Sistema para Detecção de Intrusão em Redes de Computadores com Uso de Técnica de Mineração de Dados

Lidio Mauro Lima de Campos
Universidade Federal do Pará
Faculdade de Sistemas de Informação
Castanhal, Pará, Brasil
lidio@ufpa.br

Alberto Sampaio Lima
Universidade Federal do Ceará
Fortaleza, Ceará, Brasil
albertosampaio@ufc.br

Resumo*: Neste trabalho apresenta-se um Sistema de Detecção de Intrusão Baseado em Mineração de Dados, que utiliza o "dataset" KDDCUP´99. São testados inicialmente dois classificadores Árvore de Decisão (J48) e Rede Neural Artificial (MLP), considerando duas classes de ataque normal e intrusão. Após isto, foram testados os classificadores Árvore de Decisão e Rede Bayesiana considerando cinco classes de ataque: Normal, DOS, U2R, R2L, Probing. A conclusão obtida foi que a Árvore decisão é o melhor classificador para ambos os casos.*

Palavras-Chave:Mineração de Dados, Sistema de Detecção e Intrusão.

Abstract: Introduces an Intrusion Detection System based on Data Mining using the KDDCUP'99 "dataset". Was initially tested two classifiers Decision Tree (J48) and Artificial Neural Network (MLP) considering two classes of attack normal and intrusion. After this was tested two classifiers Decision Tree and Bayesian Network considering five classes of attack: Normal, DOS, U2R, R2L, Probing. It is concluded that the decision tree classifier is the best for both cases.

Keywords: Datamining,Intrusion Detection System.

I. INTRODUÇÃO

Com o aumento do uso das redes de computadores e do número de aplicativos em execução nessas infraestruturas, a segurança de redes torna-se cada vez mais importante. Todos os computadores e sistemas sofrem vulnerabilidades de segurança que são ambos tecnicamente difíceis e caros para serem resolvidos pelos fabricantes [1]. Torna-se indispensável a utilização de Sistemas de Detecção de Intrusão (IDSs) como dispositivos que tem a função principal a detecção de anomalias e ataques nas redes.

Na pesquisa apresentada em [2] foram empregadas 21 máquinas de aprendizado (7 classificadores: Árvore de Decisão J48 [4], Classificador *Naive Bayes* [5], *NBTree* [6], *RandomForest* [7], *RandomTree* [8], *MLP* [9] e *SVM* [10], cada um treinado três vezes) para rotular o conjunto inteiro *KDD* para os dados de treino e testes, o que possibilitou gerar 21 rótulos para cada registro do *dataset*, anotando cada um com um atributo *#sucessfullPrediction*, inicializado como zero, sendo incrementado de um para cada classificador testado. Sendo o maior valor para esse atributo 21, quando todos os classificadores forem capazes de classificar corretamente o registro.

A pesquisa no campo de detecção de intrusão utilizada nessa pesquisa utiliza a técnica de detecção de padrões anormais de conexão. O "*dataset*" utilizado foi KDDCUP´99 [3], que é amplamente utilizado nesses tipos de estudos, sendo que o mesmo apresenta uma série de deficiências. A primeira delas é o grande número de registros redundantes, analisando os dados de treino e testes do "dataset" evidenciou-se que 78% e 75% dos dados são duplicados no conjunto de treino e testes respectivamente. Essa quantidade enorme de registros redundantes pode ocasionar a polarização dos algoritmos de aprendizado em direção dos registros mais frequentes, impedindo o aprendizado de registros menos frequentes que geralmente são mais prejudiciais as redes, tais como os ataques U2R. A existência desses registros repetidos no conjunto de testes fará com que a avaliação dos resultados seja polarizada pelos métodos que tem altas taxas de detecção nos registros frequentes [2].

No presente artigo foi utilizado o "*dataset*" KDDCUP´99, resultado do estudo apresentado em [2].

