Análise de riscos de seguridade na emprega de contedores para HPC

Manuel Simón Novoa

Grao en Enxeñaría Informática

Universidade de Santiago de Compostela
Centro de Supercomputación de Galicia

manuel.simon@rai.usc.es

Traballo titorizado por María Purificación Cariñena Amigo e Carlos Fernández Sánchez

19 de xullo do 2018

Diferenciación entre Máquinas Virtuais e Contedores

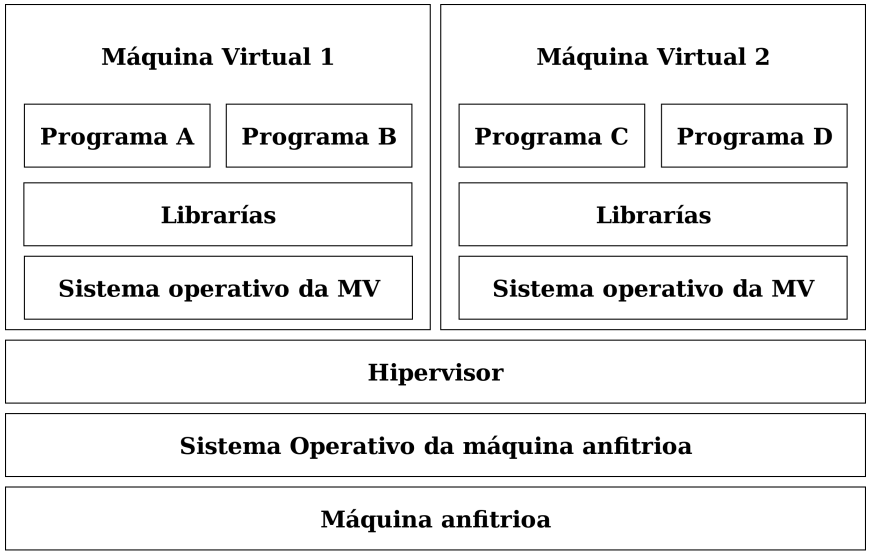


Figura: Funcionamento MVs

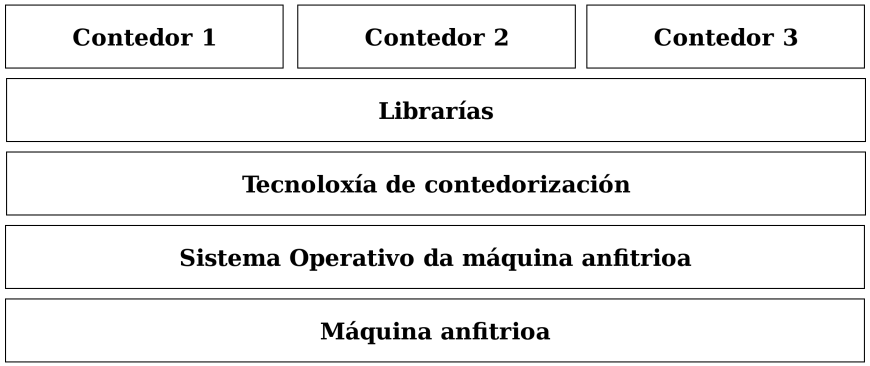
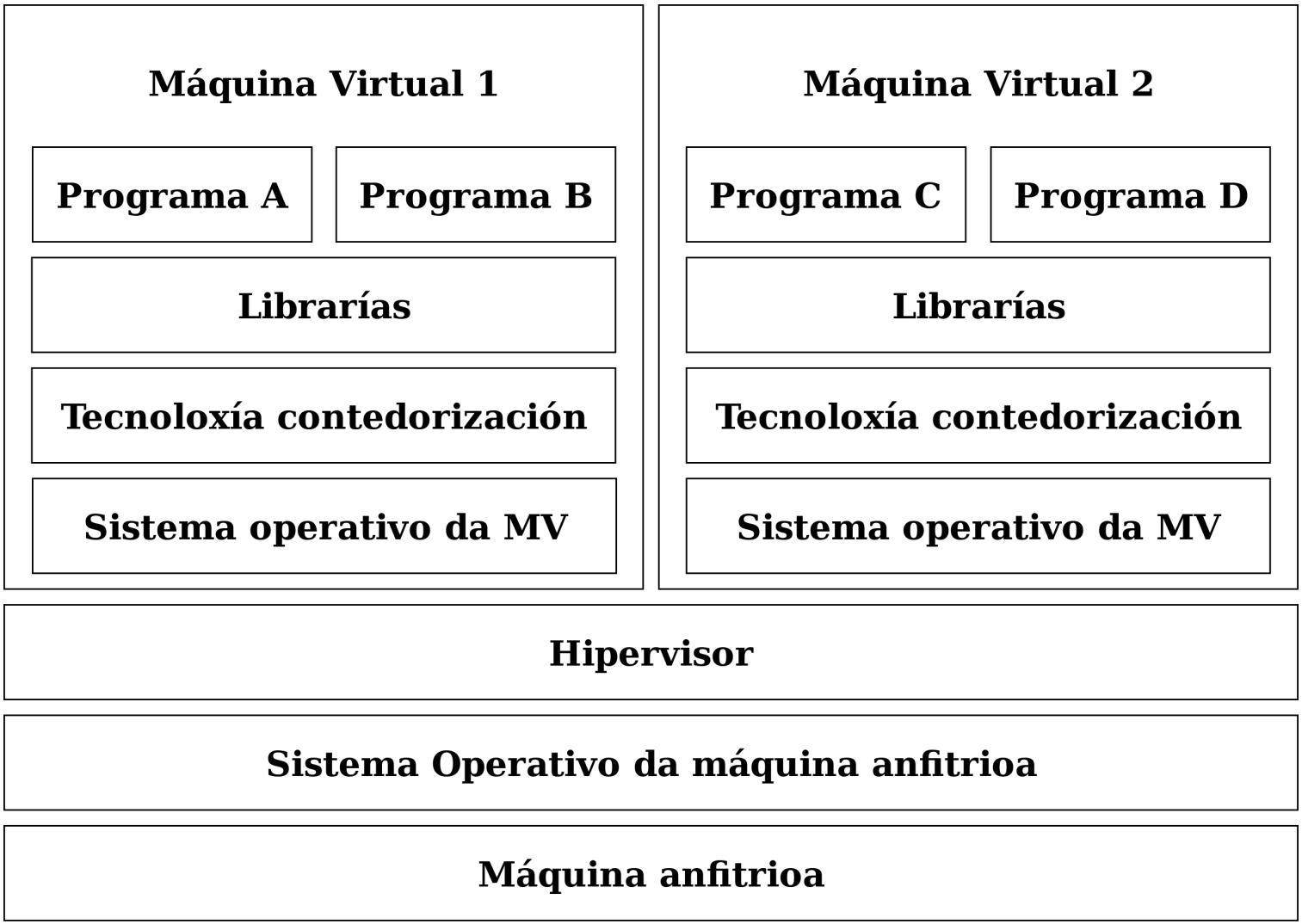


