Dizang : Uma solução para coleta de evidências forenses de ataques de injeção na nuvem

Hamilton J. S. Fonte II, Marcos A. Simplicio Jr.
Escola Politécnica, Universidade de São Paulo (USP) São Paulo, SP, Brasil
Email: hamiltonii@gmail.com, mjunior@larc.usp.br

***Resumo*—Arquiteturas em nuvem são cada vez mais comuns, e também o número de problemas de segurança envolvendo essa tecnologia. Infelizmente, devido à natureza volátil de recursos na nuvem, a coleta de evidências para análise forense nesse ambiente tem esbarrado em desafios práticos e legais. Este trabalho, com foco técnico, analisa propostas voltadas a resolver os desafios existentes na coleta evidências na nuvem, discute suas limitações, e então apresenta uma solução visando suplantá-las. Especificamente, a proposta tem como foco a reprodutibilidade do processo de coleta, sem com isso violar jurisdições ou a privacidade dos não envolvidos na investigação.**

I. Introdução

Técnicas de virtualização, replicação de serviços e compartilhamento de recursos entre múltiplos usuários (multi-inquilinato) dão a nuvens computacionais alta escalabilidade [1]. Porém, tais mecanismos também levam à volatilidade dos recursos virtuais que executam aplicações. Afinal, quando submetida a uma carga elevada, uma aplicação hospedada na nuvem pode criar clones das máquinas virtuais (*virtual machines* – VMs) que a hospedam e balancear a carga entre elas, buscando atender à demanda sem prejuízos na qualidade do serviço oferecido. Após esse pico, as VMs clonadas são normalmente desativadas, seus recursos liberados e o sistema retorna à capacidade anterior, evitando-se custos desnecessários.

Embora interessante do ponto de vista de eficiência e custos, do ponto de vista forense a volatilidade da nuvem traz problemas em caso de ataques. Por exemplo, caso uma das instâncias de processamento virtuais criadas temporariamente seja alvo de ameaças que atuam diretamente na sua memória, sem deixar rastros em discos (e.g., arquivos de *log*), as evidências desse evento podem ser completamente perdidas após elas serem desativadas e terem seus recursos liberados. Essa dificuldade é ainda agravada por aspectos como multi-inquilinato e multi-jurisdição típicas de soluções em nuvem [2]. Especificamente, o aspecto multi-inquilino dificulta a obtenção do *hardware* que executa as aplicações de interesse, pois, como ele é compartilhado por vários usuários, removê-los para análise poderia levar a uma violação de privacidade dos usuários não relacionados à investigação. Já a natureza distribuída da nuvem pode levar à alocação de informações relevantes à investigação em vários países, dificultando a obtenção das mesmas em especial quando não existem acordos de cooperação entre as entidades envolvidas [3]. Combinadas, tais características dificultam a coleta de evidências com a credibilidade necessária para que elas possam ser aceitas em processos legais, o que exige o respeito à privacidade, à jurisdição e à cadeia de custódia, bem como a reprodutibilidade do processo de coleta [4].

Embora existam soluções na literatura voltadas à coleta de informações na nuvem para a análise forense, a maioria delas aborda a coleta, o transporte e o armazenamento de forma isolada. Por exemplo, trabalhos como [5] e [6] tratam de fatores como multi-inquilinato e multi-jurisdição, discutindo formas de coleta e preservação da evidência fora da nuvem. Já estudos como [7] se concentram na análise forense para a coleta de evidência de VMs enquanto elas estão em execução, enquanto trabalhos como [8] tratam a garantia de cadeia de custódia durante o transporte da evidência. Por outro lado, não foram identificadas propostas que (1) descrevam como o dado é coletado e armazenado observando a cadeia de custódia, e (2) permitam que, mesmo que um recurso virtualizado seja desalocado, haja condições de se reproduzir o processo de coleta de evidências.

O presente trabalho visa suplantar tais limitações, apresentando uma proposta com foco: (1) na reprodutibilidade do processo de coleta, (2) no estabelecimento de vínculo entre a evidência coletada e sua origem, (3) na preservação da jurisdição e da privacidade dos não envolvidos na investigação. Em suma, a solução descrita provê uma forma de correlacionar evidências e sua origem virtual, permitindo transportar e armazenar tais dados de modo a preservar sua credibilidade e assim contribuir para sua aceitabilidade em um processo legal. Por fim, gestão de incidentes, que cada vez mais compartilha processos e ferramentas com a forense digital, pode se beneficiar de uma solução de coleta e preservação de evidências que esteja pronta na fase de *preparação ao incidente* e seja capaz de preservar as evidências para a fase *pós-incidente* permitindo assim que na fase de *detecção e análise* não seja necessário se preocupar com perda parcial ou total das mesmas. Para isso, supõe-se que o sistema sendo monitorado é executado dentro de um recurso em nuvem univocamente identificável. A solução tem como alvo específico ataques de injeção de código [9], pois estes, quando usados contra uma arquitetura em nuvem, não deixam rastros quando recursos de processamento virtuais são desativados e sua memória é liberada [10], [9].

