Unleashing D* on Android Kernel Drivers

Aravind Machiry (@machiry_msidc)

$ whoami

- Fourth year P.h.D Student at University of California, Santa Barbara.
- Vulnerability Detection in System software.
- machiry.github.io

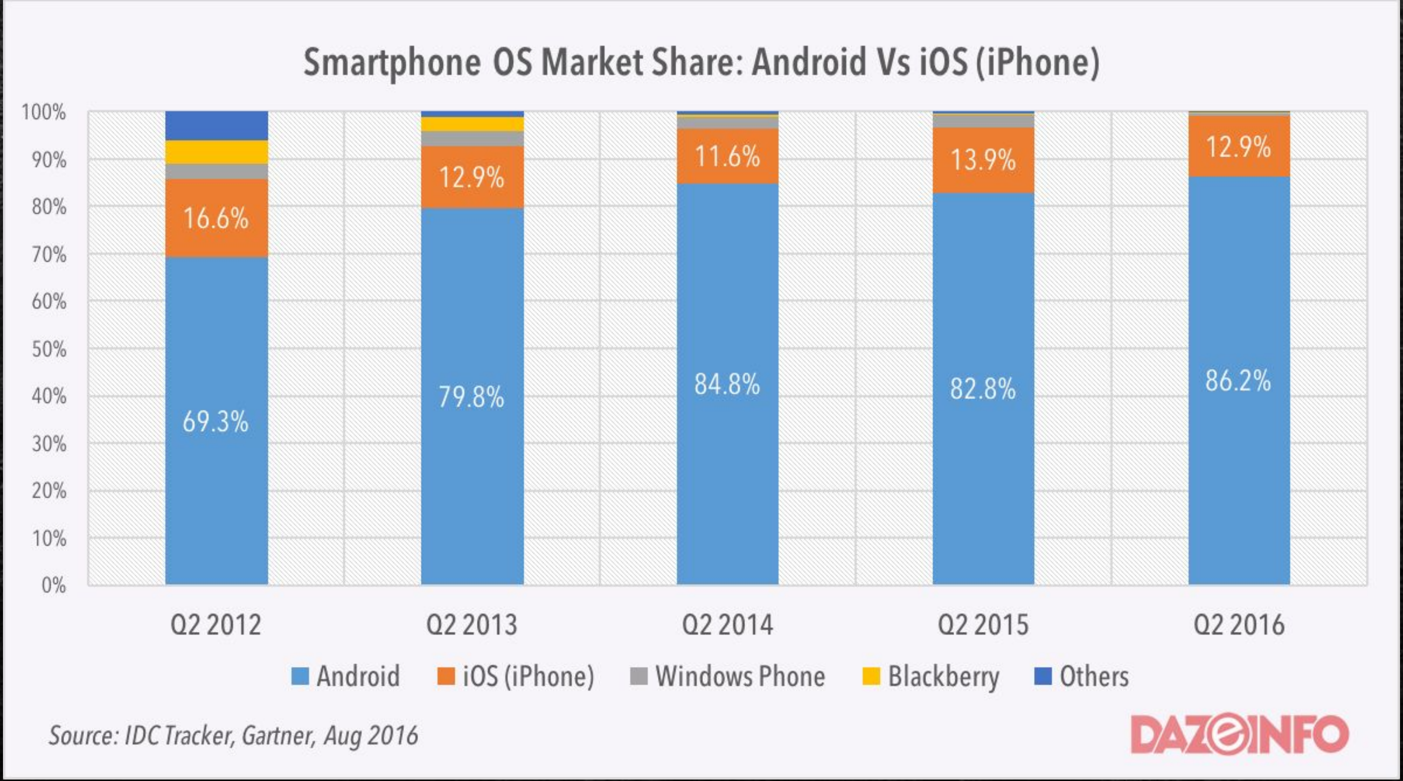
TrustZone®

System Security by ARM

BOOTLOADER

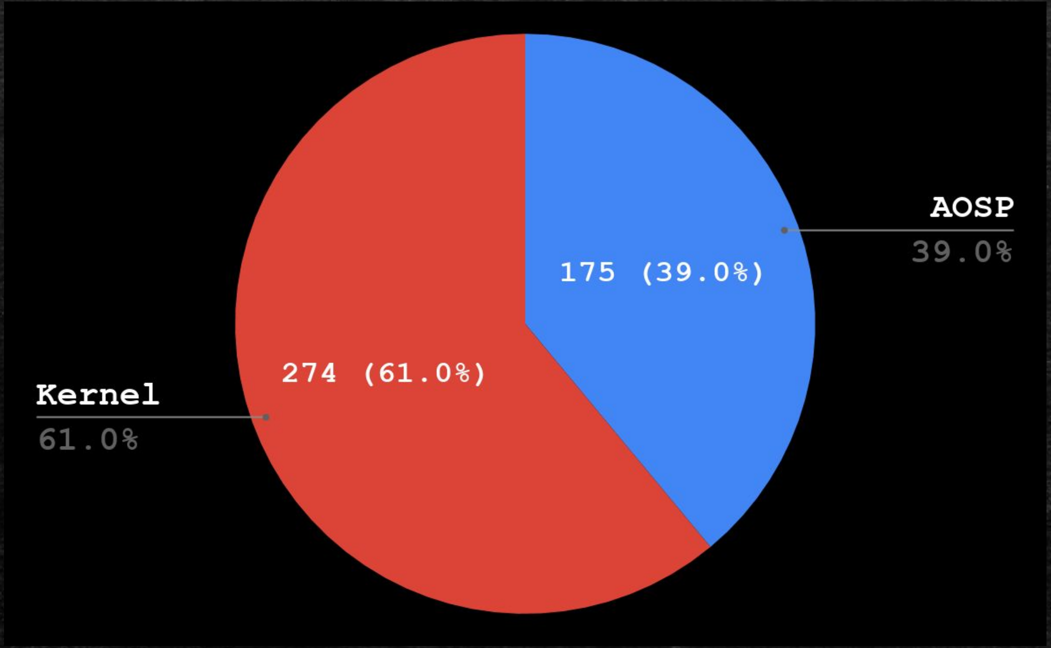
Firmware

$ Android is everywhere!!



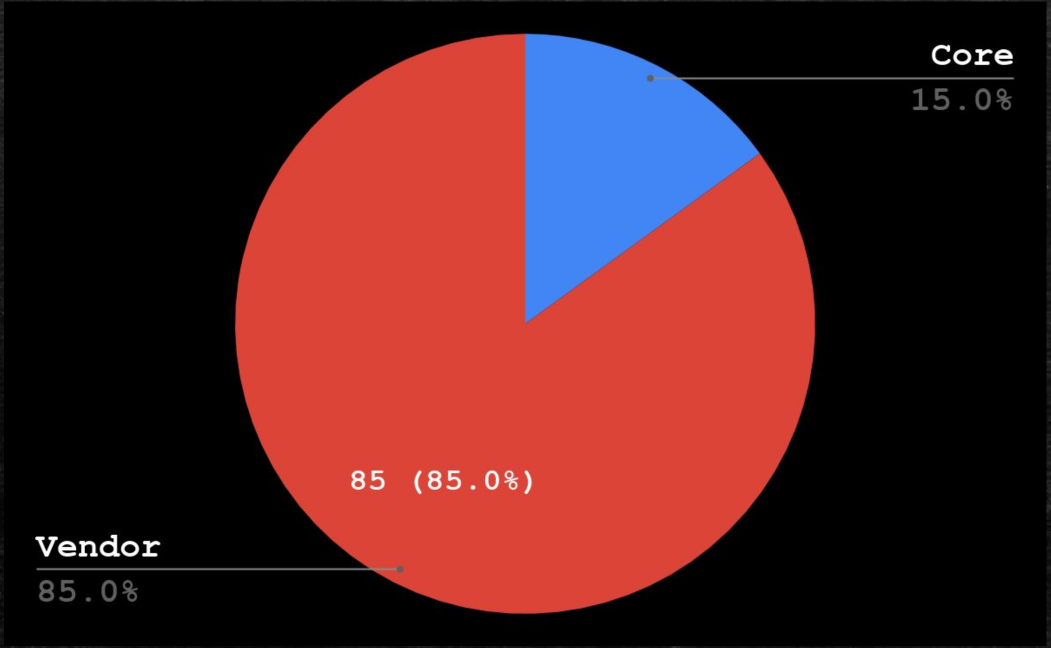
Source: IDC Tracker, Gartner, Aug 2016

$ Bugs in Android (First Half of 2017)[1]



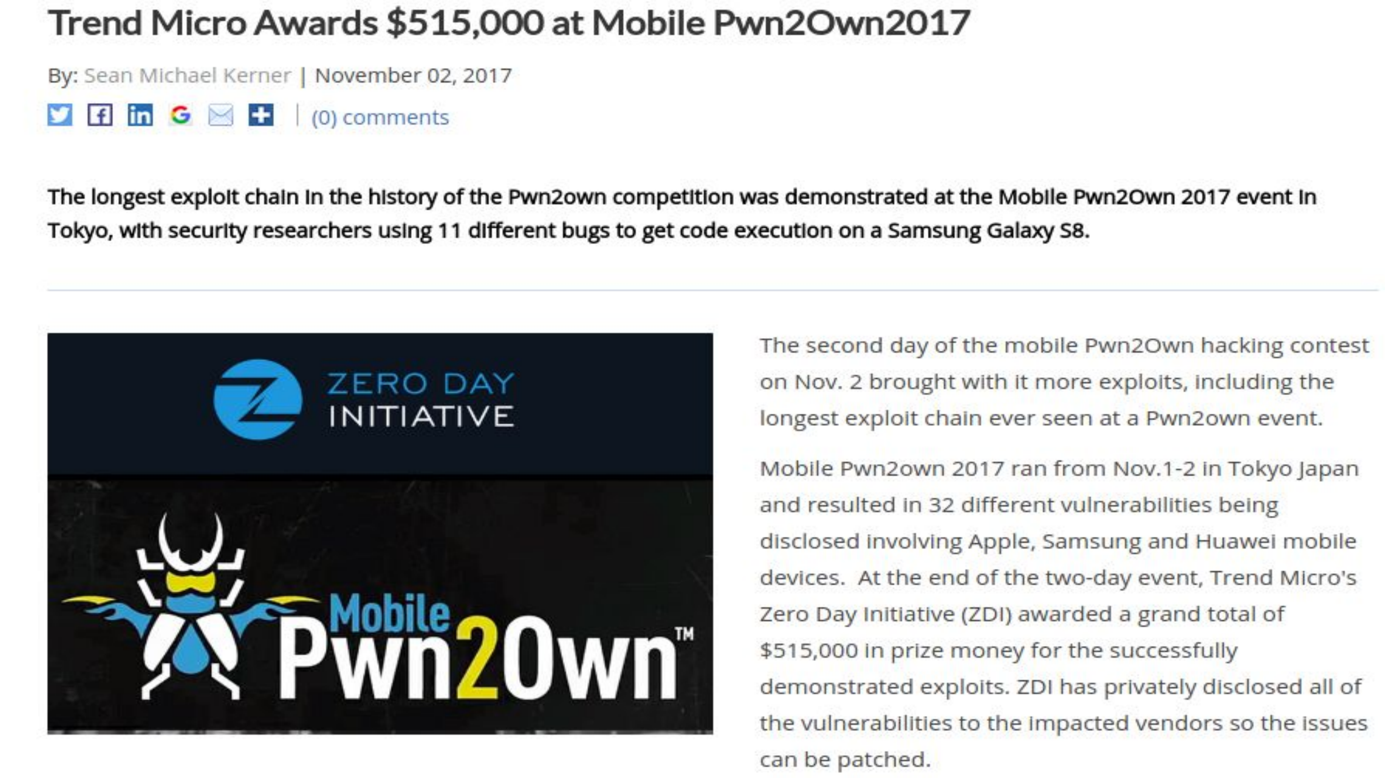
[1]https://www.slideshare.net/JunLI174/android-bug-hunting-from-finding-bugs-to-getting-bounty-20170603

$ Where are Android kernel bugs?



android: Protecting the kernel, Jeff Vander Stoep, Linux Foundation 2016

$ Lot of $$$



Trend Micro Awards $515,000 at Mobile Pwn2Own2017

By: Sean Michael Kerner | November 02, 2017

(0) comments

The longest exploit chain in the history of the Pwn2own competition was demonstrated at the Mobile Pwn2Own 2017 event in Tokyo, with security researchers using 11 different bugs to get code execution on a Samsung Galaxy S8.

The second day of the mobile Pwn2Own hacking contest on Nov. 2 brought with it more exploits, including the longest exploit chain ever seen at a Pwn2own event.

