X Congresso Brasileiro de Inteligência Computacional (CBIC'2011), 8 a 11 de Novembro de 2011, Fortaleza, Ceará

DETECÇÃO DE INTRUSOS EM REDES DE COMPUTADORES COM USO DE CÓDIGOS CORRETORES DE ERROS E MEDIDAS DE INFORMAÇÃO

Christiane Ferreira Lemos Lima, Francisco Marcos de Assis, Cleonilson Protásio de Souza
Instituto Federal do Maranhão, Universidade Federal de Campina Grande, Universidade Federal da Paraíba
cfllima@ifma.edu.br, fmarcos@dee.ufcg.edu.br, protasio@ct.ufpb.br

Resumo – A identificação de tipos de ataques a uma rede de computadores pode ser vista como uma tarefa de classificação multiclasses, por envolver a discriminação de ataques em diversos tipos ou categorias. Com base nessa abordagem, o presente artigo apresenta estratégias para decompor problemas multiclasses em múltiplos subproblemas binários, usando a abordagem de matriz código, em que para cada uma das M classes foi associada uma palavra código de comprimento igual a N, em que N é o número de classificadores binários selecionados na fase de decomposição, através do uso de medidas de informação. Para esta abordagem as palavras códigos que formam a matriz código são restritas a um sub-código de um código linear do tipo BCH. Esta abordagem permite que a decodificação seja feita utilizando-se algoritmos de decodificação algébrica e assim, quando o número de classes for elevado, evitar uma decodificação exaustiva através do uso de tabela. Nesse contexto, o uso de medidas de informação e algoritmo genético guiado por código são aplicados no projeto de matrizes códigos para serem utilizadas na identificação de ataques em redes de computadores, apresentando bons resultados.

Palavras-chave – Detecção de Intrusos, Classificação multiclasse, Codificação, Medidas de Informação, Algoritmos Genéticos.

Abstract – The identification of types of attacks in a network can be viewed as a multiclass classification task because it involves attacks discrimination on various types or categories. Based on this approach, this paper presents strategies for decomposing multiclass problems into multiple binary subproblems, using the code matrix approach, where for each M classe was associated with a codeword of length equal to N, where N is the number of binary classifiers generated during the decomposition. For this approach the codewords that form the code matrix form a sub-code of a linear code of type BCH. This approach allows the decoding is done using algebraic decoding algorithms, so that when the number of classes is high, preventing an exhaustive decoding through the use of the table. In this context, the use of information measures and genetic algorithm guided by the code are applied in the design code matrix to be used in identification of attacks on computer networks, presenting good results.

Keywords – intrusion detection, multiclass classification, codification, information measures, genetic algorithm.

1. Introdução

Um sistema de detecção de intrusão (SDI) realiza o processo de monitorar, identificar e, em alguns casos, isolar eventos suspeitos ocorridos em um sistema ou rede de computadores em particular, através da análise dos pacotes que trafegam nos segmentos de rede e/ou eventos registrados nos arquivos de auditoria dos sistemas monitorados, baseando-se no fato de que a maioria dos ataques seguem determinados padrões de comportamento.

Considerando que na detecção de intrusos uma das dificuldades se dá ao fato de que o volume de tráfego que circula na rede e arquivos de auditoria são imensos, tornando inviável para especialistas supervisionarem todos os sinais de suspeitas de intrusão, faz-se necessário o uso de técnicas que possibilitem a descoberta de padrões, similaridades e conhecimento de forma automática, a partir de dados, facilitando, dessa forma, a análise de informações a serem utilizadas na tomada de decisão [1]. Dentre os principais métodos utilizados têm-se: a classificação, a regressão, o agrupamento e a associação.

O método de classificação permite obter padrões através da construção de classificadores, tendo como base um conjunto de dados (instâncias) em diferentes classes, levando em consideração as propriedades de seus atributos. Com base nessa abordagem, a identificação de tipos de ataques a uma rede de computadores pode ser vista como uma tarefa de classificação multiclasses, por envolver a discriminação de ataques em diversos tipos ou categorias.

Essa tarefa, a princípio, é mais complexa do que em sistemas que apenas classificam conexões como normais ou suspeitas (classificação binária), uma vez que o classificador gerado deve ser capaz de separar os dados em um número maior de categorias, o que também aumenta a chance de erros de classificação serem cometidos.

Uma estratégia adotada num problema multiclasses é decompor-lhe em vários subproblemas binários, sendo estes resolvidos individualmente, cujas saídas são combinadas na obtenção do classificador multiclasses final. Essa estratégia, conhecida como decomposicional, possui algumas vantagens que motivam seu uso, tais como:

- Possibilita a redução da complexidade envolvida na separação das classes [2] [3];
- Viabiliza a utilização de algoritmos específicos que só conseguem tratar problemas com duas classes [4] [5];

- Proporciona o processamento paralelo [4] [5];
- Permite reduzir o número de erros de classificação (com custos distintos) de diferentes classes [4] [5].