Este documento está assim organizado. A Seção II discute brevemente soluções em nuvem e suas características. A Seção III analisa os trabalhos relacionados na área de forense de memória. A Seção IV detalha a solução proposta e avalia como ela trata os desafios alvo deste trabalho. A Seção V apresenta

as conclusões do estudo.

II. Adoção de arquiteturas em nuvem e contêineres

Uma nuvem computacional é um modelo de infraestrutura no qual recursos compartilhados em quantidade configurável, acessíveis via rede, são alocados e desalocados com esforço mínimo de gerenciamento por parte de um provedor de serviços. [11] Há três modelos principais de comercialização de uso da nuvem [11]: *software* como serviço (*Software as a Service* – SaaS), na qual se provê o *software* que será usado pelo cliente; plataforma como serviço (*Platform as a Service* – PaaS), na qual se provê o ambiente para que o cliente desenvolva, teste e execute seu *software*; e, o tipo mais pertinente para este trabalho, Infraestrutura como serviço (*Infrastructure as a Service* – IaaS), na qual são fornecidos recursos computacionais básicos, como processamento e memória, em geral de forma virtualizada.

A virtualização de recursos na nuvem, embora tradicionalmente feita por meio de VMs, vem sendo crescentemente feita também na forma de contêineres. De fato, segundo estudo realizado em 2016 com 235 empresas que têm desenvolvimento de software como sua atividade fim ou como suporte à atividade fim [12], 76% dos respondentes utilizam contêineres para melhorar a eficiência do processo de desenvolvimento e em suas arquiteturas de micro-serviços em nuvem. Diferentemente de VMs, que envolvem a criação de um *hardware* virtual e também de um SO acima do sistema nativo, a virtualização com contêineres é feita no nível do SO nativo, tem uma implementação mais simples eliminado camadas entre o aplicativo executado e o *hardware* físico. Uma tecnologia bastante utilizada para esse propósito são Contêineres Linux (LXC), que aproveitam-se de funcionalidades como cgroups e namespacing do kernel do Linux para auxiliar no gerenciamento e isolamento de recursos virtuais.

III. Trabalhos relacionados

Existem vários aspectos relativos à análise forense na nuvem, indo desde a coleta de informações até a garantia da cadeia de custódia de evidências. Para uma discussão mais estruturada dos trabalhos disponíveis na literatura sobre o tema, resumidos na Tabela I, a seguir eles são apresentados com base nos diferentes aspectos que abordam.

Tabela I
Comparativo de soluções de coleta de informações de memória de máquinas em nuvem para análise forense

	Coleta é contínua?	**Reproduz o processo sem a VM?**	**Garante cadeia de custódia?**	**Preserva jurisdição e privacidade?**
Dizang (esta proposta)	✓	✓	✓	✓
[7]	✘	✘	✘	✓
[13]	✘	✘	✘	✓
[5]	✘	✘	✘	✓
[14]	✘	✘	✘	✓
[6]	✘	✘	✓	✓
[8]	✓	✘	✓	✓
[15]	✘	✘	✘	✓
[16]	✘	✘	✘	✓
[17]	✓	✘	✘	✓
[18]	✓	✘	✓	✓

A. Acessar e coletar as informações de memória na nuvem

Diversos trabalhos de análise forense na nuvem se concentram na coleta de dados "após o fato", ou seja, após a intrusão ser detectada [6], [13], [5], [7], [8]. Os processos de coleta descritos nesses trabalhos podem ser iniciados de forma manual ou automaticamente, via integração com um mecanismo de detecção de intrusão. No caso específico de memória volátil, tal forma de coleta não consegue descrever como era a memória antes da intrusão, pois o processo só é acionado depois da detecção do ataque. Tal limitação pode trazer prejuízos à investigação, dado que algumas análises dependem exatamente da capacidade de se comparar dois momentos da memória [9]. Entre os trabalhos estudados, a única proposta encontrada que leva tal necessidade em consideração é [17], que propõe que o dado seja armazenado no próprio equipamento sob análise. Infelizmente, entretanto, a aplicação de tal abordagem no cenário em nuvem é pouco viável, pois pode levar à perda de informações importantes caso a VM ou contêiner seja desativada, tendo seus recursos liberados.

