Contudo, as características inerentes a esses ambientes têm apresentado novos desafios para garantir um nível de segurança adequado. Intrusão baseado em Host , através da análise de system calls com machine learning em um cluster de orquestração de contêineres Kubernetes. O framework apresentado possibilita a desoneração dos nós do cluster do processamento voltado para a detecção de intrusão através de uma arquitetura distribuída e escalável. Operações de Segurança para tratar um eventual incidente de segurança.

A arquitetura proposta foi implementada no software GNS3, emulando um ambiente de rede corporativo para demonstrar a viabilidade de implementação do framework em um ambiente real.

## INTRODUÇÃO

Uma vez que os ambientes de nuvem são ambientes multi-tenant, apoiando-se em tecnologias como cgroups e namespaces para proporcionar o isolamento dos contêineres, a superfície de ataque das aplicações implantadas, aumentou substancialmente nesses ambientes [3]. Um dos recursos mais utilizados visando garantir um nível de segurança adequado aos ambientes computacionais modernos é a utilização de Sistemas de Detecção de Intrusão. A detecção de intrusão é o processo de monitoramento dos eventos que ocorrem em um sistema de computador ou rede, analisando-os em busca de indícios de possíveis incidentes [4]. Um HIDS analisa diversos recursos em um host como tráfego de rede, atividades no sistema de arquivos, logs de aplicação e system calls [4].

Em Sistemas Operacionais baseados em Unix, todas as aplicações que necessitam de recursos do SO devem fazer uso de system calls, por esse motivo HIDS baseados em system calls conseguem obter a melhor granularidade de dados [5]. Da mesma forma, contêineres Linux usualmente comunicam com o kernel do Sistema Operacional do host através de system calls [8]. HIDS baseados em anomalia inicialmente constroem um perfil normal de system calls utilizadas e processos que não se enquadram no perfil normal construído são considerados como intrusivos [6]. HIDS baseados em system calls têm ganhado atenção nos últimos vinte anos devido ao aumento crescente de ataques voltados para servidores Linux e têm sido desenvolvidos para detecção de intrusão em hosts virtuais e sistemas embarcados [7].

Neste trabalho, será realizada a proposição de um framework que possibilita a implementação de um HIDS baseado em anomalias de system calls, especificamente em ambientes que fazem uso de plataformas para execução de contêineres.

## De acordo com e , apesar dos avanços em termos de

IDS para máquinas virtuais, as contribuições para abordagens de sistemas de detecção voltados para contêiner são limitadas e poucos esforços têm sido direcionados para a área de segurança de contêineres em nuvem. Entre os trabalhos mais recentes sobre HIDS baseados em anomalias em system calls, muitas são as abordagens utilizadas para identificação de intrusões. A detecção de intrusão através de sequências de system calls foi apresentada em 1996 e ainda é utilizada em conjunto com técnicas de análise probabilísticas em trabalhos como [10], bem como redes neurais em [12]. Outros trabalhos como [5], [6], [8] e [13] fazem uso de abordagens baseadas na análise de frequência de system calls, e a utilização conjunta com redes neurais também é possível, conforme [14].

De acordo com [21], diversos algoritmos e técnicas de machine learning têm sido utilizados para detecção de anomalias. Outra limitação existente na área envolve a defasagem dos datasets de system calls disponíveis atualmente. No intuito de sanar essa lacuna, [9] propõe uma metodologia para geração de novos datasets que podem ser úteis no contexto da detecção

de intrusão. No que tange a cenários onde plataformas de orquestração de contêineres são utilizadas como ambientes de produção, poucos são os estudos de implementação de HIDS baseados em anomalias de system calls desenvolvidos até o momento.

No trabalho de [14], um framework para aprendizado distribuído foi desenvolvido visando a construção de modelos de detecção por aplicação através de redes neurais. Contudo, é sabido que o sistema implementado em cada nó da plataforma de contêineres gera uma sobrecarga que compete com a carga de trabalho real de aplicações e essa sobrecarga não foi considerada. Em [15], foi proposto um HIDS voltado para um cluster Kubernetes com detecção de anomalias através de redes neurais com aprendizado supervisionado e quatro categorias de system calls. Apesar do sistema ser capaz de monitorar os diversos nós do cluster e realizar a detecção em componente externo, as regras de filtragem a partir de um conjunto limitado de system calls podem restringir o escopo de detecção de ataques.

Restful para implementação do sistema de detecção de intrusão.

## FRAMEWORK PROPOSTO

O fluxo de dados do sistema de detecção compreende cinco camadas, onde ferramentas são utilizadas para propósitos específicos, conforme a Fig. [19] faz a leitura de cada nova linha adicionada em arquivo e as envia para o Redis [20], um banco de dados em memória externo ao cluster. Os índices no Elasticsearch contém os datasets com as system calls e as suas features já codificadas em formato numérico, próprio para o processamento feito pelos algoritmos de machine learning. Sua utilização como banco de dados para armazenamento das system calls permite que haja alta disponibilidade dos dados, além de garantir a consulta através de meta-dados úteis para filtragens, agregações, ordenações, etc. Além de conter os índices criados pelo Logstash representando os datasets, o Elasticsearch integra com o módulo de machine learning e também é utilizado para armazenar os resultados das análises de anomalias.

Para cada índice de dataset existente, após sua análise, um novo índice é criado no Elasticsearch com o escore de anomalia correspondente às janelas de system calls. As janelas de system calls consistem em uma estrutura da dados que armazenam sequências de system calls de múltiplos tamanhos e é uma abordagem comum para análise de system calls. Na camada de detecção de anomalias, o módulo de machine learning representado na arquitetura pode executar diversos algoritmos do estado da arte para detecção de anomalias. O requisito existente é que haja uma forma de integração entre a ferramenta utilizada e o Elasticsearch para leitura e gravação de dados.

Para ambientes baseados em Python, já existe uma biblioteca que faz a integração com o Elasticsearch.