Existem diversas estratégias decomposicionais. Entretanto, de forma generalizada, é possível agrupá-las em dois tipos, a saber: baseadas em matrizes códigos e hierárquicas [4] [6].

O enfoque dessa pesquisa está no uso da estratégia decomposicional baseada em matrizes códigos, as quais podem ser representadas por uma matriz $C_{M\times N}$, sendo M o número de classes possíveis e N o número de classificadores binários ou atributos a serem observados. A obtenção da classificação final envolve um processo de decodificação, considerando as saídas de todos os classificadores binários.

Com base nessa abordagem, os códigos corretores de erros são adaptados ao problema de classificação, conforme em [7] [8] [9]. Nesse contexto, a determinação de matrizes códigos adequadas ao problema de detectar intrusos em redes de computadores é abordada neste trabalho, em que para cada classe será associada uma palavra código de comprimento igual a N.

Dessa forma, as palavras código que formam as linhas da matriz $C_{M\times N}$, pertencem a um código BCH, formando um subcódigo. Propõe-se também para a seleção da matriz código mais adequada a utilização de Medidas de Informação e um Algoritmo Genético.

O restante deste artigo está organizado da seguinte forma. Na seção 2 será detalhada a estratégia de decomposicão de problemas multiclasses usando códigos corretores de erros; a seção 3 descreve o projeto da matriz código; na seção 4 são apresentados os resultados obtidos; por fim, na seção 5 serão feitas algumas considerações finais.

2. Estratégias Baseadas em Matrizes de Códigos

Um problema multiclasse resolvido a partir da aplicação de estratégias decomposicionais baseada em matrizes código pode envolver, de forma geral, três etapas:

- Determinar qual estratégia a ser usada para projetar a matriz código;
- A escolha dos classificadores binários ou atributos a serem utilizados na matriz com base;
- A estratégia de decodificação adotada.

Determinar qual estratégia a ser usada para projetar a matriz código é uma questão importante. Uma vez que o desempenho dessa abordagem está fortemente relacionada com a matriz código escolhida, o seu projeto analítico torna-se bastante complexo. Assim, procura-se a combinação de classificadores binários na matriz código que leve a um bom desempenho na solução multiclasses.

Entre as estratégias decomposicionais baseadas em matriz código, as mais comuns encontradas na literatura [5] [4] [9] [10] estão a:

- Um-Contra-Todos - consiste em comparar cada uma das M classes do problema original contra as demais;
- Todos-Contra-Todos - consiste em criar conjuntos de dados com todas as combinações possíveis das classes do problema original, juntado-as duas a duas;
- Baseada nos princípios dos Códigos Corretores de Erros.

Os códigos corretores de erros são freqüentemente usados em sistemas de comunicação digital para proteger a informação de ruídos e interferências e reduzir o número de erros de bits [11] [12] [13]. Contudo, a teoria de codificação é uma área independente de pesquisa e um número de direções não tradicionais têm aparecido, em especial na resolução de problemas multiclasses [7] [14] [6] [8] [15] [9] [10].

2.1 Códigos Corretores de Erros

Existem diferentes classes de códigos corretores de erros. Nessa pesquisa, serão abordados apenas os códigos de bloco lineares, especialmente os códigos cíclicos. Os códigos cíclicos são representados por polinômios e a sua grande popularidade resulta da sua facilidade de implementação através de registradores de deslocamento e lógica combinacional [12] [13].

Um código de bloco é caracterizado geralmente na forma (n, k). Neste tipo de codificação, a informação original é fragmentada em segmentos com k símbolos. O codificador transforma esses k símbolos num bloco codificado de n símbolos. Adicionam-se portanto $n - k$ símbolos redundantes a cada bloco de k símbolos de informação com o propósito de corrigir, ou pelo menos detectar erros. Na forma sistemática, os símbolos de redundância são anexados aos símbolos de informação. Para cada seqüência de k símbolos de dados, existe uma palavra de código distinta de n símbolos.

A distância de *Hamming*, $d(\overline{v}_i, \overline{v}_j)$, é definida como o número de posições em que o par de palavras código distintas diferem. Se $\lfloor \frac{d-1}{2} \rfloor$ erros ocorrem, então o decodificador provavelmente irá corrigir os erros, presumindo que a palavra código mais próxima da palavra recebida foi a transmitida.

Os códigos BCH são códigos largamente utilizados que pertencem a classe dos códigos cíclicos. Um código $BCH(n, k, d)$ é um subespaço vetorial linear com elementos (palavras código) de comprimento n, com dimensão k e distância de *Hamming*

mínima d_{min}. A estrutura imposta pelos espaços vetoriais propiciam meios estruturados para a construção de codificadores e decodificadores. Para um código $BCH(n, k, d)$ binário existem 2^k palavras códigos que podem ser combinadas em grupos de M para formar uma matriz código $C_{M \times N}$.