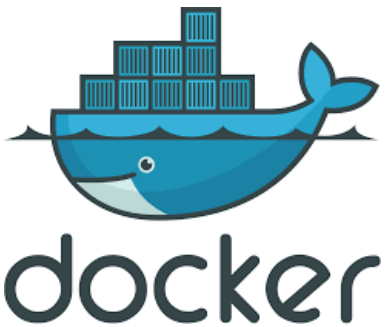
Figura: Funcionamento contedores

Combinacion de MVs e contedores



Obxectivos do proxecto

1. Analizar distintas implantacións de tecnoloxías de contedorización, dende o punto de vista da seguridade.



2. Determinar os cambios necesarios para poder empregar contedores nun entorno HPC de xeito seguro.

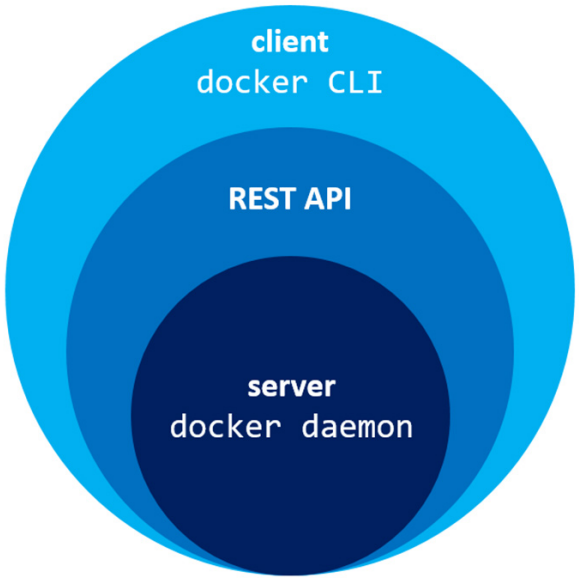
Docker

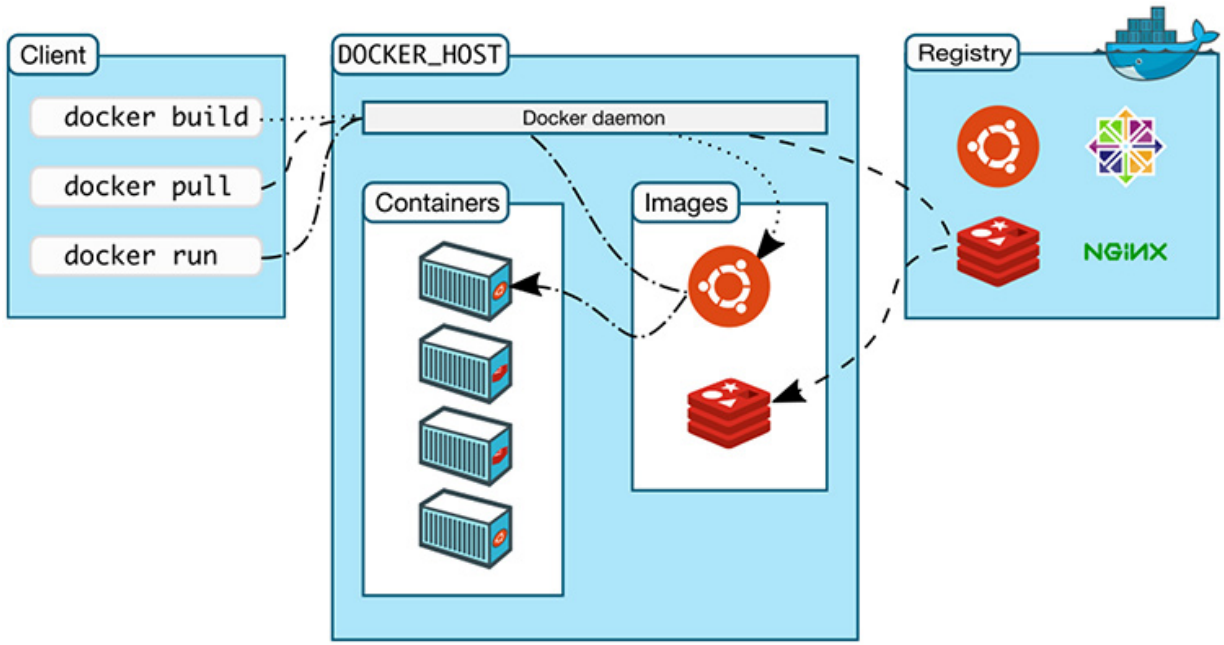
Filosofía

Usuarios autorizados facendo uso de contedores.

Funcionamento

Baseado no demo Docker, o cal precisa permisos de **superusuario**.





Singularity

Filosofía

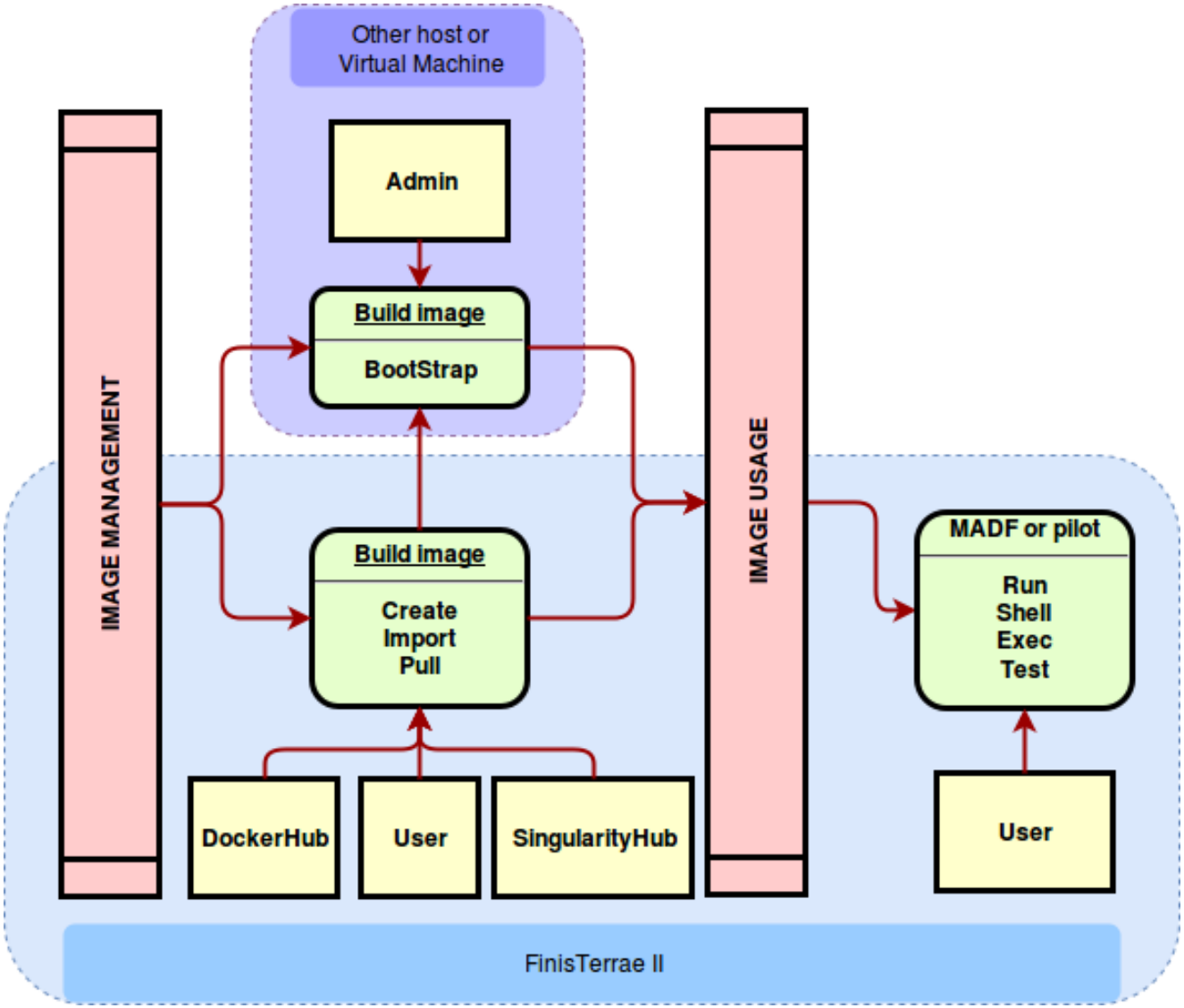
- Contedores non fiábeis executados por usuarios non fiábeis.
- Os privilexios dentro do contedor son os mesmos que fóra del.

