Proposta para o uso de Contadores de Performance em *Hardware* para detecção de *Malware*

Bruno Dal Pontte[1], **Simone Dominico**[1], **Marco A. Z. Alves**[1]

[1]Departamento de Informática – Universidade Federal do Paraná (UFPR)

{bdp20,sdominico,mazalves}@inf.ufpr.br

Resumo. *Malware são programas maliciosos que causam danos em diferentes sistemas. Isso pode resultar em prejuízos aos usuários e organizações atacados. Nesse contexto, este artigo apresenta uma proposta de método de testagem para verificar até que ponto o não-determinismo do hardware e dos sistemas operacionais e a imprecisão nos eventos registrados pelos Hardware Performance Counters (HPC) podem impactar na detecção precisa de malware.*

1. Introdução

Ao longo dos anos, os ataques utilizando *softwares* maliciosos (*malware*) vêm aumentando consideravelmente. No segundo trimestre de 2020, estimou-se um aumento de 12% na detecção de novos *malware* em relação ao trimestre anterior, incluindo um aumento de 103% em *malware* que afetam aplicativos de escritório [Samani et al. 2020].

Uma forma de ataque é usar os recursos computacionais disponíveis para obter dados confidenciais do usuário ou de organizações, como senhas, dados bancários, entre outros. É crucial detectar e remover o *malware* para evitar danos e prejuízos. Para esse fim, existem os antivírus, que apesar do nome, detectam e removem não apenas vírus, mas também outras categorias de *malware*, como *worms* e *trojans* [Demme et al. 2013].

A detecção de *malware* pode ser realizada de duas formas principais. Na primeira forma, é feita uma análise estática, na qual é gerada uma assinatura a partir do código do programa por meio de um algoritmo, antes de ser executado. Essa assinatura é então verificada em uma base de dados de *malware* conhecidos. A principal vantagem desse método é permitir a detecção de *malware* sem a execução antecipada do programa. Uma das limitações é que a análise estática pode não ser capaz de detectar a presença de *malware* em variações do código, ou caso o formato do arquivo seja ambíguo [Jana and Shmatikov 2012].

A segunda forma é pela análise dinâmica. Esse método é capaz de analisar o comportamento do programa, além do código, com o objetivo de detectar e impedir possíveis danos causados por *malware* [Jacob et al. 2008]. No entanto, para aplicar técnicas tradicionais de análise de *malware*, é necessário que o antivírus intercepte chamadas de sistema e outras operações sensíveis, o que pode prejudicar o desempenho do computador. Como solução, a análise dinâmica é realizada em ambientes controlados, como *Datacenters*, que monitoram em tempo real o comportamento de códigos suspeitos.

Uma limitação dos antivírus é que eles são uma solução em *software*, o que traz consigo outras limitações inerentes. Por exemplo, isso possibilita que *malware* realizem modificações em seus componentes para evitar a detecção, incluindo *root-*

kits[1] [Singh et al. 2017]. Além disso, os antivírus podem ser desativados pelos usuários para a instalação de um determinado programa, abrindo espaço para a instalação de *malware* sem que o usuário perceba, criando assim uma brecha (*backdoor* [2]) de segurança.

No contexto de detecção de *malware*, o uso de *Hardware Performance Counter* (HPC) vem sendo investigado como uma forma mais eficiente e com custo reduzido em relação aos sistemas tradicionais de antivírus [Malone et al. 2011]. Isso ocorre porque os HPC estão muito mais próximos do processador do que o programa, permitindo uma análise dinâmica do comportamento do *software* sem a necessidade de interceptar suas operações. Este artigo descreve uma proposta que usa os HPC para a detecção de *malware*. Para isto, é proposto testar a eficácia e acurácia do uso de HPC, com o objetivo de verificar até que ponto as fontes de não-determinismo e imprecisão afetam os resultados da detecção de *malware*.

2. Trabalhos Correlatos

Os HPC têm sido incluídos em grande parte dos processadores de propósito geral desde a década de 1990 [Malone et al. 2011]. Ao longo dos anos, eles vêm sendo usados principalmente para análise de desempenho e performance de programas [Das et al. 2019].

