**Clair安装**

Clair：

$ curl -L https://raw.githubusercontent.com/coreos/clair/master/contrib/compose/docker-compose.yml -o $HOME/docker-compose.yml

$ mkdir /usr/src/clair-2.0.1/clair\_config

$ curl -L https://raw.githubusercontent.com/coreos/clair/master/config.yaml.sample -o /usr/src/clair-2.0.1/clair\_config/config.yaml

$ $EDITOR /usr/src/clair-2.0.1/clair\_config/config.yaml # Edit database source to be postgresql://postgres:password@postgres:5432?sslmode=disable

$ diff /usr/src/clair-2.0.1/clair\_config/config.yaml /usr/src/clair-2.0.1/clair\_config/config.yaml.org

23,24c23

< #source: host=localhost port=5432 user=postgres sslmode=disable statement\_timeout=60000

< source: postgresql://postgres:password@postgres:5432?sslmode=disable

---

> source: host=localhost port=5432 user=postgres sslmode=disable statement\_timeout=60000

$ $EDITOR $HOME/usr/src/clair-2.0.1/docker-compose.yml # Edit image source to be quay.io/coreos/clair:v2.0.1

$ diff /usr/src/clair-2.0.1/docker-compose.yml /usr/src/clair-2.0.1/docker-compose.old.yml

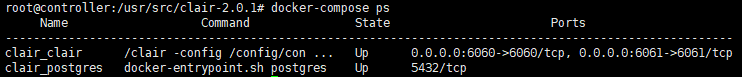
< image: quay.io/coreos/clair-git:latest

---

> image: quay.io/coreos/clair:v2.0.1

$ docker-compose -f /usr/src/clair-2.0.1/docker-compose.yml up –d

$ docker-compose ps



Cliarctl：

# curl -L https://raw.githubusercontent.com/jgsqware/clairctl/master/install.sh | sh

# clairctl version



# clairctl health



**Clair API**

**layers：**

上传layer：

# curl -s -H "Content-Type: application/json" -X POST -d '{

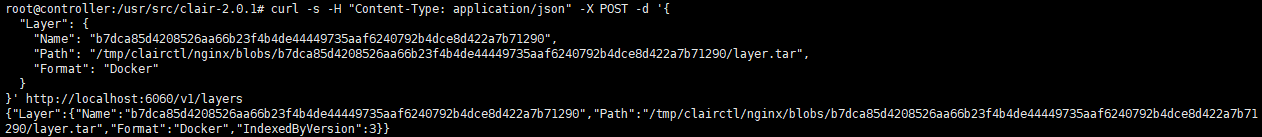
"Layer": { "Name":"b7dca85d4208526aa66b23f4b4de44449735aaf6240792b4dce8d422a7b71290",

"Path":"/tmp/clairctl/nginx/blobs/b7dca85d4208526aa66b23f4b4de44449735aaf6240792b4dce8d422a7b71290/layer.tar",

"Format": "Docker"

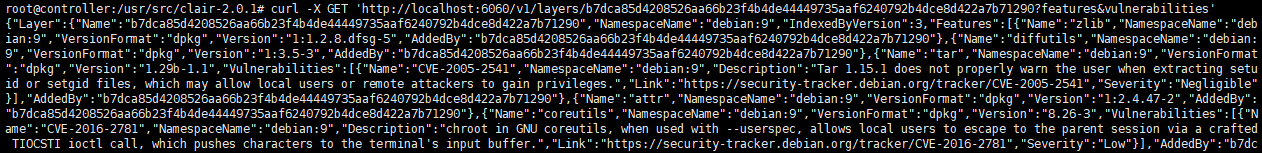
}

}' <http://localhost:6060/v1/layers>



查询layer信息：

# curl -X GET curl -X GET 'http://localhost:6060/v1/layers/b7dca85d4208526aa66b23f4b4de44449735aaf6240792b4dce8d422a7b71290?features&vulnerabilities'