O conjunto de palavras código (polinômios) é definido com uma estrutura algébrica denominada ideal contida no anel $GF(q)[x]/(x^n - 1)$. Portanto, um código cíclico é um conjunto de todos os múltiplos de um polinômio $\overline{g}(x) \in GF(q)[x]/(x^n - 1)$, chamado de *polinômio gerador* do código, por polinômios de grau menor ou igual a $k - 1$.

2.2 Decomposição por Códigos Corretores de Erros

Apresentada em 1995 por Dietterich e Bakiri [7], a estratégia conhecida como decomposição por códigos corretores de erros (*Error Correcting Output Codes* - ECOC), utilizada nessa pesquisa, destaca-se pela sua boa capacidade de generalização.

Com base na teoria da codificação, Dietterich e Bakiri [7] sugerem que o número de classificadores seja maior que o mínimo necessário para diferenciar cada classe unicamente. Esses classificadores adicionais inserem redundância na codificação das classes e têm a utilidade de fornecer ao sistema a capacidade de corrigir, ou pelo menos detectar possíveis erros que possam estar presentes nos atributos observados, nos dados de treinamento utilizados no projeto da matriz ou erros cometidos por alguns dos classificadores binários no processo de classificação de um evento.

Para possibilitar a correção de erros e determinar a classe de um evento com maior precisão, [7] também sugerem que a distância de *Hamming* mínima entre as linhas e entre as colunas da matriz código seja grande. As matrizes códigos obtidas devem ainda obedecer algumas regras, conforme [5]:

- Inexistência de colunas iguais ou colunas complementares, pois representam o mesmo problema, dando origem ao mesmo modelo de decisão;
- Nenhuma coluna apenas com 0 ou 1, pois não representam problemas de decisão.

As características supracitadas serviram de motivação para a escolha dessa estratégia, pois do ponto de vista da detecção de intrusos, a grande maioria dos ataques podem ser codificados, podendo existir assinaturas variadas, porém próximas, a cerca de atividades que exploram as mesmas vulnerabilidades. Portanto, o uso de códigos corretores de erro pode ser adequado para identificar novos tipos de ataques.

Nessa abordagem, a decomposição mínima de um problema com M classes pode ser realizada com o uso de $\lceil \log_2 M \rceil$ classificadores binários e a decomposição máxima é dada por $2^{M-1} - 1$. Dentro desse intervalo, qualquer valor pode ser usado. Baseado nisso, Dietterich e Bakiri [7] propuseram quatro técnicas para o projeto de matrizes códigos usando Códigos Corretores de Erros. A escolha de cada uma delas é determinada pelo número de classes do problema, a saber:

- Para $3 \leq M \leq 7$, é recomendado o uso de um código exaustivo, que consiste na combinação de $2^{M-1} - 1$ colunas. A distância de *Hamming* mínima do código obtida pelo método exaustivo é $d_{min} = 2^{M-2}$.
- Se $8 \leq M \leq 11$, é aplicado um método que seleciona colunas do código exaustivo. O algoritmo GSAT [16] é indicado pelos autores para solucionar este problema.
- Para $M > 11$, tem-se duas opções: um método baseado no algoritmo de busca aleatória *hill-climbing* e a geração de códigos BCH.

Nesse contexto, tenta-se minimizar também o número de classificadores binários contidos nessas matrizes, representando assim a busca por decomposições mais simples, que apresentem um bom desempenho de classificação. Dessa forma, definir qual subconjunto de todos os atributos possíveis, consegue predizer melhor atividades intrusivas é um problema que precisa ser tratado. Uma boa análise do conjunto de atributos contribui bastante para o sucesso do projeto da matriz código.

Com relação à fase de decodificação, levando em consideração que o número de classes ou tipos de ataques em uma rede de computadores pode ser bastante elevado, pretende-se realizar através de algoritmos de decodificação algébrica [12]. Quando o número de classes é grande, esse processo é menos exaustivo do que o uso de tabelas por códigos que não possuem nenhuma estrutura e a decodificação só pode ser feita por comparação da palavra recebida com as palavras do código.

3. Projeto da Matriz Código

A estratégia estudada nessa pesquisa serão representadas por uma matriz código C, com elementos $c_{ij} \in \{0, 1\}, j = 0, ..., N - 1$. Cada linha de N bits forma uma palavra código restrita às palavras de um código BCH, que corresponde a uma das classes ou hipóteses $H_i \in \Omega = H_0, ..., H_{M-1}$.

Na abordagem aqui utilizada, as palavras código que formam a matriz código formam um subcódigo de um código linear. Este subcódigo não é linear, mas os parâmetros do código tais como a distância de *Hamming* mínima do código se mantém. Portanto, a distância entre as linhas da matriz código é limitada pela distância mínima do código BCH, enquanto que para um código aleatório podem ocorrer valores menores.

