Charm: Facilitating Dynamic Analysis of Device Drivers of Mobile Systems

Seyed Mohammadjavad Seyed Talebi, Hamid Tavakoli, Hang Zhang, Zheng Zhang, Ardalan Amiri Sani, Zhiyun Qian

UC Irvine **UC Riverside**

What is the problem?

Key ideas to solve the problem

Design

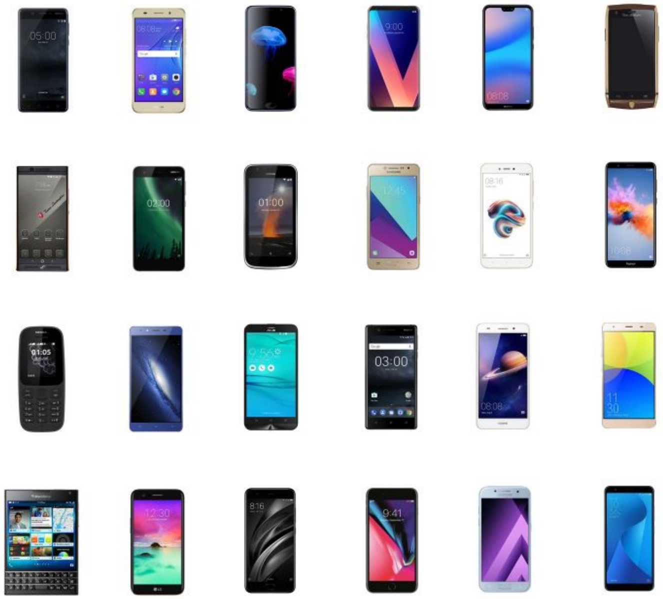
Evaluation

Summary

Security of mobile systems is vital

Mobile systems are diverse

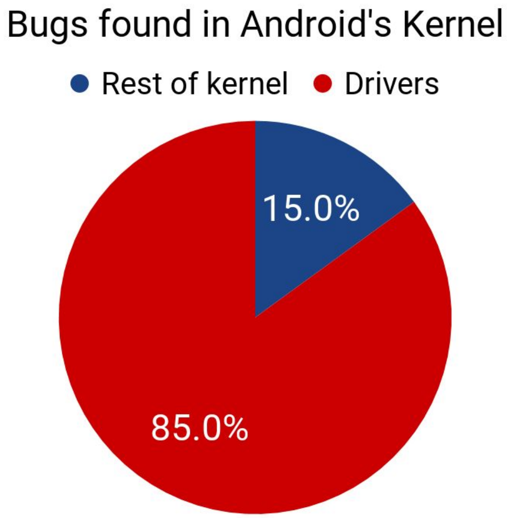
- More than **1,000** Android device manufacturers
- More than **24,000** distinct Android devices



Diverse hardware → many device drivers

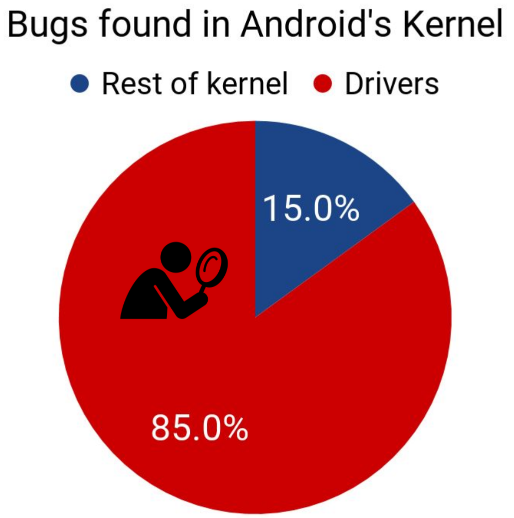
Vendors competition → more features
→ more hardwares → more device drivers

Device drivers are a major risk to the security of mobile systems



Source: Jeffrey Vander Stoep. 2016. Android: protecting the kernel. In Linux Security Summit. Linux Foundation.

How to investigate bugs in device drivers of mobile systems?

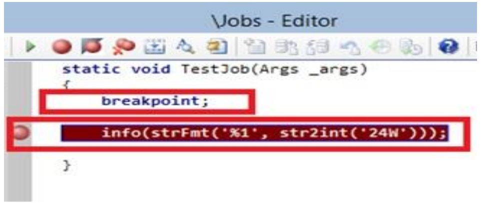


Source: Jeffrey Vander Stoep. 2016. Android: protecting the kernel. In Linux Security Summit. Linux Foundation.

Dynamic analysis is useful to find vulnerabilities

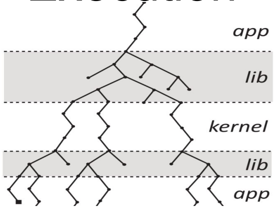
Fuzzing

Interactive debugging

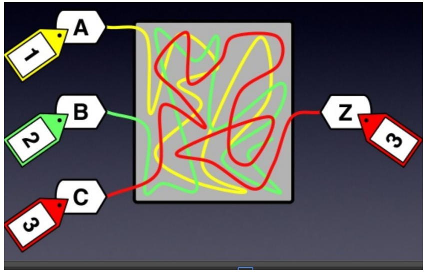


Record-and-replay

Selective Symbolic Execution



Dynamic taint analysis



Many existing dynamic analysis tools use virtual machines

Fuzzing

- kAFL
- Digtool

Interactive debugging

- GDB

Record-and-replay

- QEMU

Selective Symbolic Execution

- S^2E

Dynamic taint analysis

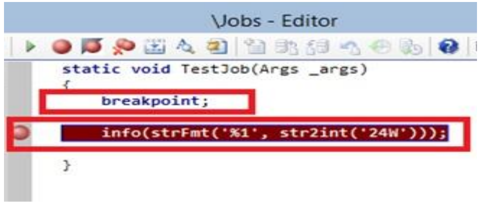
- DECAF

Many existing dynamic analysis tools use virtual machines

Fuzzing

- kAFL
- Digtool

Interactive debugging



- GDB

Record-and-replay

- QEMU

Selective Symbolic Execution

- S^2E

Dynamic taint analysis

- DECAF

Many existing dynamic analysis tools use virtual machines

Fuzzing

- kAFL
- Digtool

Interactive debugging

- GDB

Record-and-replay

- QEMU

Selective Symbolic Execution

- S^2E

Dynamic taint analysis

- DECAF

Many existing dynamic analysis tools use virtual machines

Fuzzing

- kAFL
- Digtool

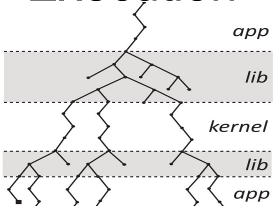
Interactive debugging

- GDB

Record-and-replay

- QEMU

Selective Symbolic Execution



- S^2E

Dynamic taint analysis

- DECAF

Many existing dynamic analysis tools use virtual machines

Fuzzing

- kAFL
- Digtool

Interactive debugging

- GDB

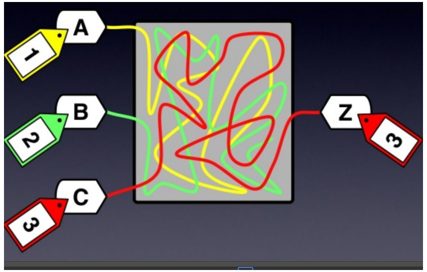
Record-and-replay

- QEMU

Selective Symbolic Execution

- S^2E

Dynamic taint analysis



- DECAF

Applying these tools to device drivers in mobile systems is hard

Hardware assisted virtual machine

Not available

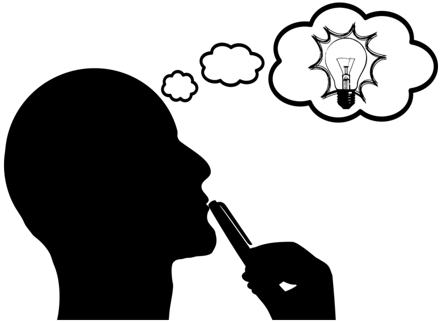
Applying these tools to device drivers in mobile systems is hard