Mobile Pwn2own 2017 ran from Nov.1-2 in Tokyo Japan and resulted in 32 different vulnerabilities being disclosed involving Apple, Samsung and Huawei mobile devices. At the end of the two-day event, Trend Micro's Zero Day Initiative (ZDI) awarded a grand total of $515,000 in prize money for the successfully demonstrated exploits. ZDI has privately disclosed all of the vulnerabilities to the impacted vendors so the issues can be patched.

ZERODIUM Payouts for Mobiles*

RJB: Remote Jailbreak with Persistence
RCE: Remote Code Execution
LPE: Local Privilege Escalation
SBX: Sandbox Escape or Bypass

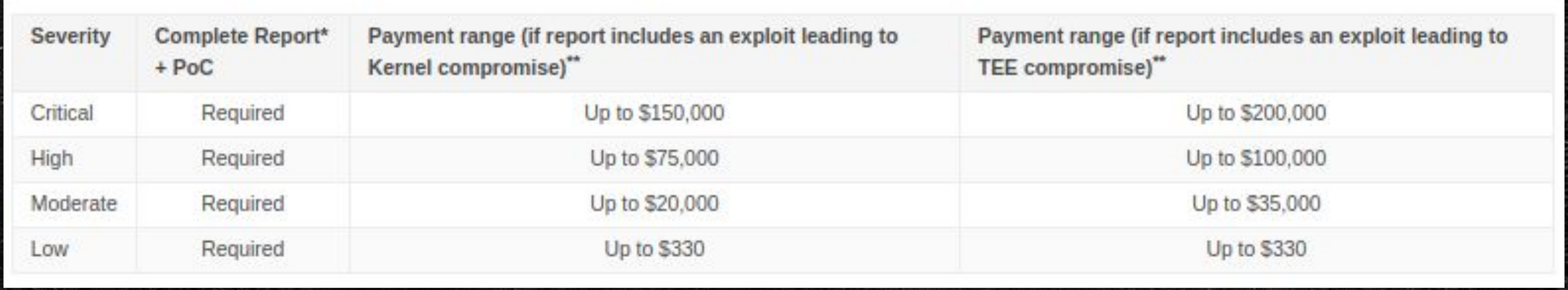
🟥 IOS
🟫 Android
🟦 Any OS

Up to $1,500,000									1.001 iPhone RJB Zero Click IOS
Up to $1,000,000									1.002 iPhone RJB IOS
Up to $500,000	2.001 WeChat RCE+LPE IOS/Android	2.002 Viber RCE+LPE IOS/Android		2.003 FB Messenger RCE+LPE IOS/Android	2.004 Signal RCE+LPE IOS/Android	2.005 Telegram RCE+LPE IOS/Android	2.006 WhatsApp RCE+LPE IOS/Android	2.007 iMessage RCE+LPE IOS	2.008 SMS/MMS RCE+LPE IOS/Android	2.009 Email App RCE+LPE IOS/Android
Up to $150,000	3.001 Baseband RCE+LPE IOS/Android						2.010 Media Files RCE+LPE IOS/Android	2.011 Documents RCE+LPE IOS/Android	4.001 Chrome RCE+LPE IOS/Android	4.002 Safari RCE+LPE IOS
Up to $100,000	5.001 Code Signing Bypass IOS	3.002 WiFi RCE+LPE IOS/Android	3.003 SS7					6.001 LPE to Kernel IOS/Android	4.003 SBX for Chrome Android	4.004 SBX for Safari IOS
Up to $50,000	5.002 Code Signing Bypass Android	5.003 Secure Boot IOS	3.004 RCE via MitM IOS/Android			6.002 LPE to Root IOS/Android	4.005 Chrome RCE w/o SBX IOS/Android	4.006 Chrome UXSS/SOP IOS/Android	4.007 Safari UXSS/SOP IOS	4.008 Safari RCE w/o SBX IOS
Up to $25,000	5.004 TrustZone Android	5.005 Verified Boot Android			6.003 LPE to System Android	7.001 ASLR Bypass IOS/Android	7.002 kASLR Bypass IOS/Android	7.003 Seccomp Bypass Android	7.004 RKP Bypass Android	7.005 Knox Bypass Android
Up to $15,000	9.001 Information Disclosure IOS/Android							8.001 Passcode Bypass IOS	8.002 Touch ID Bypass IOS	8.003 PIN Bypass Android

* All payouts are subject to change or cancellation without notice, at the discretion of ZERODIUM. All trademarks are the property of their respective owners.

2017/08 © zerodium.com

$ Lot of $$$



Severity	Complete Report* + PoC	Payment range (if report includes an exploit leading to Kernel compromise)**	Payment range (if report includes an exploit leading to TEE compromise)**
Critical	Required	Up to $150,000	Up to $200,000
High	Required	Up to $75,000	Up to $100,000
Moderate	Required	Up to $20,000	Up to $35,000
Low	Required	Up to $330	Up to $330

$ Okay! Why is it hard to find these bugs?

Good ‘ol knowledge on Bug Detection:

- Static Analysis
- Dynamic Analysis

$ Okay! Why is it hard to find these bugs?

Good ‘ol knowledge on Bug Detection:

- **Static Analysis**
- Dynamic Analysis

$ Static Analysis

- Kernel drivers are open source (GPL).
- Use well known source code analysis tools.
- $$$

$ Static Analysis: Existing tools

	CppCheck	flawfinder	RATS	Sparse
Qualcomm	18	4,365	693	5,202
Samsung	22	8,173	2,244	1,726
Huawei	34	18,132	2,301	11,230
MediaTek	168	14,230	3,730	13,771
Total	**242**	**44,990**	**8,968**	**31,929**

$ Static Analysis: Existing tools

	CppCheck	flawfinder	RATS	Sparse
Qualcomm	18			
Samsung	22			
Huawei	34			
MediaTek	168			
Total	**242**			

$ Ideal Static analysis tool

- Track user data.
- Check if user data is used in sensitive places.
  - Example: memcpy(src, dst, <user_data>);

$ Tracking user data: Pointer analysis

```
struct foo obj;

scanf(“%d”, &idx);

if(*) {

          obj.input_var = &idx;

} else {

          obj.input_var = *;

}

....

bar(&obj);
```

$ Tracking user data: Pointer analysis

```
struct foo obj;

scanf(“%d”, &idx);

if(*) {

          obj.input_var = &idx;

} else {

          obj.input_var = *;

}

....

bar(&obj);
```

$ Tracking user data: Pointer analysis

```
struct foo obj;
scanf(“%d”, &idx);
if(*) {
        obj.input_var = &idx;
} else {
        obj.input_var = *;
}
....
bar(&obj);
```

$ Tracking user data: Pointer analysis

```
struct foo obj;
scanf(“%d”, &idx);
if(*) {
          obj.input_var = &idx;
} else {
          obj.input_var = *;
}
....
bar(&obj);
```

$ Tracking user data: Pointer analysis

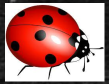
```
struct foo obj;

scanf("%d", &idx);

if(*) {

          obj.input_var = &idx;

} else {

          obj.input_var = *;

}

....

bar(&obj);
```

```
void bar(struct foo *obj_ptr) {

            …

     arr[*(obj_ptr->safe_input)] = 1;

     …

     arr[*(obj_ptr->input_var)] = 0;

      …

}
```

$ Tracking user data: Pointer analysis

```
struct foo obj;

scanf("%d", &idx);

if(*) {

        obj.input_var = &idx;

} else {

        obj.input_var = *;

}

....

bar(&obj);
```

```
void bar(struct foo *obj_ptr) {

            …

    arr[*(obj_ptr->safe_input)] = 1;

    …

    arr[*(obj_ptr->input_var)] = 0;

     …

}
```



$ Tracking user data: Pointer analysis

We should know which pointer points to what object.

```
void bar(struct foo *obj_ptr) {
    …
}
```

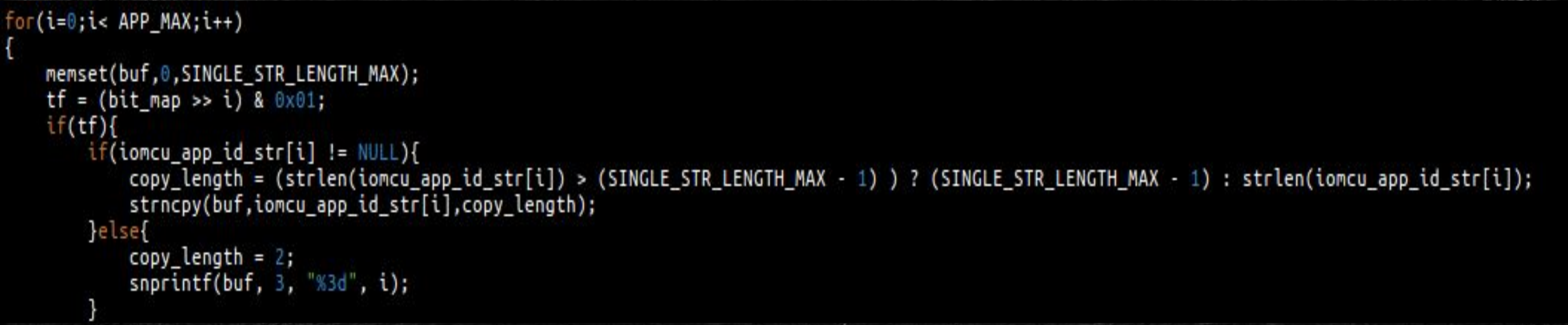
$ Tracking user data: Pointer analysis

We should know which field of a structure object points to what object.

```
void bar(struct foo *obj_ptr) {
          …
    arr[*(obj_ptr->safe_input)] = 1;
    …
    arr[*(obj_ptr->input_var)] = 0;
     …
}
```

$ Tracking user data: Pointer analysis

Real world code is complex: Loops, Type casting



```
for(i=0;i< APP_MAX;i++)
{
    memset(buf,0,SINGLE_STR_LENGTH_MAX);
    tf = (bit_map >> i) & 0x01;
    if(tf){
        if(iomcu_app_id_str[i] != NULL){
            copy_length = (strlen(iomcu_app_id_str[i]) > (SINGLE_STR_LENGTH_MAX - 1) ) ? (SINGLE_STR_LENGTH_MAX - 1) : strlen(iomcu_app_id_str[i]);
            strncpy(buf,iomcu_app_id_str[i],copy_length);
        }else{
            copy_length = 2;
            snprintf(buf, 3, "%3d", i);
        }
```

$ Tracking user data: Taint Propagation

We should **follow** the **flow** of user data.

```
scanf(“%d”, &in);
…
su = in + 5;
…
arr[su] = 0;
```

$ Tracking user data: Taint Propagation

What about library functions?

```
scanf(“%10s”, inputstr);
…
usrint = atoi(inputstr);
…
arr[usrint] = 0;
```

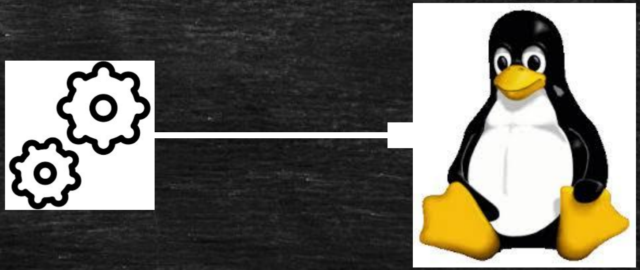
$ Tracking user data: Taint Propagation

What about library functions: Simple policy doesn’t work

```
scanf(“%10s”, inputstr);
…
usrint = atoi(inputstr);
…
ptr = malloc(usrint);
…
ptr[0] = 0;
```

$ Kernel drivers are small!!

- ~ 31 to 240K SLOC
- 80% of drivers <= 7K SLOC



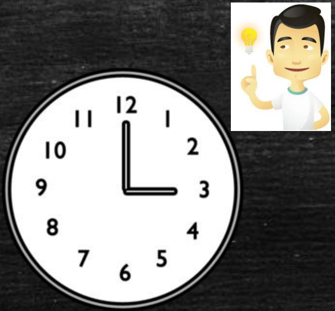
Let’s separate driver code from kernel code!!!

$ Optimizations: **Soundy Traversal**

- ❖ Let's analyze **code inside loops fixed number of times**.
- ❖ Assume that **all kernel functions are safe**.

