (Kernel threads): threads internas do núcleo do sistema operacional *(aio0, aio1, asyncmgr, ata0, ata1, ata_aux, bdi-default, bluetooth, cpuset, crypto0, crypto1, edac-poller, events0, events1, ext4-dio-unwrit, flush, jbd2sda1-8,*

jbd2sda2-8, kacpi_hotplug, kacpi_notify, kacpid, kauditd, kblockd0, kblockd1, khelper, khubd, khungtaskd, kintegrityd0, kintegrityd1, kjournald, kmpath_handlerd, kmpathd0, kmpathd1, kpsmoused, kseriod, ksmd, ksnapd, ksoftirqd0, ksoftirqd1, kstriped, ksuspend_usbd, kswapd0, kswapd1, kthreadd, mpt0, mpt_poll_0, netns, pm, scsi_eh_0, scsi_eh_1, scsi_eh_2, scsi_eh_3, sync_supers, usbhid_resumer).

- O (Outros): processos que não se enquadram nos demais grupos *(xOrg, cupsd, Xvnc, fail, asyncmgr, linux, kblockd0, fail2ban-server).*

3.4 Sistemas de arquivos /proc

No núcleo do GNU/Linux as informações dos processos, como uso de memória, processador, arquivos abertos, estão no sistema de arquivos */proc.* O primeiro diretório */proc* foi implementado em 1984, sendo projetado para substituir a chamada *ptrace* (um processo pode controlar outro processo, manipulando seus descritores de arquivo, memória, registradores). De acordo com (FAULKENER e GOMES, 1991), em 1991 surgiu a versão definitiva do */proc*, onde os arquivos já suportavam chamadas *read(), write() e ioctl(),* com 37 *ioctl()* no total, o que permitia o controle básico de processos, como *signals/fault/syscall*, manipulação de registros e informações de status.

Isso possibilitou a criação de uma poderosa base para construir ferramentas, como a *ps*, sem ter a necessidade de realizar chamadas de sistema especializadas. A chamada de sistema *ptrace* é substituída pelo sistema de arquivos /proc, onde da mesma forma, um programa pode ler ou escrever dados diretamente no endereço do processo através do descritor de arquivo correspondente no */proc.*

Segundo (SHALOM, 2010), a maioria dos sistemas operacionais baseados em Unix possuem um sistema de arquivos */proc*, atuando como uma interface para estrutura de dados internas do núcleo e possibilitando a alteração de parâmetros do núcleo em tempo de execução. O sistema de arquivos */proc* está estrutura conforme a Figura 3, e possui as seguintes características:

• visualização de informações de estatísticas;

• visualização de informações de hardware;

• alteração de parâmetros em tempo de execução;

• visualização e modificação dos parâmetros de rede e host;

• visualização de informações de desempenho da memória.

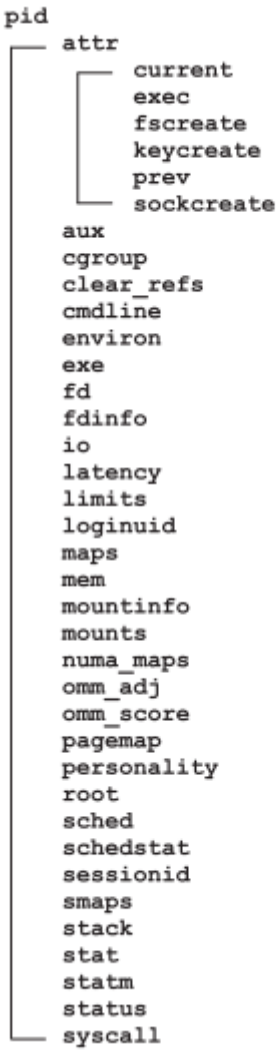

Figura 3. Árvore de estrutura /proc.

3.5 Conclusão

Os processos são a base para o funcionamento do Sistema Operacional, entender e interpretar as ações dos processos é importante para compreender o que está sendo executado, ou até mesmo para identificar alguma falha que poderá causar algum dano ao sistema.

Todas as informações dos processos no Linux estão contidas no diretório */proc,* o que possibilita um grande recurso para a extração de informações a serem utilizadas em mecanismos segurança, ferramentas de administração de sistema, estatísticas, ou até mesmo de monitoramento do sistema.

4 METODOLOGIA

A metodologia proposta nesse trabalho é de caráter quantitativo, e visa utilizar a aprendizagem não supervisionada para a detecção de anomalias em grupos de processos já definidos, com o objetivo de identificar processos que não apresentam o comportamento semelhante aos do mesmo grupo. A construção do modelo é representada pela Figura 4, que engloba as seguintes atividades:

1. Extração dos atributos: coletar informações dos processos presentes em um sistema Linux, gerando a base de dados para a análise de *Outliers*.

2. Seleção de atributos: reduzir a dimensionalidade do vetor de atributos através de algoritmos de seleção de atributos.

3. Detecção de anomalias: identificar anomalias na base de dados de processos através do algoritmo LOF.

4. Análise dos resultados: verificar os resultados do algoritmo de detecção de anomalias, identificando os processos e atributos que não se enquadram como normais.

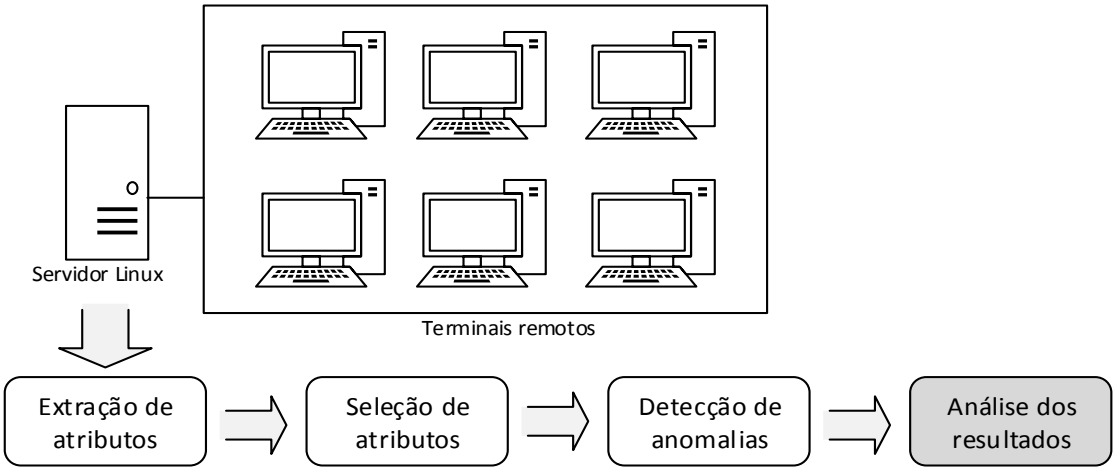

Figura 4. Processo de detecção de anomalias

4.1 Extração e seleção de atributos

No processo de extração e seleção de atributos foram utilizadas as bases de dados criadas da pesquisa de Classificação Automática de Processos em um Sistema operacional de Uso Geral (ARAUJO et al, 2011). A base de dados original foi formada pela extração de atributos que coletou informações de um servidor com a distribuição Linux Fedora Core 12, utilizado pelos alunos de graduação de uma instituição de ensino como terminais remotos SSH/VNC. Esse ambiente possui a frequência de mais de 10 usuários, podendo chegar a 30 usuários conectados simultaneamente para realização de atividades acadêmicas.