Hardware assisted virtual machine

Not available

Software only virtual machine

Poor performance



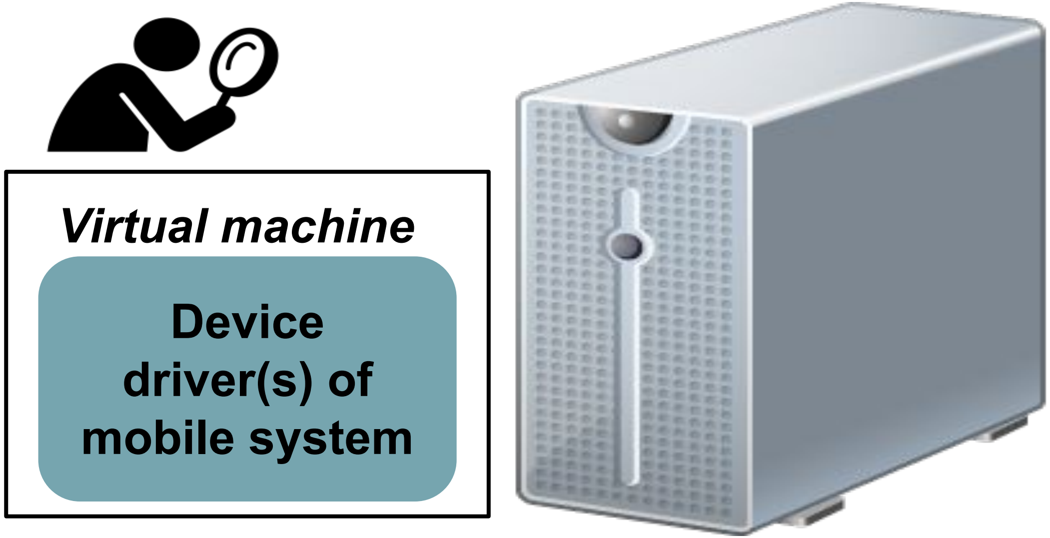
Key ideas to solve the problem

Design

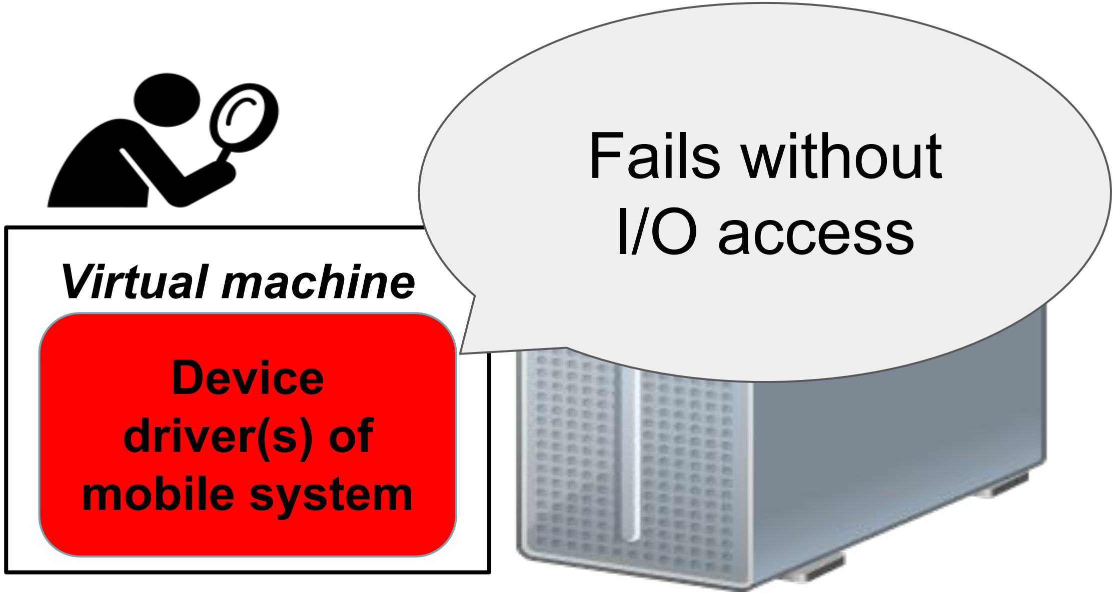
Evaluation

Summary

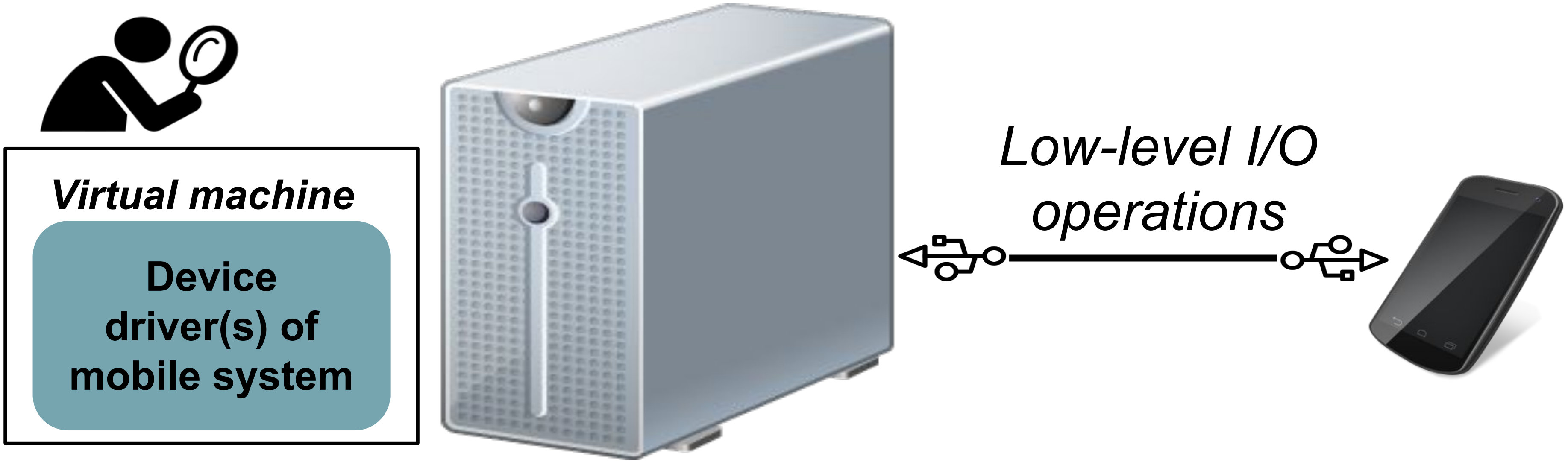
Key idea 1: running device drivers of a mobile system in a virtual machine on a workstation

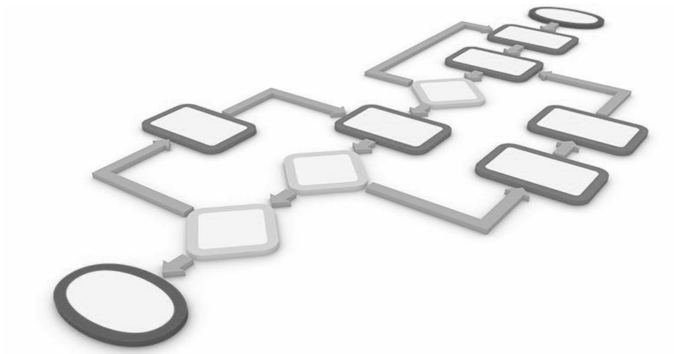


Key idea 1: running device driver of a mobile system in a virtual machine on a workstation