Foram propostas algumas melhorias ao mesmo, através de modificações no "*dataset*" por meio de pré-processamento visando reduzir o número de atributos de 42 para 27. A partir do "*dataset*" modificado, foi realizado um estudo sobre o problema de detecção de intrusão utilizando Mineração de Dados, testando-se classificadores (j48 e Redes Neurais MLP) considerando inicialmente duas classes de detecção Normal e Intrusão. Após realizadas as simulações, foram comentadas melhorias em relação ao trabalho de [2]. Após essa fase, utilizando o "dataset" KDDCUP´99 [3] (10%), resolver o mesmo problema utilizando dois classificadores (J48 e Redes Bayesiana) considerando 5 classes de detecção, Normal, DOS, R2L, U2R, Probing.

A seção II detalha o "*dataset*" KDDCUP´99, enquanto na secção III são apresentados os conceitos de Mineração de Dados. Na seção IV apresenta-se a definição de Sistemas de Detecção de Intrusão (IDS), na secção V apresentam-se a metodologia e os resultados de simulação, na secção VI as conclusões.

II - DETALHAMENTO DO DATASET KDDCUP´99.

O "*dataset"* utilizado neste trabalho foi o KDDCUP'99, proposto por [11], construído com base nos dados capturados pelo DARPA´S98, programa de avaliação de IDS [12]. É composto por 4GB de arquivo comprimido de dados brutos de conexão de conexão tcp, de 7 semanas de tráfego de rede, que pode ser transformado em cerca de 5 milhões de registros de conexões cada um com 100bytes. Duas semanas de teste tem aproximadamente dois milhões de registros de conexão.

O "*dataset*" de treino consiste de aproximadamente 4.900.000 vetores de conexão simples que contem 41 caraterísticas, sendo rotulado como normal ou ataque, com exatamente um tipo de ataque. Os dados de teste não possuem a mesma distribuição de probabilidade dos dados de treino e incluem tipos específicos de ataque que não estão presentes nos dados de treino o que faz a tarefa mais realista. 24 Tipos de ataque no treino e um total de somente 14 nos dados de teste. Uma descrição detalhada dos tipos de ataque pode ser encontrados em [13].

Os ataques simulados caem em uma das quatros categorias, ilustradas nas Tabela 1.As classes de ataque no KDDCUP´99 podem ser categorizadas em 5 classes principais (uma normal e quatro classes principais de intrusão : *DOS, R2L, U2R and Probing*) de acordo com que é ilustrado na Tabela 2. É importante notar que os conjuntos de dados de treino e testes não apresentam a mesma distribuição e probabilidade. Além disso, os dados de testes contem registros não disponíveis nos dados de treino o que faz a tarefa mais realista. Muitos especialistas em segurança acreditam que novos ataques são variações de ataques conhecidos e que as características desses são suficientes para capturar novas variações.

Os atributos básicos de uma conexão TCP, são: [*Duration*, tempo de conexão em segundos, Contínuo], [*Protocol_type*, tipo de conexão UDP, TCP, Categórico], [*Service*, Tipo de Serviço de Destino (HTTP, *Telnet*), Categórico], [*Flag*, Estado da Conexão, Categórico], [*Src_bytes*, Números de bytes da origem ao destino, Contínuo], [*Dst_bytes*, Número de bytes do destino à origem, Contínuo], [Land, 1 se o Host e a porta da origem e destino são os mesmos, 0 caso Contrário, Categórico], [*Wrong_fragment*, Número de fragmentos errados, Contínuo], [*Urgent*, Número de Pacotes Urgentes, Contínuo].