Essa fonte de dados contempla uma diversidade de processos como aplicações interativas, programas de cálculos, tratamento em *batch*, e um grande número de *daemons* e outros processos. A base completa foi composta por 97.391 amostras distintas, contendo 108 atributos cada, sendo coletadas as informações do diretório /proc do servidor Linux, duas vezes por dia, por 185 dias.

No trabalho de Geral (ARAUJO et al, 2011), a base completa foi classificada, utilizando o algoritmo de agrupamento DBScan que permitiu identificar 7 grupos distintos de processos, sendo observado que sempre os processos relativos a threads do núcleo do sistema operacional se mantém isolados, e que os demais tipos de processos, como *batches, interativos e daemons*, na maioria das vezes se encontravam isolados em grupos distintos. Assim, a partir do agrupamento dos processos, os dados foram submetidos a algoritmos de seleção de atributos.

O papel da seleção de atributos pode influenciar no resultado final da etapa de detecção de anomalias. De acordo com (KOHAVI e JOHN, 1997), os algoritmos de abordagem filtro tem como objetivo selecionar um subconjunto de atributo que preserve as informações relevantes do conjunto de atributos. Assim, no trabalho de (ARAUJO et al, 2011), para selecionar o melhor subconjunto foram utilizados os algoritmos de Busca Genética (BG), *Ranker* (RK), *Rank Search*(RS) e *Best First* (BF), e como procedimento de avaliação foram escolhidos os métodos *Information Gain* e *CFS*.

Os atributos que compõe cada base são descritos de acordo:

- Base BG : formada por 40 atributos selecionados pelo método Busca Genética, com avaliação por CFS: *fdinfo:openfiles, io:rchar, oom_score, sched:avg_atom, sched:prio, sched:se.iowait_sum, sched:se.load.weight, sched:se.nr_forced2_migrations, sched:se.nr_wakeups, sched:se.nr_wakeups_sync, sched:se.sleep_start, sched:se.wait_start, stat:cstime, stat:endcode, stat:flags, stat:num_threads, stat:pgrp, stat:pid, stat:priority, stat:rss, stat:rsslim, stat:rt_priority, stat:sigcatch, stat:sigignore, stat:startcode, stat:starttime, stat:state, stat:tpgid, stat:utime, stat:vsize, stat:wchan, statm:resident, statm:share, statm:size, statm:text, status:TracerPid, status:VmHWM, status:VmLib, status:VmPeak, status:VmStk.*

- Base RS : formada por 19 atributos selecionados pelo método Rank Search, com avaliação por CFS: *sched:se.nr_failed_migrations_affine, stat:endcode, stat:flags, stat:kstkeip, stat:sigcatch, stat:sigignore, stat:startcode, stat:state, stat:tgpid, stat:tty_nr, stat:vsize, stat:wchan, statm:size, statm:text, status:TracerPid, status:VmExe, status:VmLib, status:VmPeak, status:VmSize.*

- Base RK : formada por 10 atributos selecionados pelo método Ranker, com avaliação por Information Gain: *oom_score, stat:endcode, stat:vsize, statm:size, statm:text, status:VmData, status:VmExe, status:VmLib, status:VmPeak, status:VmSize.*

- Base BF : formada por 9 atributos selecionados pelo método Best First, com avaliação por CFS*: stat:endcode, stat:ppid, stat:sigignore, stat:startcode, stat:tty_nr, stat:vsize, stat:wchan, statm:text, status:VmLib.*

Esses subconjuntos de dados foram submetidos ao algoritmo LOF de detecção de anomalias, conforme os experimentos representados na Figura 5.

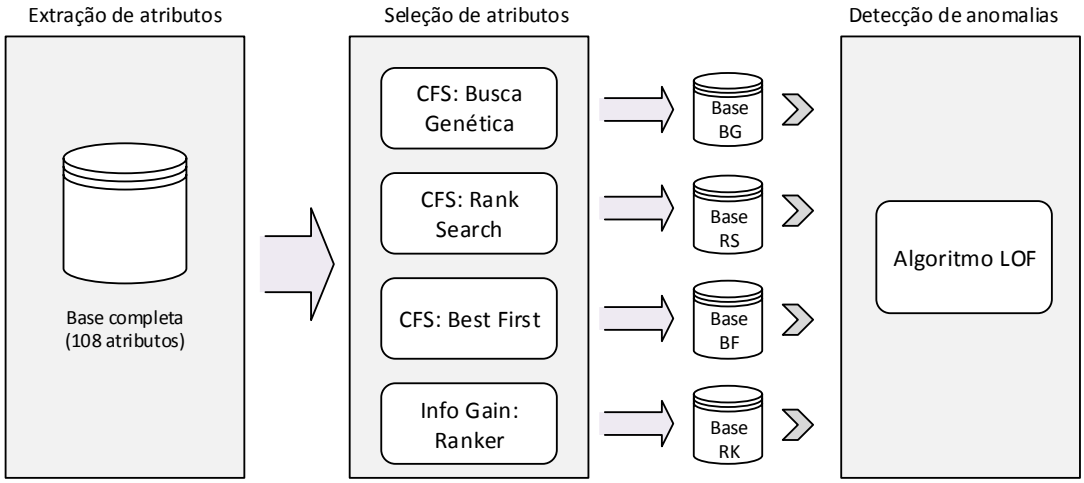

Figura 5. Experimentos realizados Algoritmo LOF

4.2 Detecção de anomalias e experimentos realizados

A detecção de anomalias tem como objetivo identificar dados que não estão no padrão do conjunto total. O algoritmo LOF é bastante utilizado em detecção de anomalias para diversas áreas, conforme descrito no Capítulo 3.

Devido à complexidade do algoritmo, e o tempo de processamento, para os experimentos foram utilizadas 10.000 amostras distintas de cada base de dados (Base BG, Base RS, Base BF e Base RK), essas bases foram extraídas pela indução dos algoritmos de seleção de atributos realizada na pesquisa de (ARAUJO et al, 2011).

