RIVALDO LUIZ TOMIO

UM MODELO BASEADO EM APRENDIZAGEM DE MÁQUINA PARA DETECÇÃO AUTÔNOMA DE INTRUSÃO

Dissertação de mestrado submetida ao cumprimento de requisito parcial para o mestrado em Ciência da Computação no Programa Pós-Graduação em Ciência da Computação da Pontifícia Universidade Católica do Paraná, Brasil.

CURITIBA

2021

RIVALDO LUIZ TOMIO

UM MODELO BASEADO EM APRENDIZAGEM DE MÁQUINA PARA DETECÇÃO AUTÔNOMA DE INTRUSÃO

Dissertação de mestrado submetida ao cumprimento de requisito parcial para o mestrado em Ciência da Computação no Programa Pós-Graduação em Ciência da Computação da Pontifícia Universidade Católica do Paraná, Brasil.

Área de Concentração: Ciência da Computação
Orientador: Dr. Eduardo Kugler Viegas
Co-orientador: Dr. Altair Olivo Santin

CURITIBA

2021

Dedico este trabalho à Ludmila Platek B. (in memorian) e Sigmund Balcewicz (in memorian) pela contínua sensatez e elegante perspicácia.

Agradecimentos

Agradeço ao professor Eduardo Kugler Viegas, por estar sempre acessível durante a orientação do mestrado, e pelo aprendizado acadêmico e pessoal.
Um agradecimento especial aos meus familiares.
Agradeço também ao meu co-orientador Altair Olivo Santim e aos funcionários do PPGIA.
Gostaria de agradecer aos colegas do PPGIA, em especial, a Roger, Felipe e Jonathan.
Meus agradecimentos ao apoio institucional da Fundação Araucária e Renault Groupe.

Sumário

Sumário

Agradecimentos 3

Sumário 4

Lista de Figuras 7

Lista de Tabelas 8

Lista de Algoritmos 9

Resumo 10

Lista de Abreviações 11

Capítulo 1 12

Introdução 12

1.1 Motivação 13

1.2 Objetivo 14

1.3 Contribuições 15

1.4 Estrutura do Documento 15

Capítulo 2 17

Fundamentação 17

2.1 Lapsos temporais 17

2.1.1 Definição de Janela de Vulnerabilidade 18

2.1.2 Definição de Lacuna de Segurança da Informação 18

2.2 IDS 19
2.3 IDS baseado em anomalia 22
2.3.1 IDS baseado em assinatura 24
2.4 IDS baseado em redes (NIDS) 25
2.4.1 IDS baseado em hosts (HIDS) 30
2.5 Aprendizagem de Máquina para IDS 32
2.5.1 Aprendizagem de Máquina Semi-supervisionada 33
2.6.1 Co-training 34
Capítulo 3 36
Trabalhos Relacionados 36
3.1 Aprendizagem de Máquina para IDS 36
3.2 Aprendizagem de Máquina Semi-supervisionada para IDS 37
3.3 Co-Learning para IDS 38
3.4 Mudança de comportamento em IDS 39
3.5 Discussão 40
Capítulo 4 43
Proposta 43
4.1 Visão Geral 44
4.2 Classificação Multivisão Confiável 45
4.3 Atualização Autônoma do Modelo 47
4.4 Discussão 48
Capítulo 5 50
Avaliação 50
5.1 Definição do problema 50
5.1.1 Conjunto de dados MAWIFlow 51
5.1.2 Performance das técnicas tradicionais 55

5.2 Avaliação da proposta 57
5.2.1 Treinamento dos modelos 57
5.2.2 Performance ao longo do tempo 57
5.3 Discussão 61
Capítulo 6 63
Conclusão 63
Referências 65
Apêndice 1 69

Lista de Figuras

Figura 1: Ilustra o processo do co-training onde cada classificador provê instâncias de rótulos para o outro classificador – adaptado de [25] 35
Figura 2: Estrutura do modelo multivisão proposto 43
Figura 3: Visão Nigel Random Forest (100 árvores) 56
Figura 4: Visão Orunada Random Forest (100 árvores) 56
Figura 5: Visão Viegas Random Forest (100 árvores) 56
Figura 6: Class-Related-Threshold (CRT) 58
Figura 7: Taxa de acurácia e rejeição sem atualização de modelos. 60
Figura 8: Taxa de acurácia e rejeição com atualização autônoma de modelos. 60
Figura 9: Taxas de Falso Negativos ao longo do tempo. 61
Figura 10: Taxas de Falso Positivos ao longo do tempo. 61

Lista de Tabelas

Tabela 1: Exemplos de HIDS 32
Tabela 2: Exemplos NIDS 30
Tabela 3: Visão Nigel 52
Tabela 4: Visão Orunada 53
Tabela 5: Visão Viegas 55

Lista de Algoritmos

Algoritmo 1: Classificação Multivisão Confiável 46
Algoritmo 2: Atualização Autônoma do Modelo 48

Resumo

As mudanças do comportamento do tráfego de rede ao longo do tempo são, de modo geral, negligenciadas por autores que fazem uso de técnicas de aprendizagem de máquina aplicadas à detecção de intrusão. Geralmente, assume-se que atualizações periódicas do modelo podem ser facilmente realizadas, independentemente do custo computacional e operacional relacionado às atualizações do mecanismo de detecção. Neste trabalho, propõe-se um novo modelo de detecção de intrusão capaz de atuar de maneira confiável sem a assistência humana durante as atualizações e ainda mantendo a acurácia ao longo do tempo. Para tanto, são avaliados os valores de confiança das classificações geradas pela arquitetura multivisão (multi-view) proposta para manter a confiança (*reliability*) geral do sistema ao longo do tempo sem atualizações. Não obstante, a arquitetura proposta é capaz de atualizar os modelos de forma autônoma, sem assistência humana, de acordo com o resultado das classificações. Os experimentos realizados através de um tráfego real, com 2 anos de duração, demonstraram que o método multi-visão proposto é capaz de permanecer robusto por períodos mais extensos quando comparados a literatura, aumentando sua taxa de falso-positivo (FP) em média em apenas 3,5% no intervalo de 2 anos, em contraste com a técnica de visão única, que aumenta seu FP em até 9,2%.

Palavras-chave: *Detecção de Intrusão; Aprendizagem de máquina; Multi-view.*

Lista de Abreviações

GB	Gigabyte
HIDS	Host-based Intrusion Detection System
IDS	Intrusion Detection System
MAWI	Packet traces from WIDE backbone
MAWILAB	Related work that provides daily labeling of network traces from MAWI
NIDS	Network-based Intrusion Detection System
PCAP	Packet Capture
RF	Random Forest
TB	Terabyte
IPS	Intrusion Prevention System
AM	Aprendizagem de Máquina
FP	False Positive
TP	True Positive
FN	False Negative
TN	True Negative

Capítulo 1

Introdução

Os ataques cibernéticos ainda são um problema mundial que precisa ser tratado de todas as formas possíveis. Os números da empresa de segurança cibernética Karpersky mostram o quão grande é esse problema. Somente no segundo trimestre de 2020, as soluções Kaspersky bloquearam mais de 800 milhões de ataques cibernéticos realizados online em 191 países em todo o mundo [1].

A primeira ferramenta que um administrador de sistemas dispõe para se proteger de ataques vindos da internet ou de hosts de sua própria rede é um Sistema de Detecção de Intrusão Baseado em Rede (*Network-based Intrusion Detection System,* NIDS). Um NIDS trabalha analisando pacotes de rede que transitam em seu domínio, seja comparando com ataques já conhecidos, seja deduzindo comportamentos maliciosos das anormalidades encontradas. Dispara alarmes quando encontra um comportamento malicioso e deixa passar de maneira discreta todos os pacotes tidos como tráfego normal de rede. Um NIDS pode ser usado para proteger apenas um computador, trabalhando de forma personalizada para o tráfego daquele ponto da rede. Logo, a capacidade de filtrar o que é ataque torna um NIDS a ferramenta ideal para trabalhar como um elemento estrutural de segurança na arquitetura de rede. Seu trabalho de proteção pode ser usado de forma mais generalizável para links de rede, especialmente links *backbones* (de infraestrutura).

Um NIDS pode comumente ser classificado tanto como baseado em assinatura ou como baseado em anomalia. O NIDS baseado em assinatura é o mais utilizado em ambientes de

produção. Possui alta acurácia e baixa taxa de falsos positivos, no entanto, como só reconhece aqueles ataques que constam em seu banco de assinaturas, não tem a capacidade de detectar novos ataques que ainda não foram documentados, estando assim sempre alguns passos atrás do atacante. Por sua vez, o NIDS baseado em anomalia parte do pressuposto que um ataque gerará uma anomalia no sistema e que essa anomalia pode ser reconhecida. Esse tipo de IDS tem a capacidade de detectar ataques ainda não documentados, porém apresenta, como efeito colateral, a inconveniência de gerar muitos falsos-alarmes.

A utilização de aprendizagem de máquina (AM) como módulo de classificação para um NIDS é promissora, visto que essa técnica já é utilizada em várias outras áreas com sucesso. Ao treinar um modelo com um conjunto de dados de treinamento, esse modelo é aplicado a novas instâncias do mesmo evento no conjunto de dados. Essa técnica, quando configurada corretamente, é capaz de fazer previsões com altos níveis de precisão. Abordagens baseadas em aprendizagem de máquina são capazes de melhorar potencialmente o desempenho do NIDS, identificando tanto ataques conhecidos quanto desconhecidos (novos).

1.1 *Motivação*

Na literatura, o NIDS baseado em anomalia tem sido frequentemente realizado por meio de técnicas de aprendizagem de máquina, em que abordagens de reconhecimento de padrões são normalmente utilizadas. Este tipo de esquema de detecção depende de um conjunto de dados de treinamento para extrair um modelo comportamental. Então, o modelo construído pode ser usado para a classificação de outros eventos [2]. Consequentemente, se o comportamento do ambiente mudar, por exemplo, um novo ataque é descoberto ou um novo serviço é fornecido, o modelo de AM subjacente se torna não confiável, ou seja, a acurácia da predição diminui. Isso ocorre porque o conjunto de dados de treinamento no qual o modelo de AM foi construído não contém o comportamento do ambiente atual, tornando o modelo de AM desatualizado [3].

Modelos obsoletos de AM não conseguem atingir o mesmo nível de precisão daqueles medidos durante a fase de teste. Assim, devido a um aumento na taxa de erro ao longo do tempo, os operadores muitas vezes descartam outros alarmes [4]. No entanto, apesar das mudanças no comportamento do tráfego de rede ao longo do tempo serem um problema conhecido no campo NIDS, tal desafio é frequentemente negligenciado em outros trabalhos [5]. Na literatura, a

maioria dos esquemas de detecção de intrusão baseados em AM propostos buscam maior precisão de classificação, prestando pouca ou nenhuma atenção aos desafios envolvidos durante a tarefa de atualização do modelo[6]. Em contraste, os pesquisadores muitas vezes presumem que o tráfego da rede é estácionario e nenhuma mudança ocorre ao longo do tempo, ou mesmo que atualizações periódicas do modelo são realizadas, sem nem mesmo avaliar a necessidade de tais atualizações. Idealmente, o modelo de AM deve ser o mais recente possível, levando em consideração que o comportamento do tráfego da rede pode mudar drasticamente em um pequeno período, tornando-o pouco confiável [7]. No entanto, a tarefa de retreinamento do modelo é um processo computacionalmente caro que muitas vezes exige assistência humana para a rotulagem de eventos, por exemplo, marcar o tráfego de rede como normal ou de ataque, que nem sempre está disponível ou está disponível com um alto custo [8]. Como resultado, a tarefa de atualização do modelo de AM permanece esquecida na literatura e a vida útil das técnicas baseadas em AM amplamente utilizadas ainda não são avaliadas.

1.2 *Objetivo*

O objetivo geral deste trabalho é desenvolver uma abordagem de detecção de intrusão em nível de rede que utilize técnicas baseadas em co-learning fazendo uso de diversidade de dados ao longo do tempo e de maneira confiável (com alta acurácia a longo prazo). A abordagem busca adaptar-se às mudanças constantes do tráfego de rede ao longo do tempo de modo autônomo, ou seja, sem o auxílio de um especialista para prover os rótulos dos eventos. Para tanto, este trabalho parte da premissa de que o uso de diversidade de dados pode alavancar a classificação correta do que é ataque e do que é tráfego normal. Para prover confiabilidade de modo autônomo por longos períodos, somente classificações com alto grau de confiança farão parte da solução. Deste modo, o IDS proposto é capaz de atuar de forma mais robusta ao longo do tempo em comparação com as abordagens tradicionais, mesmo quando desatualizado em face a novos comportamentos no tráfego de rede.

Sendo assim, para atingir o objetivo do trabalho os seguintes objetivos específicos devem ser alcançados:

1. Avaliar algoritmos tradicionais de aprendizagem de máquina supervisionados para detecção de intrusão de rede durante longos períodos;

2. Desenvolver e avaliar uma abordagem confiável de detecção de intrusão através de técnica multivisão;
3. Desenvolver e avaliar uma abordagem que alie a técnica de multivisão e atualização autônoma ao longo do tempo.

1.3 *Contribuições*

Este trabalho avança o estado da arte em detecção de intrusão em nível de rede ao prover um modelo baseado em aprendizagem de máquina confiável capaz de tratar as mudanças de comportamento no tráfego de rede de modo autônomo, ou seja, sem a intervenção humana para o provimento dos rótulos dos eventos de rede. Mais especificamente este trabalho visa a contribuir da seguinte forma:

- Comparar o desempenho de algoritmos de aprendizagem de máquina aplicado a visões únicas de dados com o desempenho aplicado a multivisão. O trabalho demonstra que as técnicas atualmente utilizadas na literatura são incapazes de tratar as mudanças de tráfego de rede ao longo do tempo, demandando atualizações periódicas aos modelos;
- Um novo modelo de detecção de intrusão baseado em co-learning através de diversidade de dados capaz de atuar de modo confiável ao longo do tempo mesmo sem atualizações sendo realizadas. Além disso, o modelo é capaz de atualizar os modelos de aprendizagem de máquina de maneira autônoma sem intervenção humana;

Devido as contribuições acima elencadas, o trabalho conta com uma publicação em conferência internacional qualis A1, sendo ela:

TOMIO, RIVALDO L. ; VIEGAS, EDUARDO K. ; SANTIN, ALTAIR O. ; DOS SANTOS, ROGER R. . *A Multi-View Intrusion Detection Model for Reliable and Autonomous Model Updates*. In: ICC 2021 IEEE International Conference on Communications, 2021, Montreal. ICC p. 1-6.

1.4 *Estrutura do Documento*

O restante deste documento tem a seguinte organização. O Capítulo 2 descreve os fundamentos teóricos que embasam este trabalho. O Capítulo 3 analisa trabalhos relacionados a vários temas abordados na presente proposta, e ressalta pontos em comum e pontos

divergentes. No Capítulo 4 se descreve em detalhes a arquitetura proposta. No Capítulo 5 são avaliados os resultados alcançados. E finalmente no capítulo 6 se descreve a conclusão dos trabalhos.

Capítulo 2

Fundamentação

Neste capítulo são descritos os fundamentos relativos a Sistemas de Detecção de Intrusão (IDS), seus vários tipos e classificações, bem como a definição de janela de vulnerabilidade, a definição de lacuna de segurança da informação, o embasamento de aprendizagem de máquina para IDS, aprendizagem semi-supervisionada e finalmente os princípios da técnica de Co-learning.

2.1 Lapsos temporais

Entre a descoberta de uma vulnerabilidade por parte de uma equipe de segurança, ou pela equipe de desenvolvimento do software envolvido até a atualização (patch) de segurança, há lapsos de tempo relevantes para a área de segurança da informação. Esses lapsos de tempo são relativos às atividades que estão sendo realizadas para tratar uma vulnerabilidade e que na prática podem ser exploradas por pessoas mal-intencionadas.

Um NIDS precisa atuar mesmo dentro desses lapsos temporais, detectando ameaças que não estão totalmente documentadas.

A seguir é definido os diferentes lapsos de tempos que envolvem essa disponibilização de atualizações de segurança (patches) e como isso é tratado pela cadeia de desenvolvimento e suporte de sistemas.

2.1.1 Definição de Janela de Vulnerabilidade

A janela de vulnerabilidade é o lapso de tempo entre a descoberta de uma vulnerabilidade de um software e disponibilidade de atualizações de segurança (patches) efetivada pela equipe de desenvolvimento de software. Considera-se sensato anunciar a vulnerabilidade mesmo antes de se disponibilizar a atualização de segurança para que os administradores de sistemas mais atentos possam tomar medidas de contenção se necessário.

Nesse período, atacantes podem vir a explorar a vulnerabilidade ou mesmo automatizar uma maneira de ataque e disponibilizá-la para uso. Outros atacantes interessados em explorar essa vulnerabilidade, seja para fins lícitos como testes de sistema, seja para fins ilícitos como invasão de sistemas, podem, dessa forma, realizar um ataque sem mesmo conhecer em detalhes a falha explorada.

Como exemplo, a Microsoft anunciou em 26 de abril de 2014 uma vulnerabilidade que afetava todas as versões do Internet Explorer, e no dia 1 de maio uma atualização de segurança (patch) foi disponibilizada[9].

2.1.2 Definição de Lacuna de Segurança da Informação

No gerenciamento diário de suporte a sistemas, a prática de geração de imagens de sistemas operacionais é uma forma bastante útil para agilizar a instalação de computadores em ambientes com muitos computadores e padronizar a instalação de softwares. Por exemplo, Symantec Ghost [10] é um programa popularmente usado para capturar imagens do sistema operacional.

Uma vez gerada uma imagem, cada computador que receber essa imagem possuirá os mesmos aplicativos e as mesmas configurações da imagem aplicada. Sendo assim, o pessoal de suporte de primeiro nível tem condições de tratar cada problema de forma mais rápida, uma vez que as instalações estão padronizadas.

A Lacuna de Segurança da Informação é o lapso de tempo entre a disponibilização de uma atualização de segurança (patch) pela equipe de desenvolvimento do respectivo software e atualização da imagem e sua consequente aplicação nos computadores dos usuários finais do sistema[9].

Percebe-se que por questões de logística e uniformização de instalações, principalmente em ambiente onde há muitos computadores como em grandes empresas, universidades e grandes instituições, o tempo disponível para que um atacante para explore uma vulnerabilidade é o resultado da soma entre a Janela de Vulnerabilidade com a Lacuna de Segurança da Informação. Em organizações menores apenas a janela de vulnerabilidade é importante, uma vez que as atualizações de segurança podem ser aplicadas diretamente no computador do usuário final com mais agilidade, não obstante, também porque o número de computadores é menor.

