# KVM Security - Where Are We At, Where Are We Going

Klaus Heinrich Kiwi, IBM LTC

# Agenda

High-level architecture

Secure Management

Virtual Network

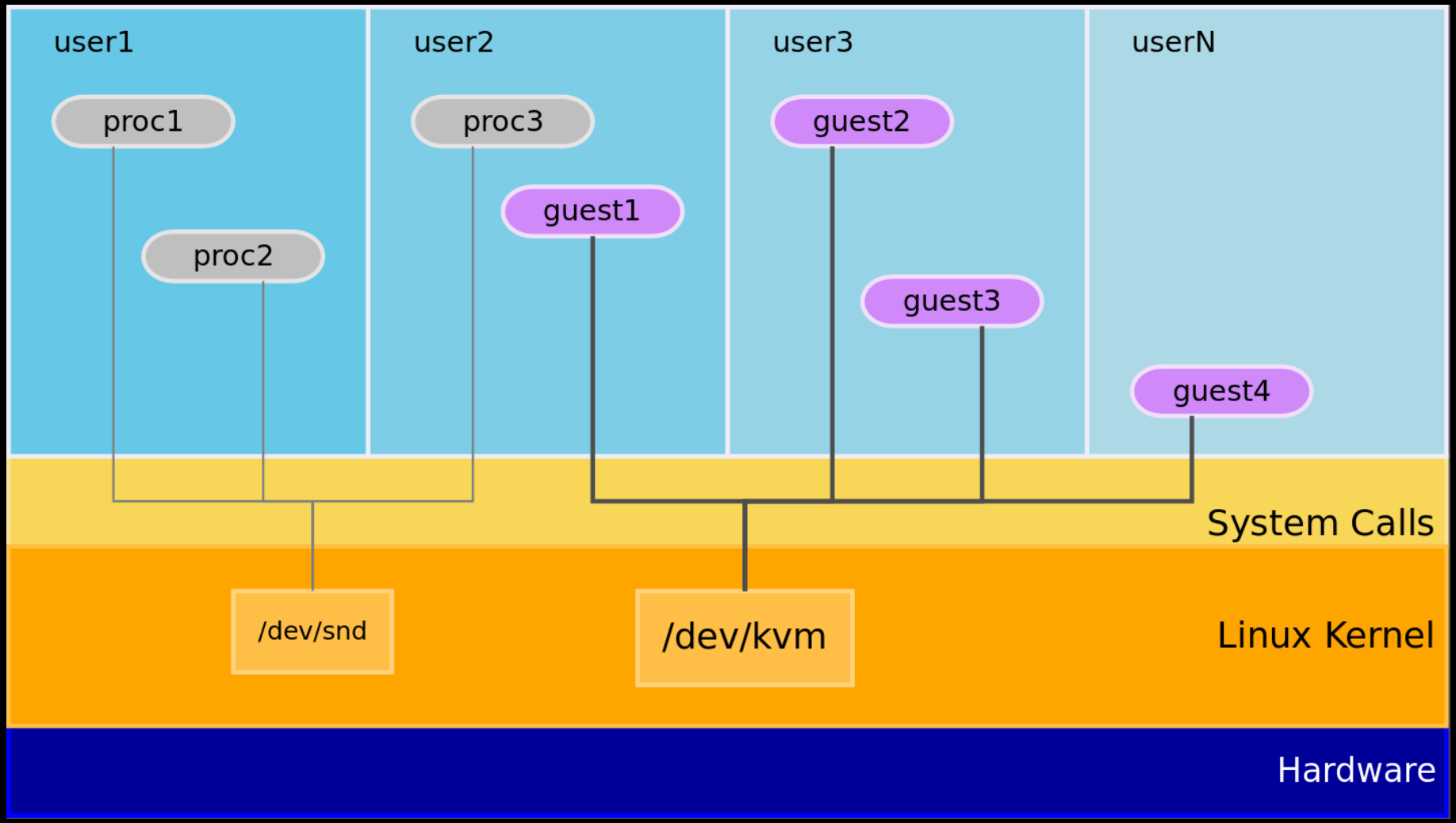
SELinux + sVirt

Auditing

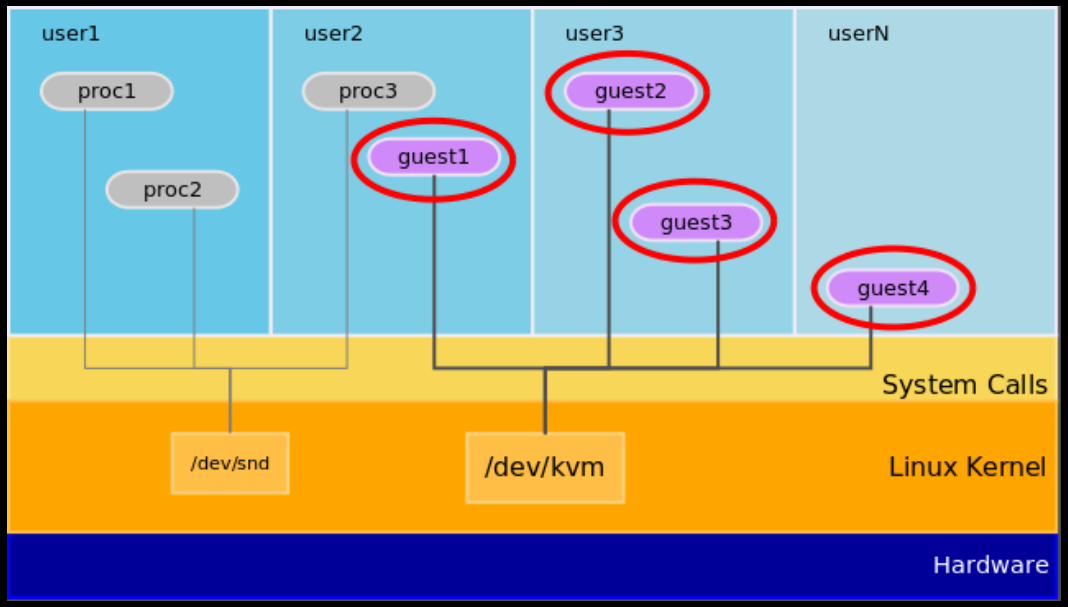
Guest-image cryptography

Future

# High-level architecture



# High-level architecture - "What makes KVM so cool?"




KVM Guest

=

Userland process

(like any other)

