EDUARDO KUGLER VIEGAS

DETECÇÃO DE INTRUSÃO BASEADA EM ANOMALIA PARA AMBIENTES DE PRODUÇÃO

Dissertação apresentada ao Programa de Pós-Graduação em Informática da Pontifícia Universidade Católica do Paraná como requisito parcial para obtenção do título de Mestre em Ciência da Computação.

CURITIBA

2016

EDUARDO KUGLER VIEGAS

DETECÇÃO DE INTRUSÃO BASEADA EM ANOMALIA PARA AMBIENTES DE PRODUÇÃO

Dissertação apresentada ao Programa de Pós-Graduação em Informática da Pontifícia Universidade Católica do Paraná como requisito parcial para obtenção do título de Mestre em Ciência da Computação.

Área de Concentração: *Ciência da Computação*

Orientador: Prof. Dr. Altair O. Santin

CURITIBA

2016

Viegas, Eduardo Kugler
V656d Detecção de intrusão baseada em anomalia para ambientes de produção /
2016 Eduardo Kugler Viegas; Altair O. Santin. -- 2016
66 f. : il. ; 30 cm

Dissertação (mestrado) – Pontifícia Universidade Católica do Paraná, Curitiba, 2016
Bibliografia: f.63-66

1. Informática. 2. Redes de computação (Gerência). 3. Redes de computadores – Confiabilidade, 4. Sistemas de detecção de intrusão (Segurança do computador). I. Santin, Altair Olivo. II. Pontifícia Universidade Católica do Paraná. Programa de Pós-Graduação em Informática. III. Título.

CDD 20. ed. – 004.068

Pontifícia Universidade Católica do Paraná

PUCPR
GRUPO MARISTA

ATA DE DEFESA DE DISSERTAÇÃO DE MESTRADO
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

DEFESA DE DISSERTAÇÃO DE MESTRADO Nº 16/2016

Aos 29 dias do mês de Fevereiro de 2016 realizou-se a sessão pública de Defesa da Dissertação **" Detecção de Intrusão Baseada em Anomalia para Ambientes de Produção"** apresentado pelo aluno **Eduardo Kugler Viegas,** como requisito parcial para a obtenção do título de Mestre em Informática, perante uma Banca Examinadora composta pelos seguintes membros:

Prof. Dr. Altair Olivo Santin
PUCPR (Orientador) _____ (assinatura) Aprov. (Aprov/Reprov)

Prof. Dr. Luiz Eduardo Soares Oliveira
UFPR _____ (assinatura) APROV (Aprov/Reprov)

Prof. ªDr.ª Michele Nogueira
UFPR _____ (assinatura) Aprov. (Aprov/Reprov)

Conforme as normas regimentais do PPGIa e da PUCPR, o trabalho apresentado foi considerado Aprovado (aprovado/reprovado), segundo avaliação da maioria dos membros desta Banca Examinadora. Este resultado está condicionado ao cumprimento integral das solicitações da Banca Examinadora registradas no Livro de Defesas do programa.

Prof.ª Dr.ª Andreia Malucelli.
Coordenadora do Programa de Pós-Graduação em Informática.

Programa de Pós-Graduação em Informática
PPGIa
PUCPR

Dedico este trabalho aos meus familiares que sempre me incentivaram e aconselharam ao longo de minhas conquistas.

Agradecimentos

Agradeço primeiramente a minha família pelo apoio incondicional provido ao longo deste trabalho. Agradeço ao meu orientador, Dr. Altair O. Santin, pelo apoio e confiança em mim depositada desde a graduação, pelos conselhos e oportunidades que me fizeram crescer pessoal e profissionalmente durante esta jornada. Agradeço aos meus amigos do SecPLab por todos os dias e noites de estudos no laboratório. Agradeço ao Cleverton, Rafael e Vilmar que se tornaram grandes amigos e companheiros ao longo desta caminhada. Agradeço a minha amada namorada pelo apoio incondicional. Agradeço a Intel por viabilizar esta pesquisa e me permitir crescer profissionalmente através de nossos encontros. Agradeço aos meus colegas da UTFPR, André, Ricardo e Volnei, agradeço também ao Luiz Eduardo S. Oliveira da UFPR pela orientação.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

DoS	Denial-of-Service
HIDS	Host-based Intrusion Detection System
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
kNN	k-Nearest Neighbors
NIDS	Network-based Intrusion Detection System
NTP	Network Time Protocol
R2L	Remote-to-Local
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
SSH	Secure Shell
SVM	Support Vector Machine
TCP	Transmission Control Protocol
U2R	User-to-Root
UDP	User Datagram Protocol

Resumo

A detecção de intrusão (IDS, *Intrusion Detection System*) baseada em anomalia é amplamente estudada. Apesar dos resultados promissores na literatura, dificilmente é utilizada em ambientes de produção (ambiente real). Isso ocorre devido à diferença entre a taxa de acerto obtida durante o treinamento do modelo de ataque e a observada durante a sua utilização em produção. Neste trabalho é proposto um novo método de criação de bases de intrusão que objetiva garantir os seguintes aspectos: atualização, reprodução e geração de tráfego real, válido e representativo. Adicionalmente, é apresentado um novo método de avaliação de modelos de ataque e de rejeição, baseado na distribuição atual dos atributos para garantir a classificação de forma confiável. Foram desenvolvidas 16 bases, cada uma validando as premissas comumente utilizadas na literatura. Os resultados mostram que a maioria das premissas usualmente utilizadas para o desenvolvimento de um *Network-based*-IDS não se sustentam quando a abordagem de aprendizagem de máquina é utilizada em cenários onde o conteúdo do tráfego muda. Entretanto, o método de rejeição permite garantir a confiabilidade do IDS independentemente do cenário considerado.

Palavras-Chave: *Classificador baseado em anomalia; Detecção de Intrusão; Confiabilidade da classificação; Bases de Intrusão.*

Abstract

The anomaly-based intrusion detection system (IDS, Intrusion Detection System) is extensively studied. Besides the promising results reported in the literature, it is hardly used in production environments (real environments). Such factor occurs mainly due to the differences between the accuracy rate obtained during the attack model training and the accuracy rate observed during its usage in production. In this work, it is proposed a new intrusion database creation method, which aims at ensuring the following properties: update, reproduction and guarantee that the traffic generated is real, valid and representative. Additionally, it is presented a new evaluation model and a rejection method (classification reliability), based on the current feature distribution to ensure a reliable classification. A total of 16 databases were created, each one evaluating the common premises used in the literature. The results show that most of the premises usually adopted to the development of a Network-based-IDS does not hold when the machine learning approach is used in scenarios where the network traffic content changes. However, the proposed rejection method allowed to ensure the IDS reliability regardless of the considered scenario.

Keywords: *Anomaly-based Classifier; Intrusion Detection; Classification Reliability; Intrusion Databases.*

Capítulo 1

Introdução

O relatório anual de ameaças [1] relata a detecção de 6549 novas vulnerabilidades em 2014. A média anual desde 2006 supera 4600 ocorrências, o que representa mais de 12 novas vulnerabilidades identificadas diariamente [2], indicando que qualquer sistema conectado à internet está potencialmente exposto a esta quantidade de ameaças.

A detecção de ataques, utilização maliciosa ou inadequada de um sistema computacional ou de uma rede de computadores, pode ser realizada através dos sistemas de detecção de intrusão (*Intrusion Detection System* - IDS) [3]. Dentre os tipos de detecção relacionados na literatura [4], os IDS baseados em anomalia prometem a detecção de novos ataques. Para tanto, a detecção baseada em anomalia em um IDS é principalmente tratada como um problema de reconhecimento de padrões, através da utilização de técnicas de aprendizagem de máquina.

A técnica de aprendizagem de máquina utilizada por um IDS normalmente utiliza um algoritmo de inferência (classificador) e um modelo que representa os ataques. O processo consiste em inferir um comportamento a partir de uma base de eventos capturados da rede, obtendo-se o modelo do atacante. O modelo do atacante é utilizado para detecção de intrusão em ambientes de produção (ambientes reais, em que o sistema será utilizado). Esse modelo é aprendido a partir de uma entrada composta de um conjunto de eventos (pacotes de rede, por exemplo) e um conjunto determinado de atributos (características). Neste contexto, os atributos derivam de campos dos pacotes da rede. Porém, eventos de classes diferentes (normal ou ataque) com comportamentos similares, podem ser classificados de forma errônea pelo IDS. Para reduzir erros dessa natureza (falsos-negativos/positivos) a acurácia do classificador deve

ser avaliada a partir de uma base de testes.

Ao aplicar a tecnologia de IDS para detecção de ataques de rede, é esperado que a acurácia do classificador, previamente obtida no processo de teste do modelo, seja a mesma obtida durante sua aplicação em um ambiente de produção. Desse modo, a base de eventos empregada no momento do teste do IDS deve representar de forma precisa o comportamento da rede (em produção).

O trabalho de Mahbod Tavallaee [5] constatou que em torno de 50% dos trabalhos da literatura fizeram uso da base DARPA1998 [6] ou bases similares para testes. DARPA1998 foi desenvolvida em 1998 e atualizada em 2000. Dessa forma, os resultados obtidos através de sua utilização são questionáveis, uma vez que o tráfego de rede sofre constantes mudanças. Novos serviços e ataques, que são reportados diariamente, são exemplos disso.

Na literatura existem diferentes propostas de bases de teste de IDS [5], cuja abordagem, em sua maioria, é deficitária em algum aspecto: não permite reprodutibilidade, representatividade, ou ainda não estão disponíveis publicamente [6], impedindo o seu emprego para comparação com outros trabalhos.

A base de eventos pode ser desenvolvida através de duas abordagens: (i) monitoramento do ambiente de produção ou (ii) criação em um ambiente controlado. O monitoramento do ambiente de produção permite obter uma base de eventos representativa. Porém, o compartilhamento público (em formato nativo) é inviável sem violar a privacidade. A criação de uma base em ambiente controlado através de ferramentas de geração de tráfego permite o seu compartilhamento, porém ela pode sofrer de problemas de invariabilidade do tráfego.

O trabalho de Sommer e Paxson [8] destaca a falta de aplicabilidade real dos resultados reportados na literatura e ressalta: a ausência de métodos de avaliação coerentes e a baixa utilização de IDS baseado em anomalia nos ambientes de produção. Os autores apontam a baixa confiabilidade da acurácia do classificador como principal razão de suas críticas. Adicionalmente, quando um IDS é empregado em produção, os alarmes não são necessariamente ataques (falso-positivos).

Os problemas relatados anteriormente dificultam a utilização de IDS baseado em anomalia em produção. Esta é a razão da maioria dos IDSs de uso comercial utilizarem genuinamente a abordagem baseada em assinatura [8]. O método baseado em assinatura compara cada evento recebido com uma base de assinaturas (ataques conhecidos), que funciona bem para ataques conhecidos, porém é falha para a detecção de variações ou novos ataques.

O relato de Mahbod Tavallaee [5] afirma que somente 7% dos trabalhos encontrados na literatura realizaram testes de acurácia do classificador em produção. Embora existam diferentes métodos para obter o modelo do ataque utilizando IDSs, nenhuma das abordagens considera que a acurácia estimada no momento do treinamento pode ser diferente em produção.

1.1. Motivação

Devido à crescente quantidade de vulnerabilidades conhecidas, torna-se necessária a utilização de uma técnica de detecção de intrusão que permita a identificação de novos ataques em face a mudanças no cenário. A motivação deste trabalho se concentra em solucionar os problemas relativos à utilização de técnicas para detecção confiável de intrusão baseadas em anomalia em ambientes de produção, mesmo em cenários que mudam de comportamento de forma repentina.

A detecção de intrusão baseada em anomalia é amplamente estudada na literatura [3], porém dificilmente é aplicada a ambientes de produção. Nota-se que os resultados reportados, como altas taxas de detecção e baixo custo computacional não são evidenciados quando o IDS baseado em anomalia é utilizado em produção [8].

Diferentes abordagens para criação de bases de intrusão são propostas na literatura. Porém, de maneira geral, essas não garantem as propriedades esperadas: possuir tráfego real e válido; apresentar eventos previamente classificados (rotulados); apresentar diversidade de tráfego; aplicar ataques corretamente implementados (seguindo uma especificação comum); apresentar facilidade de atualização; ser reprodutível, e poder ser publicamente disponível.

Assim, o principal problema não abordado pela literatura é a ausência de métodos para testar corretamente um IDS baseado em anomalia, considerando as propriedades da rede de ambientes de produção. De modo geral, os autores da literatura utilizam bases de intrusão antigas como o DARPA1998 para a obtenção dos modelos e assumem que a acurácia obtida, sobre o tráfego antigo, permanecerá válida durante seu uso em produção, mesmo que o tráfego mude. Outro fator não abordado na literatura consiste na utilização do método de teste (avaliação) tradicional de aprendizagem de máquina para IDS, sem considerar que ambientes de rede, diferentemente de outros domínios de aplicação, pode apresentar variabilidades importantes de comportamentos nos conteúdos de seu tráfego.

1.2. Objetivo Geral

Este trabalho tem como objetivo geral o desenvolvimento de um conjunto de métodos que possibilite a utilização de um IDS baseado em anomalia com a utilização de classificadores em ambientes de produção de forma confiável. Baseado em um método para o desenvolvimento de bases de intrusão, objetiva-se criar um conjunto de cenários que represente as propriedades do ambiente de produção em ambiente controlado. Finalmente, um método que visa garantir a confiabilidade das classificações dos modelos é apresentado, tornando a abordagem baseada em anomalia com a utilização de classificadores confiável, mesmo em um cenário com mudanças de tráfego. Assim, pretende-se aferir a capacidade de detecção dos modelos, considerando as propriedades da rede e garantindo que os modelos sejam capazes de classificar os eventos de forma precisa e confiável.

1.3. Objetivos Específicos

O objetivo geral se desdobra nos seguintes objetivos específicos.

a) Definir um método para criação de bases de intrusão para testes de IDS, que apresenta as propriedades esperadas de uma base de intrusão;

b) Definir o conjunto de atributos a serem extraídos do tráfego de rede para treinamento de modelos de ataques, utilizando a literatura atual como base;

c) Definir um método de teste para avaliar os modelos dos classificadores conforme as propriedades de rede em ambientes de produção;

d) Criar um conjunto de cenários usando o método de criação de bases de intrusão proposto que permita avaliar o método de teste proposto;

e) Definir um método de rejeição a fim de garantir a confiabilidade das classificações;

f) Obter um conjunto de modelos para avaliação da proposta de teste e da confiabilidade;

g) Comparar a abordagem proposta com uma ferramenta de mercado.

1.4. Contribuições

Este trabalho fornece subsídios para o estudo e a criação de um conjunto de métodos

que permite a utilização de um IDS baseado em anomalia, com a utilização de classificadores em ambientes de produção de forma confiável. Especificamente, este trabalho apresenta as seguintes contribuições:

- Um método baseado em ferramenta para a criação de tráfego normal real e válido em um ambiente controlado e reprodutível para a criação de bases de intrusão. O método proposto permite o compartilhamento da base, uma vez que os dados não são sensíveis. O problema de invariabilidade, inerente à criação de bases em um ambiente controlado, é reduzido através da criação de comportamentos complexos. Problemas inerentes à criação do tráfego do atacante são resolvidos através da utilização de ferramentas conhecidas e padronizadas, e assim são resolvidos os principais problemas apontados pelos autores [8][14] em relação à criação de bases de intrusão;
- Um método robusto de avaliação específico para aprendizagem de máquina aplicada ao IDS baseado em anomalia é proposto e testado, através do método indicado para criação de bases de intrusão. O método de avaliação proposto provém taxas sobre cada propriedade da rede em ambientes de produção;
- Um método de rejeição baseado na distribuição dos atributos é proposto e avaliado a fim de garantir a confiabilidade na classificação. Através do método de rejeição proposto, torna-se possível garantir a confiabilidade da classificação em ambientes de produção;
- Uma comparação da abordagem proposta com a abordagem de detecção utilizada em ambientes de produção;

1.5. Estrutura do Documento

O Capítulo 2 apresenta a fundamentação teórica na qual este trabalho se baseia. O Capítulo 3 discorre sobre o estado da arte, com a exposição de trabalhos relacionados ao tema. O Capítulo 4 enuncia a proposta, enquanto o Capítulo 5 apresenta a sua avaliação. Por fim, no Capítulo 6 algumas conclusões são apresentadas.

Capítulo 2

Fundamentação Teórica

Um sistema de detecção de intrusão (IDS – *Intrusion Detection System*) possibilita ao administrador do sistema identificar possíveis quebras de políticas de segurança [3], permitindo que as ações necessárias sejam tomadas e que a quebra de política não comprometa o restante do sistema. Este capítulo abordará os sistemas de detecção de intrusão, a utilização de técnicas de aprendizagem de máquina para detecção baseada em anomalia e, por fim, discorrerá sobre as bases de treinamento e suas características esperadas.

2.1 Sistema de Detecção de Intrusão

Sistemas de detecção de intrusão (IDS) são responsáveis por identificar e classificar atividades maliciosas em um ambiente computacional e, se necessário, emitir alertas. Tipicamente, um IDS é composto por quatro módulos: (i) *Aquisição dos dados*: captura eventos de um determinado ambiente para inspeção, e.g. pacotes da rede; (ii) *Pré-processamento*: efetua o processamento para formatação do evento a ser classificado, e.g. extração de um conjunto de atributos do pacote da rede; (iii) *Detecção*: classifica o evento atribuindo-lhes uma classe (e.g. normal ou ataque); e (iv) *Alerta*: registra em arquivo ou na interface do usuário (operador) os eventos classificados como ataque.

Em meio a diversas taxonomias propostas na literatura para classificação de tentativas de intrusão, a mais utilizada é o método de Kendall [6], proposto para a base DARPA1998 [11]. Kendall [6] categorizou os ataques e intrusões em quatro categorias:

- *Probing*: ataques que se destinam a obter informações sobre o sistema a fim de identificar possíveis vulnerabilidades. Esses ataques incluem *sniffing* de tráfego e varredura de portas e de endereços de rede. Exemplos: *ip sweep*, *mscan, nmap, saint* e *satan.*

- *Denial of Service (DoS)*: ataques que tornam a memória, ou outro recurso computacional, ocupada ou tão saturada que usuários legítimos do sistema não conseguem acessar ao serviço provido. Exemplos: *apache2, back, land, mailbomb, synflood, pingo f death, process table, smurf, syslogd, teardrop* e *udpstorm*.

- *User to Root (U2R):* o atacante, que possui uma conta legítima no sistema, explora uma vulnerabilidade para obter acesso a uma conta administradora. Exemplos: *eject, ffbconfig, fdfformat, loadmodule, perl, ps* e *xterm.*

- *Remote to Local (R2L):* obter acesso a uma conta do sistema, explorando alguma vulnerabilidade presente no mesmo. Exemplos: *dictionary, ftp-write, guest, imap, named, phf, sendmail, xlock* e *xsnoop.*

Os sistemas de detecção de intrusão podem ser baseados em rede (*Network-based Intrusion Detection System* - NIDS) ou baseados em host (*Host-based Intrusion Detection System* - HIDS). O NIDS detecta ataques como *probing* e DoS (e.g. ataques massivo de requisições). Entretanto, um NIDS acessa apenas ao conteúdo em nível de rede. Já um HIDS detecta ataques em nível de aplicação, como R2L (e.g. *buffer overflow*) ou U2R. (e.g. *privilege escalation*).

Quando o módulo de detecção é baseado em assinaturas, a detecção de ataques é efetuada através da varredura por padrões de ataques conhecidos. A base de assinaturas é consultada para cada evento, demandando um mecanismo aprimorado de busca para a consulta, com todos os padrões de ataques conhecidos. Quando qualquer ataque conhecido é encontrado, o evento é classificado como intrusão. Então, quando um novo ataque é publicamente reportado, este será armazenado na base de assinaturas. A principal desvantagem da abordagem de detecção baseada em assinatura é a sua incapacidade de detectar ataques desconhecidos (não presentes na base de assinaturas).