2.2 IDS

Inicialmente é necessário definir detecção de intrusão, que é o processo de monitoramento de eventos que ocorrem em um sistema computacional ou em uma rede de computadores e análise em busca de sinais de possíveis incidentes, que podem ser violações ou ameaças iminentes de violação das políticas de segurança, do uso aceitável das políticas de segurança ou de práticas padrões de segurança. Incidentes possuem várias causas, tais como malware (e.g., worms, spyware), invasores vindos da internet ganhando acesso não-autorizado a sistemas, e usuários autorizados de sistemas que usam indevidamente seus privilégios ou tentam obter privilégios adicionais para os quais não estão autorizados. Embora muitos incidentes sejam de natureza maliciosa, muitos outros não são; por exemplo, uma pessoa pode digitar incorretamente o endereço de um computador e acidentalmente tentar se conectar a um sistema diferente, ao qual não possui autorização[11].

Um sistema de detecção de intrusão visa identificar atividades maliciosas que podem comprometer a integridade, confidencialidade e disponibilidade dos sistemas computacionais. Tentativas de violar a integridade, disponibilidade e confidencialidade das informações são consideradas como uma intrusão. Um Sistema de Detecção de Intrusão pode ser tanto um software quanto um hardware [12].

O que diferencia um IDS de um sensor simples instalado em um ambiente é que um IDS analisa dados e cria eventos a partir dessas análises. Sensores simples reportam tudo o que observam, enquanto um IDS é configurado para reportar apenas um fenômeno específico que foi inferido a partir dos dados analisados anteriormente [13].

Um IDS é normalmente instalado juntamente com outros mecanismos de prevenção em

segurança, tais como controle de acesso e autenticação, mais especificamente como uma segunda linha de defesa que protege os sistemas de informação [14].

Existem várias razões pelas quais um IDS é uma parte necessária de todo o sistema de defesa computacional. Primeiro muitos sistemas tradicionais e aplicações são desenvolvidos sem que a segurança seja uma prioridade. Em outros casos, sistemas e aplicações são desenvolvidos em um ambiente e podem se tornar vulnerável quando instalado em outro ambiente. Por exemplo, um sistema pode estar perfeitamente seguro quando isolado, mas pode se tornar vulnerável quando conectado à internet. Terceiro, por causa de limitações na segurança da informação, nas práticas de engenharia de software, sistemas computacionais e aplicações que podem ter falhas de projeto ou bugs e que podem ser usados pelos atacantes dentro da janela de vulnerabilidade ou da lacuna de segurança da informação. Assim alguns mecanismos de prevenção (e.g., firewalls) podem não ser tão efetivos como o esperado [14].

Além disso, a instalação de um IDS é devida a certas considerações, além da detecção de intrusão em si, entre elas estão:

- Se uma intrusão é detectada de forma rápida o suficiente, o intruso pode ser identificado e ejetado do sistema antes que qualquer dano seja feito ou qualquer dado seja comprometido. Mesmo se a detecção não seja tempestiva o suficiente para se conter o intruso, quanto mais cedo a intrusão for detectada, menor será o dano e mais rapidamente se poderá recuperar o ambiente.
- A detecção de intrusão serve também para se coletar informações sobre as técnicas de intrusão que podem ser usada para reforçar as medidas de prevenção a invasão[15]

Um bom IDS pode detectar uma tentativa de intrusão antes que ocorra uma interrupção de qualquer serviço. Assim os administradores podem acionar medidas para interromper o ataque e mitigar possíveis danos.

Adicionalmente a identificar incidentes, tentativas de invasão e dar suporte aos esforços de resposta ao incidente, as organizações encontraram outros usos para um IDS, incluindo os seguintes:

- Identificar problemas nas políticas de segurança. Um IDS pode prover algum grau de controle de qualidade para a implementação de políticas de segurança, tais como regras duplicadas de firewall e alertar tráfego que está ocorrendo, mas que deveria ter sido bloqueado pelas regras do firewall, e que não foram bloqueadas por algum problema na configuração do firewall.

- Documentar uma ameaça à organização. As informações contidas nos logs do IDS sobre ameaças que ele detectou, podem servir de subsídios para se entender a frequência e características dos ataques contra os recursos computacionais de uma organização. Esse entendimento pode ser útil para identificar as medidas apropriadas para proteger os recursos. Tal informação também pode ser usada para educar a gerência sobre as ameaças que a organização sofre.
- Inibir pessoas de violar políticas de segurança. Se indivíduos estão conscientes que suas ações estão sendo monitoradas por uma tecnologia de IDS que detecta violações das políticas de segurança, eles serão menos propensos a cometer atos de violação por causa do risco de detecção[11].

Devido à crescente dependência de sistemas de informação e do potencial impacto que intrusões podem causar nos sistemas de informação, o IDS tem se tornado uma adição necessária à infraestrutura de praticamente todas as organizações[11].

Diferentes técnicas de detecção de intrusão fornecem diferentes tipos de benefícios para uma grande variedade de ambientes, por isso, é essencial que se selecione e se implemente a solução mais adequada. E a chave para se selecionar o sistema de detecção correto é definir os requerimentos específicos ao ambiente onde será instalado o IDS para que se satisfaça da melhor maneira possível as necessidades de segurança da organização [16].

Em relação ao desempenho de um IDS, existem vários critérios para avaliá-lo. Entre eles estão:

- Acurácia: Dentre todas as classificações, quantas o modelo classificou corretamente. Existe a possibilidade que o sistema de detecção de intrusão possa errar na classificação de um evento. Assim, um IDS deve ser capaz de monitorar e analisar as atividades/tráfego de maneira que possa detectar as ameaças com alta acurácia. E os principais critérios para avaliar esse parâmetro são: True Positive rate (Taxa de positivo verdadeiro); False Positive Rate (Taxa de falso positivo).
- Tempestividade: Em caso de ataque, é requerido que o sistema possa detectar o referido ataque o mais rápido possível. Tempestividade é definida pela quantidade de tempo que o IDS leva para detectar qualquer ataque. Como a ação mais comum de um IDS após detectar uma ameaça é informar aos administradores, esse parâmetro se torna um importante fator para avaliar o desempenho do IDS.
- Capacidade de processamento: O IDS deve monitorar e analisar o tráfego de rede de

forma contínua. Um bom IDS deve ter a capacidade de manipular uma grande quantidade de tráfego tanto de entrada quanto de saída no perímetro que esteja monitorando e ainda assim analisar e detectar intrusões.

- Autodefesa: A capacidade de resistência de um sistema a ataques ao IDS. Como a tecnologia evolui dia-a-dia, inclusive as técnicas de ataque, um IDS deve ser capaz de lidar com ataques desferidos contra o seu funcionamento [8]. Caso o IDS seja um software embarcado em um equipamento proprietário, este deve ter mecanismos de defesa para autoproteção, como antivírus, firewall etc.

Um IDS pode ser ativo ou passivo. Quando ativo, pode realizar ações, como por exemplo, bloquear o atacante ou bloquear determinado endereço IP. Sendo ativo pode ser chamado de IPS - Intrusion Prevention System (Sistema de Prevenção a Intrusão). Quando passivo, apenas emite alertas para que o administrador do sistema esteja consciente de que a rede está sob ataque.

Um IPS reage automaticamente e tem a capacidade de, com rapidez, parar um ataque. Logo optar por um IPS ao invés de um IDS pode parecer, a priori, a decisão mais atraente. Na prática, investigando mais a fundo, se percebe que o problema dos falso-positivos pode tornar a decisão de se adotar um IPS não aconselhável. Antes de mais nada, é necessário medir o peso do impacto de falhar na defesa de um ataque real (falso negativo) contra acidentalmente derrubar um tráfego legítimo (falso positivo) para só então escolher entre um IDS e um IPS, o que torna o IDS na maioria dos casos mais adequado. E é preciso levar em consideração que normalmente se requer bastante esforço para serem configurados corretamente[9].

Além da classificação quanto a reação do IDS, é apresentado a seguir outras classificações de detecção de intrusão, segundo o critério de como se detecta a intrusão (baseado em anomalia ou baseado em assinatura) e também segundo o critério de onde o IDS irá atuar (baseado em host ou baseado em rede).

2.3 IDS baseado em anomalia

Uma técnica bastante utilizada em informática é a criação de patamares de normalidade (*baseline*). Uma vez determinado qual é o comportamento esperado para especificado contexto se pode aferir se há alguma anormalidade nas atividades correntes comparando essas com os patamares de normalidade armazenados.

Em um cenário onde exista muitos sistemas a serem monitorados, a prática da criação de patamares de normalidade é bastante útil, seja para segurança da informação, seja para avaliação de carga do sistema, da rede ou de dispositivos de armazenamento, ou para qualquer outro monitoramento necessário.

Um patamar de normalidade pode envolver várias políticas administrativas dentro do departamento de informática de uma instituição. A título de exemplo uma política de segurança pode ter, entre outras regras, as seguintes: *Protocolos* – somente os protocolos listados na atual política de segurança podem ser permitidos, todo os outros devem ser bloqueados; *Serviços* – somente os serviços listados na atual política de segurança podem estar ativos, todos os outros devem ser desabilitados; *Contas de usuários* - A conta do administrador deve ser renomeada. Essa informação deve ser tratada como segredo da companhia.

Levando-se em conta, que o simplório exemplo listado no parágrafo anterior, é apenas uma única política entre várias outras que podem existir dentro de uma companhia, e.g., política de armazenamento de dados, política de monitoramento. A tarefa de se criar patamares de normalidade pode escalar rapidamente para níveis de complexidade extremamente altos conforme leva em consideração as várias políticas existentes dentro de uma instituição [9]. Ainda assim, a criação de patamares de normalidade para a segurança da informação é uma ferramenta bastante útil, especialmente para a detecção de eventos relativos à detecção de intrusão.

O IDS baseado em anomalia parte da premissa de que ataques criam uma anomalia no comportamento do sistema, e essa anomalia pode ser identificada comparando-se com o patamar de normalidade criado anteriormente ao ataque. Portanto, um sistema de detecção de intrusão baseado em anomalia é capaz de detectar ataques nunca vistos antes.

Um operador começa o uso de um IDS baseado em anomalia medindo o comportamento normal do ambiente que deseja monitorar e essa medição se torna o patamar de normalidade. O IDS então monitora as atividades diárias comparando-as com o patamar de normalidade. Enquanto o comportamento do ambiente for similar ao patamar de normalidade, as atividades são normais. Quando as atividades mudam para limiares muito diferentes do patamar de normalidade as atividades são tidas como anormais[9].

Uma atividade anormal não significa que sempre é um evento de segurança. Um alerta pode ser ativado por um evento “falso-positivo”. Isso significa, que um alerta foi disparado por uma atividade não usual, mas que após uma análise mais profunda se verifica que era um

comportamento totalmente normal e explicável. Como por exemplo, a entrada de um novo sistema em um servidor pode gerar picos de atividades que não estão dentro do patamar de normalidade previsto para esse servidor. Em um caso assim, o IDS pode ser reconfigurado para ignorar tal alarme.

Entretanto, a principal atividade de um IDS é detectar ameaças de segurança. Por exemplo, caso um ataque seja disparado contra uma rede, poderá haver um aumento na atividade da rede. O IDS reconhecerá essa anormalidade e disparará um alarme correto. Um IDS baseado em anomalia não pode operar sem antes criar um patamar de normalidade. Se pode pensar em um IDS baseado em anomalia como um IDS baseado em anomalia porque está monitorando o comportamento de uma rede[9].

Comparando-se um IDS baseado em anomalia com um IDS baseado em assinatura aquele gera muito mais alarmes falsos que este. Uma vez que o IDS baseado em assinatura deduz o que é ataque a partir do registro de ataques que foram analisados em detalhes. Logo, a sensibilidade do IDS baseado em assinatura é muito mais alta

2.3.1 IDS baseado em assinatura

O IDS baseado em assinatura procura por pegadas no sistema, quando encontra determinado rastro identificável e único, que corresponde a uma entrada em seu banco de dados, gera um alarme. É o tipo de IDS mais encontrado em produção, mas que não consegue identificar novos ataques sem atualizar seu banco de dados de assinaturas.

Possui um design simples e bastante efetivo (para ataques já registrados em sua base), trabalha com alta sensibilidade para os ataques já conhecidos e tem a vantagem de não gerar muitos falsos alarmes porque os ataques estão descritos em detalhes. Porém ataques cibernéticos recém-criados podem contornar facilmente esse tipo de proteção, especialmente dentro da janela de vulnerabilidade e da lacuna de segurança da informação.

Entre as características que podem tornar um IDS baseado em assinatura adequado a determinado ambiente estão: examina o tráfego corrente, atividades, transações ou comportamentos que podem ser padrões de eventos específicos de ataques conhecidos; requer acesso a uma base de dados de assinatura de ataques e de alguma forma compara ativamente um evento atual com uma grande coleção de assinaturas.

Entre as características que podem tornar um IDS baseado em assinatura inadequado a determinado ambiente estão: se a definição da assinatura é muito específica, o IDS pode ignorar variações de um ataque conhecido; o banco de dados de assinaturas precisa ser constantemente atualizado; o IDS precisa ser capaz de comparar e encontrar atividades contra um grande coleção de assinaturas de ataques; o IDS pode impor uma notável carga de processamento no sistema em que determinados eventos podem ser identificados com várias assinaturas, seja parcialmente ou totalmente[17].

Exemplos de assinaturas podem ser:

- Uma tentativa de acesso via telnet com um nome de usuário "root", que é uma violação da política de segurança de uma organização;
- Um e-mail com o assunto “Fotos grátis!” e um nome de arquivo anexo "freepics.exe", que são características de uma forma conhecida de malware;
- Uma entrada no log do sistema operacional com um valor de código de status de 645, que indica que a auditoria do host foi desabilitado.

2.4 IDS baseado em redes (NIDS)

Monitora o tráfego de rede a partir de um segmento ou dispositivo e analisa as atividades de um protocolo de rede ou aplicação para identificar atividades suspeitas. É comumente instalado na fronteira entre redes, ou seja, em firewalls de borda (*border firewalls*), roteadores, servidores VPN, servidores de acesso remoto, e redes sem fio. NIDS são efetivos para se monitorar tanto o tráfego de dados que entra na rede quanto o que sai, assim como o fluxo de rede entre computadores ou entre segmentos locais de rede. NIDS também são tipicamente instalados em frente e atrás de um firewall e VPN gateways para medir a eficiência desses equipamentos de segurança e adicionar mais profundidade à segurança da rede [13]. O presente trabalho propõe um modelo de NIDS baseado em AM, assim sendo, é detalhado abaixo as formas mais comuns de implantação de um NIDS.

Adicionalmente a escolher em que parte da rede é apropriado para se instalar um NIDS, os administradores também precisam decidir onde os sensores (coletor de dados) deverão estar localizados. Sensores podem ser instalados de duas formas:

- Inline – É um coletor de dados que pode também vir a ser acoplado a um IPS. Um sensor inline é instalado de forma que o tráfego de rede deve passar por ele, muito como o tráfego de rede associado a um firewall. De fato, alguns sensores inline são dispositivos firewall/IDS híbridos, enquanto outros são simplesmente IDS. A motivação primária para se instalar um sensor IDS inline é habilitá-los a parar o tráfego de rede caso aconteça um ataque. Sensores inline são tipicamente colocados onde os firewalls e outros equipamentos de segurança são colocados – na divisão entre redes, tais como conexões com redes externas e nas bordas entre diferentes redes internas que podem ser segregadas. Sensores inline que não são dispositivos híbridos firewall/IDS são frequentemente instalados no lado mais seguro da divisão entre as redes assim têm menos tráfego para processar.
- Passivo. Um sensor passivo é instalado de tal forma que monitore uma cópia do atual tráfego de rede; mas nenhum tráfego passa pelo sensor. Ou seja, este tipo de sensor não tem a capacidade de parar um tráfego caso seja acoplado a um IPS. Sensores passivos são tipicamente instalados de forma que eles possam monitorar localizações chaves da rede, tais como divisões entre redes, e segmentos chaves de rede, tais como na sub-rede de zona desmilitarizada (DMZ). Sensores passivos podem monitorar o tráfego através de vários métodos, incluindo os seguintes:
    - *Spanning Port*. Muitos switches possuem uma *spanning port*, que é uma porta em que se pode "enxergar" todo o tráfego de rede que passa pelo switch. Conectando um sensor na spanning port se pode permitir que se monitore tanto o tráfego de entrada quanto de saída. Apesar de que esse método de monitoramento é relativamente fácil e barato, pode ser problemático. Se o switch estiver configurado ou reconfigurado de forma incorreta, a porta spanning pode não ser capaz de "enxergar" todo o tráfego de rede. Outro problema com portas spanning é que elas podem ser muito consumidoras de recursos; quando um switch está sobre uma grande carga, a porta spanning pode não ser capaz de "enxergar" todo o tráfego, ou a função spanning pode ser temporariamente desabilitada. Muitos switches possuem apenas uma porta spanning, e pode haver a necessidade de se ter múltiplas tecnologias ligadas, como ferramentas de monitoramento de tráfego de rede, ferramentas de análises forenses, e outros sensores IDS, monitorando o mesmo tráfego de rede.

- Network Tap. Uma network tap é uma conexão direta entre um sensor e um meio de comunicação físico, tal como uma fibra ótica (por exemplo). A tap provê o sensor com uma cópia de todo tráfego de rede que está ocorrendo no meio de comunicação. Instalando-se uma tap geralmente envolve algum período de desligamento da rede, e problemas com a tap podem causar um tempo adicional de desligamento da rede. Além disso, ao contrário das portas spanning, que geralmente já estão presentes em toda a organização, as taps de rede precisam ser adquiridos como complementos da rede.
- Balanceadores de carga IDS. Um balanceador de carga IDS é um dispositivo que agrega e direciona o tráfego de rede para sistemas de monitoramento, incluindo sensores IDS. Um balanceador de carga pode receber cópias do tráfego de rede de uma ou mais portas spanning ou taps de rede e agregar o tráfego de diferentes redes (por exemplo, remontar uma sessão que foi dividida entre duas redes). O balanceador de carga, então, distribui cópias do tráfego para um ou mais dispositivos de escuta, incluindo sensores IDS, com base em um conjunto de regras configuradas por um administrador. As regras informam ao balanceador de carga quais tipos de tráfego fornecer a cada dispositivo de escuta. As configurações comuns podem incluir o seguinte:
  - Enviar todo o tráfego para vários sensores IDS. Isso pode ser feito para fins de alta disponibilidade ou para que vários tipos de sensores IDS executem análises simultâneas da mesma atividade.
  - Dividir dinamicamente o tráfego entre vários sensores IDS com base no volume. Isso normalmente é feito para realizar o balanceamento de carga, de modo que nenhum sensor fique sobrecarregado com a quantidade de tráfego e a análise correspondente.
  - Dividir o tráfego entre vários sensores IDS com base em endereços IP, em protocolos ou em outras características. Isso pode ser feito para fins de balanceamento de carga, tal como ter um sensor IDS dedicado à atividade da Web e outro sensor IDS monitorando todas as outras atividades. A divisão do tráfego também pode ser feita para realizar uma análise mais detalhada de certos tipos de tráfego (por exemplo, atividade envolvendo os hosts mais importantes). A divisão do tráfego entre vários

> sensores IDS pode causar uma redução na precisão da detecção se eventos relacionados ou partes de um único evento forem vistos por diferentes sensores. Por exemplo, suponha que há dois sensores e que cada um veja etapas diferentes de um ataque; se a primeira etapa for considerada benigna por si só, mas as etapas seguintes forem consideradas maliciosas, o ataque pode não ser reconhecido [11].

