360 互联网安全中心

Virtualization System Vulnerability Discovery Framework

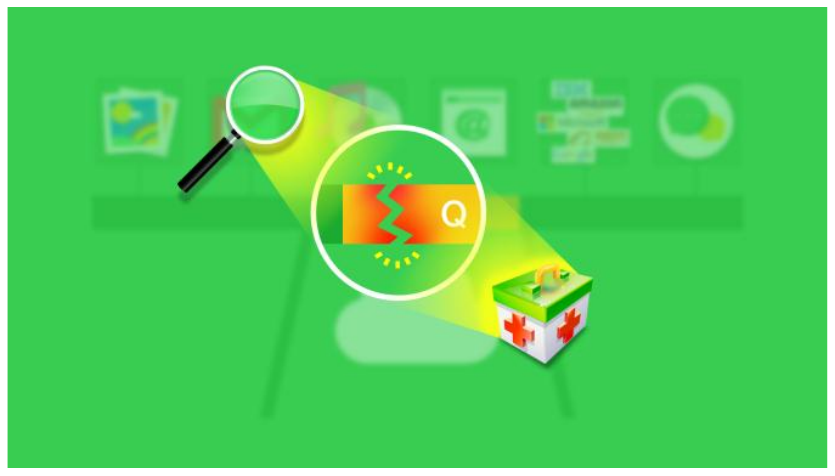
Speaker: Qinghao Tang

Title：360 Marvel Team Leader

360 Marvel Team

Established in May 2015, the first professional could computing and virtualization security team in China. Focusing on attack and defense techniques in virtualization system.

- fuzzing framework
- guest machine escape technology
- Hypervisor risk defense technology

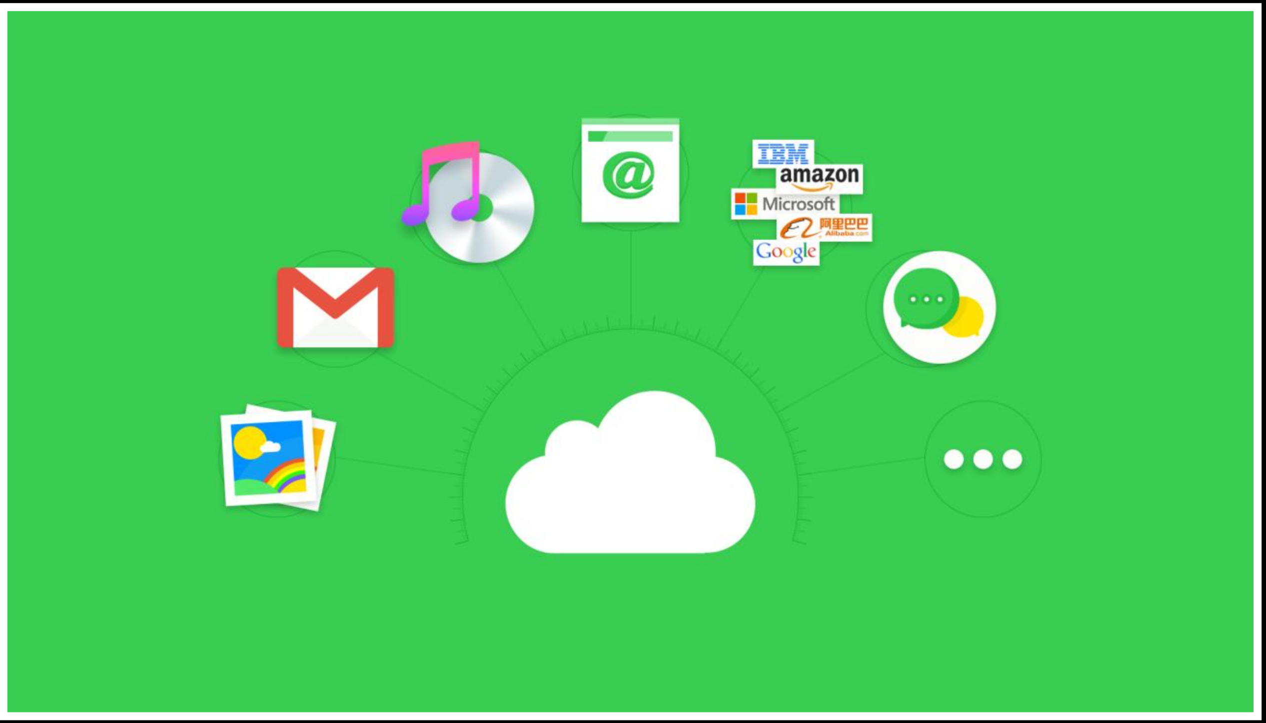


Agenda

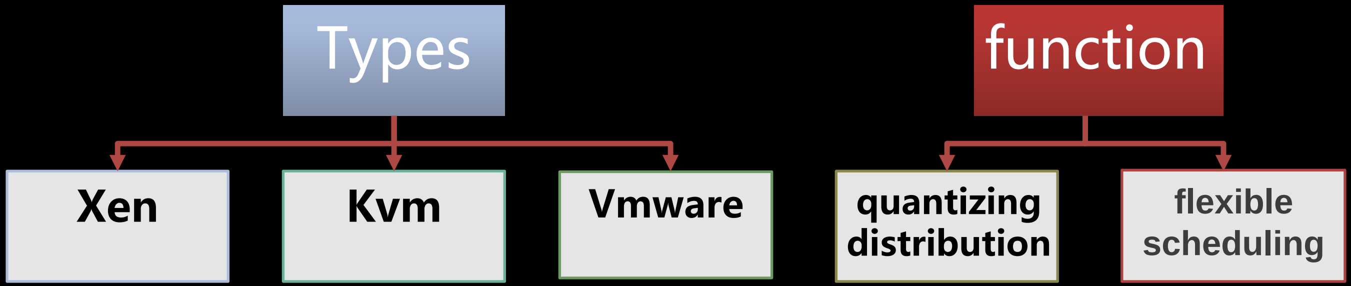
- Virtualization System Attack Surface
- The fuzzing framework
- Case study