Existem ainda trabalhos voltados à coleta de informações durante a execução do sistema, nos quais os dados são constantemente coletados sem distinção do que aconteceu antes ou depois do fato de interesse. Esse é o caso de trabalhos como [13], [5], [8], que adotam a estratégia de isolar e parar a VM para em seguida realizar o processo de coleta. Embora interessantes, as abordagens descritas nesses trabalhos podem levar a um elevado volume de dados coletados, além de também não tratarem o cenário em que é necessário coletar evidências quando os recursos virtuais contendo tais informações são liberados.

B. Capacidade de reproduzir o processo e obter os mesmos resultados

Se, durante uma análise forense, analistas diferentes obtêm resultados distintos ao executar o mesmo procedimento de coleta, a evidência gerada não tem credibilidade podendo levar a sua não aceitação em um processo legal. Por essa razão, a reprodutibilidade do processo de coleta é uma parte importante da geração de evidências para análise forense. Infelizmente, entretanto, nenhuma das propostas encontradas na literatura atualmente permite tal reprodutibilidade em cenários de nuvem em que VMs ou contêineres são desativados e seus recursos físicos liberados: todas elas dependem da existência do recurso virtual para a repetição do processo de coleta.

C. Não violar privacidade ou jurisdição das partes não envolvidas na investigação

Em um ambiente de nuvem pública, remover o *hardware* para análise posterior pode levar à violação de privacidade de usuários, uma vez que o multi-inquilinato desse cenário faz com que uma mesma máquina física guarde informações de diversos clientes, alguns dos quais podem não estar envolvidos na investigação em curso. Diversos trabalhos na literatura tratam esse problema adequadamente, por meio das duas estratégias principais: a primeira, adotada em [6], [7], [13], [5], consiste em coletar dados pertinentes à investigação e armazená-los fora da nuvem; a segunda, empregada em [8] e que constitui um caso específico de [7], depende da cooperação do provedor de serviços de nuvem para conseguir as informações necessárias à investigação. Depender do provedor de serviços de nuvem é uma estratégia pouco recomendada, entretanto, pois (1) o volume de dados de usuários pode forçar os provedores a limitar o tamanho dos *logs* armazenados, e (2) caso ocorra uma indisponibilidade causada por um ataque, o objetivo do provedor será o de restabelecer o serviço, não necessariamente o de preservar evidências[19].

IV. Solução proposta: Dizang

A presente proposta tem como objetivo principal coletar memória de recursos computacionais virtuais de modo a conseguir: (1) identificar a fonte da evidência, mesmo se o recurso virtual não existir mais; (2) descrever o sistema antes e depois do incidente; (3) transportar e armazenar a memória coletada de uma forma que garanta sua integridade e confidencialidade; e (4) não violar a jurisdição e a privacidade de outros usuários que porventura tenham recursos alocados no mesmo servidor físico. A solução aqui apresentada, denominada Dizang , é descrita em detalhes a seguir.

A. Descrição

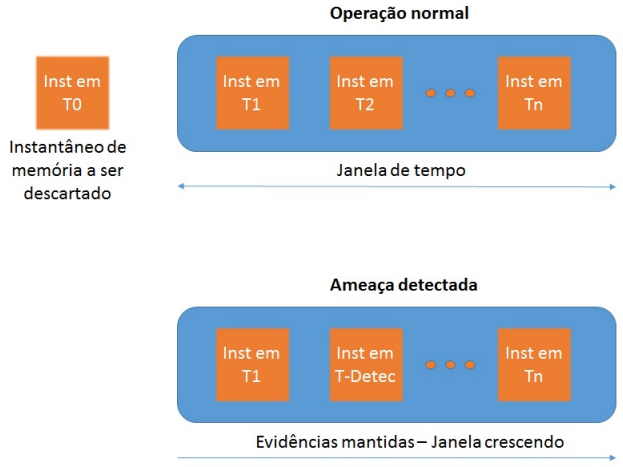
Em sistemas computacionais executados sobre uma infraestrutura física (i.e., não virtualizada), pode-se fazer uma associação direta entre um recurso qualquer, como uma informação da memória, imagem de disco ou pacotes trafegando na rede, e sua origem correspondente. Já em sistemas construídos sobre uma infraestrutura virtual, em especial quando esta é auto-escalável, os recursos computacionais são altamente voláteis e, portanto, podem ser desalocados a qualquer momento. Para conseguir correlacionar uma evidência a sua origem volátil, esta implementação faz uso de contêineres linux (LXC) para persistir a relação fonte-evidência. Embora um contêiner seja um *software* e, portanto, também volátil, cada imagem compilada e sua execução na forma de contêiner são normalmente atrelados a um *hash* que identifica univocamente essa relação. O contêiner também permite identificar com mais precisão a fonte de uma evidência, uma vez que é possível dividir as partes de um sistema em contêineres: por exemplo, um contêiner para o motor de páginas dinâmicas (e.g., Apache), outro com a lógica de negócios (e.g., *golang*) e um terceiro para um banco de dados (e.g., *Cassandra*).

