ANALISIS KEAMANAN SISTEM INTERNAL DOCKER DARI *DOCKER ESCAPES CONTAINER ATTACK* DAN *DOCKER DAEMON ATTACK*

SKRIPSI

Dibuat untuk Melengkapi Syarat-Syarat yang Diperlukan

untuk Memperoleh Gelar Sarjana Terapan

Kevin Harada

4817050202

PROGRAM STUDI TEKNIK MULTIMEDIA DAN JARINGAN

JURUSAN TEKNIK INFORMATIKA DAN KOMPUTER

POLITEKNIK NEGERI JAKARTA

2021

HALAMAN PERNYATAAN ORISINALITAS

Skripsi ini adalah hasil karya saya sendiri, dan semua sumber baik yang dikutip maupun dirujuk telah saya nyatakan dengan benar.

Nama : Kevin Harada

NIM : 4817050202

Tanggal : Rabu, 16 Juni 2021

Tanda Tangan :

HALAMAN PENGESAHAN

Skripsi diajukan oleh:

Nama : Kevin Harada

NIM : 4817050202

Program Studi : Teknik Multimedia dan Jaringan

Judul Skripsi : Analisis Keamanan Sistem Internal Docker dari Docker *Escapes Container* Attack dan *Docker Daemon Attack*

Telah diuji oleh tim penguji dalam Sidang Skripsi pada Hari Rabu, Tanggal 16, Bulan Juni Tahun 2021 dan dinyatakan **LULUS.**

Disahkan oleh

Pembimbing : Fachroni Arbi Murad, S.Kom., M.Kom.

Penguji 1 : Defiana Arnaldy, S.Tp., M.Si.

Penguji 2 : Asep Kurniawan, S.Pd., M.Kom.

Penguji 3 : Ade Rahma Yuly, S.Kom., M.Ds.

Mengetahui:

Jurusan Teknik Informatika dan Komputer

Ketua

Mauldy Laya, S.Kom., M.Kom.

NIP. 197802112009121003

KATA PENGANTAR

Puji Syukur saya panjatkan kepada Tuhan Yang Maha Esa, karena atas berkat dan rahmat-Nya, penulis dapat menyelesaikan laporan skripsi ini. Penulisan laporan skripsi ini dilakukan dalam rangka memenuhi salah satu syarat untuk mencapai gelar Sarjana Terapan Politeknik. Penulis menyadari bahwa, tanpa bantuan dan bimbingan dari berbagai pihak, dari masa perkuliahan sampai pada penyusunan laporan skripsi, sangatlah sulit bagi penulis untuk menyelesaikan skripsi ini. Oleh karena itu, penulis mengucapkan terima kasih kepada:

a. Fachroni Arbi Murad, S.Kom., M.Kom., selaku dosen pembimbing yang telah menyediakan waktu, tenaga, dan pikiran untuk mengarahkan penulis dalam penyusunan laporan skripsi ini;
b. Orang tua dan keluarga penulis yang telah memberikan bantuan dukungan moral dan material; dan
c. Sahabat dan teman-teman yang telah banyak membantu penulis dalam menyelesaikan laporan skripsi ini.

Akhir kata, penulis berharap Allah SWT berkenan membalas segala kebaikan semua pihak yang telah membantu. Semoga laporan skripsi ini membawa manfaat bagi pengembangan ilmu.

Depok, 03 Juni 2021

Penulis

Abstrak

Selama beberapa tahun terakhir, penggunaan teknologi virtualisasi telah meningkat secara signifikan. Hal ini menimbulkan permintaan akan solusi virtualisasi yang efisien dan aman menjadi tinggi. Virtualisasi berbasis container dan virtualisasi berbasis hypervisor merupakan dua jenis teknologi virtualisasi yang digunakan, dari dua jenis virtualisasi ini, virtualisasi berbasis container mampu menyediakan lingkungan virtual yang lebih ringan, efisien dan aman, tetapi bukannya tanpa memiliki masalah kerentanan keamanan. Dalam penelitian ini, akan membahas mengenai analisis kerentanan keamanan sistem internal Docker, yang sudah diketahui menggunakan teknik virtualisasi container. Dalam proses menganalisis sistem docker berbasis container mempertimbangkan dua area: keamanan internal Docker, dan bagaimana Docker berinteraksi dengan fitur keamanan kernel Linux, seperti SELinux dan AppArmor untuk memperkuat sistem host. Lebih lanjut, penelitian ini juga membahas dan melakukan penetration testing pada sistem docker dengan menggunakan metode zero entry hacking yang sudah dimodifikasi sesuai dengan kebutuhan dari penelitian untuk mencapai hasil maksimal dan akurat. Berdasarkan hasil penetration testing yang telah dilakukan menghasilkan celah kerentanan keamanan sistem internal docker berupa port API yang terbuka pada daemon docker dan tidak terdapat pembatasan hak akses terhadap container pada sistem internal docker, sehingga menimbulkan model serangan yang muncul yaitu model serangan docker daemon surface attack dan docker escapes container attack.

