Janela deslizante: O impacto do tamanho do traço no sistema de detecção de anomalias para recipientes através do aprendizado da máquina

Gabriel Ruschel Castanhel[1], **Tiago** Heinrich[1], **Fabr'Fabr´ıcio** Ceschin[1], **Carlos A.** Maziero[1][1]

Departamento de Informática
Universidade Federal do Paraná" (UFPR)

{grc15,theinrich,fjoceschin,maziero}@inf.ufpr.br

Abstrato. *A detecção de intrusão de anomalia no Sistema de Detecção de Intrusão baseado em Hospedeiro (HIDS) é um processo destinado a monitorar operações em um hospedeiro para identificar comportamentos que diferem de um comportamento "normal" do sistema. O HIDS baseado em chamadas de sistema usa traços de chamadas para representar o comportamento de um sistema. Devido ao volume de dados gerados pelas aplicações e pelo sistema operacional, janelas deslizantes são aplicadas a fim de avaliar um ambiente on-line, permitindo que intrusões sejam desativadas em tempo real enquanto ainda estão sendo executadas. O respectivo estudo explora o impacto que o tamanho da janela de observação tem sobre os algoritmos de uma classe de aprendizado de máquina (ML).*

1. Introdução

Um Sistema de Detecção de Intrusão (IDS) é responsável por realizar a identificação de tachas em um ambiente computadorizado. Existem diferentes técnicas usadas para detectar em trusão, algumas delas são baseadas em: (i) assinaturas, onde um conjunto de dados com ameaças conhecidas é usado para comparar novas assinaturas e, apesar de ser amplamente usado, ainda não é capaz de identificar ameaças que ainda não são classificadas (ataques desconhecidos); e (ii) comportamentos normais, onde o algoritmo de detecção de anomalias define um comportamento normal para o ambiente e novos comportamentos desconhecidos são classificados em duas classes: normal ou ano-maly [Yassin et al. 2013].

O Sistema de Detecção de Intrusão (HIDS) baseado em host tem o foco no host, com a capacidade de identificar ataques internos usando dados de um ou mais sistemas host [Liu et al. 2018]. Esta abordagem não se limita a um ambiente, poderia ser usada em uma infra-estrutura distribuída e poderia aproveitar os dados da rede para entender melhor o ambiente. Chamadas de sistema baseadas em HIDS exploram dados coletados de traços do sistema operacional ou aplicação individual, que podem variar de acordo com a abordagem [Liu et al. 2018]. A chamada do sistema é responsável por fazer uma interface entre o sistema operacional e o núcleo, cada chamada representa uma operação básica [Mitchell et al. 2001].

Uma variedade de estratégias pode ser usada para adaptar os dados aos modelos, uma das mais usadas é a técnica de janela deslizante, onde uma janela de tamanho n é usada para escanear os dados. Desta forma, um traço será dividido em traços menores, que podem ser adequados para detectar intrusões. Muitos pesquisadores já tentaram

definir o tamanho correto de uma janela de detecção, que pode variar de cinco a onze em muitos casos. Entretanto, como muitos parâmetros ML, o tamanho das janelas de detecção pode não ser sempre o mesmo, dado que uma escolha ótima de tamanho dependerá do ambiente e do tipo de dados [Liu et al. 2018].

Outro ponto a considerar é que muitos trabalhos na literatura consideram o traço completo para detectar uma intrusão (detecção offline). Esta abordagem, no entanto, não é real, dado que um ataque só será analisado após a sua execução. Assim, o IDS não funcionaria como esperado no mundo real, razão pela qual a técnica de janela deslizante é importante (detecção on-line). Em teoria, quanto menor o tamanho da janela, mais rapidamente uma anomalia poderia ser detectada, com o trade-off de que estas janelas podem cortar ações maliciosas pela metade, afetando sua detecção. Finalmente, selecionar um tamanho de janela ideal é importante para detectar ataques assim que eles começam a ser executados, sem a necessidade de observar o traço inteiro.

Com o objetivo de explorar o impacto do tamanho da janela em métodos ML de uma classe, nós - deixamos um conjunto de traços que foram avaliados com tamanhos diferentes, em um ambiente portuário onde, no melhor de nosso conhecimento, ainda não foi feito na literatura. A idéia principal é identificar como o tamanho de uma janela impactará o processo de identificação de uma anomalia em um ambiente, ajudando a entender como o classificador se comportará com esses dados.

