컨테이너&가상화

- CRI-O기반의 k8s 설치
- artifact관리툴 - nexus 구축하기
- onlyoffice를 컨테이너 기반에서 설치하기
- pip로 docker-compose 설치 오류
- QEMU tcp원격접속 허용하기
- VMware GuestOS에서 마우스 5버튼 작동안될 경우 조치방법
- rancher 패스워드 초기화 방법

CRI-O기반의 k8s 설치

사전사항

1. OS환경설정

```
$> swapoff -a

$> cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf
br_netfilter
EOF

$> cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
EOF

$> sudo sysctl --system
```

2. crio / kubernetees 패키지 리포지터리 구성

```
$> cat /etc/yum.repos.d/libcontainers.repo
[devel_kubic_libcontainers_stable]
name=Stable Releases of Upstream github.com/containers packages (CentOS_8)
type=rpm-md
baseurl=https://download.opensuse.org/repositories/devel:/kubic:/libcontainers:/stable/CentOS_8/
gpgcheck=1
gpgkey=https://download.opensuse.org/repositories/devel:/kubic:/libcontainers:/stable/CentOS_8/repodata/repomd.xml.key
enabled=1
```

```
$> cat /etc/yum.repos.d/cri-o.repo
[cri-o]
name=CRI-O
baseurl=https://pkgs.k8s.io/addons:/cri-o:/stable:/v1.28/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/addons:/cri-o:/stable:/v1.28/rpm/repodata/repomd.xml.key
```

> k8s 1.24이후 버전에서는 repository url이 변경되었습니다.

```
$> cat /etc/yum.repos.d/k8s.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.28/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.28/rpm/repodata/repomd.xml.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
```

3. 패키지 설치

```
$> yum install -y cri-o
$> yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes
$> systemctl enable crio --now
$> systemctl enable kubelet
```

클러스터 생성 (control palin 1번에서만 수행)

1. kubeadm 클러스터 생성

```
$> kubeadm init --control-plane-endpoint 172.21.107.238:6443 --pod-network-cidr 10.250.0.0/16 --ignore-preflight-errors=all --upload-certs

## 결과값중에 control / worker 노드별 join 명령이 다르기 때문에 별도로 복사해놓어야 함
```

##Control node용

```
$> kubeadm join 172.21.107.238:6443 --token abcd \
      --discovery-token-ca-cert-hash sha256:yyy \
      --control-plane --certificate-key zzz
```

Worker Node용

```
$> kubeadm join 172.21.107.238:6443 --token abcd \
      --discovery-token-ca-cert-hash sha256:yyyy \
      --cri-socket
 --ignore-preflight-errors=all
```

2. 인증서 정보 복사

```
$> mkdir -p $HOME/.kube
$> /bin/cp /etc/kubernetes/admin.conf $HOME/.kube/config
$> chown $(id -u):$(id -g) $HOME/.kube/config
$> export KUBECONFIG=/etc/kubernetes/admin.conf
```

3. CNI 설치(Calico)

```
$> curl https://calico-v3-25.netlify.app/archive/v3.25/manifests/calico.yaml -O
$> kubectl apply -f calico.yaml
```

클러스터 연동

1. 타 Control plain 연동 (Control Plain 한대씩 순차 작업 수행)

```
$> kubeadm join 172.21.107.238:6443 --token abcd \
      --discovery-token-ca-cert-hash sha256:yyy \
      --control-plane --certificate-key zzz
```

2. 노드 연동 확인 (Control plain에서 수행)

```
$> kubectl get no
NAME                    STATUS   ROLES                  AGE    VERSION
k8stesttx-k8s-master-dev01   Ready    control-plane,master   3h6m   v1.23.5
k8stesttx-k8s-master-dev02   Ready    control-plane,master   3h6m   v1.23.5
k8stesttx-k8s-master-dev03   Ready    control-plane,master   3h6m   v1.23.5
```

3. Worker Node 연동

```
$> kubeadm join 172.21.107.238:6443 --token abcd \
      --discovery-token-ca-cert-hash sha256:yyyy \
 --ignore-preflight-errors=all
```