Funcionamento

Dúas variantes:

1. SetUID.
   - Montaxe das imaxes en contedores.
   - Creación de espazos de nomes, coa axuda do *kernel*.
   - Compartición de directorios coa máquina anfitrioa.
2. Espazos de nomes.

Singularity: Fluxo de traballo



Udocker

Filosofía

- Contedores non fiábeis executados por usuarios non fiábeis.
- Redución ao mínimo absoluto do traballo a desenvolver polo administrador de sistemas.

Funcionamento

- Ferramenta escrita en Python:
  - Non é precisa ningunha instalación.
  - Non son precisos permisos de superusuario.
  - Ten certas dependencias: PRoot, Fakechroot, runC, ...
- Entorno tipo "*chroot*", pero limitado.

Limitacións

Ao non existiren privilexios de superusuario involucrados, calquera operación que realmente precise ditos privilexios non será realizábel.

Detección de vulnerabilidades en imaxes

A simplicidade de despregamento pode supor un problema...

- Os contedores poden supor un novo vector de ataque do sistema.
- Multitude de librarías incorporadas no sistema, que poden incluir novas vulnerabilidades.
- As vulnerabilidades poden ser herdadas se estendemos imaxes.

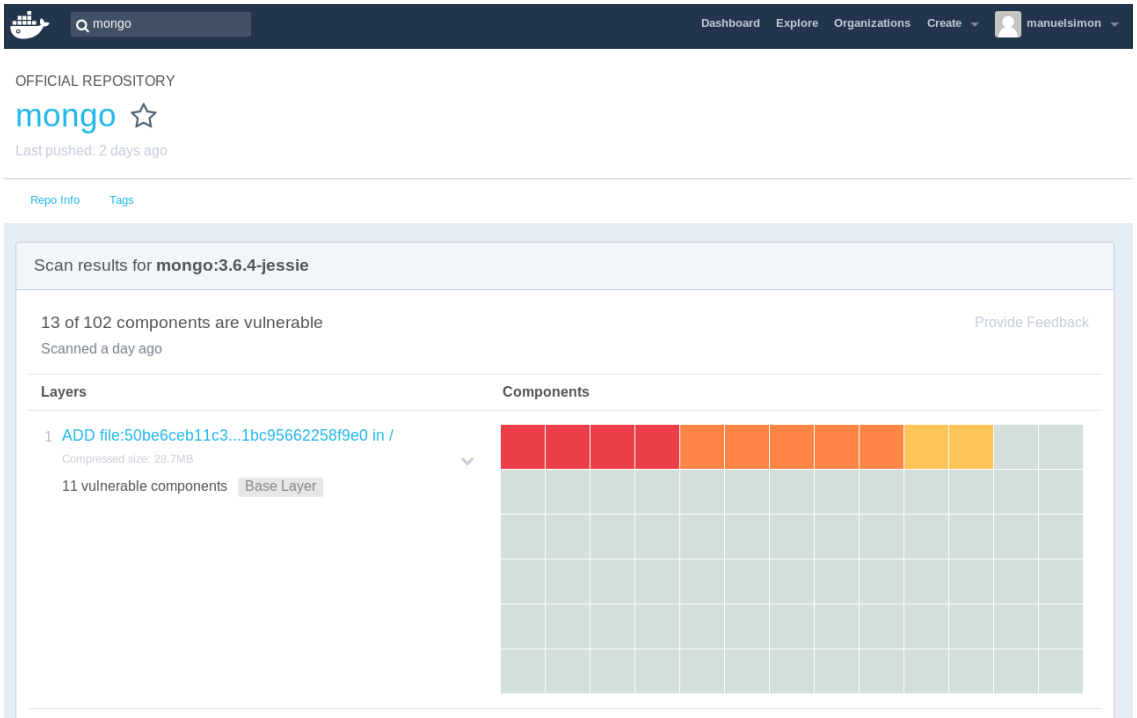
Solucións

- Debemos empregar soamente contedores obtidos de fontes fiábeis.
- Debemos posuír un mecanismo de detección de vulnerabilidades.

Detección de vulnerabilidades en imaxes: Docker Hub

Hai 2 tipos de repositorios:

1. **Non oficiais:** imaxes da comunidade.
   - Non presentan ningún mecanismo de seguridade.
2. **Oficiais:** imaxes cedidas polos propios desenvolvedores.
   - Son consideradas de maior confianza. (O son?)
   - O Docker Hub ofrece un servizo avanzado de seguridade.



Detección de vulnerabilidades en imaxes: Clair

Descrición

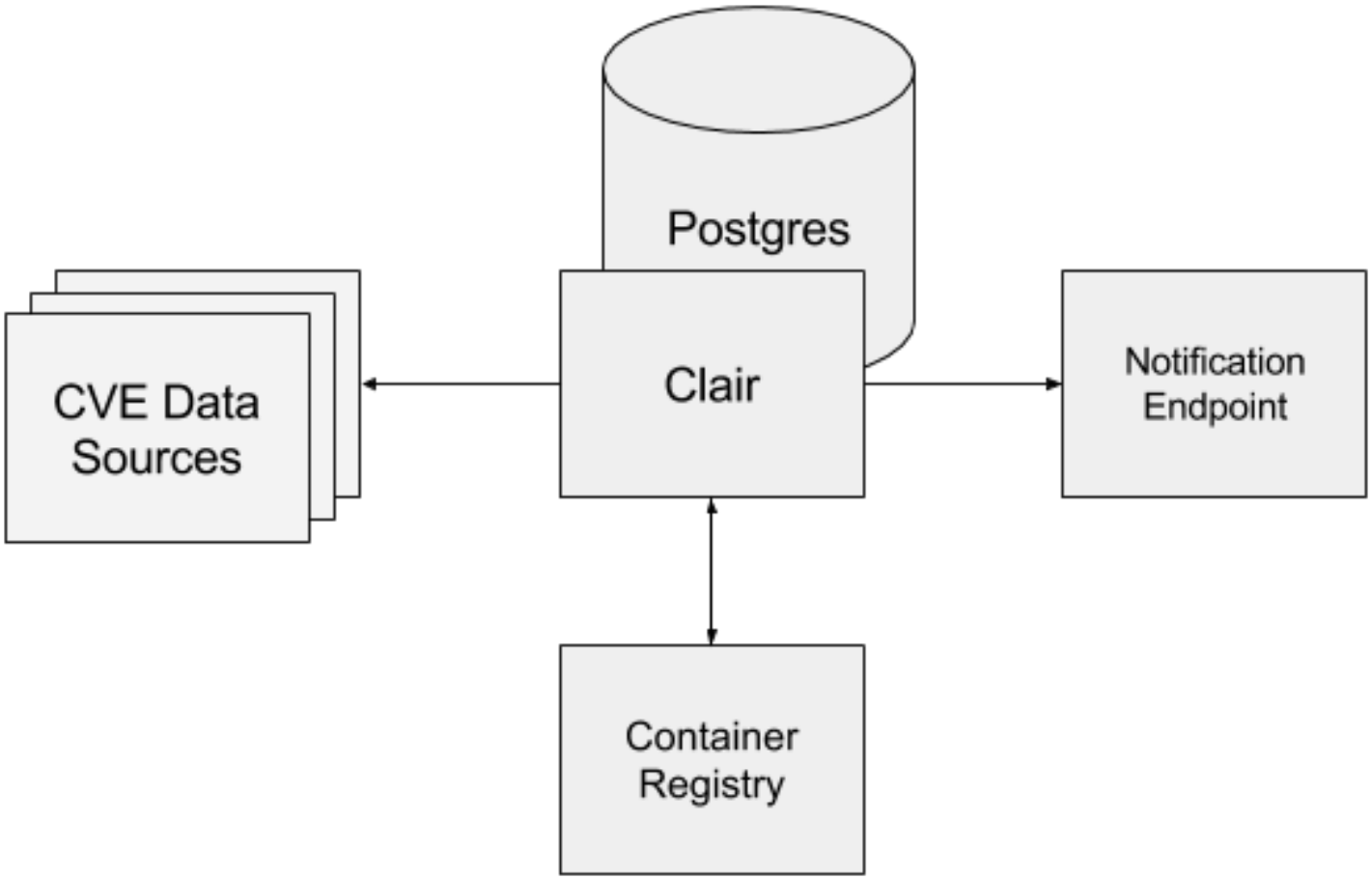
- Ferramenta para a análise **estática** de vulnerabilidades en contedores.
- Soporte para contedores Docker e Singularity.