A cópia de memória não é uma atividade atômica, pois ela é executada em conjunto com outros processos. Portanto, caso um desses processos seja um código malicioso apagando traços de sua existência da memória do contêiner, informações possivelmente importantes para a investigação podem acabar sendo perdidas. Com o objetivo de deixar o processo de cópia da memória mais atômico, Dizang interrompe temporariamente a execução do contêiner, realiza a cópia de sua memória, e em seguida retoma sua execução. Essa técnica, que é semelhante àquela adotada em [20] para VMs, produz um instantâneo da memória volátil do contêiner; isso permite sua análise em um estado de repouso, ou seja, sem a necessidade de ter o contêiner em execução. Ao realizar a coleta em intervalos de tempo adequados, é possível construir um histórico do estado da memória durante a execução no contêiner.

A maioria das técnicas forenses mais usadas atualmente são voltadas à obtenção da informação em sua totalidade, seja via cópia bit a bit, seja por meio da obtenção do *hardware* físico [21] [22]. Embora tais técnicas possam parecer interessantes à primeira vista, elas muitas vezes acabam sendo responsáveis por um problema: o crescente volume de informações que os investigadores precisam analisar [23]. Para mitigar essa dificuldade, em Dizang são adotadas duas estratégias: a primeira é a definição de um volume de dados que possa ser considerado *suficiente* para a realização de uma investigação; a segunda é a definição de uma *idade máxima* para a evidência enquanto o sistema trabalha em condições normais, isto é, quando não está sob ataque. Para detectar e analisar intrusões na memória de processos, é necessário ter uma cópia da memória antes e depois da intrusão [9]. Assim, a solução proposta implementa uma janela de instantâneos de memória cobrindo um intervalo de tempo pré-definido, como ilustrado na Fig. 1. Em condições normais de operação, as evidências são coletadas com certa periodicidade e coletas que atingem uma determinada idade são descartadas. Em contraste, após a detecção de um evento de ataque (e.g., por um sistema de detecção de intrusões), Dizang deixa de descartar as coletas mais antigas do *log* de monitoramento, sendo possível conhecer o sistema antes e depois do ataque e, assim, avaliar sua evolução.

Para persistir a relação evidência-origem e garantir a integridade da mesma, a presente proposta calcula o hash do par [evidência, identificador da imagem do contêiner] e armazena a tripla [hash, identificador da imagem do contêiner,

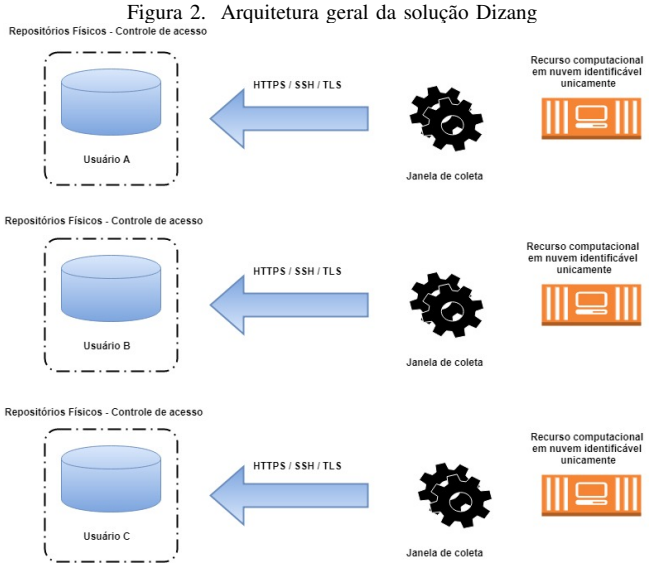
Figura 1. Janela deslizante de coleta de evidência



evidência]. Para evitar eventuais problemas com o armazenamento desses dados em países com jurisdições diferentes daquelas que devem ser aplicadas na investigação em questão, as evidências coletadas são armazenadas em um local físico fora da nuvem, após serem transportadas por meio de um canal seguro (e.g., via *Transport Layer Security* – TLS [24]).

