

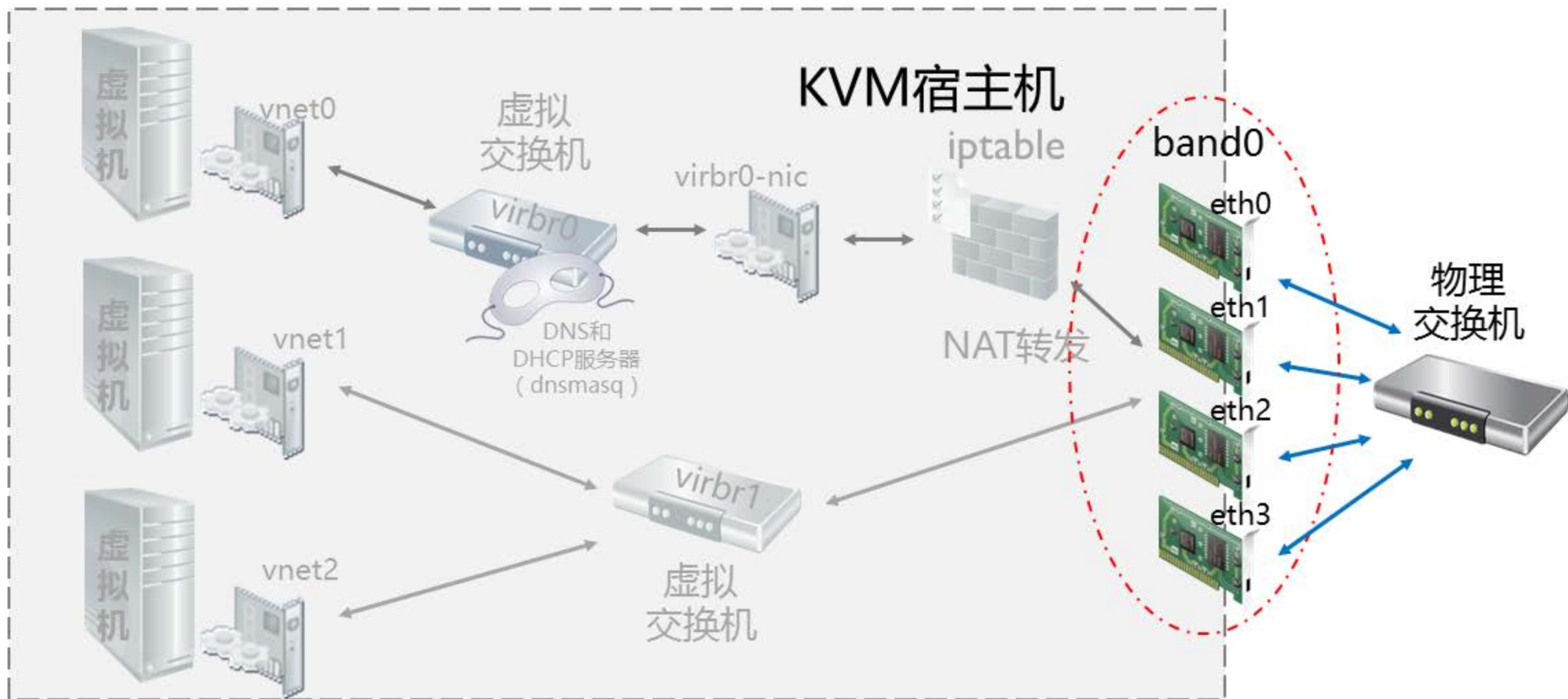
虚拟网络高级特性



概述

- ▶ ~~Linux网桥基本概念~~
- ▶ ~~qemu-kvm支持的网络~~
- ▶ ~~向虚拟机添加虚拟网络连接~~
- ▶ ~~虚拟网络配置~~
 - ▶ ~~基于NAT的虚拟网络~~
 - ▶ ~~基于网桥的虚拟网络~~
 - ▶ ~~用户自定义的隔离的虚拟网络~~
- ▶ 多物理网卡绑定
- ▶ 配置VLAN
- ▶ 通过网络过滤提高安全性