Funcionamento

1. Datos sobre vulnerabilidades importados continuamente dende un conxunto coñecido de fontes.
2. Os datos son correlacionados cos contidos indexados das imaxes.
3. É xerada unha lista de vulnerabilidades para a imaxe analizada.

- O servizo pode ser completamente automatizado.

Arquitectura de Clair



Detección de vulnerabilidades en imaxes: Clair

Fluxo de traballo

1. O desenvolvedor envía unha imaxe a Clair.
2. Clair analiza a imaxe.
3. Clair devolve un informe detallado das vulnerabilidades atopadas.
4. O desenvolvedor actúa en base ao informe:
   - Procura solución ás vulnerabilidades.
   - Con coñecemento da súa existencia, asume os riscos.

Nome do campo	**Descrición**
Timestamp	Data exacta do análise
ID da vulnerabilidade	Identificador unívoco da vulnerabilidade no CVE
Nivel de gravidade	Clasificación da gravidade da vulnerabilidade
Descrición do CVE	Descrición de cada vulnerabilidade identificada
Paquetes asociados	Nome e versión exacta dos paquetes vulnerábeis
Identificador da capa	Indicador da capa onde reside a vulnerabilidade atopada

Detección de vulnerabilidades en imaxes: Clair

Exemplo execución Clair

```
root@kali:~/go/src/clair-scanner# ./clair-scanner --ip 172.17.0.1 mongo
2018/06/20 13:48:47 [INFO] ▶ Start clair-scanner
2018/06/20 13:48:49 [INFO] ▶ Server listening on port 9279
2018/06/20 13:48:49 [INFO] ▶ Analyzing 7c050f631abe0643817d0bacb3c5731b3c6d6c4c957e89db5c3f3b179409bfe0
2018/06/20 13:48:49 [INFO] ▶ Analyzing e1472d8833b6d676115573dd41bdec0aeda00b48905aa9df3ece331befa59919
2018/06/20 13:48:49 [INFO] ▶ Analyzing b93b8ccf52d5e751da002fe5735b396fb93e51f2628d0433dd8ff5d75a31147a
2018/06/20 13:48:49 [INFO] ▶ Analyzing 9f14f29484b6ebe011a1e34f90a2c36006ca006576177bb28cc95c3687897a1f
2018/06/20 13:48:49 [INFO] ▶ Analyzing 2b0b73b390e227886baa112c53a42f25d54d55caed6516513bd82a9117377f68
2018/06/20 13:48:49 [INFO] ▶ Analyzing 64b4c5c04262f09440d4bb50a4ec0d0f53bf74bb7c798d13c295aae3f86bc8e8
2018/06/20 13:48:49 [INFO] ▶ Analyzing 80f2d06cd0a202753704875098d3ed435d90bd5ed7181a99bec7318791317342
2018/06/20 13:48:49 [INFO] ▶ Analyzing 76173dce689fef8cd095aa206ca0e39a352fa1bc1e4fe2cff96c6d4c4cd75a54
2018/06/20 13:48:49 [INFO] ▶ Analyzing 3fb4116f188479811f5cacb336613e61a66630ed00a48ae5e013a53218259913
2018/06/20 13:48:49 [INFO] ▶ Analyzing 4c8fe2dc0aaf3ddb3211ee77bdad6125921f53dcd9828411dece94177bd474ae
2018/06/20 13:48:49 [WARN] ▶ Image [mongo] contains 73 total vulnerabilities
2018/06/20 13:48:49 [ERRO] ▶ Image [mongo] contains 73 unapproved vulnerabilities
+------------+--------------------------+--------------+-----------------------+-----------------------------------------------------------------+
| STATUS     | CVE SEVERITY             | PACKAGE NAME | PACKAGE VERSION       | CVE DESCRIPTION                                                 |
+------------+--------------------------+--------------+-----------------------+-----------------------------------------------------------------+
| Unapproved | High CVE-2016-2779       | util-linux   | 2.25.2-6              | runuser in util-linux allows local users to escape to           |
|            |                          |              |                       | the parent session via a crafted TIOCSTI ioctl call,            |
|            |                          |              |                       | which pushes characters to the terminal's input buffer.         |
|            |                          |              |                       | https://security-tracker.debian.org/tracker/CVE-2016-2779       |
+------------+--------------------------+--------------+-----------------------+-----------------------------------------------------------------+
| Unapproved | High CVE-2018-6797       | perl         | 5.20.2-3+deb8u10      | An issue was discovered in Perl 5.18 through 5.26.              |
|            |                          |              |                       | A crafted regular expression can cause a heap-based             |
|            |                          |              |                       | buffer overflow, with control over the bytes written.           |
|            |                          |              |                       | https://security-tracker.debian.org/tracker/CVE-2018-6797       |
+------------+--------------------------+--------------+-----------------------+-----------------------------------------------------------------+
| Unapproved | High CVE-2017-12424      | shadow       | 1:4.2-3+deb8u4        | In shadow before 4.5, the newusers tool could be                |
|            |                          |              |                       | made to manipulate internal data structures in ways             |
|            |                          |              |                       | unintended by the authors. Malformed input may lead             |
|            |                          |              |                       | to crashes (with a buffer overflow or other memory              |
|            |                          |              |                       | corruption) or other unspecified behaviors. This                |
|            |                          |              |                       | crosses a privilege boundary in, for example, certain           |
|            |                          |              |                       | web-hosting environments in which a Control Panel allows        |
|            |                          |              |                       | an unprivileged user account to create subaccounts.             |
|            |                          |              |                       | https://security-tracker.debian.org/tracker/CVE-2017-12424      |
+------------+--------------------------+--------------+-----------------------+-----------------------------------------------------------------+
```

Validación de imaxes

Docker

Docker content trust.