Os atributos sugeridos de uma conexão TCP são: [*Hot*, Número de indicadores "importantes", Contínuo], [*Num_failed_logins*, Número de tentativas de login com falha , Contínuo], [*Logged_in*, 1 se o login obteve sucesso e 0 Caso contrário, Categórico], [*Num_Comprissed*, Número de condições comprometedoras, Contínuo], [*Root_shell*, 1 se o shell root é obtido 0 caso contrário, Categórico],[*Su_attempted*, 1 se houver tentativa de conseguir "*su root*" 0 caso contrário, Categórico], [*Num_root*, Número de acessos como root, Contínuo], [*Num_file_creation*, Número de operações de criação de arquivos, Contínuo],[*Num_shells*, Número de *shell prompts* abertos, Contínuo],[*Num_access_files*, Número de operações a arquivos de controle de acesso, Contínuo], [*Num_outbund_cmds*, Números de comandos externos (sessão FTP), Contínuo],[*Is_hot_login*, 1 se o login pertence à lista "*hot*" 0 caso contrário, Categórico], [*Is_guest_login*, 1 se o login é do tipo "*guest*", 0 caso contrário", Categórico].

Os atributos de tráfego são : [*Count*, Número de conexões para o mesmo host como conexão atual nos últimos 2 segundos, Contínuo], [Serror_rate, % de conexões que tiveram erros do tipo "*SYN*", Contínuo],[*Rerror_rate*, % de conexões que tiveram erros do tipo "REJ", Contínuo], [*Same_srv_rate*, % de conexões ao mesmo serviço, Contínuo], [*Diff_srv_rate*, % de conexões a diferentes serviços, Contínuo] ,[*Srv_count*, Número de conexões ao mesmo serviço como conexão atual nos últimos 2 segundos, Contínuo], [*Srv_serror_rate*, % de conexões que tiveram erros "SYN", Contínuo], [*Srv_rerror_rate*, % de conexões que tiveram erros "REJ", Contínuo], [*Srv_diffe_host_rate*,% de conexões a diferentes hosts, Contínuo].

Os atributos não documentados são os seguintes:
[*Dst_host_count*, Contínuo], [*Dst_host_srv_count*, Contínuo], [*Dst_host_same_srv_rate*, Contínuo],
[*Dst_host_diff_rate*,Contínuo],
[*Dst_host_same_src_port_rate*,Contínuo],
[*Dst_host_srv_diff_host_rate*, Contínuo],
[*Dst_host_serror_rate*, Contínuo],
[*Dst_host_srv_serror_rate*, Contínuo],
[*Dst_host_rerror_rate*, Contínuo],
[*Dst_host_srv_rerror_rate*, Contínuo].

III – MINERAÇÃO DE DADOS

Mineração de dados, ou "*data mining*", consiste no termo utilizado para nomear o processo de análise de conjuntos de dados com o objetivo de encontrar padrões que representem informações úteis e não triviais. Para tanto, são utilizados métodos matemáticos, heurísticas e algoritmos. A mineração de dados é parte de um processo maior e mais abrangente, o de descoberta de conhecimento em bancos de dados, que tem por objetivo extrair conhecimento de alto nível a partir de dados de baixo nível no contexto de grandes conjuntos de dados [14].

Tabela 1 – Categorias de Ataque

Categoria do Ataque	Descrição
DOS (Negação de Serviço)	Atacante envia um grande número de mensagens que esgote algum dos recursos da vítima, como CPU, memória, banda,etc. Ex: "**syn flood"**.
U2R (User to Root attack)	Atacante acessa o sistema como usuário normal (ganho por : sniffing password, um dicionário local ou engenharia social) e passa a explorar vulnerabilidades para ganhar acesso como root ao sistema. Ex: **"buffer overflow".**
R2L (Remote to local attack)	Ocorre quando um atacante tem a habilidade de enviar pacotes para uma máquina através da rede, mas não tem uma conta nessa máquina e explora alguma vulnerabilidade para ganhar acesso local como usuário da máquina. Ex: "**guessing password".**
Probing	É uma tentativa de reunir informações sobre uma rede de computador com o propósito de burlar os controles de segurança. Ex: **"port scanning".**

Descoberta de conhecimento em bancos de dados, ou "*Knowledge Discovery in Databases*", é o termo, criado em 1989, que se refere ao amplo processo de descobrir conhecimento em dados armazenados em bancos de dados. Tal processo objetiva extrair conhecimento implícito e previamente desconhecido, buscando informação potencialmente útil nos dados. O processo, descrito em [1], consiste em uma sequência de cinco etapas, partindo dos dados existentes e chegando à descoberta do conhecimento extraído dos mesmos.