Key idea 2: use the mobile device to serve low-level I/O operations



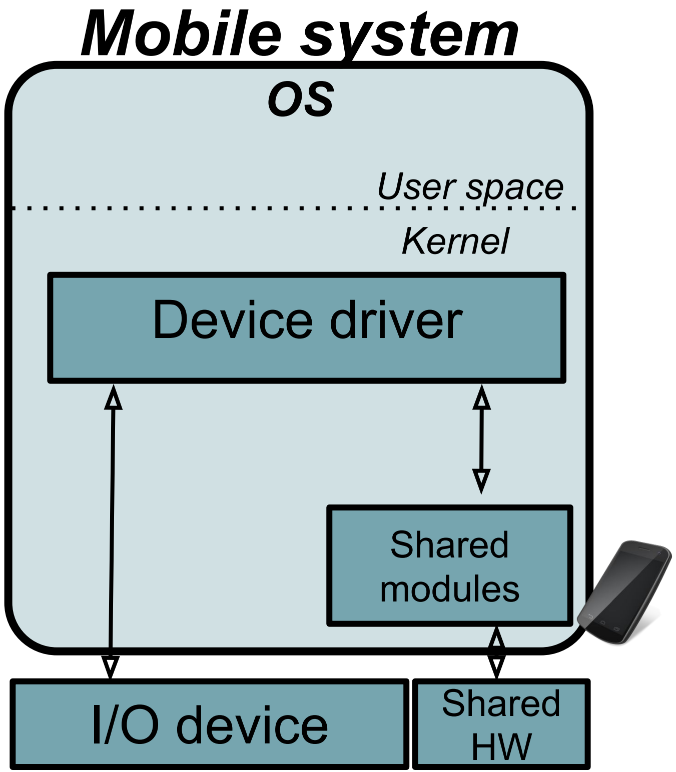


Design

Evaluation

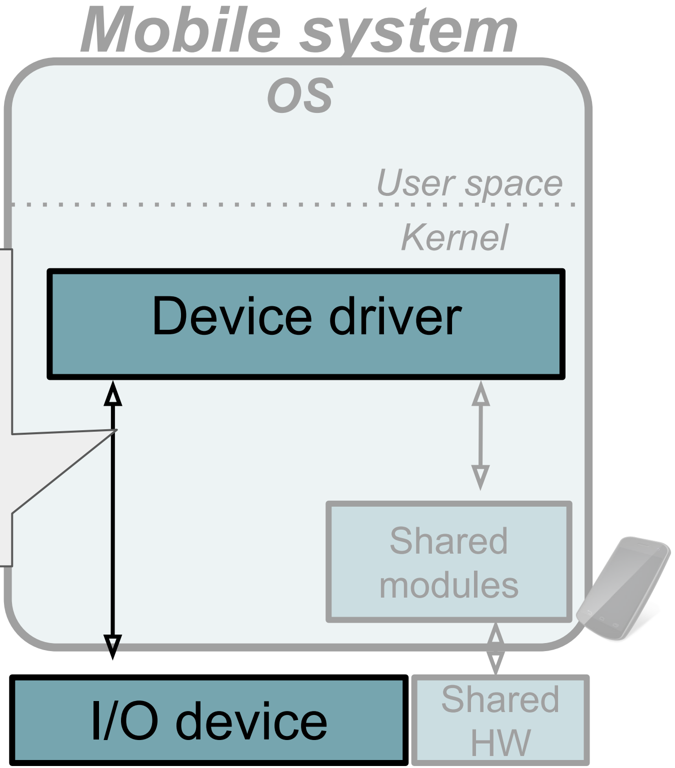
Summary

Device driver of a mobile system: a closer look

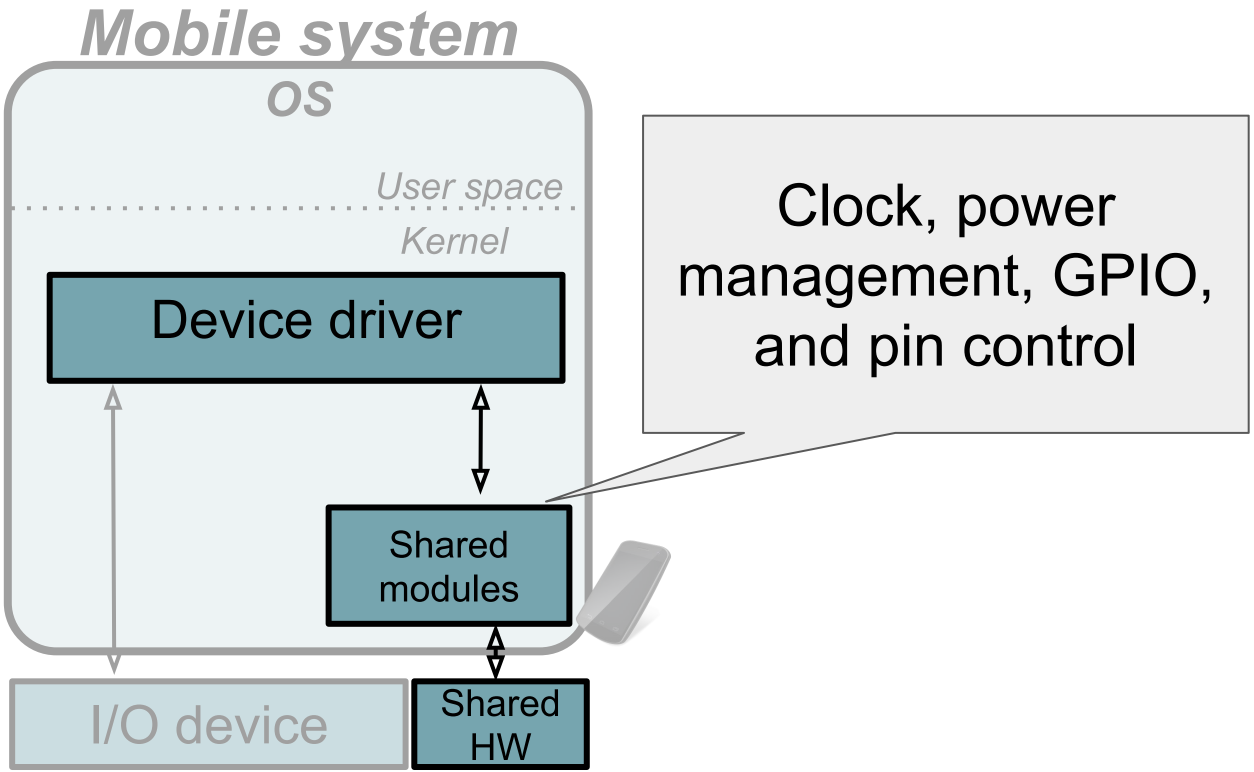


Device driver of a mobile system: a closer look

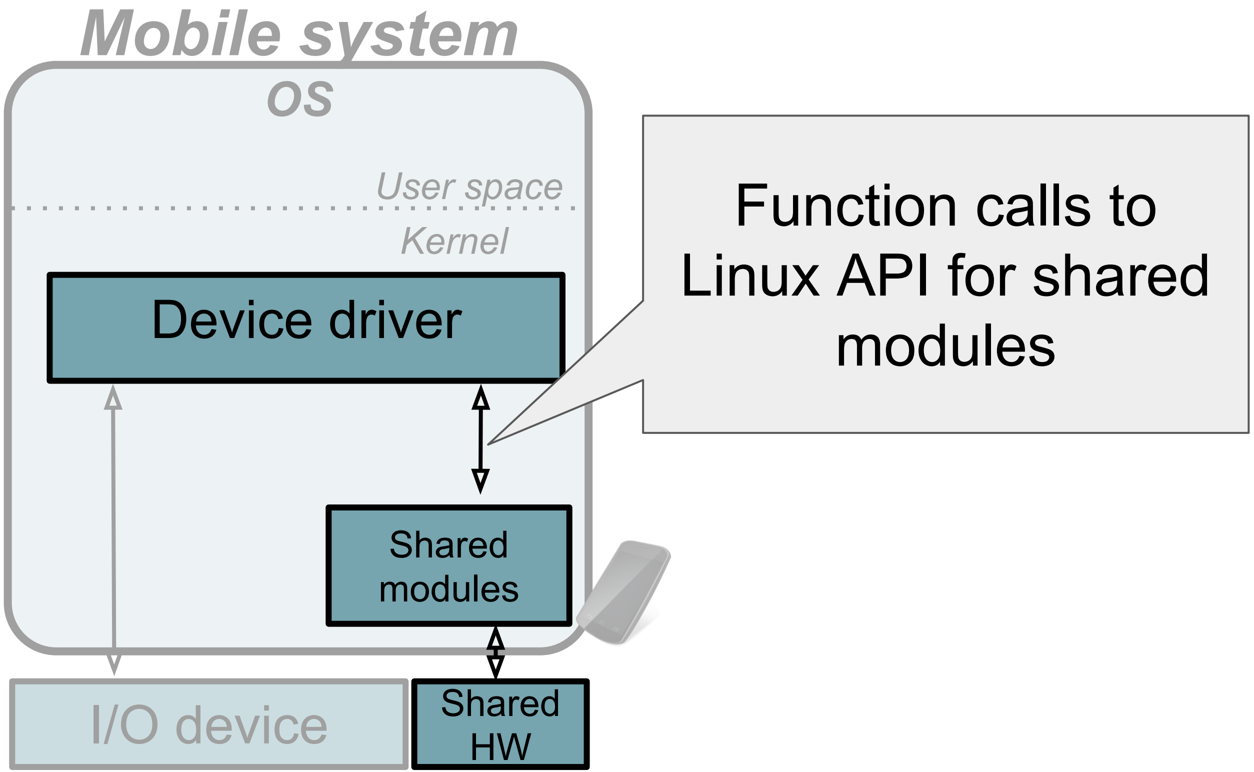
- Memory mapped register read/writes
- Interrupt



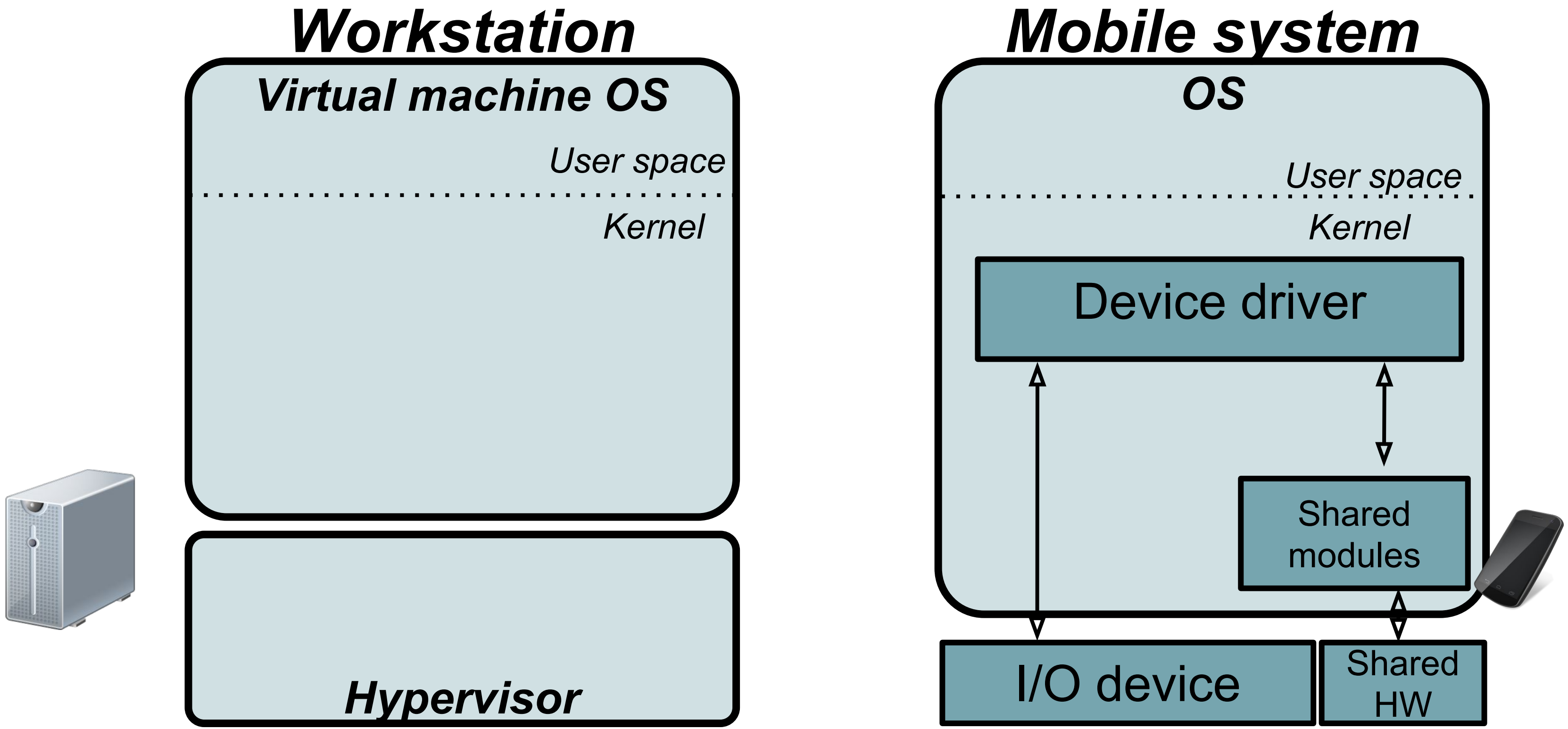
Device driver of a mobile system: a closer look



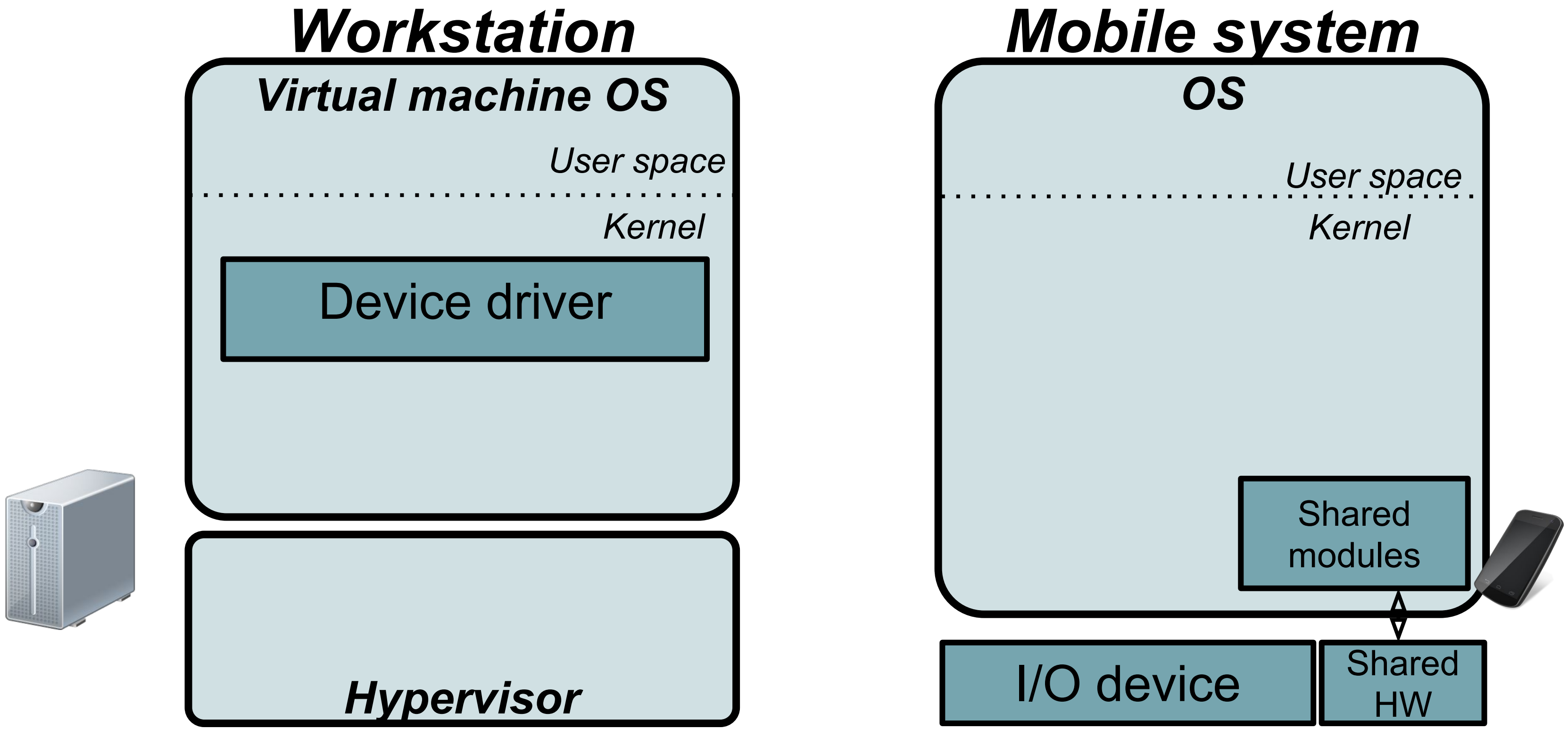
Device driver of a mobile system: a closer look