A abordagem baseada em anomalia, por outro lado, utiliza um modelo de ataque que pode ser utilizado na detecção de novos ataques. A detecção de um ataque ocorre no momento que o mecanismo de classificação identifica similaridades entre o modelo de ataque e os

atributos do evento em avaliação. Através do vetor de atributos, o classificador avalia cada evento, dando entrada no mecanismo de classificação, atribuindo-lhe uma classe. Essa ação permite que um evento com conteúdo previamente desconhecido seja classificado corretamente e para isso é necessário que exista uma relação entre os atributos e o modelo de ataque. O processo geral é representado na Figura 1.

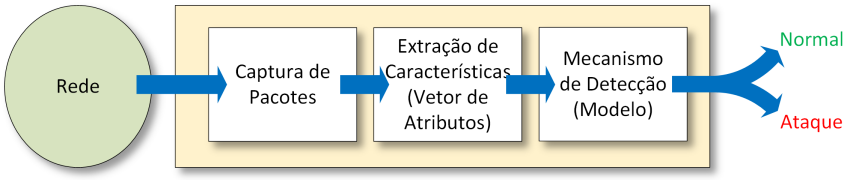

Figura 1 – Arquitetura típica de um NIDS baseado em anomalia.

O processo típico de um NIDS baseado em anomalia com a utilização de classificadores pode ser dividido em quatro etapas distintas (Figura 1). Inicialmente, a *Captura de Pacotes* é realizada através do monitoramento da rede em questão. Posteriormente, os pacotes adquiridos da rede são enviados ao módulo de *Extração de Características,* que efetua a devida extração de atributos para a formação de um vetor de atributos. Com o vetor de atributos devidamente extraído, o *Mecanismo de Detecção* utiliza o modelo de ataque, previamente obtido, para determinar se o pacote da rede é uma tentativa de intrusão ou uma atividade legítima da rede.

2.2 Aprendizagem de máquina

Na literatura, as tentativas de intrusão são majoritariamente tratadas como um problema de reconhecimento de padrões [5] e, em sua maioria, utiliza a aprendizagem de máquina para classificar os eventos da rede. A aprendizagem de máquina visa atribuir uma classe a um evento (Figura 1), normal ou ataque, por exemplo. Em um NIDS, a aprendizagem de máquina é utilizada para identificar se uma requisição específica na rede é legítima ou se é uma tentativa de intrusão. Para obtenção de um modelo que represente intrusão (modelo do atacante), alguns passos são necessários para o desenvolvimento do classificador para NIDS, sumarizado na Figura 2.

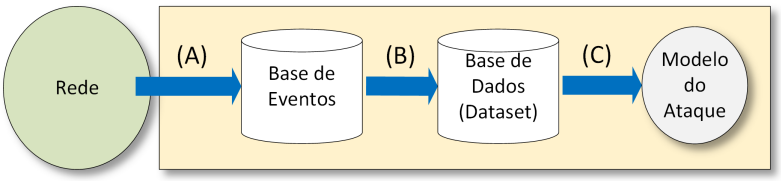

Figura 2 – Processo típico para obtenção do Modelo do Ataque utilizado em um NIDS tradicional.

O processo inicia com a captura dos pacotes da rede e o seu armazenamento em uma base de eventos (Figura 2, etapa A), esta base de eventos deve representar o comportamento do ambiente em que o sistema será utilizado em produção. Posteriormente, com a base de eventos já armazenada, os vetores de características são extraídos de cada evento (evento em um NIDS se refere a qualquer unidade a ser classificada pelo classificador, e.g. datagrama UDP, mensagem ICMP ou segmento TCP). Os vetores de características juntamente com a classe de cada evento compõem a base de dados (*dataset*) (Figura 2, etapa B). Finalmente, um algoritmo de aprendizagem de máquina é utilizado para inferir um padrão e criar um modelo, que represente o conjunto de atributos extraídos (Figura 3, etapa C).

De forma geral, durante o processo de obtenção do modelo do ataque, três bases de dados são utilizadas para treinamento, teste e validação. A base de treinamento é utilizada para a obtenção do modelo do ataque, enquanto a base de teste é utilizada para efetuar possíveis melhorias ao modelo do ataque. Com a obtenção do modelo final, este é utilizado para verificar a acurácia do sistema através da base de validação, e então uma taxa de falso-positivo e falso-negativo pode ser obtida. A taxa de falso-positivo indica a quantidade de eventos legítimos (normal) que foram classificados como tentativas de intrusão pelo modelo. Por outro lado, a taxa de falso-negativo indica a quantidade de tentativas de intrusão (ataque) que foram classificados como eventos legítimos. Uma abordagem para garantir a confiabilidade da classificação é através da rejeição. A rejeição visa rejeitar saídas dos classificadores potencialmente erradas, de acordo com uma métrica prédefinida.

É importante ressaltar que o conjunto de atributos extraídos (Figura 1, Extração de Características) devem ser discriminantes, ou seja, devem permitir a correta identificação da classe que o evento pertence.

2.2.1 Classificadores (Modelo do Ataque)

O objetivo de um algoritmo de aprendizagem de máquina é prever o comportamento do problema, baseando-se no modelo do algoritmo. Certos algoritmos de aprendizagem de máquina são utilizados no campo de detecção de intrusão e os critérios que devem apresentar são: boa acurácia, baixo custo computacional ou menor complexidade na atualização do modelo. Dentre os algoritmos utilizados, destacam-se a árvore de decisão, naive bayes, k vizinhos mais próximos e máquina de vetores de suporte [3].

2.2.1.1 Árvore de decisão

A árvore de decisão é um algoritmo de classificação que executa uma série de instruções no formato SE-ENTÃO. O processo de classificação consiste em percorrer a árvore até que um nó folha, associado à uma classe, seja alcançado. Cada nó da árvore especifica um teste de uma característica, e o ramo correspondente especifica um valor possível da característica [38]. A Figura 3 exibe um exemplo de árvore de decisão para detecção de ataques *DoS*.

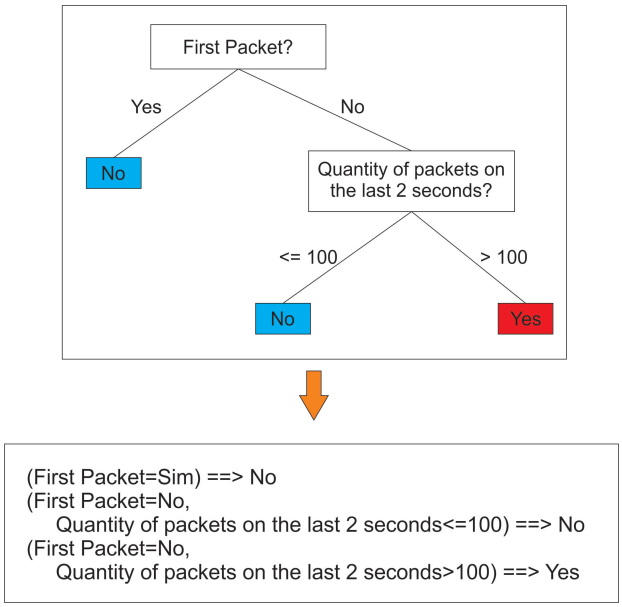

Figura 3 – Árvore de decisão exemplo para classificação de ataques de *DoS*.

Como o processo de classificação envolve apenas um conjunto de verificações, a árvore de decisão torna-se adequada para a classificação em tempo real.

2.2.1.2 Naive Bayes

O classificador Naive Bayes é um algoritmo baseado em probabilidade que parte do pressuposto de que as características são condicionalmente independentes, o que significa que a ocorrência de uma determinada característica não afeta a probabilidade de ocorrência de outra característica. O modelo consiste em uma tabela de probabilidades para cada característica, inferidas a partir da base de treinamento. No processo de classificação, a probabilidade a priori de cada classe, que não é dependente das características, é multiplicada pela probabilidade individual de cada característica. A classe com maior probabilidade é atribuída ao evento a ser classificado [39].

2.2.1.3 k-Vizinhos mais próximos (kNN – k-Nearest Neighbors)

O classificador kNN não requer um modelo e a base de treinamento é utilizada como referência. Um evento é classificado baseado na similaridade com as instâncias de treinamento. Durante a etapa de classificação, os K vizinhos mais próximos são determinados de acordo com uma métrica de distância específica, tal como a distância euclidiana. A classe com maior ocorrência entre os K vizinhos é escolhida como classe do evento sendo classificado [40]. O processo é exibido na Figura 4.

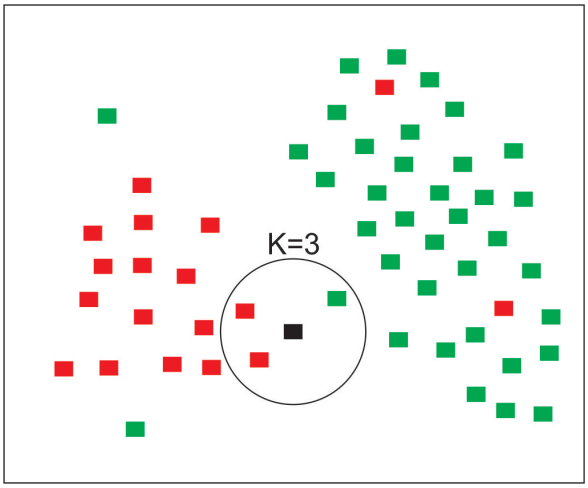

Figura 4 – KNN exemplo utilizando 3 vizinhos para cálculo da classe.

Como o classificador deve calcular a distância entre o evento com todas as instâncias da base de treinamento, o tempo de processamento é diretamente proporcional ao tamanho da

base de treinamento. Para prevenir que características com um valor absoluto maior possuam maior influência no cálculo de distância, as características devem ser normalizadas.

2.2.1.4 Máquina de Vetores de Suporte (SVM – Support Vector Machine)

O classificador SVM é um algoritmo de aprendizagem que visa mapear os exemplos da base de treinamento em um hiperplano. O mapeamento é efetuado através de uma função denominada *kernel* [41]. A Figura 5 ilustra o processo de mapeamento.

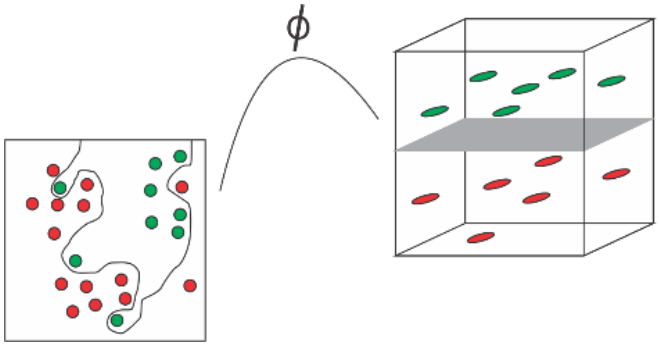

Figura 5 – Mapeamento das instâncias de treinamento em um hiperplano.

A função do mapeamento define uma fronteira entre as classes. O processo de classificação é efetuado através de vetores de suporte que definem a fronteira entre as classes que deve ser a mais ampla possível. Um exemplo de função *kernel* é o *Radial Basis Function*, e a sua fórmula é definida por:

$$k(x, y) = exp(-\gamma|x - y|^2)$$

A função pode ser definida como o quadrado da distância euclidiana entre dois vetores de características.

2.3 Base de Treinamento

Em um NIDS, a base de treinamento com o tráfego da rede é essencial para a correta obtenção do modelo. Esperam-se as seguintes propriedades de uma base de treinamento utilizada para um NIDS baseado em anomalia [7]:

- *Real*: apresentar o tráfego que pode ser identificado em ambientes de produção, em termos de conteúdo e variabilidade;
- *Válido*: conter pacotes corretamente formados resultantes da interação cliente-servidor do ambiente em questão;
- *Previamente classificada*: os eventos dispostos na base devem estar previamente e corretamente classificados, a fim de permitir a correta obtenção do modelo;
- *Diversa*: apresentar grande variabilidade de serviços, comportamentos dos clientes e dos ataques;
- *Correta*: apresentar ataques que sigam um padrão conhecido e aceito pela comunidade da área (mesmo que '*de facto*');
- *Facilmente atualizável*: permitir a inserção de novos serviços e ataques de forma fácil e simples;
- *Reprodutível*: garantir a possibilidade de reprodução para fins de comparação com outros resultados reportados;
- *Pública*: permitir o compartilhamento livre entre a comunidade;

2.3.1 Métodos de Criação de Bases

Um dos principais problemas para a adoção de um NIDS baseado em anomalia é a(s) métrica(s) de testes utilizada(s) durante a sua concepção. Normalmente, adota-se a acurácia do sistema, obtida durante o processo de avaliação, como um fator que garante que o IDS será capaz de atuar em produção, mantendo assim a taxa obtida durante o treinamento do sistema. Dessa maneira, assume-se que o comportamento do ambiente utilizado durante o treinamento do modelo (Figura 2) será o mesmo evidenciado durante a utilização do NIDS em produção. De forma geral, quatro abordagens podem ser utilizadas durante a obtenção da base de treinamento (Figura 2) [12]:

a) *Sem a utilização do tráfego normal*.

A geração do tráfego normal/legítimo de um ambiente é uma tarefa complexa, o comportamento do cliente muda constantemente demandando tempo e a constante remodelagem, uma vez que surgem novos ataques e serviços diariamente. Dessa maneira, alguns testes não utilizam tráfego normal durante a avaliação do IDS, permitindo a fácil

rotulação do tráfego, uma vez que, apenas o tráfego do atacante é gerado, estabelecendo facilmente a taxa de acerto do IDS.

O ambiente de reprodução consiste em um host ou em uma rede em que não há atividade legítima, apenas ataques são gerados, e todos devem ser identificados pelo IDS. Essa abordagem é facilmente implementada e o tráfego é automaticamente rotulado, além de não haver a necessidade de sanitização (remoção de informação sensível do tráfego) ou da geração do tráfego legítimo.

Tal abordagem parte do pressuposto que o IDS é capaz de detectar os ataques independentemente da existência ou não de atividade legítima. Sabe-se que à medida que a taxa da rede (banda) em que o IDS está operando aumenta, a performance do sistema diminui. Isso acarreta a perda de pacotes pela biblioteca utilizada para a coleta. Apenas a taxa de falso-negativo é estabelecida, a taxa de falso-positivo não é determinada, visto que não há atividade legítima no tráfego.

b) *Utilização de tráfego real*

Para estabelecer taxas de falso-positivo e falso-negativo, a utilização de tráfego real pode ser considerada. Esta abordagem é bem aceita, uma vez que captura a diversidade do ambiente de produção, já que o tráfego é real e a diversidade do tráfego é representada de forma correta.

O ambiente de reprodução normalmente consiste na reprodução do tráfego real, parte-se do pressuposto que o tráfego é livre de ataques, ou seja, composto apenas de tráfego normal, e posteriormente o tráfego do atacante é injetado na rede. A rotulação também pode ser efetuada de forma automática, visto que a origem dos ataques é conhecida.

De qualquer forma, diversos pontos devem ser considerados ao utilizar-se do tráfego real. É difícil a reprodução dos testes e torna-se tecnicamente inviável a reprodução de grandes ambientes, devido à taxa de transmissão exigida da rede. Normalmente utiliza-se um conjunto limitado de máquinas utilizadas para a geração dos ataques, assim como as máquinas alvo, logo um IDS pode identificar que apenas algumas máquinas estão sendo atacadas e assim melhorar de forma errônea a sua taxa de detecção.

Não há como garantir que a taxa de falso-positivo está correta e então pressupõe-se que o tráfego utilizado é livre de ataques, já que não é possível realizar de forma manual a limpeza do tráfego reproduzido. Dificilmente é possível publicar a base utilizada para comparações com

os trabalhos da literatura, isso ocorre devido a questões de privacidade, uma vez que, há a possibilidade de haver dados sensíveis no tráfego. Caso sejam utilizadas técnicas de higienização para a remoção de dados sensíveis, isso pode alterar os dados e fazer com que o IDS não possua uma taxa real.

c) *Utilização de tráfego real higienizado*

Para permitir a livre distribuição do tráfego utilizado durante os testes de um IDS para a comunidade, a higienização do tráfego torna-se necessária. Esse processo pretende remover quaisquer informações sensíveis ou privadas do tráfego real, mantendo assim a privacidade. Uma forma de higienização, por exemplo, é a alteração dos cabeçalhos dos protocolos, como a mudança do endereço de origem e destino do cabeçalho IP. A higienização do tráfego é uma tarefa complexa, já que não apenas o cabeçalho deve ser inspecionado, mas também os dados existentes (*payload*), demandando assim o conhecimento dos protocolos de aplicação existentes no tráfego gravado.

A remoção de informações sensíveis pode acabar removendo muita informação, tornando o tráfego resultante irrealista. Por outro lado, a possibilidade de vazamento de uma informação sensível acaba tornando inviável a distribuição do tráfego à comunidade.

d) *Geração do tráfego em ambiente controlado*

Esta abordagem é comumente utilizada para realizar testes em IDS e objetiva criar o tráfego em um ambiente controlado com hosts e uma rede que pode ser requisitada e atacada de forma livre pelo pesquisador.

O ambiente reproduzido contém uma série de máquinas alvo que serão atacadas durante os testes. Cada máquina possui o sistema operacional e os serviços de acordo com a necessidade do pesquisador. A principal dificuldade dessa abordagem é a representação do comportamento do usuário legítimo do ambiente. Os serviços disponíveis devem ser reproduzidos e as requisições efetuadas pelos clientes atendidas de forma válida e variável, a fim de representar o comportamento distinto do cliente. Isso é alcançado normalmente através da utilização de complexos *scripts* criados a fim de representar o comportamento do cliente durante os testes. Estes scripts geram o tráfego por parte do cliente a um conjunto de serviços e recursos disponíveis em cada *host*. Normalmente utiliza-se um serviço real hospedado em um *host* no ambiente controlado. Tal serviço é responsável pela análise e resposta das requisições efetuadas

durante os testes. Finalmente, o tráfego do atacante também é gerado durante os testes, e a rotulação dos eventos pode ser facilmente efetuada de acordo com o método utilizado pelo pesquisador.

A reprodução dos testes torna-se possível, uma vez que o tráfego pode ser gravado e repetido. A distribuição do tráfego à comunidade científica também se torna possível, visto que não há dados sensíveis presentes. Como todo o tráfego é gerado de forma controlada, garante-se que o tráfego do cliente é livre de ataques, assim como a identificação dos tipos de ataques. Por outro lado, esta abordagem possui alguns pontos negativos. Primeiramente é custoso e difícil desenvolver um ambiente de simulação, depois deve-se definir quais serviços serão reproduzidos, assim como o tipo de requisição para cada serviço disponível, e os ataques a serem gerados também devem ser considerados. O principal problema é a variabilidade do tráfego gerado, pois toda requisição a um serviço específico é previamente definida e programada. À medida em que o ambiente é reproduzido, o tráfego tende a se repetir e, dessa maneira, influenciar nos resultados obtidos pelo IDS.

2.4 Avaliação do Modelo

A área de NIDS baseado em anomalia enfrenta dificuldade em transformar os resultados relatados na literatura em dados reais em um ambiente de produção. A base de treinamento define não apenas o ambiente em que o NIDS irá executar, mas também a disposição dos serviços. A correta concepção da base é vital para o desenvolvimento de um NIDS baseado em anomalia. Atualmente existem poucas bases de intrusões públicas disponíveis na literatura e as bases disponíveis não representam de forma correta o tráfego atualmente evidenciado.

A escolha correta dos atributos utilizados pelo sistema assim como a disposição do ambiente e do tráfego são cruciais para a concepção do IDS. A necessidade de estabelecer diversas métricas de desempenho torna-se evidente em um IDS baseado em anomalia. É conhecido que o tráfego muda constantemente e os ambientes de produção variam, estabelecer apenas a acurácia, ou qualquer outra métrica relativa a um cenário específico obtido durante o teste do IDS é insuficiente para garantir a adoção do sistema.