Seleção dos dados: a primeira etapa consiste em escolher qual o conjunto de dados que será submetido ao processo. Seleciona-se um conjunto de dados alvo, ou foca-se em um subconjunto de variáveis ou amostras de dados.

Pré-processamento: nesta etapa, os dados podem sofrer uma qualificação, a fim de corrigir erros e inconsistências que poderão existir. Incluem-se limpeza de dados, eliminação de dados ruidosos, falta de dados e normalização.

Tabela 2 – Tipos de Ataques e classes

4 classes Principais de Ataque	22 classes de Ataque
Denial of Service (DOS)	back, land, neptune, pod, smurf, teardrop.
Remote to User (R2L)	ftp_write, guess_passwd, imap, multihop, phf, spy, warezclient, warezmaster.
User to Root (U2R)	buffer_overflow, perl, loadmodule, rootkit.
Probing	Ipsweep,nmap,portsweep,satan

Transformação: aqui os dados são convertidos em um formato adequado para serem acessados pelos algoritmos de mineração. É nela que também se realiza uma possível redução no número de variáveis, sumarizando os dados que serão submetidos à mineração.

Mineração: é a etapa mais importante do processo. É nela que o algoritmo escolhido é aplicado sobre os dados a fim de se descobrir padrões interessantes. É fundamental para que esta etapa obtenha resultados de qualidade a correta aplicação dos passos anteriores.

Interpretação dos dados e Visualização: nesta última etapa do processo, os resultados obtidos na mineração são interpretados.

As técnicas de mineração de dados podem ser aplicadas a tarefas como classificação, estimativa, associação, segmentação e sumarização. Essas tarefas são descritas na Tabela3.

IV-SISTEMAS DE DETECÇÃO DE INTRUSÃO - *IDS*

Detecção de intrusão é o processo de monitoramento e análise dos eventos em sistemas de computação ou redes com o objetivo de descobrir sinais de possíveis incidentes que tentam comprometer a confidencialidade, integridade e disponibilidade de recursos computacionais. A detecção baseada em anomalia, focada nessa pesquisa, identifica novos ataques analisando comportamentos anômalos de comportamentos normais. Esse método, entretanto possui uma alta taxa de detecção de novos ataques, mas produz muitos falsos positivos. O

mesmo usa perfis que são desenvolvidos pelo monitoramento de características de atividades típicas ao longo de um período de tempo e compara as características das atividades atuais com valores de limiares associados aos perfis [15].

Tabela 3. Tarefas em Mineração de Dados

Classificação	Consiste em construir um modelo de algum tipo que possa ser aplicado a dados não classificados visando categorizá-los em classes. Um objeto é examinado e classificado de acordo com uma classe definida [14].	-classificar solicitações de pedidos de crédito. -esclarecer fraudes na declaração do imposto de renda.
Regressão	Regressão é aprender uma função que mapeia um item de dado para uma variável de predição real estimada" [14].	-prever a demanda futura de um novo produto. -estimar expectativa de vida média dos brasileiros.
Associação	Identificação de grupos de dados que apresentem coocorrência entre si.	-quais produtos são colocados juntos em carrinhos de supermercado.
Segmentação (ou Clustering)	Processo de partição de uma população heterogênea em vários subgrupos ou grupos mais homogêneos	-agrupamento de clientes com comportamento de compras similar. -comportamento de clientes em compras realizadas na web para uso futuro.
Detecção de desvios (outliers)	Identificação de dados que deveriam seguir um padrão esperado, mas não o fazem.	-detecção de intrusão em redes de computadores.

