

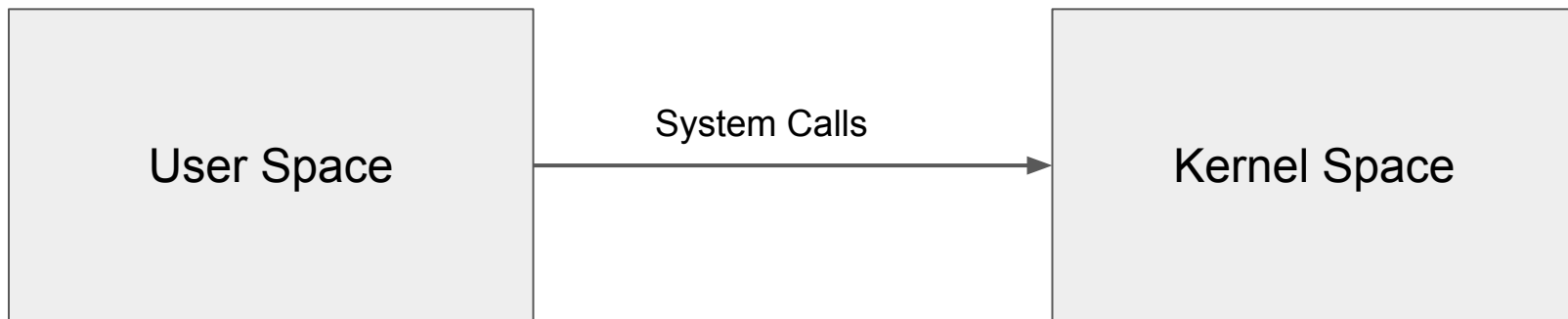
Fuzzing Kernel Drivers with Interface Awareness

By Jake Corina and Chris Salls



Linux Kernel

- Generally:



- Kernel takes care of many things (process management, memory management, filesystems, **device control**, etc).
- Device handling is done in the kernel.

Device Drivers

- Generally implemented as kernel modules
- There must be an interface exported to userland
- Same as standard interface
- How is this done?
- “Everything is a file”
- Special *device files* on disk

Operations on Files

- Let's look at how it's done for a “normal” file.
- Common syscalls: open, read, write, lseek, etc.
- file_operations structure -- “fops”, “f_ops”

```
const struct file_operations ext4_file_operations = {  
    .llseek      = ext4_llseek,  
    .read        = new_sync_read,  
    .write       = new_sync_write,  
    .read_iter   = generic_file_read_iter,  
    .write_iter  = ext4_file_write_iter,  
    .unlocked_ioctl = ext4_ioctl,  
    .mmap        = ext4_file_mmap,  
    .open        = ext4_file_open,  
    .release     = ext4_release_file,  
};
```

Device Files

- As we said, device drivers generally create special files on disk to represent the physical device (for which they drive).
- How is this done?
- What's the process, from device registration, to having a file to talk to?

```

1 static inline MINT32 ISP_RegCharDev(void)
2 {
3     MINT32 Ret = 0;
4     /*
5     LOG_DBG("- E.");
6     */
7     Ret = alloc_chrdev_region(&IspDevNo, 0, 1, ISP_DEV_NAME);
8     if (Ret < 0) {
9         LOG_ERR("alloc_chrdev_region failed, %d", Ret);
10        return Ret;
11    }
12    /* Allocate driver */
13    pIspCharDrv = cdev_alloc();
14    if (pIspCharDrv == NULL) {
15        LOG_ERR("cdev_alloc failed");
16        Ret = -ENOMEM;
17        goto EXIT;
18    }
19    /* Attatch file operation. */
20    cdev_init(pIspCharDrv, &IspFileOper);
21    /*
22    pIspCharDrv->owner = THIS_MODULE;
23    */
24    /* Add to system */
25    Ret = cdev_add(pIspCharDrv, IspDevNo, 1);
26    if (Ret < 0) {
27        LOG_ERR("Attatch file operation failed, %d", Ret);
28        goto EXIT;
29    }
30    /*
31    */
32    if (Ret < 0)
33        ISP_UnregCharDev();
34    /*
35    */
36    LOG_DBG("- X.");
37    return Ret;
38 }