Cada base de dados foi induzida ao algoritmo de detecção de anomalias LOF da ferramenta Weka, onde foram submetidos os arquivos ARFF específicos. A Figura 6 representa o formato do arquivo ARFF.

```
@attribute stat.ppid numeric
@attribute stat.tty_nr numeric
@attribute stat.vsize numeric
@attribute stat.startcode numeric
@attribute stat.endcode numeric
@attribute stat.sigignore numeric
@attribute stat.wchan numeric
@attribute statm.text numeric
@attribute status.VmLib numeric
@attribute class {A,D,F,N,C,K,O}

@data
9677,0,137314304,4194304,4264604,4096,1.8446744071580074E19,18,4320,F
2,0,0,0,0,2147483647,1.8446744071579304E19,0,0,K
12620,34831,868352,134512640,138354316,134217729,1.8446744071579496E19,938,1.844674407370955E19,C
13732,34821,868352,134512640,138354316,134217729,1.8446744071579496E19,938,1.844674407370955E19,C
13732,34821,974848,134512640,138354316,134217729,1.8446744071579496E19,938,1.844674407370955E19,C
13732,34821,1146880,134512640,138354316,134217729,1.8446744071579496E19,938,1.844674407370955E19,C
14769,34833,69603328,134512640,138354316,1,1.8446744071579337E19,938,1.844674407370955E19,C
14771,34833,69603328,134512640,138354316,134217729,1.8446744071583224E19,938,1.844674407370955E19,C
14771,34833,696320,134512640,138354316,134217729,1.8446744071579496E19,938,1.8446744073709548E19,C
14771,34833,1138688,134512640,138354316,134217729,1.8446744071579496E19,938,1.844674407370955E19,C
14771,34833,1466368,134512640,138354316,134217729,1.8446744071579496E19,938,1.844674407370955E19,C
14771,34833,1433600,134512640,138354316,134217729,1.8446744071579496E19,938,1.844674407370955E19,C
17342,0,339103744,4194304,4754764,4096,1.8446744071580074E19,137,23236,F
18389,0,459603968,4194304,4747836,16781312,1.8446744071580074E19,136,19160,F
```

Figura 6. Formato arquivo ARFF – Weka

Os parâmetros utilizados para o algoritmo LOF foram:

- NNSearch: algoritmo de pesquisa de vizinho mais próximo.
- Debug: pode gerar informações adicionais para o console. Valor = false.
- donotchekCapabilties: os recursos de filtros não são verificados antes do filtro ser construído. Valor = false.
- minPointLowerBounds: limite inferior (minPtsLB) do intervalo para k ao determinar o valor LOF máximo. Valor = 10.
- minPointUpperBounds: limite superior (minPtsUB) do intervalo para k ao determinar o valor LOF máximo. Valor = 40.
- numExecutionSlots: número de slots de execução (threads) a serem utilizados para encontrar as valores LOF. Valor = 1.

Os resultados do algoritmo LOF são apresentados a seguir pelos gráficos para cada base de dados, sendo apresentado o valor LOF, e a identificação do processo considerado como anomalia:

Análise de Outliers - Base BG (40 atributos)

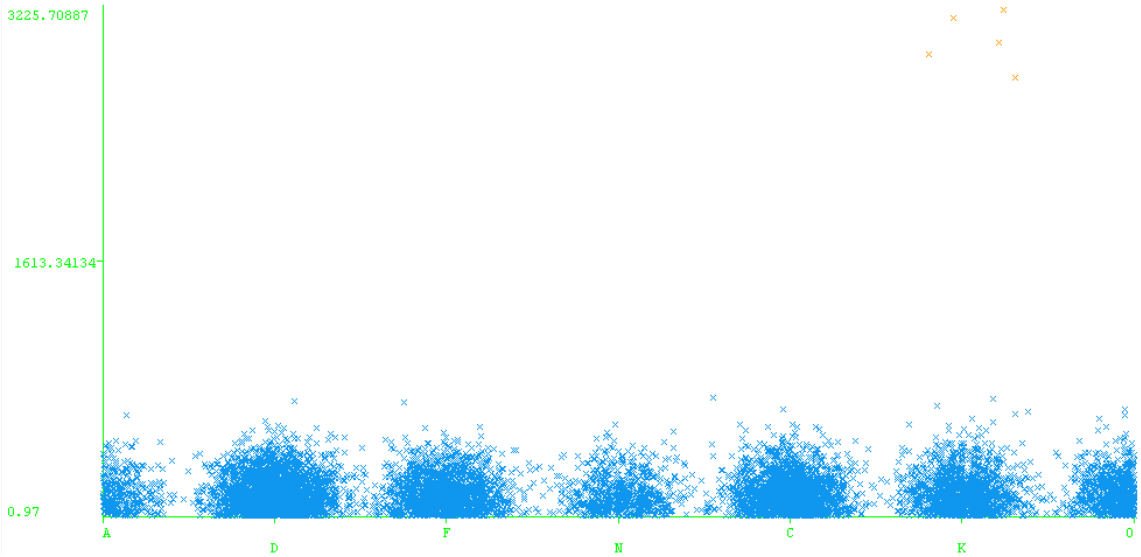

Figura 7. Resultado Algoritmo LOF – Base BG

Tabela 2. Resultado Algoritmo LOF – Base BG

Classe	Instância	LOF	Processo	Descrição
K (kernel threads)	308	3225.708867	*bluetooth*	Responsável pela comunicação entre dispositivos utilizando o protocolo *bluetooth*. O *bluettooth* é designado para transferências de dados em curta distância, como transferência de arquivos, áudios, sistemas de saúde, entre outros.
	4587	3225.708867	*bluetooth*	
	7129	3225.708867	*bluetooth*	
	7386	3225.708867	*bluetooth*	
	8963	3225.708867	*bluetooth*	

Análise de Outliers - Base RS (19 atributos)