- Emprega dunha serie de chaves: Chave *offline*, chave das etiquetas, chaves do servidor (p.e. timestamp).

Singularity

- Mecanismo máis sinxelo: *Hashing* SHA-256.
- Agárdanse melloras coa versión 3 (actual 2.5.1).

Udocker

- Non ten ningún mecanismo propio de validación.
- Depende de ferramentas externas (p.e. Docker content trust).

Redes

Diferentes perspectivas da rede:

Singularity e Udocker

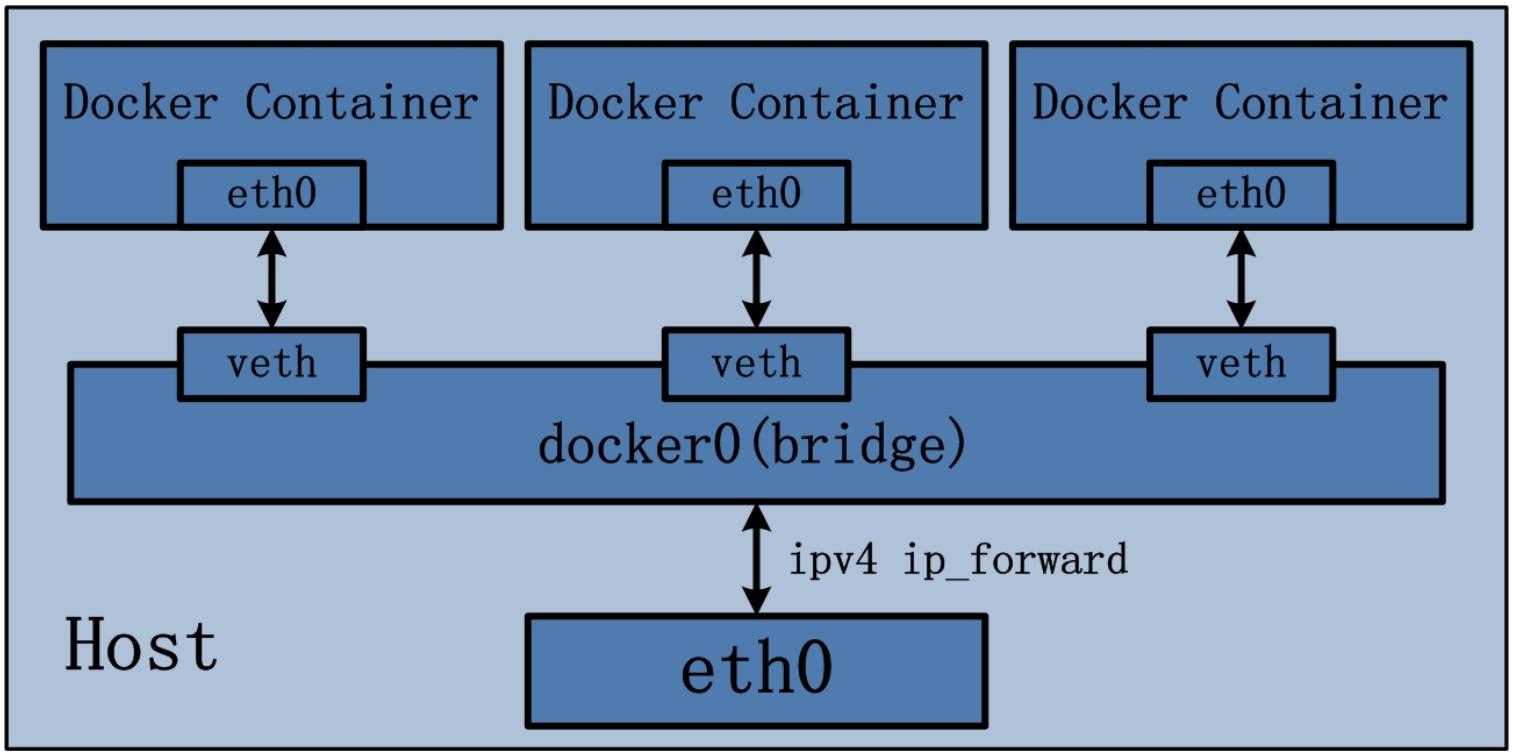
- Mesma rede que calquera outro proceso da máquina anfitrioa.
- Non emulan ningún paradigma de virtualización a nivel de hardware.
- Os usuario non teñen permisos para poder realizar:
  - Operacións complexas coa rede.
  - Ataques en rede.

Docker

- Modelo de rede composto por un subsistema de rede virtual.
- Coexisten diversos modelos: **bridge**, *host*, *overlay*, *macvlan*, *none*.