3.1 Preparação dos Dados

Um ponto importante para um projeto desta natureza é encontrar uma fonte de dados que possa ser utilizada como conjunto de treinamento e teste pelos classificadores. Portanto, nas pesquisas desenvolvidas está sendo utilizado um subconjunto da base de dados- disponibilizada na competição internacional de mineração de dados, *Knowledge Discovery and Data Mining Competition - KDD Cup 99* [17] contendo um conjunto padrão de dados para serem auditados, incluindo uma grande variedade de intrusões simuladas durante nove semanas em uma rede militar de computadores. Devido ao grande volume de dados e com o intuito de reduzir o custo computacional, selecionou-se aleatoriamente um subconjunto de 22.653 instâncias, de um total de 494.021 [18].

Essa base de dados contém 2.351 registros de conexões indicando tráfego de rede normal e 20.302 registros contendo 22 tipos de ataques identificados, conforme ilustrado na Tabela 1, os quais se enquadram em uma das quatro categorias principais.

Para cada registro, tem-se 41 atributos para caracterizarem as assinaturas de ataque. Esses atributos são divididos em três grupos: atributos básicos de conexões TCP individuais, atributos de tráfego e atributos com conteúdos relacionados ao domínio de conhecimento.

Serão utilizados atributos para reconhecer as seguintes categorias de ataques:

- **User to Root (U2R)** - quando uma pessoa que possui acesso autorizado como um usuário e tenta obter acesso como superusuário (*root*).
- **Remote to Local (R2L)** - quando uma pessoa, em uma máquina remota, tenta obter acesso ao servidor local.
- **Probing** - um intruso varre uma rede em busca de informação ou para encontrar vulnerabilidades conhecidas para explorar.
- **DOS** - ataques de negação de serviço. O objetivo do invasor é causar a indisponibilidade do serviço alvo, seja através do consumo de recursos computacionais como memória, processador ou rede.

Tabela 1: Identificação de ataques por categoria

DoS	PROBING	R2L	U2R
back (1026)	ipsweep (586)	ftp-write (8)	buffer-overflow(21)
land (11)	nmap (151)	guess-passwd(53)	loadmodule (10)
neptune(10401)	portsweep (155)	imap (11)	perl (3)
pod (69)	satan (16)	multihop (11)	rootkit (7)
smurf (7669)	spy (4)	phf (5)	
teardrop (15)		wareszclient (60)	
		warezmaster (20)	

Os atributos nessa base de dados contêm dados de diversos tipos, com resolução variando significativamente em intervalos. Entretanto, como os códigos BCH não são capazes de processar dados em todos os formatos, isso representa um grande desafio. Daí o pré-processamento se faz necessário para transformar os dados em um formato aceitável pelo código. Portanto, as estratégias de transformação, bem como redução de dados são atividades cruciais e requerem conhecimento amplo do domínio, pois contribuem diretamente para o sucesso do processo de decomposição.

Assim, os atributos contínuos foram discretizados com o objetivo de redução de numerosidade. Para isso, foram realizados vários experimentos para particionar esses atributos em diversas faixas de intervalos. Os melhores resultados foram encontrados utilizando-se os algoritmos para discretização de atributos *Minimum Description Lenght* (MDL), proposto por Fayyad and Irani [19], o qual usa a Informação Mútua [11] como critério para encontrar o ponto onde há alternância de classe e o algoritmo proposto por Igor Kononenko [19], sendo que este fornece uma correção para a tendência que a medida de entropia tem para atributos com múltiplos valores.

Nesta fase, diversas bases de dados foram geradas e, em seguida, os atributos discretos, tais como *service* (com 70 símbolos diferentes) e os atributos transformados anteriormente foram mapeados para valores binários. Todos os atributos originalmente binários, tais como *logged_in*, permaneceram inalterados. Essas bases de dados binarizadas foram usadas na fase de decomposição, com o objetivo de encontrar os melhores classificadores binários a serem utilizados na matriz código.

3.2 Seleção dos classificadores binários

Um dos problemas centrais na detecção de intrusão é identificar um conjunto representativo de atributos que caracterize atividades intrusivas. Os algoritmos de árvore de decisão são ainda uma forma simples e computacionalmente eficiente para extrair um conjunto de atributos chave.

A construção da árvore de decisão, baseada nos algoritmos *ID3 -Induction of Decision Tree* [20] e C4.5 [21], utiliza medidas de informação de Shannon [22] [23] como critério de seleção dos atributos que serão utilizados na geração do modelo a partir de uma base de dados de treinamento. Para isso, os atributos são avaliados através do cálculo de sua entropia e o conjunto de atributos escolhidos são aqueles que maximizam a informação mútua de cada iteração.

Esse critério é usado no algoritmo ID3 para selecionar quais os atributos que serão utilizados a cada passo, enquanto a árvore é construída. Entretanto, apesar de apresentar bons resultados, o mesmo possui uma forte tendência a favorecer os testes de atributos com muitos valores. Para contornar este problema, Quinlan [21] propõe uma alternativa a ser utilizada no algoritmo C4.5 nomeada como **razão de ganho** e será esse o critério a ser utilizado na escolha dos classificadores binários, que irão compor a matriz código, sendo cada um responsável por produzir exatamente um bit da palavra código.