4. 노드 연동 확인 (Control plain에서 수행)

```
$> kubectl get no
NAME                    STATUS   ROLES                  AGE    VERSION
...
k8stesttx-k8s-worker-dev01   Ready    <none>                 40m    v1.23.5
k8stesttx-k8s-worker-dev02   Ready    <none>                 40m    v1.23.5
k8stesttx-k8s-worker-dev03   Ready    <none>                 40m    v1.23.5
```

k8s 인증서 10년으로 연장

1. 인증서 연장 스크립트

```
$> git clone https://github.com/yuyicai/update-kube-cert.git
$> cd update-kube-cert
$> chmod 755 update-kubeadm-cert.sh
$> ./update-kubeadm-cert.sh all
```

2. 업데이트 전 인증서 정보 확인

```
$> kubeadm certs check-expiration
[check-expiration] Reading configuration from the cluster...
[check-expiration] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'

CERTIFICATE                EXPIRES                  RESIDUAL TIME   CERTIFICATE AUTHORITY   EXTERNALLY MANAGED
admin.conf                 Apr 17, 2023 06:09 UTC   364d            ca                      no
apiserver                  Apr 17, 2023 06:09 UTC   364d            ca                      no
apiserver-etcd-client      Apr 17, 2023 06:09 UTC   364d            etcd-ca                 no
apiserver-kubelet-client   Apr 17, 2023 06:09 UTC   364d            ca                      no
controller-manager.conf    Apr 17, 2023 06:09 UTC   364d            ca                      no
etcd-healthcheck-client    Apr 17, 2023 06:09 UTC   364d            etcd-ca                 no
etcd-peer                  Apr 17, 2023 06:09 UTC   364d            etcd-ca                 no
etcd-server                Apr 17, 2023 06:09 UTC   364d            etcd-ca                 no
front-proxy-client         Apr 17, 2023 06:09 UTC   364d            front-proxy-ca          no
scheduler.conf             Apr 17, 2023 06:09 UTC   364d            ca                      no

CERTIFICATE AUTHORITY   EXPIRES                  RESIDUAL TIME   EXTERNALLY MANAGED
ca                      Apr 17, 2032 04:17 UTC   9y              no
etcd-ca                 Apr 17, 2032 04:17 UTC   9y              no
front-proxy-ca          Apr 17, 2032 04:17 UTC   9y              no
```

3. 인증서 업데이트 (Control Plain 1대씩 순차 작업 수행, 서버단위로 30초 가량 대기 필요)

```
$> chmod +x cert_update.sh
$> ./cert_update.sh
...
```

4. 인증서 갱신정보 확인

```
$> kubeadm certs check-expiration
[check-expiration] Reading configuration from the cluster...
[check-expiration] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'

CERTIFICATE                EXPIRES                  RESIDUAL TIME   CERTIFICATE AUTHORITY   EXTERNALLY MANAGED
admin.conf                 Apr 17, 2032 06:09 UTC   9y              ca                      no
apiserver                  Apr 17, 2032 06:09 UTC   9y              ca                      no
apiserver-etcd-client      Apr 17, 2032 06:09 UTC   9y              etcd-ca                 no
apiserver-kubelet-client   Apr 17, 2032 06:09 UTC   9y              ca                      no
controller-manager.conf    Apr 17, 2032 06:09 UTC   9y              ca                      no
etcd-healthcheck-client    Apr 17, 2032 06:09 UTC   9y              etcd-ca                 no
etcd-peer                  Apr 17, 2032 06:09 UTC   9y              etcd-ca                 no
etcd-server                Apr 17, 2032 06:09 UTC   9y              etcd-ca                 no
front-proxy-client         Apr 17, 2032 06:09 UTC   9y              front-proxy-ca          no
scheduler.conf             Apr 17, 2032 06:09 UTC   9y              ca                      no

CERTIFICATE AUTHORITY   EXPIRES                  RESIDUAL TIME   EXTERNALLY MANAGED
ca                      Apr 17, 2032 04:17 UTC   9y              no
etcd-ca                 Apr 17, 2032 04:17 UTC   9y              no
front-proxy-ca          Apr 17, 2032 04:17 UTC   9y              no
```

Reference

- *https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/*
- *https://github.com/yuyicai/update-kube-cert.git*

artifact관리툴 - nexus 구축하기

시작하는 말

안녕하세요, 고니 입니다.
기존에 작성했던 컨텐츠들 업데이트를 하면서 문서의 리팩토링(Refactoring)을 진행해보려고 합니다.

Devop pipeline의 산출물(artifact)을 관리할 수 있는 대표적인 툴이 Sonatype에서 제공하는 Nexus와 JFrog에서 제공하는 jfrog 인데요.
두가지 툴의 장단점 비교는 별도로 작성해 볼 예정이고, 가장많이 사용하고 있는 Nexus를 소개해보려고 합니다.

