High-Speed, Multi-Tenant Bro System using SR-IOV and Containers

(...Proof -of-Concept...)
Who R U?

- Hailing from Columbia, MD
- Been in IT Security for ~12 years (see grey above head for more details)
- Support Federal customers in security engineering and evaluation, defensive cyber exercises, and other random work that pays for toys + high electric bills
- Big fan of Bro-IDS and the community
- Have a love of squeezing max performance out of things (more of an tester/improver than a builder)
Agenda

1. Why a multi-tenant Bro system?
2. Quick refresher on SR-IOV & Containers/VMs
3. Overview of proposed solution
4. Hardware/Software used for demo
5. Demonstration of multi-tenancy
6. Taking it to the next level NFV/SDN concepts
What problems are we solving?

- **Security Consolidation**
  - Being implemented in service-providers and data-centers (IDSaaS)
  - Numerous tenants on one hardware platform
  - Already seeing a lot of movement in Govt

- **Network Function Virtualization (NFV)**
  - Coming to a security stack near you!
  - Bandwidth isn’t getting any slower
Where NFV and Consolidation meet

Network Consolidation

$Lower Budgets

Move to Cloud

Sealing Technologies, Inc.
www.sealingtech.org
Where NFV and Consolidation meet

Network Consolidation

Virtual Multi-Tenant System

Move to Cloud

Lower Budgets
What is “Multi-Tenancy” to our customers?

1. Utilize **VLANs** as the segregator
   - VLAN 1000 = Tenant 1
   - VLAN 1002 = Tenant 2
   - ...

2. **Tenant** controlled policies/rules

3. If one tenant puts in a bad rule, shouldn’t affect anyone else.

4. Logs should be sent to tenant-specified location

5. Applicable to Boundary and Datacenters
Autonomous Multi-Tenant Perimeter (ISP or Shared Datacenter)
Higher Headquarters Example
Enough overview,

Let’s talk tech…
Bare vs VM vs Containers

**Traditional Bare Metal**
- No inherent app separation
- Shared Kernel and Libraries
- Maximum performance of security applications

**Virtual Machines**
- Two sets of hardware (real and virt)
- Two kernels (bad)
- Max security and autonomy

**Containers**
- One set of hardware
- One kernel
- Separation of libraries
- Separation of kernel resources

Sealing Technologies, Inc.
www.sealingtech.org
# VMs vs Containers

<table>
<thead>
<tr>
<th><strong>Virtual Machines</strong></th>
<th><strong>Containers</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Benefits</strong></td>
<td><strong>Benefits</strong></td>
</tr>
<tr>
<td>- Hardware abstraction provides better isolation/security</td>
<td>- Only one kernel we need to bypass/accelerate</td>
</tr>
<tr>
<td>- Tenants have control over Kernel in VM</td>
<td>- Startup/Shutdown measured in milliseconds</td>
</tr>
<tr>
<td>- Proven resource quota system</td>
<td>- Templating built-in to design</td>
</tr>
<tr>
<td><strong>Cons</strong></td>
<td><strong>Cons</strong></td>
</tr>
<tr>
<td>- 2x Kernel network stack</td>
<td>- Any change to kernel takes down all applications</td>
</tr>
<tr>
<td>- Hardware abstraction is slower</td>
<td>- Bro needs privileges for NIC access</td>
</tr>
<tr>
<td>- Slow start-up times</td>
<td></td>
</tr>
</tbody>
</table>

---

![Docker Logo]  

*chosen for PoC...*
Brief overview of SR-IOV

- Divides a NIC into numerous “Virtual Functions (VF)” that appear as separate NICS
- Separate Hardware RX/TX queues
- Can be used with host application, VMs, or containers
- Cards support up to 32 or 64 VFs depending on the chipset
- Hardware supported VLAN forwarding to each VF

**Source:** http://dpdk.org/doc/guides/nics/intel_vf.html
Kernel Drivers with SR-IOV

1. Most Intel drivers have a “VF” Equivalent
   • igb = igbvf
   • ixgbe = ixgbevf
   • i40e = i40evf

2. This is what the Kernel or VM will load.

3. Causes problems with kernel bypass

<table>
<thead>
<tr>
<th>Bypass Tech</th>
<th>Supported in Bro</th>
<th>Supports VF drivers</th>
</tr>
</thead>
<tbody>
<tr>
<td>PF-RING</td>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>DPDK</td>
<td>NO</td>
<td>YES</td>
</tr>
<tr>
<td>AF-PACKET</td>
<td>YES</td>
<td>YES**</td>
</tr>
<tr>
<td>Netmap</td>
<td>YES</td>
<td>ixgbevf</td>
</tr>
</tbody>
</table>
Hardware used for PoC

Supermicro
SYS-5018D-FN4T ($1400)

Intel Xeon D-1541
- 8-core 2.10 GHz w/ HT
- 16 Logical Procs
- Turbo to 2.70 GHz
- CPU only uses 45 Watts