Um Sistema de detecção de intrusão baseado em rede (NIDS) visa localizar atividades maliciosas em um ambiente de rede [5]. Para alcançar tal objetivo uma arquitetura comum de NIDS compreende quatro módulos. Primeiramente o módulo de aquisição de dados coleta os dados de uma rede, e.g. lê pacotes de dados de uma placa de interface de rede. Os dados coletados são avaliados pelo módulo de extração de características que extrai características comportamentais dos dados da rede para compor o vetor de características, também chamado de visão. O vetor de características extraídas é avaliado pelo módulo de decisão que estabelece o rótulo adequado ao evento, e.g. aplica o modelo de AM para classificar o evento como normal ou ataque para que o módulo de alerta reporte o evento de maneira apropriada.

Em geral, em ambientes de produção, técnicas baseadas em assinaturas são empregadas pelos NIDS [6]. Tais abordagens buscam por ataques conhecidos entre os dados coletados, mas são incapazes de detectar novos ataques. Consequentemente devido ao crescente número de ataques, um significante esforço de pesquisa é realizado para se desenvolver novas técnicas de detecção em NIDS, tipicamente por meio de abordagens baseadas em comportamento.

Pontos de transição entre redes são, a priori, os melhores locais para se monitorar os perímetros de redes envolvidos. Logo, a instalação de um NIDS ou de seus sensores deve ser priorizada para os seguintes segmentos de rede:

- DMZ/Perímetro
- Rede Corporativa Central
- Rede Wireless
- Rede Virtualizada
- Outros segmentos críticos de rede

Como um NIDS somente escutará o tráfego passante, logo não consumirá muita banda de rede[18]. A tabela 2 exibe exemplos de NIDS.

Produto	Características	Site
Snort	Open Source Intrusion Prevention System (IPS)	https://www.snort.org/
Kismet	*Framework* WIDS (wireless intrusion detection), para detecção em redes sem fio.. Podendo também trabalhar como sniffer, detector e ferramenta de *wardriving*.	https://www.kismetwireless.net/
Cisco Secure IPS	Secure IPS recebe novas politicas de segurança e assinaturas de ameaçaas a cada 2 horas, ligado ao Cisco Talos que é vendida maior rede de detecção de ameças do mundo.	https://www.cisco.com/c/en/us/products/security/ngips/index.html#~features
Bro Intrusion Detection System	Analisador passivo de tráfego de rede *open source*. (Em 2018 foi renomeado para zeek)	https://zeek.org/
Enterasys Dragon IDS	Network Intrusion Detection System (NIDS)	https://www.extremenetworks.com/
Firestorm NIDS	NIDS focado em desempenho mas com poucas funcionalidade implementadas	https://www.scaramanga.co.uk/firestorm/
GFI LANguard	Contém um banco de dados com mais 60.000 assinaturas de vulnerabilidades e um extenso catalogo de módulos	https://www.gfi.com/products-and-solutions/network-security-solutions/gfi-languard
SecureMetrics	Oferece soluções modulares, entre elas, o agendamento de busca de vulnerabilidades na rede.	https://www.securitymetrics.com/vulnerability-scan
Shoki	Open Source, voltado para a simplicidade e modularidade	http://shoki.sourceforge.net/

SNIPS	Propõe-se a preencher a lacuna semântica entre o que o Snort captura (padrões de pacote) e o que o usuário realmente deseja saber (atividades maliciosas)	https://people.cs.ksu.edu/~xou/argus/software/snips/
McAfee IntruShield Network IPS Solution	IPS baseado em rede que é usado na prevenção de ataques DoS (Denial of Service) de *zero day*, *spyware*, *malware*, *botnets* e ameaças *VoIP*. Agora é chamado de McAfee Network Security Platform	https://www.mcafee.com/enterprise/en-us/products/network-security-platform.html
Suricata	IDS, IPS, network security monitoring (NSM), *Open Source*	https://suricata.io/

Tabela 1: Exemplos NIDS

2.4.1 IDS baseado em hosts (HIDS)

Monitora características de um único computador (host) e eventos que ocorram naquele computador que possam ser atividades suspeitas, e.g. logs, APIs, sistema de logs, processos rodando, atividade de aplicações, acesso a arquivos, modificações de arquivos, mudança na configuração do sistema operacional ou de aplicações [19]. Frequentemente é instalado apenas em computadores que exijam um nível de segurança mais elevado, como servidores de acesso público ou servidores com informações sensíveis. E assim como os NIDS, que é um IDS baseado em rede, o papel mais comum de um HIDS é o de detecção passiva, apenas gerando alertas [19].

HIDSs atuam de forma semelhante aos softwares antivírus [19] [4], entretanto não são substitutos por esses, e com capacidade estendida para o nível de segurança que provêm. HIDS são adequados para combater ameaças contra computadores (hosts) porque possuem capacidade de monitorar e responder a ações de usuários e a tentativas de acesso à arquivos em servidores. Uma vez que é comum que ameaças se originem dentro das organizações, vindos de diferentes fontes como empregados desapontados ou espiões corporativos[19].

Caso seja necessário monitorar vários computadores, se precisa de uma instalação para cada computador monitorado. A alta manutenção exigida por um HIDS o torna mais custoso em termos de tempo e mão-de-obra. A tabela 1 exibe exemplos de HIDS.

Produto	**Características**	**Site**
OSSEC (Open Source Security)	Monitoramento de log	https://www.ossec.net/
Tripwire	Ferramenta de integridade de dados (arquivos, diretórios, links, etc)	https://github.com/Tripwire/tripwire-open-source
HP-UX HIDS	Monitoramento de arquivos de log e monitoramento de arquivos críticos e NIDS	https://myenterpriselicense.hpe.com/cwp-ui/free-software/HPUX-HIDS
CACIC	Controle de softwares adquiridos versus inventariados	https://softwarepublico.gov.br/social/cacic
Nagios	Monitoramento de Log, monitoramento de aplicações, e NIDS	https://www.nagios.com/
Radmind	Ferramenta de integridade de dados (arquivos, diretórios, links, etc)	http://www.radmind.org/
Advanced Intrusion Detection Environment (AIDE)	Verificador de integridade de arquivos e diretórios	https://aide.github.io/
Another File Integrity Checker (AFICK)	Ferramenta de integridade de dados (arquivos, diretórios, links, etc)	http://afick.sourceforge.net/

Enterasys Dragon Host Sensor	Host Intrusion Detection System (HIDS)	https://www.extremenetworks.com/
GrSecurity – PaX	Extensivo aprimoramento de segurança para o kernel Linux	https://grsecurity.net/
LIDS	Patch para o kernel do Linux	http://lids.jp/develop/
Logsurfer	Monitor de qualquer arquivo de log baseado em texto	http://logsurfer.sourceforge.net/
Samhain	Verificação de integridade de arquivos e monitoramento / análise de arquivos de log, bem como detecção de rootkit, monitoramento de portas, detecção de executáveis SUID desonestos e processos ocultos.	https://www.la-samhna.de/samhain/

Tabela 2: Exemplos de HIDS

2.5 Aprendizagem de Máquina para IDS

Uma vez que o panorama corrente de ameaças tecnológicas é muito dinâmico e muda rapidamente, é requerido que os sistemas de detecção possam mudar rapidamente para se ajustarem a novos ataques. Os sistemas de detecção tradicionais que se baseiam em ajustes feitos manualmente, em limiares fixos, ou patamares de normalidades (*baseline*) fixos estão propensos a disparar excessivos alarmes falso-positivos [18]. Visto que os limiares precisam ser ajustados manualmente e os patamares de normalidades precisam ser atualizados caso novos serviços sejam adicionados.

Nos últimos anos, a maioria das abordagens de detecção baseadas em comportamento para NIDS se fundamentam em técnicas de aprendizagem de máquina (AM), tipicamente por meio de reconhecimento de padrões [7]. Nesses casos, um modelo de AM de comportamento

é construído utilizando-se de um conjunto de dados de treinamento, que compreende o tráfego de rede esperado por um determinado período. O procedimento de treinamento é frequentemente realizado em um contexto supervisionado em que a rotulagem dos eventos deve ser previamente realizada, para que sejam levados em consideração na construção dos modelos de AM. No geral, esforços em pesquisas em AM para NIDS são conduzidas como em qualquer outro campo em que AM foi aplicado com sucesso. Porém, contextos de rede de computadores apresentam uma miríade de desafios que não são evidenciados em outros campos [6]. O comportamento do tráfego de rede não é estacionário, ao contrário, muda constantemente, ou devido a novos tipos de serviços ou devido a novos ataques sendo descobertos [5]. Essas mudanças no comportamento do tráfego de rede, com o tempo, tornam os modelos de AM obsoletos, o que incorre em aumento da taxa de erros. Soma-se a isso, que as técnicas baseadas em AM são frequentemente concebidas para encontrar similaridades nos dados de entrada, ao invés de encontrar comportamentos não vistos anteriormente. Como resultado, se a ocorrência de um novo tráfego de rede, sendo ou um novo ataque ou um novo de serviço, também pode incorrer em uma significante mudança no comportamento de tráfego de rede, ocasionando que o modelo de AM construído pode diminuir significativamente sua acurácia. Adicionalmente, conduzir um procedimento de atualização de modelo em ambientes de redes não é uma tarefa fácil de ser executada, dado que o tráfego de rede deve ser rotulado, tipicamente com assistência humana, e um processo de retreinamento de modelo que é computacionalmente muito custoso, deve ser efetuado. Como resultado, apesar de vários trabalhos reportarem altas acurácias em modelos de AM, tais técnicas continuam em sua maioria como tópicos de pesquisa, dificilmente sendo implementados em ambientes de produção [6] [7].

2.5.1 Aprendizagem de Máquina Semi-supervisionada

O principal objetivo da aprendizagem de máquina semi-supervisionada é aproveitar os dados não rotulados para a construção de melhores procedimentos de aprendizagem [20]. Os métodos de classificação semi-supervisionada são particularmente relevantes para cenários onde os dados rotulados são escassos. Nesses casos, pode ser difícil construir um classificador supervisionado alta acurácia ao longo do tempo[20].

O custo mais persistente na execução do aprendizagem de máquina é a criação de rótulos para fins de treinamento [21]. E no espaço entre aprendizagem supervisionada (com a presença

de rótulos) e aprendizagem não-supervisionada (sem a presença de rótulos), há uma família de técnicas de AM que podem atuar a partir de poucos rótulos [21]. Essa família de técnicas é denominada de AM semi-supervisionada.

Aprendizagem de máquina semi-supervisionada é um grupo de técnicas que se mostrou especialmente eficaz quando não há disponibilidade de muitos rótulos para a aprendizagem inicial, pois durante o treinamento inicial são feitas várias iterações de classificação sobre os rótulos e se vai aperfeiçoando a rotulagem faltante após várias iterações do algoritmo de AM semi-supervisionada.

Nas últimas duas décadas, uma ampla variedade de classificação de algoritmos semi-supervisionados foram propostas [20]. Não havendo um consenso claro e definitivo na classificação de algoritmos semi-supervisionados. Contudo, os principais algoritmos de AM semi-supervisionada são os seguintes:

- Self-training – No self training um único classificador é treinado iterativamente sobre um pequeno, mas crescente número de rótulos, a partir de um número inicial reduzido de eventos rotulados.

  Normalmente os passos para a realização de uma iteração do self training envolvem:

  - Treinar o classificador com rótulos aprendidos da interação anterior;
  - Aplicar o classificador treinado sobre dados não rotulados ainda;
  - Ordenar os rótulos preditos pela sua probabilidade de acerto do classificador, colocando os com maiores probabilidades juntamente com os rótulos preditos na tabela de rotulados, o que implica retira-los da tabela dos não-rotulados.
- *Co-training* – Similarmente ao self training, o co-training envolve várias iterações sobre os dados não rotulados, com a diferença que se utiliza de dois classificadores complementares aplicados simultaneamente [22]. Será descrito mais sobre esse método no item 2.3.2.
- *Generative models* - *Generative models* são treinados para aprender a distribuição subjacente dos dados. Depois de treinar um *generative model*, pode-se usá-lo para produzir novos exemplos de uma classe [23].

2.6.1 Co-training

Uma das técnicas pertencentes a AM semi-supervisionada é o co-learning. Em 1998, A. Blum e T. Mitchell [24] propuseram essa técnica ao qual a partir de um pequeno número de eventos rotulados em um conjunto de dados, são rotulados novos eventos por dois classificadores diferentes, onde um classificador aprende a partir dos rótulos gerados pelo outro classificador, e vice-versa (figura 2). Após várias iterações, são rotulados automaticamente um número muito maior do que o número de rótulos disponíveis inicialmente, e com alta acurácia.

Para se levar a cabo essa tarefa é necessário que os subconjuntos de dados, chamados de visões, se baseiem em dois princípios: Consenso e Complementariedade. Consenso assume que todas as visões devem maximizar o acordo entre as distintas visões e a complementariedade assume que cada visão deve conter algum conhecimento que as outras visões não possuem.

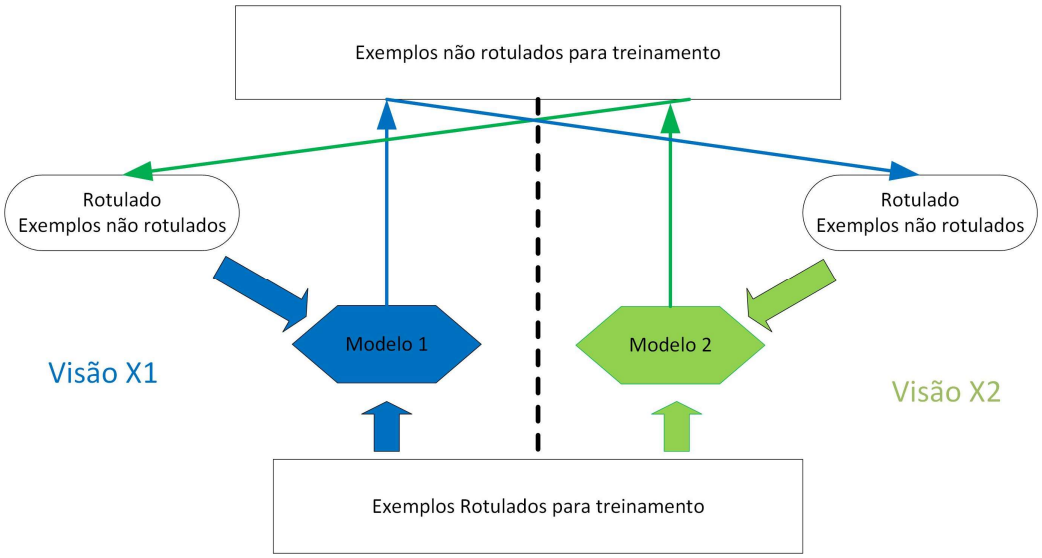


Figura 1: Ilustra o processo do co-training onde cada classificador provê instâncias de rótulos para o outro classificador – adaptado de [25]

c

Capítulo 3

Trabalhos Relacionados

A seguir, são listados trabalhos que se assemelham ou que abordam as técnicas utilizadas na arquitetura proposta. São utilizados para a comparação os seguintes parâmetros: Diversidade de dados; Atualização do mecanismo; Robustez; Tráfego de rede estacionário; Origem do tráfego; Intervenção humana.

3.1 Aprendizagem de Máquina para IDS

Na literatura, os autores frequentemente visam alcançar altas acurácias em um conjunto de dados único e estacionário (Mirza et al [26], Singh et al.[27]). Por exemplo, Mirza et al [26] propõe um *"ensemble"* (combinações de modelos fracos para gerar um modelo forte) de classificadores para melhorar a acurácia de detecção. Os autores assinalam a cada classificador um peso de acordo com as acurácias medidas, quanto mais acurado o modelo mais influência ele tem na classificação. Nessa abordagem foi implementado três tipos diferentes de classificadores, i.e, *neural networks, decision tree e logistic regression*. E foi demonstrado um aumento da acurácia.

As dimensões do conjunto de dados foram reduzidas utilizando-se de PCA (*Principal Components Analisys*). Foi observado que cada classificador trabalha melhor para uma classe em particular, onde decision tree favorecia mais a classe normal. Por outro lado, *neural*

networks e logistic regression favoreciam ambas as classes. Então a acurácia geral foi melhorada pela introdução de *ensemble learning*, e pela designação de certos pesos para cada um dos classificadores, então foi decidido se a amostra era anômala ou não. Normalmente assume-se que a proporção de anomalias para tráfego normal seja bastante baixa. Mas o conjunto de dados usado nesse trabalho a taxa de anomalias era bastante alta, i.e., 80%. Mesmo com esse problema, ainda assim o uso de ensemble learning obteve uma acurácia geral de 96,14%. Vale ressaltar que o conjunto de dados utilizado não tinha uma proporção realística de ataque versus tráfego normal, ainda assim o estudo foi levado adiante.

Singh et al. [27] propõe uma abordagem híbrida em dois passos, no primeiro passo é utilizado um agrupamento de dados via agrupamento K-means e no segundo passo os dados são classificados utilizando-se de um classificador Random Forest para detecção de intrusão. Essa mesma abordagem foi repetida utilizando-se primeiramente Gaussian Mixture clustering ao invés de K-means clustering e permanecendo na classificação o Random Forest.

