Propuesta de proyecto: Análisis de seguridad de Docker

Maximiliano Osorio
mosorio@inf.utfsm.cl

Valparaíso, 20 de abril de 2015

1. Descripción del tema

1.1. Conceptos y motivación

Durante los años el uso de tecnologías de virtualización han aumentado rápidamente, esta tecnología permite particionar el sistema de un computador en múltiples ambientes virtuales además ofrecen beneficios importantes que han llevado a que la virtualización sea muy utilizada [1]. Uno de los usos comunes para esta tecnología es la virtualización de servidores en *datacenters*. Con la virtualización de servidores, un administrador puede crear una o mas instancias virtuales o máquinas virtuales (VMs) en un servidor, hoy en día se utiliza comúnmente en *datacenters* y también en plataformas de *cloud* como Amazon EC2, RackSpace, Dreamhost y otros [3]. El crecimiento del uso de la virtualización ha hecho necesario la búsqueda de una solución que permita tener un ambiente escalable y seguro. Un gran numero de soluciones han nacido en el mercado y se pueden clasificar en dos tipos: *container-based virtualization* y *hypervisor-based*.

Container-based virtualization es una virtualización liviana a nivel de software usando el kernel de host para correr múltiples ambientes. Estos ambientes son nombrados con *containers* (contenedores). Hoy en día Linux-VServer, OpenVZ, libcontainer y Linux Container (LXC) son las principales implementaciones para utilizar los contenedores. En a figura 1 se puede observar que la arquitectura de *container-based virtualization* que trabaja a nivel de sistema operativo por lo tanto no es necesario cargar n veces el sistema operativo por n containers. Para el sistema operativo *host* el *container* es un proceso más que corre encima del kernel, gracias al kernel y otras herramientas proveen un ambiente aislado con los recursos necesarios para ejecutar las aplicaciones en el *container*. [5]

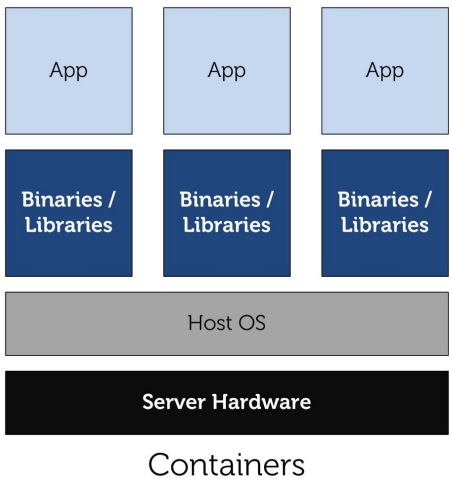

Figura 1: container-based virtualization.

Por otra parte *hypervisor-based* establece una *virtual machine* encima del sistema operativo, cada máquina virtual no solamente incluye la aplicación y las dependencias sino además incluye el sistema operativo completo con otro kernel separado.

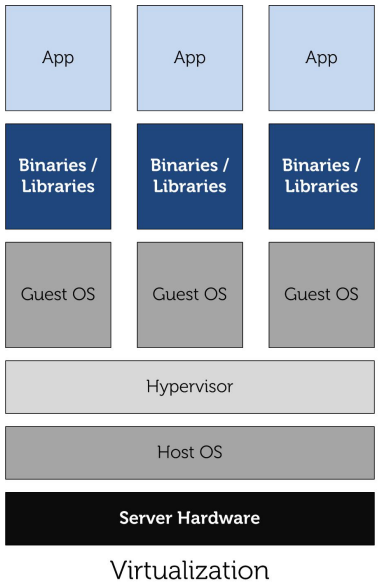

Figura 2: hypervisor-based virtualization.

Las diferencias en la arquitectura presenta ventajas interesantes para *container-based virtualization* como una mayor densidad de ambientes virtuales en un host debido a que no debe cargar el sistema operativo y puede compartir los binarios y librerías con otros containers en el mismo host. Segundo se ha demostrado que la *container virtualization* es capaz de ser más liviana y eficiente [6] [7] [3]. A partir de esto nace la motivación de estudiar más a fondo soluciones que utilicen los containers y la seguridad de estos.