Para todas as etapas da preparação dos dados e seleção de atributos foi utilizada a ferramenta chamada WEKA (*Waikato Environment for Knowledge Analysis*), desenvolvida em linguagem Java pela Universidade de Waikato na Nova Zelândia [24] e distribuída sob os termos GNU (General Public License). Esta ferramenta é uma coleção organizada do estado da arte de algoritmos de máquina de aprendizagem, ferramentas e filtros que auxiliam no pré-processamento de dados. O algoritmo C4.5 é implementado nessa ferramenta através do algoritmo J48.

3.3 Fase de decomposição

Dado que nessa pesquisa a identificação de ataques em redes de computadores possui $M = 23$ classes, encontrar quantos classificadores binários serão utilizados não é uma tarefa trivial. Portanto, conforme já referido, na fase de decomposição pretende-se decompor um problema multiclasse em vários problemas binários.

Construído segundo o algoritmo de árvore de decisão C4.5 e com base num conjunto de dados organizado nos experimentos de binarização, os classificadores binários são treinados de forma a aprender as classes representadas nas colunas da matriz código C. Portanto, cada problema binário terá associado um modelo de decisão, capaz de predizer, com um boa capacidade de generalização, a classe de novos eventos. Tem-se, então, a matriz código gerada com k elementos, contendo os classificadores binários mais relevantes na identificação das M classes.

Geralmente o número de classificadores binários é menor que a quantidade de palavras código disponíveis e isto possibilita que seja possível se ter diferentes combinações para a matriz código. Portanto, é necessário selecionar a mais adequada, ou seja, realizar a busca por um subcódigo dentro de um código BCH.

Ao realizar a seleção da matriz código para a abordagem proposta nessa pesquisa foi utilizado um algoritmo genético (AG) guiado por código [14] para garantir que as palavras código pertençam ao código BCH escolhido e ao mesmo tempo selecionar a que melhor represente uma determinada classe H_i. Este algoritmo é uma modificação do algoritmo genético padrão proposto em [25] [26] em que a busca é realizada no espaço das sequências de informação, que são depois codificadas em palavras código e então avalia-se a solução.

Os algoritmos genéticos são algoritmos de busca e otimização com princípios embasados na genética e na teoria da evolução natural. Parte-se de uma população inicial de possíveis soluções para o problema a ser resolvido, as quais são referencias como indivíduos. A cada iteração t, também chamada de geração, uma nova população é produzida por meio de um processo evolutivo(seleção, cruzamento e mutação) até evoluir para uma solução ótima ou sub-ótima. Com esse procedimento, as matrizes códigos podem ser adaptadas à solução de cada problema multiclasses particular.

Conforme o algoritmo utilizado neste trabalho e ilustrado na Figura 1, os indivíduos são codificados por uma estrutura de dados denominada cromossomo e representados por vetores binários de comprimento $M \times k$, correspondendo a uma matriz com M linhas e k colunas. A solução é o resultado desta matriz multiplicada pela matriz geradora $G_{k \times n}$ de um código $BCH(n, k, d)$, ou seja, uma matriz código $C_{M \times N}$.

ENTRADA: função objetivo, matriz geradora do código $G_{k \times n}$
SAÍDA: matriz código ótima ou subótima $C_{M \times N}$
INICIALIZAÇÃO:
1: $t \leftarrow 0$;
2: Gerar população inicial de indivíduos $P(t) \subset 0, 1^{M \times k}$;
3: Avaliar a aptidão dos indivíduos em $P(t)G$;
ITERAÇÃO:
3: **Repita**
4: $P'(t) \leftarrow$ variação $P(t)$;
5: Avaliar $P'(t)G$;
6: $P(t+1) \leftarrow$ selecionar $P'(t)$;
7: $t \leftarrow t + 1$;
8: **Até que** (critério de parada seja atingido)
FIM

Figura 1: Algoritmo Genético guiado por código para busca dentro de um código BCH [14]

Nesse algoritmo, a partir da população avaliada $P(t)G$, aplica-se um mecanismo para a seleção de indivíduos que passarão para a fase de reprodução, produzindo descendentes que formarão uma nova população $P'(t)$. Esta seleção deve privilegiar a escolha de indivíduos mais aptos, conforme os princípios da seleção natural.

Existem diversos métodos de seleção, sendo que nesse algoritmo foi utilizado o mais simples, o método da roleta. Nesse método, a chance de um indivíduo ser escolhido para reprodução é proporcional à sua aptidão em comparação com todos os indivíduos da população. O valor da aptidão do indivíduo é calculado através da função objetivo representada na Equação 1.

$$ap(t) = AvalP(t)G \quad (1)$$

Nessa equação, $AvalP(t)G$ refere-se ao percentual médio de acerto obtido pelo classificador multiclasses representado na matriz de códigos em um conjunto de validação escolhido aleatoriamente, conforme a Tabela 1. O AG implementado deve buscar soluções que maximizem essa medida. As classificações desconhecidas, que ocorrem caso mais de uma linha da matriz código apresente distância mínima em relação ao evento a ser testado, ou quando duas ou mais linhas da matriz são iguais, gerando falsos positivos, encontram-se embutidas nesse erro.