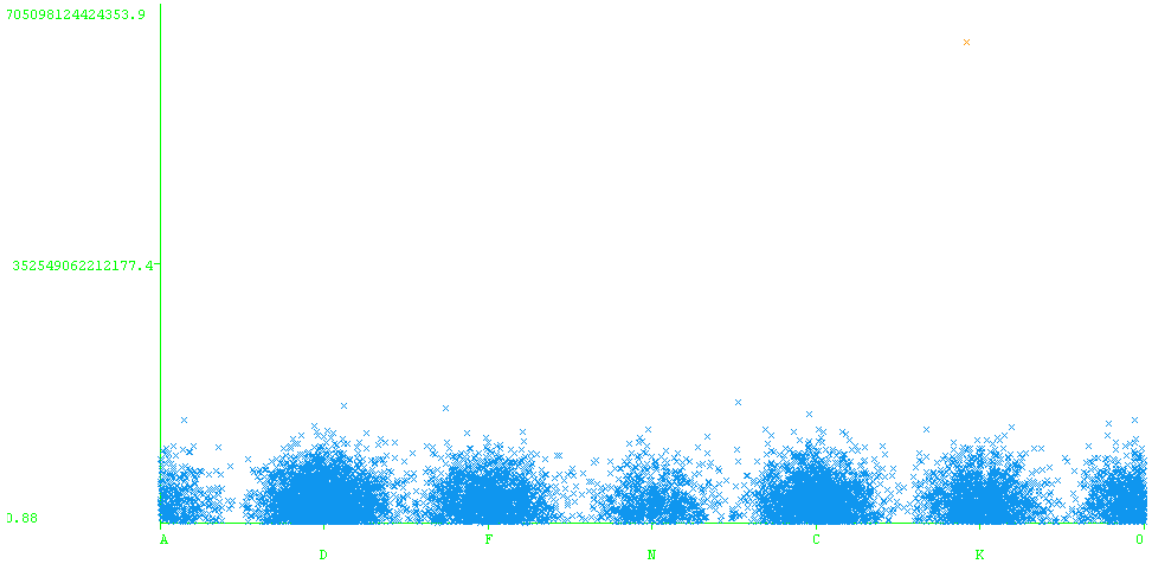

Figura 8. Resultado Algoritmo LOF – Base RS

Tabela 3. Resultado Algoritmo LOF – Base RS

Classe	Instância	LOF	Processo	Descrição
K (kernel threads)	7468	705098124424353.9	jbd2/sda1-8	Processo relacionado ao disco, gerenciador do sistemas de arquivos ext4), realiza a limpeza e gerencia as alterações de disco.

Análise de Outliers - Base RK (10 atributos)

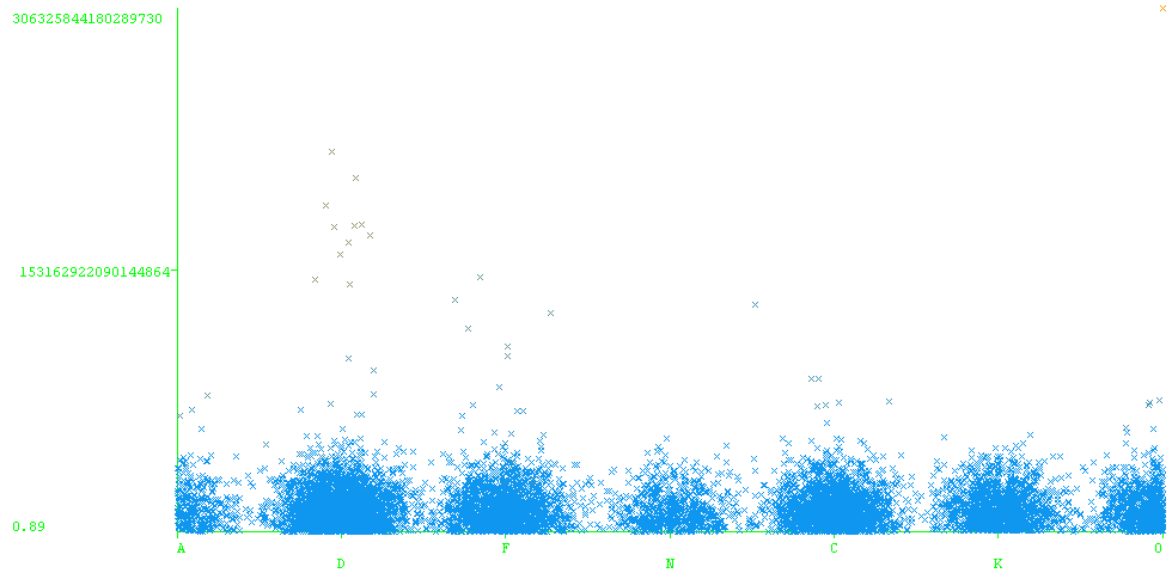

Figura 9. Resultado Algoritmo LOF – Base RK

Tabela 4. Resultado Algoritmo LOF – Base RK

Classe	Instância	LOF	Processo	Descrição
D (daemons)	132	165397203375426368	*pcscd*	O processo pcscd é um deamon do framework *MuscleCard*. É responsável por fazer o gerenciamento de recursos da comunicação com os leitores *smartcards* e *tokens* conectados ao sistema. É inicializado no *boot* do sistema operacional.
	334	165397203375426368	*pcscd*	
	505	165397203375426368	*pcscd*	
	654	165397203375426368	*pcscd*	
	781	165397203375426368	*pcscd*	
	910	165397203375426368	*pcscd*	
	1552	165397203375426368	*pcscd*	
	2455	165397203375426368	*pcscd*	
	3426	165397203375426368	*pcscd*	
	3852	165397203375426368	*pcscd*	
F (funcionalidades de desktop)	2988	112205039419585152	*xfce4-panel*	Xfce é uma área de trabalho gráfica, que utiliza a biblioteca GTK+2 para fazer a interface com o usuário. Esse processo apresenta os programas em execução, menus de painel, relógio, entre outros.
	3395	112205039419585152	*xfce4-panel*	
	3799	112205039419585152	*xfce4-panel*	
C (comandos de texto)	2063	61991242885981808	*man*	Representa o manual do sistema operacional, por ele são formatadas e apressentadas as informações do nome do programa, descrição, suas utilidades e funções.
O (outros)	4674	306325844180289730	*Thunar*	Thunar é um gerenciador de arquivos moderno para o a área de trabalho Xfce. Possui uma interface simples e intuitiva, com a característica de ser rápido.

Análise de Outliers - Base BF (9 atributos)