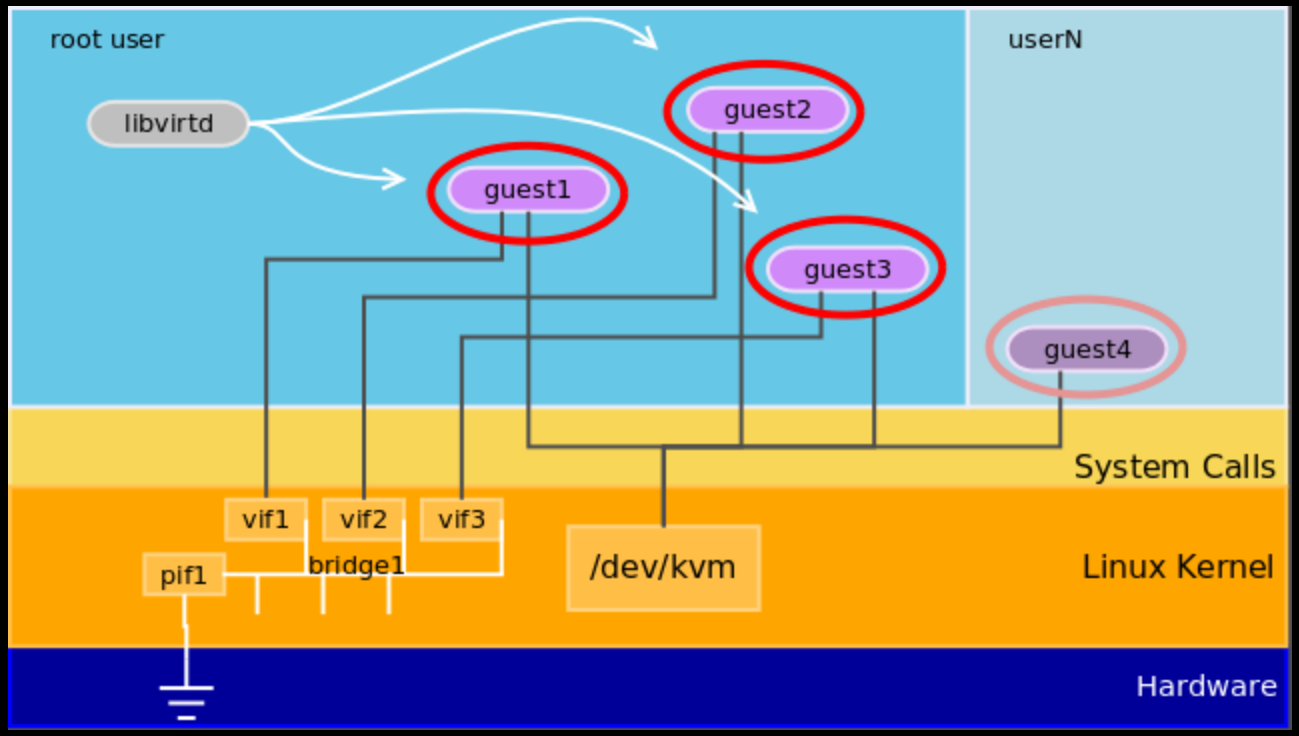
Process separation

Modular design:

- ✓Aligned with *Open-source* and *Linux* philosophy
- ✓Re-use of *know*, *tested* and *proofed* interfaces and subsystems

**Libvirt**: Simplification, better *out-of-box* security

# High-level architecture - Current real-world scenario




In practice = All guest running as "***root***" or "***qemu***"

- Privileged *network set-up* step (CAP_NET_ADMIN ...)