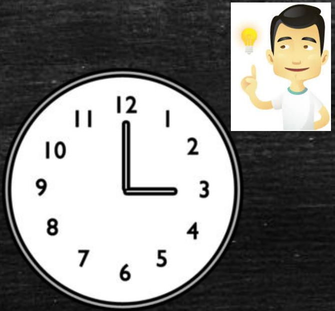
$ DR.CHECKER: Story of the name

$ DR.CHECKER: Story of the name



```
$ mkdir dr_checker
```

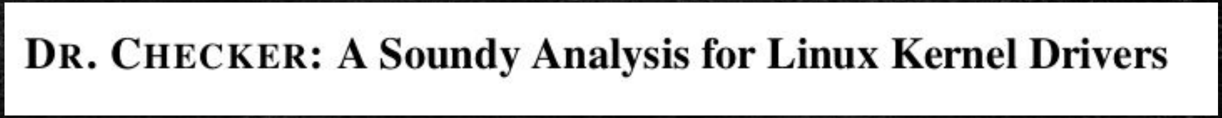
$ DR.CHECKER: Story of the name



```
$ mkdir dr_checker
```

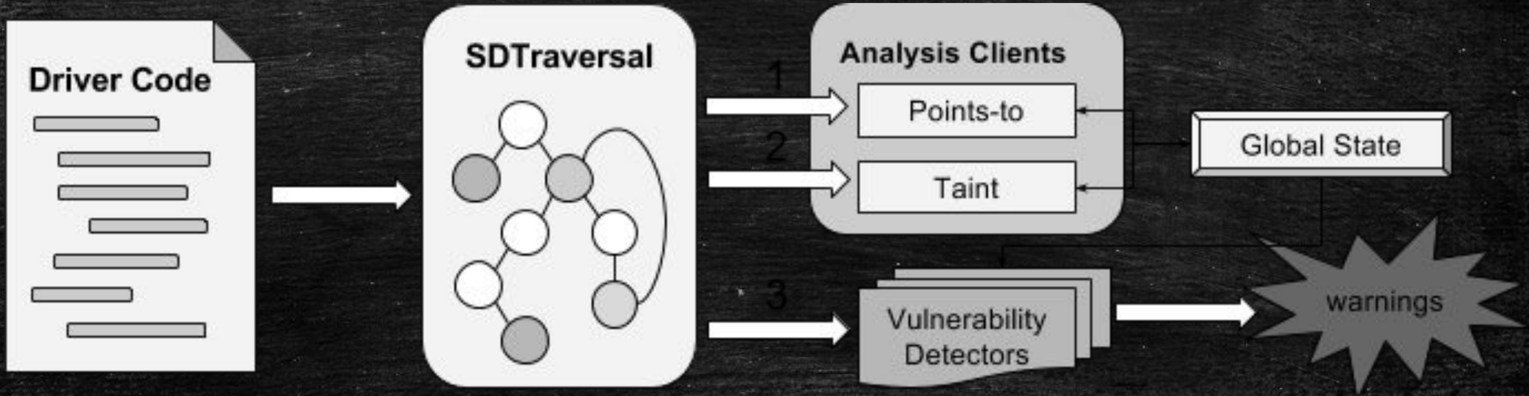
$ DR.CHECKER: Story of the name

```
$ mkdir dr_checker
```



DR. CHECKER: A Soundy Analysis for Linux Kernel Drivers

$ DR.CHECKER Overview



$ DR.CHECKER: SDTraversal

Starting from entry point of the driver:

- Analyze each instruction of the driver according to the control flow.

At each instruction:

- Maintain and track the user data.
- Consult **Vulnerability or bug detectors** for any potential bugs.

$ DR.CHECKER: Vulnerability Detectors

- Tainted Pointer Dereference.
- Tainted Integer Arithmetic.
- Tainted Size.
- Uninit Leak Detector.
- Global Variable Race Detector.
- Improper Tainted Data Use Detector.
- ….

$ DR.CHECKER: Bug in Mediatek Accdet driver

```
static char call_status;

...

static ssize_t accdet_store_call_state (struct device_driver *ddri,  const char *buf,
 size_t count)

{

    int ret = sscanf(buf, "%s", &call_status);

    if (ret != 1) {

         ACCDET_DEBUG("accdet: Invalid values\n");

         return -EINVAL;

}
```

$ DR.CHECKER: Bug in Mediatek Accdet driver

```
static char call_status;

...

static ssize_t accdet_store_call_state (struct device_driver *ddri,  const char *buf,
 size_t count)

{

    int ret = sscanf(buf, "%s", &call_status);

    if (ret != 1) {

        ACCDET_DEBUG("accdet: Invalid values\n");

        return -EINVAL;

}
```

$ DR.CHECKER: Bug in Mediatek Accdet driver

```
static char call_status;

...

static ssize_t accdet_store_call_state (struct device_driver *ddri,  const char *buf,
 size_t count)

{

    int ret = sscanf(buf, "%s", &call_status);

    if (ret != 1) { // TOO LATE..Buffer overflow already happened.

        ACCDET_DEBUG("accdet: Invalid values\n");

        return -EINVAL;

}
```

$ DR.CHECKER: Bug in Samsung SensorHub driver

```
if (unlikely(count < 2)) {
      ...
      return -EINVAL;
}
...
buffer = kzalloc(count * sizeof(char), GFP_KERNEL);
...
ret = copy_from_user(buffer, buf, count);
…
if (buffer[0] == MSG2SSP_INST_LIB_NOTI) {
  ret = ssp_sensorhub_send_cmd(hub_data, buffer,
count);
}
```

```
 ssp_sensorhub_send_cmd(...,
const char *buf, int count) {


If (buf[2] < MSG2SSP_AP_STATUS_WAKEUP
||
     buf[2] >=
MSG2SSP_AP_TEMPHUMIDITY_CAL_DONE) {
     ...
}

}
```

$ DR.CHECKER: Bug in Samsung SensorHub driver

```
if (unlikely(count < 2)) { //consider when count==2
     ...
     return -EINVAL;
}
...
buffer = kzalloc(count * sizeof(char), GFP_KERNEL);
...
ret = copy_from_user(buffer, buf, count);
…
if (buffer[0] == MSG2SSP_INST_LIB_NOTI) {
  ret = ssp_sensorhub_send_cmd(hub_data, buffer,
count);
}
```

```
 ssp_sensorhub_send_cmd(...,
const char *buf, int count) {


If (buf[2] < MSG2SSP_AP_STATUS_WAKEUP
||
     buf[2] >=
MSG2SSP_AP_TEMPHUMIDITY_CAL_DONE) {
     ...
}

}
```

$ DR.CHECKER: Bug in Samsung SensorHub driver

```
if (unlikely(count < 2)) { //consider when count==2
     ...
     return -EINVAL;
}
...//buffer is of size 2: buffer[0], buffer[1]
buffer = kzalloc(count * sizeof(char), GFP_KERNEL);
...
ret = copy_from_user(buffer, buf, count);
…
if (buffer[0] == MSG2SSP_INST_LIB_NOTI) {
  ret = ssp_sensorhub_send_cmd(hub_data, buffer,
count);
}
```

```
 ssp_sensorhub_send_cmd(...,
const char *buf, int count) {


If (buf[2] < MSG2SSP_AP_STATUS_WAKEUP
||
     buf[2] >=
MSG2SSP_AP_TEMPHUMIDITY_CAL_DONE) {
     ...
}

}
```

$ DR.CHECKER: Bug in Samsung SensorHub driver

```
if (unlikely(count < 2)) { //consider when count==2
     ...
     return -EINVAL;
}
...//buffer is of size 2: buffer[0], buffer[1]
buffer = kzalloc(count * sizeof(char), GFP_KERNEL);
...
ret = copy_from_user(buffer, buf, count);
…
if (buffer[0] == MSG2SSP_INST_LIB_NOTI) {
  ret = ssp_sensorhub_send_cmd(hub_data, buffer,
count);
}
```

```
 ssp_sensorhub_send_cmd(...,
const char *buf, int count) {


If (buf[2] < MSG2SSP_AP_STATUS_WAKEUP
||
     buf[2] >=
MSG2SSP_AP_TEMPHUMIDITY_CAL_DONE) {
     ...
}

}
```

$ DR.CHECKER: Bug in Samsung SensorHub driver

```
if (unlikely(count < 2)) { //consider when count==2
     ...
     return -EINVAL;
}
...//buffer is of size 2: buffer[0], buffer[1]
buffer = kzalloc(count * sizeof(char), GFP_KERNEL);
...
ret = copy_from_user(buffer, buf, count);
…
if (buffer[0] == MSG2SSP_INST_LIB_NOTI) {
  ret = ssp_sensorhub_send_cmd(hub_data, buffer,
count);
}
```

```
 ssp_sensorhub_send_cmd(...,
const char *buf, int count) {


If (buf[2] < MSG2SSP_AP_STATUS_WAKEUP
||
     buf[2] >=
MSG2SSP_AP_TEMPHUMIDITY_CAL_DONE) {
     ...
}

}
```

$ DR.CHECKER: Bug in Samsung SensorHub driver

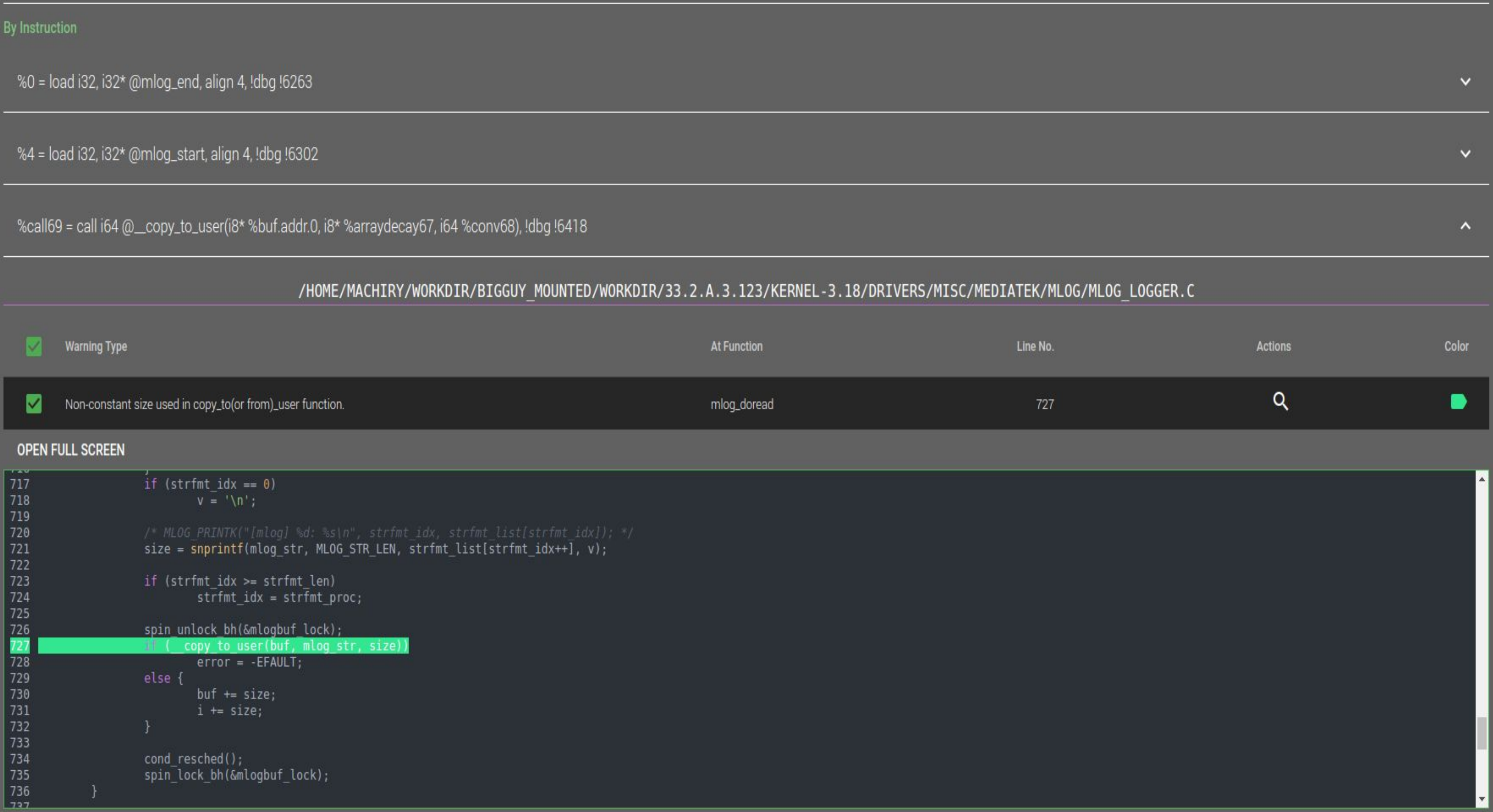
```
if (unlikely(count < 2)) { //consider when count==2
      ...
      return -EINVAL;
}
...//buffer is of size 2: buffer[0], buffer[1]
buffer = kzalloc(count * sizeof(char), GFP_KERNEL);
...
ret = copy_from_user(buffer, buf, count);
…
if (buffer[0] == MSG2SSP_INST_LIB_NOTI) {
  ret = ssp_sensorhub_send_cmd(hub_data, buffer,
count);
}
```

```
 ssp_sensorhub_send_cmd(...,
const char *buf, int count) {


If (buf[2] < MSG2SSP_AP_STATUS_WAKEUP
||
     buf[2] >=
MSG2SSP_AP_TEMPHUMIDITY_CAL_DONE) {
     ...
}

}
```