```

```

1 static inline MINT32 ISP_RegCharDev(void)
2 {
3     MINT32 Ret = 0;
4     /*
5      * LOG_DBG("- E.");
6      */
7     Ret = alloc_chrdev_region(&IspDevNo, 0, 1, ISP_DEV_NAME);
8     if (Ret < 0) {
9         LOG_ERR("alloc_chrdev_region failed, %d", Ret);
10        return Ret;
11    }
12    /* Allocate driver */
13    pIspCharDrv = cdev_alloc();
14    if (pIspCharDrv == NULL) {
15        LOG_ERR("cdev_alloc failed");
16        Ret = -ENOMEM;
17        goto EXIT;
18    }
19    /* Attatch file operation. */
20    cdev_init(pIspCharDrv, &IspFileOper);
21    /*
22     * pIspCharDrv->owner = THIS_MODULE;
23     */
24    /* Add to system */
25    Ret = cdev_add(pIspCharDrv, IspDevNo, 1);
26    if (Ret < 0) {
27        LOG_ERR("Attatch file operation failed, %d", Ret);
28        goto EXIT;
29    }
30    /*
31     */
32    if (Ret < 0)
33        ISP_UnregCharDev();
34    /*
35     */
36    LOG_DBG("- X.");
37    return Ret;
38 }

```

Request a region of char device numbers.
Place the allocated region into **IspDevNo**.
Associate the range with **ISP_DEV_NAME**

```

1 static inline MINT32 ISP_RegCharDev(void)
2 {
3     MINT32 Ret = 0;
4     /* */
5     LOG_DBG("- E.");
6     /* */
7     Ret = alloc_chrdev_region(&IspDevNo, 0, 1, ISP_DEV_NAME);
8     if (Ret < 0) {
9         LOG_ERR("alloc_chrdev_region failed, %d", Ret);
10        return Ret;
11    }
12    /* Allocate driver */
13    pIspCharDrv = cdev_alloc();
14    if (pIspCharDrv == NULL) {
15        LOG_ERR("cdev_alloc failed");
16        Ret = -ENOMEM;
17        goto EXIT;
18    }
19    /* Attatch file operation. */
20    cdev_init(pIspCharDrv, &IspFileOper);
21    /* */
22    pIspCharDrv->owner = THIS_MODULE;
23    /* Add to system */
24    Ret = cdev_add(pIspCharDrv, IspDevNo, 1);
25    if (Ret < 0) {
26        LOG_ERR("Attatch file operation failed, %d", Ret);
27        goto EXIT;
28    }
29    /* */
30 EXIT:
31    if (Ret < 0)
32        ISP_UnregCharDev();
33
34    /* */
35
36    LOG_DBG("- X.");
37    return Ret;
38 }

```

Request a region of char device numbers.
Place the allocated region into **IspDevNo**.
Associate the range with **ISP_DEV_NAME**

```

#define ISP_DEV_NAME          "camera-isp"

```

```

1 static inline MINT32 ISP_RegCharDev(void)
2 {
3     MINT32 Ret = 0;
4     /*
5     LOG_DBG("- E.");
6     */
7     Ret = alloc_chrdev_region(&IspDevNo, 0, 1, ISP_DEV_NAME);
8     if (Ret < 0) {
9         LOG_ERR("alloc_chrdev_region failed, %d", Ret);
10        return Ret;
11    }
12    /* Allocate driver */
13    pIspCharDrv = cdev_alloc();
14    if (pIspCharDrv == NULL) {
15        LOG_ERR("cdev_alloc failed");
16        Ret = -ENOMEM;
17        goto EXIT;
18    }
19    /* Attatch file operation. */
20    cdev_init(pIspCharDrv, &IspFileOper);
21    /*
22    pIspCharDrv->owner = THIS_MODULE;
23    */
24    /* Add to system */
25    Ret = cdev_add(pIspCharDrv, IspDevNo, 1);
26    if (Ret < 0) {
27        LOG_ERR("Attatch file operation failed, %d", Ret);
28        goto EXIT;
29    }
30    /*
31    */
32    if (Ret < 0)
33        ISP_UnregCharDev();
34    /*
35    */
36    LOG_DBG("- X.");
37    return Ret;
38 }