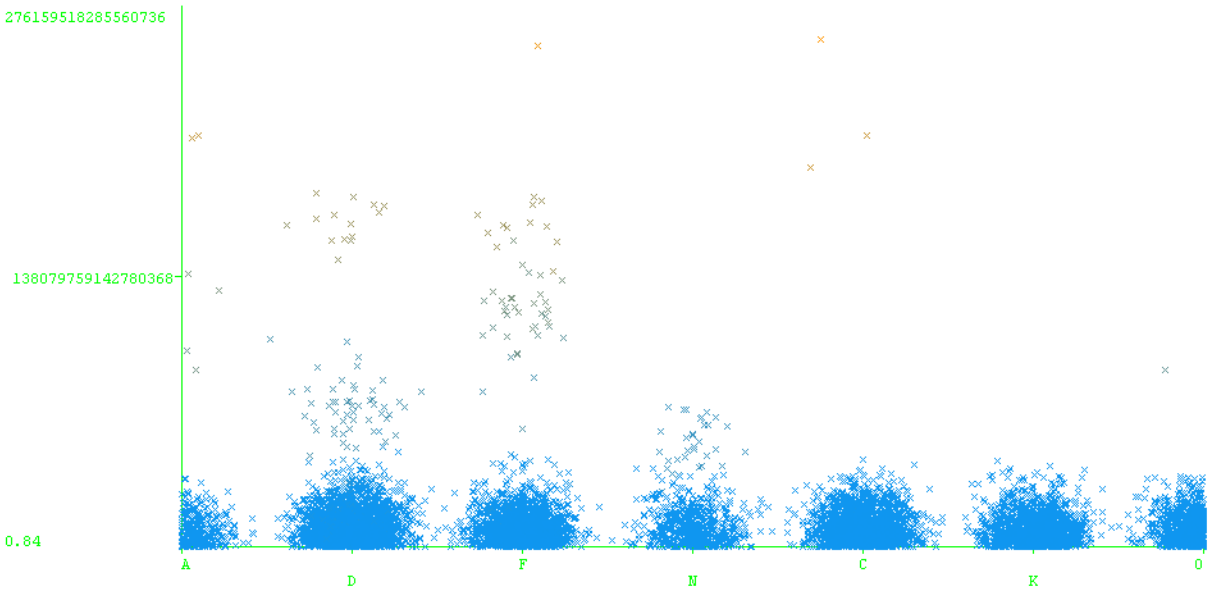

Figura 10. Resultado Algoritmo LOF – Base BF

Tabela 5. Resultado Algoritmo LOF – Base BF

Classe	**Instância**	**LOF**	**Processo**	**Descrição**
A (aplicações interativas)	8951	207555073302160608	*firefox*	Processo responsável pela execução do navegador de Internet Firefox. Considerado um navegador leve, seguro e extensível.
	9330	208353316774646752	*firefox*	
	9477	127286055721248032	*gedit*	GEDIT é o editor de texto do ambiente de trabalho GNOME. Utilizado para editar e criar arquivos textos, visando a simplicidade e facilidade de uso.
D (Daemons)	283	157304628632953952	*lxde-settings-daemons*	LXDE (Lightweight X11 Desktop Environmen) é um ambiente de trabalho leve e rápido, foi projetado utilizar baixos recursos, como menos memória RAM e CPU.
	1393	164375512574474048	*gnome-keyring-daemon*	O GNOME Keyring é um aplicativo *daemon* projetado para cuidar das credenciais de segurança do usuário, como nomes de usuários e senhas. Os dados confidenciais são criptografados e armazenados em um arquivo de *keyring* no diretório inicial do usuário. O *keyring* padrão usa a senha de login para criptografia, portanto os usuários não precisam se lembrar de outra senha.
	1621	164375512574474048	*gnome-keyring-daemon*	
	2513	164375512574474048	*gnome-keyring-daemon*	

	2982	164375512574474048	*gnome-keyring-daemon*	
	2995	164375512574474048	*gnome-keyring-daemon*	
	3389	164375512574474048	*gnome-keyring-daemon*	
	3418	164375512574474048	*gnome-keyring-daemon*	
	3817	164375512574474048	*gnome-keyring-daemon*	
	3845	164375512574474048	*gnome-keyring-daemon*	
	6309	152869226694260896	*gnome-keyring-daemon*	
	7958	152869226694260896	*gnome-keyring-daemon*	
	8410	152869226694260896	*gnome-keyring-daemon*	
F (funcionalidades de desktop)	4580	126714732280387664	*gdm-session-working*	É um serviço *backend* de autenticação PAM de ambos os processos de login e de *screensaver*. Pode compartilhar o código PAM e atender ao requisito do *Trusted Path.*
	278	257377893801090048	*lxsession*	O LXSession é o gerenciador de sessão padrão usado pelo LXDE. Inicia um conjunto de aplicativos automaticamente e configura o ambiente de área de trabalho.
	4024	165484177451243200	*bonobo-activation-server*	Processo do GNOME para o rastreamento de componentes instalados em conjunto com sua biblioteca do cliente. Executa os

	4887	165484177451243200	*bonobo-activation-server*	componentes do ambiente desde o primeiro processo iniciado no servidor.
	7208	165484177451243200	*bonobo-activation-server*	
	7661	165484177451243200	*bonobo-activation-server*	
	7808	165484177451243200	*bonobo-activation-server*	
	8565	165484177451243200	*bonobo-activation-server*	
	8778	165484177451243200	*bonobo-activation-server*	
	9122	165484177451243200	*bonobo-activation-server*	
	9428	165484177451243200	*bonobo-activation-server*	
	9630	165484177451243200	*bonobo-activation-server*	
	9843	165484177451243200	*bonobo-activation-server*	
C (comandos de texto)	1156	216352197548707680	*copia-proc.sh*	Programa *shell* criado para capturar as informações dos processos do diretório /proc, e armazená-las num arquivo.
	4531	214334905615889216	*copia-proc.sh*	
	6730	276159518285560736	*copia-proc.sh*	

4.3 Resultados obtidos

Os resultados mensurados nesse trabalho foram em relação à quantidade de instâncias anômalas, classes, processos e tempo de execução do algoritmo LOF. A Tabela 6 apresenta um comparativo dos resultados obtidos em função da indução de cada base de dados ao algoritmo LOF, mostrando que as técnicas de seleção de atributos podem influenciar no resultado.

Tabela 6. Comparação resultados algoritmo LOF

Bases	**Atributos**	**Instâncias anômalas**	**Classes anômalas**	**Tempo de execução**
Base BG	40	5	K (kernel threads)	108 min
Base RS	19	1	K (kernel threads)	60 min
Base RK	10	16	D (Daemons), F (funcionalidades de desktop) e C (comandos de texto), O (outros).	3 min
Base BF	9	32	A (aplicações interativas), D (Daemons), F (funcionalidades de desktop) e C (comandos de texto).	44 min

Os processos identificados como anomalias demonstradas nesse trabalho referem-se aos que apresentaram acima da média indicada pela ferramenta Weka, não foi realizada uma análise mais profunda em relação a todo resultado do algoritmo LOF com a correlação entre as bases de dados.

Nesse sentido, constatou-se também uma relação entre o tamanho da base de dados e o tempo de execução do algoritmo LOF, ou seja, a dimensionalidade dos atributos é reduzida, assim o tempo de processamento também é reduzido, com execução a Base RK que apresentou um tempo consideravelmente menor, isso se deve ao tamanho dos valores do vetor de atributos.

Em relação as instâncias anômalas identificadas, é possível considerar que tanto a Base RK e a Base BF apresentaram um maior número de processos identificados como anomalias, em comparação do restante do conjunto de dados. Para essas duas bases, entre as classes identificadas foram: A (aplicações interativas), D (*daemons*), F (funcionalidades de desktop), C (comandos de texto), O (outros).

Observou-se que os atributos que influenciaram no cálculo do resultado LOF para a base RK é em relação ao uso de memória pelo processo (statm:text: tamanho da memória utilizada pelo código do processo, stat:vsize: memória virtual em uso pelo processo). Este é um resultado aceitável visto que muitos problemas de desempenho dos sistemas operacionais são relacionados à alocação de memória, o que pode afetar diretamente os processos interativos, como é o caso do processo *firefox*, que em muitas vezes pode ocorrer o travamento, pois utiliza mais memória RAM do que deveria.

Já em relação a base BF o resultado foi influenciado pelo atributo *oom_score,* também relacionado à memória virtual do Linux, mostra uma pontuação dos processos que estejam consumindo muita memória em situações de saturação de memória RAM.

Pode se concluir a partir dos experimentos realizados, que as informações relacionadas à memória do processo, influenciam para a análise de seu comportamento, possibilitando a identificação de anomalias, e que nem todas as informações de processos são essenciais para a detecção de anomalias.

