Uma análise de segurança no uso de contêineres do tipo Docker em nuvens IaaS OpenStack

Kerolayne de Souza Vieira de Oliveira[1], **Charles Christian Miers**[1]

[1] Departamento de Ciência da Computação (DCC)
Universidade do estado de Santa Catarina (UDESC)

kerolayne.oliveira@edu.udesc.br, charles.miers@udesc.br

Resumo. *Nos últimos anos, o uso da virtualização baseada em nuvens cresceu substancialmente. As soluções de virtualização empregando hipervisores e contêineres foram amplamente usadas em combinação com os ambientes de PaaS e IaaS para obter maior desempenho, isolamento e escalabilidade. Entretanto, a adoção de contêineres pelo mercado é recente quando comparada a máquinas virtuais (MVs). Como consequência, a preocupação com a segurança da conteinerização é relevante. Vulnerabilidades nos mecanismos do núcleo e nas soluções do contêiner são um risco para os dados e o funcionamento do serviço de todas as entidades que compartilham o mesmo hardware. Portanto, é necessário identificar e classificar ameaças, riscos e vulnerabilidades para a segurança do contêiner. Este artigo foca na definição do problema de pesquisa para realizar uma análise de segurança em contêineres Docker em nuvens OpenStack.*

1. Introdução

A virtualização é uma técnica amplamente utilizada por plataformas do tipo *Platform-as-a-Service* (PaaS) e *Infrastructure-as-a-Service* (IaaS) a fim de provisionar um ambiente isolado, seguro e escalável para usuários / organizações [Bui 2015]. A virtualização de recursos computacionais, tipicamente, passou a ser proporcionada através de duas abordagens principais: contêiner e hipervisor. Com o surgimento do Docker em 2013, a conteinerização teve uma considerável adoção por desenvolvedores e organizações [DOCKER 2016]. Seu uso combinado com MVs proporciona um considerável grau de isolamento em aplicações na nuvem.

O presente trabalho tem como objetivo analisar a segurança do uso de contêineres para a computação em nuvem, especificamente na plataforma IaaS baseadas em OpenStack. A relevância da análise contribui para os esforços da comunidade em explorar as principais vulnerabilidades, riscos e ameaças da tecnologia de conteinerização *Docker*, quando utilizado com a solução de nuvem computacional *OpenStack*. O artigo está organizado da seguinte maneira: a Seção 2 trata da computação em nuvem, contêineres e segurança de contêineres. A Seção 3 aborda brevemente o OpenStack, com suas configurações de contêiner. A Seção 4 descreve os critérios de análise e resultados iniciais.

2. Computação em nuvem & Contêineres

A computação em nuvem é um modelo para permitir acesso ubíquo, conveniente e sob demanda via rede a um agrupamento compartilhado e configurável de recursos computacionais que pode ser rapidamente fornecido e liberado com esforços mínimos de gerenciamento ou interação com o provedor de serviços [Jansen and Grance 2011]. A virtualização é uma tecnologia presente nas mais modernas infraestruturas de computação

em nuvem, *e.g.,* Amazon EC2 e Google App Engine. A virtualização proporciona para a computação em nuvem três características relevantes: elasticidade, escalabilidade e segurança [NCC GROUP 2016]. Evitando a sobrecarga de camadas de abstração adicionais, os contêineres são capazes de desempenho quase nativo e podem apresentar melhor desempenho que os sistemas baseados em MV em quase todos os aspectos [Gao et al. 2017]. Contudo, como qualquer outra tecnologia, existem preocupações quanto a sua segurança.

O princípio da virtualização baseada em contêineres é utilizar os recursos do núcleo para criar um ambiente isolado para processos. Ao contrário da virtualização baseada em hipervisor, contêineres não abstraem o hardware do hospedeiro [Eder 2016]. Deste modo, tanto contêineres quanto MVs são considerados técnicas de virtualização, contudo, são implementadas de formas diferentes. No contexto de IaaS, é possível utilizar contêineres como substituto de MVs através de soluções como LXD que atuam como um hipervisor para contêineres. Além disso, contêineres e MVs são tecnologias complementares. É possível executar contêineres sob MVs, garantindo um nível a mais de isolamento entre componentes de uma aplicação [Bernstein 2014]. Apesar do sucesso dos serviços de contêiner, sempre existem as preocupações sobre segurança e privacidade para a execução de vários contêineres, presumivelmente pertencentes a inquilinos diferentes, no mesmo núcleo do sistema operacional [Gao et al. 2017]. [NCC GROUP 2016] enfatiza as questões de segurança das soluções existentes, responsáveis por reduzir a superfície de ataques à tecnologias de contêineres como Docker e LXC. Os principais recursos compõem a virtualização baseada em contêineres: *Namespaces* do núcleo, *Control groups*, capacidades, *pivot_root* e Controle de acesso mandatório.