```

Associate the allocated device with **IspFileOper**, a file_operations structure.

```

1 static inline MINT32 ISP_RegCharDev(void)
2 {
3     MINT32 Ret = 0;
4     /*
5     LOG_DBG("- E.");
6     */
7     Ret = alloc_chrdev_region(&IspDevNo, 0, 1, ISP_DEV_NAME);
8     if (Ret < 0) {
9         LOG_ERR("alloc_chrdev_region failed, %d", Ret);
10        return Ret;
11    }
12    /* Allocate driver */
13    pIspCharDrv = cdev_alloc();
14    if (pIspCharDrv == NULL) {
15        LOG_ERR("cdev_alloc failed");
16        Ret = -ENOMEM;
17        goto EXIT;
18    }
19    /* Attatch file operation. */
20    cdev_init(pIspCharDrv, &IspFileOper);
21    /*
22    pIspCharDrv->owner = THIS_MODULE;
23    */
24    /* Add to system */
25    Ret = cdev_add(pIspCharDrv, IspDevNo, 1);
26    if (Ret < 0) {
27        LOG_ERR("Attatch file operation failed, %d", Ret);
28        goto EXIT;
29    }
30    /*
31    */
32    if (Ret < 0)
33        ISP_UnregCharDev();
34    /*
35    */
36    LOG_DBG("- X.");
37    return Ret;
38 }

```

Add the device to the system. At this point, the device is “live”.

```

root@F3116:/ # ls -l /dev/ | grep "camera-isp"
crw-rw---- system camera 243, 0 2017-05-18 08:43 camera-isp

```

```
static const struct file_operations IspFileOper = {  
    .owner = THIS_MODULE,  
    .open = ISP_open,  
    .release = ISP_release,  
    .mmap = ISP_mmap,  
    .unlocked_ioctl = ISP_ioctl  
};
```

ioctl(s)

- Input **O**utput **C**ontrol.
- System call to allow device operations that can't be well modeled as a “normal” syscall.

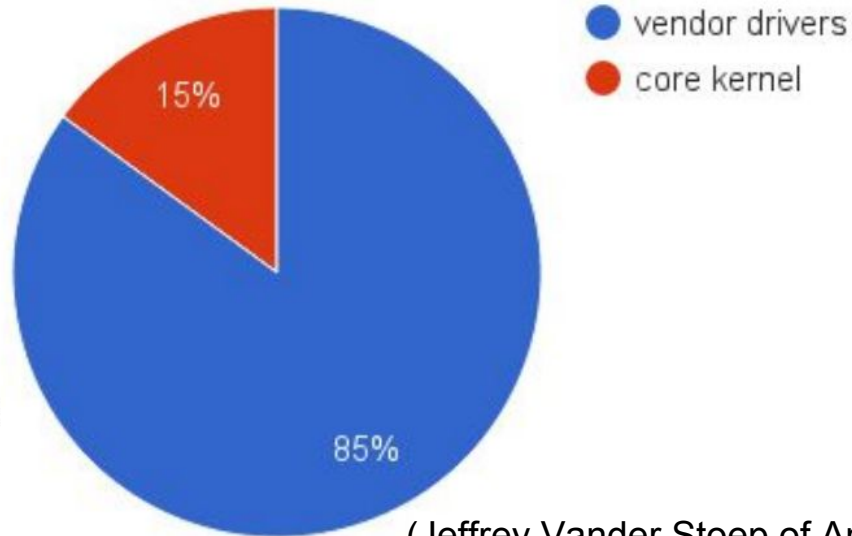
Things to Note

- Modular: Allows vendors to add support for hardware!
- Security implications?
- Where is this especially prevalent?

Android

- Based on Linux.
- Dominates the smartphone OS market. 86.8% of the market in 2016 Q3 (Source: IDC, Nov 2016)
- LOTS of hardware to support → Lots of drivers
- So what, are drivers really an issue?