5 CONSIDERAÇÔES FINAIS

Este trabalho mostrou uma abordagem do algoritmo de mineração de dados - LOF para detecção de anomalias em processos do sistema operacional Linux. O modelo de detecção de anomalias proposto buscou avaliar diferentes conjuntos de dados, apresentando que o papel da seleção de atributos é fundamental para o desempenho do resultado final, pois existem dados redundantes ou irrelevantes.

O processo de construção do modelo de detecção de anomalias com o algoritmo LOF, partindo da necessidade do entendimento das características dos processos, compõe como uma base fundamental para a análise comparativa das diversas bases de dados.

Em termos de viabilidade de aplicação e aprimoramento de conhecimento sobre os processos, este estudo pode apresentar boas perspectivas, pois é uma nova forma de identificar uma anomalia, o que poderá servir como um mecanismo de auxílio para melhorar o desempenho de um sistema operacional.

Através desse estudo pode ser contemplado outros trabalhos futuros, como extensão da identificação das anomalias associadas ao comportamento dinâmico dos processos, a verificação de detecção de anomalias em sistemas operacionais de dispositivos móveis e identificação de anomalias no contexto de segurança digital, para detecção de intrusões.

REFERÊNCIAS

AAS, J. **Understanding the linux 2.6.8.1 CPU scheduler**. SGI, 22:5, 2005.

APPLE. **Kernel Programming Guide - Drivers, Kernel and Hardware**, 2011. Disponível em: http://developer.apple.com/library/mac/documentation/Darwin/Conceptual/KernelProgramming/KernelProgramming.pdf. Acesso em: Janeiro de 2017.

ARAUJO, Priscila V., MAZIERO, Carlos A., NIEVOLA, Julio C. 2011. **Classificação automática de processos em um sistema operacional de uso geral.** Simpósio Brasileiro de Engenharia de Sistemas Computacionais - SBESC, 1:1–12. Florianópolis, 2011.

BOVET, D., CESATI, M. **Understanding The Linux Kernel**. O'Reilly & Associates Inc., 2005.

BREUNIG, Markus M., KRIEGEL, Hans-Peter, NG, Raymond T., SANDER, Jörg. **LOF: Identifying Density-Based Local Outliers.** Proc. ACM SIGMOD 2000 Int. Conf. On Management of Data, Dalles, TX, 2000.

CHANDOLA, Waruna, BANERJEE, Arindam, KUMAR, Vinpim. **Anomaly Detection: A Survey.** ACM Computing Surveys, 2009.

CHEN, M.S., HAN, J., UY, P.S. **Data Mining: An Overview from a Database Perspective.** IEEE Transactions on Knowledge and Data Engineering, 8(6):866-883, 1996.

FAULKER, R.,GOMES, R. **The Process File System and Process Model in UNIX System V.** In USENIX Conference Proceedings, 1991.

HAN, J., KAMBER, M. **Data Mining: Concepts and Techniques** (The Morgan Kaufmann Series in Data Management Systems). Morgan Kaufmann, 1st edition, 2000.

LOVE, R. **Interactive Kernel Performance in Desktop and real-time applications. Linus Symposium**, pages 285 -296, 2003.

SHALOM, H. **Creating and using proc files**. http://www.rt-embedded.com/blog/archives/creating-and-using-proc-files/, 2010.

TANENBAUM, A.. **Modern Operating Systems**. Prentice Hall PTR**.** 2001.

APÊNDICE(S)

APÊNDICE A – Amostra da leitura do arquivo ARFF submetido a ferramenta WEKA – Base BG

@attribute stat.pid numeric
@attribute stat.state {R,S,D,Z,T,W}
@attribute stat.pgrp numeric
@attribute stat.tpgid numeric
@attribute stat.flags numeric
@attribute stat.utime numeric
@attribute stat.cstime numeric
@attribute stat.priority numeric
@attribute stat.num_threads numeric
@attribute stat.starttime numeric
@attribute stat.vsize numeric
@attribute stat.rss numeric
@attribute stat.rsslim numeric
@attribute stat.startcode numeric
@attribute stat.endcode numeric
@attribute stat.sigignore numeric
@attribute stat.sigcatch numeric
@attribute stat.wchan numeric
@attribute stat.rt_priority numeric
@attribute statm.size numeric
@attribute statm.resident numeric
@attribute statm.share numeric
@attribute statm.text numeric
@attribute status.TracerPid numeric
@attribute status.VmPeak numeric
@attribute status.VmHWM numeric
@attribute status.VmStk numeric
@attribute status.VmLib numeric
@attribute io.rchar numeric
@attribute oom_score numeric
@attribute fdinfo.openfiles numeric
@attribute sched.se.wait_start numeric
@attribute sched.se.sleep_start numeric
@attribute sched.se.iowait_sum numeric
@attribute sched.se.nr_forced2_migrations numeric
@attribute sched.se.nr_wakeups numeric
@attribute sched.se.nr_wakeups_sync numeric
@attribute sched.avg_atom numeric
@attribute sched.se.load.weight numeric
@attribute sched.prio numeric
@attribute class {A,D,F,N,C,K,O}

11350,S,25752,1,4202496,0,0,20,1,135404248,137314304,332,1.8446744073709552E19,4194304,4264604,4096,18155,1.8446744071580074E19,0,33524,332,331,18,0,134096,1752,84,4320,18035,523,6,0,1354042492.451069,29626.472302,0,1,1,0.430475, 1024, 120,F
13,S,0,1,2149613632,0,0,20,1,9,0,0,1.8446744073709552E19,0,0,2147483647,0,1.8446744071579304E19,0,0,0,0,0,0,0,0,0,0,0, 0,0,99.903114,0,0,1,0,0.002808,1024,120,K
13535,T,12618,12618,4202560,0,0,30,1,175236581,868352,204,1048576000,134512640,138354316,134217729,301991130,1.8446744071579496E19,0,212,204,204,938,12620,1076,816,0,1.844674407370955E19,0,212,22,0,0,2107.758636,0,716,0,0.011069,110,130,C
14616,T,13730,13730,4202560,0,0,30,1,175249053,868352,205,1048576000,134512640,138354316,134217729,301991130,1.8446744071579496E19,0,212,205,205,938,13732,1072,820,0,1.844674407370955E19,0,212,22,0,0,2092.613791,0,716,0,0.011235,110,130,C
14665,T,13730,13730,4202560,0,0,30,1,175249131,974848,200,1048576000,134512640,138354316,134217729,301991130,1.8446744071579496E19,0,238,200,200,938,13732,1800,800,0,1.844674407370955E19,0,238,22,0,0,2092.613791,0,674,0,0.01157,110,130,C
14679,T,13730,13730,4202560,1,0,30,1,175249157,1146880,164,1048576000,134512640,138354316,134217729,301991130,1.8446744071579496E19,0,280,164,164,938,13732,1120,976,0,1.844674407370955E19,0,280,22,0,0,2092.613791,6,8989,0,0.012696,110,130,C
14771,S,14769,14769,4194304,1885,652,30,1,175258570,69603328,6888,1048576000,134512640,138354316,1,436225242,1.8446744071579337E19,0,16993,6888,6788,938,0,67976,27556,84,1.844674407370955E19,1228447,6000,22,0,1773375744.313986,4898.537188,15986,2767005,588961,0.02941,110,130,C