O especialista deve considerar: (i) a constante mudança do tráfego normal de acordo com o serviço utilizado e o perfil do cliente; (ii) o surgimento diário de novos ataques e um modelo do classificador capaz de identificar essas novas ameaças de forma confiável; (iii) a

disposição do ambiente, a quantidade de clientes requisitando os serviços disponíveis e a vazão de rede gerada, pois a utilização de métodos de classificação que demandam uma grande capacidade de processamento podem tornar a detecção de intrusão em tempo real inviável; (iv) a comparação de resultados com a literatura, pois a utilização de bases de intrusão privadas impedem a comparação e tornam os resultados obtidos questionáveis; e (v) a outras ferramentas de mercado, pois comparativamente deve-se fornecer não apenas métricas de avaliação mas também demonstrar que a abordagem baseada em anomalia permite, se devidamente concebida, efetuar a detecção de intrusão quando comparada a abordagem baseada em assinatura.

Capítulo 3

Trabalhos Relacionados

Os trabalhos relacionados encontram-se segmentados nas duas seções que seguem. A primeira seção trata dos trabalhos relacionados à criação de bases de intrusão, enquanto a segunda seção aborda os métodos de avaliação comumente utilizados.

3.1 Criação de bases de intrusão

Atualmente a base de intrusão mais utilizada na literatura [5] é o DARPA1998 [13], considerada a primeira tentativa de criação de uma base de intrusão para testes/geração de IDS de forma pública. A iniciativa foi concebida devido à dificuldade encontrada na literatura para comparação dos resultados obtidos. Conforme discutido na seção 2.3.1, a divulgação de tráfegos obtidos a partir do monitoramento de ambientes reais tornou-se inviável devido a questões de privacidade inerentes aos dados gravados. Por essa razão, todos os dados inerentes ao DARPA1998 foram gerados de forma sintética em um ambiente controlado.

O DARPA1998 consiste em uma simulação de um ambiente de rede local de uma instalação da força aérea dos Estados Unidos. Para simular de forma fiel o tráfego encontrado em uma base aérea, o ambiente real foi monitorado e o tráfego foi gerado em um ambiente controlado de forma que ambos fossem estatisticamente iguais.

Durante a simulação, a rede foi composta por duas áreas, a rede interna e rede externa, as duas redes eram conectadas por um roteador. Ao total, 11 máquinas foram utilizadas. A rede externa contava com um gerador de tráfego responsável pela produção do tráfego normal e do atacante, um servidor de rede, um *sniffer*, e duas máquinas utilizadas para a geração de ataques

de forma manual. A rede interna se utilizava de um *sniffer*, um gerador de tráfego e quatro máquinas vítimas. Os geradores de tráfego foram alterados de forma que o sistema operacional fosse capaz de alterar o endereço do cliente a fim de simular um ambiente com centenas de clientes.

Todo o tráfego foi gerado de forma automática, para que não houvesse a necessidade da interação humana, ou seja, permitindo a reprodução. O tráfego legítimo foi gerado através de *scripts* que efetuavam as requisições pelo cliente, sendo a resposta por parte do servidor, também efetuada por *scripts*. Os ataques foram obtidos de diversas fontes e gerados durante o período de simulação. Nove semanas de tráfego foram gerados. A rotulação do tráfego é efetuada conforme um arquivo, descrevendo cada sessão gerada e cada tipo de tráfego relacionado. No total, 32 tipos diferentes de ataques foram gerados, sendo eles divididos de acordo com a taxonomia apresentada por Kendall [6].

Diversos autores questionam a aplicabilidade dos dados presentes no DARPA1998 [8, 14]. McHugh [9] critica a ausência da validação dos dados presentes no DARPA1998, questiona os dados presentes na base e argumenta que o método utilizado para a sua criação não gera tráfego real. Em seu trabalho, McHugh [9] conduz uma série de experimentos e evidencia que a taxa da rede não condiz com o cenário proposto, pois a base apresenta uma taxa do tráfego abaixo do esperado para o ambiente simulado. Não distante, Mahoney e Chan [10] conduziram uma série de experimentos e identificaram diversas irregularidades nos dados. Esses autores identificaram baixa variabilidade dos cabeçalhos dos pacotes e concluíram que avaliações de IDS através do DARPA1998 não são confiáveis.

Desde então, inúmeras abordagens foram propostas na literatura no sentido de criar bases de intrusão. Shiravi e seus colegas [7] propuseram a criação de bases através da utilização de perfis que representam o comportamento de cada usuário. Em sua abordagem, cada cliente possui um perfil específico que é estatisticamente modelado de acordo com traços da rede de diversos serviços. Apesar da abordagem gerar tráfego de maneira fiel ao comportamento evidenciado em um ambiente de produção, essa abordagem produz uma base específica ao período de análise do perfil e ao ambiente monitorado. Assim, dificulta-se também a atualização dos dados, uma vez que a cada atualização os perfis dos usuários devem ser remodelados.

Canali e seus colegas [15] desenvolveram uma base de ataques para detecção de páginas malignas através de páginas reais da rede. Os autores coletaram as páginas benignas através do *crawling* das páginas disponíveis no site Alexa (site que lista páginas mais visitadas

mundialmente). As páginas malignas foram obtidas com a utilização de uma base conhecida. Os autores assumem que as páginas populares não são malignas. Esta abordagem permite a criação de bases públicas, uma vez que utiliza dados disponíveis a comunidade. Porém, não garante a correta pré-classificação dos eventos, uma vez que sites populares também estão sujeitos a ataques. O método de validação adotado por Canali e seus colegas [15] é discutido na seção seguinte.

Algumas abordagens também disponibilizam tráfego real para a comunidade de um ambiente específico. O CAIDA [16] disponibiliza tráfego real de ambientes de produção. O tráfego disponibilizado passa por um processo de sanitização do cabeçalho, além da remoção do *payload* dos pacotes. Este processo remove informação de forma significativa e a análise do tráfego torna-se dependente de fluxo. A pré-classificação torna-se inviável, visto que não há informação do tráfego disponível para efetuar o processo.

De forma similar, o PREDICT [17] disponibiliza um conjunto de bases de forma pública. Para a utilização em IDS, o PREDICT disponibiliza dados de fluxo de rede de universidades parceiras. Apenas os dados dos cabeçalhos já higienizados são disponibilizados.

Cláudia Pascoal e seus colegas [18] validaram seu método proposto para detecção de anomalia através do monitoramento do ambiente da rede da universidade. A abordagem garantiu a utilização de um tráfego real e válido, porém impossibilitou a comparação com outras abordagens na literatura, uma vez que o tráfego não pode ser compartilhado.

A Tabela 1 exibe a comparação entre as bases estudadas em relação às propriedades esperadas para as bases de intrusão.

Tabela 1 – Comparação dos trabalhos relacionados.

Trabalho	*Propriedade*							
	Real	*Válido*	*Previamente Classificado*	*Diversa*	*Correta*	*Facilmente Atualizável*	*Reprodutível*	*Pública*
DARPA1998 [13]	Tráfego não condiz com a realidade de uma base aérea [9]	Pacotes não são corretamente formados [10]	Sim	Tráfego com baixa variabilidade [10]	Autores não utilizaram ferramentas conhecidas [9]	Atualização complexa	Não, ferramentas não são publicamente disponíveis	Sim

Shiravi [7]	Tráfego específico ao ambiente simulado	Pacotes são corretamente formados	Sim	Tráfego com baixa variabilidade	Autores utilizaram ferramentas de mercado	Atualização complexa, requer atualização dos perfis e das aplicações	A reprodução dos perfis é possível, o ambiente simulado não foi totalmente detalhado	Sim
Canali [15]	Base condiz com o cenário tratado	Páginas são corretamente formadas	Parcialmente, não há como garantir que as páginas são benignas, o autor assumiu pela popularidade	Grande variabilidade de páginas	Páginas malignas são publicamente disponíveis e auditáveis	Sim, requer apenas o *crawling* das páginas populares novamente	Sim, o autor elenca as páginas que compõem a base	Sim
CAIDA [16]	Tráfego passa por processo de sanitização e perde informações	Conteúdo não é corretamente formado	Parcialmente, não há como garantir que o tráfego foi corretamente classificado	Grande variabilidade, tráfego de ambiente de produção é utilizado	Ataques também são sanitizados, não há como garantir que o tráfego é real	Não há como atualizar, atualização necessita do monitoramento do ambiente novamente	Não	Sim, para público restrito
PREDICT [17]	Informação dos cabeçalhos sofrem sanitização, *payload* não é fornecido, apenas informação de fluxo é disponibilizada	Conteúdo não é corretamente formado	Parcialmente, não há como garantir que o tráfego foi corretamente classificado	Grande variabilidade, tráfego de ambiente de produção é utilizado	Ataques também são sanitizados, não há como garantir que o tráfego é real	Não há como atualizar, atualização necessita do monitoramento do ambiente novamente	Não	Sim, para público restrito
Cláudia Pascoal [18]	Sim, tráfego da universidade é utilizado	Sim	Parcialmente, não há como garantir que o tráfego foi corretamente classificado	Apenas para o ambiente monitorado	Sim, autor utiliza ferramentas conhecidas	Não há como atualizar, atualização necessita do monitoramento do ambiente novamente	Não, dados não são disponibilizados	Não

3.2 Métodos de Avaliação

Os resultados obtidos na literatura relacionados à área de detecção de intrusão geram uma ampla discussão, o que acarreta uma quantidade significativa de autores que questionam os resultados publicados.

Ringberg [19] notou os problemas relativos às premissas utilizadas na área de aprendizagem de máquina em detecção de intrusão. Recomendando o uso de simulação, o autor discute quatro necessidades básicas para a completa avaliação de uma técnica baseada em detecção de anomalias. São elas: a existência de uma *baseline* de comparação, a reprodutibilidade dos experimentos, a definição do comportamento anômalo e o controle de todo o experimento.

Carrie Gates e Carol Taylor [14] questionam a adoção de aprendizagem de máquina, e seus métodos de avaliação para detecção de anomalias em rede. Os autores mostraram que muitas das premissas básicas adotadas na literatura (como a baixa frequência de ataques ou o comportamento do cliente ser facilmente modelável), não podem ser aceitas em cenários de NIDS atualmente, por isso os autores questionam a aplicabilidade dos resultados reportados na literatura.

A utilização de ferramentas de simulação para a geração do tráfego é amplamente discutida na literatura [12, 14] e, apesar da sua recomendação por ser a única alternativa para a comparação aberta dos resultados com a literatura, o seu uso deve ser estudado para evitar problemas de variabilidade do tráfego.

Os autores Sommer e Paxon [12] relatam a carência de utilização de abordagem baseada em anomalia para ambientes de produção. Infere-se que tal situação ocorre em razão das diferenças entre as taxas de acerto do classificador obtidas durante o desenvolvimento do IDS (estimativa da acurácia) e em seu uso no ambiente de produção.

O processo de avaliação dos modelos de ataque para IDS dificilmente é considerado durante o desenvolvimento dos trabalhos. De forma geral, o processo de avaliação típico de aprendizagem de máquina é utilizado, e considera-se que o ambiente de treinamento é exatamente igual ao ambiente de produção.

Não foram encontrados trabalhos que considerem mudanças nos perfis de comportamento no cenário de detecção de intrusão, fator já apontado por Carrie Gates e Carol Taylor [14]. Shiravi e seus colegas [7] criaram quatro cenários que apresentam ataques

distintos, cujo objetivo era reproduzir atividades típicas de um atacante: o acesso à rede, ataques de negação de serviço a nível de aplicação e de rede; e tentativas de acesso à máquina remota. Apesar da diversidade dos cenários, o objetivo dos autores consistiu em apenas avaliar o método proposto de criação de bases, e não elaboraram um modelo de ataque. Os autores também deixaram de considerar as mudanças nos perfis de comportamento de usuários.

Normalmente, os autores assumem algumas propriedades sobre os dados utilizados. Canali e seus colegas [15] assumem que a base de treinamento apresenta exatamente a mesma distribuição dos atributos para o ambiente de produção e que a distribuição não muda ao longo do tempo.

Nos últimos anos, a abordagem de detecção de mudança de conceito (*concept drift*) está sendo utilizada na área de detecção de intrusão. Normalmente, os autores consideram que as mudanças nos padrões dos tráfegos também acarretam mudanças das suas propriedades estatísticas [17], como, por exemplo, atualizações nos protocolos que mudam a forma como os serviços se comunicam [18]. Tais mudanças podem ser temporárias, cíclicas, de curta ou grande duração [19]. Diversos trabalhos lidam com as mudanças de conceito ao longo do tempo através da atualização dos modelos [20, 21, 22]. Apesar dos resultados promissores, essas abordagens estão suscetíveis a propagações de erro [23]. Contudo, o processo de atualização dos modelos está fora do escopo do presente trabalho.

O processo de identificação de mudança de perfis é tratado em diversos trabalhos. O trabalho de Sun-il Kim e seus colegas [24] trata da mudança de conceito no tráfego. Os autores extraem 256 características dos pacotes de acordo com a ocorrência de cada valor possível dos campos. O processo é dividido em duas etapas, a do treinamento e a da detecção. O treinamento estabelece o valor médio de cada característica e o seu desvio padrão equivalente. Durante a etapa de detecção, calcula-se a quantidade de características (256) que estão fora dos limiares, estabelecida pela média e respectivo desvio padrão. Para os autores, o evento é considerado como anômalo (e consequentemente um ataque) caso exista um grande percentual de características fora dos limiares. Para tornar a solução proposta pelos autores resiliente a ruídos, os autores efetuaram diversos testes de níveis de similaridade.

A abordagem proposta por Sun-il Kim [24] deixa a desejar em alguns fatores. Os autores avaliaram a proposta através de duas bases distintas, o DARPA1998 e a base privada dos autores com o tráfego da universidade. O DARPA1998 apresenta problemas na variabilidade dos campos dos protocolos, assim como o conteúdo dos pacotes [10], invalidando assim a

avaliação dos autores, uma vez que as características são obtidas através da frequência dos valores possíveis. A base privada utilizada impossibilita auditoria e comparação. Os autores assumem que os ataques (perfis anômalos) apresentam distribuição significativamente distinta (estabelecida através do desvio padrão) dos eventos normais, logo assume-se que eventos normais não sofrem variação de perfis ao longo do tempo, uma vez que qualquer variação é considerada como um ataque. Por fim, a abordagem requer 256 comparações para a classificação, uma vez que todos atributos devem ser avaliados. A abordagem de detecção de anomalia através da frequência dos valores no *payload* é comumente utilizada [25, 26, 27].

A garantia da confiabilidade das classificações é tratada em outras áreas. No contexto de verificação de números escritos à mão, Luiz Oliveira e seus colegas [28] empregam a utilização da rejeição para garantir a confiabilidade das classificações. Os autores utilizam um módulo global de decisão que rejeita ou aceita o resultado do reconhecimento dos caracteres, baseando-se em limiares de rejeição específicas a cada classe.

Não foram encontrados trabalhos que adotam a técnica de rejeição na área de detecção de intrusão a fim de garantir a confiabilidade do sistema. Nota-se uma disparidade entre os métodos tipicamente utilizados na área e a sua real aplicação em ambientes de produção. De forma geral, os autores não consideram o ambiente de avaliação utilizado, o processamento necessário para utilizar a técnica estudada ou a confiabilidade das detecções realizadas.

Capítulo 4

Proposta

A proposta deste trabalho aponta um método que possibilita a correta validação de um modelo do ataque em ambientes de produção, tornando viável e confiável sua utilização. Para tanto realizou-se cinco etapas:

1) *Método de criação de bases de intrusão*

Primeiramente, a fim de indicar as propriedades apresentadas na seção 2.3, um método para criação de bases de intrusão para validação de NIDS é proposto.

2) *Extrator de características*

Um conjunto de características com base nos trabalhos correlatos na literatura é definido, considerando o cenário de um NIDS.

3) *Método de obtenção de modelos de ataques*

O método para obtenção de modelos de ataques é proposto com base nas abordagens utilizadas na literatura.

4) *Método de avaliação de modelos de ataques*

O método de avaliação de modelos de ataque é proposto a fim de avaliar a capacidade de detecção dos modelos de ataques sobre as propriedades evidenciadas em um ambiente de produção (seção 2.4).

5) *Método de rejeição*

Finalmente, um método de rejeição é proposto para garantir a confiabilidade das classificações efetuadas pelos modelos.

4.1 Método de criação de bases de intrusão

O método indicado para criação de bases de intrusão é representado pela Figura 6. A metodologia de geração de tráfego em um ambiente controlado é adotada na proposta, pois permite o livre compartilhamento dos dados. Duas classes são consideradas: (i) normal (conteúdo legítimo) e (ii) intrusão (ataque). Para tanto, o tráfego do servidor é gerado através de um *honeypot*, gerando tráfego real (que pode ser observado em uma rede de computadores) e válido (com pacotes corretamente formados para requisição e respostas). Para a simulação do tráfego do cliente, ferramentas de *workload* são utilizadas, gerando tráfego real e válido. Os ataques são gerados utilizando ferramentas conhecidas de auditoria.

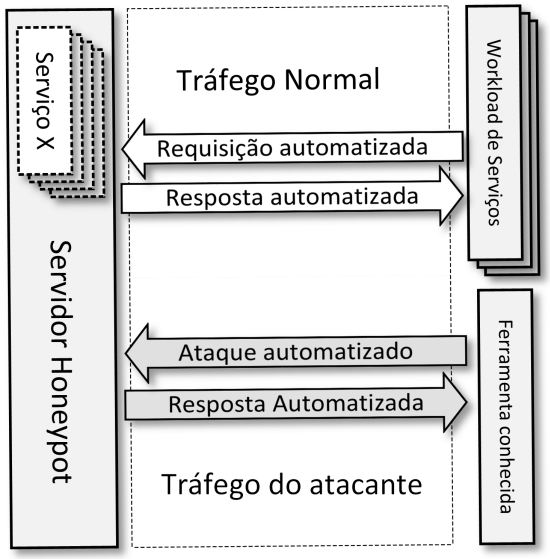

Figura 6 – Método proposto para criação de bases de intrusão.

A técnica de *honeypot* é usualmente utilizada para simular um host vulnerável na rede. Na presente proposta, o *honeypot* é responsável pela geração do tráfego do servidor de maneira real, válida e facilmente atualizável, uma vez que cabe à ferramenta a atualização do tráfego a cada atualização da base.

4.1.1 Método para criação do tráfego normal

O método utilizado para gerar tráfego normal visa garantir o comportamento do usuário de forma similar ao evidenciado em ambientes de produção, na perspectiva do cliente e do

servidor (Figura 4). É esperado que os serviços fornecidos pelo servidor e os conteúdos requisitados pelos clientes apresentem uma variabilidade significativa para garantir a diversidade do tráfego gerado (seção 2.3). O objetivo é evitar o tráfego repetitivo, que é indesejável durante o treinamento e testes de um IDS, por tornar o modelo tendencioso.

Para imitar o comportamento imprevisível do cliente, um conjunto de conteúdos para cada serviço é selecionado. Cada cliente efetua uma requisição aleatória, porém válida e real (a partir do conteúdo predeterminado), implementada por uma ferramenta de *workload*. Essa ferramenta é utilizada para automatizar o papel do cliente, realizando interações aleatórias com o servidor, promovendo o comportamento evidenciado de um cliente em ambientes de produção.

4.1.2 Método para criação do tráfego do atacante

A ausência de padronização das ferramentas utilizadas em ataques é o principal desafio na geração do tráfego do atacante. Na literatura, os autores tipicamente implementam ataques segundo a sua discrição, mas não é possível garantir, neste caso, que se encontre o padrão do ataque fielmente, o que dificulta a reprodutibilidade do ataque. Isso impossibilita comparação entre abordagens.

Após um ataque tornar-se conhecido e reportado, iniciativas como a Common Vulnerabilities Exposure (CVE, cve.mitre.org), detalham as vulnerabilidades/ataques conhecidos e os serviços afetados. Implementações que são CVE compatíveis, garantem que o comportamento do ataque é de fato o esperado de acordo com o relato da vulnerabilidade. Assim, ferramentas que utilizam um padrão conhecido tornam-se auditáveis e permitem a reprodutibilidade.

Neste trabalho são usadas ferramentas conhecidas e padronizadas para a geração de ataques (Figura 6). Esse fator tende a garantir que todos ataques gerados na base de intrusão foram corretamente implementados e geraram o tráfego do ataque da maneira correta. Assim, foi usado o *honeypot* para implementar o servidor, pois ele imita o comportamento de um servidor real, interpretando cada requisição e fornecendo a sua devida resposta. Um conjunto predeterminado de serviços vulneráveis é fornecido pelo *honeypot*, pois isso faz com que a proposta gere tráfego (normal) real, válido e de fácil atualização.