Um dos primeiros artigos a tratar sobre o uso de HPC para outros fins, como avaliar o comportamento de programas e gargalos (*bottlenecks*) arquiteturais envolvidos foi o artigo de [Demme and Sethumadhavan 2011]. Nele é apresentada uma proposta mais genérica, que permite analisar o comportamento de programas paralelos em execução, mesmo que executem em várias *threads*. Essa possibilidade adicional de uso dos HPC foi a base para detecção de *malware* por meio de HPC.

O mesmo conceito de análise do comportamento de programas por meio dos HPC foi estendido e aplicado para detecção de *malware* em [Malone et al. 2011]. O método proposto pelos autores foi de verificação da integridade de um programa por meio de uma análise dinâmica realizada a partir dos HPC. Isso permitiu maior eficiência e rapidez na análise, não sendo mais necessário interceptar chamadas de sistema e outras operações. Outro ponto importante é que para a detecção de *malware* é usada a relação entre os contadores mensurados, permitindo que esse método detecte também variantes de *malware*.

Em [Demme et al. 2013], uma técnica similar foi testada, demostrando não só a eficiência da técnica, mas também a viabilidade dela para detectar o comportamento de *malware* e também de variações, incluindo *rootkits* e ataques de *side-channel*[3]. Embora esses últimos não sejam considerados *malware*, ainda representam vulnerabilidades de segurança. Assim como no trabalho anterior, é proposto que as informações de comportamento capturadas pelos HPC podem ser usadas para detectar *malware*, mesmo que possua pequenas modificações que não alteram significativamente o comportamento, mas que podem evadir os sistemas existentes de detecção dos antivírus.

Ainda assim, uma questão importante, mas nem sempre considerada no uso de HPC para fins de segurança, são as limitações na portabilidade e reprodutibilidade dos

[1] Tipos de *malware* que modificam o *kernel* do Sistema Operacional (SO) [Singh et al. 2017].

[2] Meios de acesso secreto, ou não facilmente detectável, em uma aplicação ou sistema que permitem burlar restrições de segurança.

[3] Ataques de *side-channel*, são ataques que exploram fontes de informações indiretamente relacionadas.

HPC em diferentes microarquiteturas. Em particular, existem duas categorias de HPC: eventos arquiteturais, que têm resultados consistentes entre as diferentes microarquiteturas dos processadores, e eventos não-arquiteturais (ou micro-arquiteturais), cujos resultados variam conforme a microarquitetura em questão [Das et al. 2019].

A falta de determinismo e imprecisões na contagem de eventos nos HPC em processadores reais da arquitetura x86-64 foi demonstrada em [Weaver et al. 2013]. Além disso, foram registradas variações na contagem entre as diferentes microarquiteturas de processadores, afetando quais dos HPC funcionam de forma determinística.

Em [Das et al. 2019], os autores notam que a maior parte das considerações sobre HPC fora da área de segurança não recomenda o seu uso por duas razões principais: em primeiro lugar, devido à falta de determinismo nos valores de desempenho contados, e em segundo lugar, devido à falta de portabilidade de eventos de HPC entre diferentes arquiteturas e microarquiteturas de processadores. Ainda assim, alguns erros e inconsistências nos eventos registrados pelos HPC no trabalho anterior já não estão mais presentes em processadores mais recentes. Isso indica que alterações microarquiteturais podem aumentar a confiabilidade dos resultados gerados pelos HPC.

3. Proposta

A proposta desse artigo consiste em buscar até que ponto a falta de determinismo dos HPC e do SO, além de imprecisões na contagem, afetam a acurácia dos eventos registrados pelos HPC e a assinatura do programa gerada a partir deles. Para esse fim, seria testado um mesmo programa, que executa um determinado padrão de operações, em diferentes máquinas e arquiteturas de processadores. Os HPC a serem mensurados seriam os que resultam em desempenho mais determinístico, para fins de reprodutibilidade e portabilidade entre diferentes arquiteturas. Portanto, seria dada preferência aos contadores arquiteturais. Segundo [Weaver et al. 2013], os contadores mais determinísticos e confiáveis nesse sentido são os contadores de instruções aposentadas.