Nexus 소개

1. NeXus Repository Manager (약어;NXRM)
2. 지원 OS : Windows / Linux / Unix / OSX
3. 설치버전 : 3.29.2 OSS ('21.01.10 기준)
4. License : Eclipse Public License-v 1.0
5. Pro버전 추가 기능 : Atalssian Crowd지원, Staging / Build 분리 / 태깅 / 계정별 토큰 / HA / 아티팩트별 OSS보안 취약점 진단 / 기술지원 / 그룹 저장소 관리 기능 / SAML / SSO 기능

Nexus 시스템 요구사항

1. hardware requirements

Type	Minimal	requirement
CPU	4Core	8Core
Memory	Host : 8Gb JAVA Heap : 2Gb	Host : 8Gb Java Heap : 4GB
Disk	330M 이상	330M 이상

- 사용가능 디스크 공간이 4GB이하일 경우 Readonly모드로 작동
- Java Directory 메모리 설정공식은 (호스트메모리 * 0.6) - java heap

2. 지원 가능 파일시스템

파일시스템 종류	Embbed	Blob	비 고
XFS	O	O	
NFS V4	O	O	
NFS V3	X	X	오류발생
AWS - EBS	O	O	
AWS - EFS	X	O	Embbed 응답 이슈
AWS - S3	X	O	임베디드 데이터 적용불가
AWS- S3호환	X	X	S3 호환 저장소는 지원 불가
Samba/CIFS	X	O	임베디드 데이터 적용불가
GlusterFS	X	X	Split-brain 발생, 응답속도 지연

파일시스템 종류	Embbed	Blob	비 고
Fuse	X	X	데이터 오류 발생

* Embbed : 응답성이 빠른 스토리지 (DB / Elastic Search)
* Blob : binary 저장소

3. 디렉토리 구조

디렉토리명	설 명
bin	시작스크립트 / 시작관련 구성파일
etc	nxrm 구성파일 /런타임 구성파일
lib	nxrm 라이브러리 파일
public	nxrm 공개 리소스데이터
system	프로그램 구성 데이터 / 플러그인
blobs	blob 저장소
cache	캐시된 데이터
db	orientDB 데이터 저장
elasticsearch	현재 구성된 elasticsearch 데이터
health-check	저장소 상태 확인 기능 보고서
keystores	저장소 관리자를 식별하는 키 데이터
log	로그데이터 nexus.log → 관리자 프로그램 로그, request.log → http요청로그 jvm.log → jvm stdout, stderror 및 덤프데이터 karaf.log → apache karaf 컨테이너 로그 파일 audit → 감사 활성시 감사로그 데이터 staks → 작업별 수행한 로그 tmp → 임시저장소

Nexus 기능

1. 리포지터리 종류

- hosted : 서버에 저장되는 리포지터리
- proxy : hosted와 연결하여 사용하는 cache용 리포지터리
- group : 여러개의 hosted or proxy를 그룹으로 묶은 논리개념의 리포지터리

2. 리포지터리 버전관리 방식

- release : 정식릴리즈되는 아티팩트 저장소 / 동일버전으로 한번만 업로드 가능(필요시 삭제 후 업로드해야 함)
- snapshot : 수시로 릴리즈 되는 아티팩트 저장소 / 같은 버전으로 여러번 업로드 가능

3. 리포지터리 유형

- maven2 : 메이븐에서 제공하는 메타데이터로 버전관리 (group / artifact id / version의 메타데이터로 관리함)
- raw : 별도의 메타데이터는 없으며 파일 위치와 파일이름만 생성
- yum : yum 리포지터리 저장소

4. 아티팩트 관리 방식

- curl을 이용한 바이너리 관리

- test_repo에 있는 img-1.png 파일 다운로드

```
$ curl -u 'test:test123' http://192.168.10.10:11000/repository/test_repo/img-1.png -o img-1.png
```

- test.tar.gz 파일을 test_repo로 업로드

```
$ curl -u 'test:test123' --upload-file ./test.tar.gz http://192.168.10.10:11000/repository/test_repo/test.tar.gz
```

- 다중파일 업로드시

```
$find ./ -exec curl -u 'test:test123' -T ./{} http://192.168.10.10:11000/repository/test_repo/test.tar.gz/{} \;
```