4.2 Extração de características

Para a leitura de cada pacote de rede, um conjunto predeterminado de atributos são extraídos e enviados ao mecanismo de detecção para identificação da classe (Figura 1, normal ou ataque).

No total, 53 atributos foram definidos e distribuídos em três grupos distintos de características de acordo com a literatura [20, 21, 22]. Sendo eles, (i) baseado em cabeçalho, referentes à informação extraída diretamente do cabeçalho dos protocolos; (ii) baseado em *hosts*, refere-se ao histórico da comunicação entre dois *hosts* na rede; e, por fim, (iii) baseado em serviço, referente ao histórico da comunicação entre dois serviços. As tabelas 2 e 3 exibem os atributos definidos conforme o estudo sobre trabalhos relacionados.

Tabela 2 – Conjunto de atributos extraídos diretamente dos cabeçalhos dos protocolos.

#	Nome	Descrição
Baseado em cabeçalho – Cabeçalho IP		
1	Ip_type	Tipo de serviço
2	Ip_len	Tamanho do cabeçalho
3	Ip_id	Identificação
4	Ip_offset	Offset
5	Ip_RF	Flag Reservada
6	Ip_DF	Flag não fragmente
7	Ip_MF	Flag mais fragmentos
8	Ip_proto	Protocolo
9	Ip_chk	Checksum
Baseado em cabeçalho – Cabeçalho UDP		
10	Udp_sport	Porta de origem
11	Udp_dport	Porta de destino
12	Udp_len	Tamanho do datagrama
13	Udp_chk	Checksum
Baseado em cabeçalho – Cabeçalho ICMP		
14	Icmp_type	Tipo
15	Icmp_code	Código
16	Icmp_chk	Checksum
Baseado em cabeçalho – Cabeçalho TCP		
17	Tcp_sport	Porta de origem
18	Tcp_dport	Porta de destino
19	Tcp_seq	Número de sequência
20	Tcp_ack	Número de confirmação
21	Tcp_ffin	Flag FIN
22	Tcp_fsyn	Flag SYN
24	Tcp_frst	Flag RST
23	Tcp_fpsh	Flag PUSH
25	Tcp_fack	Flag ACK
26	Tcp_furg	Flag URG
27	Tcp_win	Número da janela
28	Tcp_chk	Checksum
29	Tcp_urg	Urgente
30	Fr_length	Tamanho da frame

Para o propósito do trabalho, três protocolos são considerados, TCP, UDP e ICMP. Para cada pacote presente na base de dados gerada, um total de 53 atributos são extraídos, independentemente do protocolo. Dessa forma, não há atributos inexistentes, evitando problemas inerentes à falta de atributos ocorrentes em determinados classificadores [29]. Isso será efetuado através da utilização de um valor pré-determinado para atributos não existentes.

Tabela 3 – Conjunto de atributos extraídos conforme o histórico da comunicação.

#	Nome	Descrição
	Baseado em hosts	
31	Count_fr_cli_soc	Quantidade de pacotes enviados do cliente para o servidor
32	Count_fr_soc_cli	Quantidade de pacotes enviados do servidor para o cliente
33	Num_bytes_cli_soc	Quantidade de bytes enviados do cliente para o servidor
34	Num_bytes_soc_cli	Quantidade de bytes enviados do servidor para o cliente
35	Num_pushed_cli_soc	Quantidade de flags PSH enviados do cliente para o servidor
36	Num_pushed_soc_cli	Quantidade de flags PSH enviados do servidor para o cliente
37	Num_syn_fin_cli_soc	Quantidade de flags SYN e FIN enviados do cliente para o servidor
38	Num_syn_fin_soc_cli	Quantidade de flags SYN e FIN enviados do servidor para o cliente
39	Num_fin_cli_soc	Quantidade de flags FIN enviados do cliente para o servidor
40	Num_fin_soc_cli	Quantidade de flags FIN enviados do servidor para o cliente
41	Num_ack_cli_soc	Quantidade de flags ACK enviados do cliente para o servidor
42	Num_ack_soc_cli	Quantidade de flags ACK enviados do servidor para o cliente
43	Num_syn_cli_soc	Quantidade de flags SYN enviados do cliente para o servidor
44	Num_syn_soc_cli	Quantidade de flags SYN enviados do servidor para o cliente
45	Num_rst_cli_soc	Quantidade de flags RST enviados do cliente para o servidor
46	Num_rst_soc_cli	Quantidade de flags RST enviados do servidor para o cliente
47	First_packet	1, se é o primeiro pacote, 0, caso contrário
	Baseado em serviços	
48	Conn_status	Estado da conexão TCP
49	Count_serv_cli_soc	Quantidade de pacotes enviados do serviço do cliente para o serviço do servidor
50	Count_serv_soc_cli	Quantidade de pacotes enviados do serviço do servidor para o serviço do cliente
51	Num_bytes_serv_cli_soc	Quantidade de bytes enviados do serviço do cliente para o serviço do servidor
52	Num_bytes_serv_soc_cli	Quantidade de bytes enviados do serviço do cliente para o serviço do servidor
53	First_packet_serv	1, se é o primeiro pacote entre os serviços, 0, caso contrário

4.3 Método para obtenção dos modelos do ataque

Para a finalidade desta proposta, quatro classificadores foram estabelecidos: (i) árvore de decisão, decorrente de sua rápida classificação, fator desejável em um IDS, (ii) naive bayes, pela simplicidade e também por sua rápida classificação, (iii) k vizinhos mais próximos, devido a sua fácil atualização do modelo, e por fim (iv) máquina de vetores de suporte, devido a sua popular utilização na área de detecção de intrusão.

Para o naive bayes, será utilizada a técnica de discretização de Usama M. Fayyad e Keki B. Irani [30]. Isso será necessário, pois não é possível adotar uma distribuição para o conjunto

dos atributos numéricos. A árvore de decisão não necessitará de uma etapa de pré-processamento.

A base de treinamento é composta por 25% da base de dados total. A base de testes será 25% do total e a base de validação do modelo corresponde a 50% da base relativa. Espera-se que a base esteja desbalanceada, composta em sua maioria por eventos normais. Assim, será utilizado um processo de estratificação [31], que tornará a distribuição entre as classes iguais, para o treinamento e teste.

Para a obtenção dos modelos dos classificadores, será utilizado o ambiente de aprendizagem de máquina Weka[1], garantindo assim a correta implementação dos modelos utilizados.

4.4 Método de avaliação de modelos de ataques

Através da utilização do método de criação de bases de intrusão (descrito na seção 4.1), propõe-se um novo método de avaliação específico à área de NIDS baseado em anomalia. O objetivo é validar as suposições comumente utilizadas na literatura [8, 14]. O processo geral é exibido na Figura 7 e descrito nas seções posteriores.

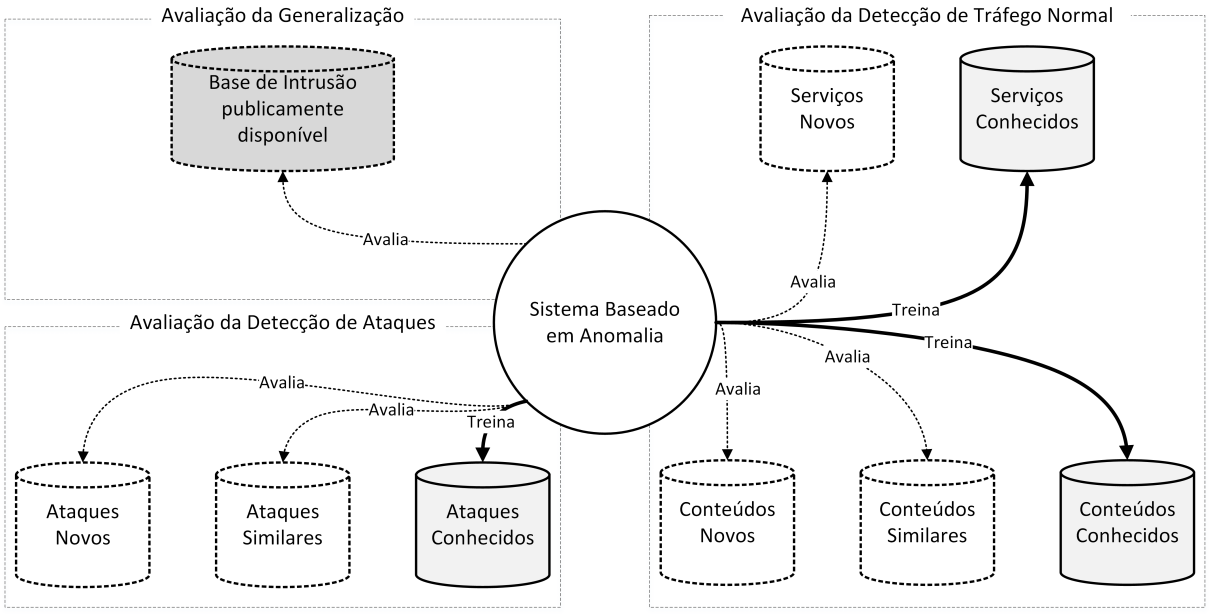

Figura 7 – Método proposto para avaliação de NIDS baseado em anomalia.

[1] http://www.cs.waikato.ac.nz/ml/weka/

4.4.1 Avaliação de detecção de ataques

A principal premissa sobre um NIDS baseado em anomalia é a sua capacidade de detectar novos ataques. A premissa para diferenciar um ataque novo de um já conhecido é que o ataque deve apresentar um comportamento significativamente diferente da utilização típica do sistema, e assim pode ser identificado via detecção de similaridades. No entanto, um esquema de aprendizagem de máquina requer exemplos que descrevam o comportamento de cada uma de suas consideradas classes (por exemplo normal e ataque). Os eventos de entrada são geralmente classificados com base em uma métrica de similaridade entre os eventos conhecidos da base de treinamento. Assim, impõe-se que novos ataques sejam necessariamente semelhantes a ataques já conhecidos.

Treinar um algoritmo de aprendizagem de máquina com ataques não conhecidos é, por definição, impossível. Porém, é possível induzir tal capacidade através da manipulação dos eventos presentes nas bases de validação. Um sistema pode ser treinado com um tipo limitado de ataque e validado com ataques semelhantes ou totalmente diferentes (Figura 7, avaliação de detecção de ataques). A definição de similaridade entre os eventos é dependente de contexto e deve ser definida de acordo com o conhecimento do especialista. O processo é exibido na Figura 8.

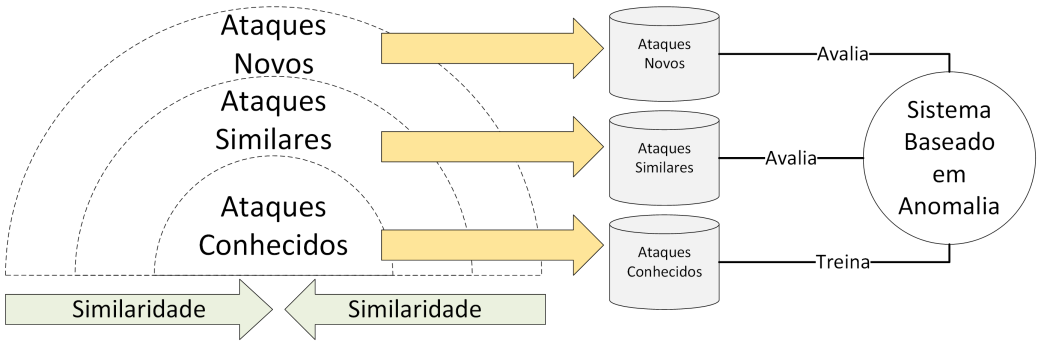

Figura 8 – Método proposto para avaliação da capacidade de um NIDS baseado em anomalia para detectar ataques.

Inicialmente, o classificador é treinado com um conjunto limitado de ataques, que são semelhantes entre eles e esperados em um ambiente de produção. Posteriormente, o modelo de detecção obtido é validado utilizando as bases de validação com ataques similares e novos. Ataques semelhantes nesse contexto são os ataques com uma alta taxa de similaridade com

outros ataques, contra os quais o sistema foi treinado. Por outro lado, os novos ataques são altamente diferentes dos ataques conhecidos.

A taxa de detecção de ataques similares define a capacidade do sistema na detecção de eventos com comportamentos semelhantes aos ataques conhecidos. Essa capacidade é desejável em sistemas de detecção de intrusão, devido à grande variabilidade dos ataques existentes e também a ataques ofuscados, por exemplo. Os ataques ofuscados apresentam um padrão diferente do conhecido e podem evadir um NIDS baseado em assinatura, em que a detecção é realizada pela busca de padrões de ataques conhecidos. No entanto, os ataques ofuscados apresentam o comportamento semelhante ou igual, podendo ser identificados por um esquema de detecção à base de anomalia, se as características utilizadas forem discriminantes o suficiente.

A taxa de detecção de novos ataques define a capacidade de o sistema identificar diferentes tipos de ataques. Os ataques que são diferentes não possuem um comportamento semelhante aos ataques contra os quais o sistema foi treinado. Esse tipo de incidência ocorre em ambientes de produção, uma vez que não é possível treinar o sistema com todos os ataques conhecidos ou novos. O modelo utilizado deve ser capaz de relacionar o novo comportamento de ataque com os ataques conhecidos, para detectá-los de forma adequada, o que é um pressuposto para usar qualquer método de detecção à base de anomalia.

Durante a criação das bases de validação, o tráfego normal também deve ser gerado. A incidência de um determinado ataque impacta sobre a forma como o serviço responde a requisições legítimas, por exemplo: um ataque pode tornar um serviço indisponível enquanto outro tipo de ataque faz com que o serviço responda apenas algumas requisições. Portanto, a ocorrência de um ataque pode impactar na taxa de detecção do tráfego normal. Observado esse fator, as bases de validação para a taxa de detecção de ataque devem usar a mesma abordagem da criação do tráfego normal. Essa abordagem utiliza o tráfego normal como *baseline* e permite identificar como os mesmos serviços reagem a cada tipo de ataque e como o modelo detecta tais mudanças.

4.4.2 Avaliação da detecção de tráfego normal

O tráfego normal/legítimo é composto por duas entidades: o cliente e o servidor. O cliente origina as requisições de acordo com o conteúdo e serviços disponíveis no servidor. O

servidor, em contrapartida, dispõe os serviços e seus conteúdos. Assim, o tráfego normal pode variar em nível de conteúdo e serviço.

Devido à ausência de dados públicos disponíveis, os especialistas normalmente assumem que o tráfego da rede é estático [14]. De forma geral, um NIDS baseado em anomalia é validado através de uma base única de validação, dividida em três partes, com o mesmo comportamento de tráfego entre eles.

É sabido que o tráfego de rede sofre constantes alterações [12, 17]. Assim, não é correto assumir que o comportamento evidenciado em um certo período, durante a criação da base de intrusão, permanecerá constante. O modelo de ataque utilizado deve ser capaz de detectar as mudanças de comportamento do tráfego normal e deve adicionalmente realizar sua detecção em uma taxa razoável, quando a atualização do modelo não é possível.

Ao validar o método de detecção de intrusão, deve-se considerar as limitações da base de intrusão, pois não é possível criar todos os serviços e os conteúdos que serão evidenciados em ambientes de produção. No entanto, o modelo do ataque deve ser capaz de detectar diferentes conteúdos de tráfego normal, e também diferentes serviços, com o seu novo tipo de conteúdo. As bases de validação podem ser manipuladas para testar tais propriedades do modelo.

Seguindo o método de detecção de taxa de ataque (seção 4.4.1), o método de detecção de tráfego normal consiste em limitar as bases de validação utilizadas de acordo com duas perspectivas, os tipos de serviços e o conteúdo dos serviços. A taxa de detecção de novos serviços é estabelecida limitando o número de serviços na base treinamento e validando o sistema com um conjunto diferente de serviços não vistos durante a fase de treinamento (Figura 9). Por outro lado, a taxa de detecção de conteúdos novos e semelhantes é estabelecida através da limitação dos conteúdos que os clientes podem solicitar. O método de avaliação da detecção do tráfego normal é exibido na Figura 9.

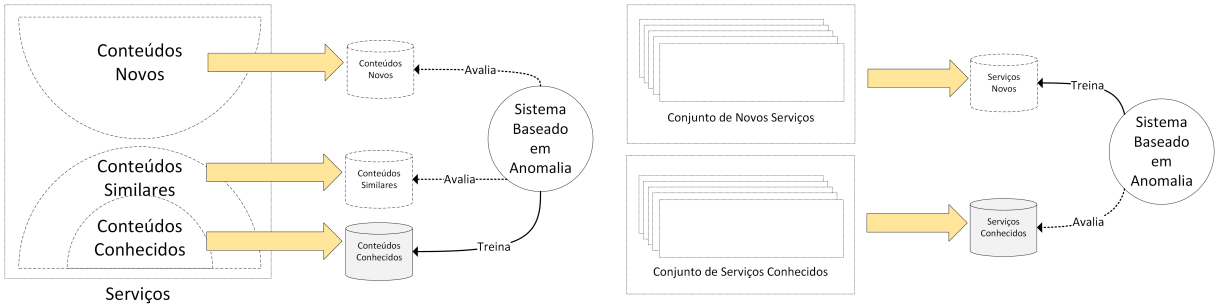

Figura 9 – Método proposto para avaliação da capacidade de um NIDS baseado em anomalia para detectar tráfego normal.

Na validação da taxa de detecção de tráfego normal, os ataques são utilizados como *baseline*. Os mesmos conjuntos de ataques são gerados em cada cenário, permitindo evidenciar a ocorrência de ataque em cada serviço utilizado e os seus conteúdos gerados.

4.4.3 Avaliação da generalização

Diversos métodos têm sido propostos na literatura para a criação de bases de intrusão [8]. Entretanto, apesar dos extensivos esforços, todos esses trabalhos estão expostos a problemas inerentes ao método utilizado durante sua criação. Dessa forma, para validar não apenas a base de dados utilizada para projetar o sistema, mas também o método utilizado para detectar os eventos, uma validação que utilize uma base publicamente disponível torna-se necessária, pois ela fornecerá uma *baseline* de comparação e também a capacidade de generalização.

A capacidade de generalização é uma propriedade desejável de qualquer técnica de aprendizagem de máquina. O conjunto de características extraídas deve permitir ao classificador de generalizar o problema, distinguindo corretamente as classes, independentemente do ambiente em que o mesmo opera. Assim, permite-se que o modelo obtido possa ser utilizado em outros ambientes, e a partir do conjunto de características extraídas, poderá detectar o mesmo tipo de eventos.

Desta forma, a validação que utiliza uma base de intrusão publicamente disponível garante que o modelo seja capaz de detectar um ataque, independentemente do ambiente em que ele foi concebido.

4.4.4 Método de avaliação – sumário

A detecção de intrusão baseada em anomalia enfrenta desafios significativamente diferentes de outros campos em que a aprendizagem de máquina é aplicada com sucesso [8]. O método de validação proposto pretende validar as propriedades esperadas de um NIDS baseado em anomalia com a utilização de classificadores. As seguintes propriedades são obtidas pelo método de validação proposto (Figura 7):

- Capacidade para detectar ataques conhecidos, similares e novos;
- Capacidade para detectar serviços conhecidos e novos;
- Capacidade para detectar conteúdos conhecidos, semelhantes e novos de serviços;
- Capacidade de generalização, que é uma propriedade esperada de um algoritmo de aprendizagem de máquina;

A próxima seção apresenta um método para garantir a confiabilidade do sistema diante da ocorrência de novos comportamentos no tráfego.

4.5 Método de Rejeição

Mudanças no comportamento do tráfego podem tornar os modelos de ataque utilizados obsoletos, o que os torna não confiáveis para utilização em longo prazo em ambientes de produção. O método de rejeição, visa prover detecção de intrusão baseado em anomalia de forma confiável. A abordagem proposta neste trabalho tem pretensão de ser computacionalmente eficiente, permitindo assim a sua utilização em ambientes de produção com altas taxas na rede.