多物理网卡绑定



实验：配置多网卡绑定的KVM桥接模式

▶ 绑定网卡

1. 启用Bonding
2. 配置物理网卡
3. 配置绑定接口
4. 重新启动服务
5. 测试

▶ 配置网桥

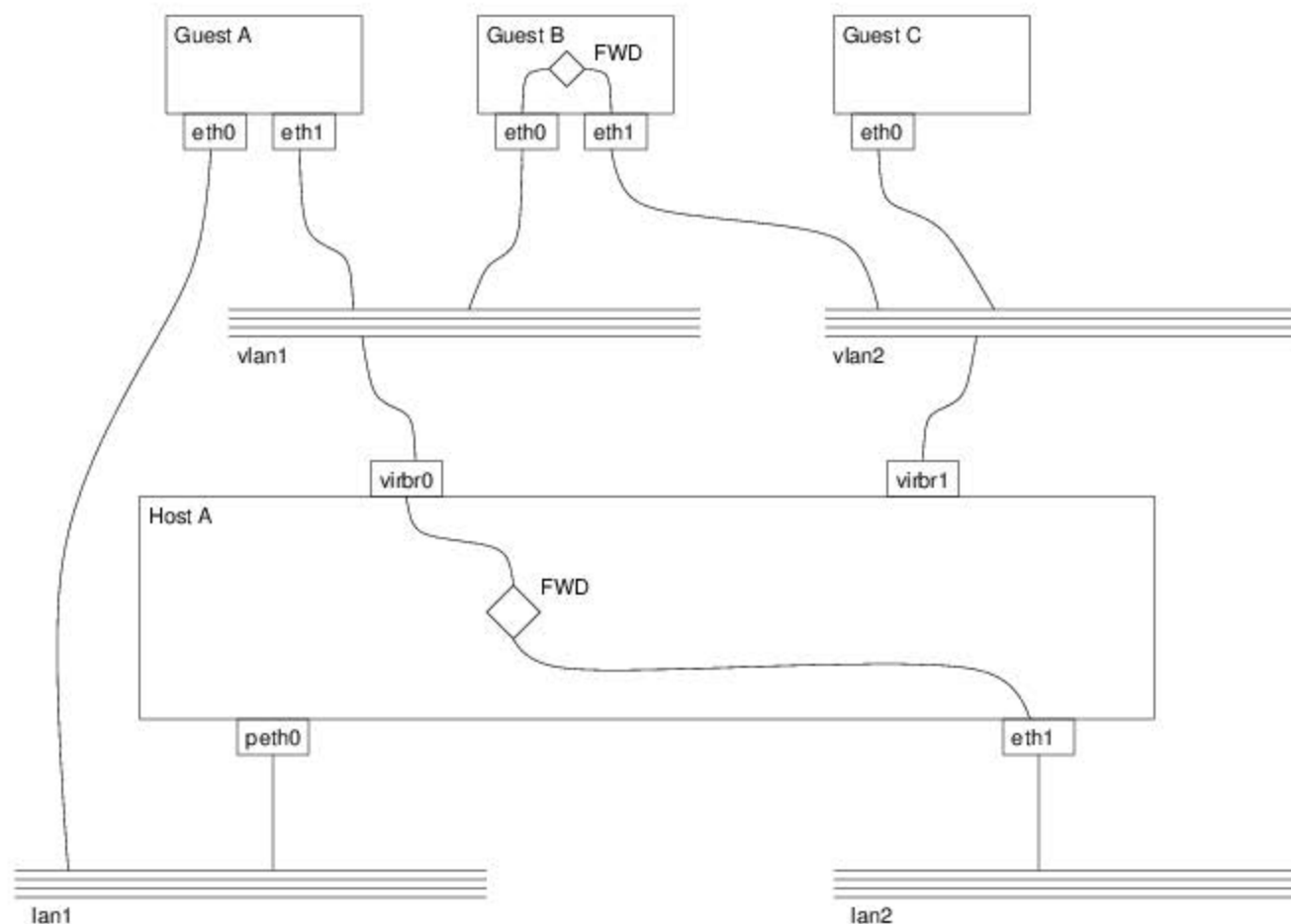
```
# vi ifcfg-bond0
DEVICE=bond0
ONBOOT=yes
NM_CONTROLLED=no
USERCTL=no
BONDING_OPTS="mode=1 miimon=100"
BOOTPROTO=static
IPADDR=192.168.200.11
NETMASK=255.255.255.0
```

```
mode=0 (Balance Round Robin)
mode=1 (Active backup)
mode=2 (Balance XOR)
mode=3 (Broadcast)
mode=4 (802.3ad)
mode=5 (Balance TLB)
mode=6 (Balance ALB)
```

```
# brctl show
bridge name      bridge id        STP enabled      interfaces
virbr0           8000.5254008dc0c2  yes              virbr0-nic
virbr1           8000.000c2942ae81  no               bond0
                                                           vnet0
                                                           vnet1
                                                           vnet2
```