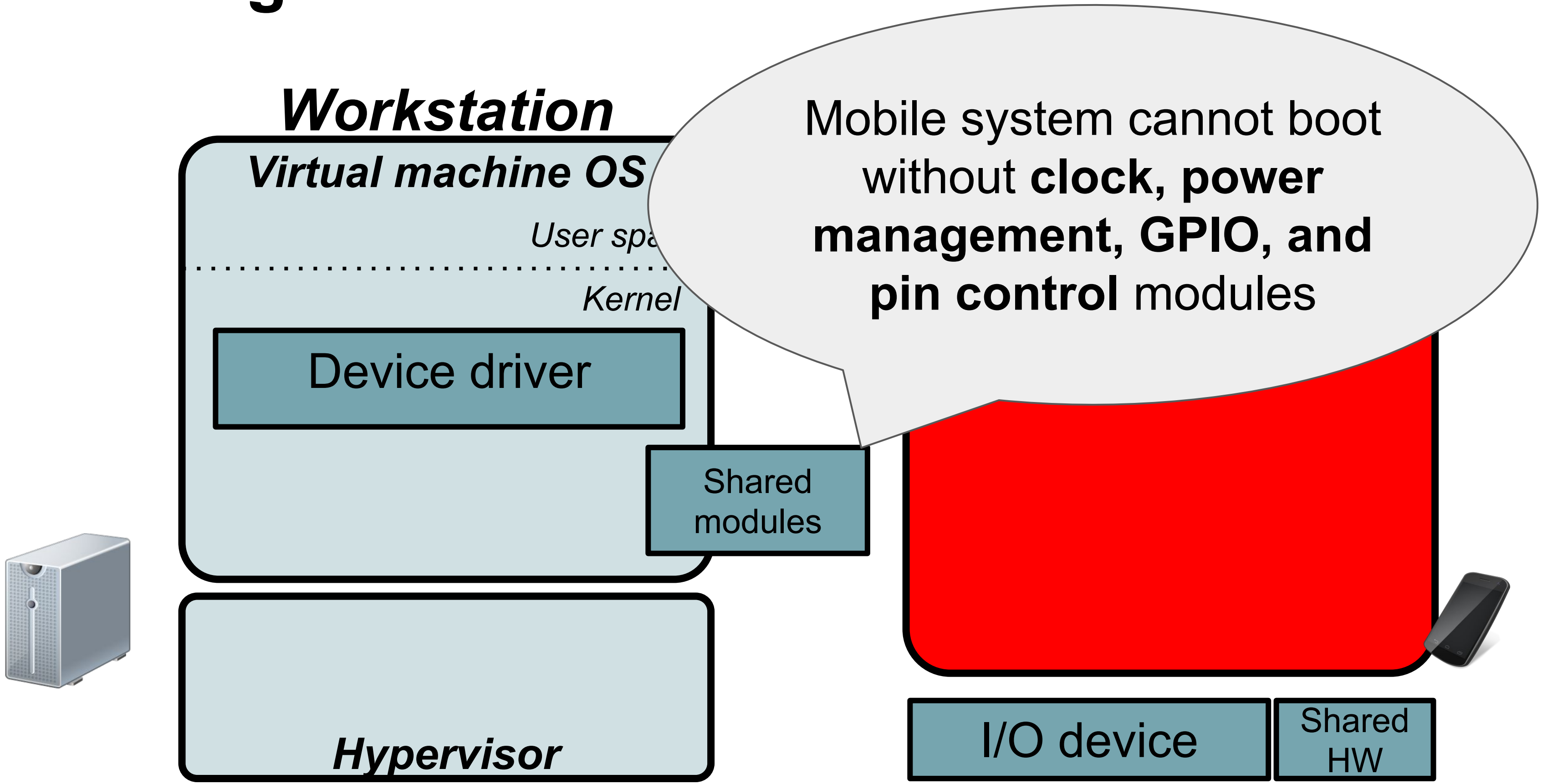
Move the device driver to a workstation



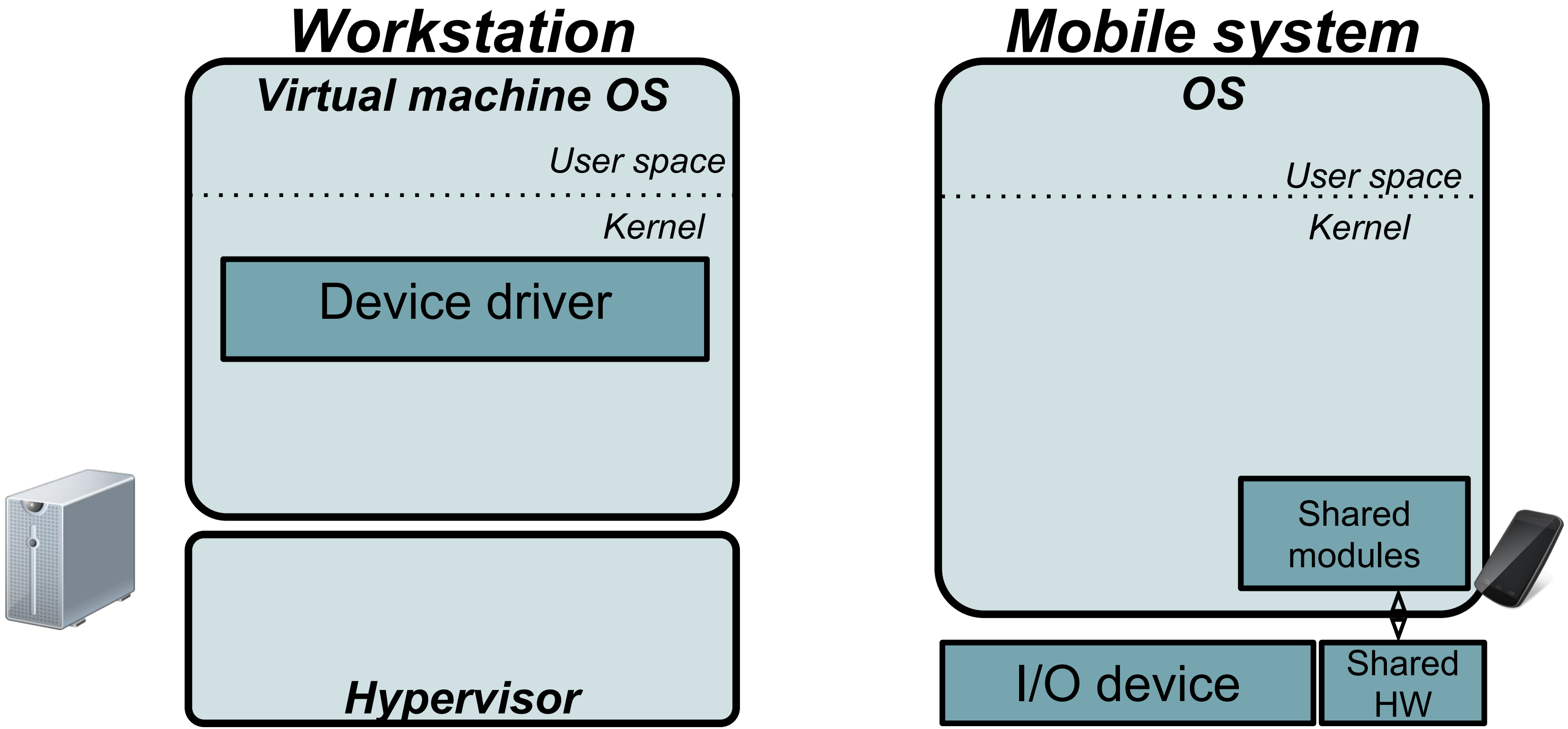
Move the device driver to a workstation



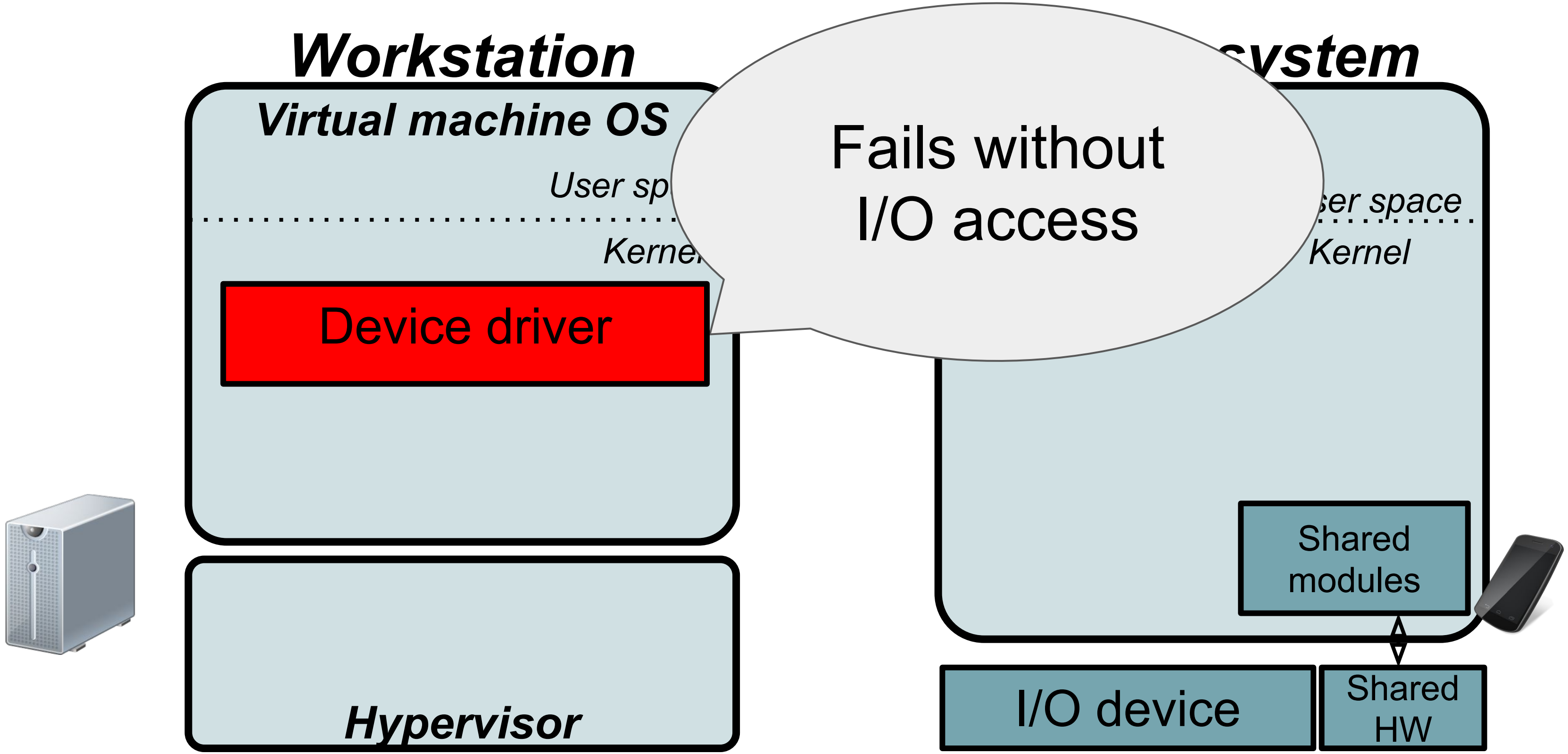
Challenge: cannot move shared modules



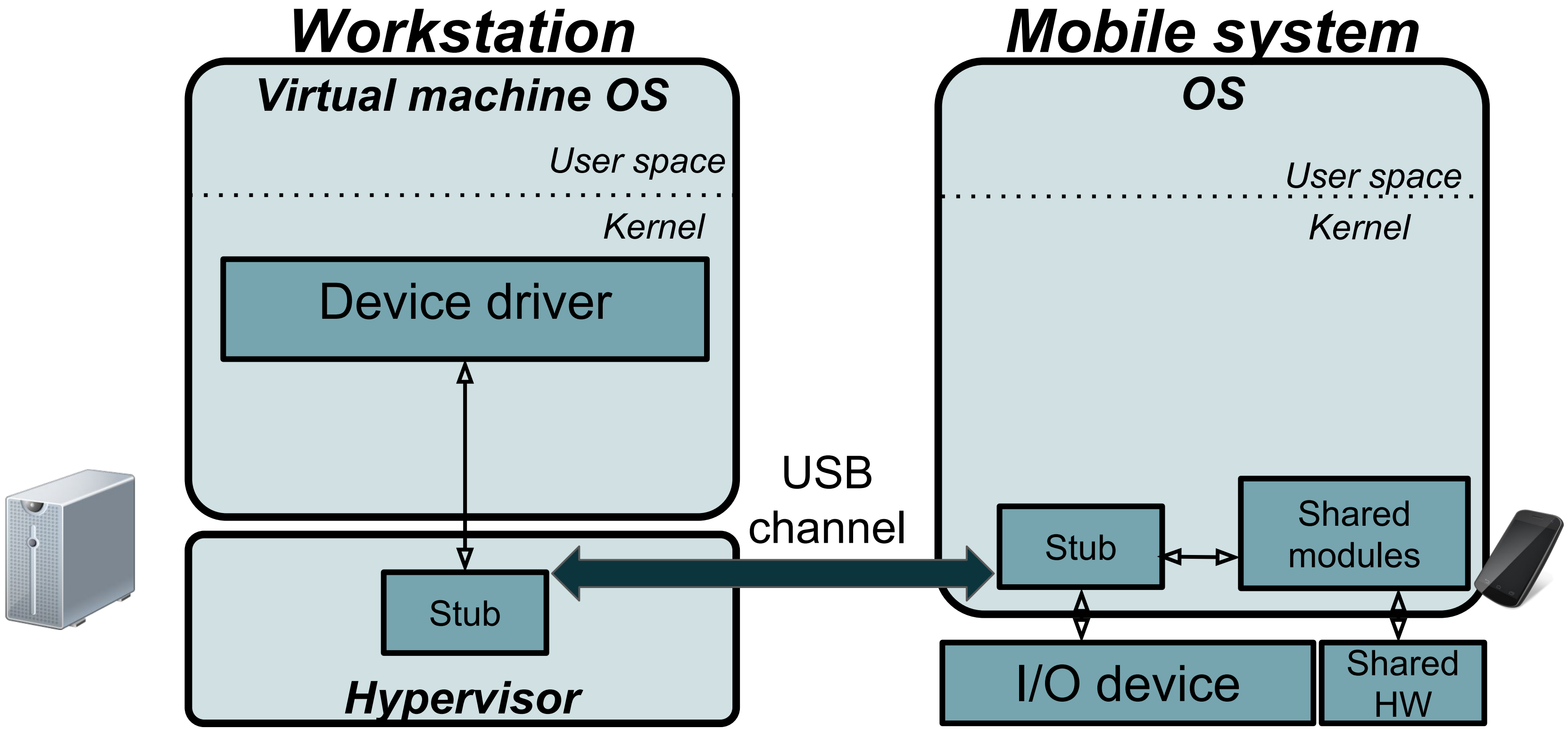
Do not move shared modules



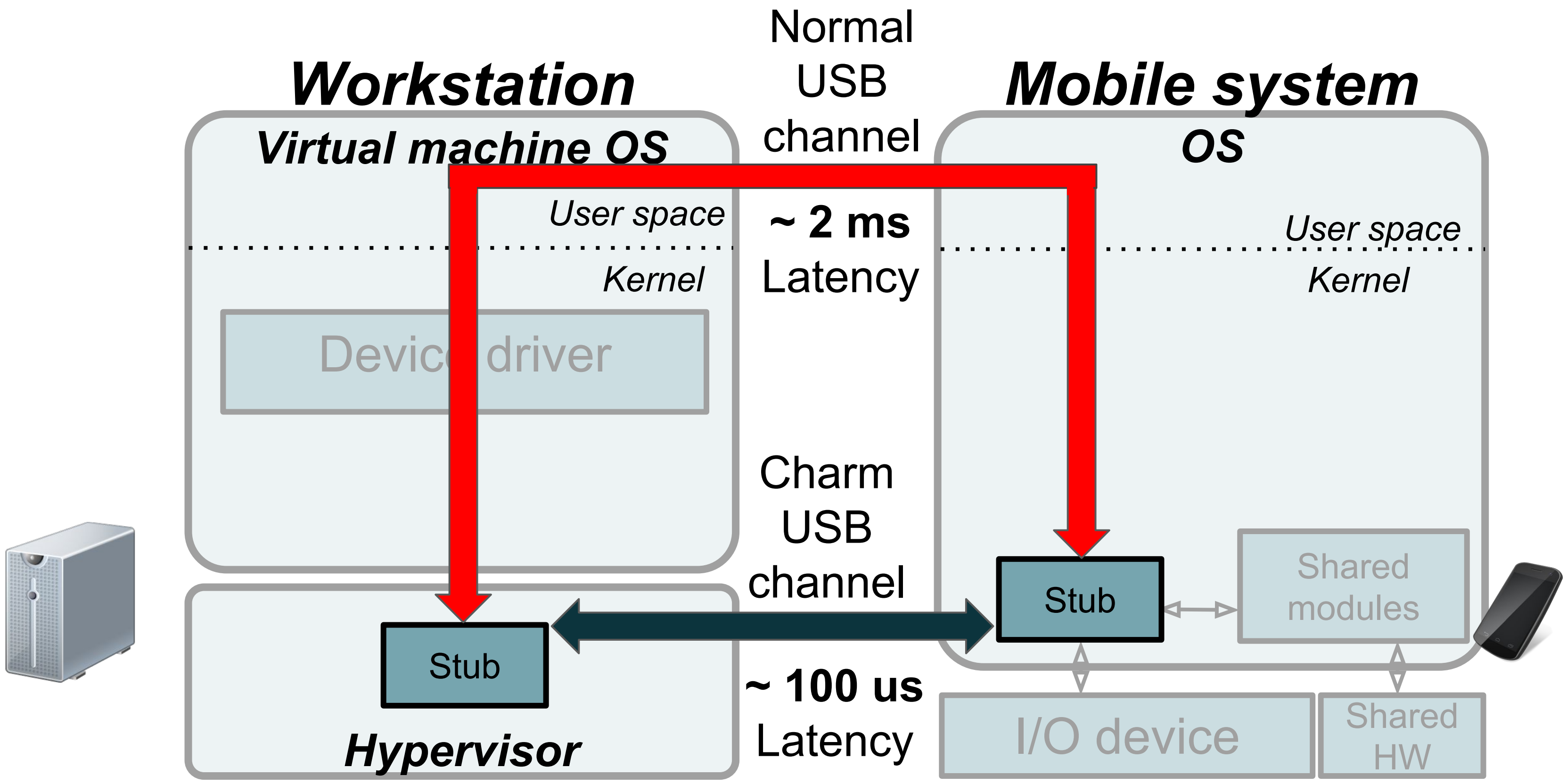
Remote I/O operations



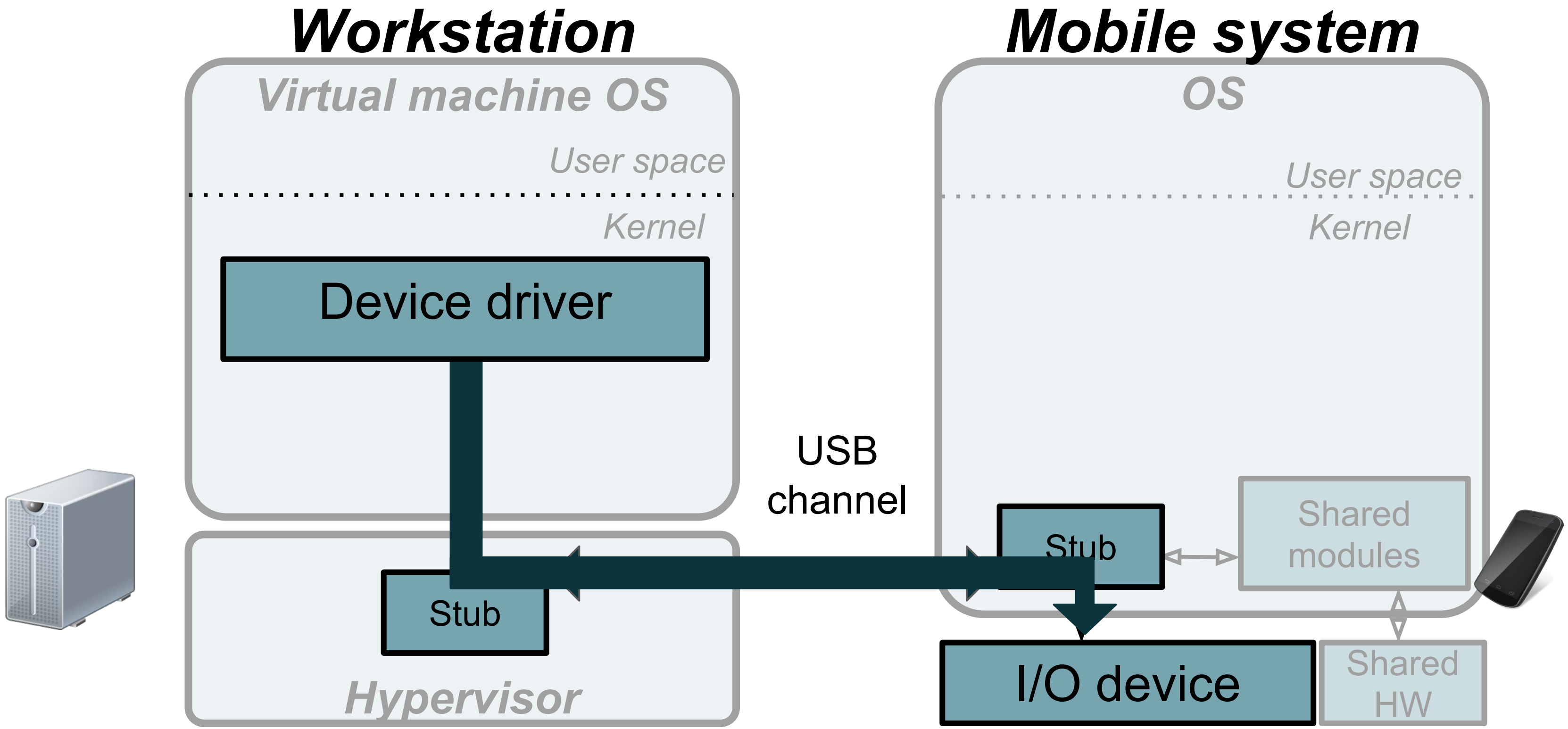