3. OpenStack

No contexto de computação em nuvem, o OpenStack é uma plataforma que com frequência incorpora novas tecnologias em seu núcleo. O uso de contêineres recebeu forte adoção pela plataforma. Desde 2014, a comunidade do OpenStack incorporou o uso de contêineres em sua arquitetura através de ferramentas e serviços que permitem a experimentação de tecnologias no estado da arte [OPENSTACK.ORG 2015] e possibilita que usuários possam gerenciar todo o ciclo de vida do contêiner do mesmo modo que MVs. A arquitetura do OpenStack é constituída por módulos que permitem que tecnologias como a conteinerização possam ser implantadas. O uso de contêineres como serviço é possível através da API *Magnum*, havendo duas formas principais de conteinerização completa na plataforma: via API *Magnum* e modo manual (Figura 1).

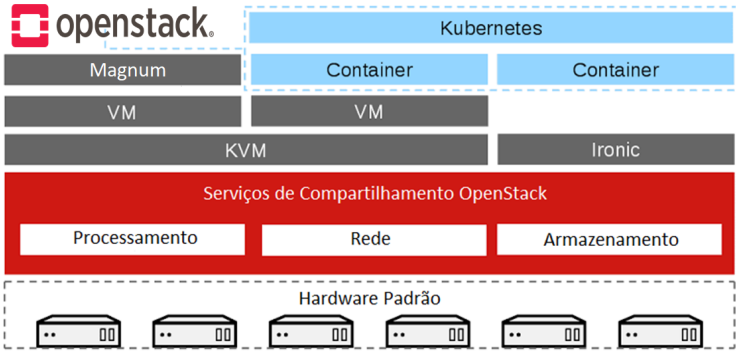

Figura 1. Organização mais comum de contêineres e MVs no OpenStack.

A API *Magnum* disponibiliza a orquestração de contêineres Docker e *engines* de contêineres (*e.g.,* Docker Swarm, Kubernetes e Apache Mesos) para a plataforma OpenS-

tack. Na questão de isolamento, *Magnum* provê o mesmo nível de segurança que o módulo *Nova* oferece na execução de MVs pertencentes a diferentes inquilinos sob um mesmo nó de computação [OPENSTACK SECURITY 2016]. Isso é garantido porque a solução utiliza instâncias do *Nova* para compor *bays* (*clusters*). Esses *clusters* são isolados entre os inquilinos, garantindo que contêineres de diferentes *clusters* não serão executados no mesmo núcleo.

O gerenciamento dos recursos é feito pela API do *Magnum* que envia requisições para *Heat* que por sua vez cria *clusters* e aloca os recursos como volumes de armazenamento (*Cinder*) para as instâncias do *Nova*. Ainda, o *Heat* realiza a orquestração da imagens de SOs que contém o Docker e a *engine* do contêiner para executar as imagens em máquinas virtuais ou até diretamente sobre o *bare metal* numa configuração em *cluster*. Além disso, o *Magnum* provisiona através do *Neutron* recursos de rede (*e.g.,* IP) aos contêineres, permitindo a comunicação entre contêineres sobre um mesmo *cluster*.

4. Critérios de análise & Resultados iniciais

Com base nas referências analisadas, constata-se a relevância em identificar e classificar as preocupações e mecanismos de segurança para contêineres. De acordo com as preocupações levantadas, são definidos três critérios:

1. Controle e limitação de recursos: Analisar os mecanismos de controle de acesso, de limitação de recursos e capacidades que garantem o isolamento e controle de acessos à recursos entre o núcleo do sistema operacional (SO) e contêineres vizinhos. Uma motivação para o critério é a ativação por padrão desses mecanismos na inicialização do contêiner de acordo com [DOCKER 2017].
2. Segurança das imagens de contêineres: Verificar a existência de vulnerabilidades em imagens de contêineres hospedadas em repositórios da comunidade e por fim, recomendar soluções para auditar tais vulnerabilidades que já são conhecidas e catalogadas por banco de dados de vulnerabilidades como CVE e NVD. Mais de 80% das imagens do repositório *Docker Hub* possuem pelo menos uma vulnerabilidade de severidade alta, de acordo o índice de risco da CVE [Shu et al. 2017].
3. Segurança dos dados na comunicação entre contêineres: Analisar a segurança na comunicação entre contêineres que utilizam o mesmo núcleo. Contêineres devem possuir um perímetro de segurança para proteger, detectar e prevenir contra ataques como também qualquer outra camada de rede virtual utilizada [CSA 2017].

Com base na experimentação realizada, é possível observar várias potenciais vulnerabilidades e ameaças que podem ser exploradas no contêiner. Na experimentação das políticas de segurança do *AppArmor* foi possível verificar que o Docker possui por padrão um *profile* as regras que bloqueiam e permitem ações como a montagem de unidades e a modificação de arquivos do sistema, tal que configurações incorretas podem acarretar no mal funcionamento do contêiner. Esse é um caso similar ao da experimentação da filtragem de chamadas com *Seccomp*.

Em relação às restrições de capacidades do núcleo, é notável que a capacidade *CAP_SYS_ADMIN* é a capacidade que mais concede privilégios para o contêiner, o resultado de sua concessão permitiu com que fosse criado um novo *namespace* do usuário com privilégios administrativos. Logo, para garantir um nível considerável de segurança basicamente é necessário tomar cuidado com as três capacidades mais impactantes:

NET_SYS_ADMIN, SYS_ADMIN e SYS_MODULE. Caso a aplicação ou serviço que executará sob o contêiner não necessitar delas, é recomendável sua remoção.

Relativo a comunicação entre contêineres é observável o impacto do uso da rede do tipo *bridge* com a interface *docker0*. Afinal, com apenas algum pouco esforço já foi possível capturar o tráfego entre contêineres sob a mesma rede, sem contar que por ser uma configuração padrão do contêiner acaba aumentando a superfície de ataque. Assim, criar contêineres em redes isoladas é uma prática de segurança que deve ser seguida.

5. Considerações & Trabalhos futuros

Os critérios escolhidos para a experimentação foram definidos pensando nos guias de segurança, na abordagem comumente utilizada pela comunidade e nos diversos pontos de integração entre contêineres Docker e a plataforma de nuvem computacional OpenStack. Dessa forma, os critérios definidos foram: controle e limitação de recursos, segurança na imagem de contêineres e a comunicação segura entre contêineres. Com base nesses critérios, a análise reforçou a preocupação quanto a segurança da conteinerização no ambiente de produção. Como próximos passos, está sendo realizada a análise usando o serviço *Magnum* do OpenStack.

Agradecimentos: Os autores agradecem o apoio do LabP2D/UDESC e a FAPESC.

Referências

Bernstein, D. (2014). Containers and cloud: From lxc to docker to kubernetes. *IEEE Cloud Computing*, 1(3):81–84.

Bui, T. (2015). Analysis of docker security. *CoRR*, abs/1501.02967.

CSA (2017). Security guidance for critical areas of focus in cloud computing v4.0. technical report, Cloud Security Alliance.

DOCKER (2016). Modern app architecture for the enterprise.

DOCKER (2017). Docker security.

Eder, M. (2016). Hypervisor- vs. container-based virtualization. *Network Architectures and Services*, pages 11–17.

Gao, X., Gu, Z., Kayaalp, M., Pendarakis, D., and Wang, H. (2017). Containerleaks: Emerging security threats of information leakages in container clouds. In *2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 237–248.

Jansen, W. and Grance, T. (2011). Sp 800-144. guidelines on security and privacy in public cloud computing. Technical report, Gaithersburg, MD, United States.

NCC GROUP (2016). Understanding and hardening linux containers. technical report, NCC Group.

OPENSTACK SECURITY (2016). Exploring opportunities: Containers and openstack. technical report, OpenStack.

OPENSTACK.ORG (2015). Exploring opportunities: Containers and openstack.

Shu, R., Gu, X., and Enck, W. (2017). A study of security vulnerabilities on docker hub. In *Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy*, CODASPY '17, pages 269–280, New York, NY, USA. ACM.