$ DR.CHECKER: Bug in Samsung SensorHub driver

```
if (unlikely(count < 2)) { //consider when count==2
     ...
     return -EINVAL;
}
...//buffer is of size 2: buffer[0], buffer[1]
buffer = kzalloc(count * sizeof(char), GFP_KERNEL);
...
ret = copy_from_user(buffer, buf, count);
…
if (buffer[0] == MSG2SSP_INST_LIB_NOTI) {
  ret = ssp_sensorhub_send_cmd(hub_data, buffer,
count);
}
```

```
 ssp_sensorhub_send_cmd(...,
const char *buf, int count) {


If (buf[2] <
MSG2SSP_AP_STATUS_WAKEUP ||
     buf[2] >=
MSG2SSP_AP_TEMPHUMIDITY_CAL_DONE) {
     ...
}

}
```



By Instruction

%0 = load i32, i32* @mlog_end, align 4, !dbg !6263

%4 = load i32, i32* @mlog_start, align 4, !dbg !6302

%call69 = call i64 @__copy_to_user(i8* %buf.addr.0, i8* %arraydecay67, i64 %conv68), !dbg !6418

/HOME/MACHIRY/WORKDIR/BIGGUY_MOUNTED/WORKDIR/33.2.A.3.123/KERNEL-3.18/DRIVERS/MISC/MEDIATEK/MLOG/MLOG_LOGGER.C

☑	Warning Type	At Function	Line No.	Actions	Color
☑	Non-constant size used in copy_to(or from)_user function.	mlog_doread	727	🔍	🟩

OPEN FULL SCREEN

```
717             if (strfmt_idx == 0)
718                     v = '\n';
719
720             /* MLOG_PRINTK("[mlog] %d: %s\n", strfmt_idx, strfmt_list[strfmt_idx]); */
721             size = snprintf(mlog_str, MLOG_STR_LEN, strfmt_list[strfmt_idx++], v);
722
723             if (strfmt_idx >= strfmt_len)
724                     strfmt_idx = strfmt_proc;
725
726             spin_unlock_bh(&mlogbuf_lock);
727             if (__copy_to_user(buf, mlog_str, size))
728                     error = -EFAULT;
729             else {
730                     buf += size;
731                     i += size;
732             }
733
734             cond_resched();
735             spin_lock_bh(&mlogbuf_lock);
736     }
```

$ DR.CHECKER: Open Source and Dockerized

Tested on Qualcomm, MediaTek, Huawei and Samsung kernels

CVE-2016-8433, CVE-2016-8472, CVE-2016-8470, CVE-2016-8471, CVE-2016-8448, CVE-2017-0797 and more..

https://github.com/ucsb-seclab/dr_checker

Use it and get rich from bug bounties :) 💲

$ DR.CHECKER is not enough!!

- Use-after-free.

- Bugs because of improper usage of kernel API functions.

```
buf = kmalloc(count, GFP_KERNEL); // if count is Zero.
 if(!buf) {  // buf will be  ZERO_PTR
    buf[0] = 1; // Kernel panic..
          ...
 }
```

$ Okay! Why is it hard to find these bugs?

Good 'ol knowledge on Bug Detection:

- Static Analysis
- **Dynamic Analysis**

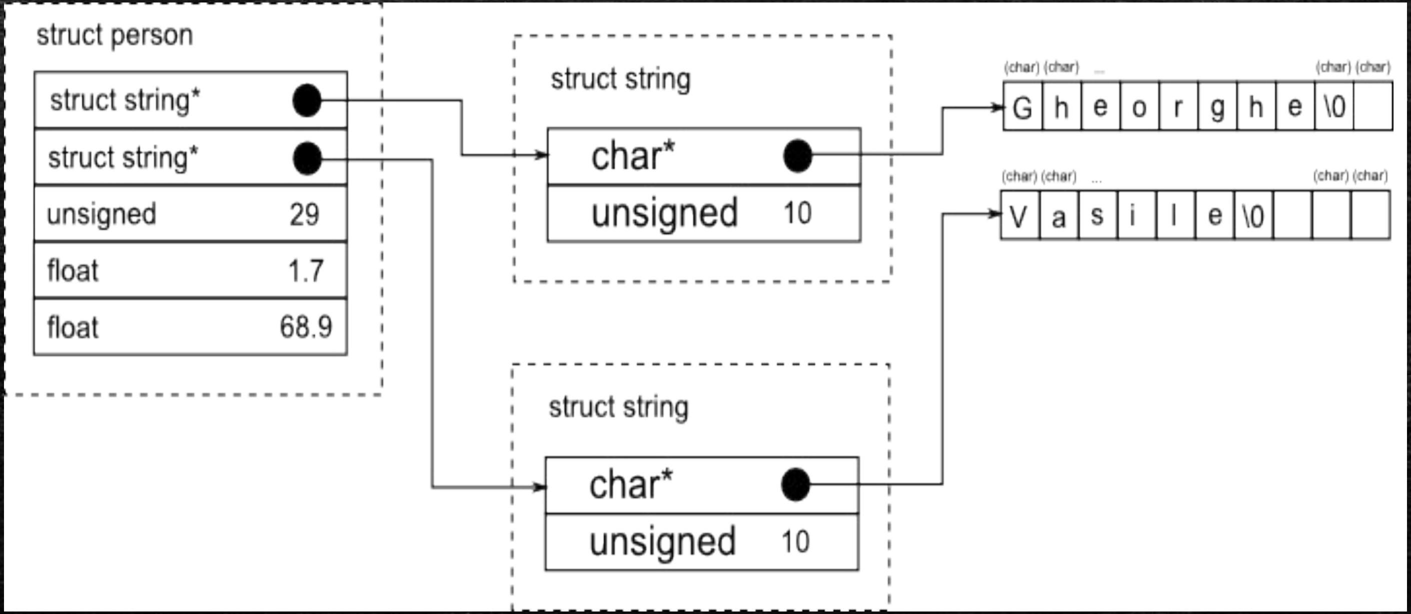
$ Dynamic Analysis: Fuzzing!!

- Hundreds of tools: AFL, Peach etc

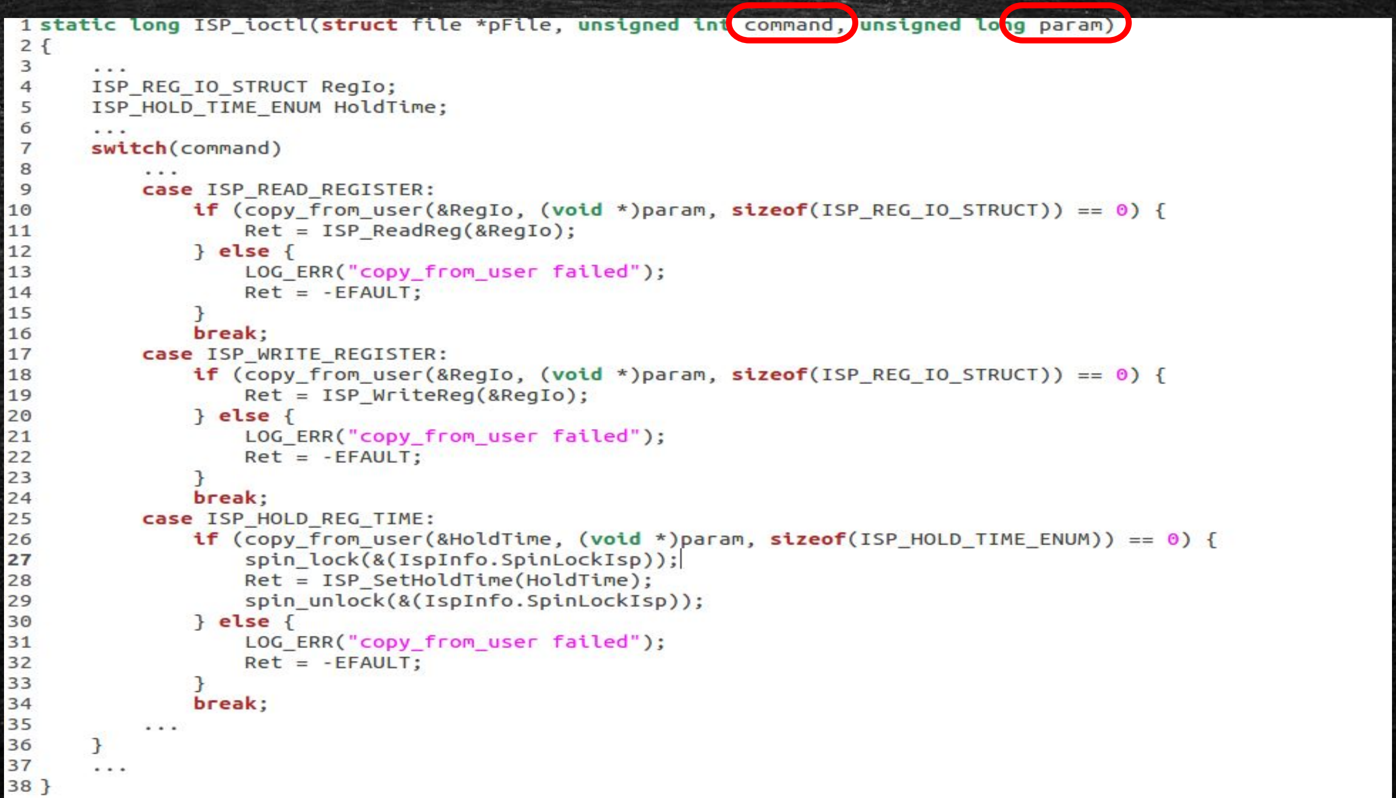
- Just use AFL..

$ Fuzzing: Good Luck!!