***Kata Kunci**: Container, Docker, Security, Penetration Testing, Vulnerability*

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1 Latar Belakang

Pesatnya perkembangan teknologi informasi saat ini, terutama di bidang IT (*Information Technology*) mempengaruhi aspek dalam lingkungan pada *data center* untuk *development* dan *production* sistem *web* dan *mobile*. *Cloud computing* menjadi solusi dalam *development* dan *production* pada web dan mobile. *cloud computing* tidak perlu harus menyiapkan server fisik pada perusahaan IT yang sedang mengerjakan *development* dan *production* sistem *web* dan *mobile*.

Cloud computing adalah *next generation internet computing* dan *next generation data center* yang merupakan hasil dari inovasi pengembangan teknologi komputer sebelumnya seperti *grid computing, utility computing dan software as a services* dan lain-lain. *Cloud computing* merupakan teknologi virtualisasi yang berfungsi sebagai sarana untuk perbandingan luas layanan *cloud* dan strategi penempatan kerja, dan memungkinkan akses jaringan dimana-mana seperti penyediaan *server*, *storage*, *applications* dan *services*, yang dapat dengan cepat disediakan dan dirilis dengan upaya manajemen minimal atau interaksi penyedia layanan. Sehingga Teknik virtualisasi menawarkan manfaat besar yang telah mendorong perkembangannya dengan cepat (Amalia, 2019).

Pertumbuhan penggunaan teknologi virtualisasi mendorong permintaan akan solusi virtualisasi yang dapat menyediakan lingkungan pengguna yang padat, dapat disesuaikan, dan aman Pada teknik virtualisasi dapat diklasifikasikan menjadi dua kelas utama yaitu Teknik virtualisasi berbasis *hypervisors* dan Teknik virtualisasi berbasis *container*. *Hypervisor* merupakan teknologi virtualisasi yang menjadi landasan agar berbagai sistem operasi dapat berjalan secara bersamaan pada sebuah mesin (Kurniawan, 2016). Sedangkan *container* merupakan suatu teknologi virtualisasi yang melakukan isolasi pada tingkat sistem operasi. (Rad & Ahmadi, 2017)

Pada teknologi virtualisasi berbasis *hypervisor* dalam pengelolaan server pada *data center* untuk *development* dan *production* sistem *web* dan *mobile*. administrator harus mengalokasi *resources* yang cukup besar sehingga saat *development* sistem *web* atau *mobile* membutuhkan waktu yang lama, serta pada teknik virtualisasi berbasis *hypervisor* harus dapat memiliki akses ke kernel host. Sedangkan pada virtualisasi berbasis *container* dalam pengalokasian *resource* pada sistem operasi host tidak mebutuhkan banyak resource dikarenakan virtualisasi berbasis *container* memiliki sifat *flexible* dan *scalable*. (Hashemi & Bardsiri, 2012).

Maka dari dua jenis teknik virtualisasi tersebut, teknik virtualisasi berbasis *container* mampu menyediakan lingkungan virtual sepuluh kali lebih banyak untuk dapat berjalan di server fisik dibandingkan dengan virtualisasi berbasis *hypervisor*.

Salah satu aplikasi atau *platform* yang menawarkan teknologi virtualisasi berbasis container adalah docker. Secara teori, docker yang menawarkan virtualisasi berbasis container dianggap lebih bagus performa, keamananan, dan kehandalan dibandingkan virtualisasi yang berbasis hypervisor seperti virtual box karena lebih efisien berdasarkan arsitektur teknologinya dan juga semakin banyak diterapkan pada lingkungan virtualisasi server yang biasa digunakan untuk *development* dan *production* dari sebuah aplikasi web ataupun mobile. Namun, docker berbasis *container* ini juga memiliki beberapa masalah kerentanan keamanan. Sehingga pada penelitian skripsi ini saya menganalisis model serangan dan kerentanan keamanan sistem pada internal docker dari ancaman *docker escapes container attack* dan d*ocker daemon surface attack* pada saat membangun dan mengelola sistem internal docker. Serta melakukan *penetration testing* dan melakukan pengamanan terhadap sistem internal docker yang telah dibangun dan dikelola dengan melakukan singkornisasi keamanan pada kernel sistem operasi *host / guest host (Server).* (Rad & Ahmadi, 2017)

1.2 Perumusan Masalah

Berdasarkan hal-hal yang telah diuraikan dalam latar belakang, maka rumusan masalah dalam skripsi ini adalah sebagai berikut ini:

Bagaimana cara untuk membuat dan membangun suatu sistem menggunakan virtualisasi berbasis *container*?