14779,S,14769,14769,4194368,0,0,30,1,175258613,69603328,6888,1048576000,134512640,138354316,134217729,3020075
14,1.8446744071583224E19,0,16993,6888,6788,938,0,67976,27556,84,1.844674407370955E19,13623340,16992,22,0,17733
70363.656787,2384.82118,159,3738,3636,0.057327,110,130,C
14781,T,14769,14769,4202560,6,0,30,1,175258642,696320,160,1048576000,134512640,138354316,134217729,301991130,
1.8446744071579496E19,0,170,160,160,938,14771,67984,2412,0,1.8446744073709548E19,0,170,22,0,0,2084.95716,67,2199
7,0,0.01496,110,130,C
15704,T,14769,14769,4202560,1,0,30,1,175259999,1138688,163,1048576000,134512640,138354316,134217729,301991130,
1.8446744071579496E19,0,278,163,163,938,14771,1112,968,0,1.844674407370955E19,0,278,22,0,0,2084.95716,5,8942,0,
0.012069,110,130,C
15723,T,14769,14769,4202560,0,0,30,1,175260314,1466368,299,1048576000,134512640,138354316,134217729,301991130,
1.8446744071579496E19,0,358,299,299,938,14771,1432,1432,0,1.844674407370955E19,0,358,22,0,0,2084.95716,0,2998,0,
0 .01026,110,130,C
15730,T,14769,14769,4202560,0,0,30,1,175260339,1433600,325,1048576000,134512640,138354316,134217729,301991130,
1.8446744071579496E19,0,350,325,325,938,14771,1400,1380,0,1.844674407370955E19,0,350,22,0,0,2084.95716,2,4590,0,
0.014681,110,130,C
17430,S,17342,1,4202496,67,0,30,1,177103510,339103744,3757,1048576000,4194304,4754764,4096,0,1.844674407158007
4E19,0,82789,3757,2624,137,0,335500,15028,88,23236,2313208,165578,30,0,1773408542.231238,4749.645213,24,4165,112
2, 0.255906 ,110,130,F
18399,S,18389,1,4202496,14,0,20,1,175830641,459603968,2317,1.8446744073709552E19,4194304,4747836,16781312,8192
0, 1.8446744071580074E19,0,112208,2317,1700,136,0,571736,9316,84,19160,6946895,112208,20,0,1773389131.565401,
4830.427142,15,1536,1532,0.057162,1024,120,F
19950,T,18584,18584,4202560,0,0,30,1,177137164,598016,116,1048576000,134512640,136885440,1,436208858,1.8446744
071579496E19,0,146,116,100,580,18589,584,464,84,1.844674407370955E19,0,292,22,0,0,4749.645213,0,1611,0,0.006876,1
10,130,C
20056,T,18584,18584,4202560,0,0,30,1,177137613,1658880,316,1048576000,134512640,136885440,1,436208858,1.844674
4071579496E19,0,405,316,300,580,18589,1620,1500,84,1.8446744073709552E19,0,810,22,0,0,4749.645213,0,2973,0,0.0097
46,110,130,C
3641,S,2589,-1,4202560,1248,0,20,1,55419,43208704,81,1.8446744073709552E19,4194304,4376244,0,0,1.84467440715800
74E19,0, 10549,81,60,45,0,42196,604,84,4696,19,21,5,0,1773557086.186143,71424.37305,9,4432499,1,0.053555,1024,
120, D
6,S,0,-1,2216722496,0,0,-100,1,2,0,0,1.8446744073709552E19,0,0,2147483647,0,1.8446744071579177E19,99,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,1,524837,0,0.008518,177522,0,O
831,S,0,-1,2149613632,0,0,20,1,1501,0,0,1.8446744073709552E19,0,0,2147483647,0,1.8446744071580738E19,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,1772795715.928121,232614.548453,11,34633,0,0.079339,1024,120,K
1,S,1,-1,4202752,97,8855978,20,1,1,4157440,182,1.8446744073709552E19,4194304,4328382,4096,671835171,1.84467
44071580074E19,0,1015,182,146,33,0,4076,772,84,1588,934531448715.000064,235372,8,0,1816020445.653894,
4705.19025,177,35776,301,0.397118,1024,120,D

APÊNDICE B – Amostra da leitura do arquivo ARFF submetido a ferramenta WEKA – Base RS

@attribute stat.state {R,S,D,Z,T,W}
@attribute stat.tty_nr numeric
@attribute stat.tpgid numeric
@attribute stat.flags numeric
@attribute stat.vsize numeric
@attribute stat.startcode numeric
@attribute stat.endcode numeric
@attribute stat.kstkeip numeric
@attribute stat.sigignore numeric
@attribute stat.sigcatch numeric
@attribute stat.wchan numeric
@attribute statm.size numeric
@attribute statm.text numeric
@attribute status.TracerPid numeric
@attribute status.VmPeak numeric
@attribute status.VmSize numeric
@attribute status.VmExe numeric
@attribute status.VmLib numeric
@attribute sched.se.nr_failed_migrations_affine numeric
@attribute class {A,D,F,N,C,K,O}

@data
S,0,-1,4202496,137314304,4194304,4264604,239971684696,4096,18155,1.8446744071580074E19,33524,18,0,
134096,134096,72,4320,0,F
S,0,-1,2149613632,0,0,0,0,2147483647,0,1.8446744071579304E19,0,0,0,0,0,0,0,0,K
T,34831,12618,4202560,868352,134512640,138354316,1075120198,134217729,301991130,1.8446744071579496E19,212,93
8,12620,1076,848,3752,1.844674407370955E19,0,C
T,34821,13730,4202560,868352,134512640,138354316,1075120198,134217729,301991130,1.8446744071579496E19,
212,938,13732,1072,848,3752,1.844674407370955E19,0,C
T,34821,13730,4202560,974848,134512640,138354316,1077671096,134217729,301991130,1.8446744071579496E19,238,93
8,13732,1800,952,3752,1.844674407370955E19,0,C