Os autores demonstraram que dessa maneira eles conseguiram manter altas taxas de detecção enquanto também diminuíram os falsos alarmes. Como resultado obtiveram com K-means 99,7 % de taxa de detecção com apenas 0,09% de taxa de falso alarmes, e com Guassian obtiveram 99,6% de taxa de detecção com apenas 0,04% de taxa de falsos alarmes. Entretanto, apesar das altas taxas reportadas esses trabalhos utilizou um conjunto de dados obsoleto, o KDD99 [28], mas por que não descreve um tráfego de dados de produção e que não considera a variação natural que ocorre ao longo do tempo.

3.2 Aprendizagem de Máquina Semi-supervisionada para IDS

Em outro tipo de abordagem, alguns autores aplicam técnicas de multivisão para detecção de intrusão utilizando-se de diversidade de dados, por meio de visões distintas e complementares de 3 (três) conjuntos de dados, i.e., NSL-KDD, UNSW-NB15 e CICIDS 2017. He et al. [29] propuseram uma abordagem multimodal aplicando a técnica de multivisão para detecção de intrusão. Curiosamente para os conjuntos de dados NSL-KDD e UNSW-NB15 foram criadas 3 visões que descrevem para cada visão: dados da conexão; dados do “payload” dos pacotes; e finalmente indicadores de conexões. E para o conjunto de dados CICIDS 2017 foram criadas apenas duas visões que descrevem para cada visão: informações de pacotes; indicadores de tráfego.

Os autores aumentaram significativamente a acurácia da detecção em vários conjuntos de dados, entre obsoletos (NSL-KDD), com tráfego de ataques forjados (UNSW-NB15) e tráfego normal forjado (CICIDS 2017).

Em outra abordagem Li et al. [30] propuseram uma abordagem multivisão para detecção de spam em ambientes com recursos restritos. Os autores se utilizaram de aprendizagem de máquina semi-supervisionado para rotulagem de eventos. O conjunto de dados foi separado em duas visões distintas, que receberam os nomes de IFS (Internal Feature Set) e de EFS (External Feature Set). Sendo o IFS (Internal Feature Set) composto por campos como tamanho do assunto, tamanho da mensagem, quantidade de anexos, tamanho dos anexos, quantidade de palavras no assunto, quantidade de palavras no corpo do email, e quantidade de imagens incoporadas no email. E sendo o EFS (External Feature Set) composto por campos como quantidade de destinatários, quantidade de respostas, nível de importância, frequência de envio de e-mails, frequência de recebimento de e-mails, e tamanho dos nomes dos remetentes.

Apesar dos autores considerarem cenários mais realistas o conjunto de dados continha apenas 7.133 e-mails sendo apenas 2.300 que estavam sem rótulos de spam ou normal. Que comparativamente com ambiente de NIDS além de serem poucas instâncias traz uma percentagem de rotulagem muito alta em relação ao total de instâncias do conjunto de dados. Ainda assim, os resultados obtidos pela classificação multivisão foram os mais acurados.

3.3 Co-Learning para IDS

Utilizando-se de Multivisão para detecção e identificação de Malwares, Velayati e Seyed [31] separaram características estáticas de características dinâmicas como visões distintas. Na visão de características estáticas incluiam-se Bytecodes, Opcodes e características de formato. Na visão de características dinâmicas em que foi necessário executar o arquivo para se capturar comportamentos, foram incluídas características como chamadas ao sistema (system calls) e acesso a portas. Tal abordagem apresentou acurácias sempre acima de 90%, porém é adequada apenas para sistemas de detecção de intrusão baseadas em host (HIDS), uma vez que trabalha apenas com partes de arquivos executáveis como visões (Bytecodes, Opcodes, e características de formato de arquivo).

3.4 Mudança de comportamento em IDS

Mesclando técnicas de detecção de anomalias no comportamento do usuário e detecção de anomalias em rede de computadores de um campus universitário, Zhao et al. [32] propuseram um modelo de detecção de anomalias baseado em padrões frequentes de comportamento de usuário e de rede ao longo do tempo, alcançando bons resultados. O modelo extrai padrões frequentes e aproximados do tráfego de rede para cada usuário e usa o fator de redução de tempo para discriminar entre os pesos do tráfego de rede atual e histórico tráfego de rede. Para testar foram utilizados dois conjuntos de dados, KDD99 e BUCT, sendo o último a coleta de tráfego de rede de 274 usuários da biblioteca da Beijing University of Chemical Technology, por apenas 3 dias e com a injeção de ataques forjados por uma ferramenta denominada Scapy [33].

A intenção do trabalho é minerar os padrões de comportamento do usuário e com detecção de anomalias de rede determinar se o comportamento do usuário é suspeito, como por exemplo se a conta do usuário foi raptada por um outro usuário. Predizendo assim possíveis comportamentos suspeitos dentro de uma universidade. Como o comportamento do usuário é mutável ao longo do tempo, é previsto um mecanismo que atenua o efeito de dados históricos na criação de modelos de reflitam o comportamento do usuário.

E para as técnicas de detecção de anomalias no comportamento do usuário foram utilizados logs de auditoria, o que torna a solução do referido trabalho um híbrido entre HIDS e NIDS. Porém tal abordagem não é possível em um ambiente de link de internet como o contexto do presente trabalho.

F. Breve and L. Zhao [34] descreve a aplicação de uma técnica de aprendizagem semi-supervisionado, abordagem de competição e cooperação de partículas, construída para atuar em cenários de *concept-drift*, porém o pressuposto principal dessa técnica é que as mudanças sejam graduais e incrementais, e que os dados sejam recebidos em pequenas porções. Essas duas premissas não podem ser garantidas num contexto de link de *back-bone* de internet, como é o objeto de estudo presente. Além disso o referido trabalho foi testado contra o conjunto de dados KDD Cup 1999.

Em A. Rahman and B. Verma [35] é proposta uma abordagem onde se começa com o processo de agrupamento (K-Means) que particiona um conjunto de dados em segmentos que contêm pontos de dados altamente correlacionados, que é tido como uma camada.

A decisão de classificação de um padrão de teste é alcançada encontrando a decisão do classificador de base correspondente em cada camada e fundindo suas decisões usando a votação por maioria.

Chegou-se às seguintes conclusões:

1) A precisão da classificação aumentou com o aumento do número de camadas. O impacto do número de camadas na precisão da classificação é substancial. 2) O impacto do número de clusters também é notável, pois a melhor precisão de classificação é alcançada quando os conjuntos de dados são segmentados em dois ou mais clusters em geral. 3) A abordagem proposta tem desempenho 4,27%, 1,08% e 1,73% melhor do que *clustered ensemble, bagging, e boosting*, respectivamente.

O. Y. Al-Jarrah [36] adaptaram essa abordagem para a detecção de intrusão em IDS e IPS. E compararam essa abordagem com o Tri-Training obtendo uma acurácia melhores com uma quantidade de rótulos iniciais menores.

A proposta foi testada contra os conjuntos de dados Kyoto 2006+ e NSL e apresentaram uma acurácia acima de 99%. Porém, apesar de o conjunto de dados Kyoto apresentar 3 anos de tráfego entre a data de 2006 até 2009 tornando obsoleto por ter aproximadamente 15 anos de sua criação.

3.5 Discussão

A grande maioria dos trabalhos relacionados visam a garantir uma acurácia melhor que a alcançada até o momento da confecção do referido trabalho. Apesar de ser um esforço louvável uma vez que mesmo uma porcentagem pequena de falsos positivos possa representar uma quantidade inadmissível de alarmes quando se trabalha com links de internet de alto tráfego como é comum atualmente em empresa, ainda assim, não se leva em consideração que conjunto de dados que não são realísticos podem levar a uma falsa sensação de segurança. Uma vez que os algoritmos avaliados por conjuntos de dados com tráfegos forjados (seja tráfego normal forjado, seja tráfego de ataque forjado) ou obsoletos podem criar vieses de dedução que não são aparentes numa primeira avaliação.

Um segundo ponto a se considerar é que as atualizações dos modelos passam desapercebidas pelas propostas, uma vez que normalmente o objetivo principal é demonstrar uma alta acurácia, tarefa que é realmente alcançada pela escolha de um bom algoritmo mas que também é influenciada pelos dados utilizados para a avaliação. Mas em períodos que discorram durante anos pode-se levar o modelo a comportamentos que não são avaliados, e essa falta de avaliações é uma prática corrente. Logo a confiabilidade dos modelos não é avaliada já que eles não são expostos a dados de longo prazo e tampouco é previsto uma forma de atualização dos modelos, apenas se subentende que o operador do sistema irá realiza-las.

Como abordado na fundamentação, um NIDS normalmente atua como uma segunda linha de defesa, atuando juntamente com outros mecanismos de defesa. Assim, apesar de a acurácia ser um ponto importante, a autonomia e a resiliência precisam também serem levadas em consideração para a proposta de uma arquitetura de NIDS para que possa ser efetiva em produção.

Outra prática comum evidenciada nos trabalhos relacionados é que normalmente são avaliados vários algoritmos, mas apenas a título de comparação e visando evidenciar o quanto o algoritmo proposto é acurado. Não se discute que com o passar do tempo mesmo um dos outros algoritmos listados nos referidos trabalhos possam passar a ser o algoritmo mais acurado dado que a natureza mutável da internet e dos novos ataques perpetrados. Em nenhum trabalho se vislumbrou a possibilidade de se levar em consideração o uso de vários algoritmos e executar uma concorrência entre eles para que se elevar o nível das classificações.

A acurácia geral da solução ao longo do tempo não é avaliada, seja pela falta de subsídios dos conjuntos de dados empregados, seja porque apesar da atualização dos modelos serem apenas subentendidas, mas não é levado a cabo no comportamento geral da solução proposta após várias atualizações. Assim, ganhos marginais obtidos pela escolha de determinado algoritmo pode ser perder com o passar das atualizações, uma vez que a internet e as tecnologias de rede não são um ambiente imutável ao longo do tempo.

Ainda que se espere que uma solução tenha um nível de detecção ótimo, não se pode desprezar que o problema é bem maior, detectar intrusões a longo prazo sem exigir a constante atenção de um escasso e caro especialista torna o desafio de se detectar intrusões mais complexo.

Trabalho	Características do trabalho					
	Diversidade	*Atualização do mecanismo*	*Robustez a longo prazo*	*Tráfego de rede estacionário/dinâmico*	*Origem do tráfego*	*Intervenção humana*
Mirza et al [26]	Sim, de modelos	Não é tratado	Não é tratado	Tráfego estacionário	Ambiente Simulado	Sim, no retreino
Singh et al. [27]	Sim, de modelos	Não é tratado	Não é tratado	Tráfego estacionário	Ambiente simulado	Sim, no retreino
He et al [29]	Sim, de dados	Não é tratado	Não é tratado	diversos	Vários ambientes simulados	Sim, no retreino
Li et al [30]	Sim, de dados	Não é tratado	Não é tratado	Não se aplica	Emails	Sim, no retreino
Velayati e Seyed [31]	Sim, de dados	Não é tratado	Não é tratado	Não se aplica	Malwares	Sim, no retreino
Zhao et al [32]	Sim, de dados	Sim, assume disponibilidade de especialista	Não é tratado	Apenas 3 dias	Rede interna de uma universidade	Sim, no retreino
Chiche 2021[37]	Sim, de modelos	Sim	Não é tratado	Tráfego estacionário	Ambiente Simulado	Não
Krishnaveni 2021 [38]	Sim, de modelos	Sim	1 dos 3 conjuntos de dados	diversos	Diversos	Sim, no retreino
F. Breve and L. Zhao[34]	Não	Não	Sim, mas não testado	Tráfego estacionário	Ambiente Simulado	Não
O. Y. Al-Jarrah[36]	Sim, de classificadores e de dados (layer)	Não é tratado	Sim, apesar dos dados antigos	Dinâmico	HoneyPots	Não é tratado
Abordagem proposta	**Sim, de dados**	**Sim, de modo autônomo, sem necessidade de intervenção humana**	**Sim, através da avaliação da confiabilidade da classificação**	**Tráfego de 2 anos em que tanto o tráfego normal quanto ataque são reais**	**Roteadores da internet**	**Sim, apenas durante o treinamento inicial**

Capítulo 4

Proposta

Nesta seção é apresentado um novo modelo de detecção de intrusão multivisão para abordar o comportamento em evolução do tráfego de rede, composto por duas etapas principais (Classificação e Atualização do Modelo de Modo Autônomo, conforme mostrado na Figura 3) e que visa garantir, assim, a acurácia do sistema a longo prazo, e realizar atualizações de modelo autônomas, sem assistência humana.

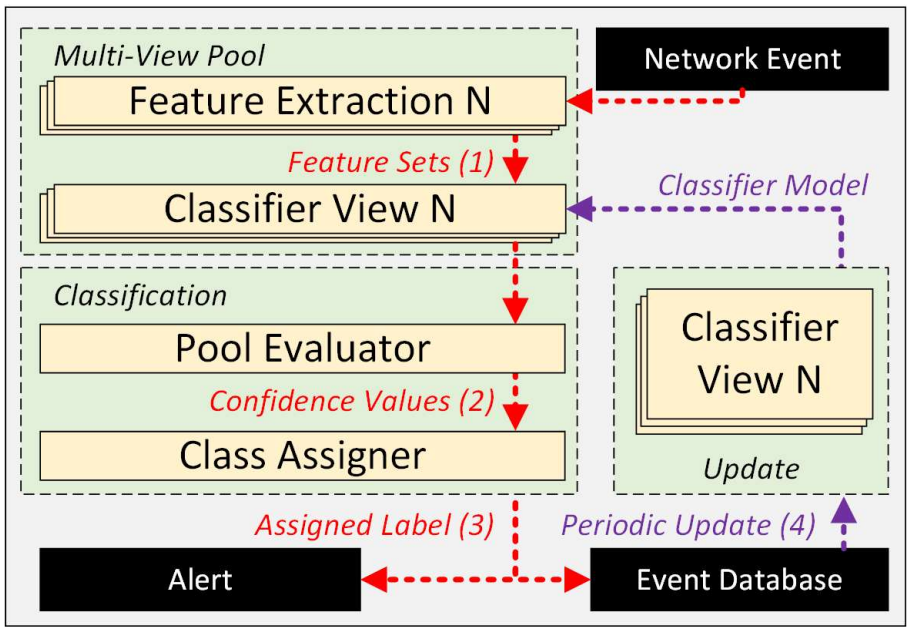


Figura 2: Estrutura do modelo multivisão proposto

4.1 Visão Geral

A proposta considera um esquema de classificação multivisão, onde um *pool* de modelos de AM é usado, cada modelo é construído através de um conjunto distinto de características, a saber, uma visão. O procedimento de classificação começa com um evento de rede para classificação, por exemplo, um conjunto de pacotes de rede relacionados ao serviço ocorrido dentro de um intervalo de tempo. O comportamento dos eventos de rede é então extraído para um conjunto de características pelo módulo de extração, em que cada módulo extrai um conjunto de características distinto. Os conjuntos de características processados são encaminhados para classificação, onde cada classificador, com sua própria visão, produz um valor de confiança de classificação. Os valores de confiança da classificação são usados como medida de correção da classificação pelo avaliador do pool, cujo objetivo é aceitar apenas classificações altamente confiáveis como uma tentativa de manter sua confiabilidade ao longo do tempo, mesmo com modelos desatualizados. Portanto, como saída de classificadores distintos, são aceitos apenas classificações altamente confiáveis, cada um com uma visão de evento exclusiva, e são usadas para o processo de rotulagem de eventos, descartando possíveis classificações desatualizadas e erros mais prováveis.

Como uma tentativa de fornecer modelos de AM atualizados, nossa proposta realiza atualizações periódicas do modelo autônomo de acordo com o rótulo de evento atribuído, conforme obtido pelas visões mais confiáveis. Portanto, uma visão que está atualmente produzindo classificações de baixa confiança pode ser atualizada de forma confiável por outros visões complementares, que geram valores de confiança mais altos e classificações mais provavelmente precisas, atualizando, assim, os modelos de AM de maneira autônoma e confiável ao longo do tempo.

Neste trabalho foram realizadas atualizações a cada 6 meses, como prova de conceito, mas esse é um parâmetro a ser definido no momento da implantação do NIDS levando-se em consideração a capacidade de processamento de dados do hardware disponível, o ambiente de onde o NIDS irá atuar e suas necessidades de segurança, e o tamanho do tráfego passante.

Também foram usadas 3 (três) visões como prova de conceito, mas esse número pode ser alterado conforme as necessidades do usuário e disponibilidade de dados, além dos motivos citados anteriormente. Em suma o modelo proposto permite uma certa personalização conforme os requisitos e disponibilidade de recursos do usuário.

A seguir é descritos os dois componentes do modelo, sendo eles a classificação multivisão confiável e atualização autônoma do modelo.

4.2 Classificação Multivisão Confiável

Uma abordagem multivisão se baseia em dois princípios básicos (i) concordância que assume que todas as visões devem maximizar a concordância em visões distintas, e (ii) complementariedade que assume que cada visão deve conter algum conhecimento que outra visão não possui. A proposta explora tais características como um esforço para melhorar a confiança geral do sistema, mesmo exposta a mudanças no tráfego de rede. A principal suposição é que a visões complementares podem melhorar uma à outra ao longo do tempo, , ou pelo menos mitigar a degradação dos modelos, durante ambos os processos de classificação e atualização autônoma do modelo.

A abordagem multivisão confiável é implementada de duas formas. Primeira, o comportamento do evento de rede a ser classificado é extraído pelo extrator de características (Feature Extraction N) que resulta em um conjunto de características denominada visão. O conjunto de características extraídas é classificado pelo algoritmo Random Forest com 100 árvores de decisão, que retorna um grau de confiança. O classificador Random Forest retorna o grau de confiança de acordo com a proporção de classificadores de base (“base-learners”) que classificaram a instância com o respectivo rótulo. Os graus de confiança são usados pelo módulo “pool evaluator” para identificar classificações com alto grau de confiança que devem ser usados para o processo de rotulagem. Finalmente, a módulo “Class assigner” assinala o rótulo do evento a partir do mais alto valor de grau de confiança.

O pseudocódigo a seguir detalha o procedimento de classificação da abordagem proposta.