Bagaimana cara untuk menganalisa model serangan dan kerentanan keamanan sistem internal docker dari *Docker escapes attack* dan *Docker daemon attack* yang terjadi pada sistem internal docker?

Bagaimana cara untuk melakukan *penetration testing* pada sistem internal docker dari model serangan dan kerentanan keamanan yang telah diketahui.

Bagaimana cara untuk mengamankan sistem internal docker model serangan dan kerentanan keamanan sistem internal docker dari *Docker escape container attack* dan *Docker daemon attack* dengan melakukan sinkronisasi keamanan pada kernel sistem operasi host yaitu *namespaces*, *cgroups*, pembatasan *resources*, dan *Apparmor*, serta meminimalisir *misconfiguration* pada sistem internal docker.

1.3 Batasan Masalah

Pada penelitian ini, ruang lingkup penelitian ini meliputi:

1. Menganalisa model serangan dan celah kerentanan keamanan dari *Docker escape container attack dan Docker daemon Attack* pada sistem internal docker menggunakan *tool Docker Bench Security*.

2. Melakukan ujicoba keamanan atau yang biasa disebut *penetration testing* pada sistem internal docker menggunakan *automation tool* yaitu *BoTB dan Docker Bench Security* serta menggunakan *manual tool.*

3. Mengamankan sistem internal docker dengan melakukan *audit* dan langkah pencegahan supaya dapat meminimalisir terjadinya ancaman dari serangan *Docker escape container attack* dan *Docker daemon attack.*

1.4 Tujuan dan Manfaat

1.4.1 Tujuan

a. Membuat dan membangun suatu sistem menggunakan virtualisasi berbasis *container* yaitu dengan platform yang bernama Docker.
b. Mengetahui model serangan dan kerentanan keamanan sistem internal docker dari *Docker escapes attack dan Docker daemon attack* yang terjadi pada sistem internal docker
c. Melakukan *penetration testing* pada sistem internal docker dari model serangan dan kerentanan keamanan yang telah diketahui.
d. Mengamankan sistem internal docker model serangan dan kerentanan keamanan sistem internal docker dari Docker *escape container attack* dan Docker *daemon attack* dengan melakukan sinkronisasi keamanan pada kernel sistem operasi host yaitu *namespaces, cgroups*, pembatasan *resources*, dan *Apparmor*, serta meminimalisir misconfiguration pada sistem internal docker.

1.4.2 Manfaat

a. Mengetahui teknik virtualisasi server yang lebih baik dari segi performa, keamanan, dan kehandalan yaitu teknik virtualisasi berbasis container. Karena dapat membuat satu atau lebih instance sistem virtual di satu server dengan platform atau aplikasi dari teknik virtualisasi berbasis *container* yaitu docker.
b. Melakukan langkah pencegahanan terhadap celah kerentanan keamanan pada sistem internal docker dari model serangan *Docker escape attack* dan *Docker daemon attack* dengan melakukan *monitoring* pada sistem internal docker.
c. Melakukan evaluasi dan peningkatan keamanan pada sistem internal docker secara berkala.

BAB V

PENUTUP

5.1 KESIMPULAN

Berdasarkan hasil dari penelitian yang telah dilakukan oleh penulis, maka dapat ditarik kesimpulan sebagai berikut:

1. Pembuatan sistem internal docker pada sistem operasi *guest host* berhasil dilakukan sesuai dengan prosedur.
2. Pengujian sistem internal docker menggunakan cara penetration testing dengan menggunakan metode pengujian *Zero Entry Hacking* (ZEH) berjalan sesuai dengan mekanisme yang ada yaitu dimulai dari proses pengintaian sistem (*Reconnaissance*), pemindaian sistem (*Scanning*), exploitasi (*Exploitation*), serta auditing dan pengamanan sistem (*Auditing* dan *Securing*)
3. Alat-alat atau tools yang digunakan pada proses *penetration testing* ini merupakan tools yang terbaik diantara tool sejenisnya, dikarenakan alat-alat atau tools yang digunakan memenuhi semua parameter aturan yang telah ditetapkan. alat-alat tools yang digunakan diantaranya adalah *Break Out of The Box (BoTB)*, *Docker Bench Security*, NMAP, serta tool kernel sistem keamanan dari sistem operasi *guest host* yang disingkronkan dengan sistem internal docker yaitu *capabilities, namespaces, selinux dan apparmor*
4. Hasil yang didapatkan dari *penetration testing* ialah celah kerentanan keamanan pada sistem internal docker berupa port API yang terbuka pada *daemon* docker dan tidak membatasi hak akses terhadap *container* docker yang dapat diexploitasi oleh attacker.