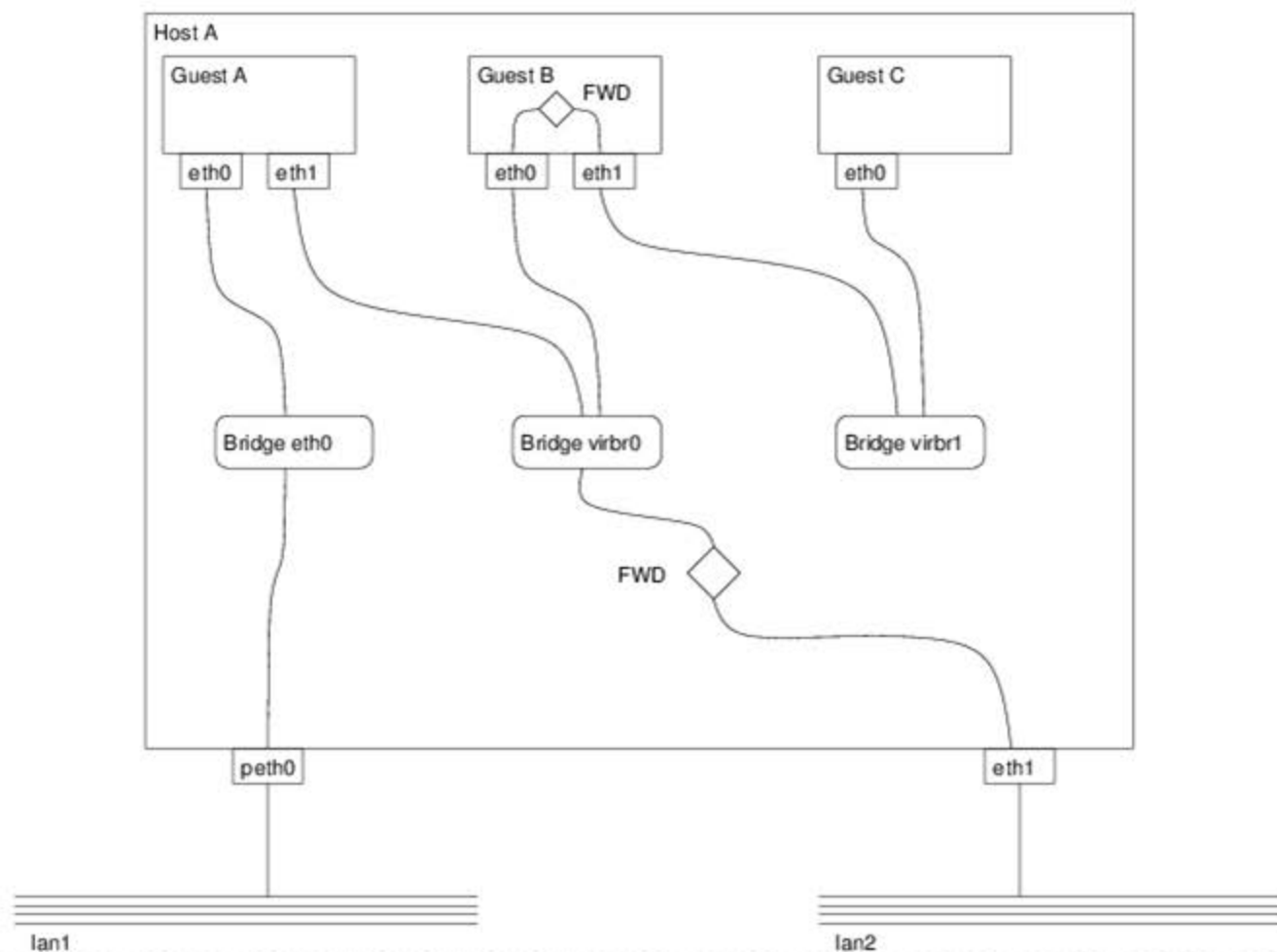
综合应用示例：逻辑视图

<http://libvirt.org/archnetwork.html>

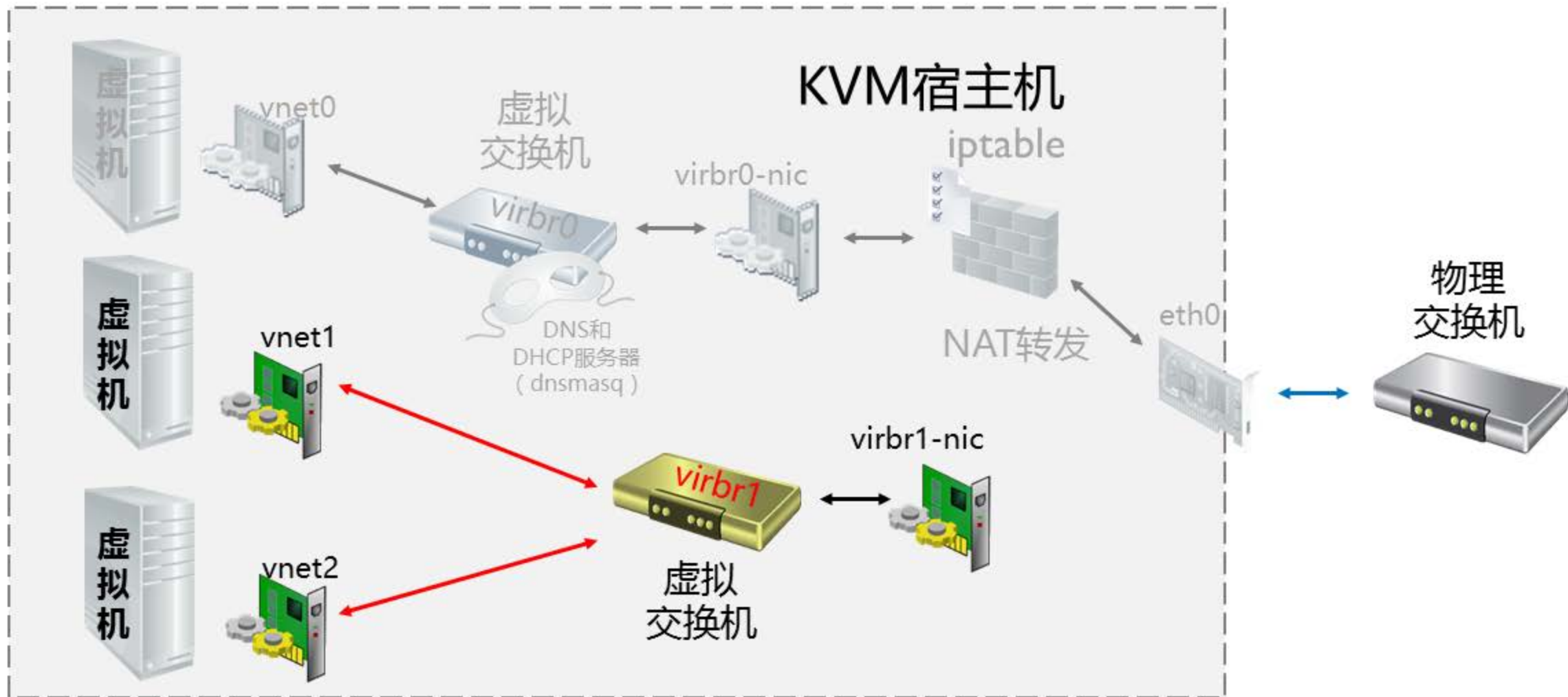


综合应用示例：物理视图

<http://libvirt.org/archnetwork.html>



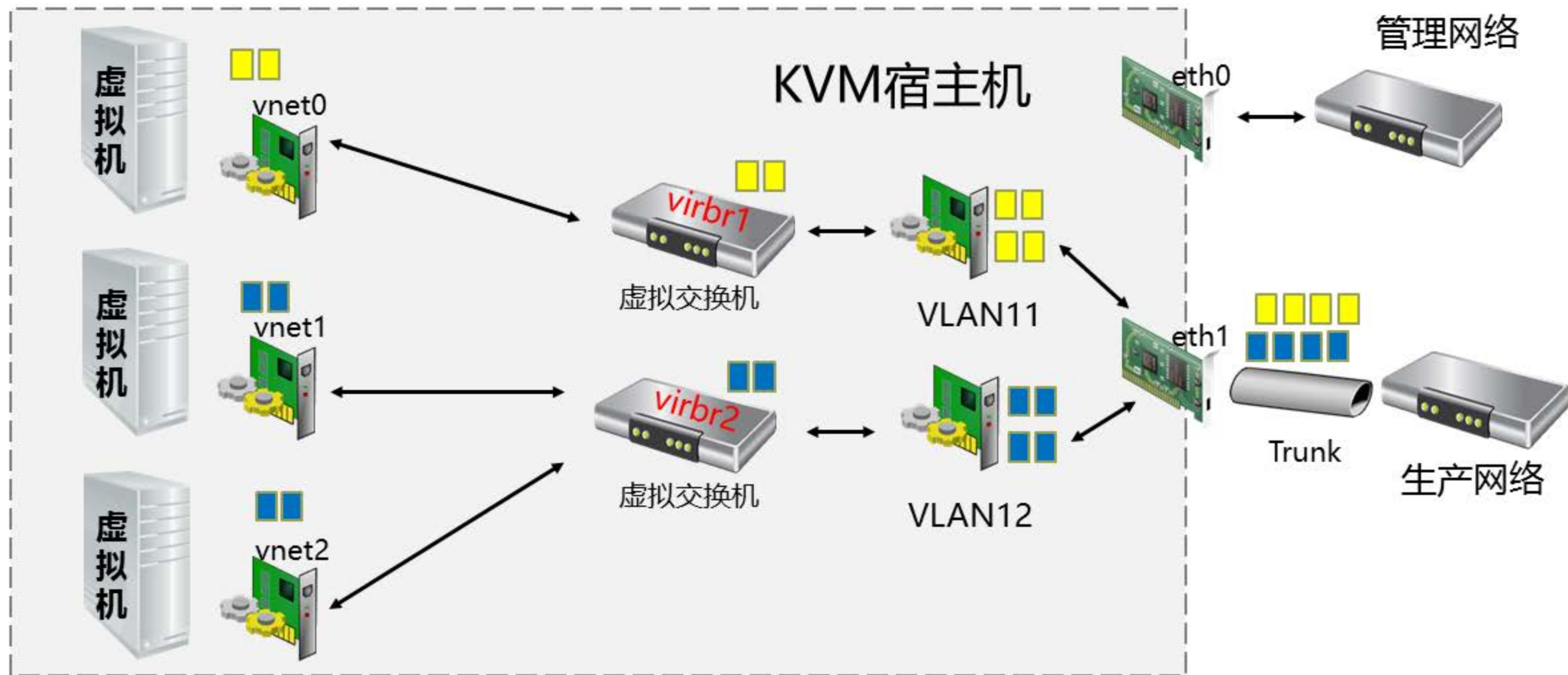
隔离网络



◆ 配置VLAN

- ▶ KVM下VLAN配置概述
- ▶ 在Linux中配置VLAN
- ▶ KVM使用VLAN