Low latency USB channel



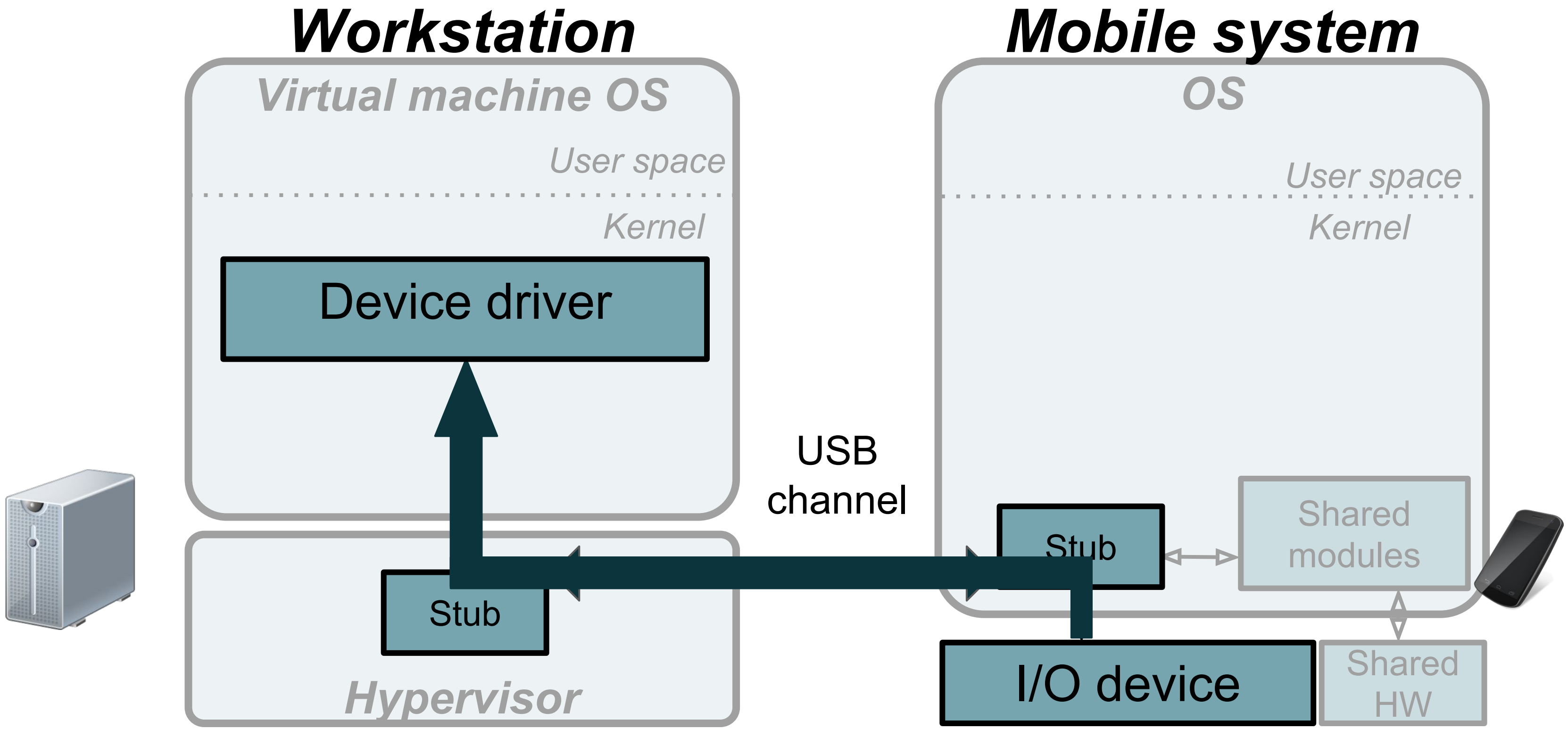
Design decision 2: low latency USB channel



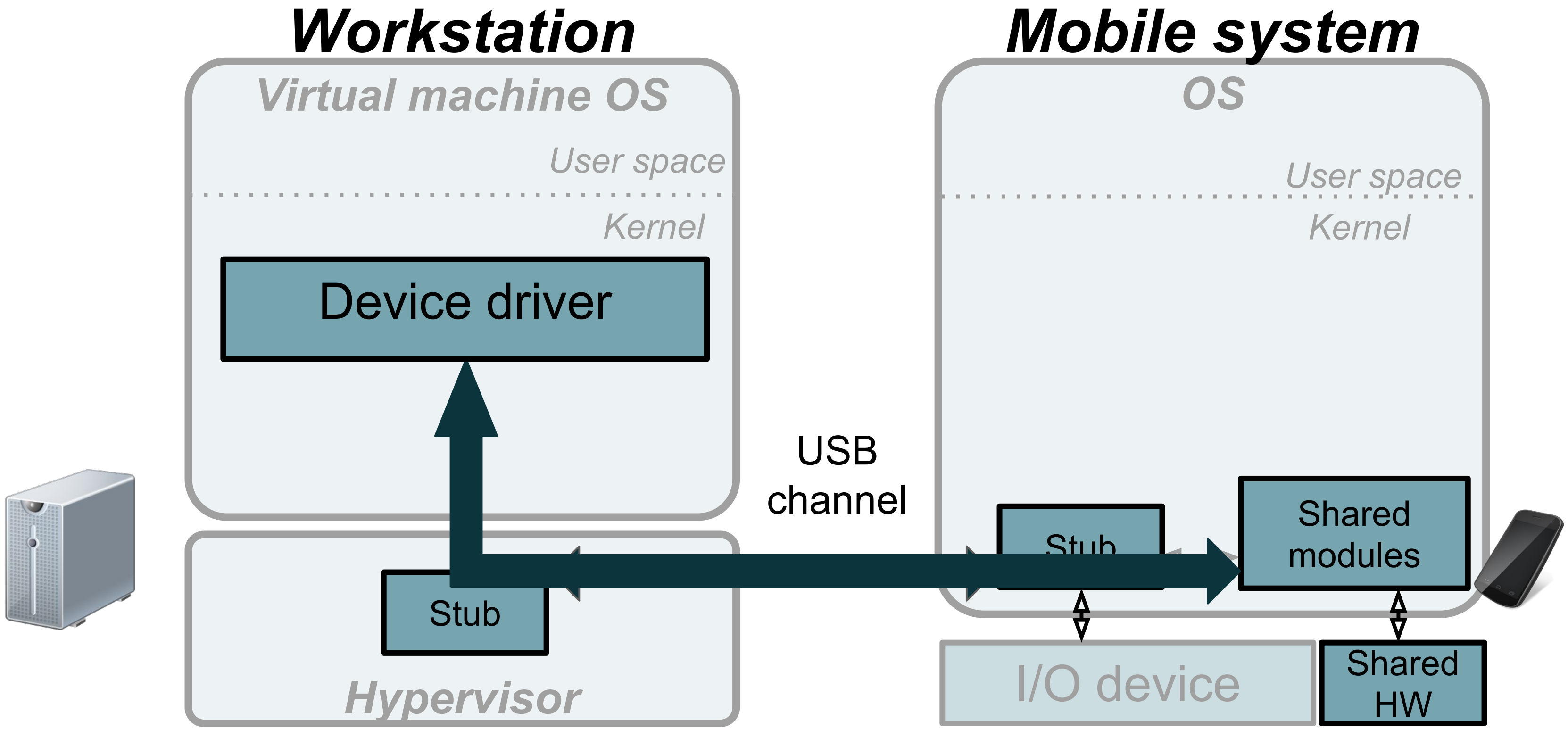
Remote I/O interface 1: remote register read/write



Remote I/O interface 2: remote interrupt handling



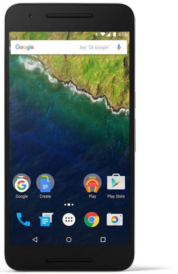
Remote I/O interface 3: Remote Procedure Call (RPC)



Evaluation

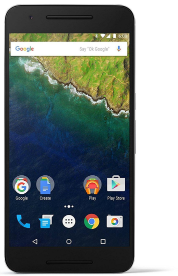
Summary

Charm supports various drivers and devices



Model	Nexus 5X	Nexus 6P	Galaxy S7
Manufacturer	LG	Huawei	Samsung
Supported drivers	Camera, Audio	GPU	IMU Sensors
Lines of Code Ported	65,000 + 30,000	31,000	3000
Porting time	-	7 days	2 days

Time it takes to port a driver to Charm



Model	Nexus 5X	Nexus 6P	Galaxy S7
Manufacturer	LG	Huawei	Samsung
Supported drivers	Camera, Audio	GPU	IMU Sensors
Lines of Code Ported	65,000 + 30,000	31,000	3000
Porting time	-	7 days	2 days