Algorithm 1 Classificação Multivisão Confiável

Require:
Instance $inst = \{x_1, ..., x_N\}$
Classifiers $pool = \{c_1, ..., c_N\}$
Thresholds $threshold = N$ ▷ Definição do Limiar de Confiança

procedure CLASSIFICATION($inst$, $pool$, $threshold$)
  for each $classifier, t \in \{pool\}$ **do**
    $class, conf \leftarrow classify(classifier, inst)$
    if $class = normal$ **and** $conf \geq threshold$ **then**
      $vote(inst, normal)$ ▷ Aceita a classificação
    else if $class = attack$ **and** $conf \geq threshold$ **then**
      $vote(inst, attack)$ ▷ Aceita a classificação
    else
      $reject(inst)$ ▷ Rejeita a classificação
    end if
  end for
  if $getNumberOfVotes(inst) = 0$ **then**
    $reject(inst)$ ▷ Rejeita por não ter sido eleito
  else
    $alert(getMajorityVote(inst))$ ▷ Aceita a classe mais votada
  end if
end procedure

Algoritmo 1: Classificação Multivisão Confiável

As visões escolhidas descrevem características diferentes, não podendo a mesma característica estar em visões diferentes e ainda que tenham pouca ou nenhuma correlação entre si. Isso garante, a princípio, a complementariedade. E como cada visão sempre descreve o mesmo evento é garantida a concordância.

Em uma implementação a quantidade de visões irá impactar no desempenho da solução, por isso, é necessário a avaliação de desempenho de acordo com o hardware a ser instalado e quantidade de tráfego naquele ponto da rede.

Este modelo permite o registro de quais visões são escolhidas para o rótulo final, permitindo que se avalie o histórico de desempenho de cada visão. É possível que numa implantação em produção que uma visão tenha seu histórico analisado seja aposentada, e uma nova visão pode ser colocada em produção desde que se tenha dados suficientes para se treinar o modelo inicial da nova visão. Apesar do modelo permitir esse tipo de ajuste, isso está fora do escopo inicial desse trabalho. É citada essa possibilidade apenas como demonstração da flexibilidade e alto grau de ajuste que esse modelo possibilita.

Percebe-se que esse componente possui visões de entrada que são estáticas em suas estruturas. Essas estruturas foram definidas a partir de trabalhos anteriores sobre o mesmo

conjunto de dados, a definição de como estruturar cada visão também está fora do escopo desse trabalho. As visões escolhidas são suficientes para sozinhas, gerarem uma predição de resultado, e essa é a premissa para que a multivisão possa alavancar a acurácia das classificações. Sendo os classificadores instâncias do *Random Forest*.

4.3 Atualização Autônoma do Modelo

Apesar da avaliação do grau de confiança feita durante a classificação, é esperado que os modelos de aprendizagem de máquina irão degradar com o passar do tempo. Logo, um aumento na taxa de rejeição ocorrerá, causado pela falta de atualização dos modelos de aprendizagem de máquina, pois eles não serão capazes de enfrentar as mudanças no comportamento do tráfego de rede atual. Porém, realizar atualizações periódicas não é uma tarefa fácil, pois demanda assistência de um especialista para prover a adequada rotulagem. É proposto neste trabalho que se atualize os modelos de maneira autônoma sem a assistência humana pelo uso do multivisão.

O procedimento de atualização é realizado periodicamente, por exemplo, a cada semestre. Para cada tarefa de atualização, o banco de dados de eventos é usado para o procedimento de atualização dos modelos. O banco de dados de eventos é composto de eventos de rede rotulados de maneira autônoma pelo módulo de classificação confiável. Assim, os rótulos dos eventos são assinalados de acordo com a classificação com mais alto grau de confiança e consequentemente são usados para melhorar outros modelos que apresentam graus de confiança menores. Como resultado, é proposto uma técnica que pode melhorar os modelos, ou mitigar a sua degradação, a cada atualização desses, provendo assim, modelos atualizados mesmo com o passar do tempo e sendo atualizados com alta confiança e de maneira autônoma.

O pseudocódigo a seguir detalha o procedimento de atualização autônoma do modelo da abordagem proposta.

Algorithm 2 Atualização Autônoma do Modelo

Require:
Dataset $dataset = getInstances()$
Classifiers $pool = \{c_1, ..., c_N\}$
Label Provider $l = getLabelProvider()$
procedure MODELUPDATE($dataset$, $pool$, l)
  for each $instance \in dataset$ **do**
    $label = provideRejectedHighConfLabel(l, instance)$
    for each $learner \in pool$ **do**
      $Update(learner, instance, label)$ ▷ Atualiza pelas classificações acima do limiar de confiança e que foram rejeitadas
    end for
  end for
end procedure

Algoritmo 2: Atualização Autônoma do Modelo

A melhoria da resiliência, ou seja, a capacidade de melhorar a acuraria após um período de baixa acurácia é a função desse módulo. Que constitui uma segunda linha de mitigação da degradação dos modelos, além do uso da multivisão.

Um objetivo secundário é de demonstrar que as estratégias de mitigação da degradação de modelos podem ser complementares. Num futuro é admissível que a incorporação de uma ou mais linhas de defesa além dessas apresentadas aqui, e que podem vir a promover um resultado ainda melhor. Esse modelo segue o princípio geral de alta coesão e baixo acoplamento que permite o aprimoramento e sofisticação futura da solução. Essa característica é importante visto que a expectativa é de aumento do tráfego de rede e consequente aumento do número de ataques conforme o número de serviços e protocolos de rede também aumentam. Logo uma solução que permita a expansão futura está alinhada com o cenário real de produção.

4.4 Discussão

Propõe-se nesta seção tratar das questões relacionadas a detecção de intrusão levantadas no capítulo 1. Uma arquitetura de NIDS baseado em aprendizagem de máquina é proposta para tratar dessas questões, sendo que:

A quantidade e crescimento do tráfego e sua mutabilidade são tratados por meio de classificações que analisam os dados a partir de visões diferentes. Com a complementariedade entre as visões espera-se alavancar a acurácia num contexto mutável e de longo prazo, ou pelo menos, mitigar a degradação dos modelos.

A solução é proposta de forma que o modelo de aprendizagem de máquina possa atuar de forma ininterrupta durante longos prazos. E quando necessário a atualização dos classificadores se possa fazer, de forma automática, sem a necessidade de se esperar por uma assistência humana especializada.

A proposta é modular para que possa também ser expansível, uma vez que o tráfego é crescente e mutável.

As restrições dessa proposta são principalmente em relação a hardware versus o número de visões e hardware versus o tamanho do tráfego passante no ponto de proteção da rede. Essas são questões que precisam ser balizadas no momento de uma eventual implantação. O modelo proposto apresenta flexibilidade suficiente para ser adequado aos cenários mais prováveis de serem encontrados em produção.

A contribuição é a de apresentar um modelo factível em situações reais. Testar suas premissas de uso a longo prazo. E ser o suficiente modular para admitir novos módulos e flexível o suficiente para se adequar ao hardware que por ventura se venha a ter disponível. Não obstante, ainda assim, são feitas avaliações para demonstrar a melhoria em relação ao uso de visões únicas.

O modelo apresentado é simples, modular, expansível e nos módulos apresentados têm parâmetros flexíveis, que permitem uma personalização ao cenário do usuário.

Capítulo 5

Avaliação

Neste capítulo serão abordadas as avaliações realizadas ao longo do trabalho. Para tanto, primeiramente será apresentado o conjunto de dados utilizados nas avaliações, e uma avaliação da performance dos algoritmos comumente utilizados na literatura em face a mudanças do comportamento do tráfego. Posteriormente, o modelo proposto será avaliado e comparado com as técnicas da literatura. Para tanto, a avaliação do modelo proposto busca responder as seguintes questões de pesquisa.

1. *A avaliação dos valores de confiança das classificações pode melhorar a acurácia do modelo proposto*?
2. *O módulo proposto de pool evaluator é capaz de manter a confiança (reliability) ao longo do tempo, sem atualizações periódicas do modelo?*
3. *O esquema proposto é capaz de atualizar o modelo de maneira autônoma e manter-se robusto (reliable) ao longo do tempo?*

5.1 Definição do problema

As mudanças no comportamento do tráfego de rede ao longo do tempo são um desafio conhecido mas desprezado na literatura do NIDS. O tempo de vida dos esquemas de detecção de intrusão propostos permanece desconhecido. A degradação da precisão causada pelas mudanças naturais no tráfego da rede com o passar do tempo nem mesmo é avaliada. Esta seção

investiga mais a fundo como as técnicas tradicionais baseadas em AM funcionam ao detectar tentativas de intrusão relacionadas à rede por longos períodos. Primeiro, é apresentado um novo conjunto de dados de intrusão que contém tráfego de rede real e rotulado por um intervalo de 2 anos. Em seguida, avaliamos uma técnica de AM amplamente utilizada em relação à degradação da precisão ao longo do tempo, com vários conjuntos de recursos de trabalhos relacionados.

5.1.1 Conjunto de dados MAWIFlow

Conjuntos de dados comumente usados podem ter décadas desde a sua criação, contendo tráfego de dados que não são realistas, ataques obsoletos, e vários problemas conhecidos [26][27][28]. Em princípio, conjunto de dados usados para esse propósito devem conter tráfego de rede válido e real, descrevendo fluxos de comunicação entre os serviços e os ataques que realmente ocorrem em produção. Entretanto, o monitoramento e a rotulagem de tráfego real para desenvolvimento de um projeto de NIDS não é possível devido a problemas de privacidade. Como por exemplo, gravar o tráfego de dados de uma universidade.

Neste trabalho é utilizado o conjunto de dados MAWIFlow, um conjunto de dados disponibilizado publicamente que contém tráfego real, válido e rotulado de ambiente de produção por um longo período. Para prover tais características, MAWIFlow é construído sobre o arquivo de tráfego do grupo de trabalho MAWI [39] (MAWI Working Group Traffic Archive) que contém o tráfego de rede do *samplepoint-F do MAWI*, um link entre Japão e EUA coletado por 15 minutos diariamente. Para este trabalho, os dados utilizados serão de um intervalo de 2 anos de 2015 a 2016.

Os dados da rede são coletados diariamente, a partir disso, é criado um arquivo PCAP para cada dia durante o intervalo de 2 anos, compreendendo mais de 7 TB de dados e mais de 70 bilhões de fluxos de rede. Os dados coletados são sumarizados em cada comunicação. Cada fluxo de comunicação compreende 15s de dados de cliente/serviço e servidor/serviço, que é sumarizado em um vetor de características (visão).

O conjunto de dados construído, chamado de MAWIFlow, é feito de 3 visões distintas, extraídas de cada fluxo de rede, sendo Nigel [40], Orunada [41], e Viegas [42], em que cada um é composto por 20, 15 e 47 características respectivamente.

A visão Nigel é apresentada a seguir, são descritos cada campo da visão:

#	Descrição
1	Comprimento mínimo do pacote de rede durante o intervalo de tempo do envio
2	Comprimento médio do pacote de rede durante o intervalo de tempo do envio
3	Comprimento máximo do pacote de rede durante o intervalo de tempo do envio
4	Desvio padrão do comprimento do pacote de rede durante o intervalo de tempo do envio
5	Comprimento mínimo do pacote de rede durante o intervalo de tempo do recebimento
6	Comprimento médio do pacote de rede durante o intervalo de tempo do recebimento
7	Comprimento máximo do pacote de rede durante o intervalo de tempo do recebimento
8	Desvio padrão do comprimento do pacote de rede durante o intervalo de tempo do recebimento
9	Tempo mínimo de chegada de pacote de rede enviado durante o intervalo de tempo
10	Tempo médio de chegada do pacote de rede enviado durante o intervalo de tempo
11	Tempo máximo de chegada de pacote de rede enviado durante o intervalo de tempo
12	Desvio padrão do tempo de chegada do pacote de rede enviado durante o intervalo de tempo
13	Tempo mínimo de chegada de pacote de rede recebido durante o intervalo de tempo
14	Tempo médio de chegada de pacote de rede recebido durante o intervalo de tempo
15	Tempo máximo de chegada de pacote de rede recebido durante o intervalo de tempo
16	Desvio padrão do tempo de chegada do pacote de rede recebido durante o intervalo de tempo
17	Protocolo (UDP, TCP ou ICMP)
18	Número de pacotes de rede enviados
19	Número de bytes recebidos
20	Número de pacotes de rede recebidos

Tabela 3: Visão Nigel

A visão Orunada é apresentada a seguir, são descritos cada campo da visão:

#	Descrição

1	Porcentagem de pacotes vistos entre a comunicação host / host com o *flag* TCP SYN definido.
2	Porcentagem de pacotes vistos entre a comunicação host / host com o *flag* TCP ACK definido.
3	Porcentagem de pacotes vistos entre a comunicação host / host com o *flag* TCP RST definido.
4	Porcentagem de pacotes vistos entre a comunicação host / host com o *flag* TCP FIN definido.
5	Porcentagem de pacotes vistos entre a comunicação host / host com o *flag* TCP CWR definido.
6	Porcentagem de pacotes vistos entre a comunicação host / host com o *flag* TCP URG definido.
7	Tamanho médio do pacote visto entre a comunicação host / host
8	Valores TTL médios vistos entre a comunicação host / host
9	Porcentagem de pacotes vistos entre a comunicação host / host com conjunto de *flag* de redirecionamento ICMP.
10	Porcentagem de pacotes vistos entre a comunicação host / host com o conjunto de *flag* excedido de tempo ICMP.
11	Porcentagem de pacotes vistos entre a comunicação host / host com conjunto de *flag* de ICMP inacessível.
12	Porcentagem de pacotes vistos entre a comunicação host / host com o conjunto de *flag* ICMP de outros tipos.
14	Número de hosts únicos enviando pacotes de rede para o host
15	Número de serviços exclusivos que enviam pacotes de rede para o host

Tabela 4: Visão Orunada

A visão Viegas é apresentada a seguir, são descritos cada campo da visão:

1	Número de Pacotes
2	Número de Bytes
3	Tamanho médio do pacote
4	Porcentagem de pacotes (PSH Flag)

5	Porcentagem de pacotes (*flag* SYN e FIN)
6	Porcentagem de pacotes (*flag* FIN)
7	Porcentagem de pacotes (*flag* SYN)
8	Porcentagem de pacotes (*flag* ACK)
9	Porcentagem de pacotes (*flag* RST)
10	Porcentagem de pacotes (*flag* de redirecionamento ICMP)
12	Porcentagem de pacotes (*flag* de redirecionamento ICMP)
13	Porcentagem de pacotes (*flag* de tempo excedido ICMP)
14	Porcentagem de pacotes (ICMP *Unreachable Flag*)
15	Porcentagem de pacotes (*flag* de outros tipos ICMP)
16	Tamanho médio do pacote, taxa de transferência em bytes
17	Protocolo
18	Número de pacotes - ambas as direções
19	Número de bytes - ambas as direções
20	Tamanho médio do pacote - ambas as direções
21	Porcentagem de pacotes (*flag* PSH) - ambas as direções
22	Porcentagem de pacotes (*flag* SYN e FIN) - ambas as direções
23	Porcentagem de pacotes (*flag* FIN) - ambas as direções
24	Porcentagem de pacotes (*flag* SYN) - ambas as direções
25	Porcentagem de pacotes (*flag* ACK) - ambas as direções
26	Porcentagem de pacotes (*flag* RST) - ambas as direções
27	Porcentagem de pacotes (ICMP *Redirect Flag*) - Ambas as direções
28	Porcentagem de pacotes (ICMP *Redirect Flag*) - Ambas as direções
29	Porcentagem de pacotes (ICMP *Time Exceeded Flag*) - Ambas as direções
30	Porcentagem de pacotes (ICMP *Unreachable Flag*) - Ambas as direções
31	Porcentagem de pacotes (sinalizador de outros tipos *ICMP*) - ambas as direções
32	Taxa de transferência em bytes - da origem ao destino

33	Número de pacotes - fonte para destino
34	Número de bytes - fonte para destino
35	Tamanho médio do pacote - da origem ao destino
36	Porcentagem de pacotes (PSH *Flag*) - Fonte para Destino
37	Porcentagem de pacotes (*flag* SYN e FIN) - Origem para o destino
38	Porcentagem de pacotes (*flag* FIN) - Origem para o destino
39	Porcentagem de pacotes (*flag* SYN) - Origem para o destino
40	Porcentagem de pacotes (*flag* ACK) - Origem para o destino
41	Porcentagem de pacotes (*flag* RST) - Origem para o destino
42	Porcentagem de pacotes (ICMP Redirect *flag*) - Origem para o destino
43	Porcentagem de pacotes (ICMP Redirect *flag*) - Origem para o destino
44	Porcentagem de pacotes (ICMP Time Exceeded *flag*) - Origem para o destino
45	Porcentagem de pacotes (ICMP Unreachable *flag*) - Origem para o destino
46	Porcentagem de pacotes (*flag* de outros tipos ICMP) - Origem para o destino
47	Taxa de transferência em bytes - destino para a origem

Tabela 5: Visão Viegas

Consequentemente, cada conjunto de características provê uma visão distinta para o mesmo evento, representado pelo conjunto de características extraídas. Para rotulagem, é aplicado o algoritmo não supervisionado MAWILab [43] que identifica anomalias na rede e que rotula os ataques.

5.1.2 Performance das técnicas tradicionais

Foi investigado o impacto das mudanças naturais no tráfego de rede em relação ao algoritmo de aprendizagem de máquina em cada visão [40] [41] [42].

Foi aplicado o algoritmo Random Forest, um classificador amplamente usado, com 100 árvores de decisão como classificadores base (“base learner”), usando o primeiro mês de janeiro como o conjunto de dados de treinamento e testado cada visão separadamente durante o ano

todo sem atualização do modelo. Devido à natureza altamente desbalanceada do conjunto de dados, em que a maioria dos eventos são normais (aproximadamente 99% de tráfego normal e 1% de ataques), foi conduzido um "random undersampling" sem o procedimento de estratificação no dados de treinamento para equilibrar a ocorrência de classes e foi utilizado a API do *scikit-learn* para em todo os procedimentos. As classificações foram avaliados de acordo com suas taxas de Falsos-positivos (FP) e Falsos-negativos (FN). Os FPs denotam a proporção de eventos normais que foram classificados como ataques, enquanto que os FN denotam a proporção de eventos de ataque que foram classificados como benignos.