Drivers expect **highly structured and constrained input.**

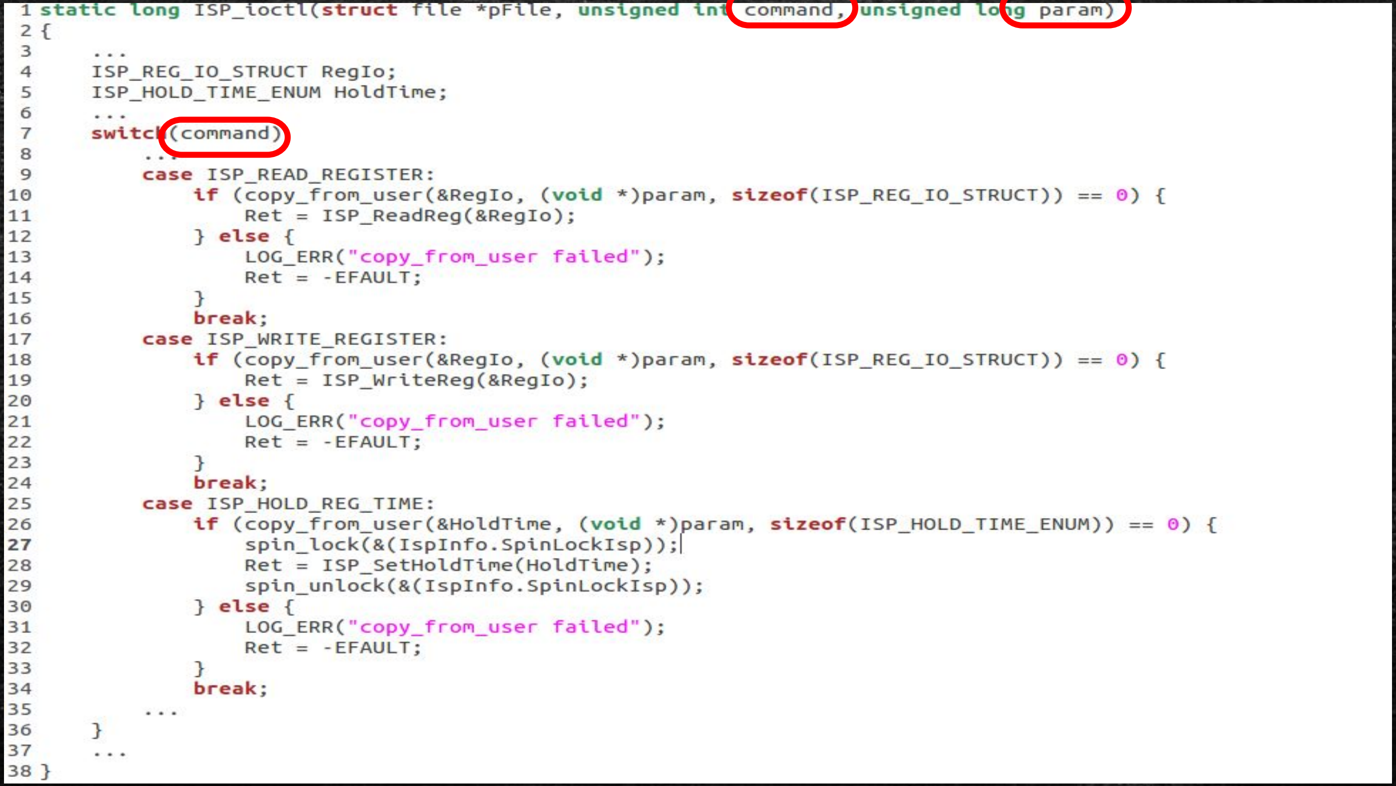


$ Fuzzing: Highly constrained input



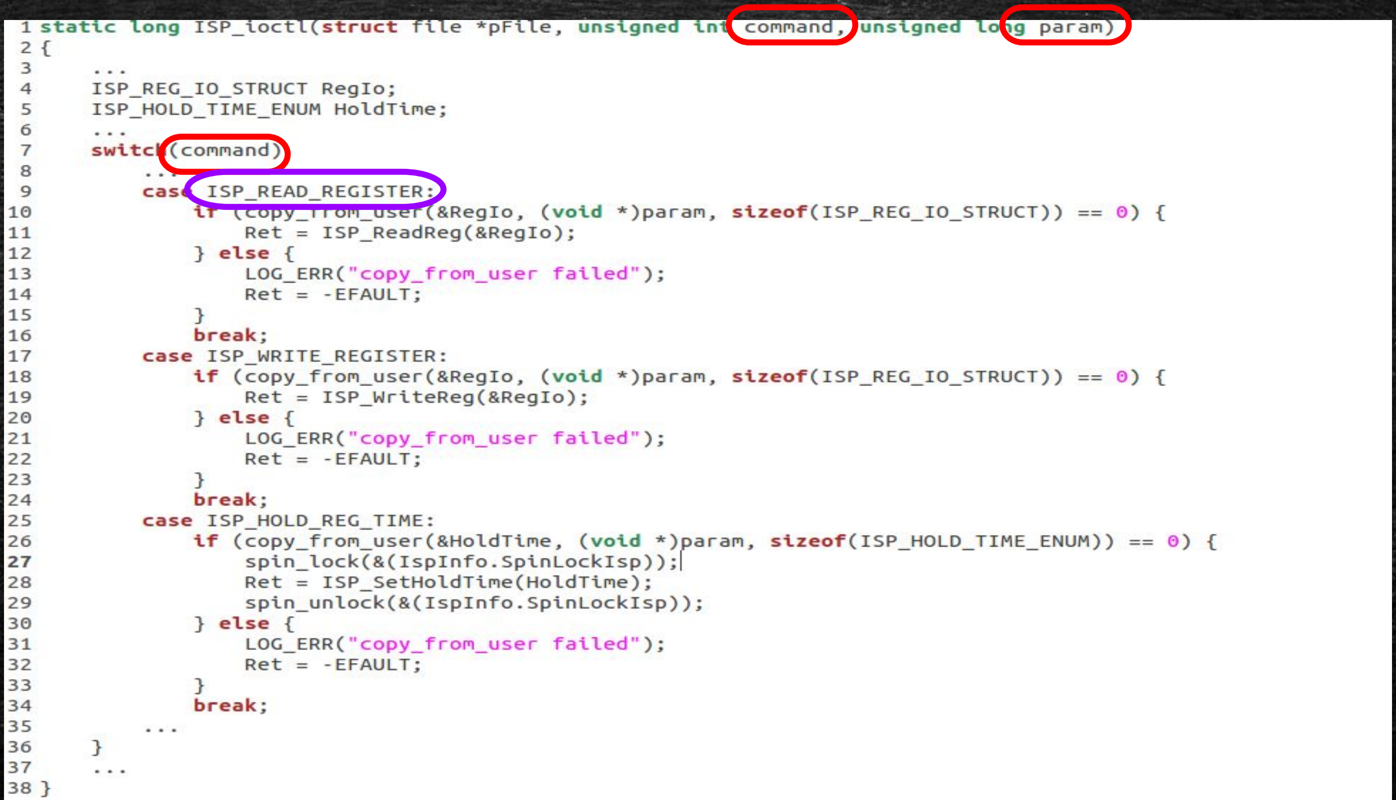
```
static long ISP_ioctl(struct file *pFile, unsigned int command, unsigned long param)
{
    ...
    ISP_REG_IO_STRUCT RegIo;
    ISP_HOLD_TIME_ENUM HoldTime;
    ...
    switch(command)
        ...
        case ISP_READ_REGISTER:
            if (copy_from_user(&RegIo, (void *)param, sizeof(ISP_REG_IO_STRUCT)) == 0) {
                Ret = ISP_ReadReg(&RegIo);
            } else {
                LOG_ERR("copy_from_user failed");
                Ret = -EFAULT;
            }
            break;
        case ISP_WRITE_REGISTER:
            if (copy_from_user(&RegIo, (void *)param, sizeof(ISP_REG_IO_STRUCT)) == 0) {
                Ret = ISP_WriteReg(&RegIo);
            } else {
                LOG_ERR("copy_from_user failed");
                Ret = -EFAULT;
            }
            break;
        case ISP_HOLD_REG_TIME:
            if (copy_from_user(&HoldTime, (void *)param, sizeof(ISP_HOLD_TIME_ENUM)) == 0) {
                spin_lock(&(IspInfo.SpinLockIsp));
                Ret = ISP_SetHoldTime(HoldTime);
                spin_unlock(&(IspInfo.SpinLockIsp));
            } else {
                LOG_ERR("copy_from_user failed");
                Ret = -EFAULT;
            }
            break;
        ...
    }
    ...
}
```

$ Fuzzing: Highly constrained input



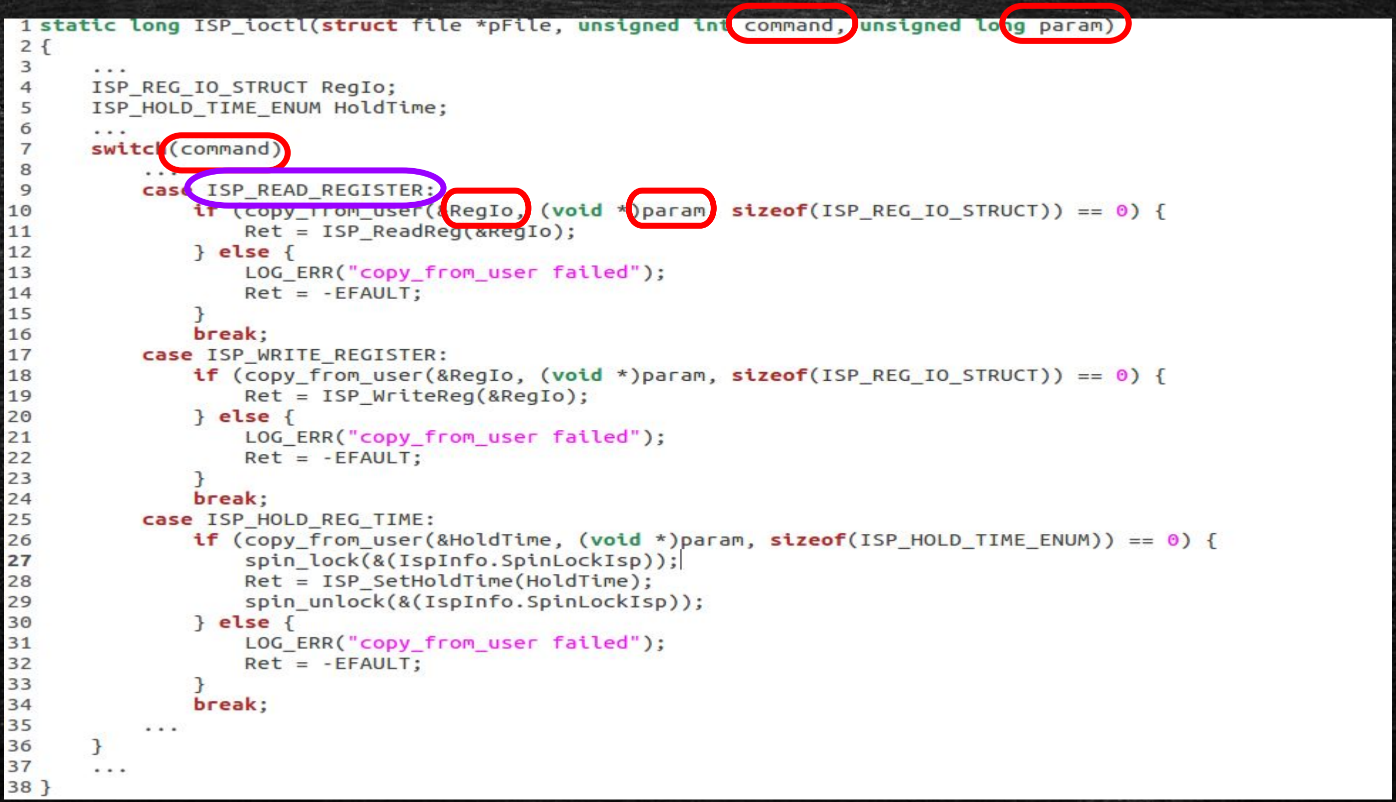
```
static long ISP_ioctl(struct file *pFile, unsigned int command, unsigned long param)
{
    ...
    ISP_REG_IO_STRUCT RegIo;
    ISP_HOLD_TIME_ENUM HoldTime;
    ...
    switch(command)
        ...
        case ISP_READ_REGISTER:
            if (copy_from_user(&RegIo, (void *)param, sizeof(ISP_REG_IO_STRUCT)) == 0) {
                Ret = ISP_ReadReg(&RegIo);
            } else {
                LOG_ERR("copy_from_user failed");
                Ret = -EFAULT;
            }
            break;
        case ISP_WRITE_REGISTER:
            if (copy_from_user(&RegIo, (void *)param, sizeof(ISP_REG_IO_STRUCT)) == 0) {
                Ret = ISP_WriteReg(&RegIo);
            } else {
                LOG_ERR("copy_from_user failed");
                Ret = -EFAULT;
            }
            break;
        case ISP_HOLD_REG_TIME:
            if (copy_from_user(&HoldTime, (void *)param, sizeof(ISP_HOLD_TIME_ENUM)) == 0) {
                spin_lock(&(IspInfo.SpinLockIsp));
                Ret = ISP_SetHoldTime(HoldTime);
                spin_unlock(&(IspInfo.SpinLockIsp));
            } else {
                LOG_ERR("copy_from_user failed");
                Ret = -EFAULT;
            }
            break;
        ...
    }
    ...
}
```