B. Implementação

Figura 2. Arquitetura geral da solução Dizang



Os métodos propostos foram implementados em uma plataforma de testes visando avaliar a eficácia de Dizang em coletar as informações de memória dos contêineres de forma reprodutível, sem violar jurisdições ou a privacidade de usuários. A solução, ilustrada na Fig. 2, consistiu criação de uma instancia *t2.micro* na zona Ohio da AWS com 3.3Mhz, 1Gb de RAM e sistema operacional de 64 bits. Nesta instância AWS foi instalado o Docker Engine 1.10 e a API Docker 1.21, com os quais foram criados 3 contêineres executando o nginx 1.0 em diferentes portas. Usando uma aplicação Java que descobre o identificador de processo associado a cada contêiner, pode-se copiar conteúdo do *descritor de alocação de memória não uniforme* (**/proc/pid/numa_maps**), o qual contém a alocação das páginas de memória, os nós que estão associados a essas páginas, o que está alocado e suas respectivas políticas de acesso [25]. A cópia e gravação do arquivo é tal que, a cada minuto, a aplicação (1) pausa o contêiner em questão, (2) copia a diretório **numa_maps**, (3) concatena os dados obtidos com o *hash* de identificação da imagem do contêiner, (4) calcula o *hash* do conjunto e (5) salva o resultado em um arquivo cujo nome é o identificador da imagem do contêiner e a extensão é **.mem**. O transporte seguro da evidência para um armazenamento físico fora da AWS foi implementado usando uma instância *t2.micro* na zona Ohio da AWS onde foi instalado um servidor *OpenVPN*. A instância EC2 que contém as evidências foi configurada para aceitar conexões apenas de máquinas na VPN. Uma máquina fisica fora da AWS, usou o cliente do *OpenVPN* para estabelecer uma conexão VPN com a instância que contém as evidências e as transportou para o disco da máquina física. Após a conclusão do processo de transporte, a máquina física verifica se existem arquivos **.mem** em disco mais antigos que um certo intervalo de tempo "t", descartando-os.

C. Resultados experimentais

Para avaliar a efetividade de Dizang na coleta de evidências, alguns experimentos foram realizados usando o ambiente implementado (descrito na Seção IV-B). Primeiramente, o sistema foi configurado para realizar coletas de memória em intervalos de 1 minuto, salvá-las em disco em uma máquina física fora da nuvem e apagar amostras coletadas há mais de 5 minutos. O sistema foi então executado por 30 minutos, tempo durante o qual foram coletadas como métricas (1) o uso de espaço em disco utilizado pelos instantâneos de memória salvos e (2) o tempo de pausa no contêiner necessário para a cópia das mesmas e (3) o tempo o tempo de transporte da evidência para a máquina física fora da nuvem. A cada coleta, foi executado o comando `du -sh *.mem` do *Unix* no disco de armazenamento físico fora da nuvem, para retornar a lista dos arquivos onde os instantâneos de memória foram armazenados e o espaço em disco ocupado pelos mesmos. Ao fim do experimento, os contêineres foram removidos.

A ocupação em disco devido aos instantâneos de memória capturados durante o experimento é mostrada no gráfico da Fig. 3. O gráfico mostra que o aumento do uso do espaço em disco é linear e o crescimento se interrompe quando é atingido o limite de tempo configurado para a janela, pois as coletas com tempo de vida maior que tal limite são apagadas do disco. Assim, que a solução mantém sob controle o espaço em disco ocupado pelas amostras coletadas. Ao mesmo tempo, instantâneos de memória salvos pela solução depois que os contêineres e a instância AWS são removidos continuam no disco da máquina física fora da nuvem, podendo ser associados a sua origem, conforme esperado para uma análise forense. Essa capacidade se mantém após a detecção de uma ameaça, pois nesse caso coletas mais antigas deixam de ser apagadas.

Logo, é possível descrever o estado do sistema antes e depois do incidente [9], permitindo-se, por exemplo, que ataques de injeção de código em memória sejam analisados.

Uma potencial limitação da solução proposta é que a pausa de um contêiner para coleta de dados poder, em princípio, causar perdas no desempenho da aplicação sendo executada. Para avaliar esse impacto, durante o experimento foram medidos os tempos de cópia da memória do contêiner. Os resultados são mostrados no gráfico da Fig. 4. É possível notar que, após a inicialização da aplicação, o tempo para realizar a cópia é bastante reduzido, variando entre 20 e 40 milisegundos. Em especial, para contêineres executando um motor de páginas web dinâmicas, como é o caso do experimento em questão, essa latência deve ser pouco perceptível por usuários finais.