**Libvirt** rapidly evolving to cover more scenarios, better security...

# Agenda

High-level architecture

Secure Management

Virtual Network

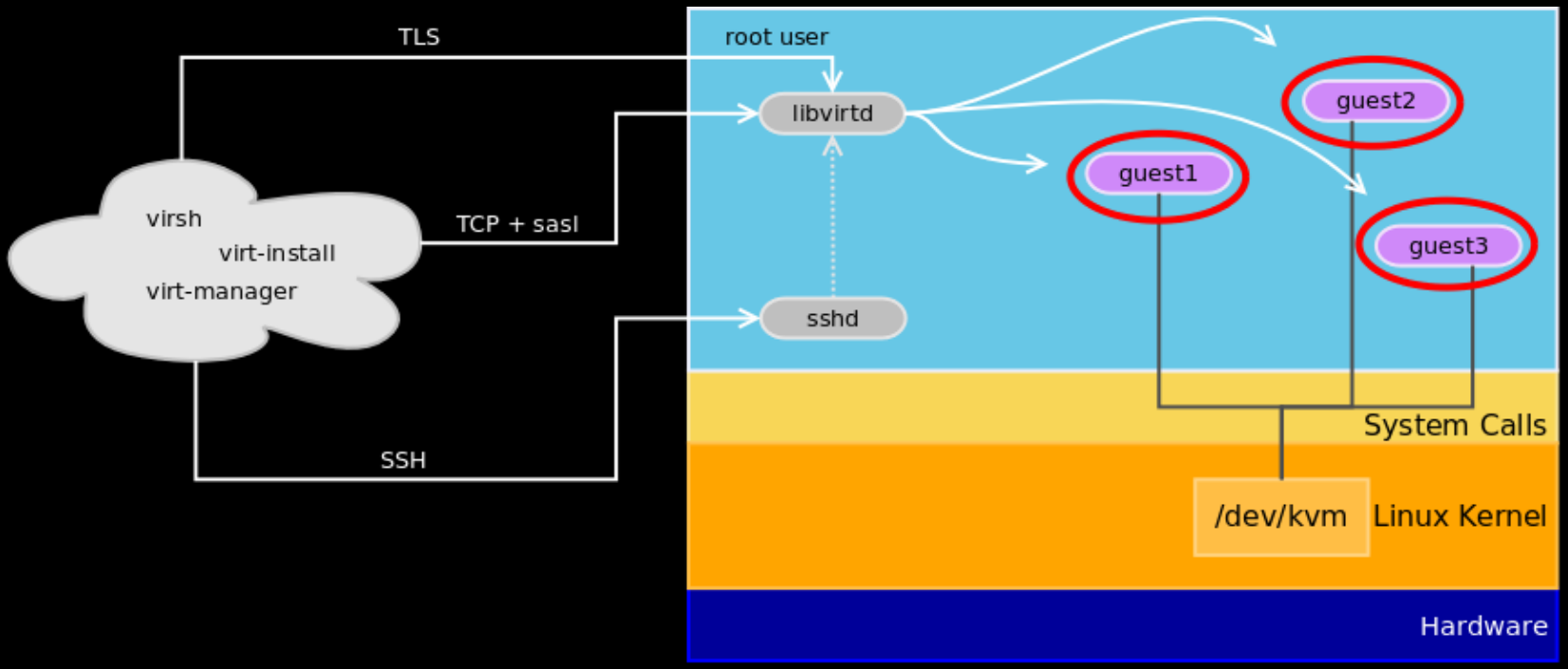
SELinux + sVirt

Auditing

Guest-image cryptography

Future

# Secure Management

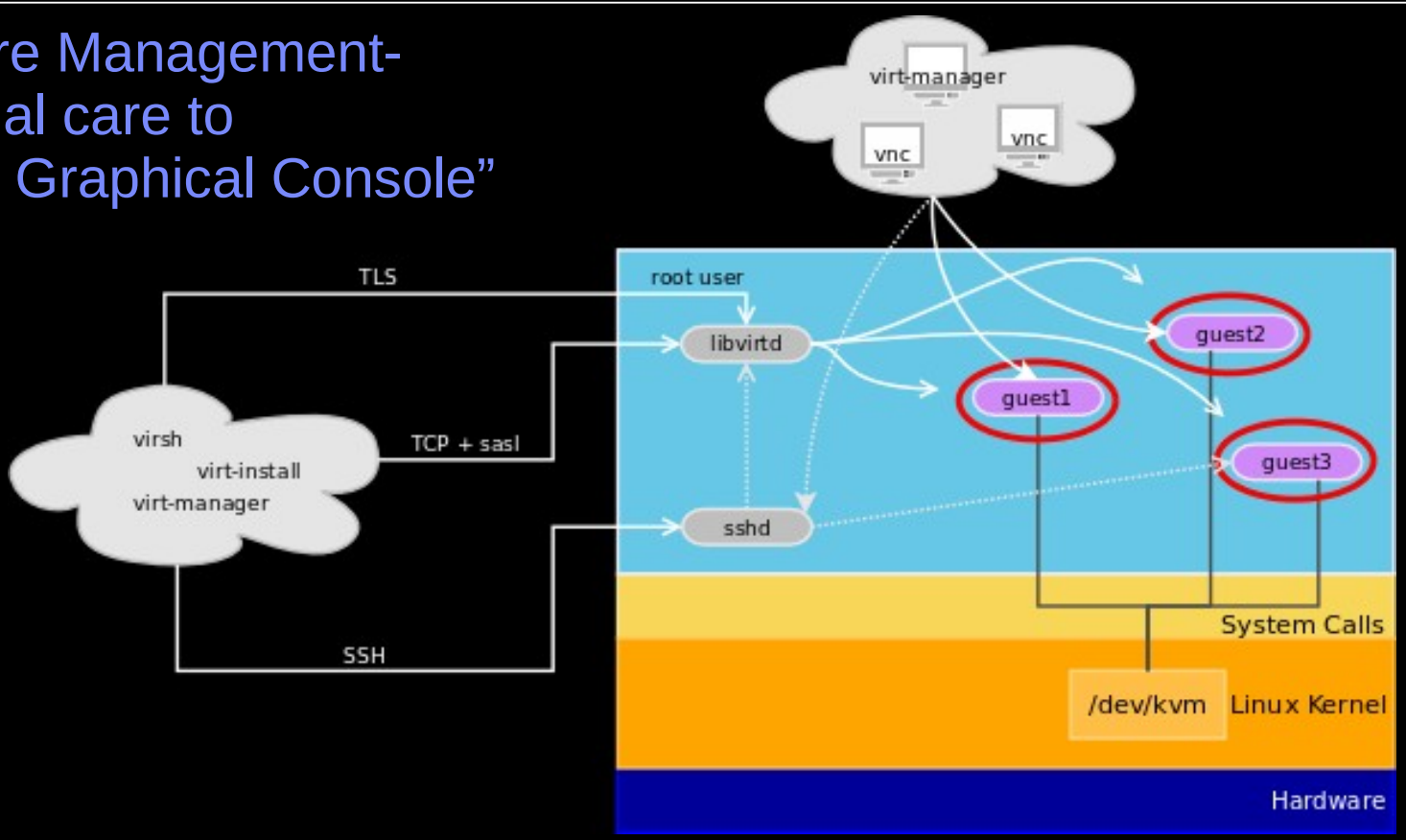



1)Local (or SSH) – Standard UNIX perms, PolicyKit

2)TCP + sasl – separate credential DB, encrypted channel

3)TLS – Standard PKI with certificates

# Secure Management- Special care to "VNC Graphical Console"




Default is still *clear-text* TCP session available to every local user

- *vncpasswd* considered weak

Alternative: VNC over TLS (**with** client cert verification) or SASL

# Secure Management

## Things to keep in mind

- Isolate the management network. Use firewalls.
- “VNC Graphic console”
  - Dynamic port range
  - Weak authentication per default 
  - Local users **can** try to connect
  - If graphical console can't be avoided:
    - VNC-over-TLS or SASL
- Avoid using Unix “*root*” to perform management
  - Prefer SASL credential database
  - Can combine with TLS certificates!