As figuras 4, 5 e 6 mostram as taxas de erros em cada classificador Random Forest, com os conjuntos de dados Nigel [40] , Orunada [41] e Viegas [42]:

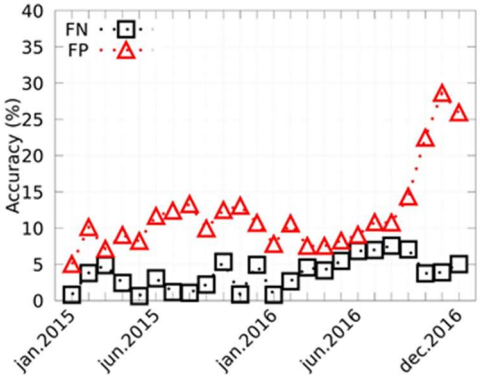


Figura 3: Visão Nigel Random Forest (100 árvores)

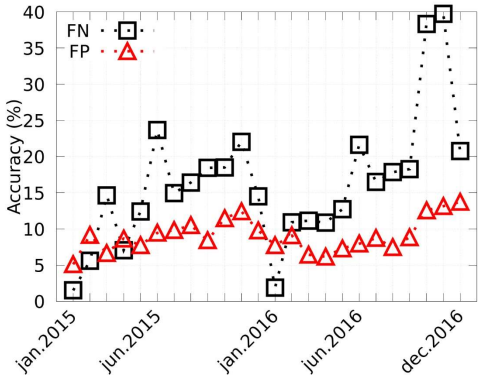


Figura 4: Visão Orunada Random Forest (100 árvores)

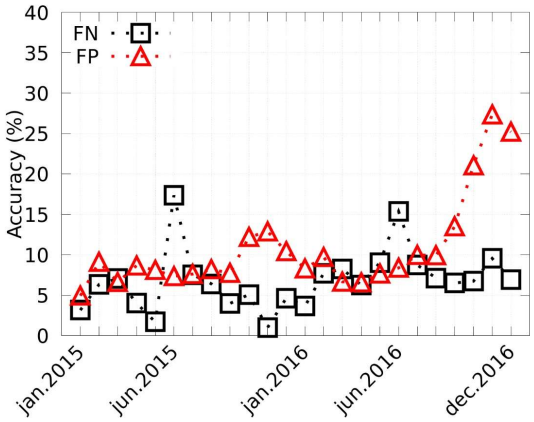


Figura 5: Visão Viegas Random Forest (100 árvores)

É possível notar uma taxa de erro crescente nos meses que seguem ao período de treinamento (Janeiro de 2015), independentemente da visão utilizada. Porém o aumento da taxa de erro varia de acordo com a visão utilizada. Por exemplo, a visão Nigel [40] aumenta o taxa de FN ao longo de 2015 enquanto mantém a taxa de FP estável no mesmo período, enquanto que a visão Orunada [41] aumenta significativamente tanto na taxa de FP quanto em FN em 2015 enquanto prove resultados mais acurados em 2016.

Na média, os classificadores aumentaram suas taxas de FP e FN em 3,7% e 4,8% no primeiro mês após o período de treinamento. A pior queda na acurácia foi notada por todas as visões utilizadas em Novembro de 2016, assim como foi notado um aumento em suas taxas de FP e FN. Logo, independentemente da visão utilizada, as mudanças naturais ao longo do tempo no tráfego de rede afetam a confiabilidade das classificações. Porém cada visão apresenta um impacto diferente em sua confiabilidade ao longo do tempo, assim demonstrando que a visão usada pode ajudar a melhorar a confiabilidade do sistema se o mecanismo proposto puder explorar tais comportamentos.

5.2 Avaliação da proposta

5.2.1 Treinamento dos modelos

O modelo de detecção de intrusão de multivisão confiável proposto foi construído, levando em consideração as três visões avaliadas anteriormente (Figuras 4, 5 e 6). Portanto, três classificadores de *Random Forest* foram treinados, cada um com uma visão de evento única, usando as visões Nigel [39], Orunada [40] e Viegas [41]. Da mesma forma, os classificadores *Random forest* foram treinados com 100 árvores de decisão como seus *base-learners*. Os valores de confiança, usados por nossa proposta para garantir a confiabilidade foram obtidos por meio da API scikit-learn, conforme calculado pela função *predict_proba*. A API calcula os valores de confiança do classificador de *Random Forest* como a proporção de árvores individuais que classificaram a instância avaliada como a classe gerada.

5.2.2 Performance ao longo do tempo

O primeiro experimento visa responder à primeira questão de pesquisa da avaliação, sendo ela: *o uso de valores de confiança das classificações pode melhorar a acurácia do modelo de aprendizagem de máquina?*

Para tanto será comparado cada relação de compromisso (*tradeoff*) entre erro e rejeição de cada visão. Cada classificador é treinado em janeiro de 2015 e avaliado a partir de fevereiro de 2015. Os limites (threshold) de classificação foram definidos por meio da abordagem Class-Related-Threshold (CRT). Os valores de confiança calculados para cada visão em fevereiro são avaliados para estabelecer a relação de compromisso (*tradeoff*) entre erro e rejeição. Esse permite definir o conjunto de limites que devem ser usados ao longo do intervalo de 2 anos remanescentes (limites de confiança). Para todas as visões avaliadas, mostradas na figura 7, pode-se ver que é possível diminuir a taxa de erro com um aumento na taxa de rejeição como compensação. Portanto, a confiança pode ser usada para avaliar a qualidade da classificação. Além disso, cada visão fornece diferentes compensações, com o conjunto de recursos Viegas, fornecendo a melhor relação entre erro e rejeição.

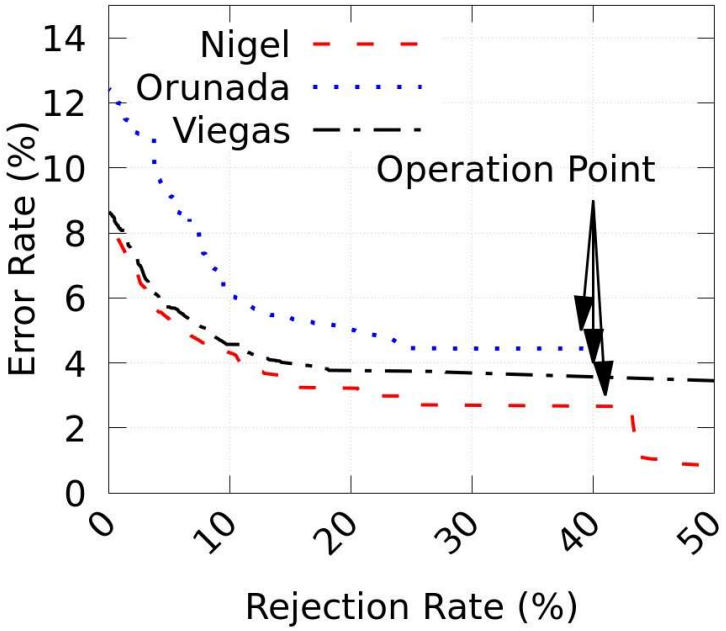


Figura 6: Class-Related-Threshold (CRT)

O segundo experimento visa responder à segunda questão de pesquisa da avaliação, sendo ela: *o proposto pool evaluator é capaz de manter a robustez (reliability) ao longo do tempo, sem atualização do modelo?*

Para tanto busca-se aplicar os limites de classificação obtidos na primeira questão ao longo do intervalo restante de 2 anos. Para atingir tal objetivo, cada limite de classificação de visão é definido em uma taxa de rejeição de 40% conforme medido em fevereiro de 2015. O conjunto definido de limites para cada visão é usado ao longo do intervalo de 2 anos restantes sem atualizações de modelo. Assim, se uma classificação de uma visão não atende ao limite relacionado à classe definido, ela não é usada para o processo de votação por maioria do grupo e se o resultado da classificação de todas as visões também não atingiu seus limites, o evento é rejeitado.

A Figura 8 mostra as taxas de precisão e rejeição da proposta obtidas ao longo do tempo sem atualizações ao aplicar os limites de classificação definidos ao longo do intervalo de 2 anos do conjunto de dados MAWIFlow. Percebe-se que apesar de utilizar um ponto de operação de taxa de rejeição de 40%, a proposta é capaz de rejeitar significativamente menos instâncias, rejeitando em média 14,2% instâncias no conjunto de dados. Com o tempo, as taxas de precisão também permaneceram estáveis em comparação com o período de treinamento (janeiro de 2015), aumentando seu FP em média em apenas 2,1%, mantendo sua taxa de FN estável em 2015. Assim, quando comparada à abordagem tradicional de uma visão (figura 9), o método multivisão proposto é capaz de permanecer robusto por períodos mais extensos, aumentando seu PF em média em apenas 3,5% no intervalo de 2 anos, em contraste com a técnica de visão única, que aumenta seu FP em 9,2%, 6,2% e 8,0% com as visões Nigel, Orunada e Viegas respectivamente.

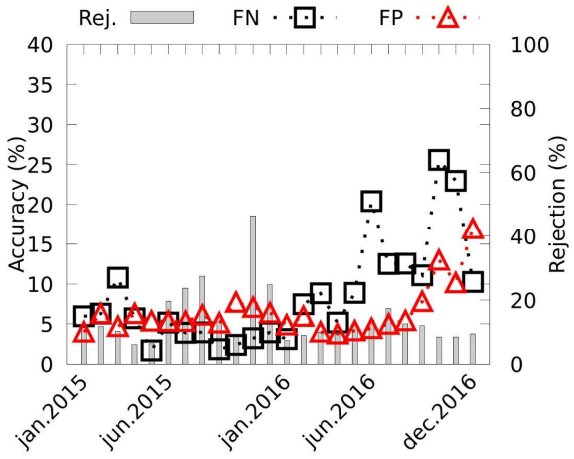


Figura 7: Taxa de acurácia e rejeição sem atualização de modelos.

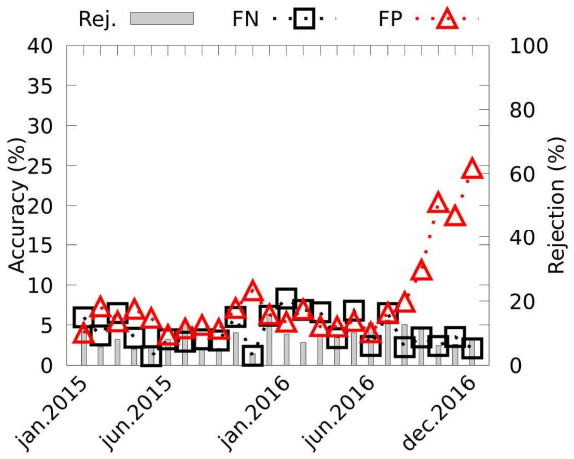


Figura 8: Taxa de acurácia e rejeição com atualização autônoma de modelos.

Por fim, o último experimento busca responder à terceira pergunta de pesquisa da avaliação, sendo ela: *o esquema proposto é capaz de atualizar o modelo e manter-se robusto (reliable) autonomamente ao longo do tempo?*

Portanto, atualizamos de forma autônoma os modelos de aprendizagem de máquina semestralmente (intervalo de 6 meses), de acordo com o rótulo de evento atribuído, sem assistência humana. As Figuras 10 e 11 mostram que a proposta obteve taxas de precisão e rejeição (com atualizações do modelo em um intervalo de 6 meses) ao se aplicar os limites de classificação definidos durante um intervalo de 2 anos para o conjunto de dados MAWIFlow. O esquema proposto diminui significativamente a taxa de rejeição ao mesmo tempo em que melhora sua precisão, quando as atualizações autônomas do modelo são realizadas, rejeitando em média apenas 8,8%, em contraste com 14,2% se nenhuma atualização do modelo for realizada, melhorando sua precisão em 4,3%.

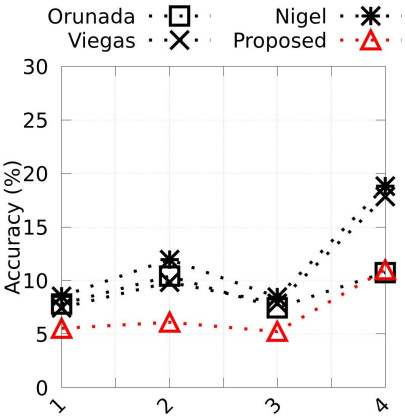


Figura 9: Taxas de Falso Negativos ao longo do tempo.

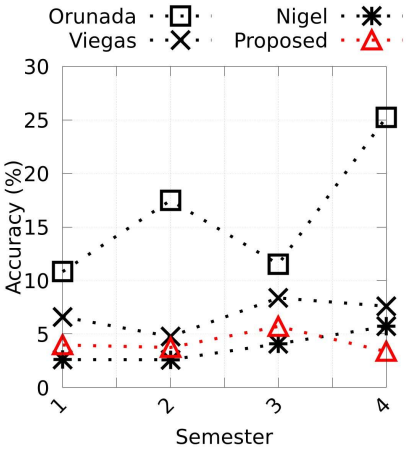


Figura 10: Taxas de Falso Positivos ao longo do tempo.

5.3 Discussão

O esquema proposto atualizou de forma confiável os modelos de AM subjacentes, aproveitando a técnica de pool evaluator e multivisão. O pool evaluator avalia a qualidade da classificação para garantir que apenas classificações altamente confiáveis sejam aceitas. Por outro lado, o multivisão garante que o sistema permaneça confiável ao longo do tempo, usando cada visão para melhorar outras visões complementares durante as atualizações de classificação e modelo.

Os resultados obtidos nas questões de pesquisa indicam que o modelo proposto é viável, uma vez que os resultados obtidos ao longo de dois anos confirmaram as expectativas de aumento de confiança (reliability) e acurácia, em relação às classificações utilizando-se visão única.

Faz-se necessário notar que a degradação dos modelos foi mitigada pelo uso da técnica de multivisão. E quando aplicado um retreino a cada 6 meses essa degradação foi mitigada mais ainda.

Foram utilizados dados de tráfego real e por um longo prazo, o que não é típico em pesquisas da área. Ainda assim é necessário que a solução continue a ser melhorada com a incrementação de outras técnicas que possam ser complementares a técnica apresentada aqui. Entre elas pode-se citar a diversidade de modelos e o uso de janelas deslizantes para que os modelos “esqueçam” ataques antigos e consigam distinguir novos ataques do tráfego normal.

Os esforços de pesquisa nessa área são contínuos e frequentemente complementares. Sendo assim, a contribuição aqui apresentada pode ser reutilizada e melhorada futuramente. E essa é uma importante característica do modelo apresentado porque, a expetativa é de um contínuo aumento de tráfego, aumento do número de serviços e aumento do número de ataques. O que torna a detecção e defesa perfeita um alvo móvel. Logo é importante avançar nas tecnologias de detecção (como foi demonstrado aqui) e manter uma perspectiva de melhoria contínua.

Capítulo 6

Conclusão

A tendência de aumento de tráfego e aumento das ameaças advindas de um cenário de crescente uso da internet faz com que o desenvolvimento de técnicas de NIDS estejam em constante evolução.

A adoção do uso de aprendizagem de máquina para NIDS traz desafios bem maiores que os encontrados em outras áreas onde seu uso obteve sucesso. Entre eles estão a grande lacuna (*gap*) semântica entre os dados analisados e sua interpretação, a mão-de-obra cara e escassa, a perspectiva pouco realista que é comumente encontrada na literatura com uso de conjunto de dados inadequados derivados de cenários pouco realistas, e a falta de abordagens que levem em consideração a passagem do tempo a longo prazo.

Os trabalhos comumente apresentados na literatura sobre intrusão de detecção se direcionam a busca incessante do aumento da acurácia pela proposta de novos algoritmos ou pela combinação de algoritmos que normalmente apresentam um ganho marginal de acurácia.

É proposto neste trabalho uma abordagem multivisão utilizando-se de um conjunto de dados coletados de um cenário real em um link *back-bone* de internet, em que a atualização autônoma dos modelos visa a estender suas vidas úteis diminuindo assim a necessidade de especialistas. A técnica de multivisão em conjunto com a atualização autônoma visa a manter a acurácia ao longo do tempo (robustez) da solução.

A detecção de intrusão é uma área complexa e muitas facetas do problema de se aprimorar um IDS são ignorados, mas são contempladas na arquitetura proposta. Entre elas se estão:

A confiança do sistema ao longo do tempo, que frequentemente é ignorada na literatura, é abordada no presente trabalho, sendo que os resultados preliminares com o uso de diversidade de dados se mostram promissores.

A manutenção da resiliência da acurácia da solução, seja pela utilização de classificações com altas taxa de confiabilidade, seja pela atualização autônoma dos modelos é concebida como um item importante da arquitetura proposta.

Autonomia da arquitetura, que visa não só diminuir os custos de se ter um especialista retreinando os modelos, mas também tornar o uso do NIDS baseado em aprendizagem de máquina, mais atraente mesmo como uma segunda linha de defesa da segurança do sistema computacional. Exigindo menos manutenção e menos atenção por parte dos operadores.

Apesar da abundância de trabalhos que relatam esquemas de detecção de intrusão baseados em AM de alta precisão, as técnicas propostas dificilmente são usadas em produção por não serem testados em cenários realistas. Nesse sentido a atual proposta visa a dar avançar na utilização de técnicas inovadoras em um cenário realista.

Como trabalho futuro será explorado a expansão da arquitetura proposta utilizando-se tanto da diversidade de visões (multivisão) quanto da diversidade de modelos.

Referências

[1] "IT threat evolution Q2 2020." https://securelist.com/it-threat-evolution-q2-2020-pc-statistics/98292/.

[2] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho, "A survey of network-based intrusion detection data sets," *Comput. Secur.*, vol. 86, pp. 147–167, 2019, doi: 10.1016/j.cose.2019.06.005.

[3] E. Viegas, A. Santin, A. Bessani, and N. Neves, "BigFlow: Real-time and reliable anomaly-based intrusion detection for high-speed networks," *Futur. Gener. Comput. Syst.*, vol. 93, no. Ml, pp. 473–485, 2019, doi: 10.1016/j.future.2018.09.051.

[4] C. Brenton and C. Hunt, *Mastering network security*, 2nd ed. Indianapolis, IN: Sybex, 2002.

[5] E. K. Viegas, A. O. Santin, V. V Cogo, and V. Abreu, "A Reliable Semi-Supervised Intrusion Detection Model : One Year of Network Traffic Anomalies."

[6] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," *Proc. - IEEE Symp. Secur. Priv.*, pp. 305–316, 2010, doi: 10.1109/SP.2010.25.

[7] C. Gates and C. Taylor, "Challenging the anomaly detection paradigm a provocative discussion," *Proc. New Secur. Paradig. Work.*, pp. 21–29, 2007, doi: 10.1145/1278940.1278945.

[8] O. P. (Eds.). Goyal, D., Balamurugan, S., Peng, S., & Verma, *Design and analysis of security protocol for communication*. Wiley, 2020.

[9] R. Johnson and C. Easttom, *Security Policies and Implementation Issues, 3rd Edition*. Jones & Bartlett Learning, 2020.

[10] "No Title." https://www.broadcom.com/products/cyber-security/endpoint/management/ghost-solutions-suite#system-requirements.

[11] J. Kizza and F. Migga Kizza, "Intrusion Detection and Prevention Systems," *Secur. Inf. Infrastruct.*, pp. 239–258, 2011, doi: 10.4018/978-1-59904-379-1.ch012.