Outra preocupação levantada está relacionado ao tempo de transporte das evidências para o armazenamento fora da nuvem. O transporte da evidência pode levar mais tempo que o intervalo de geração levando a perdas da evidências durante o transporte. Para avaliar esse impacto, durante o experimento foram medidos os tempos de transporte das evidências para o armazenamento fora da nuvem. Os resultados são mostrados no gráfico da Fig. 5. É possível notar que o tempo de transporte estabiliza após atingido o tamanho da janela. O tempo de transporte da evidência fica, em média proximo dos 30 segundos. Este tempo pode diminuir escolhendo-se uma topologia de rede diferente. Neste experimento o gerador de evidências estava na América do Norte enquanto que a máquina física para onde as evidências foram transportadas estava na América do Sul.

D. Limitações

Como a solução descrita tem como foco coletar informações de memória no espaço do usuário (*user space*), ela não consegue acessar o espaço de kernel (*kernel space*). Assim, Dizang em princípio não provê suporte a técnicas de investigação de malware que se baseiam em informações do *kernel space*, como, por exemplo, a comparação de informações do bloco do ambiente do processo (*Process Environment Block – PEB*), que ficam no *user space*, com informações do descritor de endereços de memória virtual (*Virtual Address Descriptor – VAD*), que fica no *kernel space*. Análise de ameaças que realizam manipulação direta dos objetos do kernel (D.K.O.M.– *Direct Kernel Object Manipulation*) também não se beneficiam com a solução aqui proposta.

V. Considerações Finais

Ameaças digitais que atuam diretamente na memória de sistema não costumam deixar rastros em disco após terem os recursos correspondentes desalocados, dificultando análises forenses posteriores. Esse problema é especialmente notável em sistemas de computação em nuvem, nos quais a alocação e desalocação de recursos virtualizados (e.g., VMs e contêineres) é frequente. Essa característica, aliada a aspectos como multi-inquilinato e multi-jurisdição de nuvens computacionais, dificulta a coleta de evidências para a investigação de incidentes.

Nesse cenário, a proposta apresentada visa relacionar o instantâneo de memória a sua origem, utilizando o *hash* calculado da imagem do contêiner como identificador da evidência armazenada. Para evitar uso excessivo de memória, a quantidade de dados armazenados usa uma janela de armazenamento, o que permite descrever a memória antes e depois de um ataque (e.g., de injeção de memória). Combinada com uma ferramenta para identificação de ameaças, essas características de Dizang o transformam em uma solução poderosa para prover evidências e, assim, viabilizar análises forenses na nuvem.

Como trabalho futuro, pretende-se estudar o impacto provocado pela pausa do contêiner no desempenho de um sistema em produção e realizar uma análise da memória dos contêiners deste mesmo sistema utilizando as coletas feitas por Dizang . Com esses resultados pretende-se implementar uma ferramenta mais flexível, que não precise ter acesso a memória completa da máquina, requisito comum de ferramentas de análise de malware atuais (e.g., FROST [5] e Volatility Framework [26]).

Referências

[1] A. M. Morsy, J. Grundy, and I. Muller, "An Analysis of the Cloud Computing Security Problem," in *APSEC Cloud Workshop*. Sydney, Australia: Cornell University, 2010. [Online]. Available: https://arxiv.org/abs/1609.01107

[2] P. Gilbert and S. Sujeet, *Advances in Digital Forensics IV*, 1st ed. Orlando: Springer-US, 2008, vol. 1.

[3] J. Dykstra and A. Sherman, "Acquiring forensic evidence from infrastructure-as-a-service cloud computing: Exploring and evaluating tools, trust, and techniques," *Digital Investigation*, vol. 9, pp. S90–S98, 2012, (Proc. of the 12th Annual DFRWS Conference). [Online]. Available: dx.doi.org/10.1016/j.diin.2012.05.001

[4] S. Rahman and M. N. A. Khan, "Review of live forensic analysis techniques," *International Journal of Hybrid Information Technology*, vol. 8, no. 2, pp. 379–388, 2015. [Online]. Available: www.sersc.org/journals/IJHIT/

[5] J. Dykstra and A. T. Sherman, "Design and implementation of FROST: Digital forensic tools for the OpenStack cloud computing platform," *Digital Investigation*, vol. 10, pp. S87–S95, 2013, (Proc. of 13th Annual DFRWS Conference). [Online]. Available: dx.doi.org/10.1016/j.diin.2013.06.010

[6] Z. Reichert, K. Richards, and K. Yoshigoe, "Automated forensic data acquisition in the cloud," *IEEE Int. Conf. on Mobile Ad Hoc and Sensor Systems*, pp. 725–730, 2015.

[7] S. George, H. Venter, and F. Thomas, "Digital Forensic Framework for a Cloud Environment," in *IST Africa*. Tanzania: IIMC, 2012, pp. 1–8.