Virtualization System Attack Surface

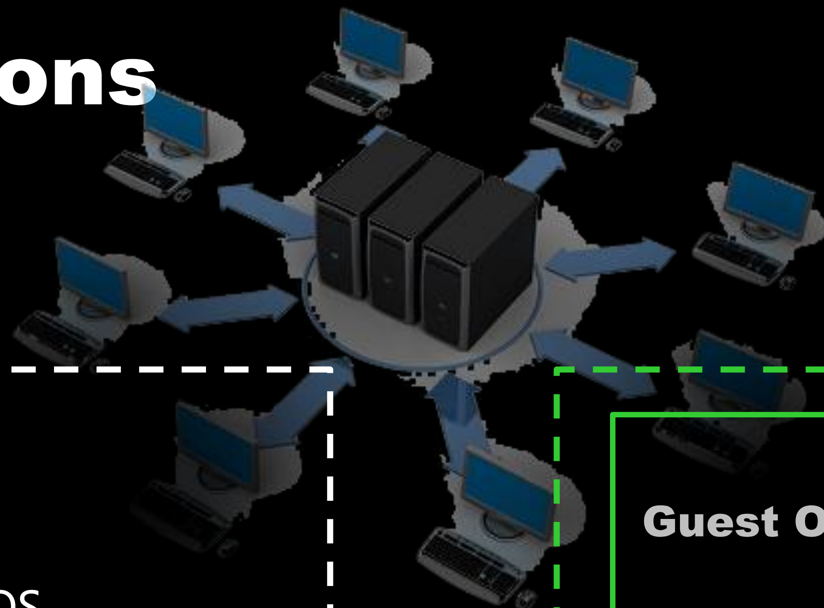
Cloud Computing



Hypervisor


Types
Xen
Kvm
Vmware
function
quantizing distribution
flexible scheduling

Distinctions



OS

Physical Devices

Normal Server

Guest OS

Hardware emulator

Guest OS

Hardware emulator

Hypervisor

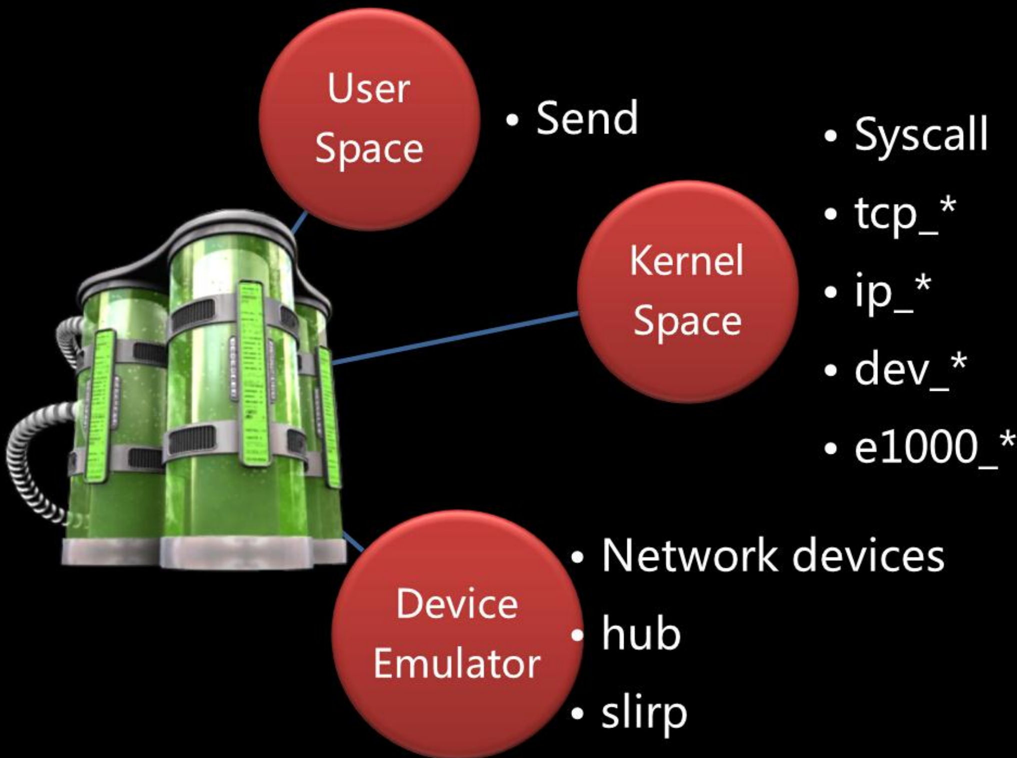
Physical Devices

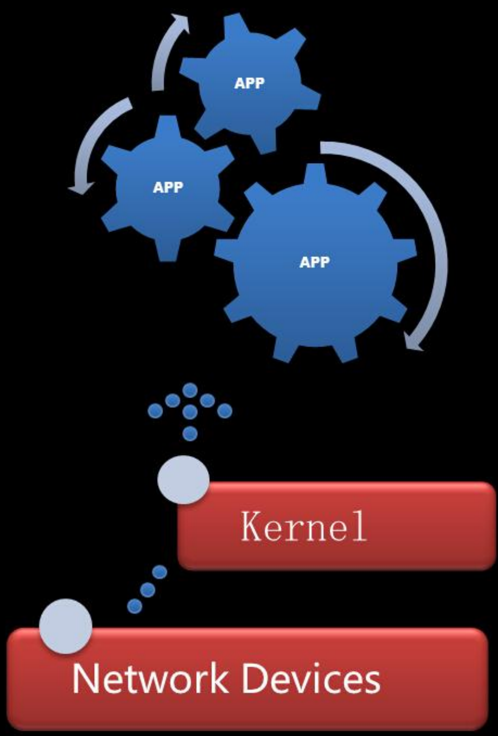
Virtualization Server

Attacking Processes in cloud computing

1. Enter VM via web or other devices
2. Exploit virtualization system vulnerabilities to escape VM
3. lateral movements to others VMs on host
4. Access to host network

Operation Principles of device emulators





The attack surface

- Hardware virtualization components'　diversity

  Qemu：30+

  Vmware：20+

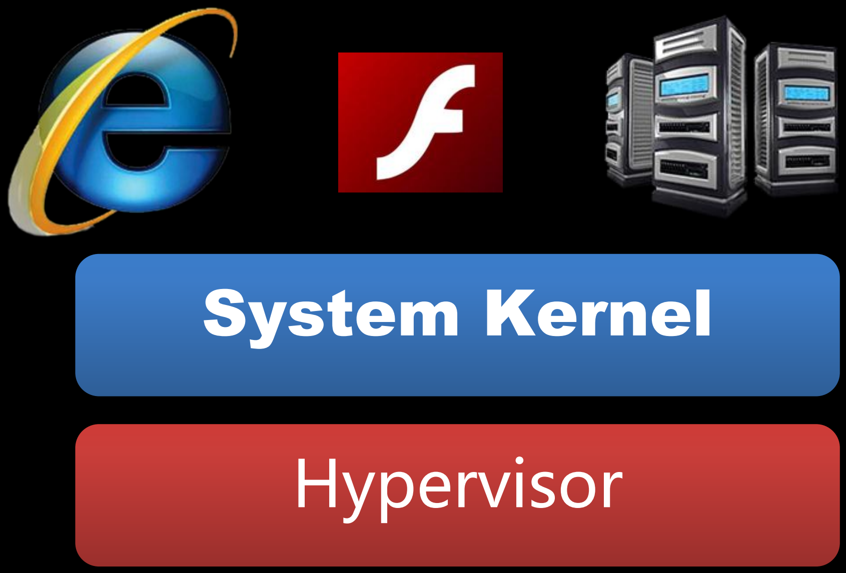
- Bridge between inside-outside

  VM os -> device emulators -> Host os

- Related Vulnerabilities result big dangers

Compare to traditional targets

- Hardware virtualization focus on lower



- Testing data totally different

Vulnerabilities found by us

CVE-2015-5225	CVE-2015-5279	CVE-2015-6815
CVE-2015-6855	CVE-2015-8345	CVE-2015-7504
CVE-2015-7549	CVE-2015-8567	CVE-2015-8568
CVE-2015-8558	CVE-2015-8613	CVE-2015-8701
CVE-2016-1568	CVE-2016-1570	CVE-2016-2392

Fuzzing Framework

Basic intro

Attack surface ： hardware virtual components

Environment ： qemu ， vmware

Testing results ： more than 25 vulnerabilities

Challenges ： lower layers hard to predict ;

Methods for testing hardware virtual components

1. Analyze data which flowed to components
2. Change flowed-in data’ s contents and timing
3. Recording all of tiny abnormal activities
4. Analyze abnormal activities, find reasons
5. optimize fuzz framework