[12] R. Price, *CompTIA Server certification guide : a comprehensive, end-to-end study guide*

for the SK0-004 certification, along with mock exams. 2019.

[13] M. Collins, *Data-driven network analysis: A higher stage of computer security*. Sebastopol, CA: O'Reilly Media, 2014.

[14] H. Bidgoli, *Handbook of information security: Threats, vulnerabilities, prevention, detection and management v. 3: Volume 3: Threats, vulnerabilities, prevention, detection, and management*. Nashville, TN: John Wiley & Sons, 2006.

[15] W. Stallings, *Information Privacy Engineering and Privacy by Design: Understanding Privacy Threats, Technology, and Regulations Based on Standards and Best Practices*. Addison-Wesley Professional, 2020.

[16] R. C. Newman, *Computer security: Protecting digital resources*. Sudbury, MA: Jones and Bartlett, 2009.

[17] I. Dubrawsky, *Eleventh hour security+: Exam SY0-201 study guide*. Rockland, MA: Syngress Media, 2009.

[18] Y. Diogenes and E. Ozkaya, *Cybersecurity - attack and defense strategies : counter modern threats and employ state-of-the-art tools and techniques to protect your organization against cybercriminals*, 2nd ed. 2019.

[19] T. M. Thomas and D. Stoddard, *Network security first-step*, 2nd ed. Indianapolis, IN: Cisco Press, 2011.

[20] J. E. van Engelen and H. H. Hoos, "A survey on semi-supervised learning," *Mach. Learn.*, vol. 109, no. 2, pp. 373–440, 2020, doi: 10.1007/s10994-019-05855-6.

[21] S. Raschka, D. Julian, and J. Hearty, *Python: Deeper insights into machine learning*. Birmingham, England: Packt Publishing, 2016.

[22] A. Albalate and W. Minker, *Semi-supervised and unsupervised machine learning: Novel strategies*. London, England: ISTE Ltd and John Wiley & Sons, 2010.

[23] G. Ciaburro, K. Ayyadevara, and A. Perrier, *Hands-On Machine Learning on Google Cloud Platform: Implementing smart and efficient analytics using Cloud ML Engine*. Birmingham, England: Packt Publishing, 2018.

[24] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," *Proc. Annu. ACM Conf. Comput. Learn. Theory*, pp. 92–100, 1998, doi: 10.1145/279943.279962.

[25] H. S. Bhatt, S. Bharadwaj, R. Singh, M. Vatsa, A. Noore, and A. Ross, "On co-training online biometric classifiers," *2011 Int. Jt. Conf. Biometrics, IJCB 2011*, 2011, doi:

10.1109/IJCB.2011.6117519.

[26] A. H. Mirza, “Computer network intrusion detection using various classifiers and ensemble learning,” in *2018 26th Signal Processing and Communications Applications Conference (SIU)*, 2018, pp. 1–4, doi: 10.1109/SIU.2018.8404704.

[27] P. Singh and M. Venkatesan, “Hybrid Approach for Intrusion Detection System,” in *2018 International Conference on Current Trends towards Converging Technologies (ICCTCT)*, 2018, pp. 1–5, doi: 10.1109/ICCTCT.2018.8551181.

[28] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, “Survey of intrusion detection systems: techniques, datasets and challenges,” *Cybersecurity*, vol. 2, no. 1, p. 20, 2019, doi: 10.1186/s42400-019-0038-7.

[29] H. He, X. Sun, H. He, G. Zhao, L. He, and J. Ren, “A Novel Multimodal-Sequential Approach Based on Multi-View Features for Network Intrusion Detection,” *IEEE Access*, vol. 7, pp. 183207–183221, 2019, doi: 10.1109/ACCESS.2019.2959131.

[30] W. Li, W. Meng, Z. Tan, and Y. Xiang, “Design of multi-view based email classification for IoT systems via semi-supervised learning,” *J. Netw. Comput. Appl.*, vol. 128, pp. 56–63, 2019, doi: 10.1016/j.jnca.2018.12.002.

[31] S. Velayati, Elham; Hazrati, “Malware Detection and Identification using Multi-View Learning based on Sparse Representation,” 2020.

[32] Y. Zhao *et al.*, “Network anomaly detection by using a time-decay closed frequent pattern,” *Inf.*, vol. 10, no. 8, pp. 1–18, 2019, doi: 10.3390/info10080262.

[33] T. H. Kobayashi, A. B. Batista, A. M. Brito, and P. S. M. Pires, “Using a packet manipulation tool for security analysis of industrial network protocols,” in *2007 IEEE Conference on Emerging Technologies and Factory Automation (EFTA 2007)*, 2007, pp. 744–747, doi: 10.1109/EFTA.2007.4416847.

[34] F. Breve and L. Zhao, “Semi-supervised learning with concept drift using particle dynamics applied to network intrusion detection data,” *Proc. - 1st BRICS Ctries. Congr. Comput. Intell. BRICS-CCI 2013*, pp. 335–340, 2013, doi: 10.1109/BRICS-CCI-CBIC.2013.63.

[35] A. Rahman and B. Verma, “Novel layered clustering-based approach for generating ensemble of classifiers,” *IEEE Trans. Neural Networks*, vol. 22, no. 5, pp. 781–792, 2011, doi: 10.1109/TNN.2011.2118765.

[36] O. Y. Al-Jarrah, Y. Al-Hammdi, P. D. Yoo, S. Muhaidat, and M. Al-Qutayri, “Semi-

supervised multi-layered clustering model for intrusion detection," *Digit. Commun. Networks*, vol. 4, no. 4, pp. 277–286, 2018, doi: 10.1016/j.dcan.2017.09.009.

[37] A. Chiche and M. Meshesha, "Towards a Scalable and Adaptive Learning Approach for Network Intrusion Detection," *J. Comput. Networks Commun.*, vol. 2021, 2021, doi: 10.1155/2021/8845540.

[38] S. Krishnaveni, S. Sivamohan, S. S. Sridhar, and S. Prabakaran, "Efficient feature selection and classification through ensemble method for network intrusion detection on cloud computing," *Cluster Comput.*, vol. 24, no. 3, pp. 1761–1779, 2021, doi: 10.1007/s10586-020-03222-y.

[39] "MAWI Working Group Traffic Archive." https://mawi.wide.ad.jp/mawi/.

[40] N. Williams, S. Zander, and G. Armitage, "A preliminary performance comparison of five machine learning algorithms for practical IP traffic flow classification," *Comput. Commun. Rev.*, vol. 36, no. 5, pp. 7–15, 2006, doi: 10.1145/1163593.1163596.

[41] J. Dromard, G. Roudière, and P. Owezarski, "Online and Scalable Unsupervised Network Anomaly Detection Method," *IEEE Trans. Netw. Serv. Manag.*, vol. 14, no. 1, pp. 34–47, 2017, doi: 10.1109/TNSM.2016.2627340.

[42] E. Viegas, A. O. Santin, A. França, R. Jasinski, V. A. Pedroni, and L. S. Oliveira, "Towards an Energy-Efficient Anomaly-Based Intrusion Detection Engine for Embedded Systems," *IEEE Trans. Comput.*, vol. 66, no. 1, pp. 163–177, 2017, doi: 10.1109/TC.2016.2560839.

[43] R. Fontugne, P. Borgnat, P. Abry, and K. Fukuda, "Mawilab: Combining diverse anomaly detectors for automated anomaly labeling and performance benchmarking," *Co-NEXT'10*, Jan. 2010.

Apêndice 1

O presente trabalho gerou a publicação do artigo "A Multi-View Intrusion Detection Model for Reliable and Autonomous Model Updates" na IEEE International Conference on Communications realizada de 14 a 23 de Junho de 2021, realizada em Montreal, Canadá.

Abaixo está transcrito o artigo como publicado na referida conferência.

A Multi-View Intrusion Detection Model for Reliable and Autonomous Model Updates

Rivaldo L. Tomio*, Eduardo K. Viegas*, Altair O. Santin*, Roger R. dos Santos*
*Graduate Program in Computer Science
Pontifical Catholic University of Parana, Brazil
{rivaldo.luiz, eduardo.viegas, santin, robson.roger}@ppgia.pucpr.br

***Abstract*—Changes in network traffic behavior over time are neglected by authors who use machine learning techniques applied to intrusion detection. In general, it is assumed that periodic model updates are performed, regardless of the challenges related to such a task. This paper proposes a new multi-view intrusion detection model capable of reliably performing model updates without human assistance while also maintaining its accuracy over time. The proposal evaluates the classification's confidence values in a multi-view configuration to maintain its reliability over time, even without model updates. Besides, it is able to perform model updates autonomously, according to the result of the multi-view classification. Our experiments, performed with 7TB of real network traffic over a 2-year interval, show that our proposed scheme can maintain its accuracy over time without model updates, rejecting only 14.2% of its classification. However, when autonomous model updates are performed, the rejection rate drops to just 8.8%, while also improving the model's accuracy by 4.3%.**

***Index Terms*—Intrusion Detection, Machine Learning, Multi-View, Model Updates.**

I. Introduction

In the second quarter of 2020, Kaspersky solutions have identified over 800 million network attacks [1]. As a result, operators need access to security solutions to detect this growing number of threats [2]. In general, administrators often employ Network-based Intrusion Detection Systems (NIDS) to detect malicious activities in network traffic through either *misuse-based* or *behavior-based* approaches [3]. The previous searches for attack footprints, namely signatures, within the network traffic, thus, can only detect well-known attacks. On the other hand, *behavior-based* techniques aim to detect attacks by analyzing the network traffic behavior. As a result, it can detect new attacks, as long as their behavior is similar to known threats or significantly differs from benign ones [2].

In the literature, *behavior-based* NIDS has often been performed through machine learning (ML) techniques, wherein pattern recognition approaches are typically used. This kind of detection scheme relies on a training dataset to extract a behavioral model. Then, the built model can be used for the classification of further events [4]. Consequently, if the environment behavior changes, e.g., a new attack is discovered or a new service is provided, the underlying ML model becomes unreliable. This is because the training dataset in which the ML model was built does not contain the current environment behavior, rendering the ML model outdated [5].

Obsolete ML models are unable to reach the same level of accuracy as those measured during the test phase. Thus, due to an increase in the error rate over time, operators will often discard further alarms [6].However, despite the changes in network traffic behavior over time being a known issue in NIDS field, such a challenge is often neglected in related works [2]. In the literature, the majority of proposed ML-based intrusion detection schemes pursue higher classification accuracies, paying little or no attention to the challenges involved during the model update task [7]. In contrast, researchers often assumes that the network traffic is static, and no changes occur over time, or even that periodic model updates are performed, without even evaluating the need for such updates. Ideally, the ML model must be as recent as possible, taking into account that network traffic behavior may change drastically in a small period, rendering it unreliable [6]. However, the model retraining task is a computationally expensive process that often demands human assistance for the labeling of events, e.g., tag network traffic as either normal or attack, which is not always available or is available with a high cost [7]. As a result, the ML model update task remains overlooked in the literature, and even the lifespan of widely used ML-based techniques is yet to be known.

This paper proposes a novel multi-view intrusion detection model aiming for autonomous model updates and higher detection reliability over time, in a twofold way. First, we leverage a multi-view technique to address the lack of reliability in network traffic classification over time. We assume that we can improve the system reliability when using several distinct and complementary feature sets, namely views. Our insight is that the classification confidence can be used to measure reliability in classification, wherein each view in a multi-view procedure can be used to improve classification reliability even with outdated models. Second, we provide an autonomous model update scheme that leverages the proposed reliable multi-view approach to label network traffic for the model update procedure, thus operating without human assistance. In summary, the main contributions of our paper are:

- We evaluate commonly used ML-based intrusion detection approaches concerning their classification reliability over time. Experiments performed with a 2-year long labeled network traffic, composed of over 7TB of data, have shown that current approaches in the literature

significantly decreases their accuracy months after the training period, regardless of their used feature sets;
- We propose and evaluate a novel multi-view intrusion detection model that can autonomously withstand reliably for long periods of time even without model updates. If so, our proposal can further increase detection accuracy while rejecting fewer events without human assistance;

II. Preliminaries

A. Network-based Intrusion Detection

Network-based Intrusion Detection Systems (NIDS) aims at finding malicious activities within a network environment [7]. A typical NIDS architecture is comprised of four sequential modules, namely *data acquisition, feature extraction, decision* and *alert*. The *data acquisition* module collects data from the environment, reading network packets from a NIC (network interface card). The *feature extraction* module extracts behavioral features from the network data to compound a feature vector, also named as view, e.g. summarizes network packets in a 15-s interval network flow. Finally, the *decision* module establishes the proper event label from the feature vector evaluation e.g., applies an ML model to classify it as normal or attack for the *alert* module proper report it.

In general, in production environments, *signature-based* detection techniques are used by NIDS tools [2]. Such an approach searches the collected data for known attack patterns. Recently significant research effort has been conducted in developing novel attack detection techniques in NIDS, typically through *behavior-based* approaches.

B. Machine Learning for NIDS

The majority of *behavior-based* detection approaches in NIDS relies on machine learning (ML) techniques, typically through pattern recognition means [6]. In such a case, a behavioral model is built through a training dataset, which comprises the expected network traffic behavior for a given period. The training procedure is often performed in a supervised setting, wherein the network event label must be previously known, as the ML model will be built taking it into consideration. However, networked environments present a plethora of challenges not commonly evidenced in other fields [2].

Network traffic behavior constantly changes, either due to new types of services or due to new attacks being discovered [7]. These changes in network traffic behavior over time render the built ML model outdated, which may increase the error rate. However, ML-based techniques are often designed for finding similarities in its input data, rather than finding not previously seen behavior [6]. As a result, if the occurrence of new network traffic, a significant change in the network traffic behavior, as represented by its feature set (view), the built ML model may significantly decrease its accuracies, rendering it unreliable.

Conduct a model update procedure in networked environments is not an easily achieved task, given that the network traffic must be labeled, typically with human assistance, and a computationally expensive process of model retraining must be performed. As a result, despite the reports of several works regarding highly accurate ML models, such techniques remain mostly a research topic, hardly being deployed in production environments [2] [6].

III. Related Works

In the literature, authors often aim at providing higher detection accuracies in a single and static dataset. For instance, Mirza *et al.* [11] proposes an ensemble of classifiers to improve their system detection accuracy. The authors assign a classifier weight according to their measured accuracies, wherein more accurate models have higher classification influence. Singh *et al.* [12] proposes a Random Forest classifier with a K-Means in a hybrid approach for intrusion detection. The authors demonstrated that, in this way, they could maintain high detection rates while also decreasing false alarms. However, although high accuracy rates are reported, these works rely on outdated datasets, through KDD99 [13], that do not consider the natural variability of network traffic behavior over time.

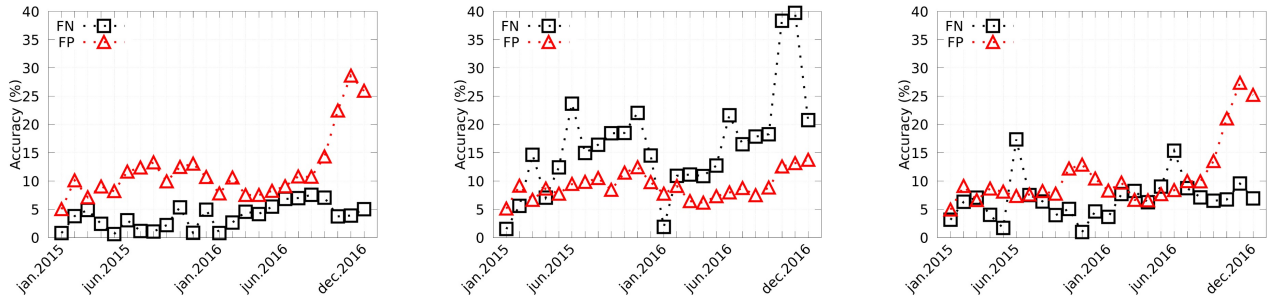
As another approach to improve accuracy, some authors resort to multi-view detection techniques by leveraging distinct and complementary feature sets. For instance, He *et al.* [14] proposes a multimodal approach as a multi-view technique for intrusion detection. The authors are able to significantly improve detection accuracy on a variety of datasets, including outdated and novel ones. However, the authors do not take into account the network traffic behavior changes over time. On the other hand, Li *et al.* [15] proposes a multi-view approach for spam detection in resource-constrained environments. The authors assume a semi-supervised setting wherein a multi-view setting is used for the label of other events. Although the authors consider a more realistic scenario with model updates, their used dataset does not present a long data period without natural behavior changes.

To address the model update challenge, authors often assume that periodic model updates can be performed. For instance, Xiao *et al.* [16] applies an online random forest to detect disk failures over time. However, the authors assume that the event label can be provided when needed, an unrealistic NIDS assumption. Doak *et al.* [17] proposes a self-updating model with error remediation. Their proposed scheme is able to update their model over time autonomously. However, their technique demands the periodic evaluation of their underlying error rates, the proper event label, also not feasible in NIDS.

To the best of our knowledge, we are the first to address the model update challenge in a realistic setting in NIDS. That is, autonomously and reliably updating the underlying ML models through a multi-view classification scheme.

IV. Problem Statement

The changes in network traffic behavior over time is a known and overlooked challenge in NIDS literature. The lifespan of proposed intrusion detection schemes remains unknown. The accuracy degradation caused by the natural



(a) Random Forest - Nigel [8] features. (b) Random Forest - Orunada [9] features. (c) Random Forest - Viegas [10] features.

Fig. 1: Accuracy behavior in a 2-year interval with a Random Forest classifier varying its used feature sets (view). Classifier is trained in January 2015 and evaluated in the remaining period without updates.

changes in network traffic as time passes is not even evaluated. This section further investigates how traditional ML-based techniques perform while detecting network-related intrusion attempts for long periods. We first introduce our novel intrusion dataset that contains real and labeled network traffic for a 2-year interval. Then, we evaluate a widely used ML technique concerning the accuracy degradation over time, with several feature sets from related works.

A. *MAWIFlow Intrusion Dataset*

Widely used datasets are even decades old, containing unrealistic network traffic, outdated attacks, and several known flaws [13]. In principle, datasets used for such purposes must contain real and valid network traffic, containing the proper communication from services and attacks experienced in the wild. However, the monitoring and labeling of real network traffic for NIDS design purposes, e.g., record the network traffic from a university, is not possible due to possible privacy concerns.