Na reprodução dos indivíduos selecionados, busca-se combinar as suas características ou genes na obtenção dos descendentes, representando o conceito de hereditariedade. Essa combinação é realizada pela aplicação de um operador genético denominado cruzamento. O cruzamento é um operador binário, sendo aplicado sobre dois indivíduos. Estes indivíduos são denominados pais e os seus cromossomos são combinados na produção de dois novos indivíduos, os filhos.

O operador de cruzamento comumente aplicado e usado no algoritmo de AG guiado por código utilizado neste trabalho é o cruzamento de um ponto com uma taxa préfixada p_c, sendo $0,6 \leq p_c \leq 0,99$. Dados dois pais, o algoritmo escolhe aleatoriamente um ponto de corte. Então, os filhos são obtidos trocando-se as partes dos pais posteriores ao ponto de corte.

Por meio da combinação produzida pelo cruzamento, obtém-se também uma variabilidade nas novas soluções. O conceito de variabilidade genética é reforçado pela aplicação de um operador unário denominado mutação. A mutação altera genes dos indivíduos da nova população gerada na etapa de cruzamento. Dado um indivíduo, pode-se implementar a mutação alterando o valor de um de seus genes segundo uma taxa p_m, sendo $0,001 \leq p_c \leq 0,1$.

3.4 Fase de decodifição

Nessa pesquisa, com base na matriz $C_{M\times N}$ gerada, na fase de decodificação, cada exemplo é classificado através de algoritmos de decodificação algébrica, que utiliza o algoritmo de *Berlekamp-Massey* (BMA) [27], sendo este um método que busca encontrar a palavra código mais próxima $c(x)$ de uma palavra recebida $v(x)$. Isto evita uma decodificação massiva por tabela em sistemas em que o número de classes a serem identificadas é grande.

Devido a erros que podem estar presentes nos atributos do exemplo em função de uma atividade maliciosa, nos dados de treinamento utilizados ou em falhas no processo de aprendizado, a palavra recebida ou exemplo a ser avaliado pode estar corrompido ou com indícios de intrusão. Essa palavra é então comparada às linhas de $C_{M\times N}$.

Se até $\left\lfloor \frac{d-1}{2} \right\rfloor$ erros ocorrem, então a decodificação será correta. Caso existam mais de uma classe com a mesma distância mínima da palavra recebida, então uma delas é escolhida aleatoriamente, podendo gerar falsos positivos e falsos negativos. Entretanto, conforme já mencionado, a minimização desse problema está incluído na função de avaliação (Equação 1) no algoritmo genético guiado por código utilizado nessa pesquisa.

4. Resultados

Primeiramente, no processo de discretização e binarização, foram gerados em média, 320 atributos binários para cada experimento. Em seguida, foi investigada a importância desses atributos em relação a cada tipo de ataque. Com base nos resultados obtidos usando árvore de decisão C4.5, os atributos mais relevantes foram determinados, num total de 74. Esses atributos binários correspondem a cada modelo de decisão utilizado nas colunas da matriz C.

Num segundo momento, verificou-se o poder preditivo do conjunto de classificadores binários selecionados na etapa anterior, na tarefa de detectar tipos de ataques, bem como identificar o tráfego normal numa rede de computadores. Como o número de classificadores binários ($k = 74$) não corresponde a um valor padrão de um código BCH, houve a necessidade de se utilizar um código puncionado $(n - l, k, d_p \leq d)$, removendo l bits de redundância, através da remoção de l colunas de sua matriz geradora. Para isso, as linhas da matriz $C_{M\times k}$ foram multiplicadas pela matriz geradora $G_{k\times n}$ de um código $BCH(123, 74, 15)$. Os resultados encontram-se na Tabela 2, coluna 3.

Optou-se também em utilizar um código BCH padrão. Nesse contexto, dentre os 74 atributos, 71 foram selecionados para teste. O critério utilizado foi remover as colunas da matriz código com $d = 1$. Os resultados obtidos com um código $BCH(127, 71, 19)$ encontram-se na Tabela 2, coluna 5.

Na etapa seguinte, foi utilizado um AG guiado por código para encontrar palavras código que melhor se adequassem ao problema proposto. A população inicial do AG é geralmente definida por soluções geradas de maneira aleatória, entretanto, nessa pesquisa, a matriz gerada através do uso de árvore de decisão C4.5, por ser previamente considerada relevante na resolução do problema proposto, foi utilizada pelo AG guiado por código como ponto inicial no processo de busca. Em todas as buscas realizadas pelo referido algoritmo, utilizou-se uma taxa de cruzamento de $0,6$ e uma taxa de mutação de $0,001$. A condição de parada foi o número máximo de iterações.