Um sistema de detecção de intrusão (IDS) monitora e analisa o tráfego da rede, utilizando múltiplos sensores para detectar intrusões de redes externas e internas. O IDS analisa a informação coletada pelos sensores e retorna uma síntese da entrada dos sensores para o administrador do sistema ou sistema de prevenção de intrusão, que executa a prescrição controlada pelo IDS. Hoje em dia mineração e dados tornou-se uma ferramenta indispensável para analisar a entrada dos sensores no IDS [15]. A Figura 1 ilustra um IDS típico. Um IDS ideal deveria ter uma taxa de detecção de ataque de 100% com 0% de falso positivo. Entretanto, na prática isso é difícil obter, os parâmetros mais importantes na estimação do desempenho de um IDS são mostrados na Tabela 4.

Tabela 4 – Parâmetros na definição de um IDS.

Parâmetro	Definição
Taxa de detecção (DR)	Ataque ocorre e o alarme dispara.
Falso Positivo (FP)	Ataque não ocorre , mas o alarme dispara.
Verdadeiro Negativo (TN)	Ataque não ocorre e alarme não dispara.
Falso Negativo (FN)	Ataque ocorre, mas alarme não dispara.

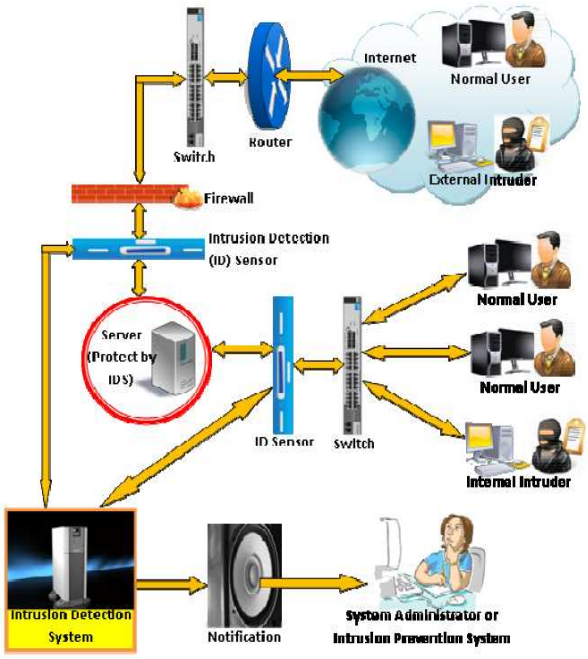

Figura 1 - Sistema de Detecção de Intrusão Típico

V-METODOLOGIA E RESULTADOS DE SIMULAÇÃO

Inicialmente para realização dos experimentos utilizou-se o dataset KDDCUP´99 modificado, proposto por [2] e disponível em [16], com algumas modificações. Para treino utilizou-se o arquivo *KDDTrain+.arff* e para testes *KDDTest+.arff* , entretanto reduziu-se o número de atributos de 42 para 27, as justificativas para eliminação dos atributos foram as seguintes: utilizando-se estatísticas do software "Weka" foram eliminados atributos que possuíam valor único dentre eles "*num_outbound_cmds*" e "*is_host_login*". Foram eliminados atributos com alto valor de correlação, convencionou-se chamar atributos fortemente correlacionados aqueles que possuíam coeficientes de correlação maiores ou iguais a 0.8. O interesse era fazer seleção de atributos e não síntese por isso eliminou-se os atributos altamente correlacionados que menos ajudavam na classificação [1]. Atributos altamente correlacionados influenciam um ao outro e trazem pouca informação, então não é interessante ter atributos correlacionados em qualquer problema, usamos PCA (*Principal Components Analysis*): *sensor_rate, same_srv_rate, srv_serror_rate, st_host_srv_serror_rate, rerror_rate, srv_rerror_rate, dst_host_srv_rerror_rate, srv_count*.