[8] T. Sang, "A log-based approach to make digital forensics easier on cloud computing," *Intelligent System Design and Engineering Applications (ISDEA)*, pp. 91–94, 2013.

[9] A. Case, M. Ligh, L. Jamie, and A. Walters, *The Art of Memory Forensics: Detecting malware and threats in Windows, Linux and Mac memory*. Hoboken, NJ: Wiley, 2014.

[10] S. Vömel and J. Stüttgen, "An evaluation platform for forensic memory acquisition software," *Digit. Investig.*, vol. 10, pp. S30–S40, 2013, elsevier Science Publishers. [Online]. Available: http://dx.doi.org/10.1016/j.diin.2013.06.004

[11] P. Mell and T. Grance, "The NIST definition of cloud computing," NIST SP 800-145, p. 7, 2011. [Online]. Available: csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf

[12] DevOps and ClusterHQ, "Container market adoption survey 2016," https://clusterhq.com/assets/pdfs/state-of-container-usage-june-2016.pdf, 2016.

[13] R. Poisel, E. Malzer, and S. Tjoa, "Evidence and cloud computing: The virtual machine introspection approach," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, vol. 4, no. 1, pp. 135–152, 2013. [Online]. Available: citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.469.937

[14] D. Barbara, "Desafios da perícia forense em um ambiente de computação nas nuvens," Univ. do Planalto Catarinense, Tech. Rep., 2014, revista. uniplac.net/ojs/index.php/tc_si/article/view/1911.

[15] B. Dolan-Gavitt, T. Leek, M. Zhivich, J. Giffin, and W. Lee, "Virtuoso: Narrowing the semantic gap in virtual machine introspection," in *IEEE Symposium on Security and Privacy*. Plymouth, UK: IEEE, May 2011, pp. 297–312.

[16] A. Aljaedi, D. Lindskog, P. Zavarsky, R. Ruhl, and F. Almari, "Comparative analysis of volatile memory forensics: Live response vs. memory imaging," in *IEEE 3rd Int. Conf. on Privacy, Security, Risk and Trust*, 2011, pp. 1253–1258.

[17] F. Dezfouli, A. Dehghantanha, R. Mahmoud, N. Sani, and S. Shamsuddin, "Volatile memory acquisition using backup for forensic investigation," in *Int. Conf. on Cyber Security, Cyber Warfare and Digital Forensic (CyberSec)*. Plymouth, UK: IEEE, June 2012, pp. 186–189.

[18] R. B. van Baar, H. M. A. van Beek, and E. J. van Eijk, "Digital Forensics as a Service: A game changer," *Digital Investigation*, vol. 11, pp. S54–S62, 2014. [Online]. Available: dx.doi.org/10.1016/j.diin.2014.03.007

[19] S. Alqahtany, N. Clarke, S. Furnell, and C. Reich, "Cloud forensics: A review of challenges, solutions and open problems," in *Int. Conference on Cloud Computing (ICCC)*. Plymouth, UK: IEEE, April 2015, pp. 1–9.

[20] M. Rafique and M. N. A. Khan, "Exploring Static and Live Digital Forensics: Methods, Practices and Tools," *IJSER*, vol. 4, no. 10, pp. 1048–1056, 2013. [Online]. Available: www.ijser.org/researchpaper{\% }5CExploring-Static-and-Live-Digital-Forensic-Methods-Practices-and-Tools.pdf

[21] S. Simou, C. Kalloniatis, E. Kavakli, and S. Gritzalis, "Cloud forensics: Identifying the major issues and challenges," in *Advanced Information Systems Engineering (CAiSE 2014)*, vol. 8484. Cham, CH: Springer International Publishing Switzerland 2014, 2014, pp. 271–284.

[22] D. Bem, F. Feld, E. Huebner, and O. Bem, "Computer forensics - past , present and future," *Journal of Information Science and Technology*, vol. 5, no. 3, pp. 43–59, 2008.

[23] D. Quick and K. K. R. Choo, "Impacts of increasing volume of digital forensic data: A survey and future research challenges," *Digital Investigation*, vol. 11, no. 4, pp. 273–294, 2014. [Online]. Available: dx.doi.org/10.1016/j.diin.2014.09.002

[24] R. E. Dierks T, "The Transport Layer Security (TLS) Protocol," IETF: https://tools.ietf.org/html/rfc5246, Fremont, CA, 2008.

[25] Unix Man Pages, "Numa Maps - Non Uniform Memory Architecture," man7.org/linux/man-pages/man7/numa.7.html, acessado em: 24-06-2017.

[26] Volatility Foundation, "Volatility Framework," www. volatilityfoundation.org/, 2014, acessado em: 24-06-2017.