Foram feitas simulações com AG, usando os códigos $BCH(123, 74, 15)$ e $BCH(127, 71, 19)$. Para realizar a avaliação de cada indivíduo, foi utilizada a função objetivo representada na Equação 1. Os resultados encontram-se na Tabela 2, colunas 4 e 6, respectivamente.

Comparando os resultados preliminares, pode-se observar que a versão de AG guiado por código para a determinação de matrizes códigos foi capaz de gerar soluções com bom desempenho, conforme Tabela 2, com percentual de acerto de até 100% para alguns tipos de ataques, tais como ataques do tipo *buffer-overflow*, além de ataques de negação de serviço, cuja importância se dá ao fato desses ataques serem responsáveis por ocasionar um elevado índice de incidentes de intrusão.

Tabela 2: Resultados obtidos

Categoria	Classes	C4.5_(123, 74, 15)	AG_(123, 74, 15)	C4.5_(127, 71, 19)	AG_(127, 71, 19)
DoS	back	80,02%	99,81%	97,86%	100,00%
	land	0,44%	9,09%	11,39%	14,58%
	neptune	100,00%	100,00%	98,92%	99,88%
	pod	100,00%	100,00%	23,63%	98,55%
	smurf	99,63%	99,99%	0,00%	99,94%
	teardrop	28,30%	100,00%	4,44%	93,75%
Probing	ipsweep	96,38%	99,15%	69,76%	98,98%
	nmap	0,00%	80,65%	3,92%	76,68%
	portsweep	83,02%	96,77%	13,83%	96,20%
	satan	0,07%	5,88%	0,24%	6,25%
	spy	2,61%	100,00%	0,20%	100,00%
R2L	ftp_write	1,72%	37,50%	0,35%	37,50%
	guess_passwd	92,45%	96,23%	91,07%	90,74%
	imap	0,45%	33,33%	0,46%	36,36%
	multihope	0,51%	36,84%	0,38%	16,13%
	phf	0,53%	100,00%	0,23%	100,00%
	warezclient	71,67%	93,33%	10,14%	91,80%
	warezmaster	0,95%	76,19%	0,69%	80,95%
U2R	buffer_overflow	50,00%	100,00%	6,21%	100,00%
	loadmodule	0,32%	30,00%	0,00%	36,36%
	perl	0,43%	100,00%	0,12%	100,00%
	rootkit	0,27%	93,33%	10,14%	28,57%
Normal		46,01%	94,22%	81,83%	91,76%

O modelo proposto apresentou resultado menos significativo, dentro da categoria DoS, na identificação de ataques do tipo *land*, os quais possuem os endereços de origem e destino iguais. Entretanto, esse ataque que pode ocasionar lentidão e atém mesmo colocar em loop infinito versões antigas e desatualizadas de sistemas operacionais, podem ser facilmente detectadas a partir de regras bem definidas. Portanto, não constitui um grande problema a ser tratado.

Embora geralmente as duas estruturas de código BCH tenham alcançado resultados semelhantes, em alguns casos verificou-se uma maior adequabilidade de uma sobre a outra. Confirmou-se nos experimentos que o algoritmo C4.5 podem auxiliar na escolha dos classificadores binários a serem utilizados na matriz código. Entretanto, outras medidas de informação podem ser utilizadas como estratégia de seleção dos classificadores binários.

5. Conclusão

A idéia do modelo de classificação multiclasses baseado em matrizes código, proposto nessa pesquisa, consiste basicamente em relacionar uma palavra código de um código BCH a cada classe que o evento possa assumir, sendo cada classificador responsável por produzir exatamente um bit da palavra código. Foi também proposto o uso de algoritmo genético para a seleção das melhores matrizes. Os resultados obtidos abrem várias possibilidades de pesquisas futuras. Entre os benefícios verificados tem-se uma estrutura que permite o uso de decodificadores algébricos. Além disso, embora a abordagem proposta reduza o espaço das possíveis soluções, o novo espaço de busca é mais estruturado e a distância mínima do código BCH escolhido para a distância mínima entre as palavras-código é inicialmente conhecida, não sendo necessário avaliar este parâmetro durante o processo de busca da matriz código.

Alguns pontos para trabalhos futuros são: introduzir estratégias de codificação e decodificação por listas, em que uma classe seria associada a mais de uma palavra código, buscando melhorar o desempenho do modelo proposto, minimizando o número de classificações desconhecidas; uso de outras medidas de informação a serem utilizadas na seleção dos classificadores binários, tais como as entropias de Rényi e Tsallis [22] [23]; adicionar ao espaço de busca do algoritmo genético o conceito de classes laterais.

Agradecimentos

Os autores agradecem à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) e à Fundação de Amparo à Pesquisa e ao Desenvolvimento Científico e Tecnológico do Maranhão (FAPEMA).

Referências

[1] S. S. V. S. R. Kulkarni, G. Lugosi. "Learning Pattern Classification — A Survey". vol. 44, no. 6, pp. 2178–2206, October 1998.