Other factors of fuzz framework

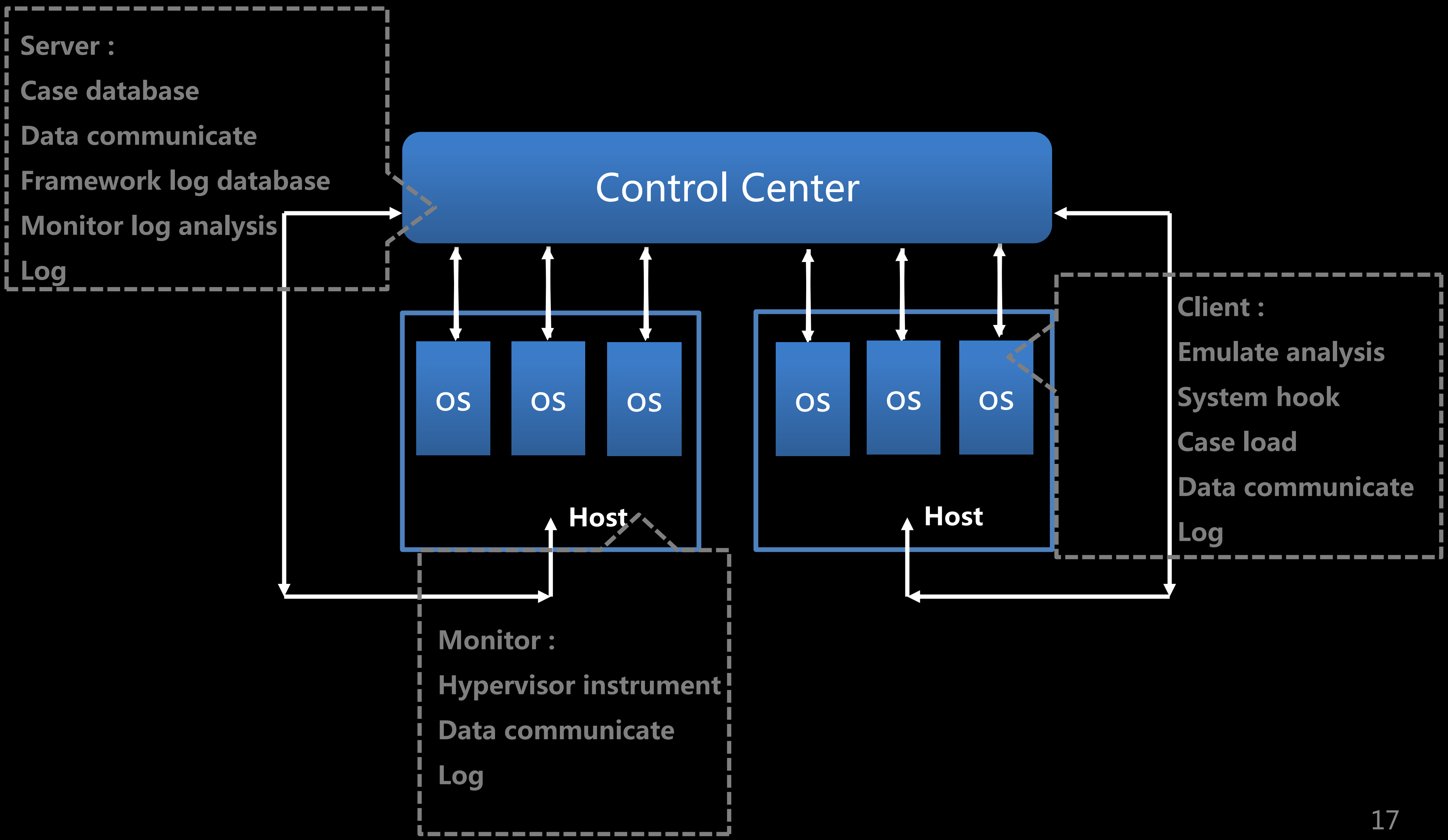
1.Flexibility (other OS)