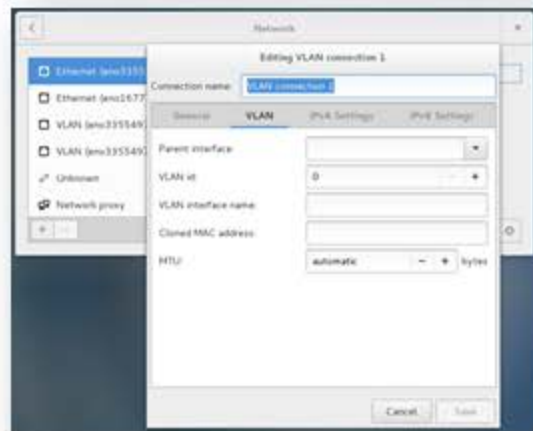
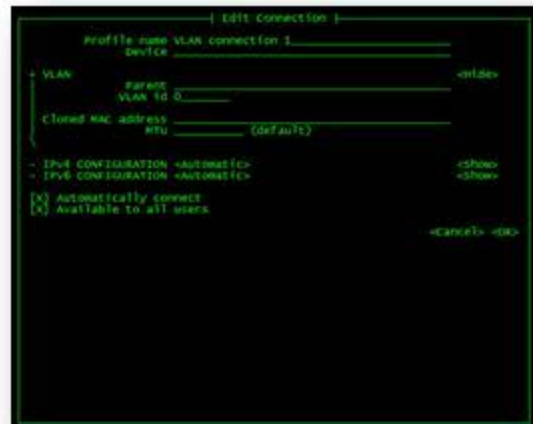
KVM下VLAN配置概述



在Linux中配置VLAN

- ▶ Redhat/CentOS 7 已经没有vconfig命令了
- ▶ Redhat/CentOS配置VLAN常用方法：
 - ▶ 命令行修改网络脚本文件（推荐）
 - ▶ Network Manager
 - ▶ nmtui：NetworkManager的文本用户接口
 - ▶ nmcli：NetworkManager的命令行工具
 - ▶ 图形界面工具
 - ▶ 命令行的ip命令

```
# ip link add link eth0 name eth0.8 type vlan id 8
# ip -d link show eth0.8
4: eth0.8@eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
noqueue state UP mode DEFAULT
    link/ether 52:54:00:ce:5f:6c brd ff:ff:ff:ff:ff:ff promiscuity 0
    vlan protocol 802.1Q id 8 <REORDER_HDR>
```