# Agenda

High-level architecture

Secure Management

Virtual Network

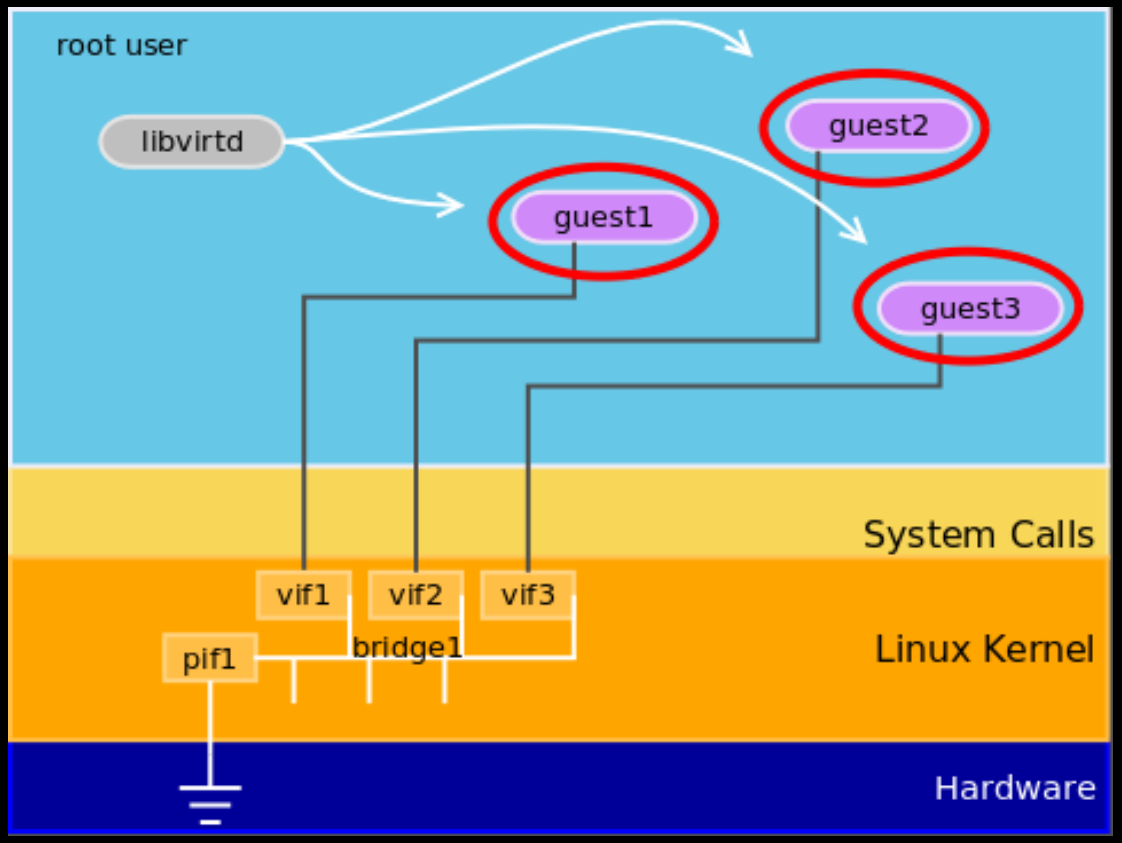
SELinux + sVirt

Auditing

Guest-image cryptography

Future

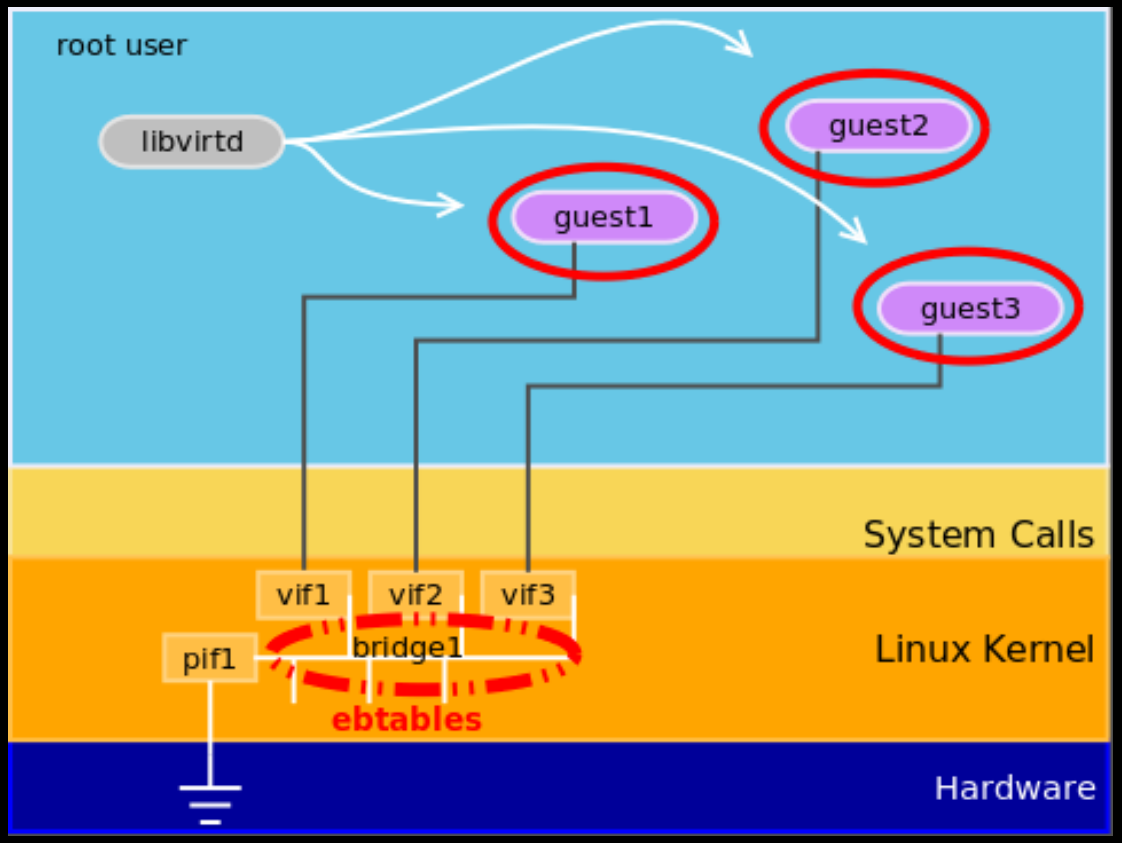
# 1) Simple Bridge




- Link-layer (OSI layer 2) – **ETHERNET FRAME** forwarding
- Not specific to Virtualization (remember *re-use, Linux philosophy*...)
- Well-known **Risks**:
  - ARP spoofing
  - MAC flooding

## Virtual Network

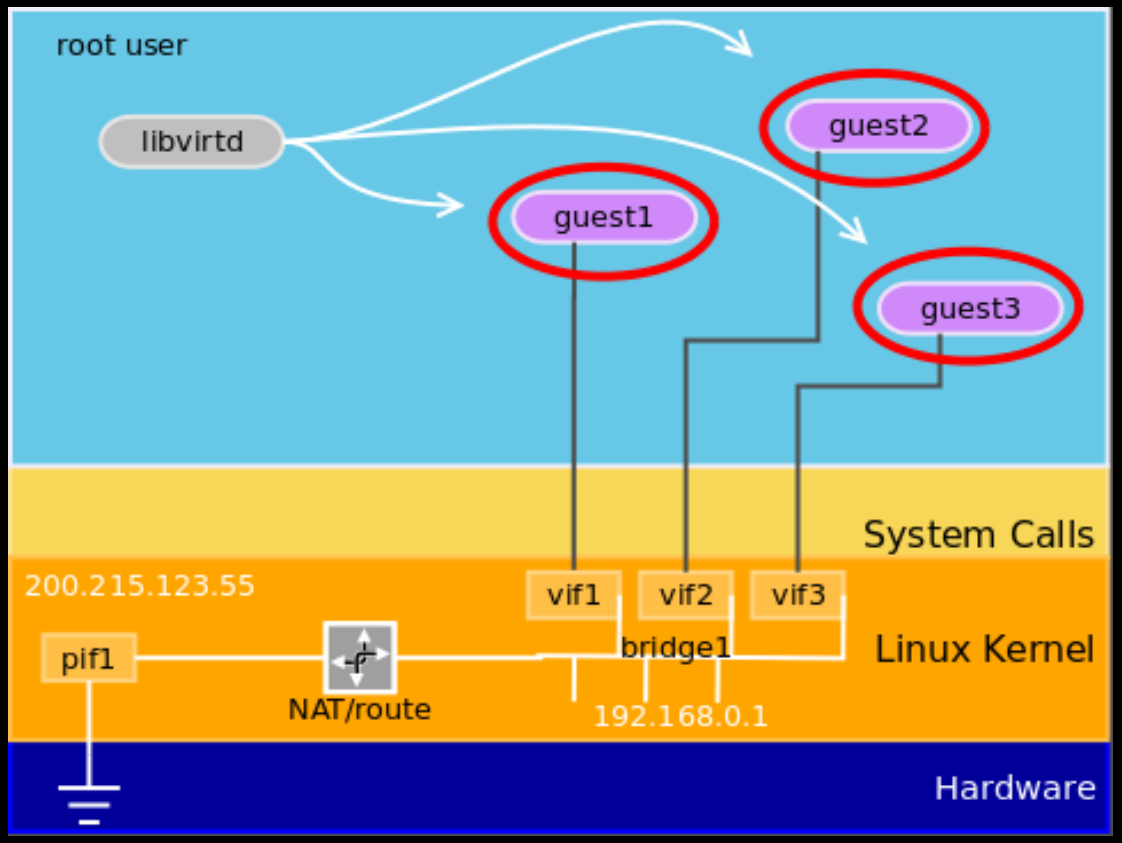
**2) Simple Bridge + ebtables**




```
# ebtables -P FORWARD DROP
# ebtables -s 54:52:00:11:22:33 -i vif1 -j ACCEPT
# ebtables -o vif+ -j ACCEPT
```