- test_repo에 있는 test.tar.gz 파일 삭제

```
$> curl --request DELETE -u 'test:test123' http://http/192.168.10.10:11000/repository/test_repo/test.tar.gz
```

Nexus 설치 (Docker 기반)

1. container pull 하기

```
$> docker pull sonatype/nexus3:3.29.2     #2.x을 사용할 경우 sonatype/nexus를 사용하면 됨
```

2. nexus 저장용 데이터 생성하기

```
$> mkdir /data/nexus-repo
$> chown -R 200 /data/nexus-repo/
```

3. container 실행하기

```
$ docker run -d -p 88081:8081 \
-e TZ=Asia/Seoul  \
-v /etc/localtime:/etc/localtime:ro \
-v /data/nexus-repo/:/nexus-data \
-name nexus-repo sonatype/nexus3:3.29.2
```

4. Container log 확인 (started sonatype 메시지가 출력되면 정상구동 완료)

```
$ docker logs -f nexus-repo
...
2021-01-19 23:32:02,793+0900 INFO  [jetty-main-1] *SYSTEM org.sonatype.nexus.repository.httpbridge.internal.ViewServlet - Initialized
2021-01-19 23:32:02,870+0900 INFO  [jetty-main-
1] *SYSTEM org.eclipse.jetty.server.handler.ContextHandler - Started o.e.j.w.WebAppContext@637c45d9{Sonatype Nexus,/,file:~/~//opt/sonatype/nexus/public/

2021-01-19 23:32:02,928+0900 INFO  [jetty-main-
1] *SYSTEM org.eclipse.jetty.server.AbstractConnector - Started ServerConnector@45bb642d{HTTP/1.1, (http/1.1)}{0.0.0.0:8081}
2021-01-19 23:32:02,929+0900 INFO  [jetty-main-1] *SYSTEM org.eclipse.jetty.server.Server - Started @41677ms
2021-01-19 23:32:02,930+0900 INFO  [jetty-main-1] *SYSTEM org.sonatype.nexus.bootstrap.jetty.JettyServer -
~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~--
\\Started Sonatype Nexus OSS 3.29.2-02
\\~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~--
```

5. web-ui 접속하기 – http://{{서버IP}}:88081/

6. 초기 접속 화면

7. 로그인 패스워드 확인

```
$docker exec -it nexus cat /nexus-data/admin.password
1123-aad-xxxx-xxxx-
```

8. 로그인 계정 (admin)

9. admin 패스워드 설정 (변경할 패스워드 입력 후 Next)
10. 마침

Repository Cleanup 정책관리

1. cleanup policies 정책 수립하기

- 상단 톱니바퀴 모양 → Repository → Cleanup Policies → Create Cleanup Policy
- Cleanup policy 규칙 메뉴 구성
- 항목별 세부구성 (예시는 90일 보관 후 삭제하는 정책)

항 목	설 명	설정값	비 고
Name	규칙이름 설정	proxy_cleanup_{{보관일}}d	고유의 값 설정
Format	Repository format 설정	raw	모든 유형의 repository로 할 경우 all formats 선택
notes	규칙에 대한 설명정보		옵션
Component Age	저장된 데이터의 보관 주기	90	체크박스 선택 후 보관일 입력
Component Usage	다운로드 되지 않은 데이터 보관주기	90	체크박스 선택 후 보관일 입력
preview repository	해당 정책을 수립시 삭제되는 데이터 리스트가 보임		

Repository 관리

1. Hosted 기반의 리포지터리 생성

- 상단 톱니바퀴 모양 → Repository → Create Repository → RAW (hosted)
- repository 생성 메뉴구성
- 항목별 세부설명

항 목	설 명	설정값	비 고
Name	Repository 고유 이름	repo이름 입력	문자, 숫자로 구성되어야 하고 특수문자는(-, _)만 허용되고 마침표로 마칠수 없음
Online	repository 활성화 여부	체크	기본값은 체크되어 있음
Content Disposition	컨텐츠 관리 방법	attachment	Attachment / inline 중 택1, Attachment로 구성해야 파일 전송이 유용함
Blob store	바이너리 저장 경로	default	별도 바이너리 저장소를 설정하지 않는 경우 default로 구성
Deployment Policy	아티팩트 배포 / 업데이트 허용 여부	Allow redeploy	Allow redeploy : 재 배포 허용 Deny redeploy : 재배포 불가 read only : 업데이트 불가, 읽기전용으로 구성
Cleanup policies	Repository 오래된 파일 삭제		Cleanup 정책 수립했을때 적용