Após a seleção de atributos normalizaram-se alguns atributos:
*wrong_fragment,num_failed_logins,num_compromised,num_file_creations,num_access_files,count , dst_host_count e duration***,** foram normalizados de forma que todos os

valores fiquem no intervalo [0,1], usou-se o filtro normalize do ambiente *weka*. Ele se torna necessária para que os dados tenham a mesma ordem de grandeza. Se não houvesse essa normalização poderiam existir grandeza que teriam mais importância que outras**.** Ou seja, após essas modificações o "dataset" fornecido por [16] *KDDTrain+.arff* e para testes *KDDTest+.arff* passou a ter 27 atributos, como isso obteve-se melhorias, em relação ao trabalho de [2], de acordo com o que é ilustrado na Tabela 5,

Utilizou-se uma árvore de decisão (*J48-Weka*) para realizar o treinamento e teste. O algoritmo J48 é uma implementação em java do Algoritmo C4.5. Na Tabela 5 , percebe-se que utilizando-se os dados de testes e árvore de decisão, obteve-se uma taxa de detecção 99.4% para conexões normais e 91.1% para intrusão e falsos positivos de 8.9% e 6%. Utilizando-se os dados de testes e uma rede MLP com 23 neurônios na camada de entrada, dois na camada intermediária, um na camada de saída, taxa de aprendizagem de 0.3 , momentum 0.2, função de ativação sigmoide para todos os neurônios, 50000 épocas, obteve-se uma taxa de detecção de 95% para conexões normais, 92.3% para intrusão e 7.7% de falsos postivos para conexões normais e 5% para intrusão O total de instâncias classificadas corretamente, pela árvore de decisão, foi de 95.12% e pelo trabalho de [2] foi 93.89% e pela rede *MLP* FOI DE 93.47% e pelo trabalho de [2] foi de 92.26%, portanto a redução de atributos de acordo com as técnicas mostradas anteriormente melhoraram os desempenhos do classificadores *J48* e *MLP*.

Os experimentos realizados utilizando-se o "*dataset* modificado" proposto por [2] considerou-se apenas duas classes para classificação. Realizaram-se experimentos adicionais considerando-se cinco classes: Normal, DOS, *Probing*, *R2L* e *U2R*. Considerando o "*dataset*" proposto por [3], considerando-se os 42 atributos. Utilizando-se os classificadores *J48* e rede bayesiana os resultados são mostrados na Tabela 6.

Tabela 5 – Resultados para teste utilizando Árvore de Decisão J48, considerando o dataset fornecido por [2] e a redução e atributos de 42 para 27.

Classificador	Normal	Intrusão
J48 (DR%) | MLP	**99.4% | 95%**	**91.1% | 92.3%**
J48 (FP%) | MLP	**8.9% | 7.7%**	**6% | 5%**

Algumas regras extraídas da árvore de decisão utilizada nas simulações são ilustradas na tabela 7.

VI - CONCLUSÕES

Nos modelos propostos no presente estudo foram consideradas algumas simplificações: Não existe definição de ataque, por exemplo *probing* não é necessariamente um tipo de ataque a não ser que o número de iterações exceda um limiar específico.

Similarmente um pacote que cause um *buffer_overflow* não é necessariamente um ataque. Coletores de Tráfego como o *TCP DUMP* que é usado no DARPAS´98 são fáceis de serem sobrecarregados e derrubar pacotes em carga de tráfego pesado, não foram checadas as possibilidades de derrubada de pacotes.

Foram efetuados testes com um grupo de classificadores, onde o melhor classificador para ambos os casos foi o J48. O *MLP* foi um bom classificador para 2 classes, enquanto a rede bayesiana não foi um bom classificador para 5 classes.

Tabela 6 – Resultados para teste para o dataset [3], considerando 5 classes.

Class	Normal	Probe	DOS	R2L	U2R
J48 (DR%)	**98.9%**	**98.3%**	**99.7%**	**95.2%**	**93.9%**
J48 (FP%)	**0.04%**	**0.02%**	**0.03%**	**0.01%**	**0.01%**
Bayesian Network (DR%)	**99.1%**	**93.5%**	**98.7%**	**69.3%**	**90.03%**
Bayesian Network (FP%)	**0.13%**	**0.05%**	**0.02%**	**0.14%**	**0.06%**

Tabela 7 – Regras obtidas , pela Árvore de Decisão, "dataset" usado [3].