# Virtual Network

## 3) IP Routing
## Or
## NAT

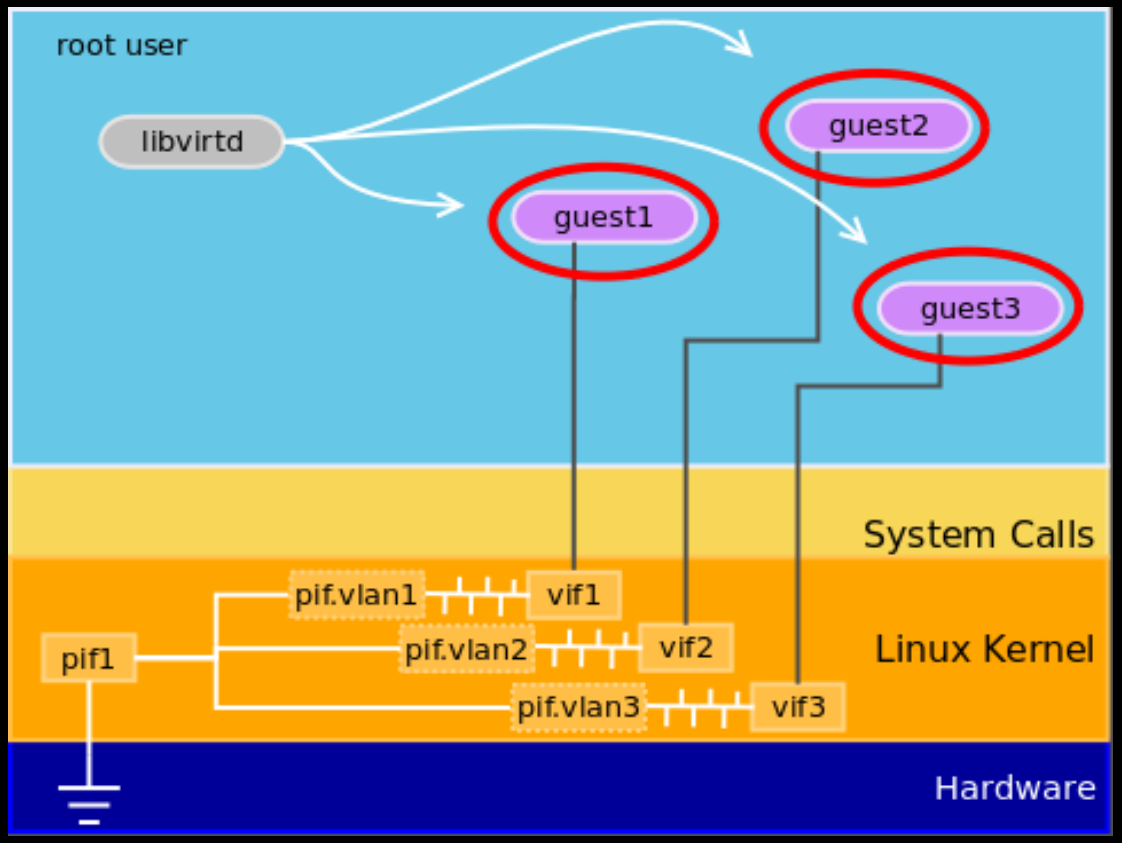



Network-layer (OSI layer 3) – **IP PACKET** forwarding

+ Requires IP addresses (added complexity?)
+ Requires L2 bridge anyway (risks are still valid!)