Figura 3. Evolução do uso do espaço em disco com o Dizang .

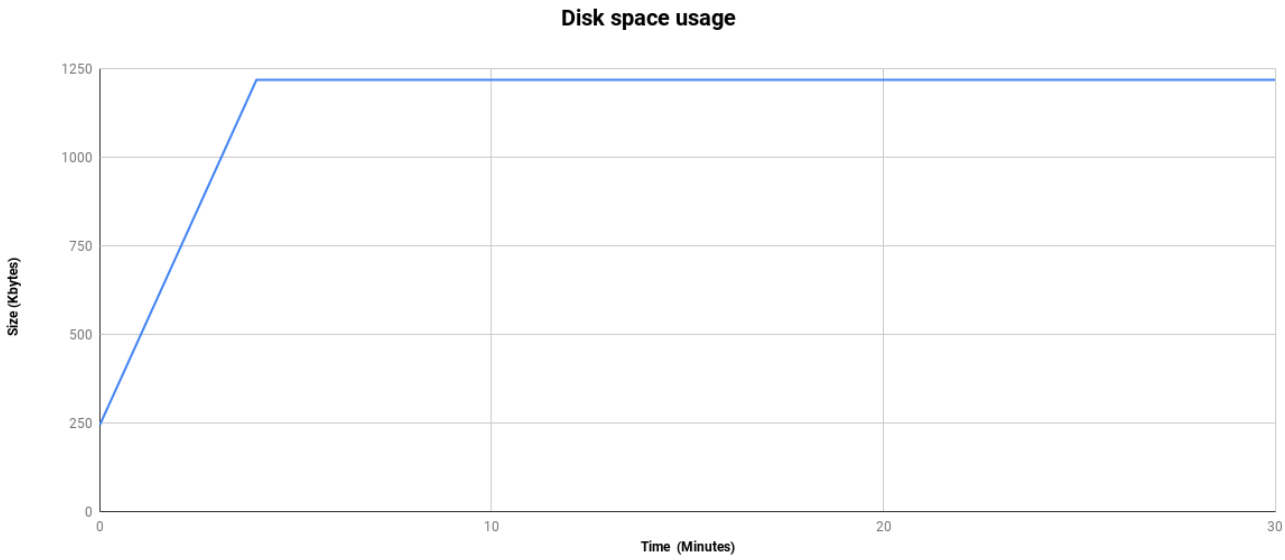


Figura 4. Tempo de cópia da memória de um contêiner

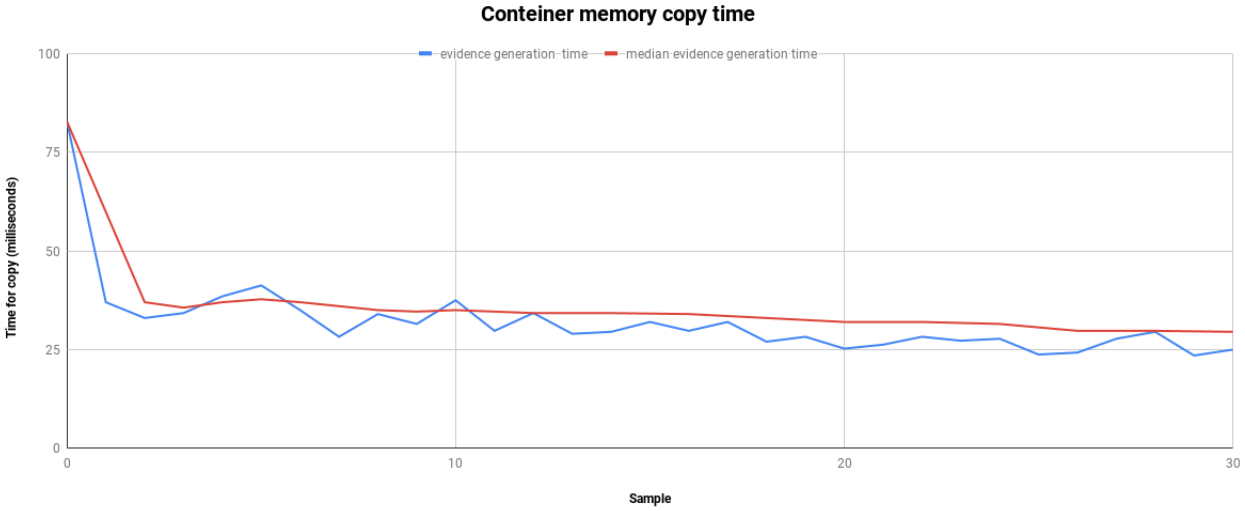


Figura 5. Tempo de cópia da memória de um contêiner