Este trabalho está estruturado em cinco Seções. A Seção 2 apresenta trabalhos relacionados. A Seção 3 apresenta a proposta e os antecedentes para o estudo. A Seção 4 apresenta os resultados e a Seção 5 apresenta as conclusões.

2. Trabalho Relacionado

A literatura apresenta abordagens para detectar o tamanho de janela mais adequado para os conjuntos de dados UNM [Systems 1998] e ADFA-LD [Xie e Hu 2013], concluindo que um tamanho entre seis e sete era o melhor para eles. Considerando as máquinas virtuais, o tamanho de janela poderia diversificar entre seis e dez [Liu et al. 2018]. O tamanho da janela terá impacto no tamanho n-grama criado (um n-grama representará uma seqüência de chamadas de sistema extraídas do traço) e é responsável pela divisão do traço em grupos que serão usados para treinamento e testes.

A detecção de anomalias no host com chamadas ao sistema é uma área ativa na pesquisa de computadores. [Wang et al. 2006], por exemplo, apresenta o *Anagram*, um detector de conteúdo de n-gramas de alta ordem (n *>1*) projetado para detectar cargas úteis de pacotes de rede anômalos e "suspeitos". O trabalho fornece uma comparação entre abordagens usando diferentes tamanhos de n-gramas (ou tamanhos de janela). Os autores desvalorizaram que uma estratégia semi-supervisionada, utilizando o Bloom Filter, obteve 100% de detecção com taxas de 0,006% de falsos positivos em alguns casos.

[Forrest et al. 1996] introduziram o uso de chamadas de sistema para detecção de anomalias anos atrás, onde propuseram uma abordagem utilizando uma técnica baseada em um par de look-ahead, onde cada entrada no banco de dados representava uma chamada de sistema, e uma subseqüente imediata de chamadas de sistema em uma janela de tamanho n. A janela deslizante então se move por uma posição de cada vez para formar o banco de dados. Em seu trabalho prolongado, eles propuseram a abordagem Sequence Time Delay Embedding (STIDE), baseada na análise da seqüência curta das chamadas de sistema, das quais foi observado que as seqüências contíguas de comprimento fixo têm melhor poder discriminatório do que os pares de olhar para frente. Uma janela deslizante de tamanho fixo é usada para produzir as seqüências curtas de chamadas do sistema.

Em um ambiente de nuvem com foco em máquinas virtuais, a seqüência de

chamadas de sistema pode ser usada para a detecção de anomalias através de abordagens como "Bag of system calls" [Alarifi e Wolthusen 2012]. Os dados são coletados a partir de uma virtualização KVM

ambiente através de ferramentas de rastreamento. É utilizado um tamanho de janela de seis, com 11,1% de falsos positivos. Alguns problemas com o conjunto de dados poderiam ser levantados, pois o período de coleta de dados supõe que o comportamento normal não será afetado por atacantes.

3. Proposta

A literatura apresenta limitações para a detecção de anomalias em um ambiente de contêineres [Liu et al. 2018]. Tentamos explorar o impacto que uma visão parcial das informações poderia ter nos resultados. Observando como esses dados podem ser recuperados e aplicados nos métodos ML, nos concentramos em como o tamanho da janela terá impacto na detecção de anomalias. O recipiente é uma técnica de virtualização para aplicações e é muito popular hoje em dia por causa de ferramentas como o Docker. A virtualização acontece em nível de Sistema Operacional (OS), diminuindo a carga e a execução de aplicações que têm hardware gerenciado pelo contai- ner [Merkel 2014]. Isto é possível porque o container é responsável por reunir todos os componentes necessários em uma única imagem.

Nosso conjunto de dados[1] foi desenvolvida capturando dados de um ambiente de contêineres, observando que a coleta acontece do ponto de vista do sistema operacional. Desta forma, não há visão parcial e limitações para a captura de dados [Castanhel et al. 2020]. O contái- ner estava rodando WordPress 4.9.14, com Docker 19.03.11-ce, sob o Linux 5.4.44-1- MANJARO. Os dados escritos em disco consistem de chamadas e sinais emitidos pelo container, contendo os comportamentos normais do ambiente e comportamentos anômalos que exploram as vulnerabilidades da Remote Code Execution (RCE) [NVD 2020].