$ Fuzzing: Highly constrained input



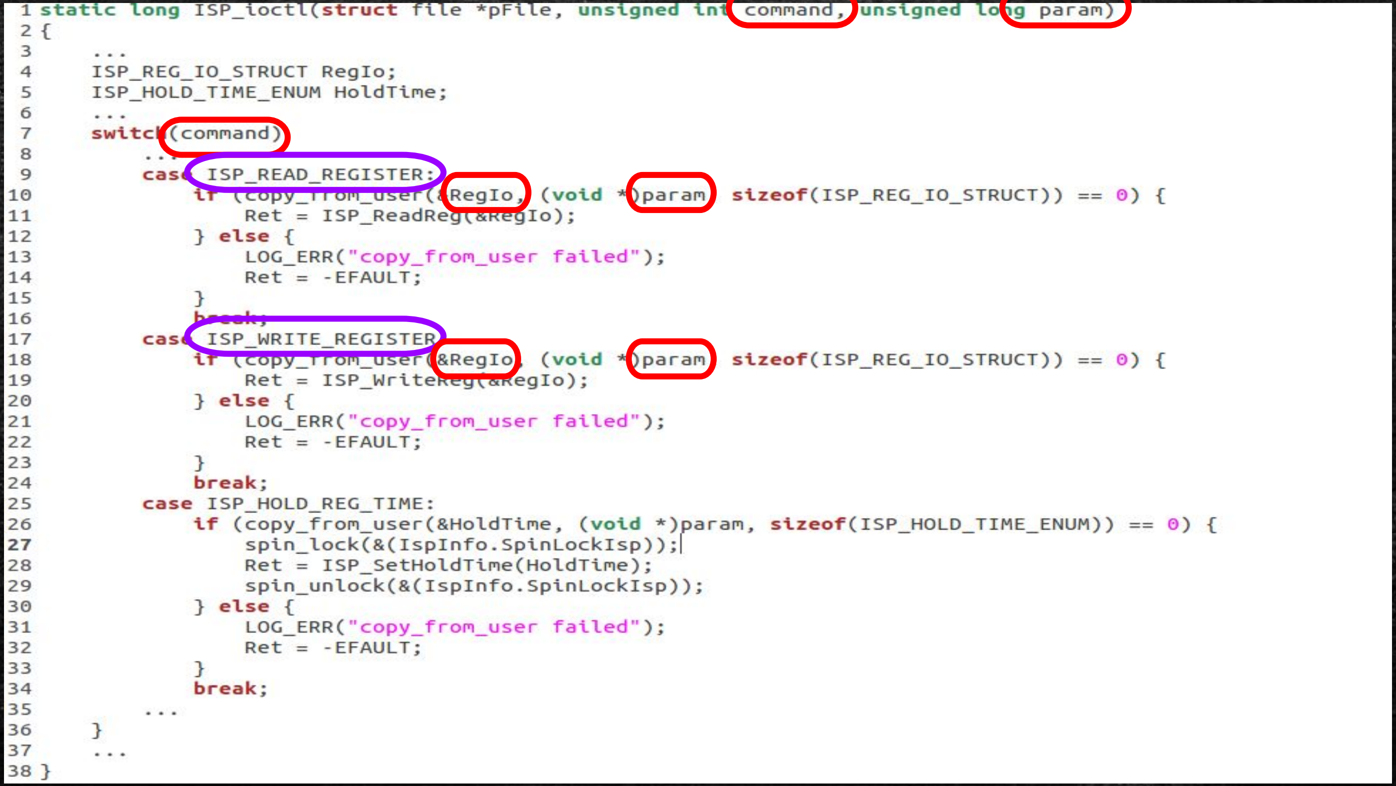
```
static long ISP_ioctl(struct file *pFile, unsigned int command, unsigned long param)
{
    ...
    ISP_REG_IO_STRUCT RegIo;
    ISP_HOLD_TIME_ENUM HoldTime;
    ...
    switch(command)
        ...
        case ISP_READ_REGISTER:
            if (copy_from_user(&RegIo, (void *)param, sizeof(ISP_REG_IO_STRUCT)) == 0) {
                Ret = ISP_ReadReg(&RegIo);
            } else {
                LOG_ERR("copy_from_user failed");
                Ret = -EFAULT;
            }
            break;
        case ISP_WRITE_REGISTER:
            if (copy_from_user(&RegIo, (void *)param, sizeof(ISP_REG_IO_STRUCT)) == 0) {
                Ret = ISP_WriteReg(&RegIo);
            } else {
                LOG_ERR("copy_from_user failed");
                Ret = -EFAULT;
            }
            break;
        case ISP_HOLD_REG_TIME:
            if (copy_from_user(&HoldTime, (void *)param, sizeof(ISP_HOLD_TIME_ENUM)) == 0) {
                spin_lock(&(IspInfo.SpinLockIsp));
                Ret = ISP_SetHoldTime(HoldTime);
                spin_unlock(&(IspInfo.SpinLockIsp));
            } else {
                LOG_ERR("copy_from_user failed");
                Ret = -EFAULT;
            }
            break;
        ...
    }
    ...
}
```

$ Fuzzing: Highly constrained input



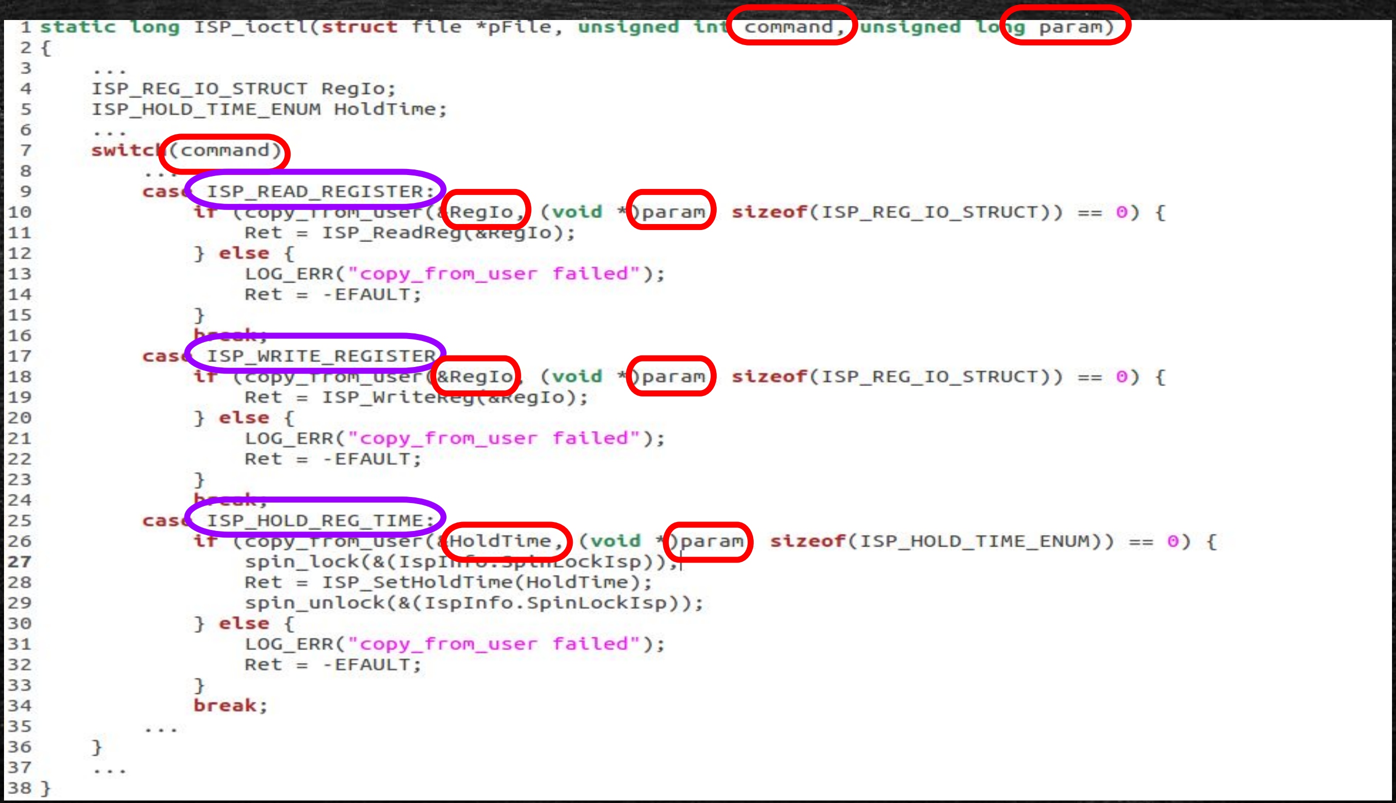
```
static long ISP_ioctl(struct file *pFile, unsigned int command, unsigned long param)
{
    ...
    ISP_REG_IO_STRUCT RegIo;
    ISP_HOLD_TIME_ENUM HoldTime;
    ...
    switch(command)
        ...
        case ISP_READ_REGISTER:
            if (copy_from_user(&RegIo, (void *)param, sizeof(ISP_REG_IO_STRUCT)) == 0) {
                Ret = ISP_ReadReg(&RegIo);
            } else {
                LOG_ERR("copy_from_user failed");
                Ret = -EFAULT;
            }
            break;
        case ISP_WRITE_REGISTER:
            if (copy_from_user(&RegIo, (void *)param, sizeof(ISP_REG_IO_STRUCT)) == 0) {
                Ret = ISP_WriteReg(&RegIo);
            } else {
                LOG_ERR("copy_from_user failed");
                Ret = -EFAULT;
            }
            break;
        case ISP_HOLD_REG_TIME:
            if (copy_from_user(&HoldTime, (void *)param, sizeof(ISP_HOLD_TIME_ENUM)) == 0) {
                spin_lock(&(IspInfo.SpinLockIsp));
                Ret = ISP_SetHoldTime(HoldTime);
                spin_unlock(&(IspInfo.SpinLockIsp));
            } else {
                LOG_ERR("copy_from_user failed");
                Ret = -EFAULT;
            }
            break;
        ...
    }
    ...
}
```

$ Fuzzing: Highly constrained input



```
static long ISP_ioctl(struct file *pFile, unsigned int command, unsigned long param)
{
    ...
    ISP_REG_IO_STRUCT RegIo;
    ISP_HOLD_TIME_ENUM HoldTime;
    ...
    switch(command)
        ...
        case ISP_READ_REGISTER:
            if (copy_from_user(&RegIo, (void *)param, sizeof(ISP_REG_IO_STRUCT)) == 0) {
                Ret = ISP_ReadReg(&RegIo);
            } else {
                LOG_ERR("copy_from_user failed");
                Ret = -EFAULT;
            }
            break;
        case ISP_WRITE_REGISTER:
            if (copy_from_user(&RegIo, (void *)param, sizeof(ISP_REG_IO_STRUCT)) == 0) {
                Ret = ISP_WriteReg(&RegIo);
            } else {
                LOG_ERR("copy_from_user failed");
                Ret = -EFAULT;
            }
            break;
        case ISP_HOLD_REG_TIME:
            if (copy_from_user(&HoldTime, (void *)param, sizeof(ISP_HOLD_TIME_ENUM)) == 0) {
                spin_lock(&(IspInfo.SpinLockIsp));
                Ret = ISP_SetHoldTime(HoldTime);
                spin_unlock(&(IspInfo.SpinLockIsp));
            } else {
                LOG_ERR("copy_from_user failed");
                Ret = -EFAULT;
            }
            break;
        ...
    }
    ...
}
```

$ Fuzzing: Highly constrained input



```
static long ISP_ioctl(struct file *pFile, unsigned int command, unsigned long param)
{
    ...
    ISP_REG_IO_STRUCT RegIo;
    ISP_HOLD_TIME_ENUM HoldTime;
    ...
    switch(command)
        ...
        case ISP_READ_REGISTER:
            if (copy_from_user(&RegIo, (void *)param, sizeof(ISP_REG_IO_STRUCT)) == 0) {
                Ret = ISP_ReadReg(&RegIo);
            } else {
                LOG_ERR("copy_from_user failed");
                Ret = -EFAULT;
            }
            break;
        case ISP_WRITE_REGISTER:
            if (copy_from_user(&RegIo, (void *)param, sizeof(ISP_REG_IO_STRUCT)) == 0) {
                Ret = ISP_WriteReg(&RegIo);
            } else {
                LOG_ERR("copy_from_user failed");
                Ret = -EFAULT;
            }
            break;
        case ISP_HOLD_REG_TIME:
            if (copy_from_user(&HoldTime, (void *)param, sizeof(ISP_HOLD_TIME_ENUM)) == 0) {
                spin_lock(&(IspInfo.SpinLockIsp));
                Ret = ISP_SetHoldTime(HoldTime);
                spin_unlock(&(IspInfo.SpinLockIsp));
            } else {
                LOG_ERR("copy_from_user failed");
                Ret = -EFAULT;
            }
            break;
        ...
    }
    ...
}
```