2. Proxy 기반의 리포지터리 생성

- 상단 톱니바퀴 모양 → Repository → Create Repository → RAW (Proxy)
- repository 생성 메뉴구성
- 항목별 세부설명

항 목	설 명	설정값	비 고
Name	Repository 고유 이름		문자, 숫자로 구성되어야 하고 특수문자는(-, _)만 허용되고 마침표로 마칠수 없음
Online	repository 활성화 여부		기본값음 체크되어 있음
Remote storage	hosted 리포지터리 주소 입력		
Blocked	하위 Proxy 연결을 차단할 경우 체크	체크수행	
Auto blocking enabled	Remote Storage에 연결이 되지 않을 경우 하위 Proxy 연결 차단	체크수행	
Maximum component age	cache된 바이너리의 최대 보관 주기 (분단위)	60	
Maximum metadata age	cache된 메타데이터 최대 보관 주기 (분단위)	60	
Blob store	바이너리 저장 경로	default	별도 Storage로 구성할 경우 구성한Stoage 이름 선택
Strict Content Type Validation	바이너리 저장시 MIME유형 여부 확인	체크 해제	
Not found cache enabled	원격 저장소에 없는 컨텐츠 요청시 캐쉬로 응답	체크 수행	
Not found cache TTL	원격 저장소에 파일없다는 캐쉬 정보 보관주기 (분단위)	60	
Cleanup policies	Repository 오래된 파일 삭제		Cleanup 정책 수립했을때 적용
HTTP authentication	원격저장소의 계정명 입력	host용 nexus에 등록한 계정정보 입력	

3. Proxy Repo에 cache 정보 강제 삭제
   - 톱니바퀴 → Repository → Proxy타입의 Repository 선택 → Invalidate cache 선택

Proxy 기능 테스트

1. 보관 주기별 Proxy 데이터 전송

보관주기	Proxy 반영시간
-1	수동으로 캐쉬 삭제전까지 반영 안됨
0	즉시
1	1분뒤 Proxy repo에 반영
2	2분위 Proxy Repo에 반영

2. Proxy 전달 Traffic 정보

- 테스트정보 : Proxy에 1차버전 caching 된 상태에서 host에 새로운 바이너리가 업데이트 된 경우, caching flush 후 다시 caching 하는 과정 확인 (flush되기 전까지 지속적 호출)

- 패킷 정보 (Hosted → Proxy), 192.168.10.10 (Proxy), 192.168.100.10 (Hosted)

```
15:42:31.554851 IP 192.168.10.10.42224 > 192.168.100.10.88081: Flags [S], seq 1521618240, win 29200, options [mss 1460,nop,nop,sackOK], length 0
15:42:31.555029 IP 192.168.100.10.88081 > 192.168.10.10.42224: Flags [S.], seq 2385004980, ack 1521618241, win 29200, options [mss 1460,nop,nop,sackOK], length 0
15:42:31.555223 IP 192.168.10.10.42224 > 192.168.100.10.88081: Flags [.], ack 1, win 29200, length 0
15:42:31.555234 IP 192.168.10.10.42224 > 192.168.100.10.88081: Flags [P.], seq 1:262, ack 1,
```

onlyoffice를 컨테이너 기반에서 설치하기

MSoffice를 대체할 수 있는 offic중에 onlyoffice가 있어서 한번 설치해봤습니다.

1. Onlyoffice용 데이터 디렉토리 생성

```
$ mkdir /data/office/CommunityServer/{logs,data,lib,db}
$ mkdir /data/office/ControlPanel/{logs,data,lib,db}
```

2. onlyoffice용 DB구성 저는 기존에 실행중인 DB용 컨테이너가 있어서 기존 DB컨테이너에 DB인스턴스와 계정만 추가할께요

```
mysql> CREATE USER 'onlyoffice_user'@'localhost' IDENTIFIED BY 'onlyoffice_pass';
mysql> CREATE USER 'mail_admin'@'localhost' IDENTIFIED BY 'Isadmin123';
mysql> GRANT ALL PRIVILEGES ON * . * TO 'onlyoffice_user'@'%' IDENTIFIED BY 'onlyoffice_pass';
mysql> GRANT ALL PRIVILEGES ON * . * TO 'mail_admin'@'%' IDENTIFIED BY 'Isadmin123';
mysql> FLUSH PRIVILEGES;
```