4.5.1 Mudanças na distribuição dos atributos

A acurácia do modelo é dependente do modo em que os valores das características estão distribuídos durante a etapa de treinamento (normalmente a partir de um tráfego real). Portanto, se a distribuição das características sofrerem qualquer alteração, o modelo deve ser atualizado, necessitando assim de conhecimento do especialista para identificar o novo comportamento do tráfego e treinar novamente o modelo.

Mudanças na distribuição dos atributos podem diminuir a acurácia de um modelo. Porém, em ambientes de produção, um classificador não é capaz de checar a acurácia de um modelo, uma vez que as instâncias não são previamente classificadas, como ocorre em uma base de treinamento. Dessa forma, um método é necessário para garantir que as classificações efetuadas por um modelo sejam confiáveis, mesmo frente a mudanças no conteúdo do tráfego. Esta seção descreve um cenário em que a mudança do conteúdo do tráfego ocorre e propõe uma abordagem de rejeição para melhorar a confiabilidade de um classificador, enquanto um novo modelo é obtido.

Ao contrário das propostas presentes na literatura, este trabalho propõe um método de rejeição que considera as alterações frequentes no conteúdo, observadas no tráfego de rede do mundo real. Características discriminantes não apresentam a mesma distribuição dos valores para todos os atributos, como, por exemplo, as classes normal e ataque que apresentam distribuições das características distintas de seus valores. Caso contrário, a classificação não é confiável e o evento deve ser rejeitado, uma vez que a distribuição dos atributos utilizados para a obtenção do modelo e seus valores atuais não são suficientemente parecidos para permitir uma classificação confiável. A rejeição de um evento significa que o classificador não foi capaz de atribuir uma classe ao evento de forma confiável, logo o evento é rejeitado ao invés de ser potencialmente classificado de forma errada.

4.5.2 Cenário de estudo

A Figura 10 exibe um cenário do mundo real em que as distribuições dos atributos mudam durante o tempo. Considera-se o conjunto de atributos de ataques *SYNFlood*[2] como os ataques conhecidos pelo modelo. Quando um ataque de *HTTPFlood*[3] acontece, o mesmo é corretamente detectado pelo modelo, uma vez que as distribuições dos atributos de ambos os ataques são similares. Porém, se o tráfego da rede muda de forma significativa (como em um ataque *Exploit*[4]), a classificação não será confiável devido a mudanças relevantes na distribuição dos atributos de cada classe. Dessa forma, enquanto o modelo do ataque ainda não é atualizado, outra técnica deve ser utilizada a fim de permitir a classificação de forma confiável em ambientes de produção.

[2] Ataque de inundação de requisições de abertura de conexão
[3] Ataque de inundação de requisições de páginas HTTP
[4] Ataque que abusa de uma vulnerabilidade presente no alvo

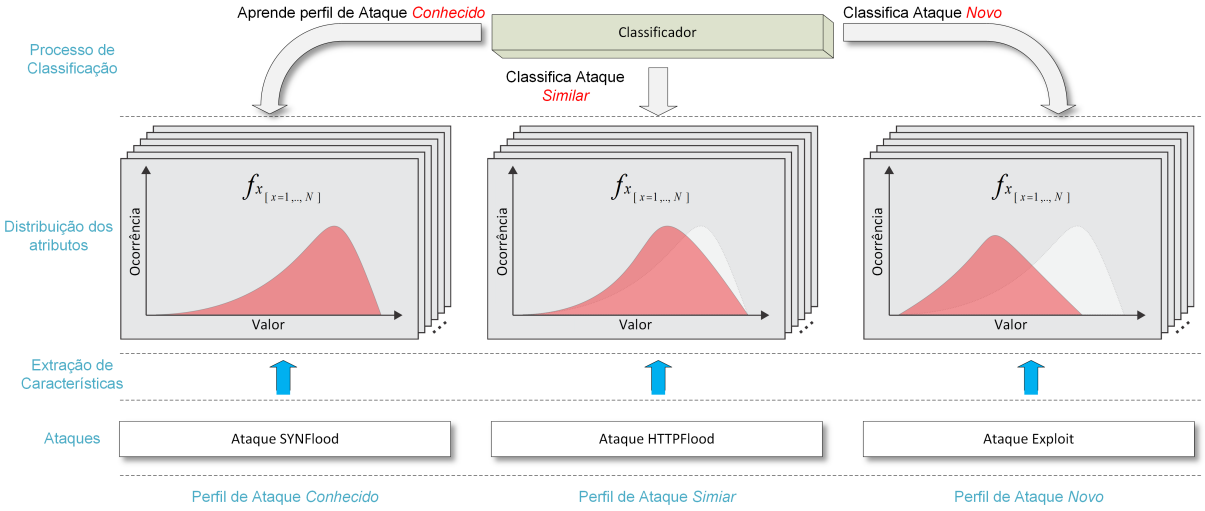

Figura 10 – Mudanças na distribuição dos atributos, considerando SYNFlood como ataque *conhecido* pelo classificador.

4.5.3 Mecanismo de Rejeição

Quando o modelo é obtido a partir da base de treinamento, as mudanças no comportamento do tráfego podem ser detectadas observando mudanças nos valores dos atributos (Figura 10, Ataque *Exploit*). Uma mudança significativa na distribuição dos atributos pode indicar que um novo ataque está ocorrendo. Porém, o estabelecimento dos valores confiáveis dos atributos em tempo real para a definição do nível de similaridade é uma tarefa computacionalmente custosa.

Como reportado em [32, 33, 34], nos NIDS aproximadamente 50 atributos devem ser considerados para efetuar a detecção de intrusão. O estabelecimento do histórico dos valores dos atributos em tempo real torna-se inviável devido à quantidade de atributos utilizados e à necessidade do cálculo do histórico de cada atributo para cada evento.

Para superar esses problemas, esta proposta utiliza um conjunto fixo de valores que define o intervalo válido para cada atributo. Para avaliar o método proposto, foram utilizados três cenários de tráfego: um cenário *conhecido*, um cenário *similar*, e um cenário com *novos* ataques (Figura 10). O cenário *conhecido* é utilizado para obter os limites da faixa de rejeição (ou limiares), definindo assim o intervalo de confiança para os valores dos atributos e também os modelos de ataque, enquanto que os outros cenários são utilizados para avaliar o método proposto de rejeição.

Para cada atributo, dois limiares (t_{lower} e t_{upper}) são computadas (*fx, x=1,2,...,N*). Esses limites definem os intervalos dentro dos quais um atributo é válido. Os intervalos são diferentes para eventos normais e ataques, como a distribuição dos atributos para cada classe também são diferentes. Os limiares são estabelecidos em relação a um valor α (Figura 11), que define um percentual de eventos no conjunto de validação que estão fora dos limites, porém ainda se mantém a confiabilidade do modelo. Para determinar o valor de α, uma análise deve ser executada.

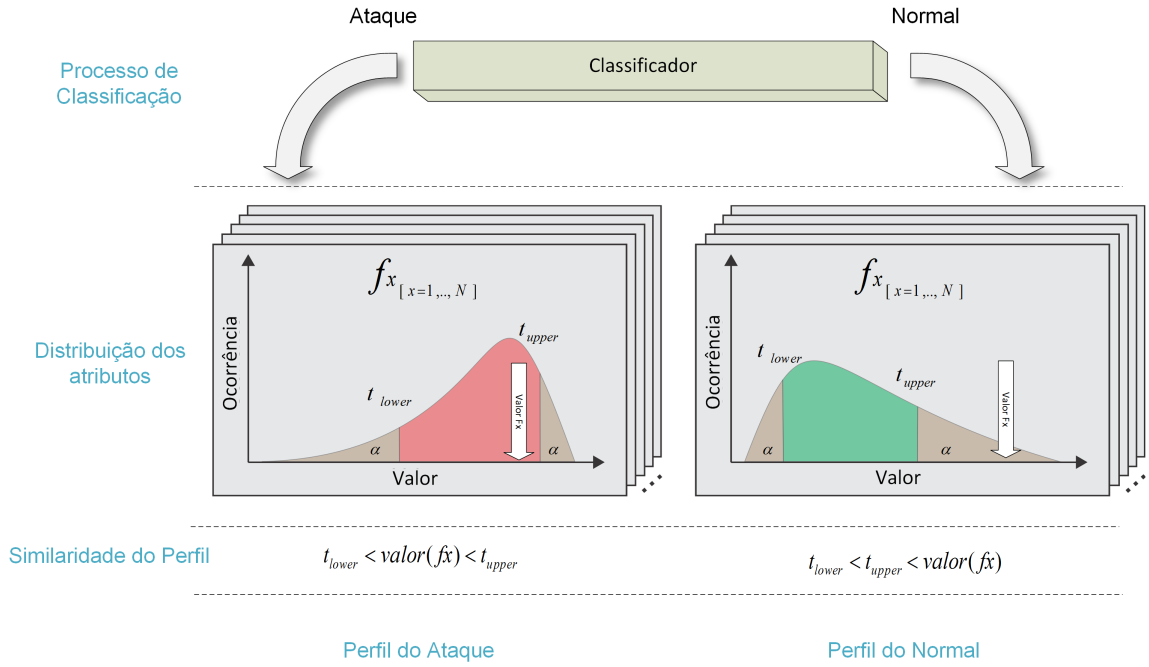

Figura 11 – Atributo dentro do intervalo da classe Ataque.

Para cada atributo *fx*, $perfil_{similaridade\_fx} = 1$, se os valores de *fx* estiverem dentro dos intervalos definidos ($t_{interval}$): $t_{lower} < value\ (f_x) < t_{upper}$; senão, $perfil_{similaridade\_fx} = 0$. Por exemplo: $perfil_{similaridade\_fx} = 1$ para a classe ataque (Figura 9) e $perfil_{similaridade\_fx} = 0$ na Figura 10. Assim, se N denomina a quantidade de atributos no vetor de características, o nível de similaridade do $perfil$ ($perfil_{similaridade}$) é definido da seguinte maneira:

$$perfil_{similaridade} = \frac{\sum_{x=1}^{N} perfil_{similaridade\_fx}}{N}$$

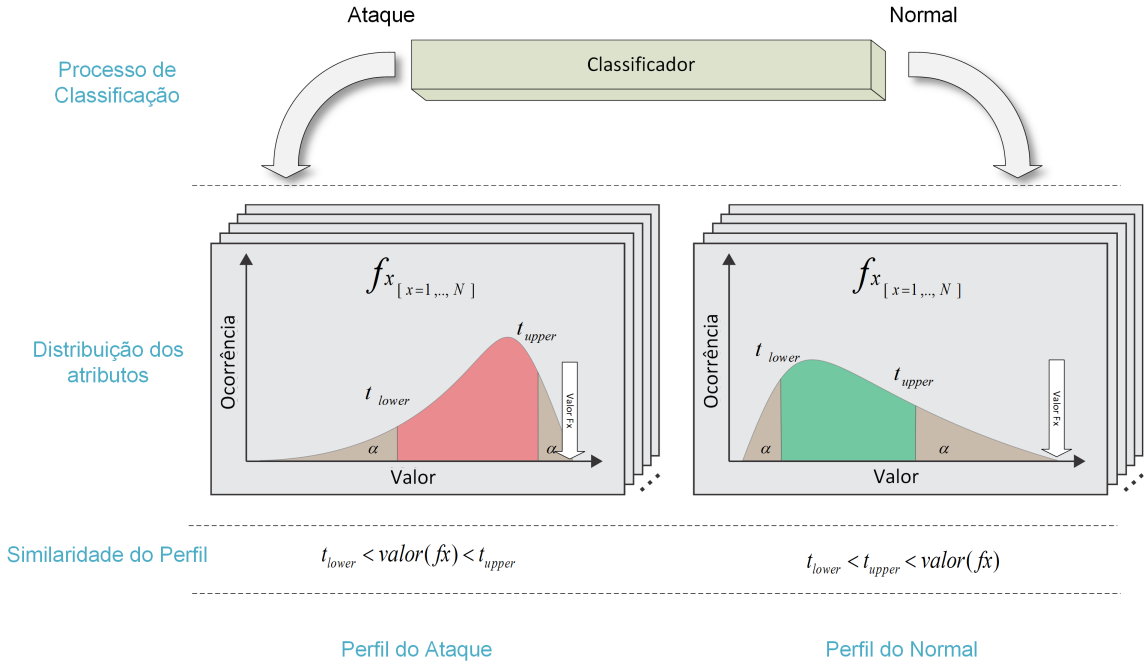

Figura 12 – Valor do atributo fora dos intervalos para ambas as classes.

A saída de um classificador deve ser rejeitada quando é apresentado um baixo nível de similaridade do perfil (e.g. $perfil_{similaridade} < 0.7$), caso contrário o evento é aceito e a classe atribuída pelo modelo é estabelecida ao evento. Através dessa abordagem, torna-se possível estabelecer o nível de similaridade do perfil, identificando uma mudança na distribuição dos atributos do evento, sem a necessidade de se efetuar o cálculo do histórico desses valores.

Capítulo 5

Avaliação

Esta seção apresenta a avaliação dos métodos propostos neste trabalho. Inicialmente são descritos os cenários adotados em cada base de avaliação, a fim de utilizar o método de avaliação proposto. Posteriormente, através do método de avaliação proposto, um conjunto de métodos habitualmente utilizados para detecção de intrusão são avaliados. Por fim, o método de rejeição é avaliado e a abordagem baseada em anomalia utilizada no presente trabalho é comparada a uma ferramenta de mercado.

5.1 Obtenção das bases de intrusão

As subseções a seguir detalham como o método de criação de bases de intrusão (seção 4.1) foram utilizados. Uma extensa descrição sobre a geração do tráfego normal e do atacante é fornecida e posteriormente a infraestrutura de testes é detalhada.

5.1.1 Geração do tráfego normal

A propriedade mais desejável de uma base de intrusão é que o tráfego normal seja o mais realista possível. O tráfego normal deve ser altamente variável, real e válido (seção 2.3).

A geração do tráfego normal é uma tarefa complexa, principalmente devido à dificuldade envolvida na modelagem do comportamento do usuário [12], que geralmente é sazonal e dependente da aplicação. O tráfego gerado é dependente da demanda dos usuários por uma aplicação e específico para o ambiente ser reproduzido.

Levando em consideração esse fator, os clientes são tratados como uma entidade não modelável, que não seguem uma distribuição estatística, padrão reportado em [12]. Cada cliente possui um comportamento único ao solicitar um serviço. Cada cliente pode solicitar um ou mais serviços.

Para alcançar esta propriedade, um conjunto predeterminado de serviços foi estabelecido, com base nos serviços observados em [35]. Os seguintes serviços foram considerados no ambiente de testes: HTTP, SNMP, SMTP, NTP e SSH. Cada resolução de nomes (DNS) também foi gerada como consequência da utilização dos protocolos listados.

Para criar o servidor *honeypot* (Figura 6), responsável por hospedar os serviços, no método proposto, a ferramenta *honeyd*[5] versão 1.5c foi utilizada. Para desenvolver o cliente, requisitando o servidor, utilizaram-se ferramentas de *workload* específicas ao serviço requisitado. É importante ressaltar que a única finalidade da utilização de ferramentas de *workload* neste trabalho é realizar solicitações válidas e reais.

Cada cliente requisita um serviço hospedado no servidor *honeypot*, os serviços providos são descritos na Tabela 4. Para garantir a variabilidade de tráfego, cada cliente varia aleatoriamente o conteúdo solicitado, de acordo com a descrição apresentada na Tabela 4, e o tempo entre as requisições varia de 0 a 4 segundos. A variação entre o tempo de cada pedido e o conteúdo solicitado foi realizada a fim de simular o comportamento não modelável dos clientes. De tal maneira, é possível imitar um usuário navegando numa página web enquanto envia um e-mail, por exemplo.

Cada cliente gera um pedido real e válido para seu serviço, e recebe uma resposta real e válida pelo servidor *honeypot*. Assim, todo o tráfego normal gerado é real e válido. Finalmente, para diminuir a possibilidade de repetição no tráfego gerado, cada cenário é executado por 30 minutos, um tempo razoável considerando a variabilidade de requisição apresentados na Tabela 4. Com o objetivo de permitir a reprodutibilidade entre os cenários, o comportamento de cada cliente é gravado.

Tabela 4 – Serviços e comportamento de cada cliente utilizados para a geração do tráfego legítimo.

Serviço	Descrição
HTTP	1000 sites mais visitados mundialmente foram baixados (www.alexa.com/topsites) e hospedados no servidor *honeypot*, cada cliente HTTP requisita uma página aleatória dentre as disponíveis
SMTP	Cada cliente SMTP envia um e-mail com 50 a 400 *bytes* de assunto e 100 a 4000 *bytes* de conteúdo

[5] http://www.honeyd.org/

SSH	Cada cliente SSH autentica no servidor *honeypot* e executa um comando aleatório dentre 100 possíveis comandos
SNMP	Cada cliente SNMP percorre uma MIB predefinida a partir de uma lista de possíveis MIBS
NTP	Cada cliente sincroniza o horário através do servidor *honeypot*
DNS	Toda resolução de nome também é efetuada ao servidor *honeypot*

5.1.2 Geração do tráfego de ataque

Para gerar o tráfego do atacante, a taxonomia proposta por Kendall [6] foi adotada. Para validar a proposta, dois grupos de ataques foram considerados como ataques de *baseline*: (i) *probing* e (ii) DoS. As ferramentas utilizadas e os tipos de ataque, bem como suas descrições, são fornecidos na Tabela 5.

Tabela 5 – Serviços e comportamento de cada cliente utilizados para a geração do tráfego do atacante.

Categoria	Ataque	Ferramenta Utilizada	Descrição
DoS	*SYNFlood*	Hping3	Envia diversas requisições para *sockets* TCP, variando a frequência de envio dos ataques e a sua duração
	UDPFlood	Hping3	Envia diversos datagramas UDP para uma porta DNS, variando a frequência de envio dos ataques, o tempo de duração e o tamanho de cada datagrama
	ICMPFlood	Hping3	Envia diversas mensagens ICMP para o alvo, variando a frequência de envio dos ataques, a duração e o tamanho de cada mensagem
	TCP Keepalive	Slowloris	Inicia diversas conexões HTTP e as mantêm abertas por um período de tempo, variando a quantidade de conexões a serem abertas
	SMTPFlood	Postal	Envia diversos e-mails ao servidor SMTP, variando a duração, tamanho do conteúdo e assunto e também a frequência de envio
	HTTPFlood	LOIC	Envia diversas requisições GET do HTTP para uma URL específica, variando o tempo de duração e a frequência
Probing	*UDPScan*	Nmap	Busca por portas UDP abertas, variando a frequência dos ataques e sua duração
	SYNScan	Nmap	Busca por portas TCP abertas através do envio de segmentos TCP com a flag SYN marcada enquanto varia a frequência dos ataques e sua duração
	NULLScan	Nmap	Busca por portas TCP abertas através do envio de segmentos TCP sem flags marcadas enquanto varia a frequência dos ataques e sua duração
	TCPConnect	Nmap	Busca por portas TCP abertas através do estabelecimento de conexões TCP enquanto varia a frequência dos ataques e sua duração
	FINScan	Nmap	Busca por portas TCP abertas através do envio de segmentos TCP com a flag FIN marcada enquanto varia a frequência dos ataques e sua duração
	XMASScan	Nmap	Busca por portas TCP abertas através do envio de segmentos TCP com as flags FIN, PSH e URG marcadas enquanto varia a frequência dos ataques e sua duração
	ACKScan	Nmap	Busca por portas TCP abertas através do envio de segmentos TCP com a flag ACK marcada enquanto varia a frequência dos ataques e sua duração
	OS Fingerprint	Nmap	Identifica o Sistema operacional do alvo (https://nmap.org/book/osdetect.html) enquanto varia a frequência dos ataques e sua duração
	Service Fingerprint	Nmap	Identifica os serviços e suas versões do alvo (https://nmap.org/book/man-version-detection.html) enquanto varia a frequência dos ataques e sua duração
Todos	Vulnerability Scan	Nessus	Identifica vulnerabilidades a nível de serviço enquanto varia a frequência dos ataques e sua duração

Para que os ataques sejam altamente variáveis, cada ataque varia a frequência e duração. Qualquer ataque é gerado por uma única máquina, permitindo a rotulagem da classe de forma automática, baseado no endereço IP de origem. É importante notar que essa abordagem gera características específicas do ambiente como, por exemplo: com base apenas no endereço IP, um sistema baseado em anomalia é capaz de identificar todos ataques. O sistema que está sendo

validado deve estar ciente de tal restrição e não deve utilizar qualquer recurso específico do ambiente, tal como TTL (*time to live*) e campos dos endereços IP.