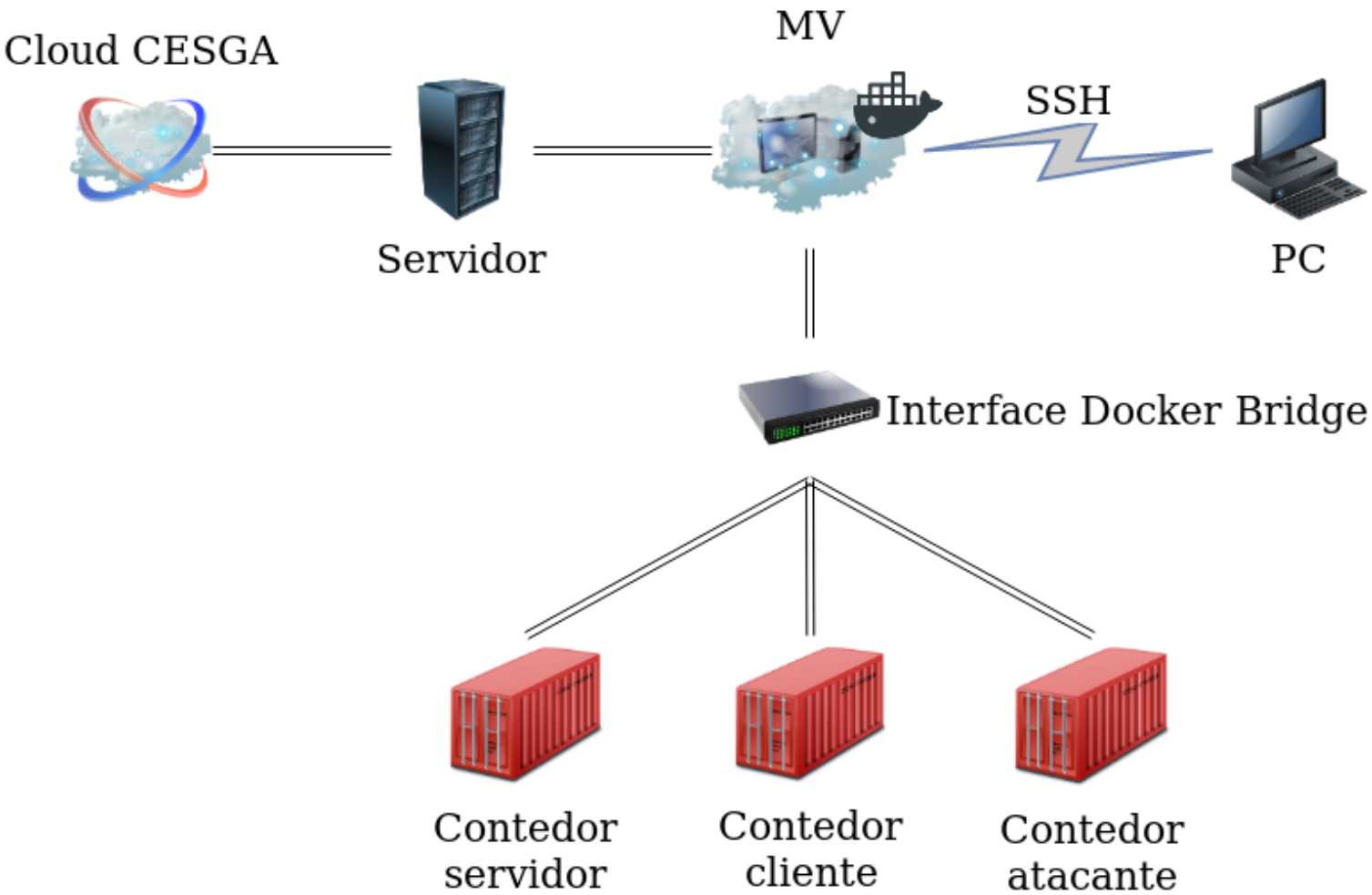
Modelo **bridge** de Docker

Dende o punto de vista arquitectónico, todos os contedores conectados en rede nun anfitrión Docker mediante unha interface *bridge* son equivalentes a máquinas físicas conectadas mediante un *switch Ethernet* común.



Redes: Explotación de vulnerabilidades de rede Docker

- ARP *spoofing*.
- *Man-in-the-middle*.
- MAC *flooding*.



ARP *spoofing*

Táboas ARP previo o ataque (perspectiva dende o servidor)

Contedor	**Enderezo IP**	**Enderezo físico**
Atacante	172.18.0.4	02:42:ac:12:00:04
Cliente	172.18.0.2	02:42:ac:12:00:02
Servidor	172.18.0.3	02:42:ac:12:00:03

Táboas ARP despois do ataque (perspectiva dende o servidor)

Contedor	**Enderezo IP**	**Enderezo físico**
Atacante	172.18.0.4	02:42:ac:12:00:04
Cliente	172.18.0.2	02:42:ac:12:00:04
Servidor	172.18.0.3	02:42:ac:12:00:03

Man-in-the-middle

Petición dunha web dende contedor cliente a contedor servidor)

```
root@20136c4451bb :/# curl nginx
```

O contedor atacante recibe o envío do servidor (*man-in-the-middle*)

```
root@9f3f255ace4a :/# urlsnarf -i eth0
urlsnarf: listening on eth0 [tcp port 80 or port 8080 or
port 3128]
arpspoof_ubuntu_1.arpspoof_default - - [06/Apr/2018:08:57:08
+0000] - GET http://nginx/ HTTP/1.1 - curl/7.47.0
```

MAC *flooding*

Recibimento de paquetes dende o contedor atacante

```
root@9403362d194e:/# tcpdump -i eth0 -X -vv
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
11:20:48.805365 IP (tos 0x0, ttl 64, id 59215, offset 0, flags [none], proto TCP (6), length 20)
    grabner.ceer.utexas.edu > [redacted]: [|tcp]
        0x0000:  4500 0014 e74f 0000 4006 9492 8174 942d  E....O..@....t.-
        0x0010:  e9e6 ff79 7bbb 0b12 4034 17e5 0000 0000  ...y{...@4......
        0x0020:  5002 0200 cff9 0000                      P.......
11:20:48.810621 IP (tos 0x0, ttl 64, id 11630, offset 0, flags [DF], proto UDP (17), length 74)
    9403362d194e.40892 > ns3.cesga.es.53: [bad udp cksum 0x90c2 -> 0xcb2c!] 5387 + PTR? [redacted].in-addr.arpa. (46)
        0x0000:  4500 004a 2d6e 4000 4011 7cba ac16 0003  E..J-n@.@.|.....
        [redacted]                                        ..".....5.6......
                                                          ........[redacted]
                                                          [redacted].in-addr
        0x0040:  0461 7270 6100 000c 0001                 .arpa.....
11:20:48.812309 IP (tos 0x0, ttl 64, id 58240, offset 0, flags [none], proto TCP (6), length 20)
    [redacted] > [redacted] [|tcp]
        0x0000:  4500 0014 e380 0000 4006 25a4 72db 1855  E.......@.%.r..U
        0x0010:  b256 3439 85ee 847f 744c 8a42 0000 0000  .V49....tL.B....
        0x0020:  5002 0200 3326 0000                      P...3&..
11:20:48.812754 IP (tos 0x0, ttl 64, id 37174, offset 0, flags [none], proto TCP (6), length 20)
    [redacted] > [redacted]: [|tcp]
        0x0000:  4500 0014 9136 0000 4006 1514 f752 7118  E....6..@....Rq.
        0x0010:  fbd8 7056 c38d c6d5 2126 581a 0000 0000  ..pV....!&X.....
        0x0020:  5002 0200 d5a4 0000                      P.......
11:20:48.813880 IP (tos 0x0, ttl 64, id 25066, offset 0, flags [none], proto TCP (6), length 20)
    [redacted] > [redacted]: [|tcp]
        0x0000:  4500 0014 61ea 0000 4006 654c a645 9731  E...a...@.eL.E.1
        0x0010:  9620 e016 89d1 f2f8 1917 2971 0000 0000  .........)q....
        0x0020:  5002 0200 3ae2 0000                      P...:...
11:20:48.814535 IP (tos 0x0, ttl 64, id 26507, offset 0, flags [none], proto TCP (6), length 20)
    a104-106-98-61.deploy.static.akamaitechnologies.com > c-68-55-96-55.hsd1.mi.comcast.net: [|tcp]
        0x0000:  4500 0014 678b 0000 4006 a443 686a 623d  E...g...@..Chjb=
        0x0010:  4437 6037 e8fc 2db9 6dd9 854a 0000 0000  D7`7..-.m..J....
        0x0020:  5002 0200 34f3 0000                      P...4...
11:20:48.814922 IP (tos 0x0, ttl 64, id 23796, offset 0, flags [none], proto TCP (6), length 20)
    [redacted] > 2.148.70.51.tmi.telenormobil.no: [|tcp]
        0x0000:  4500 0014 5cf4 0000 4006 3df1 0fbe 877a  E...\...@.=....z
        0x0010:  0294 4633 e3b3 7c25 1d2d f0f0 0000 0000  ..F3..|%.-......
        0x0020:  5002 0200 5fec 0000                      P..._...
11:20:51.453234 IP (tos 0x0, ttl 64, id 38277, offset 0, flags [none], proto TCP (6), length 20)
    tgm-199-30-104-116-pinnaclenetworksolutions.com > [redacted]: [|tcp]
        0x0000:  4500 0014 9585 0000 4006 cb02 c71e 6874  E.......@.....ht
        0x0010:  d264 1865 6b47 8528 46df f0c4 0000 0000  .d.ekG.(F.......
        0x0020:  5002 0200 6b72 0000                      P...kr..
```

Limitación de recursos

Importancia da limitación de recursos

- Cuestión crítica.
- Un contedor malintencionado podería facer uso exhaustivo, perxudicando aos outros contedores ou á máquina anfitrioa (**DoS**).