5.2 SARAN

Saran yang dapat diusulkan pada penelitian ini adalah:

1. Menggunakan software orchestra untuk mengelola banyak mesin docker dalam satu server contoh *software orchestra* ialah kubernetas dan *docker swarm*.
2. Membuat perbandingan antara *virtualization hypervisors* dengan *virtualization container.*

DAFTAR PUSTAKA

Ahamed, W. S. (2020). Information Systems Security and Assurance Management. *Security Audit of Docker Container Images in Cloud Architecture*, 1-12.

Amalia, R. (2019). Informatics and Computer Engineering. *Analisis Kinerja Web Server Menggunakan Metode Load Balancing as a Service Pada Lingkungan Virtual Openstack*, 1-9.

Bui, T. (2015, January). Docker Security. *Analysis of Docker Security*, 7.

Engerbretson, P. (2013). ethical hacking and penetration testing made easy. *The Basic of hacking and penetration testing*, 1-10.

Jian, Z., & Chen, L. (2017). *A Defense Method against Docker Escape Attack*, 142-146.

Kurniawan, I. N. (2016). Implementasi Virtualisasi Menggunakan Xen Hypervisor. Jurnal Manajemen Informatika. 36–42.

Martin, A., Raponi, S., Combe, T., & Petro, R. D. (2018). Docker ecosystem. *Vulnerbility Analysist*, 1-19.

Rad, B. B., Bhatti, H. J., & Ahmadi, M. (2017). international journal of computer science. *An Introduction to Docker and Analysis of its performace*, 228-235.

Reshotava,Elena, Karhunen, J., Nyman, T., & Asokan, N. (2014). Technology. *Security of OS-Level Virtualization Technologies*, 77-93.

Sahtyawan, R. (2019). Penerapan Zero Entry Hackin didalam Security Misconfiguration Pada VAPT (VULNERABILITY ASSESSMENT AND PENETRATION TESTING). *Information System Management*.

Salman Baset, S. B. (2016). Docker and Container Security.

Sayyed Mohsen Hashemi, &. A. (2012). ARPN Journal of Systems and Software. *Cloud Computing Vs. Grid Computing*, 188-194.

Solomon, M. G. (2009). Infromation security illumniated. *Infromation security illumniated*.

Sultan, S., AHMAD, I., & DIMITRIOU,, T. (2019). Received February 18, 2019, accepted April 10, 2019, date of publication April 17, 2019, date of current version May 1, 2019. *Container Security: Issues, Challenges,*, 52976-52996.

Yao, L., & Jie , C. (2017). 5th International Conference on Mechatronics, Materials, Chemistry and Computer Engineering. *Research of Penetration Testing Technology in Docker Environment*, 1354-1359.

Vats, P., Mandot, M., & Gosain, A. (2020). International Conference on Reliability, Infocom Technologies and Optimization. *A Comprehensive Literature Review of Penetration*, 674-680.

Vipin Jain, D. B. (2021). IOP Conference Series: Materials Science and Engineering. *Static Vulnerability Analysis of Docker Images*, 1-9.

Vrancken, J. (2020). *A Methodology for Penetration Testing Docker Systems*, 1-81.

Lampiran 1 - Daftar Riwayat Hidup

LAMPIRAN

DAFTAR RIWAYAT HIDUP

Kevin Harada

Lahir di Jakarta, 27 April 1998. Lulus dari SDIT Baitussalam pada tahun 2010, SMPIT Baitussalam pada tahun 2013, SMKN 3 Bogor pada tahun 2016 dan Diploma II program studi *Network Administrator Professional* di CCIT FTUI pada tahun 2018. Saat ini sedang menempuh Pendidikan Diploma IV Program Studi Teknik Informatika Jurusan Teknik Informatika dan Komputer di Politeknik Negeri Jakarta.