5.1.3 Ambiente de testes

Cada cenário do ambiente de testes é composto por 100 máquinas clientes interligadas. O número de clientes foi definido a fim de maximizar os possíveis comportamentos individuais de cada cliente (detalhados na Tabela 4). Cada cliente é uma máquina Ubuntu 14.04. O tráfego é dependente da ferramenta de *workload* utilizada de acordo com o serviço que está sendo solicitado. Um único servidor *honeypot* foi usado em todos os cenários criados. As máquinas dos atacantes utilizam o sistema operacional Kali Linux versão 1.0.0. No total, 16 máquinas foram utilizadas para a geração dos ataques. Cada máquina atacante gera um ataque específico (Tabela 5).

Uma única rede local interconectada em 10 Mbits/s conecta cada máquina. A velocidade da rede LAN definida permite a gravação do tráfego gerado em uma única máquina, sem perda de pacotes ou necessidade de espelhamento do tráfego [7]. Todas as requisições legítimas e ataques são gerados contra o servidor *honeypot* (Figura 6), e o tráfego gerado é armazenado no próprio servidor *honeypot*. A disposição de cada cenário é descrita nas seções seguintes.

5.1.4 Cenários de detecção de ataques

Como indicado na seção 5.1.2, duas categorias de ataques foram definidas como ataques de *baseline*: *probing* e DoS. Dessa forma, 6 cenários foram estabelecidos a fim de gerar bases de dados para validar a taxa de detecção de ataques (seção 4.4.1).

Nos cenários de detecção de ataques, cada cliente fica responsável em gerar seu tráfego normal de acordo com a distribuição apresentada no diagrama de Venn na Figura 13. As áreas sobrepostas significam que o cliente gera ambos os serviços. A distribuição entre os clientes e os serviços foi realizada para simular a distribuição do tráfego visto em [35].

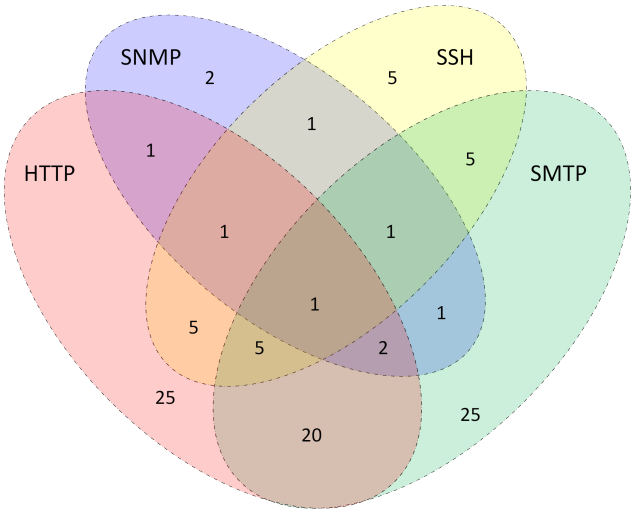

Figura 13 – Diagrama de Venn exibindo a distribuição dos clientes para os serviços.

O tráfego normal permanece constante em nível de requisição de conteúdo e disposição dos clientes. Por outro lado, o tráfego de ataque varia de acordo a cada cenário (seção 4.4.1). Para cada tipo de ataque considerado, *probing* e DoS, três cenários foram estabelecidos. Cada cenário é executado por 30 minutos. Os ataques iniciam no décimo minuto e duram os quinze minutos seguintes. O comportamento de cada atacante é descrito na Tabela 5. Desse modo, foi possível capturar o comportamento do ambiente sem a ocorrência de ataques e posteriormente com a presença dos ataques. A distribuição de tráfego e ataques usados para cada cenário são listados na Tabela 6.

Tabela 6 – Distribuição do tráfego para os cenários de detecção de ataques.

Cenário	Ataques gerados	Tráfego (Pacotes da rede)			Tamanho (Megabytes)
		Normal	*Ataque*	*Total*	
Probing (Conhecido)	UDPScan, SYNScan, NULLScan, TCPConnect, FINScan, XMASScan and ACKScan	28.618.365	36.628	28.654.993	8,476
Probing (Similar)	OS Fingerprint and Service Fingerprint	28.477.884	10.441	28.488.325	8,499
Probing (Novo)	Vulnerability Scan	28.391.914	17.753	28.409.667	8,512
DoS (Conhecido)	SYNFlood, UDPFlood, ICMPFlood and TCP Keepalive	26.747.521	761.269	27.508.790	7,945
DoS (Similar)	SMTPFlood and HTTPFlood	40.278.594	26.390.723	66.669.317	12,143
DoS (Novo)	Vulnerability Scan	27.522.317	3.429	27.525.746	7,265

Três níveis de similaridade de ataque foram definidos nas bases de intrusão: vulnerabilidades de nível de rede, vulnerabilidades de nível de serviço e *exploits* de nível de serviço. O primeiro cenário é nomeado como *conhecido*. O objetivo do cenário *conhecido* é obter o modelo do ataque e também definir a acurácia esperada do modelo (Figura 7). Assim,

apenas ataques em nível de rede, com foco em vulnerabilidades de protocolo de rede foram gerados. É importante notar que as características utilizadas no presente trabalho (descritas na secção 4.2) são utilizadas para detectar especificamente estes tipos de ataque.

As bases de dados *similar* têm ataques com um comportamento semelhante, porém com foco em vulnerabilidades de nível de serviço. Finalmente, as bases *novos* apresentam um novo tipo de ataque que se concentram em vulnerabilidades à nível de serviço.

As bases de ataques criadas imitam o comportamento evidenciado em ambientes de produção [8, 14]. Normalmente, quando um NIDS baseado em anomalia é desenvolvido, apenas ataques detectáveis por um NIDS (ataques em nível de rede) são utilizados durante o treinamento do classificador. Porém, quando o sistema é utilizado em produção, o mesmo enfrenta uma ampla gama de ataques. As bases foram obtidas de acordo com o método já descrito (seção 5.1.3).

5.1.5 Cenários de detecção normal

Os cenários de detecção de tráfego normal foram gerados utilizando os ataques como *baseline.* Dessa forma, dois ataques foram utilizados, sendo que cada um deles é gerado de forma separada, resultando, assim, em uma base diferente, possibilitando a correta validação do método. O conjunto de ataques utilizados foram o *Probing (Conhecido)* e *DoS (Conhecido)*, descritos na seção 5.1.4.

Para a geração das bases de detecção de serviço, os serviços são divididos em dois grupos, os *conhecidos* e os *novos* serviços. Os serviços *conhecidos* são compostos por clientes HTTP e SNMP. Os *novos* serviços são compostos por clientes SMTP, NTP e SSH. Cada cenário é executado por 30 minutos. A distribuição dos clientes segue o diagrama de Venn exibido na Figura 14.

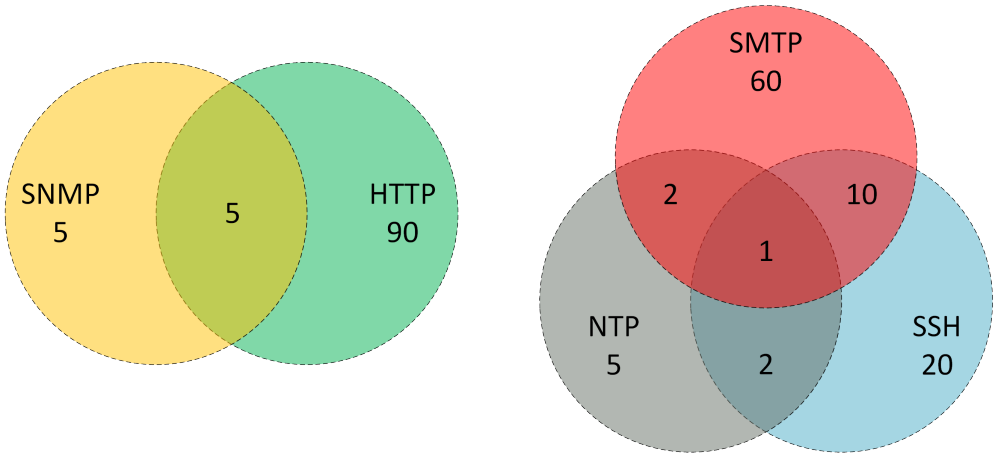

Figura 14 – Diagrama de Venn exibindo a distribuição dos clientes para o cenário *conhecido* (esquerda) e *novo* (direita) de serviços.

Para a geração dos cenários de detecção de conteúdo, os comportamentos dos clientes foram modificados. Três comportamentos diferentes foram definidos divididos em: *Conteúdo Conhecido, Conteúdo Similar* e *Conteúdo Novo*. A distribuição dos clientes segue o diagrama de Venn exibido na Figura 13. O comportamento dos clientes em cada cenário está descrito na Tabela 7. A distribuição do tráfego gerado para cada cenário de detecção de normal é listada na Tabela 8.

Tabela 7 – Comportamento dos clientes em cada cenário.

Serviço	Cenário		
	Conteúdo Conhecido	*Conteúdo Similar*	*Conteúdo Novo*
HTTP	Requisita páginas entre 1 a 200	Requisita páginas entre 1 a 500	Requisita páginas entre 501 a 1000
SMTP	Cada cliente SMTP envia e-mails com 50 a 400 *bytes* de assunto e 100 a 720 *bytes* de conteúdo	Cada cliente SMTP envia e-mails com 50 a 400 *bytes* de assunto e 100 a 1800 *bytes* de conteúdo	Cada cliente SMTP envia e-mails com 50 a 400 *bytes* de assunto e 1801 a 4000 *bytes* de conteúdo
SSH	Cada cliente SSH autentica no servidor *honeypot* e executa um comando aleatório de uma lista de 20 possíveis comandos	Cada cliente SSH autentica no servidor *honeypot* e executa um comando aleatório de uma lista de 50 possíveis comandos	Cada cliente SSH autentica no servidor *honeypot* e executa um comando aleatório de uma lista de 50 possíveis comandos não gerados nos outros cenários
SNMP	Cada cliente SNMP percorre uma MIB a partir de uma lista de possíveis MIB's	Cada cliente SNMP percorre uma MIB a partir de uma lista de possíveis MIB's	Cada cliente SNMP percorre uma MIB a partir de uma lista de possíveis MIB's
DNS	Toda resolução de nome é efetuada ao servidor *honeypot*		

Tabela 8 – Distribuição do tráfego para os cenários de detecção de normal.

Base	Cenário	Ataques Gerados	Tráfego (Pacotes)			Tamanho (Megabytes)
			Normal	Ataque	Total	
Detecção de Serviço	*Serviços Conhecidos*	*Probing (Conhecido)*	6.260.424	34.239	6.274.663	731
	Serviços Novos		36.807.285	37.973	36.845.258	12,325
	Serviços Conhecidos	*DoS (Conhecido)*	6.874.239	753.838	7.628.077	972
	Serviços Novos		37.273.872	812.384	38.086.256	12,738
Detecção de Conteúdo	*Conteúdo Conhecido*	*Probing (Conhecido)*	25.240.803	35.782	25.276.585	7,909
	Conteúdo Similar		26.216.937	36.245	26.253.182	7,959
	Conteúdo Novo		30.600.739	38.235	30.638.974	8,905
	Conteúdo Conhecido	*DoS (Conhecido)*	27.376.278	746.287	28.122.565	8,782
	Conteúdo Similar		28.241.742	784.972	29.026.714	8,932
	Conteúdo Novo		33,235,457	797,728	34,033,185	9,748

5.1.6 Discussão

O tráfego gerado é real e válido, uma vez que o *honeypot* dispõe de respostas válidas às requisições recebidas. As classes dos eventos são automaticamente estabelecidas (classificadas no vetor de características), a fim de reduzir a classificação manual e diminuir a possibilidade de erros da pré-classificação, devido à quantidade de pacotes a serem avaliados (todos os cenários geraram em torno de 469 milhões de pacotes). A pré-classificação é efetuada a partir do endereço IP de origem de cada pacote da rede.

A baixa variabilidade, ou ocorrência de tráfego repetido, é mitigada, como o conteúdo que o cliente requisita, tempo entre cada requisição e serviço requisitado é variado. A utilização de ferramentas conhecidas, permite a atualização da base a cada nova vulnerabilidade reportada, uma vez que a ferramenta se torna responsável pela atualização do tráfego e pela garantia da correta implementação. A fim de garantir a reprodutibilidade, o comportamento de todos os clientes e atacantes são gravados. Finalmente, problemas de privacidade não ocorrem, uma vez que a base é obtida em um ambiente controlado e o tráfego gerado não apresenta dados sensíveis.

Através do método proposto, viabilizou-se a criação de 16 bases de intrusão. Cada base objetiva validar as hipóteses comumente adotadas na literatura [8, 14], conforme descrito na seção 4.4. A próxima seção trata sobre a etapa de obtenção dos modelos do ataque.

5.2 Obtenção dos Modelos

Como descrito na seção 4.2, um conjunto de 53 atributos foi elencado. O conjunto de atributos é extraído de cada pacote da rede por um algoritmo implementado na linguagem C++. A leitura dos pacotes é realizada através da biblioteca *lipcap* versão 1.7.4. Uma janela de tempo de dois segundos foi utilizada a fim de computar os atributos baseados no histórico da comunicação (Tabela 3), parâmetro comumente utilizado na literatura [20, 21, 22]. Para cada pacote em cada base, um conjunto de 53 atributos é extraído e o vetor de características é escrito em uma base de eventos separada. Cada vetor de características é automaticamente classificado como *normal* ou *ataque*, baseado em seu IP de origem.

Os tráfegos capturados em cada cenário apresentam distinta proporção entre as classes (Tabelas 6 e 8). A maioria dos eventos são normais em todos os cenários. A geração de uma

base de eventos de treinamento e teste com a proporção igual entre as classes permite que a acurácia seja utilizada como uma métrica válida, sem a necessidade de verificação das taxas de falso-positivo e falso-negativo durante a obtenção dos modelos. Dessa maneira, um processo de estratificação foi utilizado [31] com o objetivo de representar de forma igual cada classe na base de treinamento e teste. O processo de estratificação utilizado consiste em selecionar aleatoriamente 25% dos eventos da classe com menor ocorrência, então o mesmo número de eventos é aleatoriamente escolhido da outra classe. As bases de eventos foram obtidas através do processo de estratificação, utilizando 25% para treinamento, 25% para teste e o restante dos eventos para validação. Toda a base de eventos é utilizada para validação quando a base não é utilizada para treinamento.

Para o processo de obtenção dos modelos dos classificadores, o ambiente Weka versão 3.7.12 foi utilizado. Todos os atributos numéricos foram discretizados de acordo com o método de Usama M. Fayyad e Keki B. Irani [30] para o classificador Naive Bayes. O algoritmo de árvore de decisão C4.5 foi utilizado com um fator de confiança de 0.25. Para o classificador kNN, um total de 5 vizinhos foram utilizados. O kernel RBF [36] foi utilizado para o SVM, com 1 para o parâmetro C e 0,9 para gama. Os parâmetros escolhidos para cada classificador foram selecionados de acordo com os trabalhos correlatos na área de aprendizagem de máquina para de detecção de intrusão.

5.3 Avaliação dos Modelos

Quatro classificadores foram utilizados para avaliar o método proposto de validação de modelo (seção 4.4). Cada classificador é denominado com um prefixo (DT, NB, SVM ou kNN) denominando qual classificador é utilizado durante o processo de detecção (Árvore de Decisão (*Decision Tree*), Naive Bayes, Máquina de Vetores de Suporte (*Support Vector Machine*) ou k Vizinhos Mais Próximos (*k-Nearest Neighbors*). A taxa de falso-positivo (FP) denomina o número de instâncias normais erroneamente classificadas como ataques. Por outro lado, o falso-negativo (FN) é relacionado à quantidade de ataques erroneamente classificados como normais.

As taxas apresentadas para os cenários que são utilizadas durante o treinamento do modelo referem-se à taxa obtida na base de validação, que é obtida através do processo de estratificação detalhado anteriormente (seção 5.2). As taxas de detecção de ataques são

apresentadas na tabela 9. Dois tipos de ataques foram utilizados e cada base segue a distribuição do tráfego apresentada na tabela 6.

Tabela 9 – Taxas obtidas para a detecção de ataques (Tabela 6).

Ataque	Classificador	Cenário								
		Conhecido			Similar			Novo		
		Acurácia (%)	FP (%)	FN (%)	Acurácia (%)	FP (%)	FN (%)	Acurácia (%)	FP (%)	FN (%)
Probing	DT	**99,99**	*0,02*	*0,00*	**98,62**	*0,04*	*2,72*	**64,66**	*0,09*	*70,59*
	NB	**99,75**	*0,27*	*0,22*	**99,04**	*0,15*	*1,76*	**57,38**	*0,18*	*85,06*
	SVM	**99,68**	*0,27*	*0,37*	**98,64**	*0,46*	*2,26*	**56,08**	*0,27*	*87,56*
	kNN	**99,36**	*0,38*	*0,91*	**98,05**	*0,57*	*3,33*	**56,06**	*0,74*	*87,13*
DoS	DT	**99,97**	*0,01*	*0,06*	**99,95**	*0,03*	*0,08*	**75,61**	*0,00*	*48,77*
	NB	**99,90**	*0,09*	*0,11*	**97,76**	*0,79*	*3,69*	**57,29**	*0,00*	*85,41*
	SVM	**99,87**	*0,12*	*0,13*	**98,78**	*0,26*	*2,15*	**65,40**	*31,62*	*37,57*
	kNN	**99,70**	*0,55*	*0,05*	**97,87**	*1,62*	*2,64*	**49,97**	*33,37*	*66,71*

Todos classificadores utilizados obtiveram uma acurácia alta nos cenários *conhecidos*, independentemente do ataque utilizado. O classificador kNN apresentou a pior taxa de detecção com 99,36% de acurácia para a detecção de ataques *probing conhecido* e 99,70% para ataques DoS *conhecido*. A taxa média de FN nos cenários *conhecido* foi somente 0,23%.

Já no cenário similar, torna-se possível notar um aumento na taxa de FN. O pior classificador para ataques *probing* foi o kNN, com 98,05% de acurácia. No entanto, para detecção de ataques DoS o NB apresentou a pior taxa de acurácia resultando em 97,76%. Em média a taxa de FN foi 2,32%, representando um aumento de 2,09% quando comparado à detecção de eventos conhecidos. É possível notar uma pequena degradação na acurácia dos modelos, 1,18% em média. As taxas de FP não sofrem grandes mudanças, 0,27% em média de aumento, devido ao comportamento constante dos clientes legítimos nos cenários de detecção de ataque (seção 4.4.1).

Finalmente, durante a detecção de novos ataques, as taxas de FN aumentam significativamente. A menor taxa de FN foi obtida pelo DT para detecção de ataques *probing*, 70,59%, e pelo SVM para ataques de DoS, 37,57%. Em média, os classificadores apresentaram 71,10% de taxa de FN, um aumento de 70,87% quando comparado à taxa obtida quando ataques conhecidos são detectados. Os resultados obtidos demonstram a ineficiência dos modelos em detectar novos ataques. Nenhum dos classificadores utilizados apresentou a mesma taxa de

acerto obtida durante a detecção de ataques conhecidos. A premissa da capacidade de detectar novos ataques [8] não foi evidenciada durante os testes.

As taxas de detecção de *normal* são apresentadas na tabela 10 e 11. A Tabela 10 mostra a taxa de detecção para serviços, enquanto a Tabela 11 apresenta a taxa de detecção de conteúdos dos serviços.

Tabela 10 – Taxas obtidas para a detecção de serviços (Tabela 8).