# Virtual Network

**VLAN**
**802.1q**



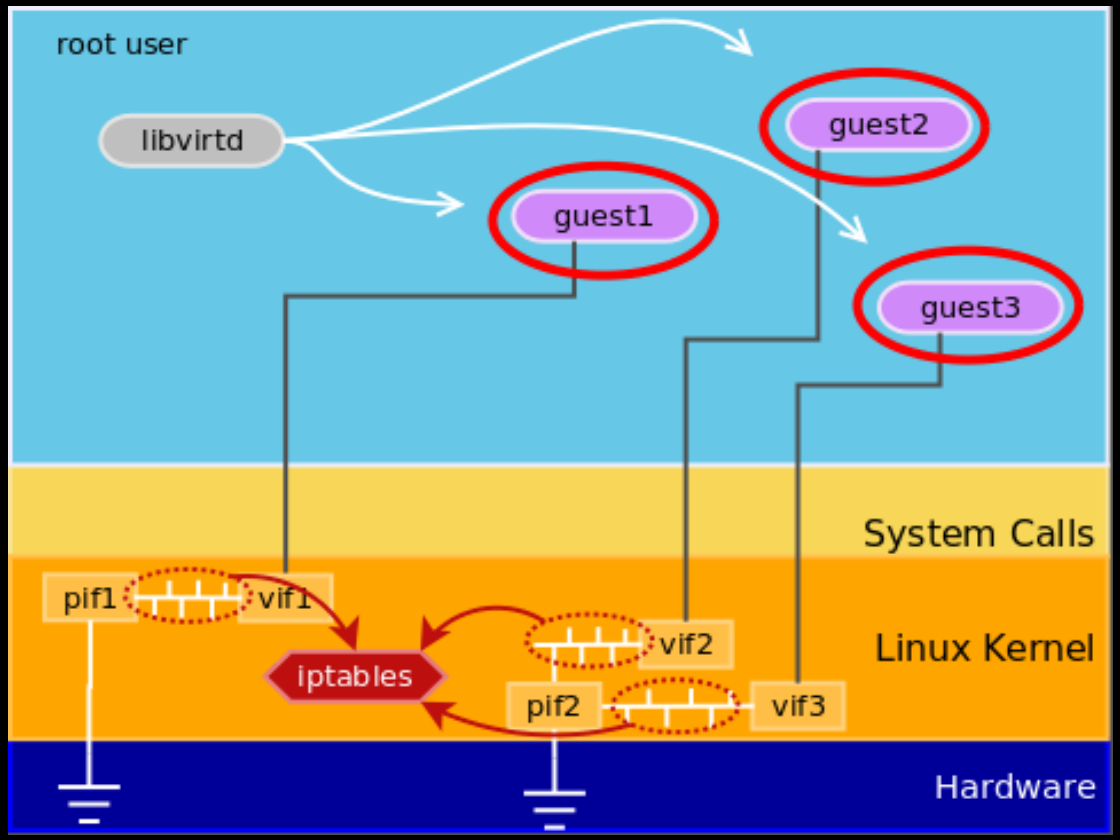

Bridge + sub-interfaces in specific VLAN ID

Configuration: *vconfig*

**Important**: keep 1 bridge per VLAN ID (avoid mixing VLANs)

## Virtual Network

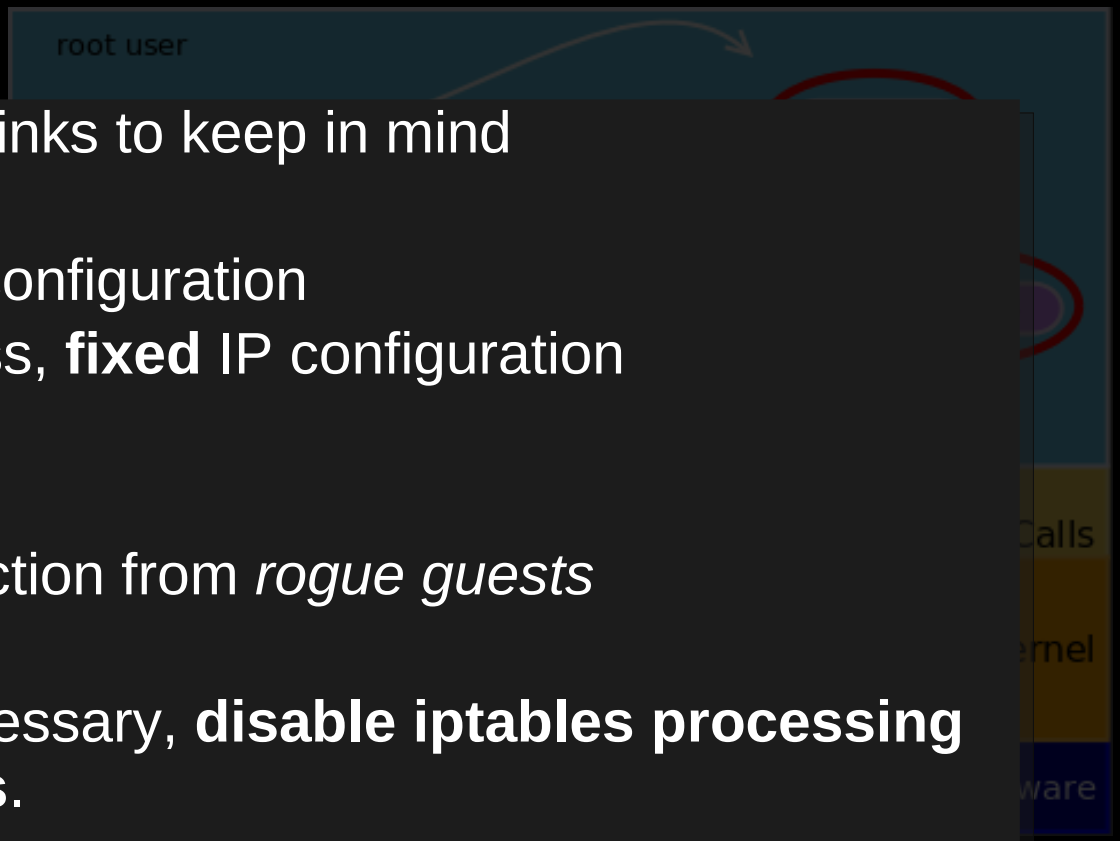
**Important Note:**
iptables processing
in
**Bridge** interfaces




*conntrack* module works on *global context*

/etc/sysctl.conf:

```
net.bridge.bridge-nf-call-ip6tables = 0
net.bridge.bridge-nf-call-iptables = 0
net.bridge.bridge-nf-call-arptables = 0
```




## Thinks to keep in mind

Avoid using dynamic configuration

- **Fixed** MAC address, **fixed** IP configuration

**ebtables**

- Flood, spoof protection from *rogue guests*

Unless absolutely necessary, **disable iptables processing on bridge interfaces**.

VLANs 802.1q = **simplification**!!