Modelo de Docker

- Relega no *kernel*.
- Controis xa deseñados:
  - Espazos de nomes.
  - *Cgroups*.
- Non debemos influir no seu funcionamento (`--privileged`)

Modelo Singularity e Udocker

- Os contedores son unha aplicación máis.
- A limitación queda fóra do seu control.
- Son precisas ferramentas externas.
  - Xestores recursos (Slurm).

Limitación de recursos: CPU e memoria

Modelo de Docker

- OOM activado por defecto.
- Posíbel establecer límites de forma sinxela e segura:
  - `--memory`
  - `--cpus`
  - `--oom-kill-disable`

Modelo Singularity e Udocker

- Relegan en ferramentas externas.
- Caso do CESGA: Slurm.

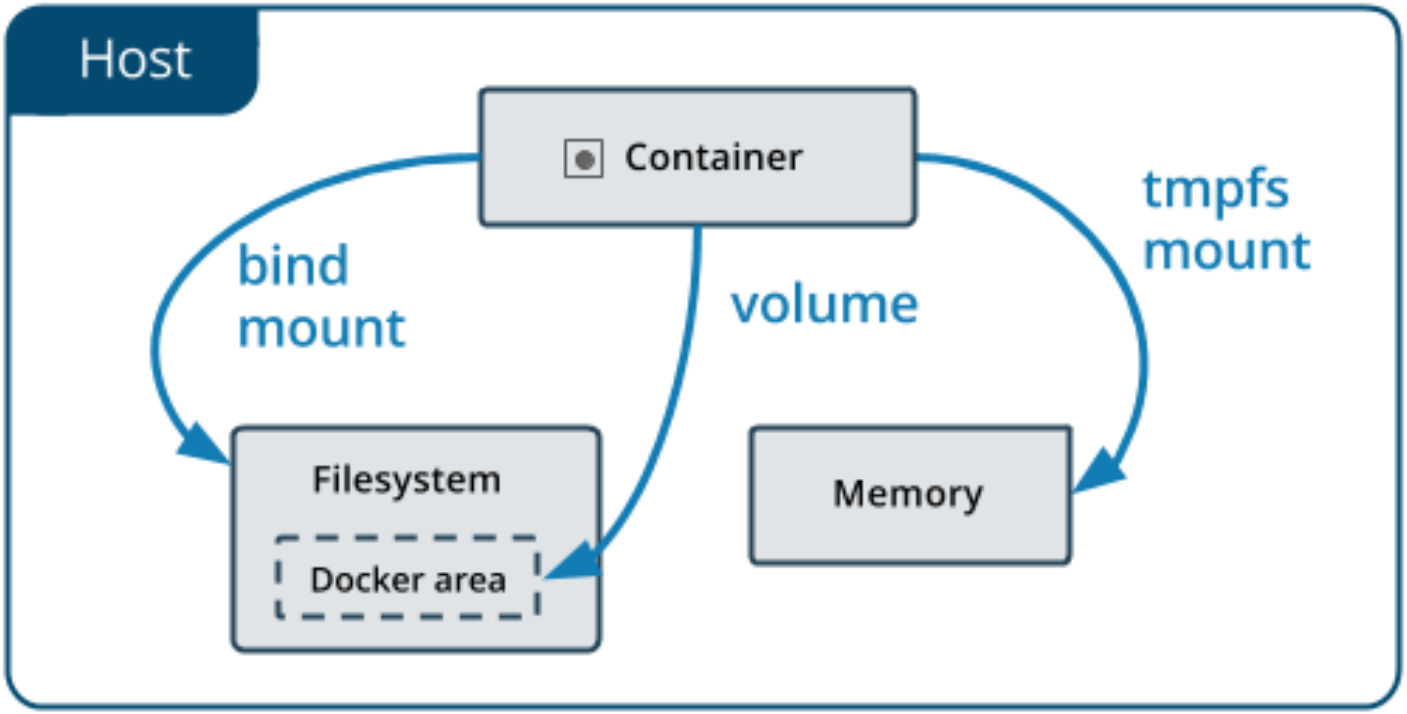
Limitación de recursos: Disco

Modelo de Docker

- As cotas de disco non son efectivas:
  - Demo root.
  - `CAP_SYS_RESOURCE`.

Modelo Singularity e Udocker

- Posto que o usuario dentro e fóra do contedor é o mesmo: **Cotas de disco**.



Limitación de recursos: Rede

Mesmo problema: Docker, Singularity e Udocker

- Pensadas para funcionar baixo "**filosofía do mellor esforzo**".
- Pode levar a ataques **DoS**.

Solución

- Emprega de **QoS**.
  - En función do usuario.
  - En función do contedor.

Outras propostas de mellora da seguridade

- Contedores correndo sobre MVs:
  - Perda de eficiencia.
  - Problemas en entornos de HPC.
- Emprega de capas externas de seguridade: SELinux.
- Manter actualizado o sistema:
  - Contedores.
  - Máquina anfitrioa, sempre que se poida.
- Realizar auditorías do sistema:
  - Están xurdindo ferramentas específicas para contedores (p.e. Docker *Bench Audit Tool*)

Conclusión de boas prácticas a seguir

- Manter só o esencial.
- Empregar sistemas de só lectura.
- Limitar as chamadas ao *kernel*.
- Limitar o uso dos recursos.
- Reducir as vulnerabilidades existentes nas imaxes.
- Manter actualizadas as tecnoloxías de virtualización e o sistema.
- Tratamento adecuado de información confidencial.
- Non dar acceso a directorios perigosos da máquina anfitrioa.
- Configurar ferramentas de xestión de OOM.
- Configurar o *socket* de Docker de xeito seguro (se se precisa conexión en rede).
- Contar con sistemas automatizados de detección de vulnerabilidades.
- Emprega de mecanismos externos de seguridade.

Traballo futuro

- Instalación da tecnoloxía Docker no FT2.
- Configuración da tecnoloxía Docker para traballar con ela de xeito máis seguro (controladores de disco adecuados, correcta habilitación dos espazos de nomes, seguir a auditoría do Docker Bench, ...)
- Posta en práctica das recomendacións acadadas e seguimento das mismas.
- Realización de maior número de probas: p.e. escalada de privilexios.

Análise de riscos de seguridade na emprega de contedores para HPC

Manuel Simón Novoa

Grao en Enxeñaría Informática

Universidade de Santiago de Compostela
Centro de Supercomputación de Galicia

manuel.simon@rai.usc.es

Traballo titorizado por María Purificación Cariñena Amigo e Carlos Fernández Sánchez

19 de xullo do 2018

Seguimento dos riscos

O seguimento dos riscos permitiu a detección da materialización dun dos riscos: **R14**.

Cadro: Ameaza A14

Identificador	A14
Nome	Caída dun dos portais de contedores a empregar.
Descrición	Un dos portais de onde se descargan as imaxes dos contedores a empregar para o desenvolvemento do proxecto cae, quedando inoperante e impedindo a obtención de imaxes importantes para o traballo.