T,34821,13730,4202560,1146880,134512640,138354316,1074705995,134217729,301991130,1.8446744071579496E19,280,938,13732,1120,1120,3752,1.844674407370955E19,0,C
S,34833,14769,4194304,69603328,134512640,138354316,4151464997,1,436225242,1.8446744071579337E19,16993,938,0,67976,67972,3752,1.844674407370955E19,0,C
S,34833,14769,4194368,69603328,134512640,138354316,4151464997,134217729,302007514,1.8446744071583224E19,16993,938,0,67976,67972,3752,1.844674407370955E19,0,C
T,34833,14769,4202560,696320,134512640,138354316,1075090616,134217729,301991130,1.8446744071579496E19,170,938,14771,67984,680,3752,1.8446744073709548E19,0,C
T,34833,14769,4202560,1138688,134512640,138354316,1074705995,134217729,301991130,1.8446744071579496E19,278,938,14771,1112,1112,3752,1.844674407370955E19,0,C
T,34833,14769,4202560,1466368,134512640,138354316,1074794648,134217729,301991130,1.8446744071579496E19,358,938,14771,1432,1432,3752,1.844674407370955E19,0,C
T,34833,14769,4202560,1433600,134512640,138354316,1074789566,134217729,301991130,1.8446744071579496E19,350,938,14771,1400,1400,3752,1.844674407370955E19,0,C
S,0,-1,4202496,339103744,4194304,4754764,239971684696,4096,0,1.8446744071580074E19,82789,137,0,335500,331156,548,23236,0,F
S,0,1,4202496,459603968,4194304,4747836,239971684755,16781312,81920,1.8446744071580074E19,112208,136,0,571736,448832,544,19160,0,F
T,34820,18584,4202560,598016,134512640,136885440,1074666686,1,436208858,1.8446744071579496E19,146,580,18589,584,584,2320,1.844674407370955E19,0,C
T,34820,18584,4202560,1658880,134512640,136885440,1074794648,1,436208858,1.8446744071579496E19,405,580,18589,1620,1620,2320,1.8446744073709552E19,0,C
S,0,-1,4202560,43208704,4194304,4376244,242754605400,0,0,1.8446744071580074E19,10549,45,0,42196,42196,180,4696,0, D
S,0,-1,2216722496,0,0,0,0,2147483647,0,1.8446744071579177E19,0,0,0,0,0,0,0,0,O
S,0,-1,2149613632,0,0,0,0,2147483647,0,1.8446744071580738E19,0,0,0,0,0,0,0,0,K
S,0,-1,4202752,4157440,4194304,4328382,239971693427,4096,671835171,1.8446744071580074E19,1015,33,0,4076,4060,132,1588,0,D

APÊNDICE C – Amostra da leitura do arquivo ARFF submetido a ferramenta WEKA – Base RK

@attribute stat.vsize numeric
@attribute stat.endcode numeric
@attribute statm.size numeric
@attribute statm.text numeric
@attribute status.VmPeak numeric
@attribute status.VmSize numeric
@attribute status.VmData numeric
@attribute status.VmExe numeric
@attribute status.VmLib numeric
@attribute oom_score numeric
@attribute class {A,D,F,N,C,K,O}

@data
137314304,4264604,33524,18,134096,134096,236,72,4320,523,F
0,0,0,0,0,0,0,0,0,0,K
868352,138354316,212,938,1076,848,0,3752,1.844674407370955E19,212,C
868352,138354316,212,938,1072,848,0,3752,1.844674407370955E19,212,C
974848,138354316,238,938,1800,952,0,3752,1.844674407370955E19,238,C
1146880,138354316,280,938,1120,1120,0,3752,1.844674407370955E19,280,C
69603328,138354316,16993,938,67976,67972,324,3752,1.844674407370955E19,6000,C
69603328,138354316,16993,938,67976,67972,324,3752,1.844674407370955E19,16992,C
696320,138354316,170,938,67984,680,0,3752,1.8446744073709548E19,170,C
1138688,138354316,278,938,1112,1112,0,3752,1.844674407370955E19,278,C
1466368,138354316,358,938,1432,1432,0,3752,1.844674407370955E19,358,C
1433600,138354316,350,938,1400,1400,0,3752,1.844674407370955E19,350,C
339103744,4754764,82789,137,335500,331156,4060,548,23236,165578,F
459603968,4747836,112208,136,571736,448832,163888,544,19160,112208,F
598016,136885440,146,580,584,584,32,2320,1.844674407370955E19,292,C
1658880,136885440,405,580,1620,1620,32,2320,1.8446744073709552E19,810,C
43208704,4376244,10549,45,42196,42196,236,180,4696,21,D
0,0,0,0,0,0,0,0,0,0,O
0,0,0,0,0,0,0,0,0,0,K
4157440,4328382,1015,33,4076,4060,176,132,1588,235372,D

APÊNDICE D – Amostra da leitura do arquivo ARFF submetido a ferramenta WEKA – Base BF

@attribute stat.ppid numeric
@attribute stat.tty_nr numeric
@attribute stat.vsize numeric
@attribute stat.startcode numeric
@attribute stat.endcode numeric

@attribute stat.sigignore numeric
@attribute stat.wchan numeric
@attribute statm.text numeric
@attribute status.VmLib numeric
@attribute class {A,D,F,N,C,K,O}

@data
9677,0,137314304,4194304,4264604,4096,1.8446744071580074E19,18,4320,F
2,0,0,0,0,2147483647,1.8446744071579304E19,0,0,K
12620,34831,868352,134512640,138354316,134217729,1.8446744071579496E19,938,1.844674407370955E19,C
13732,34821,868352,134512640,138354316,134217729,1.8446744071579496E19,938,1.844674407370955E19,C
13732,34821,974848,134512640,138354316,134217729,1.8446744071579496E19,938,1.844674407370955E19,C
13732,34821,1146880,134512640,138354316,134217729,1.8446744071579496E19,938,1.844674407370955E19,C
14769,34833,69603328,134512640,138354316,1,1.8446744071579337E19,938,1.844674407370955E19,C
14771,34833,69603328,134512640,138354316,134217729,1.8446744071583224E19,938,1.844674407370955E19,C
14771,34833,696320,134512640,138354316,134217729,1.8446744071579496E19,938,1.8446744073709548E19,C
14771,34833,1138688,134512640,138354316,134217729,1.8446744071579496E19,938,1.844674407370955E19,C
14771,34833,1466368,134512640,138354316,134217729,1.8446744071579496E19,938,1.844674407370955E19,C
14771,34833,1433600,134512640,138354316,134217729,1.8446744071579496E19,938,1.844674407370955E19,C
17342,0,339103744,4194304,4754764,4096,1.8446744071580074E19,137,23236,F
18389,0,459603968,4194304,4747836,16781312,1.8446744071580074E19,136,19160,F
18589,34820,598016,134512640,136885440,1,1.8446744071579496E19,580,1.844674407370955E19,C
18589,34820,1658880,134512640,136885440,1,1.8446744071579496E19,580,1.8446744073709552E19,C
3623,0,43208704,4194304,4376244,0,1.8446744071580074E19,45,4696,D
2,0,0,0,0,2147483647,1.8446744071579177E19,0,0,O
2,0,0,0,0,2147483647,1.8446744071580738E19,0,0,K
0,0,4157440,4194304,4328382,4096,1.8446744071580074E19,33,1588,D,C