Ataques Gerados	*Classificador*	Cenário					
		Conhecido			Novo		
		Acurácia (%)	*FP (%)*	*FN (%)*	*Acurácia (%)*	*FP (%)*	*FN (%)*
Probing (Conhecido)	*DT*	***99,96***	*0,05*	*0,02*	***92,93***	*14,12*	*0,02*
	NB	***99,83***	*0,31*	*0,03*	***96,96***	*6,05*	*0,03*
	SVM	***99,75***	*0,20*	*0,29*	***99,56***	*0,58*	*0,29*
	kNN	***99,54***	*0,43*	*0,49*	***99,05***	*1,40*	*0,50*
DoS (Conhecido)	*DT*	***98,98***	*0,02*	*0,02*	***99,78***	*0,43*	*0,02*
	NB	***99,64***	*0,20*	*0,51*	***97,99***	*3,50*	*0,51*
	SVM	***99,67***	*0,19*	*0,47*	***91,05***	*17,42*	*0,48*
	kNN	***99,37***	*0,36*	*0,91*	***97,63***	*3,68*	*1,06*

Tabela 11 – Taxas obtidas para a detecção de conteúdo (Tabela 8).

Ataques Gerados	*Classificador*	Cenário								
		Conhecido			Similar			Novo		
		Acurácia (%)	FP (%)	FN (%)	Acurácia (%)	FP (%)	FN (%)	Acurácia (%)	FP (%)	FN (%)
Probing (Conhecido)	*DT*	***99,94***	*0,12*	*0,00*	***99,97***	*0,05*	*0,00*	***99,98***	*0,04*	*0,00*
	NB	***99,67***	*0,52*	*0,14*	***98,31***	*3,23*	*0,14*	***96,90***	*6,05*	*0,14*
	SVM	***99,68***	*0,34*	*0,29*	***99,68***	*0,35*	*0,29*	***99,74***	*0,22*	*0,29*
	kNN	***99,16***	*1,16*	*0,52*	***97,12***	*5,25*	*0,51*	***94,11***	*11,27*	*0,51*
DoS (Conhecido)	*DT*	***99,99***	*0,03*	*0,00*	***99,98***	*0,04*	*0,00*	***99,97***	*0,05*	*0,00*
	NB	***99,50***	*0,36*	*0,65*	***98,33***	*2,69*	*0,65*	***96,63***	*6,08*	*0,65*
	SVM	***99,51***	*0,28*	*0,71*	***99,47***	*0,36*	*0,71*	***99,35***	*0,76*	*0,54*
	kNN	***98,85***	*1,06*	*1,24*	***96,93***	*4,86*	*1,29*	***94,06***	*10,96*	*0,93*

Todos os classificadores utilizados foram capazes de detectar eventos normais conhecidos com altas taxas de detecção, a taxa média de acurácia para serviços conhecidos (Tabela 10) foi de 99,59%, enquanto a taxa média de acurácia para conteúdos conhecidos foi de 99,53%.

Quando novos serviços são detectados pelo modelo, a taxa de reconhecimento diminui em média 2,64% para ataques de *probing* e 2,80% para ataques DoS. Dessa maneira, conclui-

se que os modelos são capazes de detectar novos serviços sem grande impacto na sua acurácia. Os ataques gerados não impactam significativamente na taxa de reconhecimento de novos serviços.

Os modelos utilizados também detectaram conteúdos. Nesse panorama, a taxa média de reconhecimento para conteúdos conhecidos no momento em que os ataques de *probing* foram gerados alcançou 99,61%, já o mesmo teste com ataques DoS em execução geraram uma taxa média de 99,46%. No entanto, ocorreu a diminuição da taxa de acurácia no momento em que os conteúdos simulares foram gerados, resultando em 0,84% de perda para *probing* e 0,78% para DoS. Essa redução de acurácia foi identificada de forma evidente no momento de detecção de novos conteúdos, cuja taxa apresentada foi de 1.93% para *probing* e de 1,96% para DoS. Mais uma vez, os ataques gerados não impactaram de forma significativa na taxa de reconhecimento de conteúdo.

Por fim, para o estabelecimento da capacidade de generalização (seção 4.4.3), a conhecida base DARPA1998 foi utilizada. Apesar dos problemas conhecidos e amplamente mencionados na literatura [9, 10, 11], a DARPA1998 é extensivamente utilizada [5] e provém uma referência de comparação razoável entre os especialistas. A base em questão consiste em uma simulação durante o período de 9 semanas de um ambiente de uma base aérea. Para cada dia o DARPA1998 fornece, dentre outros dados, o log do tráfego da rede contendo os pacotes da rede e um arquivo de descrição correspondente, descrevendo as classes para cada conexão.

O extrator de características desenvolvido neste trabalho (seção 5.2) foi modificado a fim de estabelecer as classes dos pacotes de acordo com o arquivo de descrição. As classes dos pacotes foram divididas de acordo com a taxonomia de Kendall [6]. Novamente dois grupos de ataques foram considerados: *probing* e DoS. A tabela 12 apresenta a distribuição do tráfego para as classes utilizadas no DARPA1998.

Tabela 12 – Distribuição do tráfego para o DARPA1998 para as classes consideradas.

Categoria	Classe	Quantidade de Pacotes (Representatividade)
All	Normal	28.426.093 (94,96%)
DoS	Synflood - Neptune	1.507.319 (5,04%)
Probing	Portscan – Nmap	2.211 (0,01%)
Total		**29.935.623**

Apenas ataques a nível de rede foram considerados, uma vez que o método propõe estabelecer a taxa de detecção para o mesmo tipo de ataques de forma independente do ambiente. Para a detecção, utilizou-se os modelos de *Probing (conhecido)* para detecção de *probing* e DoS *(conhecido)* para detecção de DoS (Figura 7). As taxas de detecção obtidas para cada modelo são exibidas na Tabela 13.

Tabela 13 – Taxa de acerto no DARPA1998.

Modelo Utilizado	Classe	Acurácia (%)			
		DT	NB	kNN	SVM
Dos (Conhecido)	**Normal**	**86,52**	**74,31**	**80,27**	**84,99**
	Synflood - Neptune	99,99	99,62	99,97	99,99
Probing (Conhecido)	**Normal**	**90,33**	**82,13**	**76,80**	**78,19**
	Portscan -Nmap	54,32	45,09	73,68	54.32

A taxa de reconhecimento para ataques de DoS em média foi de 99,89% no DARPA1998, o que corresponde a uma melhoria de 0,03% na acurácia obtida no cenário de treinamento do modelo (Tabela 9, 99,86% em média). Observa-se que os modelos mantiveram a acurácia para detecção de ataques DoS em diferentes ambientes. Porém, é notória uma queda significativa na taxa de detecção de ataques *probing*: em média a taxa de detecção foi de 56,85% no DARPA1998 enquanto a taxa média obtida no cenário de treinamento foi de 99,53%.

Nenhum dos modelos foram capazes de manter a taxa obtida no cenário de treinamento para a detecção de eventos normais. Nota-se uma queda significativa na acurácia dos modelos quando utilizados em outro ambiente.

5.3.1 Discussão

Os testes efetuados a fim de avaliar o método de validação proposto (seção 4.4) permitem efetuar algumas observações:

- Todos os modelos foram capazes de detectar os eventos nos cenários em que foram obtidos. O pior classificador foi o kNN, apresentando 99,36% de acurácia para detecção de *probing* e 99,70% para detecção de DoS. Uma taxa significativamente alta, uma vez que não foi adotado um processo de seleção de características para eliminação de atributos não discriminantes;
- Nenhum classificador utilizado manteve a acurácia obtida nos cenários de treinamento durante a detecção de novos ataques (Tabela 9). A premissa de que um NIDS baseado

em anomalia é capaz de detectar novos ataques [8][14] não foi evidenciada durante os testes;

- As técnicas de aprendizagem de máquina utilizadas neste trabalho são capazes de detectar ataques similares, tornando-se uma alternativa viável para detecção de tentativas de evasão, desde que os ataques apresentem um comportamento similar aos ataques utilizados durante a fase de treinamento;
- De forma geral, ocorre uma queda na acurácia durante a detecção de novos serviços (Tabela 10). Porém, a perda da acurácia é significativamente menor quando comparada à detecção de novos ataques (Tabela 9);
- Foi evidenciado um aumento na taxa de FP durante a detecção de novos conteúdos (Tabela 11, em média 1,62% para conteúdos similares e 3,94% para conteúdos novos). Porém, na maioria dos casos, os modelos foram capazes de distinguir corretamente os ataques;
- Quando os modelos são utilizados em outro ambiente (Tabela 13), a acurácia diminui significativamente, mesmo para a detecção de ataques conhecidos. O conjunto de modelos utilizados são dependentes ao ambiente de sua concepção. O comportamento do tráfego durante o processo de treinamento deve ser similar ao comportamento evidenciado no ambiente de produção;

O método de validação proposto permitiu verificar as premissas comumente utilizadas na literatura. Foi observado que quando um modelo classifica eventos com comportamentos conhecidos (durante o treinamento) a taxa de detecção é significativamente alta. Os resultados mostraram uma queda na acurácia durante a detecção de eventos nunca vistos durante o treinamento, propriedade esperada de um ambiente de produção. Um NIDS baseado em anomalia deve ser capaz de detectar tais mudanças do ambiente e atualizar o seu modelo de ataque, sendo confiável em suas classificações enquanto o modelo ainda não é atualizado. A próxima seção avalia o método de rejeição proposto (seção 4.5) como uma alternativa para garantir a confiabilidade dos modelos de ataque em ambientes de produção.

5.4 Método de Rejeição

O método de rejeição proposto pretende manter a confiabilidade do classificador ao longo do tempo, mesmo sem a atualização do modelo. Para alcançar esta esperada propriedade,

o sistema deve rejeitar decisões potencialmente erradas. O método objetiva aumentar a acurácia do sistema, enquanto rejeita a menor quantidade de eventos possível.

Para os testes de rejeição, a base de ataques *Probing* (Tabela 6, *probing conhecido, similar* e *novo*) foi utilizada. O objetivo dos testes é verificar a capacidade de detecção em face a novos comportamentos de ataques. Devido à quantidade de atributos utilizados (53 atributos, Tabelas 2 e 3), os testes para todos os valores possíveis de α para cada atributo e nível de similaridade (seção 5.2.3) não são viáveis. Dessa maneira, dois testes foram efetuados. O primeiro teste utiliza diferentes valores de α para cada grupo de atributo (baseado no cabeçalho, baseado em host e baseado em serviço) denominado *Multiplo Alfa*. Enquanto o segundo teste utiliza o mesmo valor de α para todos os atributos denominado *Único Alfa*, o nível de similaridade variou de 0% a 100% durante os testes, os resultados são exibidos na Figura 15.

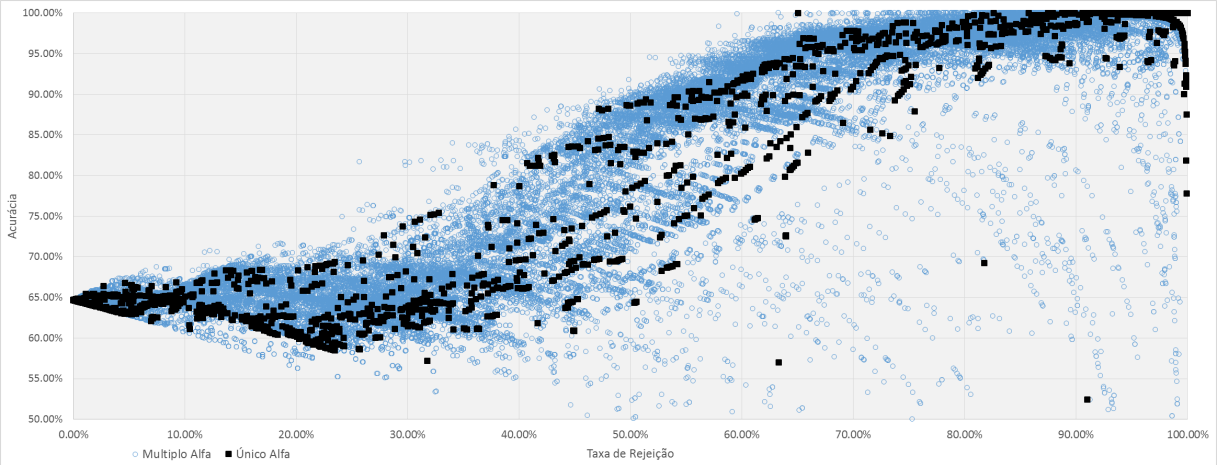

Figura 15 – *Tradeoff* entre a acurácia e a rejeição durante a detecção de novos ataques para o classificador DT.

É possível notar uma relação direta entre a acurácia e a rejeição, independentemente da técnica de obtenção do valor de α. A consideração de diferentes distribuições de frequências entre os grupos dos atributos (*Múltiplo Alfa*) permitiu a melhoria na taxa de acurácia enquanto uma quantidade menor de instancias foram rejeitadas.

A técnica de *Múltiplo Alfa* foi testada para todos os classificadores utilizados. A Figura 16 exibe a melhor acurácia para cada intervalo de rejeição no cenário de novos ataques.

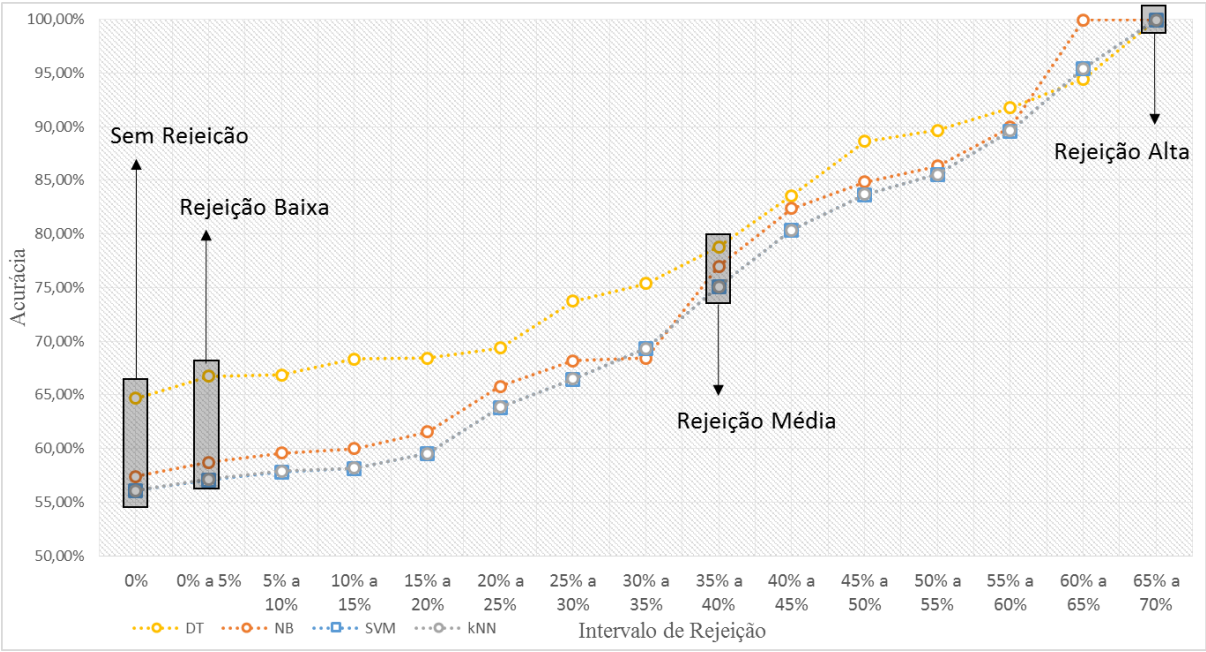

Figura 16 – *Tradeoff* entre a acurácia e a rejeição durante a detecção de novos ataques (DT).

Todos os classificadores testados foram capazes de alcançar 100% de acurácia no intervalo de 65% de rejeição para a detecção de novos ataques (ataques não conhecidos pelos modelos).

Quando o NIDS baseado em anomalia é utilizado em ambientes de produção, este não é capaz de identificar qual tipo de ataque está ocorrendo no momento. Sendo assim, os mesmos parâmetros devem ser utilizados independentemente da instância que o classificador está detectando. Dessa maneira, o conjunto de limiares utilizado deve rejeitar decisões não confiáveis feitas pelos classificadores enquanto aceitam classificações corretas. A Figura 17 exibe o *tradeoff* entre a acurácia e a rejeição para a detecção de ataques conhecidos e similares.

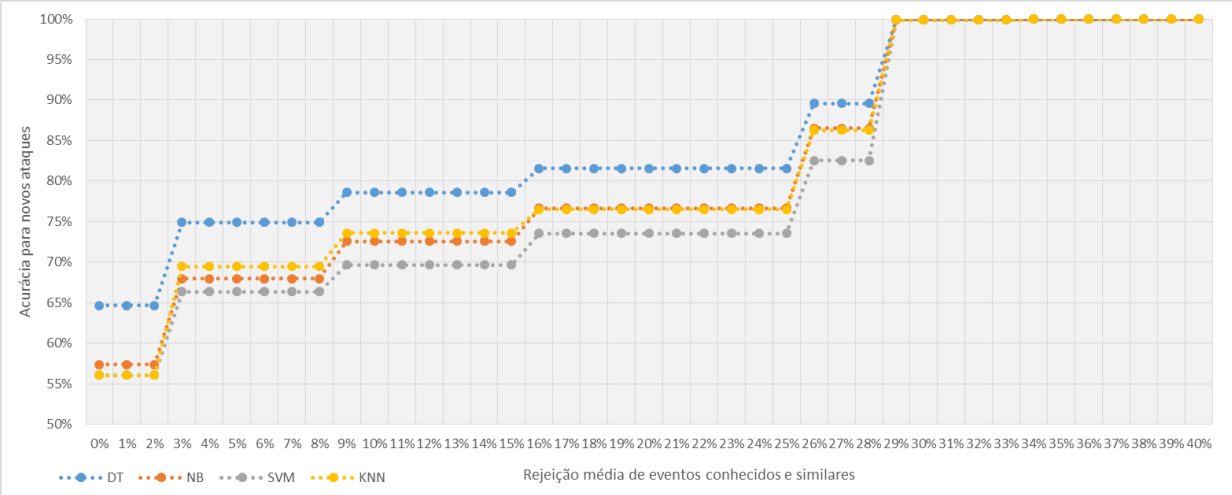

Figura 17 – *Tradeoff* entre a acurácia para detecção de novos ataques e a taxa média de rejeição para eventos *conhecidos* e *similares*.

Tabela 14 – *Tradeoff* entre rejeição e acurácia para cada cenário utilizando os pontos marcados na Figura 16.

Intervalo de Rejeição	Classificador	Cenário					
		Conhecido		*Similar*		*Novo*	
		Acurácia (%)	Rejeição (%)	Acurácia (%)	Rejeição (%)	Acurácia (%)	Rejeição (%)
Sem Rejeição	DT	99,99	-	98,62	-	64,66	-
	NB	99,75	-	99,04	-	57,38	-
	SVM	99,68	-	98,64	-	56,08	-
	kNN	99,36	-	98,05	-	56,06	-
Rejeição Baixa	DT	99,97	0,21	98,70	1,05	67,37	4,82
	NB	99,75	0,14	99,23	0,25	59,66	4,07
	SVM	99,44	0,14	98,48	0,31	58,15	3,98
	kNN	99,45	0,13	98,75	0,40	59,14	4,02
Rejeição Média	DT	99,98	5,40	98,80	5,77	74,90	24,14
	NB	99,75	5,41	99,21	5,88	67,94	23,50
	SVM	99,65	5,71	99,65	6,00	66,37	24,99
	kNN	99,59	5,46	99,35	5,49	69,46	24,26
Rejeição Alta	DT	99,99	37,27	99,87	25,79	99,92	59,57
	NB	99,96	37,34	99,92	25,42	99,92	59,61
	SVM	100,00	37,38	99,87	25,57	99,86	59,62
	kNN	99,95	37,27	99,92	25,38	99,86	59,52

Quando a acurácia para a detecção de novos ataques aumenta, a taxa de rejeição para eventos similares e conhecidos também aumenta. O conjunto de limiares utilizado deve ser estabelecido de acordo com as necessidades do usuário, e o aumento na taxa de detecção de ataques resulta no aumento da quantidade de eventos rejeitados. Porém, se um certo nível de erro for aceitável, umas quantidades menores de eventos serão rejeitados. A tabela 14 exibe a

relação entre a acurácia e a rejeição para cada cenário considerando os limiares dos pontos marcados na Figura 16.