- vm in Linux
- coding in C and Python

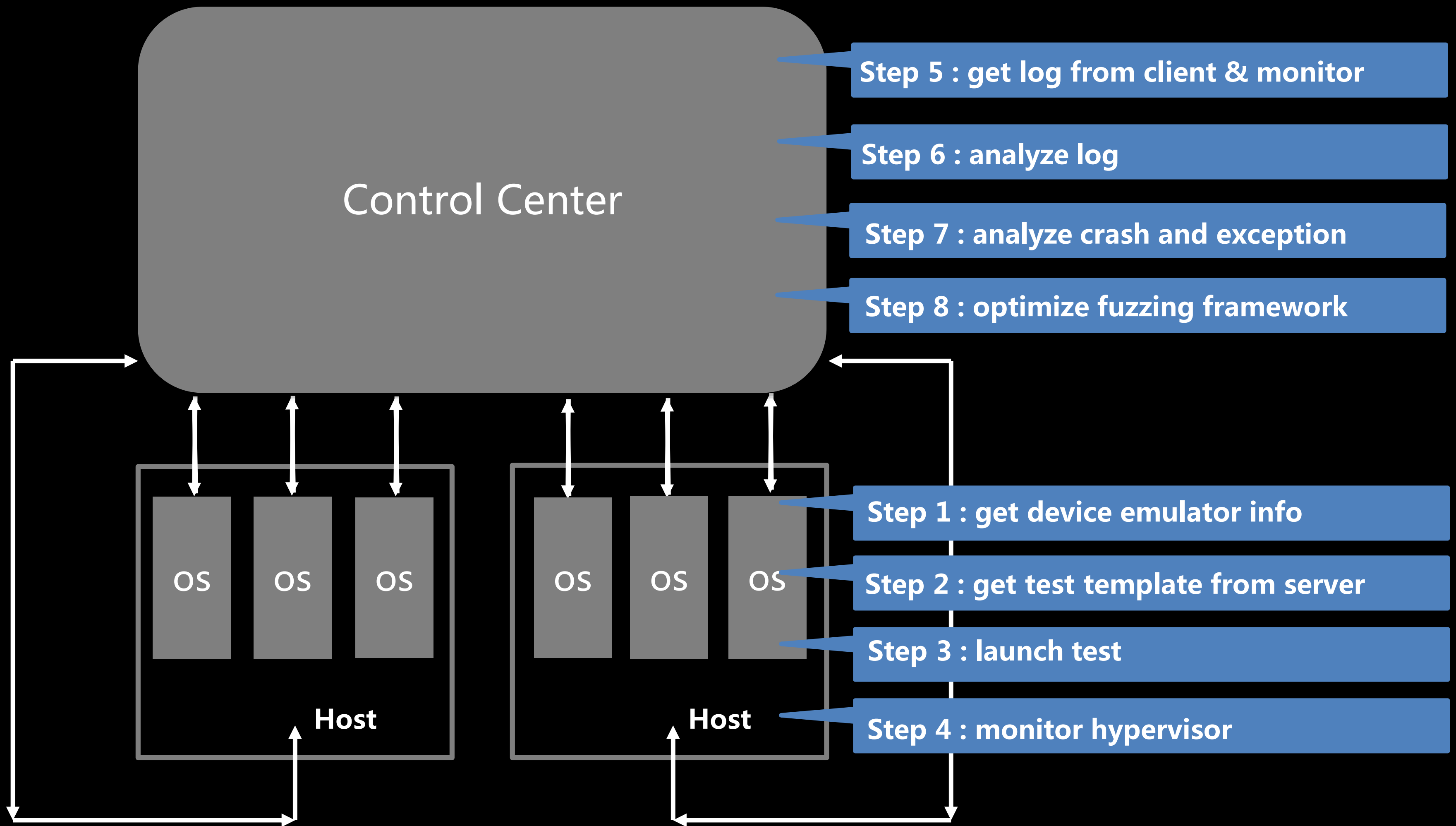
2. Deeply understand VM system

- language for coding
- development environment
- coding style

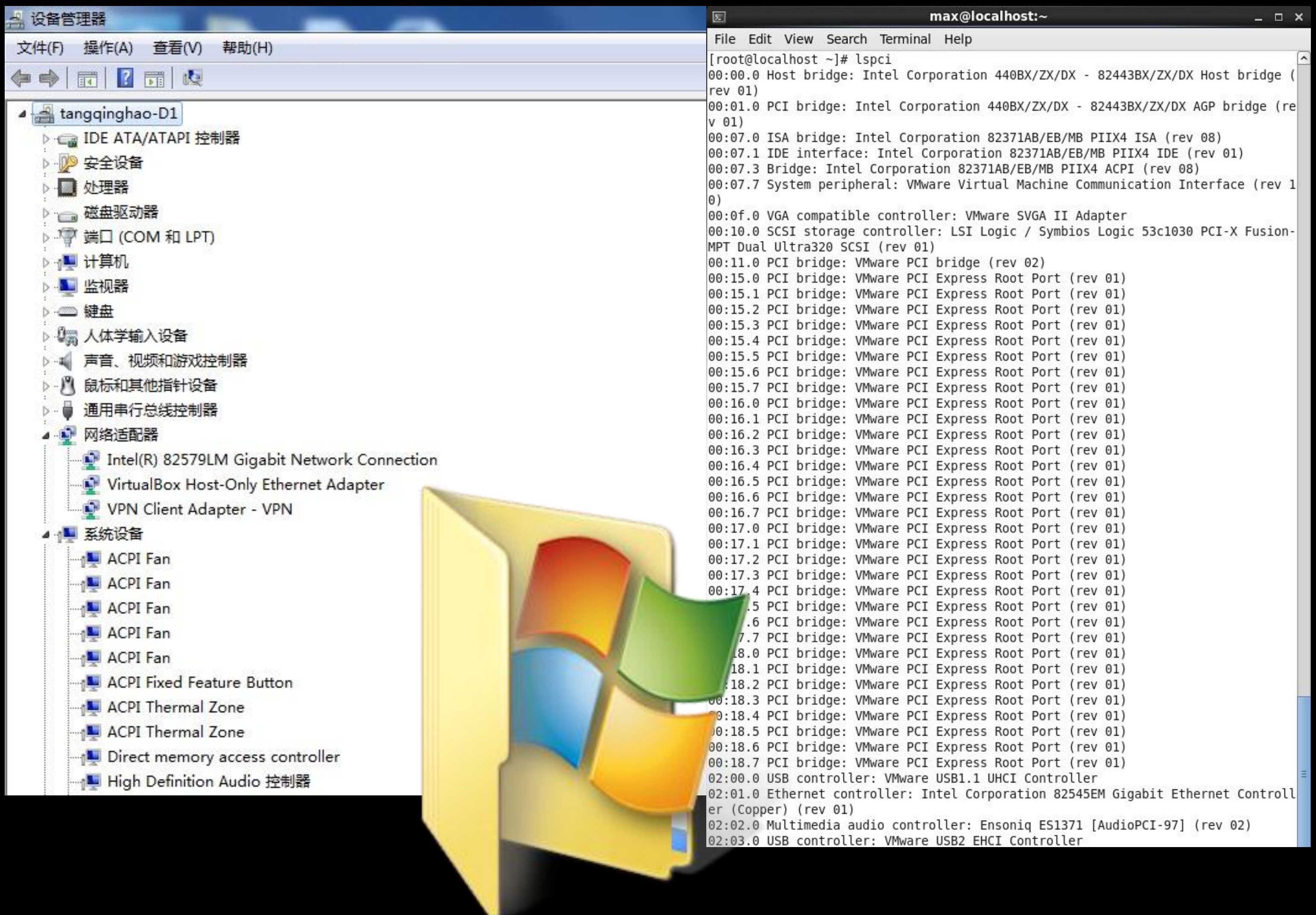
Fuzz framework structure



Fuzz framework working flow



Get target components info

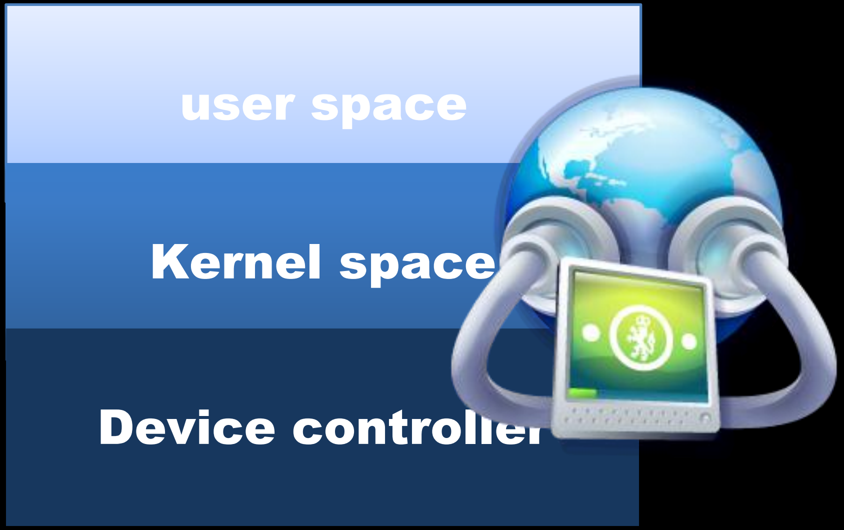


Testing data

- Device access ports
- Device deal with structures used by data.