Our work proposes the *MAWIFlow* dataset, a publicly available intrusion dataset containing real, valid, and labeled network traffic from production environments that span for an extended period. To provide such characteristics, *MAWIFlow* is built on top of MAWI [18] working group traffic archive. It contains the network traffic from MAWI samplepoint-F, a transit link between Japan and the USA collected for a 15-seconds long interval daily. For this work, the network data from a 2-year interval was used, from 2015 to 2016.

The network data is collected daily, containing a network PCAP file for each day throughout the evaluated 2-year interval, comprising over 7TB of data and over 70 billion network flows. The collected network data is summarized in network flows according to the hosts and services involved in each communication. Each network flow comprises 15-sec of client/service and server/service data, which is then summarized in a set of feature vectors (views).

The built dataset, namely *MAWIFlow*, is made of 3 distinct views, extracted for each network flow, namely Nigel [8], Orunada [9], and Viegas [10], each composed by 20, 15 and 47 features respectively. Hence, each feature set provides a distinct view for the same event, represented by the extracted set of features. For labeling purposes, our work applies MAWILab [19] unsupervised ML algorithms that identify network anomalies, which are then labeled as attacks in our data.

B. *View Performance Over Time*

We evaluate the accuracy degradation of the ML algorithm with respect to the underlying used view. It is important to note that regardless of the ML classifier, the feature set used for the classification is the defining aspect of the ML classification reliability. This is because if a network traffic behavior change occurs, it will only affect the ML classification accuracies, hence, its reliability, if the extracted features are also affected, as the ML model classifies events according to its input feature set values. Therefore, our evaluation further investigates the impact of the natural changes in network traffic to the used ML algorithm, according to distinct views commonly used in related works [9] [10] [8].

We apply the Random Forest algorithm, a widely used ML classifier, with 100 decision trees as its base learner, using the first month of January as the training dataset for the random forest and evaluated each view separately throughout the year without model updates. Due to the imbalanced nature of the dataset, most events are normal (around 99%). We conducted a random undersampling without a replacement stratification procedure in the training dataset to balance the class occurrences while implemented classifiers using the $scikit-learn$ API. The classifiers were evaluated according to their False-Positive (FP) and False-Negative (FN) rates. The FP denotes the ratio of normal events misclassified as attacks, while the FN denotes the ratio of attack events misclassified as benign ones.

Figure 1 shows the monthly measured error rates according to each built random forest classifier, with Nigel [8], Orunada [9], and Viegas [10] feature sets. It is possible to note an error rate increase in the months that followed the training period (January 2015), regardless of the underlying used feature set. However, the error rate increase varies according to the underlying ML view. For instance, Nigel [8] view increases its FN rate throughout 2015 while maintaining its FP rates stable meanwhile, while Orunada [9] view significantly

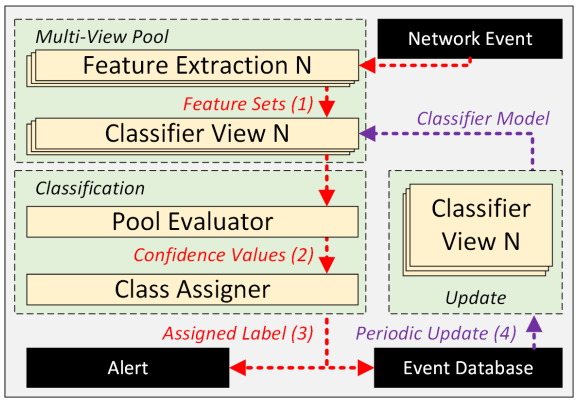


Fig. 2: Proposed multi-view intrusion detection model for reliable and autonomous model updates.

increases both FP and FN rates in 2015 while provides more accurate results in 2016.

On average, the built classifiers increased their FP and FN rates by 3.7% and 4.8% in the first month after the training period. The worst accuracy decrease was noted by all used views in November 2016, as either noted by an increase in their FP or FN rates. Therefore, regardless of the used feature set, the natural changes over time in network traffic affects their reliability. However, each used feature set presented a different impact on their reliability over time, thus showing that the used view can help improve the system reliability if the proposed mechanism can explore such property.

V. A Multi-View Intrusion Detection Model

We present a novel multi-view intrusion detection model to address the evolving behavior of network traffic evaluated previously, composed by two main steps (*Classification* and *Autonomous Model Update*, as shown in Figure 2) and aims to ensure, thus maintain, the system accuracy over time, and also perform autonomous model updates, without human assistance.

The proposal considers a multi-view classification scheme, wherein a pool of ML models are used, each model is built through a distinct set of features, namely view (such as those evaluated in Figure 1). The classification procedure starts with a networking event for classification, e.g., a set of network packets related to service occurred within a time interval. The network events' behavior is then extracted by a set of feature extraction modules, in which each module extracts a distinct feature set. The computed feature sets are forwarded for classification, where each classifier, with its own view, outputs a classification confidence value. The classification confidence values are used as a measure of classification correctness by the pool evaluator, which its goal is to only accept highly confident classifications as an attempt to maintain its reliability over time, even with outdated models. Therefore, only highly confident classifications, as output by distinct classifiers, each with a unique event view, is used for the event labeling process, thus, discarding possible outdated classifications and most likely errors.

As an attempt to provide up-to-date ML models, our proposal performs periodic autonomous model updates according to the assigned event label, as obtained by the most confident views. Therefore, a view that is currently producing low confident classifications can be reliably updated by other complementary views, which are outputting higher confidence values, and more likely accurate classifications, thus, autonomously and reliably updating the ML models over time. The next subsections further describes the proposed multi-view intrusion detection model, including the *classification* and *autonomous model update* modules.

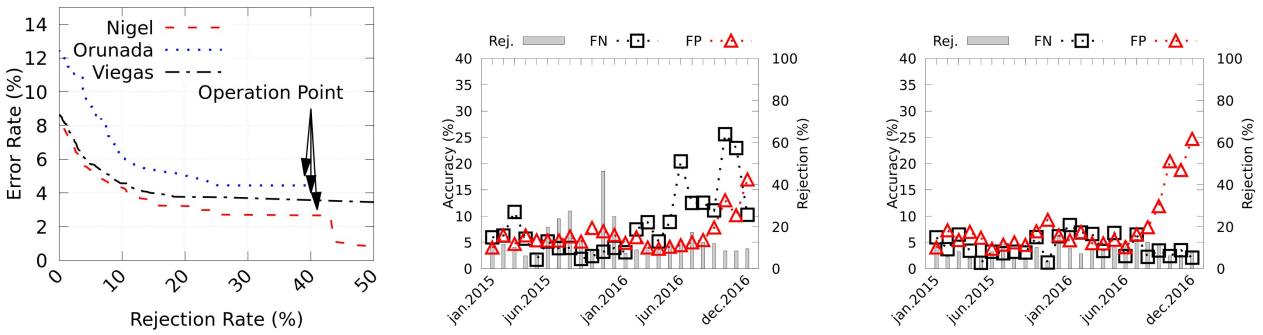
A. Reliable Multi-View Classification

A multi-view approach relies on two primary principles (i) consensus assumes that all views should maximize the agreement on multiple distinct views, while (ii)complementary assumes that each view may contain some knowledge that other views do not have. Our proposal explores such characteristics as an attempt to improve system reliability, even in the face of new network traffic. Our main assumption is that complementary views can improve one another over time, during both classification and autonomous model update.

The reliable multi-view classification approach is performed in a twofold manner. First, the behavior of the network event to be classified is extracted by a set of feature extraction pool, which outputs a set of feature sets, namely view (*Feature Sets*, Figure 2). The extracted feature sets are classified by a set of classifiers, where each model outputs a classification confidence value (*Confidence Values*, Figure 2). The classification confidence value is classifier agnostic, for instance, the Random Forest classifier outputs its classification confidence values according to the ratio of its base-learners that classified the instance as the assigned label. The confidence values are used by a pool evaluator module to identify confident classifications that should be used for the label assignment process. Identifying confident classifications should be defined according to the operator's needs, for instance, to improve reliability, one can use a higher confidence threshold despite an increase in the rejection rate. The rejection is measured as a ratio of events that did not meet the established confidence threshold and should have their alerts suppressed to maintain the system reliability over time. Finally, the class assigner establishes the event label through a majority voting procedure from the accepted classifications, i.e., classification views that met the used classification confidence threshold.

B. Autonomous Model Update

Regardless of the evaluation of the classification correctness performed during the classification task, the underlying ML models will become further unreliable as time passes. Therefore, an increase in the rejection rate will be experienced, caused by the lack of an updated ML model, as they will be unable to cope with the current environment behavior. However, perform periodic model updates is not an easily



(a) Error-reject tradeoff for each evaluated feature set (view) in February 2015.

(b) Accuracy and rejection rates over time without model updates..

(c) Accuracy and rejection rates over time with autonomous model updates.

Fig. 3: Proposed reliable multi-view intrusion detection obtained accuracy and rejection rates over time in MAWIFlow dataset.

achieved task, as it demands expert assistance to provide the proper event label. Our proposal performs autonomous model updates without human assistance by leveraging the multi-view pool to address such a challenge.

The update procedure is performed periodically, for instance, every semester. At each update task, the event database (Figure 2) is used for the pool update procedure. The event database is composed of the network events, autonomously labeled by the reliable multi-view classification pool (Section V-A). Thus, the event labels are assigned according to the most confident views and, consequently, are used to improve other less confident views at each model update. As a result, our proposal can improve the model views at each model update procedure, provide up-to-date ML models over time, and reliably and autonomously update the underlying ML models.

VI. Evaluation

The evaluation aims at answering the following research questions: ($Q1$) *Can the evaluation of the classification confidence value be used to improve the ML model accuracy?* ($Q2$) *Is the proposed pool evaluator able to maintain the system reliability over time, without model updates?* ($Q3$) *Is the proposed scheme able to update the ML model and remain reliable over time autonomously?*

A. Model Building

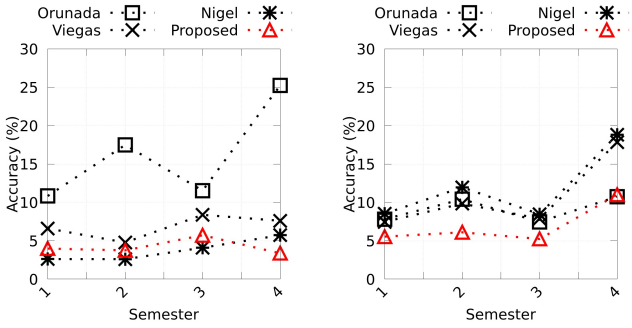
Our proposed reliable multi-view intrusion detection model was built, taking into account the three previously evaluated views (Figure 1). Therefore, three random forest classifiers were trained, each with a unique event view, using Nigel [8], Orunada [9], and Viegas [10] views, respectively. Similarly, the random forest classifiers were trained with 100 decision trees as their base-learners. The confidence values, used by our proposal to ensure reliability (Section V-A), was obtained through the scikit-learn API, as computed by the *predict_proba* API function. The API computes the random forest classifier's confidence values as the ratio of individual trees that classified the evaluated instance as the outputted class.

B. Multi-View for Long-term Classification Reliability

The first experiment aims at answering question $Q1$ and compares each view error-reject tradeoff. Each classifier is trained in January 2015 and evaluated in February 2015. The classification thresholds were defined through the Class-Related-Threshold (CRT) [7] approach. The computed confidence values for each individual view in February are evaluated to establish the error-reject tradeoff. It enables to define the set of thresholds that should be used throughout the remaining 2-year interval (confidence thresholds, Section V-A). For all evaluated views, shown in Figure 3a, one can see that it is possible to decrease the error rate with an increase in the rejection rate as a tradeoff. Therefore the confidence can be used to assess the classification quality. Besides, each view provides different tradeoffs, with the Viegas [10] feature set, providing the best relation between error and rejection.

The second experiment aims to answer question $Q2$ and apply the obtained classification thresholds in $Q1$ throughout the remainder 2-year interval. To achieve such a goal, each view classification threshold is set at a 40% rejection rate as measured in February 2015 (*Operation Point*, Figure 3a). The defined set of thresholds for each view is used throughout the remaining 2-year interval without model updates. Thus, if a classification in one view does not meet the defined class-related threshold, it is not used for the pool majority voting process, and if the classification outcome from all views also did not meet their thresholds, the event is rejected.

Figure 3b shows the proposal obtained accuracy and rejection rates over time without updates when applying the defined classification thresholds (Figure 3a) throughout *MAWIFlow* dataset 2-year interval. One can be noticed that despite using a 40% rejection rate operation point, the proposal is able to reject significantly fewer instances, rejecting an average of 14.2% instances in the dataset. Over time, the accuracy rates also remain further stable compared to their training period (January 2015), increasing their FP on average by only 2.1% while maintaining its FN rate stable in 2015. Thus, when compared to the traditional one-view approach (Figure 1), the proposed multi-view method is able to remain reliable



(a) FP rates over time. (b) FN rates over time.

Fig. 4: Proposed and traditional approaches error rates.

for more extended periods, increasing their FP on average by only 3.5% in the 2-year interval, in contrast to the one-view technique, that increases its FP by 9.2%, 6.2%, and 8.0% with the Nigel [8], Orunada [9] and Viegas [10] views respectively.

Finally, to answer question $Q3$, we autonomously update the underlying ML models over time, at a semester basis (6-month interval), according to the assigned event label, without human assistance. Figure 3c shows the proposal obtained accuracy and rejection rates with model updates in a 6-month interval basis when applying the defined classification thresholds (Figure 3a) throughout *MAWIFlow* dataset 2-year interval. The proposed scheme significantly decreases the rejection rate while also improving its accuracy, when autonomous model updates are performed, rejecting in average only 8.8%, in contrast to 14.2% if no model updates are performed while improving its accuracy by 4.3%.

Figure 4 shows a comparison of our proposed multi-view approach to a single-view one with respect to their accuracy in MAWIFlow dataset. The proposed model provided the lowest error rates over time, reaching at least, or even improving, the accuracy rates to the best view in each semester, without significantly decreasing its accuracy, as occurs in the single view approaches. On average, the proposed model maintained the same FN rate while only increasing its FP rates by only 4.8% throughout the 2-year interval. In contrast, the single view technique increased their FP by 9.2%, 6.2%, and 8.0% and also their FN by 0.1%, 11.9%, and 1.9% with the Nigel [8], Orunada [9] and Viegas [10] views respectively in the same interval.

VII. Conclusion

Despite the plethora of works that report highly accurate ML-based intrusion detection schemes, proposed techniques are hardly used in production. ML-based intrusion detection faces a broader range of challenges than those where it has been successfully applied, one of which is related to the network traffic behavior changes over time. This paper has proposed and evaluated a novel multi-view approach for autonomous and reliable model updates. Our proposed scheme reliably updated the underlying ML models by leveraging a multi-view and an evaluator technique. The evaluator assesses the classification quality to ensure that only highly confident classifications are accepted. On the other hand, the multi-view ensures that the system remains reliable over time by using each view to improve other complimentary views during both classification and model updates.

As future work, we will explore both view diversity and model diversity to provide reliable autonomous model updates.

References

[1] *IT threat evolution Q2 2020*, September 3, 2020. [Online]. Available: https://securelist.com/it-threat-evolution-q2-2020-pc-statistics/98292/

[2] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," in *2010 IEEE Symposium on Security and Privacy*. IEEE, 2010.

[3] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1153–1176, 2016.

[4] M. Ring *et al.*, "A survey of network-based intrusion detection data sets," *Computers & Security*, vol. 86, pp. 147–167, 2019.

[5] E. Viegas, A. Santin, A. Bessani, and N. Neves, "BigFlow: Real-time and reliable anomaly-based intrusion detection for high-speed networks," *Future Generation Computer Systems*, vol. 93, pp. 473–485, Apr. 2019.

[6] C. Gates and C. Taylor, "Challenging the anomaly detection paradigm: A provocative discussion," in *Proc. of the Workshop on New Security Paradigms (NSPW)*, 2006, pp. 21–29.

[7] E. Kugler, A. O. Santin, V. V. Cogo, and V. Abreu, "A reliable semi-supervised intrusion detection model: One year of network traffic anomalies," in *Int. Conf. on Comm. (IEEE ICC)*, 2020.

[8] N. Williams, S. Zander, and G. Armitage, "A preliminary performance comparison of five machine learning algorithms for practical ip traffic flow classification," *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 5, p. 5–16, Oct 2006.

[9] J. Dromard, G. Roudiere, and P. Owezarski, "Online and scalable unsupervised network anomaly detection method," *IEEE Trans. on Net. and Service Management*, vol. 14, no. 1, p. 34–47, Mar 2017.

[10] E. Viegas, A. O. Santin, A. Franca, R. Jasinski, V. A. Pedroni, and L. S. Oliveira, "Towards an energy-efficient anomaly-based intrusion detection engine for embedded systems," *IEEE Transactions on Computers*, vol. 66, no. 1, p. 163–177, Jan 2017.

[11] A. H. Mirza, "Computer network intrusion detection using various classifiers and ensemble learning," in *2018 26th Signal Processing and Communications Applications Conference (SIU)*. IEEE, May 2018.

[12] P. Singh and M. Venkatesan, "Hybrid approach for intrusion detection system," in *2018 International Conference on Current Trends towards Converging Technologies (ICCTCT)*. IEEE, Mar. 2018.

[13] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, Jul 2019.

[14] H. He, X. Sun, H. He, G. Zhao, L. He, and J. Ren, "A novel multimodal-sequential approach based on multi-view features for network intrusion detection," *IEEE Access*, vol. 7, pp. 183 207–183 221, 2019.

[15] W. Li, W. Meng, Z. Tan, and Y. Xiang, "Design of multi-view based email classification for IoT systems via semi-supervised learning," *Journal of Net. and Comp. App.*, vol. 128, pp. 56–63, Feb. 2019.

[16] J. Xiao, Z. Xiong, S. Wu, Y. Yi, H. Jin, and K. Hu, "Disk failure prediction in data centers via online learning," in *Proc. of the 47th International Conference on Parallel Processing*. ACM, Aug. 2018.

[17] J. E. Doak, M. R. Smith, and J. B. Ingram, "Self-updating models with error remediation," in *Art. Int. and Machine Learning for Multi-Domain Op. App. II*. SPIE, May 2020.

[18] MAWI, "MAWI Working Group Traffic Archive - Samplepoint F," 2020. [Online]. Available: https://mawi.wide.ad.jp/mawi/

[19] R. Fontugne, P. Borgnat, P. Abry, and K. Fukuda, "MAWILab: Combining diverse anomaly detectors for automated anomaly labeling and performance benchmarking," in *Proc. of the 6th Int. Conf. on emerging Networking EXperiments and Technologies (CoNEXT)*, 2010.