Where Android's kernel bugs are born



Data includes multiple vendors

Some vendor drivers are in upstream, others are out-of-tree

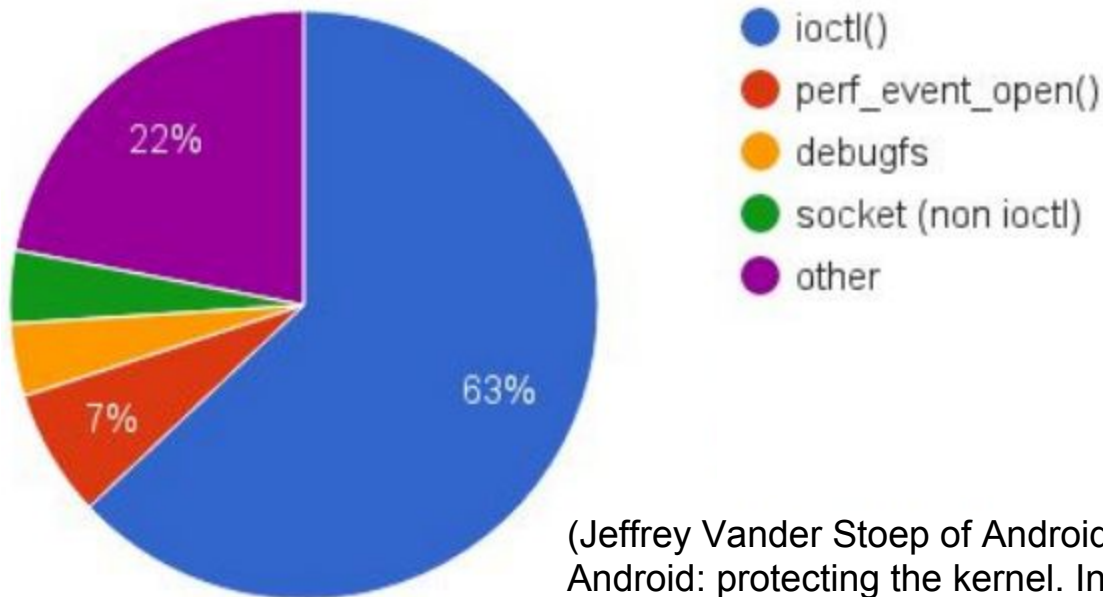
(Jeffrey Vander Stoep of Android Security. 2016. Android: protecting the kernel. In *Linux Security Summit*. Linux Foundation.)

Data: Jan 2014 → April 2016

Why?

- Well defined interface
- Which syscall is the problem?

How are kernel bugs reached - syscall (before mitigations)



(Jeffrey Vander Stoep of Android Security. 2016.
Android: protecting the kernel. In *Linux Security Summit*.
Linux Foundation.)

Data: Jan 2014 → April 2016

ioctl

```
ioctl(int fd,  
unsigned long command,  
unsigned long param);
```

```
static const struct file_operations IspFileOper = {  
    .owner = THIS_MODULE,  
    .open = ISP_open,  
    .release = ISP_release,  
    .mmap = ISP_mmap,  
    .unlocked_ioctl = ISP_ioctl ←  
};
```

```
1 static long ISP_ioctl(struct file *pFile, unsigned int Cmd, unsigned long Param)
2 {
3     ...
4     switch (Cmd) {
5         case ISP_READ_REGISTER:
6             if (copy_from_user(&RegIo, (void *)Param, sizeof(ISP_REG_IO_STRUCT)) == 0) { ←
7                 Ret = ISP_ReadReg(&RegIo);
8             } else {
9                 LOG_ERR("copy_from_user failed");
10                Ret = -EFAULT;
11            }
12            break;
13        case ISP_WRITE_REGISTER:
14            if (copy_from_user(&RegIo, (void *)Param, sizeof(ISP_REG_IO_STRUCT)) == 0) { ←
15                Ret = ISP_WriteReg(&RegIo);
16            } else {
17                LOG_ERR("copy_from_user failed");
18                Ret = -EFAULT;
19            }
20            break;
21        case ISP_WAIT_IRQ:
22            if (copy_from_user(&IrqInfo, (void *)Param, sizeof(ISP_WAIT_IRQ_STRUCT)) == 0) { ←
23                ...
24            }
25            break;
26        ...
27    }
28    ...
29 }
```

```
copy_from_user(void * to,  
               void __user * from,  
               unsigned long n)
```

```
1 static long ISP_ioctl(struct file *pFile, unsigned int Cmd, unsigned long Param)
2 {
3     ...
4     switch (Cmd) {
5         case ISP_READ_REGISTER:
6             if (copy_from_user(&RegIo, (void *)Param, sizeof(ISP_REG_IO_STRUCT)) == 0) { ←
7                 Ret = ISP_ReadReg(&RegIo);
8             } else {
9                 LOG_ERR("copy_from_user failed");
10                Ret = -EFAULT;
11            }
12            break;
13        case ISP_WRITE_REGISTER:
14            if (copy_from_user(&RegIo, (void *)Param, sizeof(ISP_REG_IO_STRUCT)) == 0) { ←
15                Ret = ISP_WriteReg(&RegIo);
16            } else {
17                LOG_ERR("copy_from_user failed");
18                Ret = -EFAULT;
19            }
20            break;
21        case ISP_WAIT_IRQ:
22            if (copy_from_user(&IrqInfo, (void *)Param, sizeof(ISP_WAIT_IRQ_STRUCT)) == 0) { ←
23                ...
24            }
25            break;
26        ...
27    }
28    ...
29 }
```