1.2. Docker

Docker es un proyecto open-source que utiliza la tecnología de los containers (libcontainer) para "construir, migrar y correr aplicaciones distribuidas". Actualmente utilizado por Yelp, Spotify, entre otros [2] [4] Docker es una solución que simplifica el uso de los containers que han estado presente durante más de una década y está divido en dos componentes: Docker engine y Docker Hub.
Docker engine es una herramienta liviana y portable para manejar *container-based virtualization* utilizando la arquitectura de la figura . Los containers corren encima del servicio de Docker que se encarga de ejecutar y manejar los containers. Docker Client, provee una interfaz para interactuar con los containers con los usuarios a través de RESTful APIs. [1]

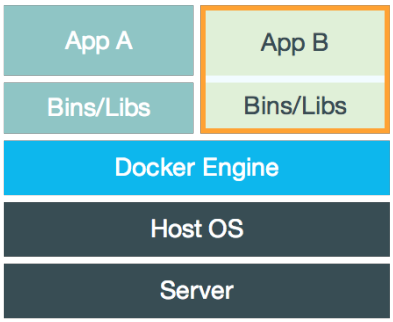

Figura 3: Docker

Docker Hub es un repositorio central donde se pueden compartir y obtener imágenes, estas imágenes pueden ser verificadas tanto en la autenticidad y integridad a través un firmado y verificación de datos.[1]

2. Propuesta y objetivos

2.1. Propuesta

Como se vio en el estado de arte, *container-based virtualization* presenta mejor rendimiento que *hypervisor-based virtualization* pero las soluciones basadas en *hypervisors* presentan un mejor aislamiento debido a que VM no puede comunicarse con el kernel de host, solo puede comunicar con su mismo kernel. Por otra parte *container-based virtualization* necesita comunicarse con el kernel del host, por lo tanto permite que un atacante puede atacar al host u otro container.
Para analizar la seguridad de Docker se creará una ambiente donde existe n containers que viven en un nodo donde k containers han sido vulnerados, el atacante que mantiene el poder sobre los k *containers*, el atacante intentará hacer un ataque de tipo de denegación de servicios y/o un escalamiento de permisos. Luego se analizará otros posibles ataques.

Los aspectos a analizar serán: *process isolation*, *filesystem isolation*, *device isolation*, *IPC isolation*, *network isolation* y *limiting of resources*. Por otra parte se buscará comparar estos ataques en una máquina virtual corriendo sobre un *hypervisor* de tipo 2 (KVM) con el fin de comparar los *containers* y las VM.
Como métricas de para definir la seguridad se evaluará: el tiempo necesitado para realizar el ataque y la dificultad de este, factibilidad de detectar el ataque y detenerlo.

2.2. Objetivos

- Objetivo general: Analizar el nivel de seguridad de Docker.
- Objetivos específicos:
  - Investigar, comprender y documentar sobre la arquitectura, funcionalidades y otras características de Docker en función al análisis.
  - Establecer un ambiente de trabajo de Docker en la versión 1.5.1 o posterior.
  - Diseñar y/o evaluar ataques acorde a Docker.
  - Evaluar el aislamiento respecto a proceso, filesystem, device, IPC, network y limite de recursos.
  - Comparar ataques conocidos a máquinas virtuales sobre *containers* y vice-versa.

3. Plan de trabajo

El plan de trabajo se dividirá en dos fase:

- Primera fase (hasta el 26 de abril): Implementación del ambiente de prueba, recolección de información y pruebas para posible ataque.
- Segunda fase (hasta el 15 de junio): Ejecución de pruebas y recolección de resultados.

4. Recursos

Se utiliza un nodo de prueba prestado por la empresa Linets.

Referencias

[1] Thanh Bui. Analysis of docker security. *arXiv preprint arXiv:1501.02967*, 2015.

[2] Docker. Use cases, 2015.

[3] Wes Felter, Alexandre Ferreira, Ram Rajamony, and Juan Rubio. An updated performance comparison of virtual machines and linux containers. *technology*, 28:32, 2014.

[4] Victor Marmol, Rohit Jnagal, and Tim Hockin. Networking in containers and container clusters.

[5] Dirk Merkel. Docker: lightweight linux containers for consistent development and deployment. *Linux Journal*, 2014(239):2, 2014.

[6] Pradeep Padala, Xiaoyun Zhu, Zhikui Wang, Sharad Singhal, Kang G Shin, et al. Performance evaluation of virtualization technologies for server consolidation. *HP Labs Tec. Report*, 2007.

[7] Nathan Regola and J-C Ducom. Recommendations for virtualization technologies in high performance computing. In *Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on*, pages 409–416. IEEE, 2010.