演示：创建VLAN

- ▶ 方法1：通过nmcli来创建VLAN
- ▶ 方法2：通过命令行修改网络脚本文件
- ▶ 方法3：通过virt-manager来创建

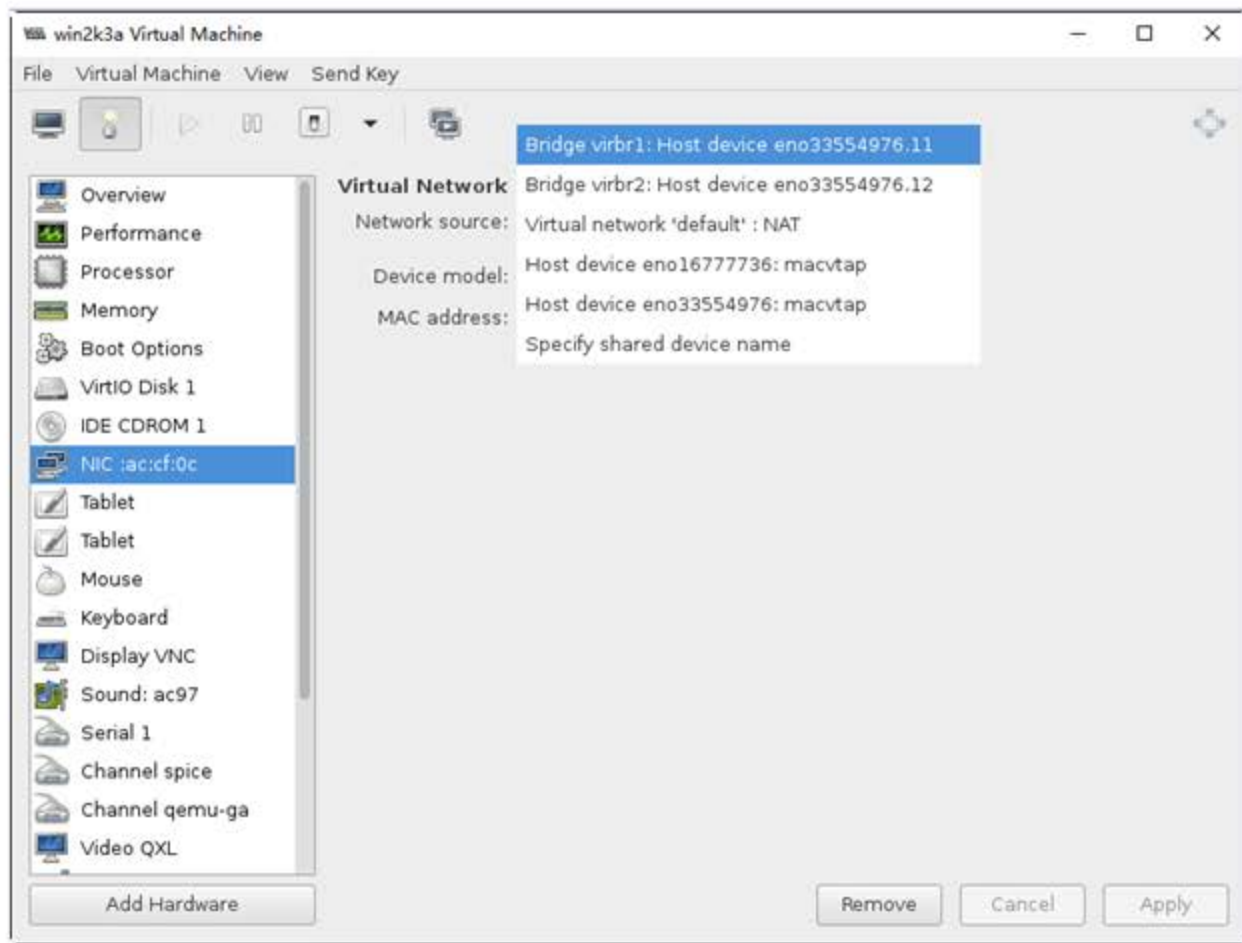
```
# nmcli connection show
```

NAME	UUID	TYPE	DEVICE
Vlan eno33554976.11	04186f9a-64da-e1fb-48ff-e2f3b18264db	vlan	eno33554976.11
Vlan eno33554976.12	7369ff99-43dd-fba1-6ff8-d63c153547d1	vlan	eno33554976.12
virbr0-nic	93318350-4af3-425c-9460-a5533f13d8a4	generic	virbr0-nic
virbr0	9eb91425-e5c2-4438-8af1-09ae9e81a3af	bridge	virbr0
eno33554976	8de27733-0c72-2949-b490-8a0d853a42d4	802-3-ethernet	eno33554976
eno16777736	51ca47f2-49c8-47af-a26c-f82d4ea6a120	802-3-ethernet	eno16777736