- Device data processing

Testing data attacks

- User space: generate testing dat, send request to client kernel
- Kernel space: apply for memory, fill memory, send info to ports
- Device emulator：testing data flow inside，trigger exceptions



Monitor

VM management

- Snapshot
- Reboot
- VM device editing

Dynamic debugging

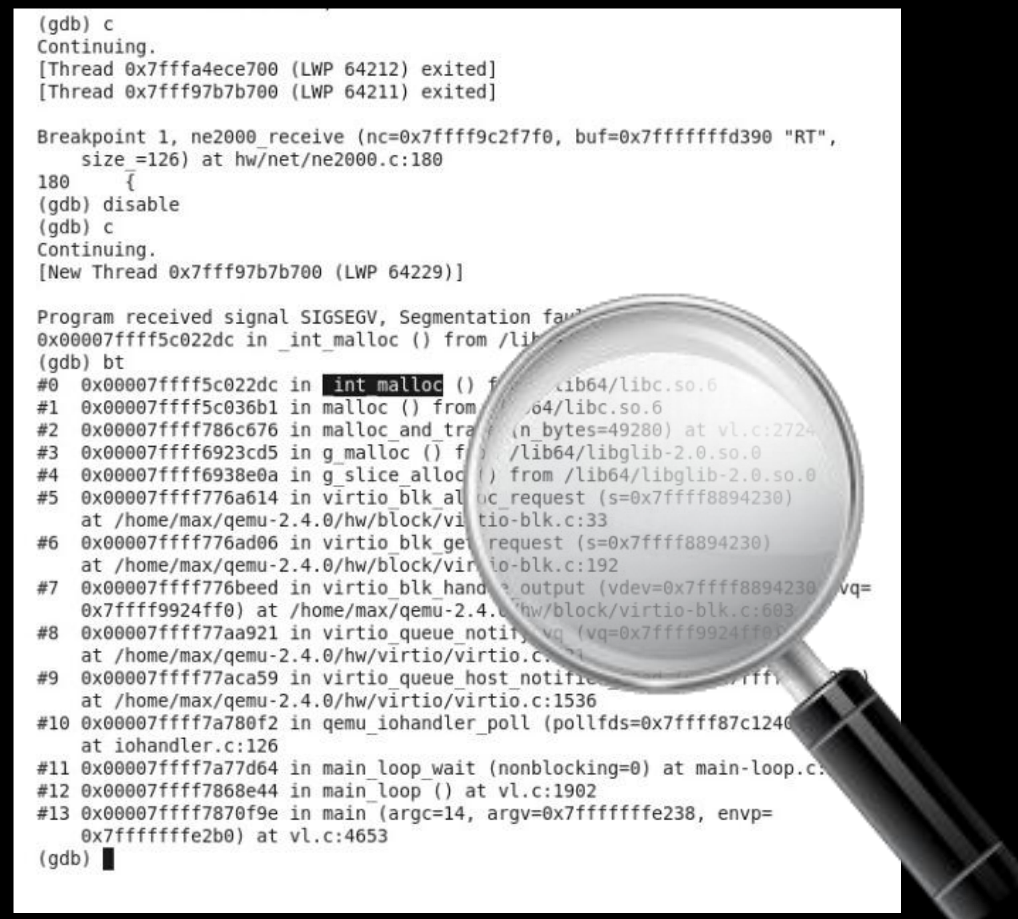
- Debugging Mode on Start
- Load Debugging Plugin