Isto define um conjunto de dados com comportamentos normais e anômalos que representam um ambiente de con- maquiagem. Estes dados são usados para treinar e testar dois métodos ML de uma classe. A Classificação de Uma Classe (OCC) aprende de apenas uma classe, diferente de outros métodos traditio- nal que visam aprender duas ou mais classes e pode ser difícil para o classificador considerando o conjunto de dados utilizado (em nosso caso, um conjunto de dados com muito desequilíbrio de classe: 367, 342 *chamadas de sistema* representando comportamentos normais e apenas 1, 628 *chamadas de sistema* representando ma- liciosos)[Zhang et al. 2015].

O processo de avaliação reflete como o crescimento do tamanho da janela irá contribuir para a detecção de ano-maly. O experimento consiste em dois casos de estudo. No primeiro caso, exploramos como o uso de uma pequena porção do traço afetará a detecção da anomalia. Este caso consiste em utilizar apenas 5% do tamanho do traço para treinar e testar os modelos. No segundo caso, o crescimento da janela não será limitado a apenas 5% do tamanho do traço, em vez disso, o crescimento acontecerá a cada vez com 5% do tamanho do traço até que todo o traço seja usado.

Considerando que o conjunto de dados tem um conjunto de traços com uma grande variedade de tamanhos e precisamos de um tamanho de janela fixo para que os métodos ML sejam treinados e testados, o menor tamanho de traço foi considerado para a avaliação do tamanho, ou seja, 100% dos traços representam um tamanho de traço de 1.628.

4. Experimentos

Os experimentos usaram dois tipos de conjunto de dados: um com dados brutos onde nenhuma técnica de filtragem é aplicada e toda a seqüência de chamadas do sistema é

usada sem nenhuma alteração, e outro onde as chamadas do sistema classificadas como de baixo nível de ameaça foram descartadas. As chamadas de sistema

[1] O conjunto de dados pode ser encontrado em: `https://github.com/gabrielruschel/hids-docker`.

A classificação foi baseada no trabalho [Bernaschi et al. 2002], e nos permite melhorar os resultados à medida que as chamadas de sistema inofensivas são removidas do conjunto de dados.

Considerando as duas abordagens de dados, a avaliação da janela visa explicar como janelas com uma pequena porção do traço terão impacto na identificação da anomalia e como seu crescimento impactará o classificador. Assim, foram definidos dois casos para treinamento e testes, que poderiam representar estes dois tipos de perspectiva de janela.

Para o menor tamanho de janela, foi utilizada uma porção entre zero e 5% do tamanho total do traço, com um crescimento de 0,5% do tamanho do traço. Esta investigação representará a maioria dos casos encontrados na literatura [Liu et al. 2018], considerando que 5% do tamanho do traço representará 81 chamadas ao sistema. Nosso objetivo aqui não é ficar limitado por este tamanho, mas verificar como um tamanho de traço maior se comportará na segunda abordagem considerando o crescimento da janela até o tamanho máximo do traço. O tamanho do traço crescerá 5% do tamanho total até atingir 100% do tamanho do traço.

A avaliação considerou dois algoritmos de OCC: Isolation Forest e One-Class SVM. Ambos são adaptações dos classificadores multiclasse Random Forest e SVM, respectivamente, treinados apenas sobre a classe majoritária (comportamento normal). Assim, estes classificadores geram limites de decisão que definem o que é normal, ou seja, cada amostra que está fora destes limites é considerada como uma anomalia. A floresta de isolamento é ligeiramente mais rápida de treinar do que a SVM de uma classe, dado que constrói um conjunto com um conjunto de árvores aleatórias, o que a torna mais adequada para aplicações reais, uma vez que poderia reagir mais rapidamente às mudanças [Liu et al. 2008]. Por outro lado, a SVM de uma classe estima os vetores de suporte (instâncias que estão próximas ao limite de decisão) dos dados de treinamento, gerando um hiperplano que contém o comportamento normal (que na prática leva muito mais tempo do que a geração de descanso aleatório) [Zhang et al. 2015].

A Figura 1 apresenta os resultados para o algoritmo Isolation Forest. Tanto os dados brutos quanto os dados do filtro atingem resultados razoáveis com janela pequena e variam sobre o tamanho do crescimento até atingir F1Score próximo a 100% depois que metade da porcentagem de traços é utilizada. A variação do tamanho da janela será responsável pela queda de 1% nos resultados, considerando que isto não é um grande impacto nos resultados, supomos que seria possível evitar este comportamento com um conjunto de dados mais complexo. Para os primeiros 5% de tamanho de janela, o crescimento instantâneo é mais claro na visualização parcial do gráfico, onde podemos ver que, com um tamanho de janela de 1% do tamanho do traço, já atingimos um F1Score de 95%.