演示：KVM中使用VLAN

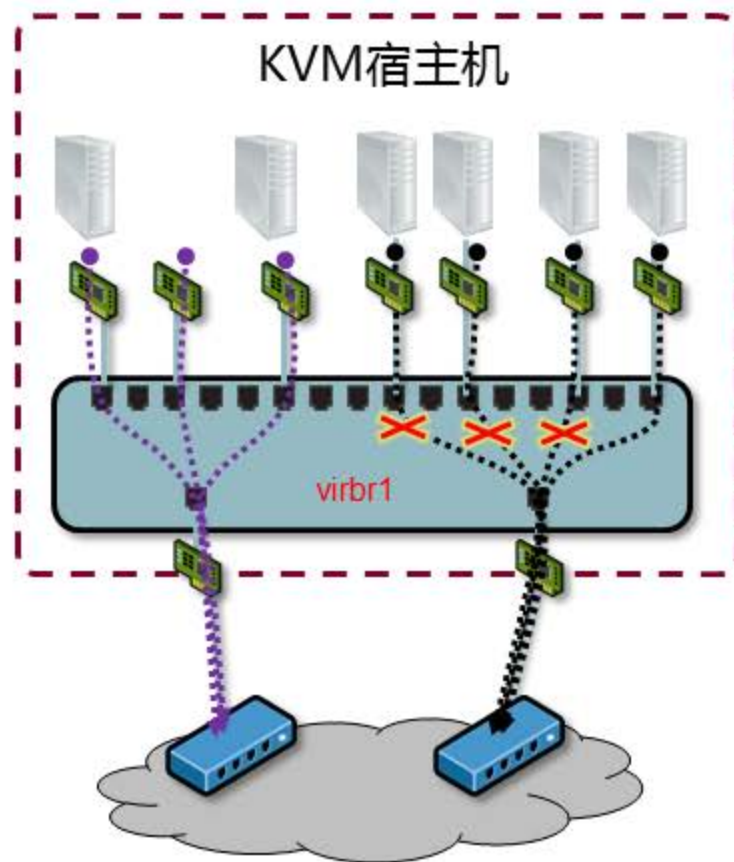
```
# vi ifcfg-virbr1
DEVICE=virbr1
TYPE=Bridge
BOOTPROTO=static
IPADDR=172.16.11.11
NETMASK=255.255.255.0
ONBOOT=yes
DELAY=0

# vi ifcfg-eno33554976.11
DEVICE=eno33554976.11
VLAN=yes
ONBOOT=yes
BOOTPROTO=none
BRIDGE=virbr1
```



◆ 网络过滤 Network filtering

- ▶ 什么是网络过滤
- ▶ 内置的默认规则
- ▶ 应用案例分析



什么是网络过滤 Network filtering

- ▶ 官方链接 <http://libvirt.org/formatnwfilter.html>
- ▶ 通过ebtable引擎来实现
- ▶ 是一种网络流量的过滤规则，管理对虚拟机网络流量的接受和转发
- ▶ 由于虚拟机不能控制过滤规则，所以对虚拟机的访问控制具有强制性
- ▶ 通过XML文件存储配置，libvirt动态调整ebtable配置
- ▶ 即可以针对特定虚拟机进行配置，也可以多个虚拟机共享配置

```
<interface type='bridge'>
  <mac address='52:54:00:24:4c:ee' />
  <source bridge='br0' />
  <model type='virtio' />
  <filterref filter='limit' />
</interface>
```



```
<interface type='bridge'>
  <mac address='12:34:56:78:90:12' />
  <source bridge='br1' />
  <model type='rtl8139' />
  <filterref filter='limit' />
</interface>
```



```
# virsh nwfilter-dumpxml limit
<filter name='limit' chain='root'>
  <uuid>1c3384c7-093a-5689-2cf3-320ef716ba2e</uuid>
  <rule action='accept' direction='inout' priority='400'>
    <icmp connlimit-above='2' />
  </rule>
  <rule action='accept' direction='inout' priority='500'>
    <tcp />
  </rule>
  <filterref filter='clean-traffic' />
  <rule action='drop' direction='inout' priority='1000'>
    <all />
  </rule>
</filter>
```



virsh中与网络过滤器有关的命令

```
virsh # help filter
```

```
Network Filter (help keyword 'filter'):
```

```
nwfilter-define    define or update a network filter from an XML file  
nwfilter-dumpxml   network filter information in XML  
nwfilter-edit      edit XML configuration for a network filter  
nwfilter-list      list network filters  
nwfilter-undefine  undefine a network filter
```


过滤规则示例

```
# virsh nwfilter-dumpxml limit
<filter name='limit' chain='root'>
  <uuid>1c3384c7-093a-5689-2cf3-320ef716ba2e</uuid>
  <rule action='accept' direction='inout' priority='400'>
    <icmp connlimit-above='2' />
  </rule>
  <rule action='accept' direction='inout' priority='500'>
    <tcp />
  </rule>
  <filterref filter='clean-traffic' />
  <rule action='drop' direction='inout' priority='1000'>
    <all />
  </rule>
</filter>
```