Seguimento dos riscos

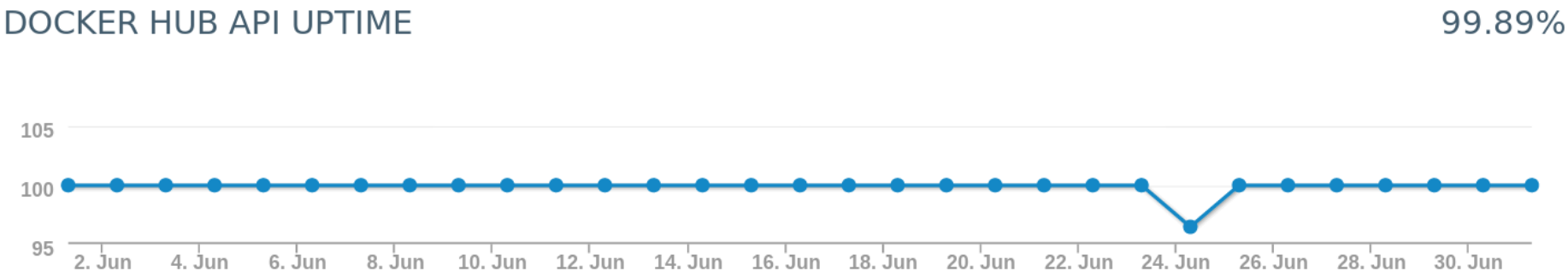
Cadro: Risco R14

Identificador	R14	**Tipo de risco**	De proxecto
Probabilidade	Baixa	**Impacto**	Serio
Exposición	Baixa	**Activos afectados**	ACT5
Tratamento	• **Aceptación:** posto que se trata dun servizo completamente externo, non é posíbel aplicar moitas medidas. Tentarase traballar coas imaxes xa almacenadas no equipo local, mais no caso de querer facer probas directamente co portal non haberá nada que facer, agás agardar á súa recuperación.		
Indicadores	Resulta imposíbel acceder a algún dos portais de imaxes de contedores cos que se traballa ao longo do proxecto (obtención de erro 404 ou similar).		

Seguimento dos riscos

Erro devolto polo Docker Hub

```
Error response from daemon: get
https://registry-1.docker.io/v2/library/...: received
unexpected HTTP status: 503 Service Unavailable.
```



Xestión da configuración

Elementos da configuración

- Memoria do proxecto.
- Código fonte dos diferentes *scripts*.
- Diagramas e figuras.

Control de versións

- Git + GitHub + GitKraken
- Traballo sobre a rama "*master*".

Metodoloxía de traballo

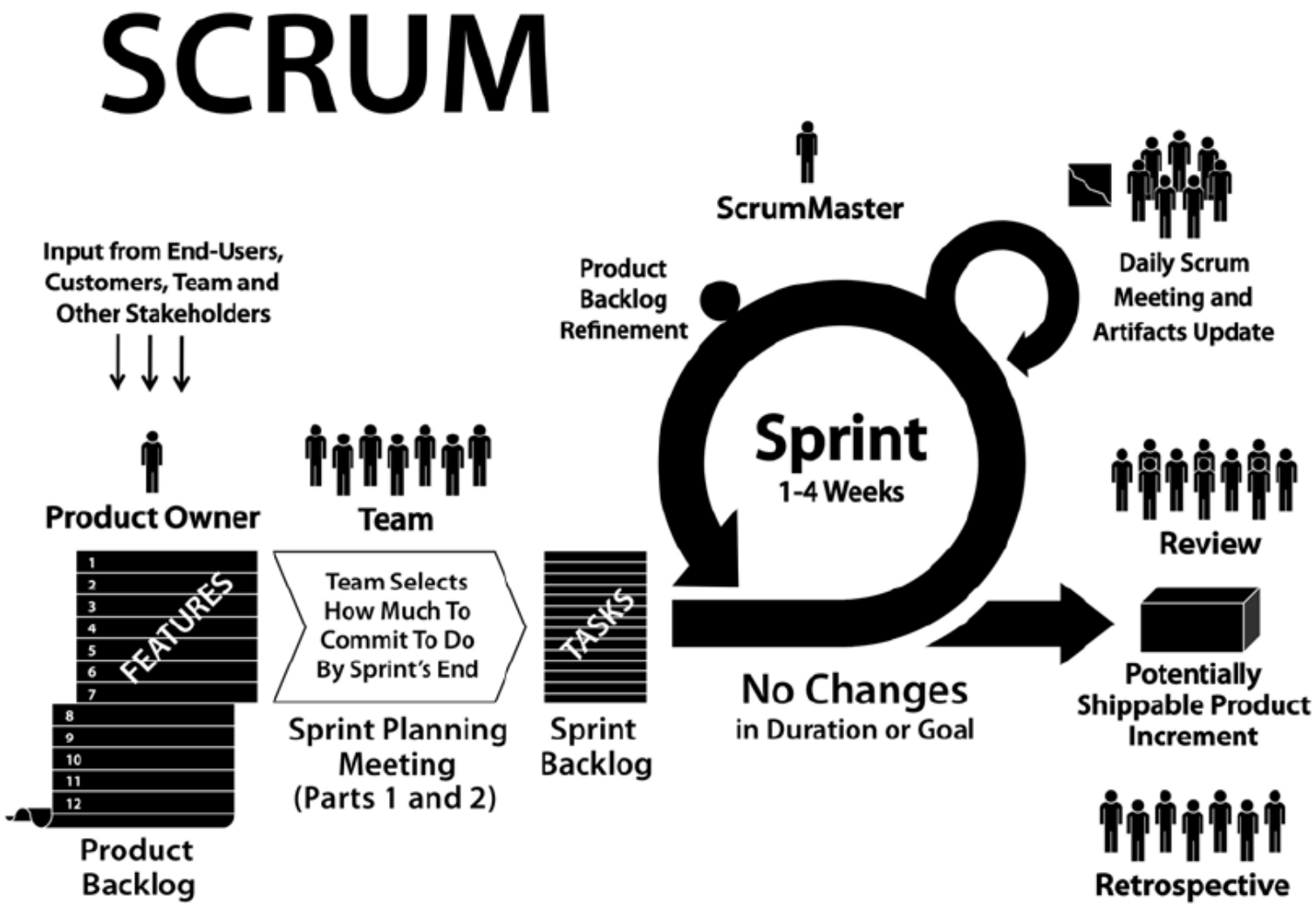
Elección da metodoloxía

- Grande nivel de incerteza.
- Necesidade de flexibilidade e tolerancia a cambios $\rightarrow$ Metodoloxía áxil.

Scrum

- Duración escollida dos *sprints*: 3 semanas.
- Realización de *sprint* diario.

Metodoloxía de traballo



Custos directos

Custos materiais

Nome	Custo
Equipo de traballo	55.56€
Material funxíbel	20€
Software	0€
Custos computacionais	1512€
TOTAL	**1587.56€**

Custos de persoal

Rol	Custo/hora	Nº horas	Custo total
Director do proxecto	22.39€	20	447.80€
Xefe do proxecto	18.86€	90	1697.40€
Asistente á investigación	10.01€	310	3103.10€
TOTAL			**5248.30€**

Cadro: Custo calculado do equipo de traballo

Meses de vida	Meses de uso no proxecto	Custo total	Custo no proxecto
72	4	1000€	55.56€

Custos

Custo directos totais

Tipo	**Custo asociado**
Material	1587.56€
Persoal	5248.30€
TOTAL	**6835.86€**

Custo indirectos

20 % dos custos directos → **1307.12€**.

Custos

Custo totais

Tipo	**Custo asociado**
Directo	6835.86€
Indirecto	1307.12€
TOTAL	**8143.03€**