Para evitar riscos de infecção das máquinas, o programa a ser testado não seria malicioso, o que permite que seja testado em máquinas reais sem perigo de infecção. O objetivo é testar, sob diferentes condições de execução, o quão similar essa assinatura de eventos permaneceria, e se, mesmo com as possíveis e eventuais diferenças, se as assinaturas geradas ainda seriam identificáveis como um tipo similar de programa.

Para gerar as assinaturas de eventos a partir dos HPC, seria usado o método proposto em [Malone et al. 2011] e o descrito em [Demme et al. 2013]. Os eventos seriam medidos com isolamento dos eventos de outros processos e sem esse isolamento, a fim de testar o quanto isso afeta a assinatura gerada. Caso as assinaturas variem significativamente sob as diferentes condições de execução e sob diferentes plataformas, seria realizada uma pesquisa para determinar quais configurações e condições de execução resultam na maior reprodutibilidade entre diferentes máquinas, arquiteturas e SOs.

Outro objetivo é testar diferentes variações do programa que apresentem o mesmo comportamento externo na execução, a fim de verificar a acurácia do uso de HPC para detecção de variantes de *malware*. Além disso, esta proposta pretende identificar as melhorias necessárias em *hardware* e *software* para que a contagem de eventos nos HPC seja mais precisa e reproduzível, evitando a contaminação dos eventos de desempenho de um processo em outro, ou, potencialmente, do *kernel* do SO.

4. Resultados e Discussões

Com o desenvolvimento e refinamento das arquiteturas, muitas das abordagens de detecção de *malware* por meio de HPC vistas até o momento vão crescer em confiabilidade e eficácia – e em alguns casos até mesmo em performance. Entretanto, aumentar a acurácia dos HPC muitas vezes implica em redução da performance arquitetural, o que deve ser levado em consideração ao buscar equilíbrio entre as duas partes. Dessa forma, torna-se necessário melhorias não só em *hardware*, mas também em *software* para que o uso de HPC para detecção de *malware* se torne mais eficaz, e, portanto, difundido. Assim, a proposta neste artigo busca esclarecer algumas das dúvidas não abordadas nos trabalhos anteriores em relação ao uso de HPC. Em particular, esclarecer como o não-determinismo e imprecisão na contagem de eventos podem dificultar o uso de HPC na área de segurança, especialmente para fins de detecção eficiente e precisa de *malware*.

Referências

Das, S., Werner, J., Antonakakis, M., Polychronakis, M., and Monrose, F. (2019). Sok: The challenges, pitfalls, and perils of using hardware performance counters for security. In *Proc. IEEE Symp. on Security and Privacy*, pages 20–38.

Demme, J., Maycock, M., Schmitz, J., Tang, A., Waksman, A., Sethumadhavan, S., and Stolfo, S. (2013). On the feasibility of online malware detection with performance counters. In *ISCA*, page 559–570. ACM.

Demme, J. and Sethumadhavan, S. (2011). Rapid identification of architectural bottlenecks via precise event counting. In *ISCA*, page 353–364. ACM.

Jacob, G., Debar, H., and Filiol, E. (2008). Behavioral detection of malware: from a survey towards an established taxonomy. In *JCV*, volume 4, pages 251–266. Springer.

Jana, S. and Shmatikov, V. (2012). Abusing file processing in malware detectors for fun and profit. In *Proc. IEEE Symp. on Security and Privacy*, pages 80–94.

Malone, C., Zahran, M., and Karri, R. (2011). Are hardware performance counters a cost effective way for integrity checking of programs. In *Proc. of the Sixth ACM Workshop on Scalable Trusted Computing*, page 71–76.

Samani, R., Beek, C., Chandana, S., Dunton, T., Grobman, S., Gupta, R., et al. (2020). Mcafee labs threats report, november 2020. Technical report, McAfee Labs.

Singh, B., Evtyushkin, D., Elwell, J., Riley, R., and Cervesato, I. (2017). On the detection of kernel-level rootkits using hardware performance counters. In *Proc. of the ACM on ASIA CCS*, page 483–493. ACM.

Weaver, V. M., Terpstra, D., and Moore, S. (2013). Non-determinism and overcount on modern hardware performance counter implementations. In *ISPASS*, pages 215–224.