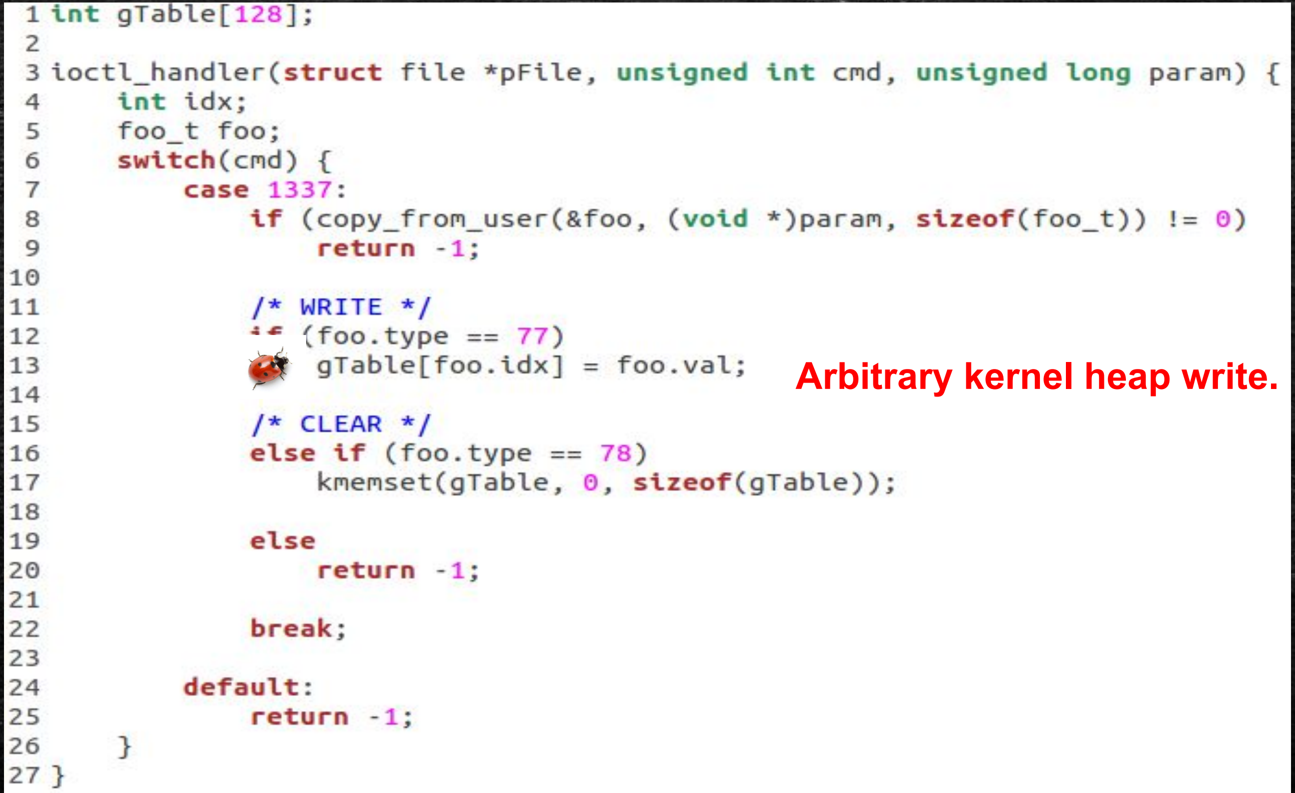
$ Drivers Expect Highly structured input

- If command == ISP_READ_REGISTER then param should be a valid user pointer to ISP_REG_IO_STRUCT.

- If command == ISP_WRITE_REGISTER then param should be a valid user pointer to ISP_REG_IO_STRUCT.

- If command == ISP_HOLD_REG_TIME then param should be a valid user pointer to ISP_HOLD_TIME_ENUM.

$ Bugs are hard to trigger



```
int gTable[128];

ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
    int idx;
    foo_t foo;
    switch(cmd) {
        case 1337:
            if (copy_from_user(&foo, (void *)param, sizeof(foo_t)) != 0)
                return -1;

            /* WRITE */
            if (foo.type == 77)
                gTable[foo.idx] = foo.val;

            /* CLEAR */
            else if (foo.type == 78)
                kmemset(gTable, 0, sizeof(gTable));

            else
                return -1;

            break;

        default:
            return -1;
    }
}
```

Arbitrary kernel heap write.

$ Bugs are hard to trigger

- You can trigger the bug, only if command == 1337 and param is a valid pointer to the structure:

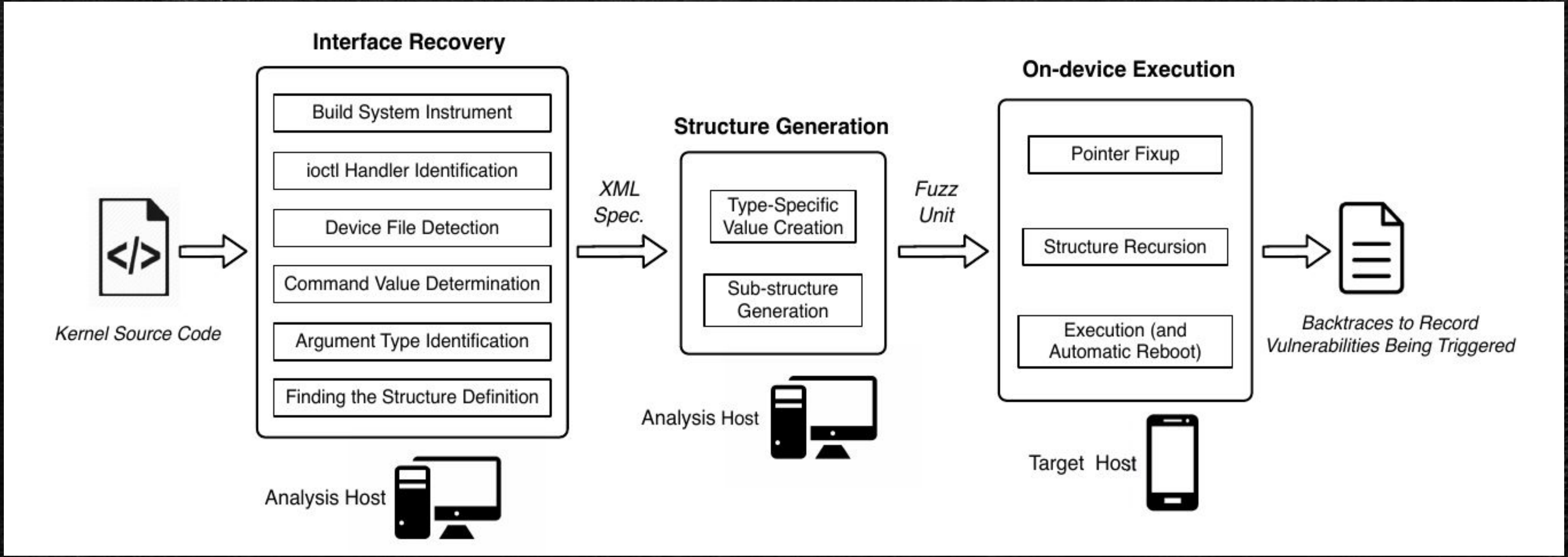
```
typedef struct {
      type_enum type; ( == 77)
      int idx; ( >= 128)
      int val;
} foo_t
```

$ DIFUZE: Idea

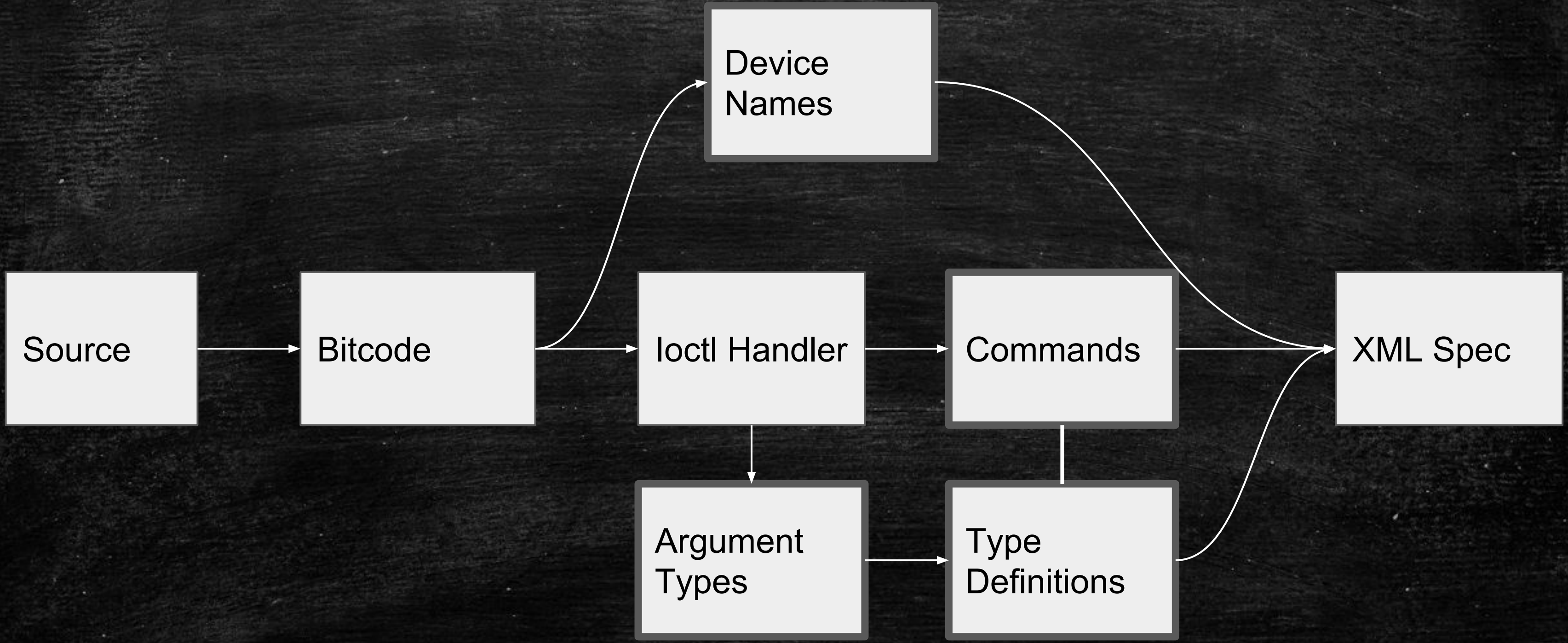
- Recover all the command values, corresponding param types automatically.

- This will reduce the state space and help in effective fuzzing.

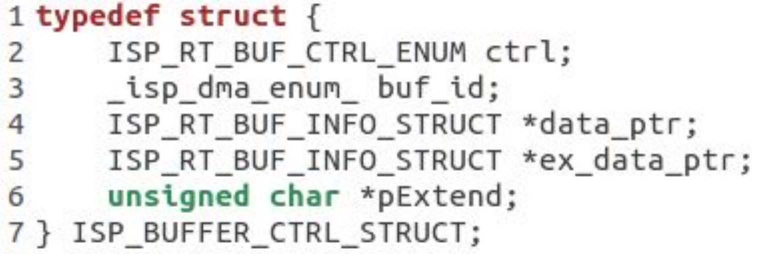
$ DIFUZE: Overview



$ DIFUZE: Interface Recovery

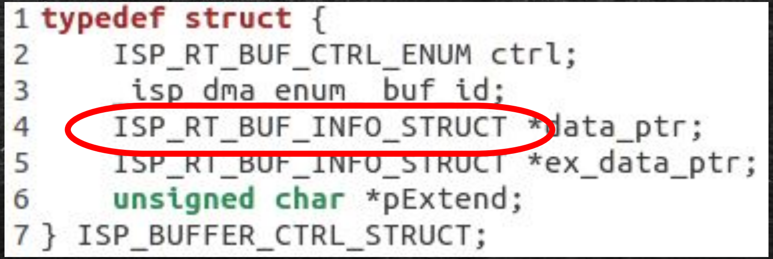


$ DIFUZE: Structure Generation

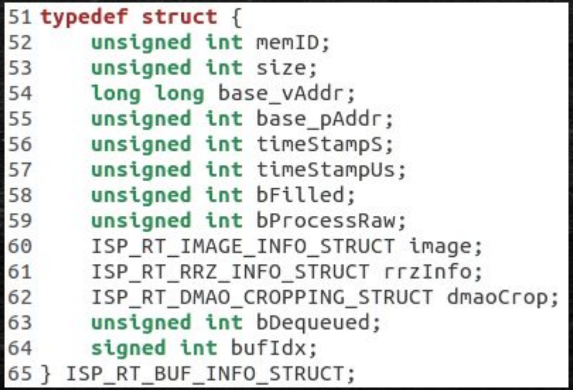


```
typedef struct {
    ISP_RT_BUF_CTRL_ENUM ctrl;
    _isp_dma_enum_ buf_id;
    ISP_RT_BUF_INFO_STRUCT *data_ptr;
    ISP_RT_BUF_INFO_STRUCT *ex_data_ptr;
    unsigned char *pExtend;
} ISP_BUFFER_CTRL_STRUCT;
```

$ DIFUZE: Structure Generation



```
typedef struct {
    ISP_RT_BUF_CTRL_ENUM ctrl;
     isp_dma_enum  buf_id;
    ISP_RT_BUF_INFO_STRUCT *data_ptr;
    ISP_RT_BUF_INFO_STRUCT *ex_data_ptr;
    unsigned char *pExtend;
} ISP_BUFFER_CTRL_STRUCT;
```



```
typedef struct {
    unsigned int memID;
    unsigned int size;
    long long base_vAddr;
    unsigned int base_pAddr;
    unsigned int timeStampS;
    unsigned int timeStampUs;
    unsigned int bFilled;
    unsigned int bProcessRaw;
    ISP_RT_IMAGE_INFO_STRUCT image;
    ISP_RT_RRZ_INFO_STRUCT rrzInfo;
    ISP_RT_DMAO_CROPPING_STRUCT dmaoCrop;
    unsigned int bDequeued;
    signed int bufIdx;
} ISP_RT_BUF_INFO_STRUCT;
```

$ DIFUZE: On Device Execution

- Run on the phone connected to host device via ADB (Android Debug Bridge).