[2] M. Moreira and E. Mayoraz. "Improved Pairwise Coupling Classification with Correcting Classifiers". In *Proceedings of the 10th European Conference on Machine Learning*, pp. 160–171, London, UK, 1998. Springer-Verlag.

[3] J. Fürnkranz. "Round Robin Classification". *Journal of Machine Learning Research*, vol. 2, pp. 721–747, 2002.

[4] A. C. Lorena. "Investigação de estratégias para a geração de máquinas de vetores de suporte multiclasses". Ph.D. thesis, Instituto de Ciências Matemáticas e de Computação - ICMC-USP, 2006.

[5] E. M. C. Pimenta. "Abordagens para decomposição de problemas multiclasse: os códigos de correção de erros de saída". Master's thesis, Faculdade de Ciência da Universidade do Porto, December 2004.

[6] A. C. Lorena and A. C. P. L. F. de Carvalho. "Evolutionary Design of Code-matrices for Multiclass Problems". *Soft Computing for Knowledge Discovery and Data Mining*, pp. 153–184, 2008.

[7] T. G. Dietterich and G. Bakiri. "Solving multiclass learning problems via error-correcting output codes". *Journal of Artificial Intelligence Research*, vol. 2, pp. 263–286, 1995.

[8] L. M. Moreira. "The use of boolean concepts in general classification contexts". Ph.D. thesis, Escola Politécnica Federal de Lausanne, Suiça, 2000.

[9] S. Escalera, D. M. Tax, O. Pujol, P. Radeva and R. P. Duin. "Subclass Problem-Dependent Design for Error-Correcting Output Codes". *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 30, no. 6, pp. 1041–1054, 2008.

[10] A. Rocha and S. Goldenstein. "Multi-class from Binary - Divide to conquer". In *Computer Vision Theory and Applications*, pp. 323–330, 2009.

[11] C. E. Shannon. "A mathematical theory of communication". *Bell Systems Technical Journal*, vol. 27, pp. 379–423 and 623–656, 1948.

[12] R. E. Blahut. *Theory and Practice of Error Control Codes*. Addison-Wesley Publishing Company, Inc., Owego, NY, 1983.

[13] S. B. Wicker. *Error Control Systems for Digital Communication and Storage*. Prentice-Hall, Inc., 1995.

[14] E. A. Santos, F. M. de Assis and E. C. Gurjão. "Redes de Sensores com Codificação BCH Distribuída". *XXVII Simpósio Brasileiro de Telecomunicações - SBrT 2009*, outubro 2009.

[15] E. L. Allwein, R. E. Schapire and Y. Singer. "Reducing multiclass to binary: a unifying approach for margin classifiers". *J. Mach. Learn. Res.*, vol. 1, pp. 113–141, September 2001.

[16] B. Selman, H. Levesque and D. Mitchell. "A new method for solving hard satisfiability problems". In *Proceedings of the tenth national conference on Artificial intelligence*, AAAI'92, pp. 440–446. AAAI Press, 1992.

[17] "KDD cup 99 Intrusion detection data set". Retrieved March 01, 2010.

[18] "KDD cup 99 Intrusion detection 10_percent_data set". Retrieved March 01, 2010.

[19] U. M. Fayyad and K. B. Irani. "Multi-interval discretization of continuousvalued attributes for classification learning". In *Thirteenth International Joint Conference on Articial Intelligence*, volume 2, pp. 1022–1027. Morgan Kaufmann Publishers, 1993.

[20] J. R. Quinlan. "Induction of Decision Trees". *Machine Learning*, vol. 1, no. 1, pp. 81–106, 1986.

[21] J. R. Quinlan. *C4.5 Programs for Machine Learning*. Morgan Kaufmann Publishers, San Diego, CA, 1993.

[22] C. F. L. Lima, F. M. de Assis and C. P. Souza. "Decision Tree Based on Shannon, Renyi and Tsallis Entropies for Intrusion Tolerant Systems". *Internet Monitoring and Protection, International Conference on*, vol. 0, pp. 117–122, May 2010.

[23] C. F. L. Lima, F. M. de Assis and C. P. Souza. "Árvores de Decisão baseadas nas entropias de Shannon, Rényi e Tsallis para Sistemas Tolerantes a Intrusão". *La Novena Conferencia Iberoamericana en Sistemas, Cibernética e Informática*, Jun 2010.

[24] I. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann Publishers, San Francisco, California, second edition, 2005.

[25] F. M. de Assis. "Genetic algorithms and packing of block codes." *International Conference on Tele-comunications, Proceedings of the ICT97*, vol. 3, pp. 1045–1048, 1997.

[26] F. M. de Assis. "Weight structure of binary codes and the performance of blind search algorithms." *Neural Networks, Brazilian Symposium on, IEEE Computer Society*, vol. 0, pp. 144, 2000.

[27] R. H. Morelos-Zaragoza. *The Art of Error Correcting Coding*. John Wiley & Sons, 2006.