VM processing log

- User space
- Kernel space

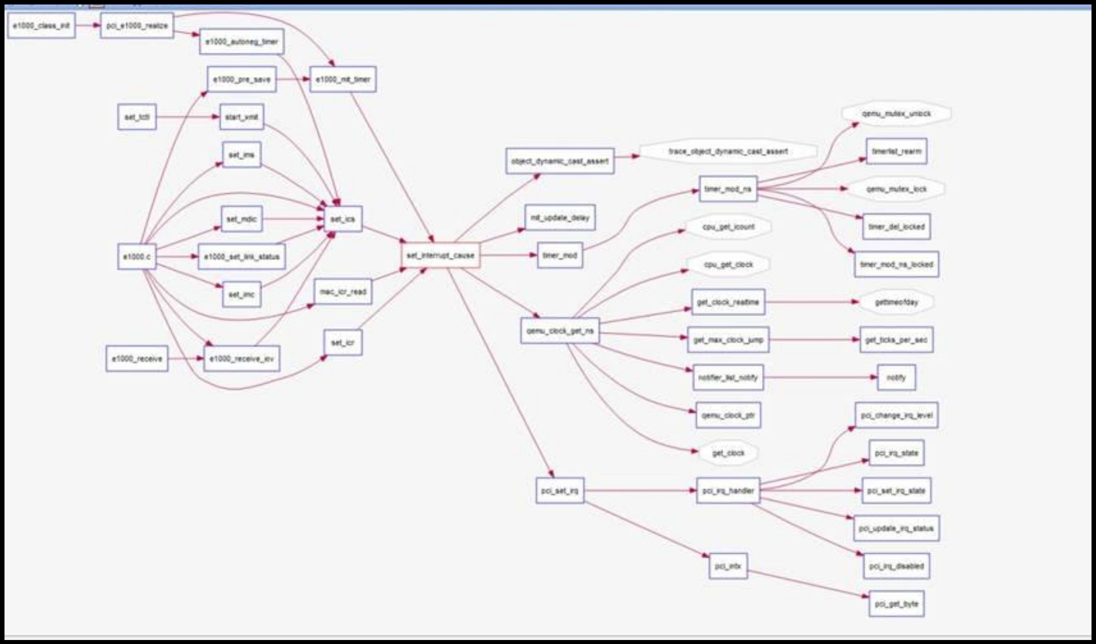
Exceptions occur in device emulator

- VM os crash
- Hypervisor crash
- Invisible results



Advanced monitoring skills

- Dynamic
- Static



Optimize fuzz framework by using log data

- Client log

  Decrease invalid combinations

- Monitor log

  Promote coverage

- Server log

Limitation & Future

- Permission limitation
- More kinds of virtualization systems：

    Hyper-V；VMWARE

- More attack surfaces：

    hypercall；virtio；guest machine client

- About open source project

Case Study

Principle of e1000 Network Device

•Initialization

Port Allocation ， Address Mapping

Device Status Setting, Resource Allocation

•Data Transfer

'Write Command' to device TDT register

process of descriptor

3 types descriptor ： context ， data ， legacy

data xfer

set status ， wait for next instruction

•Processing Details

Circular Memory

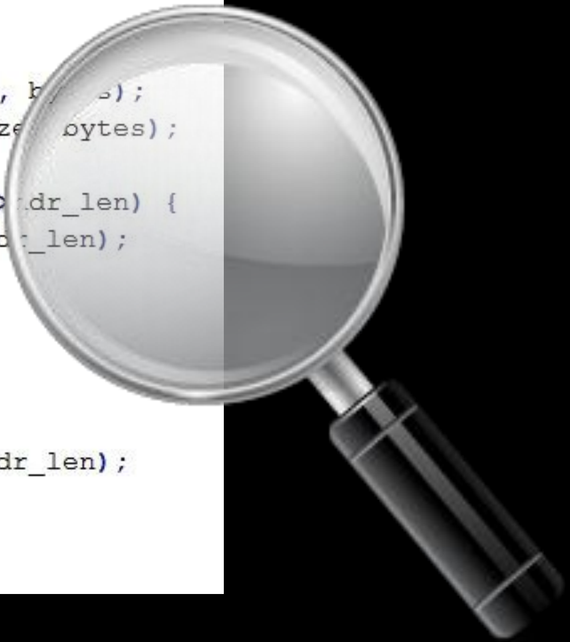
TSO ： tcp segmentation/flow control.