- Map the binary data, do pointer fix ups.

- Open device and perform the ioctl.

$ DIFUZE: Evaluation

Manufacturer	**Device**	**Chipset**
Google	Pixel	Qualcomm
HTC	E9 Plus	Mediatek
HTC	One M9	Qualcomm
Huawei	P9 Lite	Huawei
Huawei	Honor 8	Huawei
Samsung	Galaxy S6	Samsung
Sony	Xperia XA	Mediatek

$ DIFUZE: Evaluation

	Total Unique
E9 Plus	6
Galaxy S6	0
Honor 8	2
One M9	3
P9 Lite	6
Pixel	5
Xperia XA	14
Total	**36**

$ DIFUZE: Bug Types

Crash Type	Count
Arbitrary Read	**4**
Arbitrary Write	**4**
Assert Failure	**6**
Buffer Overflow	**2**
Null Dereference	**9**
Out of Bound Index	**5**
Uncategorized	**5**
Writing to non-volatile memory	**1**

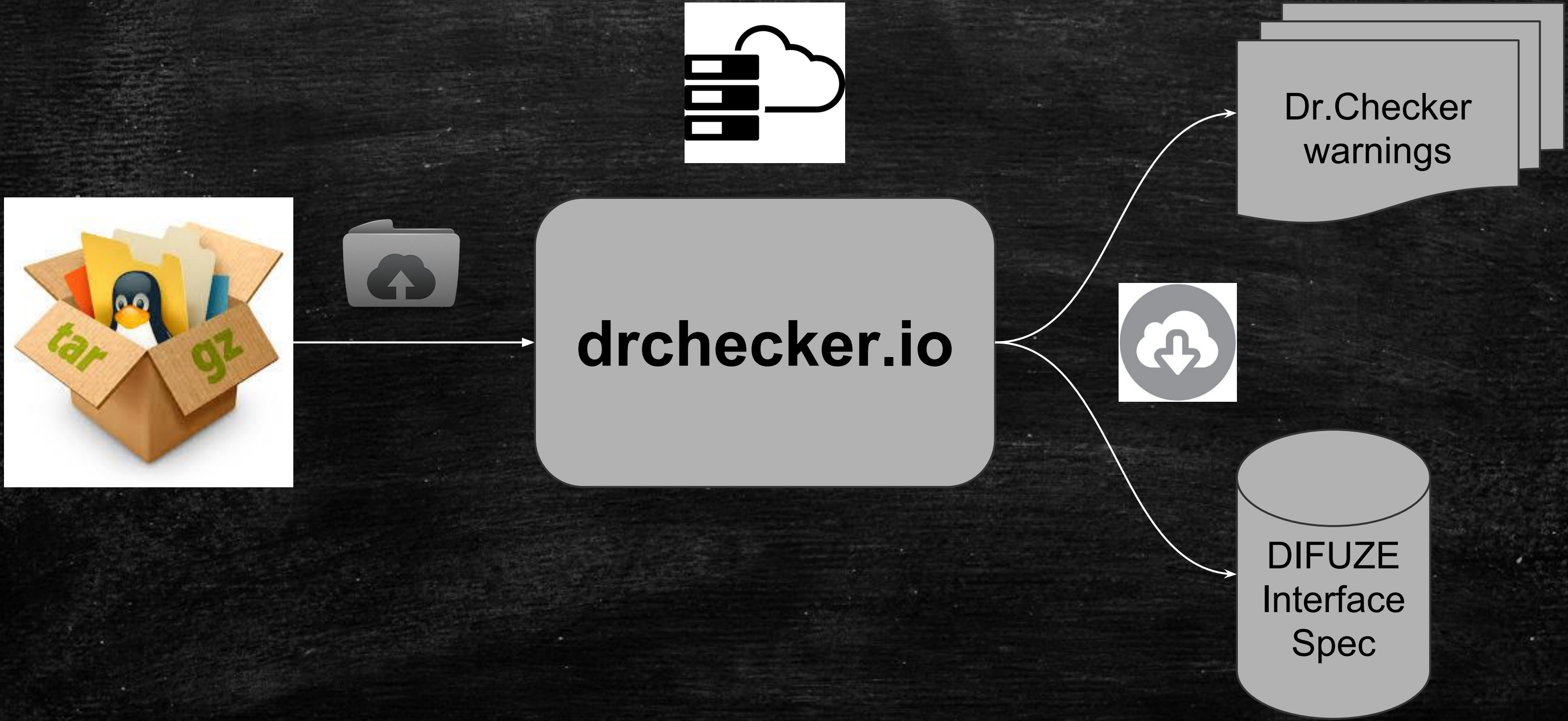
$ DIFUZE: Open Source

CVE-2017-15307, CVE-2017-0636, CVE-2017-0804, and many more.

https://github.com/ucsb-seclab/difuze

Use it and become rich :)

$ In Progress: drchecker.io

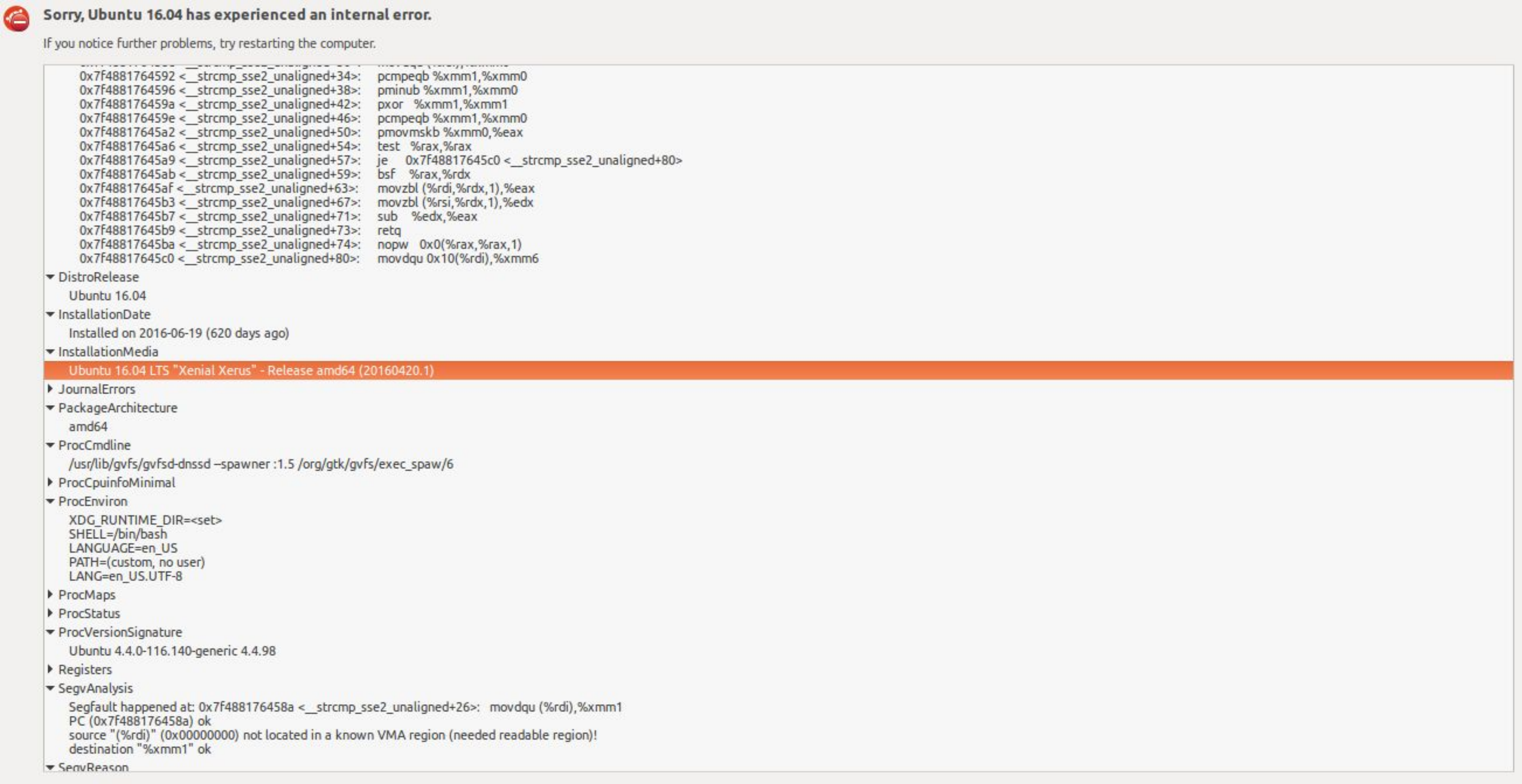


drchecker.io

Dr.Checker warnings

DIFUZE Interface Spec

$ Souvenir :)



Sorry, Ubuntu 16.04 has experienced an internal error.

If you notice further problems, try restarting the computer.

```
    0x7f4881764592 <__strcmp_sse2_unaligned+34>:   pcmpeqb %xmm1,%xmm0
    0x7f4881764596 <__strcmp_sse2_unaligned+38>:   pminub %xmm1,%xmm0
    0x7f488176459a <__strcmp_sse2_unaligned+42>:   pxor   %xmm1,%xmm1
    0x7f488176459e <__strcmp_sse2_unaligned+46>:   pcmpeqb %xmm1,%xmm0
    0x7f48817645a2 <__strcmp_sse2_unaligned+50>:   pmovmskb %xmm0,%eax
    0x7f48817645a6 <__strcmp_sse2_unaligned+54>:   test   %rax,%rax
    0x7f48817645a9 <__strcmp_sse2_unaligned+57>:   je     0x7f48817645c0 <__strcmp_sse2_unaligned+80>
    0x7f48817645ab <__strcmp_sse2_unaligned+59>:   bsf    %rax,%rdx
    0x7f48817645af <__strcmp_sse2_unaligned+63>:   movzbl (%rdi,%rdx,1),%eax
    0x7f48817645b3 <__strcmp_sse2_unaligned+67>:   movzbl (%rsi,%rdx,1),%edx
    0x7f48817645b7 <__strcmp_sse2_unaligned+71>:   sub    %edx,%eax
    0x7f48817645b9 <__strcmp_sse2_unaligned+73>:   retq
    0x7f48817645ba <__strcmp_sse2_unaligned+74>:   nopw   0x0(%rax,%rax,1)
    0x7f48817645c0 <__strcmp_sse2_unaligned+80>:   movdqu 0x10(%rdi),%xmm6
▼ DistroRelease
   Ubuntu 16.04
▼ InstallationDate
   Installed on 2016-06-19 (620 days ago)
▼ InstallationMedia
   Ubuntu 16.04 LTS "Xenial Xerus" - Release amd64 (20160420.1)
▶ JournalErrors
▼ PackageArchitecture
   amd64
▼ ProcCmdline
   /usr/lib/gvfs/gvfsd-dnssd –spawner :1.5 /org/gtk/gvfs/exec_spaw/6
▶ ProcCpuinfoMinimal
▼ ProcEnviron
   XDG_RUNTIME_DIR=<set>
   SHELL=/bin/bash
   LANGUAGE=en_US
   PATH=(custom, no user)
   LANG=en_US.UTF-8
▶ ProcMaps
▶ ProcStatus
▼ ProcVersionSignature
   Ubuntu 4.4.0-116.140-generic 4.4.98
▶ Registers
▼ SegvAnalysis
   Segfault happened at: 0x7f488176458a <__strcmp_sse2_unaligned+26>:   movdqu (%rdi),%xmm1
   PC (0x7f488176458a) ok
   source "(%rdi)" (0x00000000) not located in a known VMA region (needed readable region)!
   destination "%xmm1" ok
▼ SegvReason
```

$ Thank You

github.com/ucsb-seclab/**dr_checker**

github.com/ucsb-seclab/**difuze**