3. Control Panel 띄우기

```
$ docker run -d --net=test --restart=always --dns 219.250.36.130 -d \
-v /var/run/docker.sock:/var/run/docker.sock \
-v /data/office/CommunityServer/data:/app/onlyoffice/CommunityServer/data \
-v /data/office/ControlPanel/data:/var/www/onlyoffice-controlpanel/Data \
-v /data/office/ControlPanel/logs:/var/log/onlyoffice-controlpanel \
--name office_panel onlyoffice/controlpanel
```

4. 커뮤니티용 onlyoffice 서버 Container 띄우기

```
$ docker run -d --net=test--restart=always --dns 219.250.36.130 -p 80:80 -p 443:443 -p 5222:5222 \
-e MYSQL_SERVER_ROOT_PASSWORD=dbpass \
-e MYSQL_SERVER_DB_NAME=onlyoffice \
-e MYSQL_SERVER_HOST=db \
-e MYSQL_SERVER_USER=onlyoffice_user \
-e MYSQL_SERVER_PASS='onlyoffice_pass' \
-e CONTROL_PANEL_PORT_80_TCP=80 \
-e CONTROL_PANEL_PORT_80_TCP_ADDR=office_panel \
-v /data/office/CommunityServer/letsencrypt:/etc/letsencrypt \
-v /sys/fs/cgroup:/sys/fs/cgroup:ro \
-v /data/office/CommunityServer/data:/var/www/onlyoffice/Data \
-v /data/office/CommunityServer/logs:/var/log/onlyoffice \
-v /data/office/DocumentServer/data:/var/www/onlyoffice/DocumentServerData \
--name office onlyoffice/communityserver
```

5. 브라우져에서 접속하기 http://{{서버IP}}

Reference

- https://helpcenter.onlyoffice.com/installation/docs-community-install-docker.aspx

pip로 docker-compose 설치 오류

1. 환경정보
   - os : centos7
   - pip : pip2 / pip3
2. 작업사항
   - pip2가 설치되어 있는 서버에서 새로운 pip를 올리려고 했더니 python3에서 사용해야 한다함
   - pip 커맨드가 pip2로 심볼릭 링크 걸려있는 상태에서 pip3로 교체

```
$> ls -l pip
/usr/bin/pip -> /usr/bin/pip2
$> ln -snf /usr/bin/pip3 /usr/bin/pip
$> pip -V
pip 21.0.1 from /usr/local/lib/python3.6/site-packages/pip (python 3.6)
```