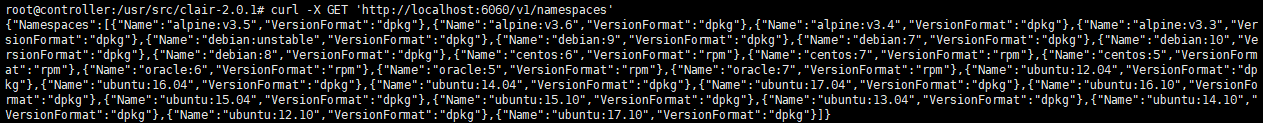
删除layer：

# curl -X DELETE 'http://localhost:6060/v1/layers/b7dca85d4208526aa66b23f4b4de44449735aaf6240792b4dce8d422a7b71290'



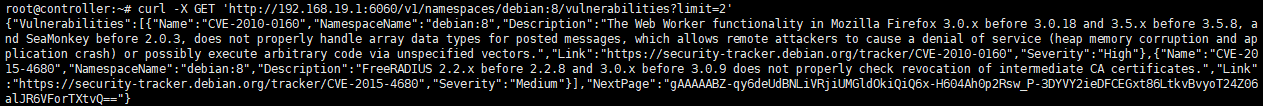
查询所有容器中的操作系统信息：

# curl -X GET 'http://localhost:6060/v1/namespaces'



查询指定操作系统的漏洞信息：

# curl -X GET 'http://localhost:6060/v1/namespaces/debian:8/vulnerabilities?limit=2'



添加新的漏洞信息(包含补丁信息)至指定OS：

# curl -s -H "Content-Type: application/json" -X POST -d '{

"Vulnerability": {

"Name": "CVE-2014-9471",

"NamespaceName": "debian:8",

"Link": "https://security-tracker.debian.org/tracker/CVE-2014-9471",

"Description": "The parse\_datetime function in GNU coreutils allows remote attackers to cause a denial of service (crash) or possibly execute arbitrary code via a crafted date string, as demonstrated by the \"--date=TZ=\"123\"345\" @1\" string to the touch or date command.",

"Severity": "Low",

"Metadata": {

"NVD": {

"CVSSv2": {

"Score": 7.5,

"Vectors": "AV:N/AC:L/Au:N/C:P/I:P"

}

}

},

"FixedIn": [

{

"Name": "coreutils",

"NamespaceName": "debian:8",

"Version": "8.23-1"

}

]

}

}' http://localhost:6060/v1/namespaces/debian:8/vulnerabilities

添加新的漏洞信息(不包含补丁信息)至指定OS：

# curl -s -H "Content-Type: application/json" -X PUT -d '{

"Vulnerability": {

"Name": "CVE-2014-9471",

"NamespaceName": "debian:8",

"Link": "https://security-tracker.debian.org/tracker/CVE-2014-9471",

"Description": "The parse\_datetime function in GNU coreutils allows remote attackers to cause a denial of service (crash) or possibly execute arbitrary code via a crafted date string, as demonstrated by the \"--date=TZ=\"123\"345\" @1\" string to the touch or date command.",

"Severity": "Low",

"Metadata": {

"NVD": {

"CVSSv2": {

"Score": 7.5,

"Vectors": "AV:N/AC:L/Au:N/C:P/I:P"

}

}

}

}

}' http://localhost:6060/v1/namespaces/debian:8/vulnerabilities/ CVE-2014-9471

查询指定的OS漏洞信息：

# curl -X GET 'http://localhost:6060/v1/namespaces/debian:8/vulnerabilities/CVE-2014-9471?fixedIn'

删除指定的OS漏洞信息：

# curl -X DELETE 'http://localhost:6060/v1/namespaces/debian%3A8/vulnerabilities/CVE-2014-9471'

查询fixes信息：

# curl -s -H "Content-Type: application/json" -X PUT -d '{

"Feature": {

"Name": "coreutils",

"NamespaceName": "debian:8",

"Version": "4.24-9"

}

}' http://localhost:6060/v1/namespaces/debian:8/vulnerabilities/CVE-2014-9471/fixes/coreutils