E1000 vulnerability analysis

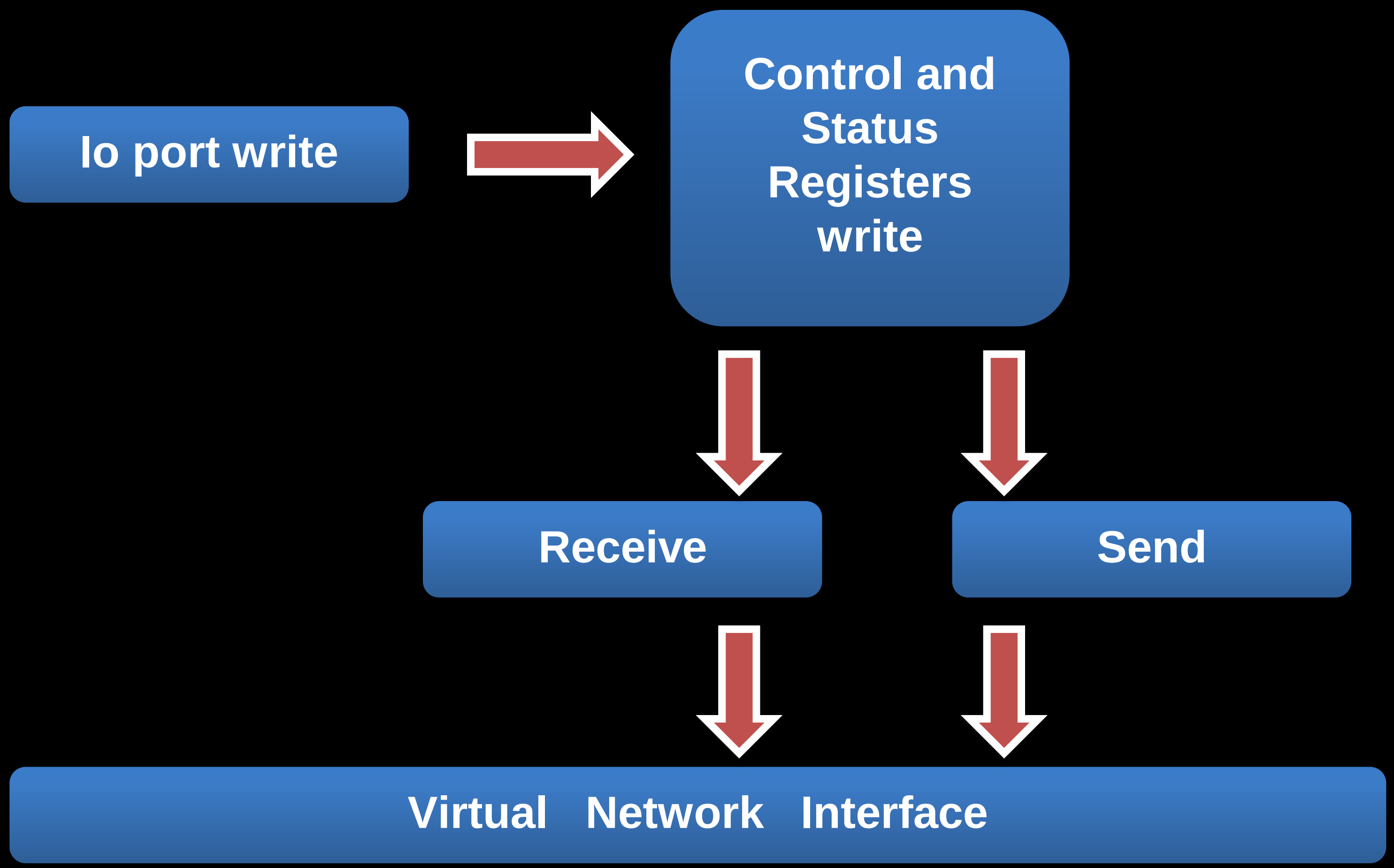
- Qemu e1000 Network Device
- Vmware e1000 Network Device

```
do {
    bytes = split_size;
    if (tp->size + bytes > msh)
        bytes = msh - tp->size;

    bytes = MIN(sizeof(tp->data) - tp->size, bytes);
    pci_dma_read(d, addr, tp->data + tp->size, bytes);
    sz = tp->size + bytes;
    if (sz >= tp->hdr_len && tp->size < tp->hdr_len) {
        memmove(tp->header, tp->data, tp->hdr_len);
    }
    tp->size = sz;
    addr += bytes;
    if (sz == msh) {
        xmit_seg(s);
        memmove(tp->data, tp->header, tp->hdr_len);
        tp->size = tp->hdr_len;
    }
} while (split_size -= bytes);
```



Pcnet network card emulator working processes



Pcnet vulnerability analysis

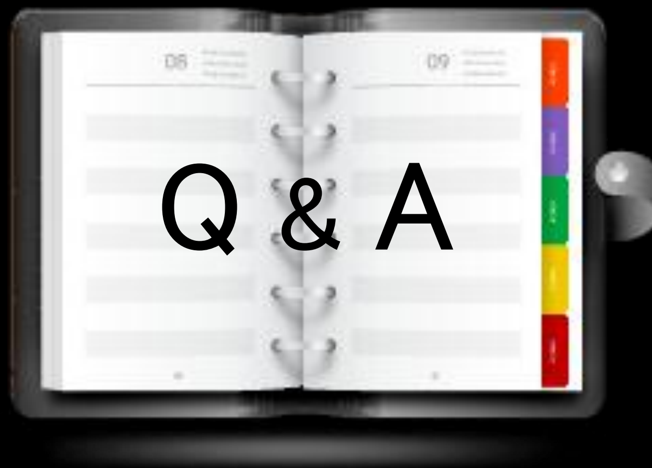
- Qemu pcnet Network Device

```
} else if (s->looptest == PCNET_LOOPTEST_CRC ||
            !CSR_DXMTFCS(s) || size < MIN_BUF_SIZE+4) {
    uint32_t fcs = ~0;
    uint8_t *p = src;

    while (p != &src[size])
        CRC(fcs, *p++);
    *(uint32_t *)p = htonl(fcs);
    size += 4;
```

Summary

Stay tuned for more achievements by 360 Marvel Team



Email：tangqinghao@360.cn
QQ：702108451