3. pip를 통해 docker-compose 설치

```
$ pip instal docker-compose
...
Collecting cryptography>=2.5 (from paramiko>=2.4.2; extra == "ssh"->docker[ssh]<5,>=4.4.3->docker-compose)
  Downloading http://python.org/pypi/+f/2d3/2223e5b0ee029/cryptography-3.4.6.tar.gz (546kB)
   100% |████████████████████████████████| 552kB 43.7MB/s
    Complete output from command python setup.py egg_info:
    WARNING: The wheel package is not available.
    ERROR: 'pip wheel' requires the 'wheel' package. To fix this, run: pip install wheel
    Traceback (most recent call last):
      File "/usr/local/lib/python3.6/site-packages/setuptools/installer.py", line 75, in fetch_build_egg
        subprocess.check_call(cmd)
      File "/usr/lib64/python3.6/subprocess.py", line 311, in check_call
        raise CalledProcessError(retcode, cmd)
    subprocess.CalledProcessError: Command '['/usr/bin/python3', '-m', 'pip', '--disable-pip-version-check', 'wheel', '--no-deps', '-w', '/tmp/tmpn5ys3xpr', '--quiet', 'cffi>=1.12']' returned non-zero exit status 1.
    The above exception was the direct cause of the following exception: Traceback (most recent call last):   File "<string>", line 1, in <module>   File "/tmp/pip-build-0pwkcfz1/cryptography/setup.py", line 152, in <module>     rust_extensions=rust_extensions,   File "/usr/local/lib/python3.6/site-packages/setuptools/__init__.py", line 152, in setup     _install_setup_requires(attrs)   File "/usr/local/lib/python3.6/site-packages/setuptools/__init__.py", line 147, in _install_setup_requires     dist.fetch_build_eggs(dist.setup_requires)   File "/usr/local/lib/python3.6/site-packages/setuptools/dist.py", line 689, in fetch_build_eggs     replace_conflicting=True,   File "/usr/local/lib/python3.6/site-packages/pkg_resources/__init__.py", line 768, in resolve     replace_conflicting=replace_conflicting   File "/usr/local/lib/python3.6/site-packages/pkg_resources/__init__.py", line 1051, in best_match     return self.obtain(req, installer)   File "/usr/local/lib/python3.6/site-packages/pkg_resources/__init__.py", line 1063, in obtain     return installer(requirement)   File "/usr/local/lib/python3.6/site-packages/setuptools/dist.py", line 745, in fetch_build_egg     return fetch_build_egg(self, req)   File "/usr/local/lib/python3.6/site-packages/setuptools/installer.py", line 77, in fetch_build_egg     raise DistutilsError(str(e)) from e distutils.errors.DistutilsError: Command '['/usr/bin/python3', '-m', 'pip', '--disable-pip-version-check', 'wheel', '--no-deps', '-w', '/tmp/tmpn5ys3xpr', '--quiet', 'cffi>=1.12']' returned non-zero exit status 1.     ============================DEBUG ASSISTANCE==========================     If you are seeing a compilation error please try the following steps to     successfully install cryptography:     1) Upgrade to the latest pip and try again. This will fix errors for most       users. See: https://pip.pypa.io/en/stable/installing/#upgrading-pip     2) Read https://cryptography.io/en/latest/installation.html for specific       instructions for your platform.     3) Check our frequently asked questions for more information:       https://cryptography.io/en/latest/faq.html     4) Ensure you have a recent Rust toolchain installed:       https://cryptography.io/en/latest/installation.html#rust     5) If you are experiencing issues with Rust for *this release only* you may       set the environment variable `CRYPTOGRAPHY_DONT_BUILD_RUST=1`.     ============================DEBUG ASSISTANCE==========================
----------------------------------------
Command "python setup.py egg_info" failed with error code 1 in /tmp/pip-build-0pwkcfz1/cryptography/
```

   - 하지만 에러...????

4. 발생원인
   - pip모듈을 2버전에서 3버전으로 단순히 심볼릭 링크만 바꾸었던게 문제인듯...
   - python모듈이 2버전으로 설치가 되어 있었을텐데..... 그래서 python3용으로 패키지 강제 재설치

5. 추가작업

```
$ pip install --upgrade setuptools pip --ignore-installed
WARNING: Running pip install with root privileges is generally not a good idea. Try pip install --user instead. Collecting setuptools Downloading http://python.org/pypi/+f/0e8/6620d658c5ca8/setuptools-53.0.0-py3-none-any.whl (784kB) 100% |████████████████████████████████| 788kB 58.3MB/s Collecting pip Downloading http://python.org/pypi/+f/37f/d50e056e2aed6/pip-21.0.1-py3-none-any.whl (1.5MB) 100% |████████████████████████████████| 1.5MB 36.2MB/s Installing collected packages: setuptools, pip
```

6. 다시 docker-compose 설치

```
$ pip install docker-compose
 WARNING: pip is being invoked by an old script wrapper. This will fail in a future version of pip.
 Please see https://github.com/pypa/pip/issues/5599 for advice on fixing the underlying issue.
 To avoid this problem you can invoke Python with '-m pip' instead of running pip directly.
 Looking in indexes: http://python.org/pypi/simple/
 Collecting docker-compose
...
Successfully installed PyYAML-5.4.1 attrs-20.3.0 bcrypt-3.2.0 cached-property-1.5.2 certifi-2020.12.5 cffi-1.14.5 chardet-4.0.0 cryptography-3.4.6 distro-1.5.0 docker-4.4.3 docker-compose-1.28.4 dockerpty-0.4.1 docopt-0.6.2 idna-2.10 importlib-metadata-3.6.0 jsonschema-3.2.0 paramiko-2.7.2 pycparser-2.20 pynacl-1.4.0 pyrsistent-0.17.3 python-dotenv-0.15.0 requests-2.25.1 six-1.15.0 texttable-1.6.3 typing-extensions-3.7.4.3 urllib3-1.26.3 websocket-client-0.57.0 zipp-3.4.0
```

7. 설치 완료!

```
$ pip list| grep docker
 docker             4.4.3
 docker-compose     1.28.4
 dockerpty          0.4.1
```