Charm supports various dynamic analysis techniques

- Fuzzing
- Record-and-replay
- Manual Interactive debugging

How Charm facilitates fuzzing

VT-x
PT

More hardware support

How Charm facilitates fuzzing

VT-x
PT

More hardware support

KASAN
KMSAN
KTSAN

More software support

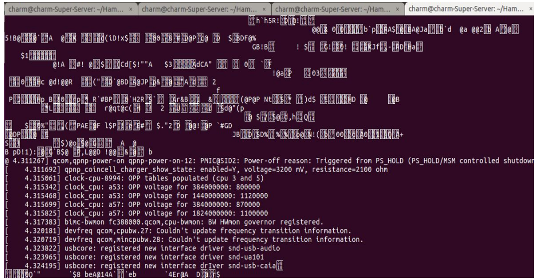
How Charm facilitates fuzzing

VT-x
PT

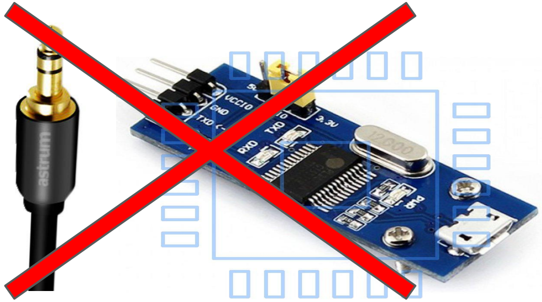
More hardware support

KASAN
KMSAN
KTSAN

More software support



Reliable console access



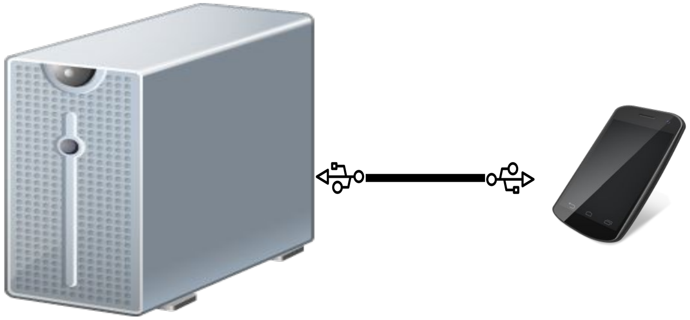
No special hardware

Fuzzing scenarios

Scenario 1
Without Charm

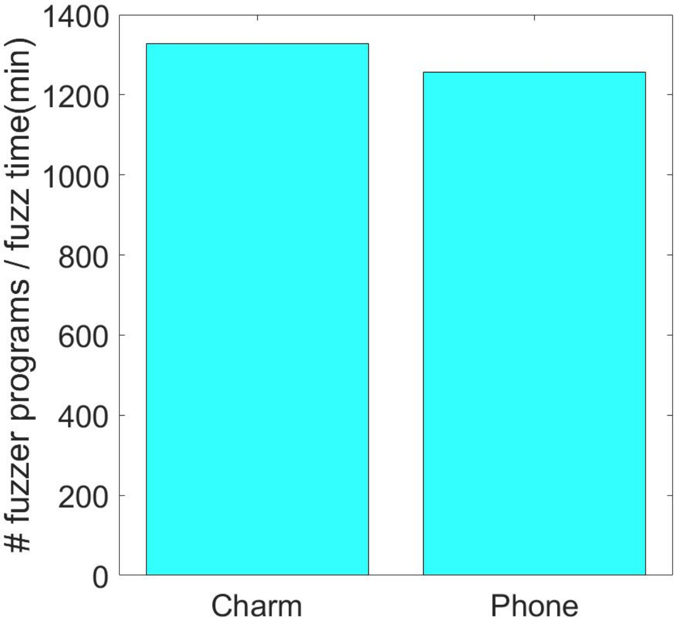
Execute fuzzer on the phone

Scenario 2
With Charm

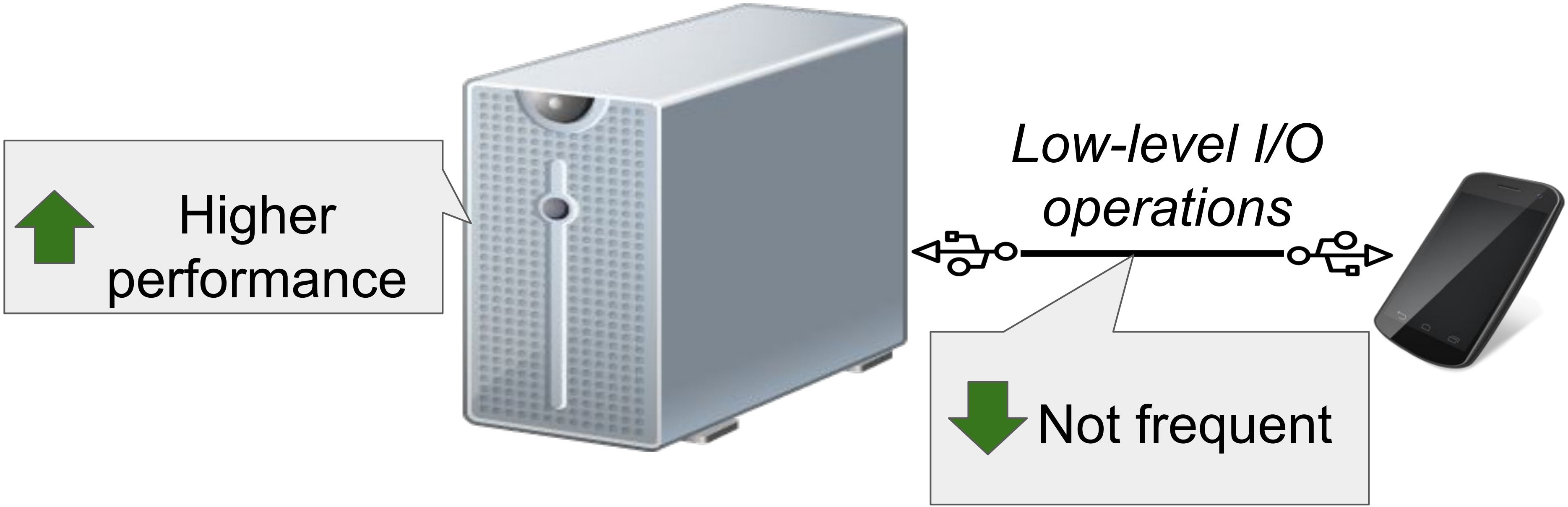


Execute fuzzer on the server

Fuzzing performance on Charm



Low overhead for fuzzing on Charm



Bugs found by Charm

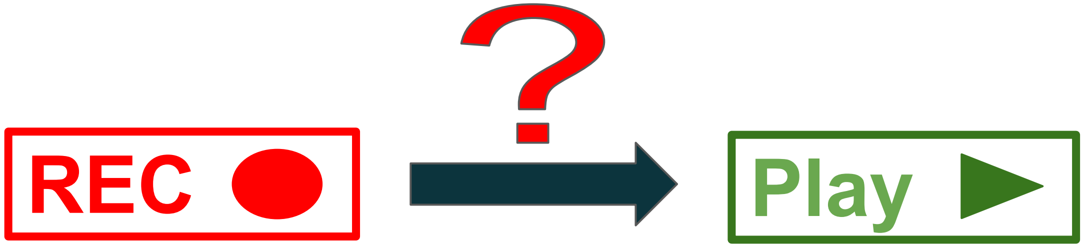
Total number of bugs	25
New bugs	14
Bugs found using KASAN	2
False positive bugs	**0**

Charm supports various dynamic analysis techniques

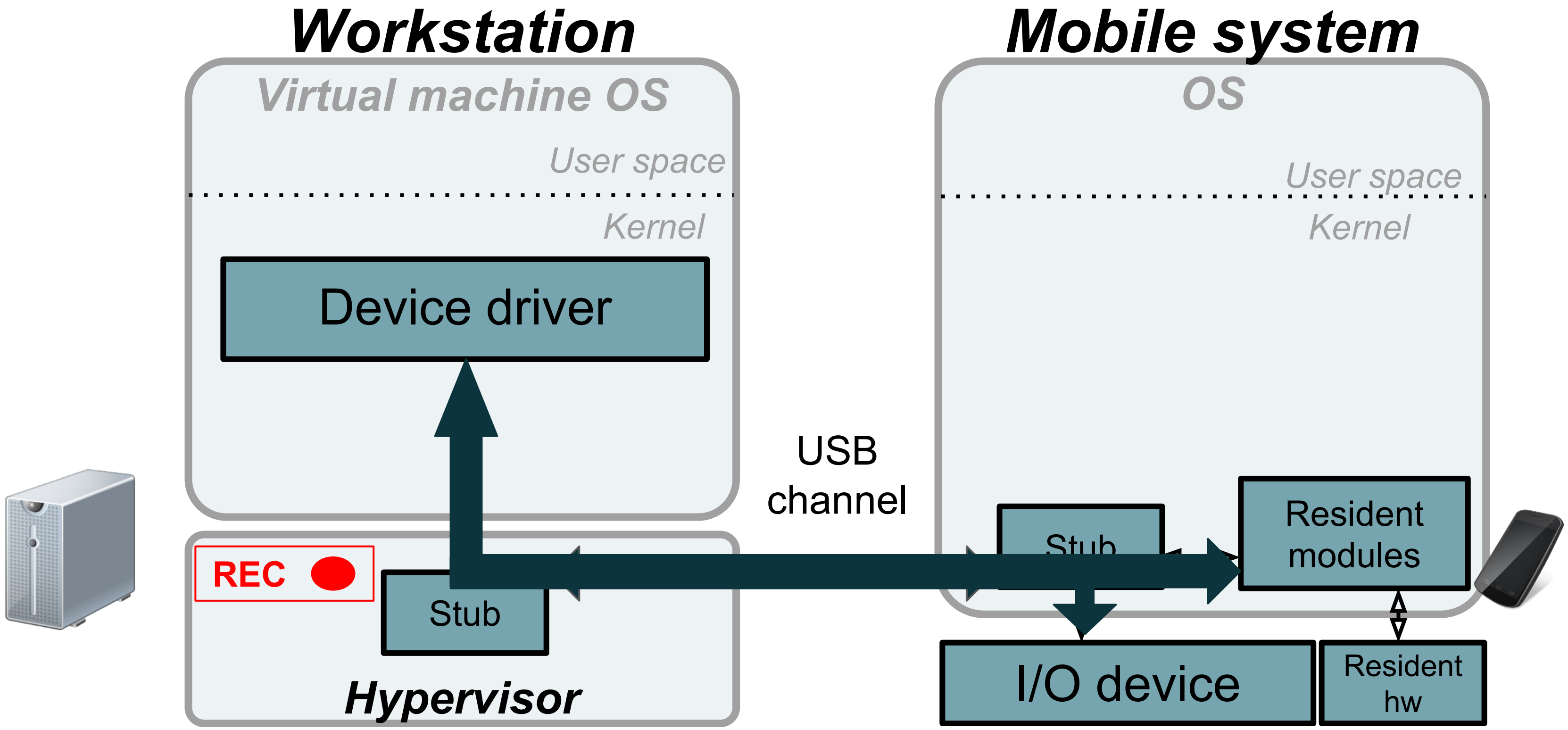
- Fuzzing
- Record-and-replay
- Manual Interactive debugging

Charm facilitates record-and-replay

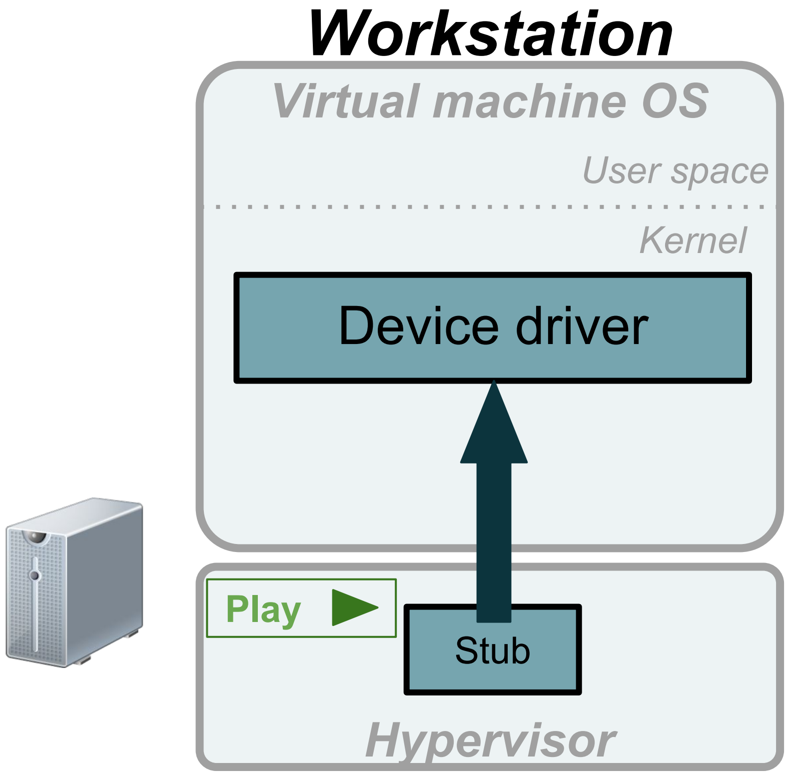
- Not feasible without Charm for mobile device drivers