Chain类型：

- 所有过滤规则都被组织到一个过滤链中
- 数据包经过这些过滤链，被选择进入虚拟机或是被DROP。
- 链都有不同的优先级，root链的优先级最高
- 所有的数据包必须先要经过root链后，才可能继续到其他过滤规则中匹配。
- 目前已经存在的链：root、mac、stp、vlan、arp、rarp、ipv4、ipv6
- priority优先级的设定：所有的链都被连接到root链中。优先级的值越小，优先级别越高。用户可以定义自己的优先级数值，取值范围在[-1000,1000]。

演示：考察默认规则

```
virsh # nwfilter-list
```

UUID	Name
17411a8d-a462-426c-a93a-3e6d93d195c8	allow-arp
13f1a780-c05b-49ba-9937-c671e6fb6bdf	allow-dhcp
1d291a19-0dc0-4b3a-af7a-b6cc19cb6d85	allow-dhcp-server
6fb77a9d-5a7e-4de6-be45-123b5023ab20	allow-incoming-ipv4
8bc2ad0f-df9a-4134-9f9f-f343fc9ff5f7	allow-ipv4
db78d3a7-c303-4834-b194-d0f0ce23a0d9	clean-traffic
703cf08e-157b-4bd0-9320-79e3c53f7961	no-arp-ip-spoofing
6ff7b9ba-b188-423a-b890-f282cc9f090e	no-arp-mac-spoofing
556589d0-f316-42a7-8147-98daa7120c1e	no-arp-spoofing
222b3bb3-2a3f-42b3-9835-979bceedab6f	no-ip-multicast
9cef3d10-2236-46e2-990b-20e8387c18d9	no-ip-spoofing
63a231cd-8e52-4ac0-9836-acb909e70614	no-mac-broadcast
ef3a7804-8249-4290-9118-42473c5d69fb	no-mac-spoofing
f59a14d8-b3d4-40bf-a0e6-213571192cda	no-other-l2-traffic
1efb813a-5916-46fb-a01f-e84d44e0d546	no-other-rarp-traffic
8e7b6bc8-9968-48ea-af02-ed84929eb6ba	qemu-announce-self
d903fee5-0af4-4238-ade5-960d76d58e33	qemu-announce-self-rarp

```
virsh # nwfilter-dumpxml clean-traffic
<filter name='clean-traffic' chain='root'>
  <uuid>db78d3a7-c303-4834-b194-d0f0ce23a0d9</uuid>
  <filterref filter='no-mac-spoofing' />
  <filterref filter='no-ip-spoofing' />
  <rule action='accept' direction='out' priority='-650'>
    <mac protocolid='ipv4' />
  </rule>
  <filterref filter='allow-incoming-ipv4' />
  <filterref filter='no-arp-spoofing' />
  <rule action='accept' direction='inout' priority='-500'>
    <mac protocolid='arp' />
  </rule>
  <filterref filter='no-other-l2-traffic' />
  <filterref filter='qemu-announce-self' />
</filter>
```

禁止VM外发MAC欺骗、ARP欺骗以及IP地址欺骗.....

演示：考察KVM网络过滤与ebtables的关系

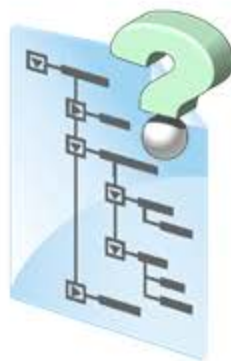
▶ 对比启动配置有网络过滤的虚拟机前后ebtables的变化

```
# ebtables -t nat -L
Bridge table: nat
Bridge chain: PREROUTING, entries: 0, policy: ACCEPT
Bridge chain: OUTPUT, entries: 0, policy: ACCEPT
Bridge chain: POSTROUTING, entries: 0, policy: ACCEPT
```

```
# virsh dumpxml win2k3a | more 只看与网络有关的
<interface type='network'>
  <mac address='52:54:00:ac:cf:0c' />
  <source network='default' bridge='virbr0' />
  <target dev='vnet0' />
  <model type='virtio' />
  <filterref filter='clean-traffic' />
  <alias name='net0' />
</interface>
```

思考：仅允许虚拟机发出“干净”的数据包

- ▶ 防止虚拟机进行MAC、IP和ARP欺骗
- ▶ 仅打开TCP 22和80端口
- ▶ 允许虚拟机向外ping，但不是允许其他机器ping此虚拟机
- ▶ 允许DNS解析流量



总结

- ▶ ~~Linux网桥基本概念~~
- ▶ ~~qemu-kvm支持的网络~~
- ▶ ~~向虚拟机添加虚拟网络连接~~
- ▶ ~~虚拟网络配置~~
 - ▶ ~~基于NAT的虚拟网络~~
 - ▶ ~~基于网桥的虚拟网络~~
 - ▶ ~~用户自定义的隔离的虚拟网络~~
- ▶ 多物理网卡绑定
- ▶ 配置VLAN
- ▶ 通过网络过滤提高安全性