Embora os dados do filtro sofram mais com o tamanho da janela de crescimento, ele é o primeiro a obter um F1Score acima de 97%, com um tamanho de janela de 1,5% do traço (isto representa um número de 24 chamadas de sistema). Os dados brutos não sofrerão o mesmo impacto com o crescimento, sendo mais estáveis com a mudança de tamanho.

A Figura 2 apresenta os resultados para o algoritmo SVM de uma classe. O gráfico mostra uma pequena variação entre os dados brutos e de filtro para um tamanho de janela com 10% do tamanho do traço, que impactará um F1Score entre 98% e 99%, nunca alcançando um F1score de 100%. Considerando os primeiros 5%, podemos ver

um alcance mais rápido para o F1score onde uma janela com menos de 0,5% do tamanho do traço já alcança um F1score de 98%.

Comparando ambos os resultados, é interessante apontar que uma pequena janela tende a atingir um F1score rapidamente, mas melhores resultados serão impactados pelo algoritmo usado e as abordagens ainda poderiam ter melhores resultados usando uma porção maior do traço em seu lugar

de janela pequena. Além disso, a SVM de uma classe tem um desempenho ligeiramente melhor do que a floresta de isolamento com traços menores, mas à medida que o tamanho da janela aumenta, a floresta de isolamento torna-se muito melhor, superando a SVM.

Em nosso caso de estudo, os dados brutos e filtrantes apresentam variações sobre o crescimento da janela, não mostrando uma diferença maior entre o uso de um tipo de dado específico. É possível apontar que a abordagem do filtro foi a primeira a obter um F1Score acima de 98% em ambas as avaliações e este tipo de dados poderia ajudar técnicas de pequenas janelas para detecção de anomalias.

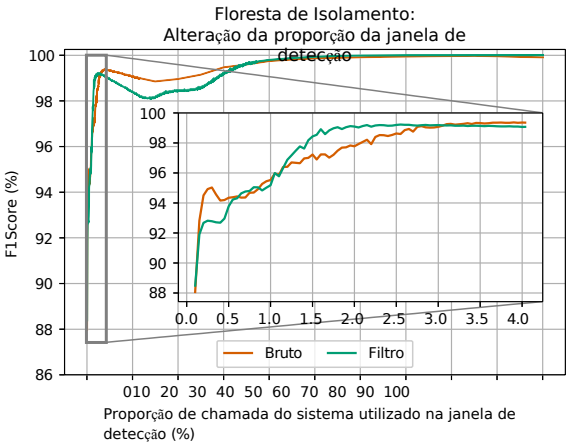

Figura 1. Crescimento da janela para o algoritmo Isola- tion Forest.

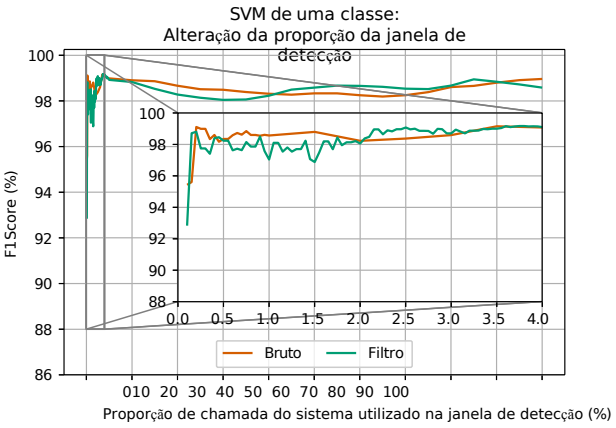

Figura 2. Crescimento da janela para o algoritmo SVM de uma classe.

5. Conclusão

O respectivo estudo se concentra no impacto da janela de tamanho em um ambiente portuário. Nossa abordagem visa esclarecer, através de um experimento de tamanho de janela de crescimento, como o tamanho terá impacto na Classificação de Uma Classe (OCC). Nossa contribuição é para o sistema de detecção de anomalias voltado para ambientes Docker, e ajuda com novos insights de segurança, devido à popularização dos containers na última década.