Quatro pontos de rejeição (*sem rejeição, rejeição baixa, rejeição média* e *rejeição alta*) foram selecionados a partir da base de novos ataques. Os mesmos valores dos limiares foram utilizados nos outros cenários para investigar o impacto da taxa de rejeição para a detecção de ataques *conhecidos* e *similares*. O ponto de *rejeição média* apresentou a melhor relação entre rejeição e acurácia. O método de rejeição foi capaz de melhorar a acurácia em 10,24% para detecção de *novos* ataques, enquanto rejeitou apenas 5,40% e 5,77% de eventos *conhecidos* e *similares,* respectivamente, utilizando o classificador DT.

Em resumo, a abordagem de rejeição proposta permitiu a detecção de novos ataques, mantendo a confiabilidade do sistema, apesar da taxa de rejeição relacionada.

5.4.1 Discussão

Os cenários utilizados durante os testes reproduzem três situações esperadas de um ambiente de produção. O cenário *conhecido* representa o comportamento do ambiente quando o sistema é concebido, incluindo todas as variações de comportamentos conhecidas *a priori*. O cenário *similar*, representa a ocorrência de ataques similares aos ataques conhecidos pelo classificador. Finalmente, o cenário *novo* representa o comportamento do ambiente após um determinado período, quando novos ataques ocorrem.

O trabalho apresentou um método de avaliação que leva em consideração tais fatores. Todos os esquemas de detecção foram capazes de detectar eventos *conhecidos* e *similares* com uma taxa alta de detecção (em média 99,69% para eventos *conhecidos* e 98,58% para comportamentos *similares*). Porém, quando o sistema enfrenta ataques não conhecidos (*novos*), a sua acurácia diminui de forma significativa (para uma média de 58,54%).

Para garantir a confiança da classificação, propomos a utilização de simples limiares superiores e inferiores para cada atributo, estabelecidos de acordo com a distribuição observada durante o treinamento do sistema. Em decorrência da quantidade de atributos utilizados, considerou-se que cada grupo de atributo (baseado no cabeçalho, baseado em host o baseado em serviço) apresenta uma distribuição distinta. A abordagem proposta de rejeição foi capaz de garantir a confiabilidade do sistema apresentando um baixo *tradeoff* entre a rejeição e a acurácia, apresentando uma melhoria na acurácia em 10,24% para a detecção de *novos* ataques enquanto rejeita apenas 5,40% de comportamentos *conhecidos* utilizando o classificador DT.

Os experimentos realizados mostraram fatores que devem ser considerados pela comunidade de detecção de intrusão baseada em anomalia. Um classificador funciona através da identificação de comportamentos similares e deve possuir uma quantidade significativa de instâncias de todas as classes consideradas [14]. A premissa de que um classificador será capaz de identificar novos ataques apenas ocorre quando o comportamento do ataque é similar ao comportamento evidenciado durante o treinamento do sistema e conforme aferido durante os experimentos. O sistema de aprendizagem de máquina se torna não confiável quando o comportamento do tráfego sofre alteração. Neste trabalho, tal propriedade foi tratada através da rejeição de decisões potencialmente erradas. Fator este, não tratado na literatura [8], em que o tráfego é considerado de forma estática e a abordagem tradicional de validação de aprendizagem de máquina é adotada sem considerar as propriedades do tráfego da rede.

A seção seguinte apresenta a comparação com uma ferramenta de mercado amplamente utilizada para detecção de ataques a nível de rede.

5.5 Comparação com ferramenta de mercado

Para a comparação com uma ferramenta de mercado foi utilizado o Snort[6], um IDS baseado em assinatura amplamente conhecido. Para o propósito dos testes, foi utilizado a base DoS (*conhecida)* (Tabela 6) que apresenta um conjunto de ataques detectáveis através de assinaturas. A detecção através de assinaturas é possível através do estabelecimento de métricas relacionadas a cada conexão dos clientes e à definição de limiares para cada valor das métricas.

O formato da assinatura utilizada é exibido na Figura 18. Durante os testes, dois parâmetros foram variados: (i) *Quantidade de Ocorrências* e (ii) *Intervalo*. Qualquer tentativa de conexão que exceda à *Quantidade de Ocorrências* durante o *Intervalo* definido é identificada como uma tentativa de intrusão. No total, 10.000 testes foram realizados utilizando a base DoS *(conhecida)*. Ambos parâmetros sofreram variações com valores entre 1 a 1.000. A Figura 19 exibe a acurácia obtida em relação a cada variação dos parâmetros utilizados. A *Quantidade de Ocorrências* é exibida até 250, uma vez que a acurácia permanece constante após o aumento deste parâmetro.

[6] https://snort.org/

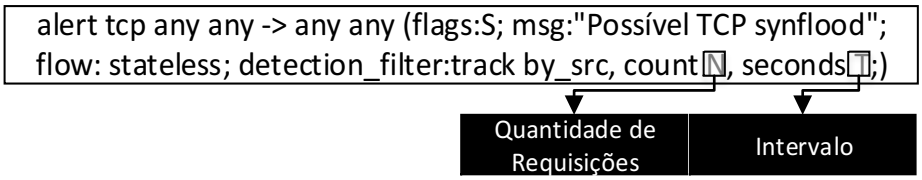

Figura 18 – Formato de assinatura utilizada.

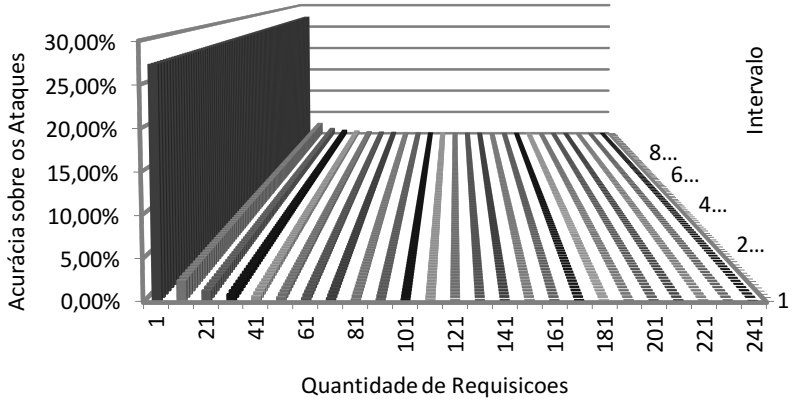

Figura 19 – Acurácia obtida pelo Snort (utilizando assinatura da Figura 18).

A acurácia do Snort é estabelecida de acordo com a quantidade de tentativa de conexões efetuadas pelos atacantes que foram corretamente identificadas pela ferramenta. A melhor acurácia utilizou 3 conexões (*Quantidade de Ocorrências*) em uma janela de tempo de 2 segundos (*Intervalo*), obtendo 27,32% de acurácia (Figura 19). Através da comparação dos resultados, torna-se possível observar que, para a detecção de ataques DoS em nível de rede, a proposta apresentada no presente trabalho apresenta em média uma taxa de detecção 3 vezes maior do que a abordagem baseada em assinatura.

Finalmente, comparou-se o tempo de processamento demandado por cada abordagem de detecção: baseada em assinatura através do Snort e baseada em anomalia através da abordagem proposta neste presente trabalho. As medidas foram realizadas em um Ubuntu 14.04 com um Intel I7 e 16GB de memória. Cada medição foi executada 100 vezes, a média de processamento de cada módulo é considerada.

Para o Snort, dois conjuntos de assinaturas foram utilizadas. O primeiro conjunto utiliza apenas a assinatura exibida na Figura 18 (com *Quantidade de Ocorrências* = 3 e *Intervalo* = 2). Esse conjunto de assinatura é denominado *Snort DoS* e visa detectar apenas ataques DoS. O segundo conjunto de assinaturas é denominado *Snort Padrão* e utiliza o conjunto padrão de assinaturas (para os experimentos foi utilizado o conjunto *Snort snapshot 2.9.62*) fornecido pelo

Snort. Dessa maneira, torna-se possível estabelecer o processamento requerido pelo Snort, quando o administrador pretende detectar apenas ataques DoS, mas também quando utiliza o conjunto padrão de assinaturas.

Para a abordagem baseada em anomalia, utilizou-se os classificadores DT e NB devido ao seu baixo custo computacional. Para o estabelecimento do tempo de processamento, uma chamada de retorno foi adicionada no início de cada módulo do Snort e da abordagem proposta. A tabela 15 apresenta a média de processamento requerido por cada módulo, *Captura de Pacotes*, *Extração de Características (Pré-processamento* para o Snort) e por fim o *Mecanismo de Detecção*.

Tabela 15 – Comparação do tempo de processamento.

Módulo	**Tempo de Processamento (Segundos)**			
	Snort DoS	*Snort Padrão*	Abordagem Proposta NB	Abordagem Proposta DT
Captura de Pacotes	125,0	138,0	19,3	19,4
Extração de Características – Pré-Processamento	113,6	979,8	31,3	31,3
Mecanismo de Detecção	46,4	400,2	7,9	1,7
Total	***285,0***	***1518,0***	***58,5***	***52,4***

Independentemente do conjunto de assinaturas utilizadas, a abordagem proposta neste trabalho apresentou menores taxas de processamento em relação ao Snort. No momento que o classificador NB encontra-se em execução, ocorre uma demanda de tempo de processamento de 18,39% e 3,45% em relação ao *Snort DoS* e *Snort Padrão,* respectivamente. Seguindo a mesma linha, o DT demandou apenas 20,53% e 3,85%.

Capítulo 6

Conclusão

As abordagens recorrentes na literatura na área de detecção de intrusão baseada em anomalia na rede não são suficientemente confiáveis para utilização em ambientes de produção. O presente trabalho teve como proposta e implementação um novo método de avaliação que avalia as propriedades esperadas, além das premissas comumente utilizadas para um NIDS baseado em anomalia.

Um novo método para criação de bases de intrusão é avaliado, tencionando a fácil atualização e reprodução dos testes. A técnica de *honeypot* é utilizada para a geração do tráfego por parte do servidor, enquanto ferramentas de *workload* são utilizadas para a geração do tráfego do cliente. O método proposto permite a fácil criação de ambientes relativamente complexos com ausência dos principais desafios reportados na literatura.

Além disso, através do método de criação de bases de intrusão proposto, um conjunto de cenários são criados a fim de avaliar as premissas comumente listadas na literatura. O método de avaliação proposto revelou que a abordagem de detecção de intrusão através da aprendizagem de máquina é eficiente para a detecção de ataques conhecidos e de suas variações.

Os resultados expressaram que premissas originadas pelo trabalho de Dorothy E. Denning [38] sobre HIDS não se sustentam quando NIDS é considerado, nenhuma das abordagens utilizadas foi capaz de detectar novos ataques. Entretanto, os classificadores testados foram capazes de detectar novos serviços e novos conteúdos de serviços e serviços com um pequeno impacto na acurácia do sistema.

Para prover confiabilidade à classificação, apresentou-se um novo método de rejeição que considera as mudanças de perfis na rede ao longo do tempo, através do estabelecimento da

similaridade do perfil, de acordo com os valores atuais das características. Através do método proposto, tornou-se possível rejeitar uma quantidade pequena de eventos *conhecidos* e *similares*, sendo 5,40% e 5,77% de forma respectiva. Enquanto a acurácia do sistema foi melhorada em 10,24% na detecção de *novos* perfis, garantiu-se a confiabilidade do sistema diante de novos comportamentos do tráfego da rede.

Finalmente, foi aferido que apesar de sua baixa utilização em ambientes de produção, a abordagem baseada em anomalia na detecção de ataques DoS apresenta uma melhor taxa de detecção (em média 99,41% contra 27,32%) quando comparado a um IDS de mercado baseado em assinatura. A abordagem baseada em anomalia também apresentou um tempo de processamento significativamente menor quando em sua comparação com o Snort, uma ferramenta de mercado baseado em assinatura.

Referências

[1] Kaspersky Lab. ZAO. Kaspersky Security Bulletin 2014: Overall statistics for 2014. [online] disponível em: securelist.com/analysis/kaspersky-securitybulletin/68010/kaspersky-security-bulletin-2014-overall-statistics-for-2014/.Acesso Jan./2016

[2] Symantec Lab. ISTR20: Internet Security Threat Report, [online] disponível em: www4.symantec.com/mktginfo/whitepaper/ISTR/21347932_ GA- internetsecurity-threat-report-volume-20-2015-social_v2.pdf. Acesso Jan./2016.

[3] Tsai, C. F., Hsu, Y. F., Lin, C. Y. e Lin, W. Y. (2009). Intrusion detection by machine learning: A review. *Expert Systems with Applications*, v. 36, n. 10, p. 11994–12000.

[4] Corona, I., Giacinto, G. e Roli, F. (ago 2013). Adversarial attacks against intrusion detection systems: Taxonomy, solutions and open issues. *Information Sciences*, v. 239, p. 201–225.

[5] Tavallaee, M., Stakhanova, N. e Ghorbani, A. A. (2010). Toward credible evaluation of anomaly-based intrusion-detection methods. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, v. 40, n. 5, p. 516–524.

[6] Kendall, K. (1999). A Database of Computer Attacks for the Evaluation of Intrusion Detection Systems. Tese de Doutorado

[7] Shiravi, A., Shiravi, H., Tavallaee, M. e Ghorbani, A. a. (2012). Toward developing a systematic approach to generate benchmark datasets for intrusion detection. *Computers and Security*, v. 31, n. 3, p. 357–374.

[8] Sommer, R. e Paxson, V. (2010). Outside the Closed World: On Using Machine Learning for Network Intrusion Detection. *Proceedings of the IEEE Symposium on Security and Privacy*, n. May, p. 305–316.

[9] McHugh, J. (2000). Testing Intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory. *ACM Transactions on Information and System Security*, v. 3, n. 4, p. 262–294.

[10] Mahoney, M. V e Chan, P. K. (2003). An Analysis of the 1999 DARPA/Lincoln Laboratory Evaluation Data for Network Anomaly Detection. *In Proceedings of the Sixth International Symposium on Recent Advances in Intrusion Detection*, v. 2820, n. Ll, p. 220–

237.

[11] Lippmann, R. P., Fried, D. J., Graf, I., et al. (2000). Evaluating intrusion detection systems: the 1998 DARPA off-line intrusion detection evaluation. *Proceedings DARPA Information Survivability Conference and Exposition. DISCEX'00*, v. 2.

[12] Paxson, V. e Floyd, S. (1995). Wide-Area Traffic: The Failure of Poisson Modeling. *IEEE/ACM Transactions on Networking*, v. 3, n. 3, p. 226–244.

[13] Tavallaee, M., Bagheri, E., Lu, W. e Ghorbani, A. a. (2009). A detailed analysis of the KDD CUP 99 data set. *IEEE Symposium on Computational Intelligence for Security and Defense Applications, CISDA 2009*, n. Cisda, p. 1–6.

[14] Gates, C. e Taylor, C. (2007). Challenging the Anomaly Detection Paradigm: A Provocative Discussion. *Proceedings of the 2006 workshop on New security paradigms*, p. 21–29.

[15] Canali, D., Cova, M., Vigna, G. e Kruegel, C. (2011). Prophiler : A Fast Filter for the Large-Scale Detection of Malicious Web Pages Categories and Subject Descriptors. *Proc. of the International World Wide Web Conference (WWW)*, p. 197–206.

[16] CAIDA. The cooperative association for internet data analysis [online] disponível em: http://www.caida.org/. Acesso Nov./2015.

[17] Maggi, F., Robertson, W., Kruegel, C. e Vigna, G. (2009). Protecting a moving target: Addressing web application concept drift. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, v. 5758 LNCS, p. 21–40.

[18] Nwanze, N. e Summerville, D. (2008). Detection of anomalous network packets using lightweight stateless payload inspection. *Proceedings - Conference on Local Computer Networks, LCN*, p. 911–918.

[19] Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M. e Bouchachia, A. (2014). A survey on concept drift adaptation. *Computing Surveys*, v. 46, n. 4, p. 1–37.

[20] Kapp, M. N., Sabourin, R. e Maupin, P. (2010). Adaptive incremental learning with an ensemble of support vector machines. *Proceedings - International Conference on Pattern Recognition*, p. 4048–4051.

[21] González-Castro, V., Alaiz-Rodríguez, R. e Alegre, E. (2013). Class distribution estimation based on the Hellinger distance. *Information Sciences*, v. 218, p. 146–164.

[22] Kmieciak, M. R. e Stefanowski, J. (2011). Semi-supervised approach to handle sudden concept drift. *Control and Cybernetics* v. 40, n. 3.

[23] Hoens, T. R., Polikar, R. e Chawla, N. V. (2012). Learning from streaming data with concept drift and imbalance: an overview. *Progress in Artificial Intelligence*, v. 1, n. 1, p. 89–101.

[24] Kim, S. e Nwanze, N. (2008). Noise-Resistant Payload Anomaly Detection for Network Intrusion Detection Systems. *Proceedings of the Performance, Computing and Communications Conference (IPCCC 2008)*, p. 517–523.

[25] Wang, K. e Stolfo, S. (2004). Anomalous payload-based network intrusion detection. *Recent Advances in Intrusion Detection*, p. 203–222.

[26] Mahoney, M. e Chan, P. K. (2001). PHAD: Packet header anomaly detection for identifying hostile network traffic. *Florida Institute of Technology technical report CS-2001-04*, n. 1998, p. 1–17.

[27] Mahoney, M. V. (2003). Network traffic anomaly detection based on packet bytes. *Proceedings of the 2003 ACM symposium on Applied computing - SAC '03*, p. 346.

[28] Oliveira, L. S., Sabourin, R., Bortolozzi, F. e Suen, C. Y. (2002). Automatic recognition of handwritten numerical strings: A Recognition and Verification strategy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 24, n. 11, p. 1438–1454.

[29] Acuna, E. e Rodriguez, C. (2004). The treatment of missing values and its effect on classifier accuracy. *Classification, Clustering, and Data Mining*, n. 1995, p. 1–9.

[30] Fayyad, U. M. e Irani, K. B. (1993). Multi-Interval Discretization of Continuos-Valued Attributes for Classification Learning. *Proceedings of the International Joint Conference on Uncertainty in AI*.

[31] Provost, F. (2000). Machine Learning from Imbalanced Data Sets 101 Extended Abstract. *Proceedings of the AAAI'2000 workshop on imbalanced data sets*, p 1-13

[32] Komviriyavut, T.; Sangkatsanee, P.; Wattanapongsakorn, N.; Charnsripinyo, C. (2009). Network intrusion detection and classification with Decision Tree and rule based approaches. *9th International Symposium Communications and Information Technology, 2009. ISCIT 2009*, p. 1046–1050.

[33] P. Gogoi, M. H. Bhuyan, D. K. Bhattacharyya e J. K. Kalita (2012), Packet and Flow Based Network Intrusion Dataset, *Contemporary Computing*, p. 322-334.

[34] Davis, J. J. e Clark, A. J. (2011). Data preprocessing for anomaly based network intrusion detection: A review. *Computers & Security*, v. 30, n. 6-7, p. 353–375.

[35] Cisco Visual Networking Index: Forecast and Methodology. Acesso em Jul./2015.

[36] Olivo, C. K., Santin, A. O. e Oliveira, L. S. (2013). Obtaining the threat model for e-mail phishing. *Applied Soft Computing Journal*, v. 13, n. 12, p. 4841–4848.

[37] Denning, D. E. (2012). An intrusion-detection model. *Proceedings - IEEE Symposium on Security and Privacy*, n. 2, p. 118–131.

[38] L. M. Freund e Yoav. The alternating decision tree learning algorithm. *Int. Conf. Machine Learning*, vol. 99, p. 124-133, 1999.

[39] David D. Lewis. Naïve (Bayes) at Forty: The Independence Assumption in Information Retrival, Proceedings of the 10th European Conference on Machine Learning, p. 4-15, 1998.

[40] James M. Keller, Michael R. Gray e James A. Givens. A Fuzzy k-Nearest Neighbor Algorithm, *IEEE Transactions on Systems, Man and Cybernetics*, v. 15, p. 580-585.

[41] Marti A. Hearst. Support Vector Machines. *IEEE Intelligent Systems and their Appliactions*. v. 13, p.18-38.