添加新的补丁信息至指定漏洞：

# curl -X GET 'http://192.168.19.1:6060/v1/namespaces/debian%3A8/vulnerabilities/CVE-2014-9471/fixes'

删除指定漏洞的补丁：

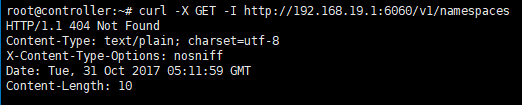
# curl -X DELETE 'http://localhost:6060/v1/namespaces/debian%3A8/vulnerabilities/CVE-2014-9471/fixes/coreutils'

查询指定OS的更新信息：

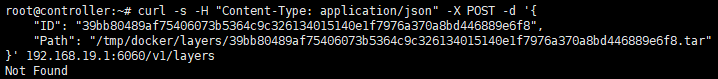
# curl -X GET 'http://localhost:6060/v1/notifications/ec45ec87-bfc8-4129-a1c3-d2b82622175a?limit=2'

**Clair问题解决**

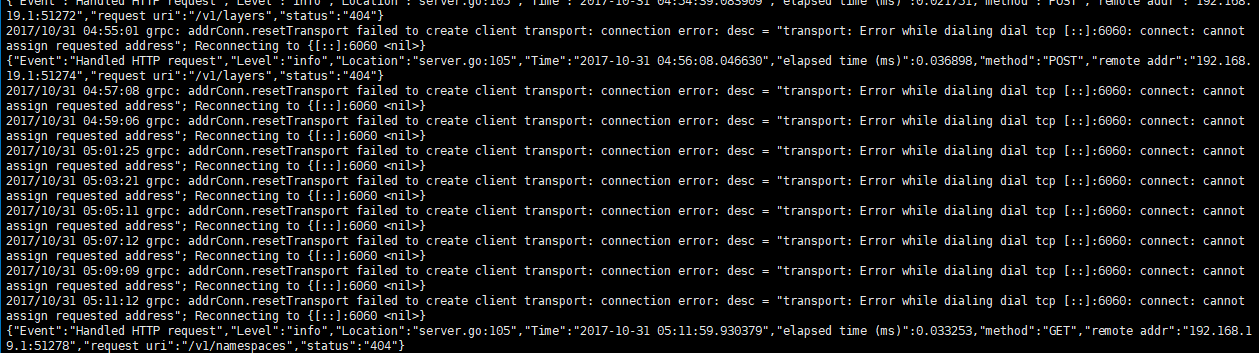
Clair访问Restful API返回404异常：



上传layer也是显示404：



Docker logs –f clair\_clair：



显示两个错误：

grpc: addrConn.resetTransport failed to create client transport: connection error: desc = "transport: Error while dialing dial tcp [::]:6060: connect: cannot assign requested address"; Reconnecting to {[::]:6060 <nil>}

{"Event":"Handled HTTP request","Level":"info","Location":"server.go:105","Time":"2017-10-31 05:11:59.930379","elapsed time (ms)":0.033253,"method":"GET","remote addr":"192.168.19.1:51278","request uri":"/v1/namespaces","status":"404"}

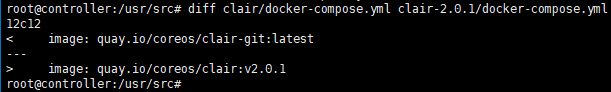
在Github的issues上找到解决方法

<https://github.com/coreos/clair/issues/474>



问题是在最新版Clair支持的gRPC API不稳定导致的，提议将Clair版本降至v2.0.1

修改docker-compose.yml，将镜像源修改：



Docker-compose up –d正常运行

后来发现Clair主要是面向企业的，主流用法是从私建的镜像仓库获取layers的tar文件，保存在Clair服务端，然后对其进行扫描。

上传layers至Clair时，若当前layer不是最底层的，则需要加上parentName参数指明parent layer，这样Clair才能确定各个layer之间的关系。

比如一个镜像有3个layers（C->B->A），必须上传所有layer，若只上传B layer和C layer，那么当访问查询B或C的信息时，将不会有回显