**Advanced filtering** sometimes requires libvirt bypassing

- *libvirt* > 0.8.2 has *advanced filtering* support

# Agenda

High-level architecture

Secure Management

Virtual Network

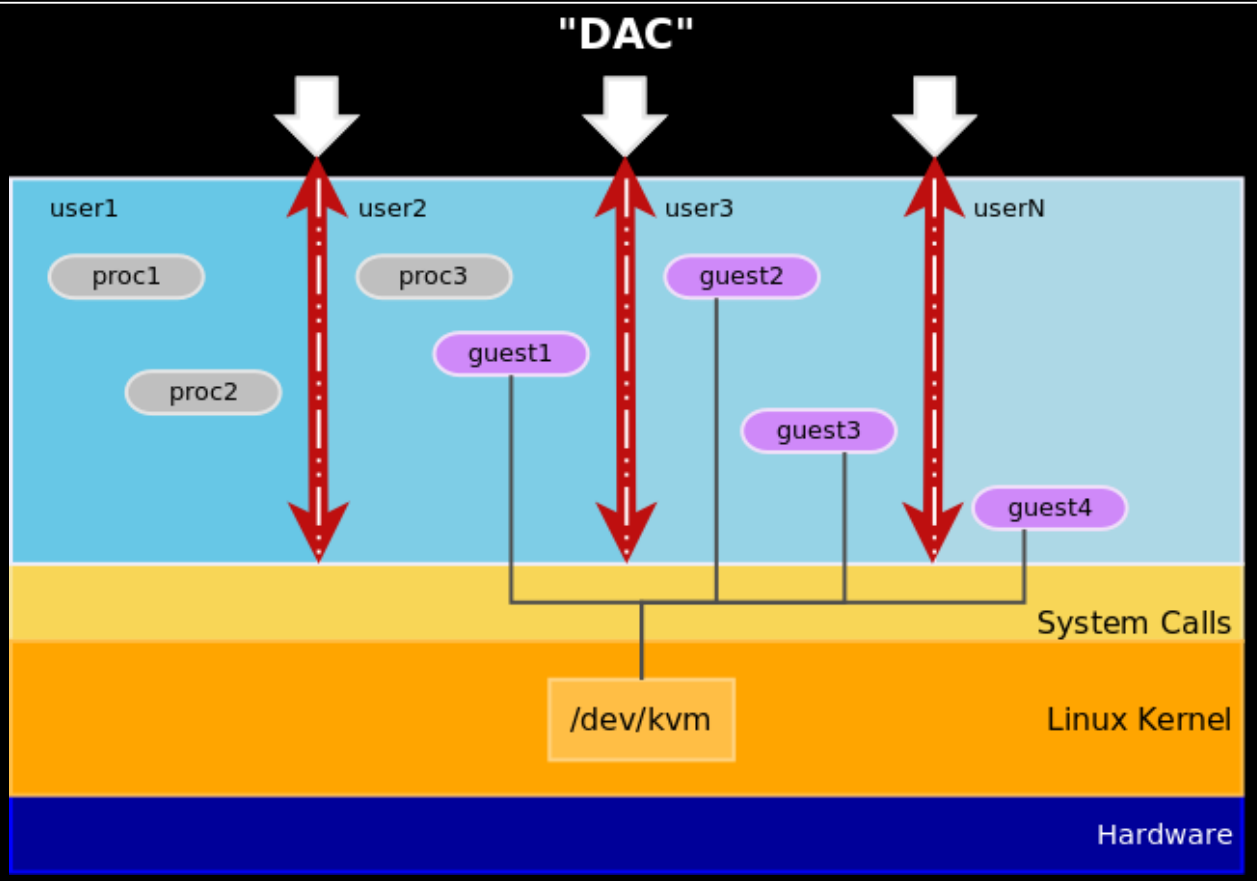
SELinux + sVirt

Auditing

Guest-image cryptography

Future

# Before SELinux



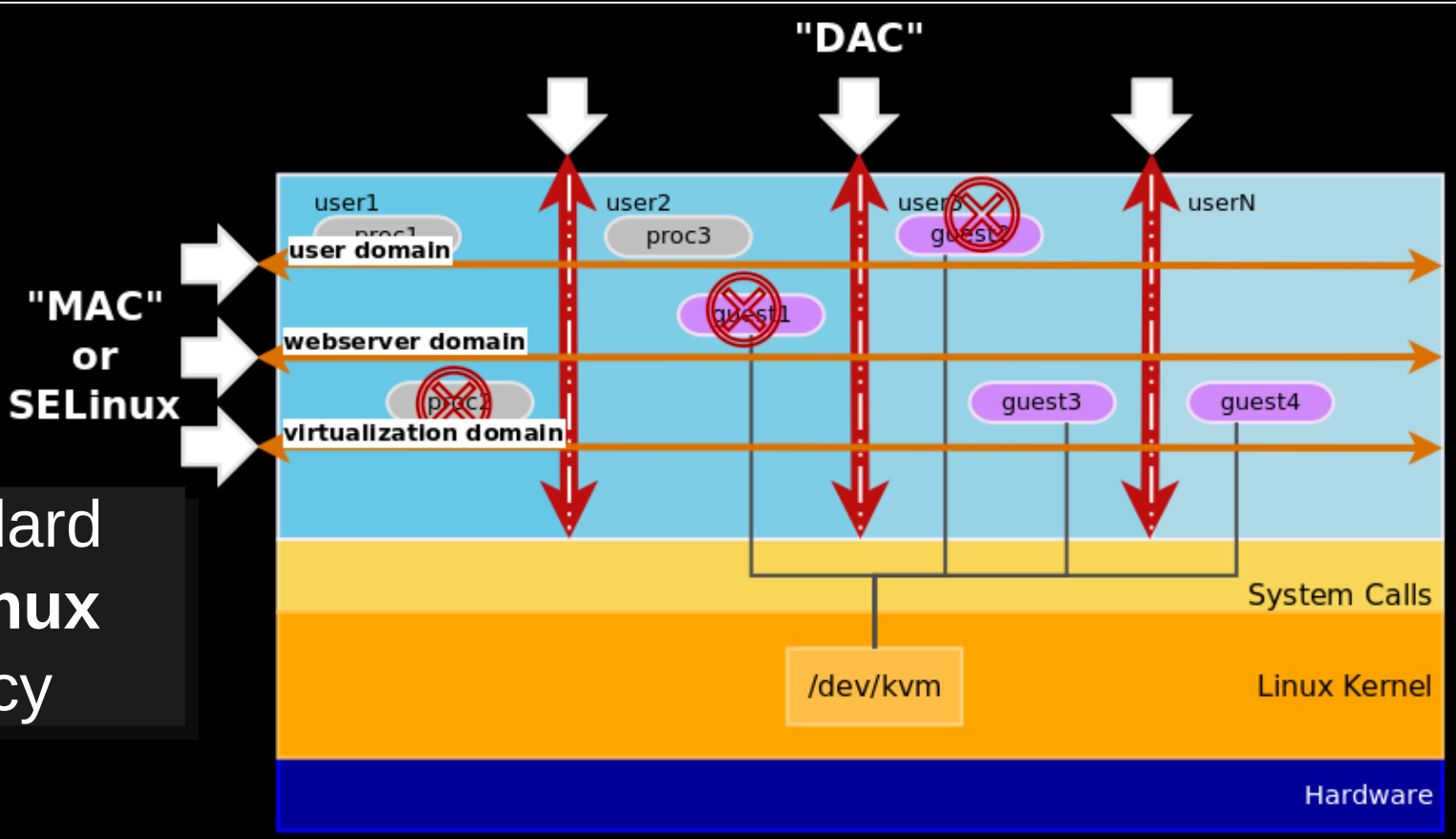

DAC = Discretionary Access Control

Standard UNIX perms (user/group/other, read/write/execute)

User **has control** over his/her objects/files

*Compromised* user **gives up** control to attacker

# SELinux



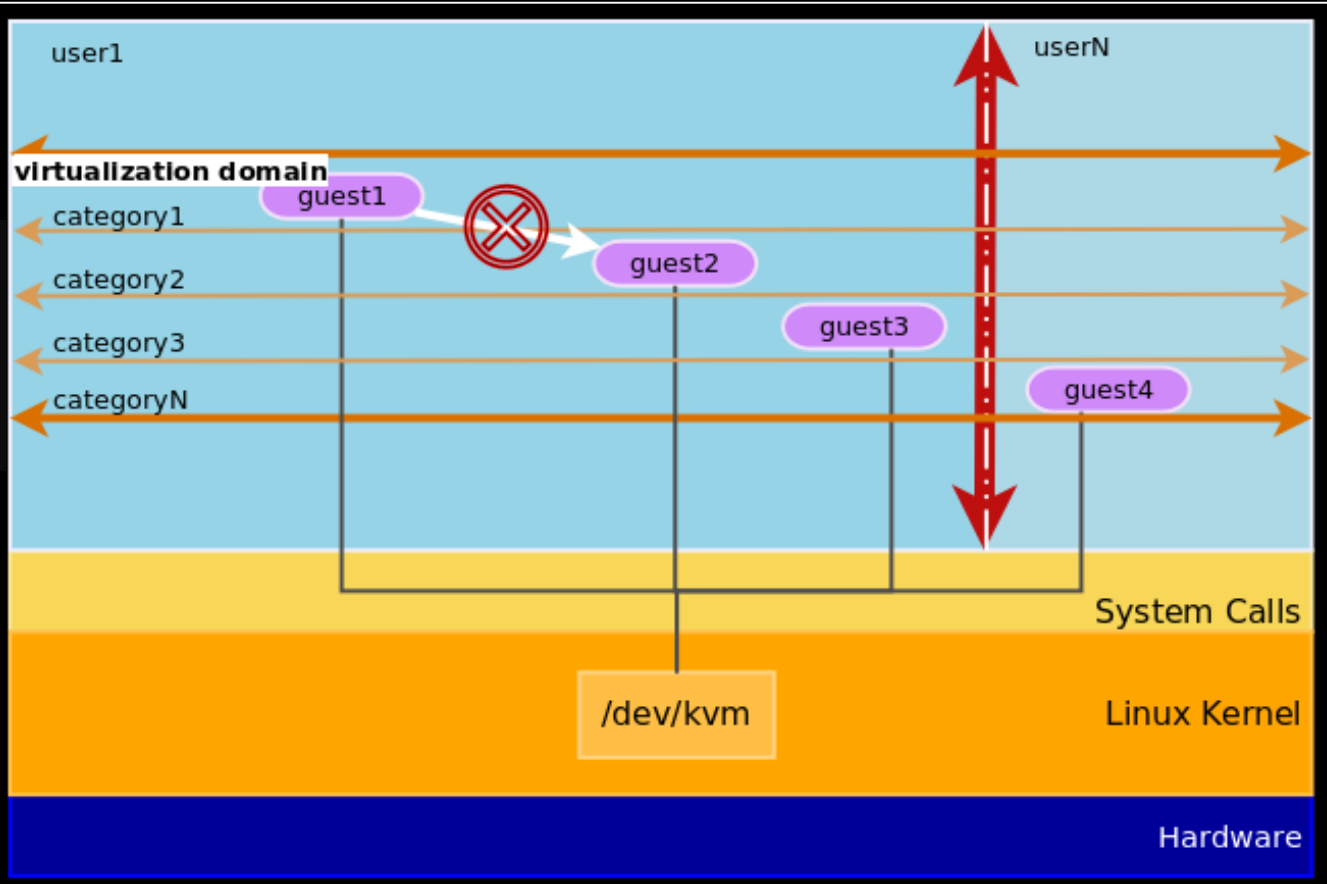

Standard **SELinux** policy

MAC = Mandatory Access Control

- System-wide policy
- Domain separation
  - Interactions must be explicitly allowed
- In practice: *Enforced* domain separation