Tipo de Ataque	Regras
DOS	**SE** ((flag=REJ **OU** flag=RSTO **OU** flag=SO) **E** land<0.5) **ENTÃO** label = neptune
Probe	**SE** (count<3.5 **E** (service=eco_i **OU** service=ecr_i) **ENTÃO** label=ipsweep
	SE(count>=3.5 **E** dst_host_count<128.5 **E** dst_host_same_src_port_rate<0.56) **ENTÃO** label=portsweep
R2L	**SE** (service=pop_3 **OU** service=telnet) **ENTÃO** label=guess_passwd
	SE((service=pop_3 **OU** service=telnet) **E** (num_failed_logins <0.5) **E** (flag!=REJ **OU** flag!=RSTO) **E** (service=http **OU** service=login)) **ENTAO** label=ftp_write
U2R	**SE** (dst_bytes<665.5) **ENTÃO** label=rootkit
	SE (dst_bytes>=665.5) **ENTÃO** label=Buffer_overflow

Referências

[1]C. E. Landwehr, A. R. Bull, J. P. McDermott, and W. S. Choi, “A taxonomy of computer program security flaws,” ACM Comput. Surv., vol. 26, no. 3, pp. 211–254, 1994.

[2]M. Tavallaee, E. Bagheri, W. Lu, and A. Ghorbani, “A Detailed Analysis of the KDD CUP 99 Data Set, *Submitted to Second IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA)*, 2009.

[3]KDD Cup 1999. Available on: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html. Dezembro de 2011.

[4] J. Quinlan, C4.5: Programs for Machine Learning. Morgan Kaufmann, 1993.

[5] G. John and P. Langley, “Estimating continuous distributions in Bayesian classifiers,” in Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence, pp. 338–345, 1995.

[6] R. Kohavi, “Scaling up the accuracy of naive-Bayes classifiers: A decision-tree hybrid,” in Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, vol. 7, 1996.

[7] L. Breiman, “Random Forests,” Machine Learning, vol. 45, no. 1, pp. 5–32, 2001.

[8] D. Aldous, “The continuum random tree. I,” The Annals of Probability, pp. 1–28, 1991.

[9] D. Ruck, S. Rogers, M. Kabrisky, M. Oxley, and B. Suter, “The multilayer perceptron as an approximation to a Bayes optimal discriminant function,” IEEE Transactions on Neural Networks, vol. 1, no. 4, pp. 296–298, 1990.

[10] C. Chang and C. Lin, “LIBSVM: a library for support vector machines,” 2001. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[11] S. J. Stolfo, W. Fan, W. Lee, A. Prodromidis, and P. K. Chan, “Costbased modeling for fraud and intrusion detection: Results from the jam project,” discex, vol. 02, p. 1130, 2000.

[12]R. P. Lippmann, D. J. Fried, I. Graf, J. W. Haines, K. R. Kendall, D. McClung, D. Weber, S. E. Webster, D. Wyschogrod, R. K. Cunningham, and M. A. Zissman, “Evaluating intrusion detection systems: The 1998 darpa off-line intrusion detection evaluation,” discex, vol. 02, p. 1012, 2000.

[13] MIT Lincoln Labs, 1998 DARPA Intrusion Detection Evaluation. Available on: http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/ index.html, october 2011.

[14] Fayyad, Usama; Piatetski-Shapiro, Gregory; Smyth, Padhraic (1996). The KDD Process for Extracting Useful Knowledge from Volumes of Data. In: Communications of the ACM, pp.27-34, Nov.1996

[15]Dewan Md. Farid, Nouria Harbi and Mohammad Zahidur Rahman, Combining naive Bayes and Decision Tree for adaptative Intrusion Detection . International Journal of Network Security & Its Applications (IJNSA), Volume 2 number 2, April 2010.

[16]KDD Cup 1999. Available on http://www.iscx.ca/NSL-KDD/. Janeiro de 2012