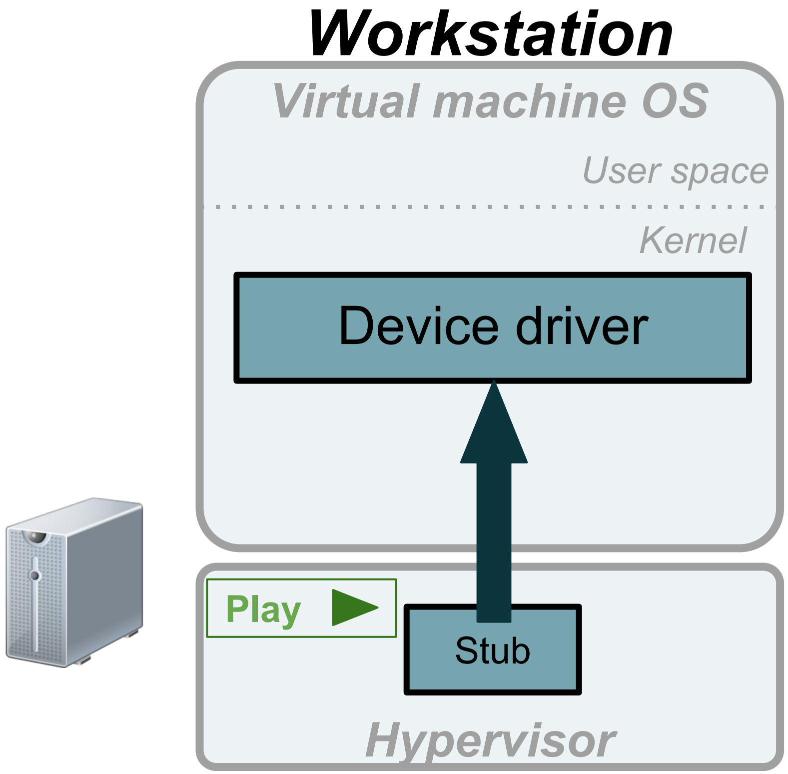
Record all remote I/O interactions



Replay the recorded interactions

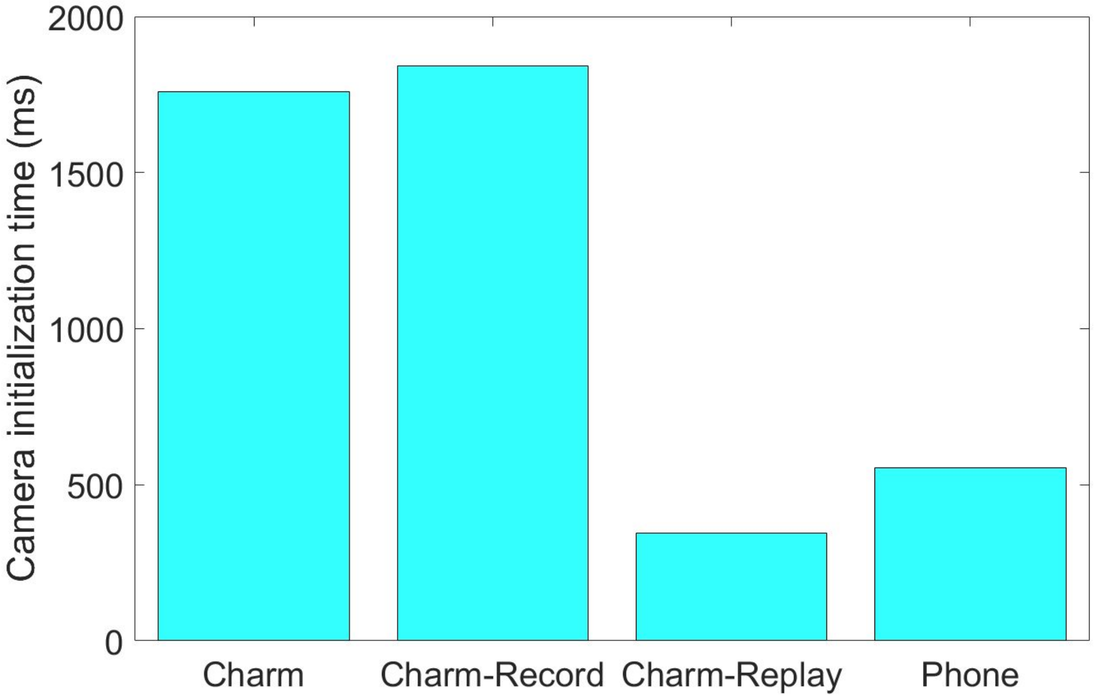


Replay the recorded interactions



Mobile system <u>is not needed</u> while replaying

Record-and-replay performance



Charm supports various dynamic analysis techniques

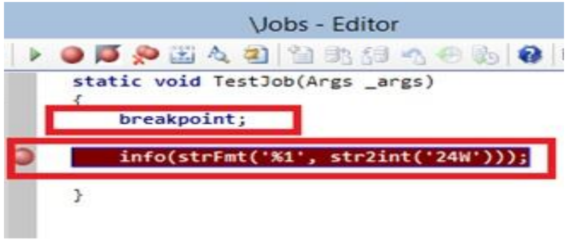
- Fuzzing
- Record and Replay
- Manual Interactive debugging

GDB

The GNU Project Debugger

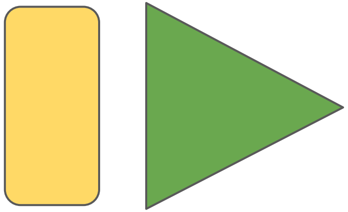
Charm facilitates manual interactive debugging

- Charm enables using GDB for device drivers



Breakpoint

Watchpoint



Single-step execution

Manual interactive debugging results

- We analyzed three known vulnerabilities
    - CVE-2016-3903: use-after-free bug
    - CVE-2016-2501: out-of-bounds access bug
    - CVE-2016-2061: out-of-bounds access bug
- We built an arbitrary kernel code execution exploit using CVE-2016-2061

Related work

	Charm	**Avatar [NDSS'14]**	**Surrogate [WOOT'15]**
Target	Mobile systems, open source device drivers	Embedded systems firmware	Embedded systems firmware
Forward I/O accesses	Yes	Yes	Yes
Communication channel	USB	UART and JTAG	PCIe FPGA board/JTAG
Performance	Near native	Poor	Near native

Limitations and Future work

Current Implementation	Future work
Manual port of drivers	Automatic port of drivers
No DMA support	DMA support
Open source drivers support	Binary drivers support

Summary

Summary

- Charm facilitates dynamic analysis of mobile device drivers
- Charm's performance is on par with actual mobile systems
- Charm supports a broad variety of device drivers with reasonable engineering effort

Summary

- Charm facilitates dynamic analysis of mobile device drivers
- Charm’s performance is on par with actual mobile systems
- Charm supports a broad variety of device drivers with reasonable engineering effort

Charm is open source:
http://trusslab.github.io/charm

Backup slides: vulnerable code snippet of CVE-2016-2061

```
int i = stream_cfg_cmd->stream_src;
if (i >= VFE_AXI_SRC_MAX) {
  ...
  return -EINVAL;
}
...
memset(&axi_data->stream_info[i], 0, sizeof(struct
   msm_vfe_axi_stream));
...
axi_data->stream_info[i].session_id =
   stream_cfg_cmd->session_id;
axi_data->stream_info[i].stream_id =
   stream_cfg_cmd->stream_id;
```

Backup slides: vulnerable code snippet of CVE-2016-2061

```
int i = stream_cfg_cmd->stream_src;
if (i >= VFE_AXI_SRC_MAX) {
  ...
  return -EINVAL;
}
...
memset(&axi_data->stream_info[i], 0, sizeof(struct
    msm_vfe_axi_stream));
...
axi_data->stream_info[i].session_id =
    stream_cfg_cmd->session_id;
axi_data->stream_info[i].stream_id =
    stream_cfg_cmd->stream_id;
```

Backup slides: building exploit

```
int i = stream_cfg_cmd->stream_src;
if (i >= VFE_AXI_SRC_MAX) {
    ...
  return -EINVAL;
}
...
memset(&axi_data->stream_info[i], 0, sizeof(struct
    msm_vfe_axi_stream));
...
axi_data->stream_info[i].session_id =
    stream_cfg_cmd->session_id;
axi_data->stream_info[i].stream_id =
    stream_cfg_cmd->stream_id;
```

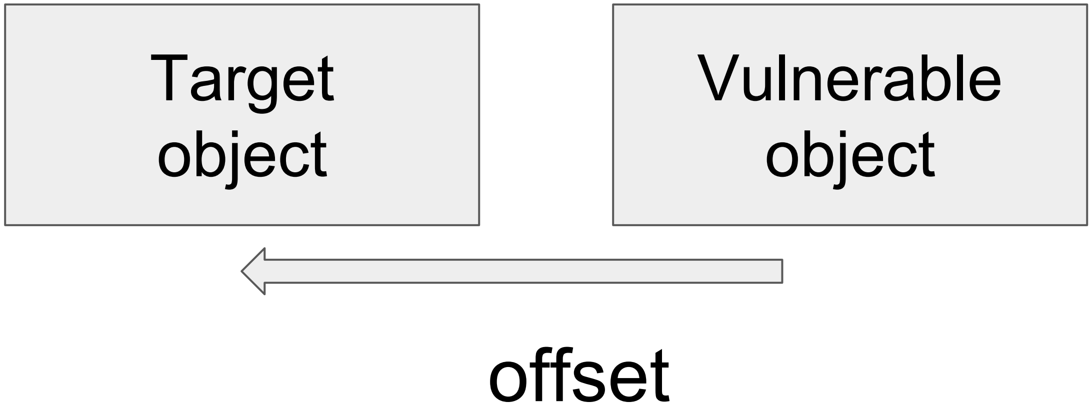
Heap or stack?

Backup slides: building exploit

```
int i = stream_cfg_cmd->stream_src;
if (i >= VFE_AXI_SRC_MAX) {
  ...
  return -EINVAL;
}
...
memset(&axi_data->stream_info[i], 0, sizeof(struct
    msm_vfe_axi_stream));
...
axi_data->stream_info[i].session_id =
    stream_cfg_cmd->session_id;
axi_data->stream_info[i].stream_id =
    stream_cfg_cmd->stream_id;
```

Heap or stack? Heap ->
Spray target objects

Backup slides: building exploit



Dynamic analysis is very useful

	Static analysis	Dynamic analysis
False positives rate	High	Low
Compiler/linker bugs	Cannot find	Can find
Code obfuscation	Vulnerable	Not vulnerable
Unknown types of bugs	Cannot find	Can find
Code coverage	High	Low

CVE-2016-3903

```
/* in msm_csid_cmd(): */
for (i = 0; i < csid_params.lut_params.num_cid; i++) {
  ...
  if (copy_from_user(vc_cfg, (void *)
   csid_params.lut_params.vc_cfg[i], sizeof(struct
   msm_camera_csid_vc_cfg))) {
    ...
    for (i--; i >= 0; i--)
      kfree(csid_params.lut_params.vc_cfg[i]);
    rc = -EFAULT;
    break;
  }
  csid_params.lut_params.vc_cfg[i] = vc_cfg;
}
...
rc = msm_csid_config(csid_dev, &csid_params);
```

```
/* in msm_csid_cid_lut(): */
...
if (csid_lut_params->vc_cfg[i]->cid >=
  csid_lut_params->num_cid ||
  csid_lut_params->vc_cfg[i]->cid < 0) {
  ...
}
```

Is it out-of-bound access?

CVE-2016-3903

```
/* in msm_csid_cmd(): */
for (i = 0; i < csid_params.lut_params.num_cid; i++) {
  ...
  if (copy_from_user(vc_cfg, (void *)
   csid_params.lut_params.vc_cfg[i], sizeof(struct
   msm_camera_csid_vc_cfg))) {
    ...
    for (i--; i >= 0; i--)
      kfree(csid_params.lut_params.vc_cfg[i]);
    rc = -EFAULT;
    break;
  }
  csid_params.lut_params.vc_cfg[i] = vc_cfg;
}
...
rc = msm_csid_config(csid_dev, &csid_params);
```

```
/* in msm_csid_cid_lut(): */
...
if (csid_lut_params->vc_cfg[i]->cid >=
  csid_lut_params->num_cid ||
  csid_lut_params->vc_cfg[i]->cid < 0) {
  ...
}
```

Is it out-of-bound access?

CVE-2016-3903

```
/* in msm_csid_cmd(): */
for (i = 0; i < csid_params.lut_params.num_cid; i++) {
  ...
  if (copy_from_user(vc_cfg, (void *)
   csid_params.lut_params.vc_cfg[i], sizeof(struct
   msm_camera_csid_vc_cfg))) {
    ...
    for (i--; i >= 0; i--)
      kfree(csid_params.lut_params.vc_cfg[i]);
    rc = -EFAULT;
    break;
  }
  csid_params.lut_params.vc_cfg[i] = vc_cfg;
}
...
rc = msm_csid_config(csid_dev, &csid_params);
```

Use after free

```
/* in msm_csid_cid_lut(): */
...
if (csid_lut_params->vc_cfg[i]->cid >=
  csid_lut_params->num_cid ||
  csid_lut_params->vc_cfg[i]->cid < 0) {
  ...
}
```

Watch points