# SELinux

**sVirt**-enabled virtualization




Further divide the system into *"categories"* (MCS)

- Guests running in same domain, but different *category*
- **Simple** and **Easy** to use!
  - By default, **No** interaction between categories is allowed
- **sVirt** = **SELinux** policy + **libvirt** driver

# Agenda

High-level architecture

Secure Management

Virtual Network

SELinux + sVirt

Auditing

Guest-image cryptography

Future

# Auditing

- **Motivation:**
  - Virtualization is a resonably contained activity
  - Privileged credentials (*root*) still needed for many mgmt tasks

- ***Linux Audit* :**

  Bootloader:

```
root (hd0,1)
kernel /vmlinuz-el5.x86_64 ro root=/dev/sda1 rhgb audit=1
initrd /initramfs-el5.x86_64.img
```

```
# chkconfig auditd on
```

`/etc/audit/audit.rules`:

  - Cut records for **any configuration change**
  - Cut records for **unusual interactions**

# Auditing

## audit.rules: **examples**

```
# Log non-standard access to the TLS private key
-a exit,always \
 -F path=/etc/pki/libvirt/private/serverkey.pem \
 -F subj_type!=virtd_t

# Non-standard changes to guest images
-a exit,always -F obj_type=virt_image_t \
 -F perm=wa -F subj_type!=qemu_t

# Qemu interactions with other domains
-a exit,always -F arch=b64 -S all -F perm=wax \
 -F subj_type=qemu_t -F obj_type!=qemu_t
```

More detailed examples in

"*Securing KVM guests and the host system*" Blueprint

http://publib.boulder.ibm.com/infocenter/lnxinfo/v3r0m0/topic/liaai/kvmsec/kvmsecstart.htm

or

http://blog.klauskiwi.com/

# Agenda

High-level architecture

Secure Management

Virtual Network

SELinux + sVirt

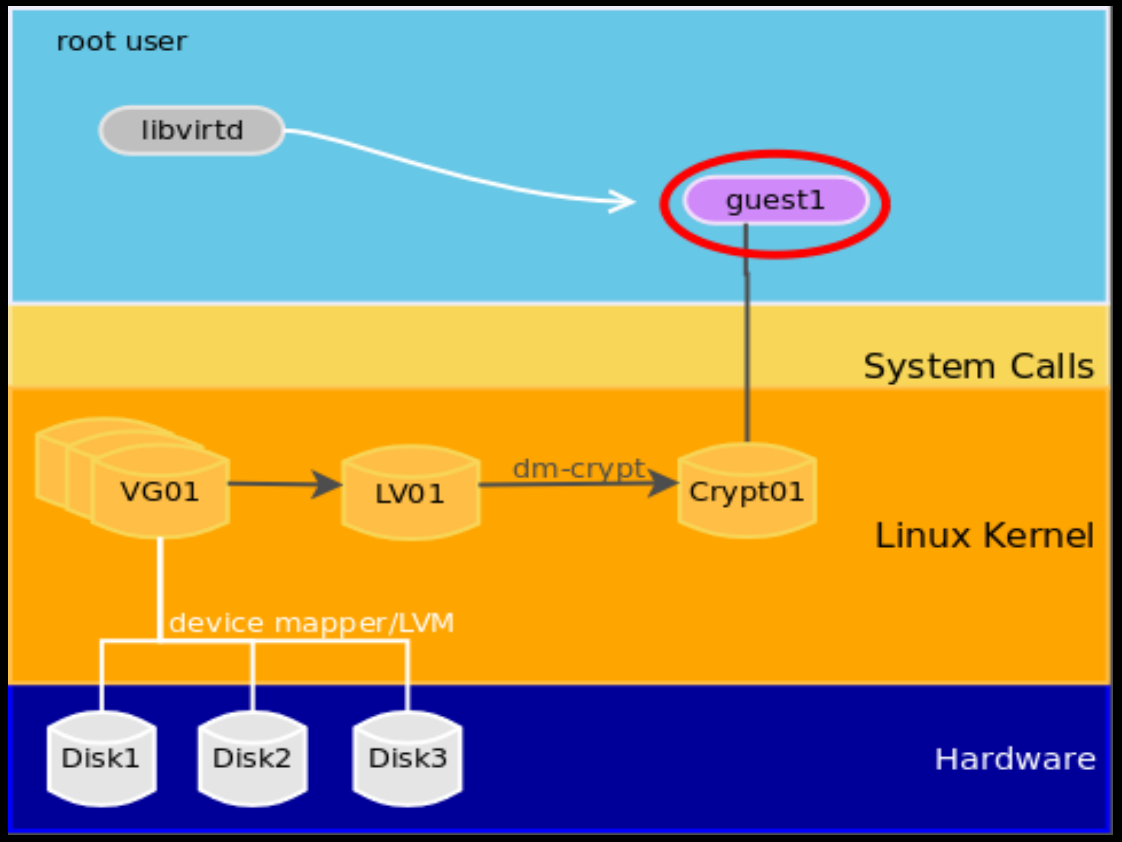
Auditing

Guest-image cryptography

Future

# Guest-image cryptography

**dm-crypt**
&
**qcow2**




- *"Data-at-rest"* tamper, unauthorized access protection
- dm-crypt = block-level transparent encryption (host or guest kernel)
  - Qemu/KVM couldn't care less (seen as *raw image* anyway)

```
# cryptsetup
```

- qcow2 = Qemu image format – adv. Features, crypto, libvirt sup.

# Agenda

High-level architecture

Secure Management

Virtual Network

SELinux + sVirt

Auditing

Guest-image cryptography

Future

# Looking forward

- **Qemu *"wrapper"*** for tap interfaces, CAP_NET_ADMIN handling
  - Run as (multiple) unprivileged users
- ***"Sandboxed"* Qemu**
  - Extend the concept of a Qemu-wrapper into a *"virt-cage"*
  - Drop access to every but minimally required set of syscalls
- *"Custom Policy editor"* for **sVirt**
  - Allow specific interactions between guests/categories
- KVM **Common Criteria** certification – RHEL 5 and RHEL6
- *Virtual Trusted Platform Module* (***vTPM***)
- More **libvirt** additions – network filtering, image crypto, mgmt

# Questions?

Klaus Heinrich Kiwi
klaus@klauskiwi.com

Slides are **already** avaiable at:
http://blog.klauskiwi.com