Lampiran 2 – File konfigurasi

File Daemon.json

```
{
  "debug": true,
  "icc": false,
  "log-level": "info",
  "log-driver": "syslog",
  "hosts": ["tcp://0.0.0.0:2375", "unix:///var/run/docker.sock"],
  "experimental": true,
  "live-restore": true
}
{
  "userns-remap":"sdocker"
}
{
  "authorization-plugins": ["sdocker"],
  "disable-legacy-registry": true,
  "userland-proxy": false,
  "userland-proxy-path": "/usr/libexec/docker-proxy",
  "no-new-privileges": true
}
```

```
{
  "default-cgroupns-mode": "private",
  "default-gateway": "",
  "default-gateway-v6": "",
  "default-runtime": "runc",
  "default-shm-size": "64M",
  "default-ulimits": {
    "sdocker": {
      "Hard": 64000,
      "Name": "sdocker",
      "Soft": 64000
        }
    }
}
{
  "seccomp-profile": "",
  "selinux-enabled": true,
  "shutdown-timeout": 100,
  "storage-driver": ""
}
```

Konfigurasi Selinux

```
# IP Spoofing protection
net.ipv4.conf.all.rp_filter = 1
net.ipv4.conf.default.rp_filter = 1

# Ignore ICMP broadcast requests
net.ipv4.icmp_echo_ignore_broadcasts = 1

# Disable source packet routing
net.ipv4.conf.all.accept_source_route = 0
net.ipv6.conf.all.accept_source_route = 0
net.ipv4.conf.default.accept_source_route = 0
net.ipv6.conf.default.accept_source_route = 0

# Ignore send redirects
net.ipv4.conf.all.send_redirects = 0
net.ipv4.conf.default.send_redirects = 0

# Block SYN attacks
net.ipv4.tcp_syncookies = 1
net.ipv4.tcp_max_syn_backlog = 2048
net.ipv4.tcp_synack_retries = 2
net.ipv4.tcp_syn_retries = 5

# Log Martians
net.ipv4.conf.all.log_martians = 1
net.ipv4.icmp_ignore_bogus_error_responses = 1

# Ignore ICMP redirects
net.ipv4.conf.all.accept_redirects = 0
net.ipv6.conf.all.accept_redirects = 0
net.ipv4.conf.default.accept_redirects = 0
net.ipv6.conf.default.accept_redirects = 0

# Ignore Directed pings
net.ipv4.icmp_echo_ignore_all = 1
```

Docker-ubuntu-full.sh

```
#!/bin/bash

# check software apabila sudah ada pada versi sebelumnya akan di remove
# bila masih kosong step ini akan mengeluarkan hasil software not installed
sudo apt-get remove docker  docker.io containerd runc
sudo apt-get update
sudo apt-get install \
    apt-transport-https \
    ca-certificates \
    curl \
    gnupg \
    lsb-release

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg

echo \
  "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu \
  $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

```
echo \
  "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu \
  $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null


#service dependencies for docker-compose
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io

echo ""
echo "Proses Installasi Telah Berhasil"
echo "Modified Software By KHProject"
```

Docker-compose-full.sh

```
#!/bin/bash
sudo apt install mysql-server mysql-client
sudo apt install python3-dev python3-pip pipenv libmysqlclient-dev

#installasi docker-compose
sudo curl -L "https://github.com/docker/compose/releases/download/1.29.1/docker-compose-$(uname -s)-$(uname -m)" \
-o /usr/local/bin/docker-compose

sudo chmod +x /usr/local/bin/docker-compose

echo ""
echo "Proses Installasi Telah Berhasil"
echo "Modified Software By KHProject"
```

Portainer

```
#!/bin/bash

#installasi portainer for monitoring docker GUI
docker run -d -p 9000:9000 --name portainer \
--restart always -v /var/run/docker.sock:/var/run/docker.sock \
-v /opt/portainer:/data portainer/portainer

echo ""
echo "Proses Installasi Telah Berhasil"
echo "Modified Software By KHProject"
```

Panduan penggunaan BoTB

SINTAK	PENJELASAN
-aggr string	Attempt to exploit RuncPWN (default "nil")
-autopwn	Attempt to autopwn exposed sockets
-cicd	Attempt to autopwn but don't drop to TTY,return exit code 1 if successful else 0
-find-docker	Attempt to find Dockerd
-find-http	Hunt for Available UNIX Domain Sockets with HTTP
-find-sockets	Hunt for Available UNIX Domain Sockets
-keyMin int	Minimum key id range (default 1) (default 1)
-keyMax int	Maximum key id range (default 100000000) and max system value is 999999999 (default 100000000)
-pwn-privileged string	Provide a command payload to try exploit --privilege CGROUP release_agent's (default "nil")
-pwnKeyctl	Abuse keyctl syscalls and extract data from Linux Kernel keyrings
-recon	Perform Recon of the Container ENV