QEMU tcp원격접속 허용하기

1. libvirt.conf 수정 (주석해제)

```
$> vi /etc/libvirt/libvirtd.conf
listen_tls = 0 (기본값은 TLS통신)
listen_tcp = 1 (기본값은 비활성화)
tcp_port = "16509" (TCP열었을때 사용하는 포트)
listen_addr = "0.0.0.0" 허용할 IP (기본값은 모두 허용)
mdns_name = "VMHOST-001" (MDNS환경에서 해당정보, 이 값은 같은 네트워크 안에 있는 서버들과 달라야 함)
unix_sock_group = "libvirt" (Qemu관리그룹, 기본은 root이기 때문에 보안상 위험;...)
unix_sock_rw_perms = “0770” (TCP통신시 적용받은 퍼미션값)
```

2. /etc/sysconfig/libvirtd 수정 (주석해제)

```
$> vi /etc/sysconfig/libvirtd
...
LIBVIRTD_ARGS="–listen"
...
```

3. TLS통신을 안할경우 인증정보가 없다는 오류를 회피하기 위한 방법

```
$> vi /etc/sysconfig/libvirtd
#KRB5_KTNAME=/etc/libvirt/krb5.tab
#KRB5_KTNAME값 주석처리
```

4. 사용자 인증구성

```
$> groupadd libvirt
$> saslpasswd2 -a libvirt svc
```

5. 서비스 시작

```
$> /etc/init.d/libvirt start
```

VMware GuestOS에서 마우스 5버튼 작동 안될 경우 조치방법

vmware guestOS에서 마우스 5버튼 작동이 안될 경우 조치방법 (일반적인 마우스 버튼은 작동하고 있고, 마우스 자체가 앞으로, 뒤로 등 5버튼이 있는 마우스에 한함.)

1. host에서 해당 VM이 설치된 경로에서 *.vmx 파일을 메모장으로 오픈

2. 내용중에 아래 내용 추가

```
...
usb.generic.allowHID = "TRUE"
mouse.vusb.enable = "TRUE"
mouse.vusb.useBasicMouse = "FALSE"
...
```

3. GuestOS종료 후 재실행 한다음에 마우스 5버튼 작동 여부 확인

reference

- https://superuser.com/questions/35830/back-forward-mouse-buttons-do-not-work-in-vmware-workstation-6-5-guest-os

rancher 패스워드 초기화 방법

rancher를 설치하고 나서 root사용자 패스워드 초기화가 필요한 경우 수행하면 됩니다.

작업방법

1. rancher pod 정보 확인

```
$> kubectl get pod -A | grep rancher
cattle-system                rancher-c56764479-nt7nb                  1/1     Running   3 (3d2h ago)    45d
```

2. rancher pod가 어느노드에서 떴는지 확인

```
$> kubectl describe pod rancher-c56764479-nt7nb -n cattle-system
Name:                 rancher-c56764479-nt7nb
Namespace:            cattle-system
Priority:             1000000000
Priority Class Name:  rancher-critical
Node:                 test2/192.168.0.25
Start Time:           Thu, 21 Sep 2023 00:51:03 +0900
Labels:               app=rancher
                      pod-template-hash=c56764479
                      release=rancher
Annotations:          cni.projectcalico.org/containerID: f44aa8d23b68a7376bc29e0bc66605670dc5a87daa9c668bd4f241aa4b63b492
                      cni.projectcalico.org/podIP: 10.233.64.107/32
                      cni.projectcalico.org/podIPs: 10.233.64.107/32
Status:               Running
IP:                   10.233.64.107
IPs:
  IP:           10.233.64.107
Controlled By:  ReplicaSet/rancher-c56764479
```

3. rancher 컨테이너 정보 확인 (test2노드에서 실행)

```
$> crictl ps | grep rancher
a68dc79ee1d8c       81ee0878ffcdc       26 minutes ago      Running             rancher                   3                   f44aa8d23b68a       rancher-c56764479-nt7nb
```

4. rancher 컨테이너로 패스워드 초기화 (test2노드에서 실행)

```
$> crictl exec a68dc79ee1d8c reset-password
W1104 17:53:31.241158     386 client_config.go:617] Neither --kubeconfig nor --master was specified.  Using the inClusterConfig.  This might not work.
New password for default admin user (user-q2kl4):
123123
```

reference

- https://nevido.tistory.com/740
-