Memory/HDD
- 32 GB PC4-2133 ($70)
- (2) Samsung 850 Pro SSD 256GB ($250)

Network
(2) 10 Gbps x552/x557 ports (built-in)
(4) 10 Gbps Intel XL710-DA4 (x8 PCIe Gen 3) ($498)

Total Cost = ~$2300

Need more power? Checkout the D-1567 (12/24 and MORE L3 cache!)
Enable SR-IOV & isolcpus

Kernel Settings:
- Isolated CPU cores 1-7,9-15
- Enable IOMMU in “passthrough”

```bash
#vim /etc/default/grub

GRUB_CMDLINE_LINUX="crashkernel=auto rd.lvm.lv=centos/root rd.lvm.lv=centos/swap isolcpus=1-7,9-15 intel_iommu=pt rhgb quiet"

#grub2-mkconfig -o /boot/grub2/grub.cfg
#reboot
```

**NOTE:** No need for NUMA settings since PoC system only have one socket.
Multi Tenant Cache Coherency

- Trade off of granting tenants maximum resources vs. providing cache coherency

Benefits

<table>
<thead>
<tr>
<th>Dedicating Cores</th>
<th>Sharing cores</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cache thrashing won’t occur</td>
<td>Tenants can “burst” consuming more cores</td>
</tr>
<tr>
<td>Better if tenant bandwidth is uniform</td>
<td>Better if tenant bandwidth isn’t uniform</td>
</tr>
</tbody>
</table>

Tenant 1  Tenant 2

Tenants share all cores
Enable Promiscuous & VLANs

- SR-IOV + Promiscuous is new in kernel 4.5
- Currently only Intel X710 & XL710 cards support it
- Must enable “trust mode”
  - kernel/iproute 4.5 or above
  - RHEL/CentOS 7.3 or above (backported patch)**

```
# echo 2 > /sys/class/net/ens2f0/device/sriov_numvfs
# ip link set dev ens2f0 vf 0 trust on
# ip link set dev ens2f0 vf 0 vlan 1000
# ip link set dev ens2f0 vf 1 trust on
# ip link set dev ens2f0 vf 1 vlan 2000

# ip link show
# ens2f0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP mode DEFAULT qlen 1000
   link/ether 3c:fd:fe:a2:0f:70 brd ff:ff:ff:ff:ff:ff
   vf 0 MAC 00:52:44:11:22:30, vlan 1000, spoof checking off, link-state auto
   vf 1 MAC 00:52:44:11:22:31, vlan 1001, spoof checking off, link-state auto
```

**https://access.redhat.com/solutions/2620231**
Build Bro Container (CentOS base image)

- Install Docker ([http://lmgtfy.com/?q=install+docker](http://lmgtfy.com/?q=install+docker))
- Dockerfile for installing Bro 2.5 container located at
  - [https://github.com/sealingtech/bro-docker](https://github.com/sealingtech/bro-docker)

```
# git clone https://github.com/sealingtech/bro-docker.git
# docker build -t bro-docker ./bro-docker/
```

... Build takes ~20 min... go enjoy your scone

```
[root@localhost ~]# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
bro-docker          latest              47a09a29cfd5        3 hours ago         568 MB
centos              latest              67591570dd29        5 weeks ago         192 MB
```
Performance Issues/Tuning

1. Lower Ring Descriptor (< 256 worked best in PoC) to keep IRQs from maxing out.

   ```
   #ethtool -G enp5s2 rx 128
   ```

2. Disable “irqbalance”

   ```
   #systemctl stop irqbalance
   ```

3. Check for cache misses and tune appropriately

   ```
   #perf stat
   ```
   ```
   ```
   ```
   ```

Shamelessly borrowed from
Mozilla/Intel/Suricata Performance Guide:  https://github.com/pevma/SEPTun
Helper Scripts on Github (Tests)

./reset-network.sh <num vfs>
- deletes all namespaces
- automatically adds VFs, adds vlans, etc for testing
- reloads kernel drivers
- some performance settings

./start-bro-docker.sh <interface>
- places interface in docker instance’s namespace
- creates eth1 inside of container and matches to SR-IOV interface
- configures bro with appropriate CPU pinning
- starts bro application in container
Steps:

1. Start-up one, two Bro containers (shared & Isolated)
2. Send traffic to each VLAN with tcpreplay
3. Profit
Demo
Where can we go from here?
Security offering for Data-Center Providers…

- OvS supports SPAN ports to other VMs/Containers.
- Use case for Bro micro-service or NSMaaS?

---

Sealing Technologies, Inc.
www.sealingtech.org
Policy-based Use Case for enhanced inspection

- Used in environments that require resource intensive scripts

- Traffic is directed to “Normal” or “Enhanced” Bro instance based on 5-tuple

- Better than bypassing inspections all-together for normal traffic (Google/AWS/etc.)
Thank you!

Questions?