A avaliação One-Class presente nos permite compreender melhor o cenário de anomalia destécnica em contêineres. Com dois grupos distintos de dados, podemos afirmar que a menor janela é capaz de alcançar resultados razoáveis de F1Score, mesmo com dados filtrados, onde este resultado pode ser alcançado com janelas menores em comparação com dados brutos.

De acordo com os experimentos, é possível assumir que uma pequena janela de tamanho já apresenta um F1Score aceitável para detecção de anomalias e é mais apropriado para um sistema de detecção on-line. Mas o tamanho pode ser afetado pelo classificador utilizado para identificar anomalias.

Dois tópicos abertos que serão abordados em trabalhos futuros: (i) geração de dados, dado que nosso conjunto de dados é limitado a um comportamento de aplicação e esperamos desenvolver um conjunto de dados mais completo, com mais aplicações que criarão mais investigações reais, e (ii) explorar novos classificadores e técnicas de detecção de anomalias considerando o tamanho da janela mostrada nesta pesquisa.

Referências

[Alarifi e Wolthusen 2012] Alarifi, S. S. e Wolthusen, S. D. (2012). A detecção de anomalias - encontra-se em ambientes iaas através da análise de chamadas de sistemas host de máquinas virtuais. Em *2012 Conferência Internacional para Tecnologia da Internet e Transações Seguras*. IEEE.

[Bernaschi et al. 2002] Bernaschi, M., Gabrielli, E., e Mancini, L. V. (2002). Remus: um sistema operacional que aumenta a segurança. *ACM Transactions on Information and System Security (TISSEC)*.

[Castanhel et al. 2020] Castanhel, G. R., Heinrich, T., Ceschin, F., e Maziero, C. A. (2020). Detecc¸a˜o de anomalias: Estudo de tecnicas de identificac¸a˜o de ataques em um ambiente de te´cnicas. Workshop de Pesquisa de Graduação - Simpósio Brasileiro de Segurança (WTICG - SBSeg).

[Forrest et al. 1996] Forrest, S., Hofmeyr, S. A., Somayaji, A., e Longstaff, T. A. (1996). Um senso de si mesmo para processos unix. No *IEEE Symposium on Security and Privacy*.

[Liu et al. 2008] Liu, F. T., Ting, K. M., e Zhou, Z.-H. (2008). Floresta de isolamento. In *Proce- edings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, página 413-422, USA. IEEE Computer Society.

[Liu et al. 2018] Liu, M., Xue, Z., Xu, X., Zhong, C., e Chen, J. (2018). Sistema de detecção de trusão baseado em host com chamadas de sistema: Revisão e tendências futuras. *Pesquisas de computação ACM (CSUR)*.

[Merkel 2014] Merkel, D. (2014). Docker: contentores de linux leves para uma desdobramento e implantação consistentes. *Revista Linux*.

[Mitchell et al. 2001] Mitchell, M., Oldham, J., e Samuel, A. (2001). *Programação avançada de linux*. Publicação New Riders.

[NVD 2020] NVD (2020). Base de dados nacional de vulnerabilidade: Rce wordpress. `https://nvd.nist.gov/vuln/search/results?form_type=Basic&results_type=overview&query=RCE+wordpress&search_type=all`.

[Systems 1998] Systems, C. I. (1998). Sequence-based intrusion detection. `http://www. cs.unm.edu/~immsec/systemcalls.htm`.

[Wang et al. 2006] Wang, K., Parekh, J. J., e Stolfo, S. J. (2006). Anagrama: Um detector de anomalias de conteúdo resistente a ataques de mímica. Em *oficina internacional sobre os recentes avanços na detecção de intrusão*, páginas 226-248. Springer.

[Xie e Hu 2013] Xie, M. e Hu, J. (2013). Avaliação de sistemas de detecção de anomalias baseados em mangueiras: Uma análise preliminar da adfa-ld. Em *2013 6º Congresso Internacional sobre Processamento de Imagem e Sinal (CISP)*, volume 03, páginas 1711-1716.

[Yassin et al. 2013] Yassin, W., Udzir, N. I., Muda, Z., Sulaiman, M. N., et al. (2013). Detecção de intrusão com base em anomalias através de aglomeração de meios k e sificação de bayes de naives. Em *Proc. 4ª Int. Conf. Compu. Informática, ICOCI*, número 49.

[Zhang et al. 2015] Zhang, M., Xu, B., e Gong, J. (2015). Um modelo de detecção de anomalias baseado em svm de uma classe para detectar intrusões na rede.