Solutions

- Static Analysis
 - Tons of false positives, huge amount of work to manually check every warning.
- Dynamic Analysis
 - Fuzzing

Fuzzing

- General idea: send random input to a program in hopes of triggering a bug
- Guaranteed real bugs, and we have a POC to go with it! :)

Kernel Fuzzing

- Model each syscall so we know how to call it and what to pass as arguments
- This is very hard for ioctls
- Recovering this interface requires LOTS of manual effort, and as such, ioctls are often neglected when fuzzing.
- Even with a recovered interface, it can be very hard to generate the correct arguments (super complex structs with embedded substructs, pointers, etc).

Fuzzing Drivers

Inputs:

Cmd: 43256
Param: 65443

Cmd: 1337
Param: 123654

Cmd: 1337
Param: → 4

Cmd: 1337
Param: → 500

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&idx, (void *)param, sizeof(int)) != 0)
9                 return -1;
10            gTable[idx] = 1;
11            break;
12
13        default:
14            return -1;
15    }
```

Fuzzing Drivers

Inputs:

Cmd: 43256
Param: 65443

Cmd: 1337
Param: 123654

Cmd: 1337
Param: → 4

Cmd: 1337
Param: → 500

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&idx, (void *)param, sizeof(int)) != 0)
9                 return -1;
10            gTable[idx] = 1; ←
11            break;
12
13        default:
14            return -1;
15    }
```

Fuzzing Drivers

Inputs:

Cmd: 43256
Param: 65443

Cmd: 1337
Param: 123654

Cmd: 1337
Param: → 4

Cmd: 1337
Param: → 500

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&idx, (void *)param, sizeof(int)) != 0)
9                 return -1;
10            gTable[idx] = 1;
11            break;
12
13        default:
14            return -1;
15    }
```

Fuzzing Drivers

Inputs:

Cmd: 43256
Param: 65443

Cmd: 1337
Param: 123654

Cmd: 1337
Param: → 4

Cmd: 1337
Param: → 500

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&idx, (void *)param, sizeof(int)) != 0)
9                 return -1;
10            gTable[idx] = 1;
11            break;
12
13        default:
14            return -1;
15    }
```

Fuzzing Drivers

Inputs:

Cmd: 43256
Param: 65443

Cmd: 1337
Param: 123654

Cmd: 1337
Param: → 4

Cmd: 1337
Param: → 500

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&idx, (void *)param, sizeof(int)) != 0)
9                 return -1;
10            gTable[idx] = 1;
11            break;
12
13        default:
14            return -1;
15    }
```

Fuzzing Drivers

Inputs:

Cmd: 43256
Param: 65443

Cmd: 1337
Param: 123654

Cmd: 1337
Param: → 4

Cmd: 1337
Param: → 500

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&idx, (void *)param, sizeof(int)) != 0)
9                 return -1;
10            gTable[idx] = 1;
11            break;
12
13        default:
14            return -1;
15    }
```

Fuzzing Drivers

Inputs:

Cmd: 43256
Param: 65443

Cmd: 1337
Param: 123654

Cmd: 1337
Param: → 4

Cmd: 1337
Param: → 500

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&idx, (void *)param, sizeof(int)) != 0)
9                 return -1;
10            gTable[idx] = 1;
11            break;
12
13            default:
14                return -1;
15    }
```

Fuzzing Drivers

Inputs:

Cmd: 43256
Param: 65443

Cmd: 1337
Param: 123654

Cmd: 1337
Param: → 4

Cmd: 1337
Param: → 500

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&idx, (void *)param, sizeof(int)) != 0)
9                 return -1;
10            gTable[idx] = 1;
11            break;
12
13        default:
14            return -1;
15    }
```

Fuzzing Drivers

Inputs:

Cmd: 43256
Param: 65443

Cmd: 1337
Param: 123654

Cmd: 1337
Param: → 4

Cmd: 1337
Param: → 500

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&idx, (void *)param, sizeof(int)) != 0)
9                 return -1;
10            gTable[idx] = 1;
11            break;
12
13        default:
14            return -1;
15    }
```

Fuzzing Drivers

Inputs:

Cmd: 43256
Param: 65443

Cmd: 1337
Param: 123654

Cmd: 1337
Param: → 4

Cmd: 1337
Param: → 500

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&idx, (void *)param, sizeof(int)) != 0)
9                 return -1;
10            gTable[idx] = 1;
11            break;
12
13        default:
14            return -1;
15    }
```

Fuzzing Drivers

Inputs:

Cmd: 43256
Param: 65443

Cmd: 1337
Param: 123654

Cmd: 1337
Param: → 4

Cmd: 1337
Param: → 500

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&idx, (void *)param, sizeof(int)) != 0)
9                 return -1;
10            gTable[idx] = 1;
11            break;
12
13        default:
14            return -1;
15    }
```

Fuzzing Drivers

Inputs:

Cmd: 43256
Param: 65443

Cmd: 1337
Param: 123654

Cmd: 1337
Param: → 4

Cmd: 1337
Param: → 500

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5
6     switch(cmd) {
7         case 1337:
8             if (copy from user(&idx, (void *)param, sizeof(int)) != 0)
9                 return -1;
10            gTable[idx] = 1;
11            break;
12
13        default:
14            return -1;
15    }
```

Fuzzing Drivers

Inputs:

Cmd: 43256
Param: 65443

Cmd: 1337
Param: 123654

Cmd: 1337
Param: → 4

Cmd: 1337
Param: → 500

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5
6     switch(cmd) {
7         case 1337:
8             if (copy from user(&idx, (void *)param, sizeof(int)) != 0)
9                 return -1;
10            gTable[idx] = 1;
11            break;
12
13        default:
14            return -1;
15    }
```

Fuzzing Drivers

Inputs:

Cmd: 43256
Param: 65443

Cmd: 1337
Param: 123654

Cmd: 1337
Param: → 4

Cmd: 1337
Param: → 500

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&idx, (void *)param, sizeof(int)) != 0)
9                 return -1;
10            gTable[idx] = 1;
11            break;
12
13        default:
14            return -1;
15    }
```

Fuzzing Drivers

Inputs:

Cmd: 43256
Param: 65443

Cmd: 1337
Param: 123654

Cmd: 1337
Param: → 4

Cmd: 1337
Param: → 500

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&idx, (void *)param, sizeof(int)) != 0)
9                 return -1;
10            gTable[idx] = 1;
11            break;
12
13        default:
14            return -1;
15    }
```

Fuzzing Drivers

Inputs:

Cmd: 43256
Param: 65443

Cmd: 1337
Param: 123654

Cmd: 1337
Param: → 4

Cmd: 1337
Param: → 500

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&idx, (void *)param, sizeof(int)) != 0)
9                 return -1;
10            gTable[idx] = 1;
11            break;
12
13        default:
14            return -1;
15    }
```

Fuzzing Drivers

Inputs:

Cmd: 43256
Param: 65443

Cmd: 1337
Param: 123654

Cmd: 1337
Param: → 4

Cmd: 1337
Param: → 500

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&idx, (void *)param, sizeof(int)) != 0)
9                 return -1;
10            gTable[idx] = 1;
11            break;
12
13         default:
14             return -1;
15     }
```

Fuzzing Drivers

```
typedef enum {  
    WRITE = 77,  
    CLEAR = 78  
} my_enum;
```

```
typedef struct {  
    my_enum type;  
    int idx;  
    int val;  
} foo_t;
```

Fuzzing Drivers

Inputs:

Cmd: 1337

Param: → [564645, 0...]

Cmd: 1337

Param: → [77, 321...]

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5     foo_t foo;
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&foo, (void *)param, sizeof(foo_t)) != 0)
9                 return -1;
10
11             /* WRITE */
12             if (foo.type == 77)
13                 gTable[foo.idx] = foo.val;
14
15             /* CLEAR */
16             else if (foo.type == 78)
17                 kmemset(gTable, 0, sizeof(gTable));
18
19             else
20                 return -1;
21
22             break;
23
24             default:
25                 return -1;
26     }
27 }
```

Fuzzing Drivers

Inputs:

Cmd: 1337

Param: → [564645, 0...]

Cmd: 1337

Param: → [77, 321...]

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5     foo_t foo;
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&foo, (void *)param, sizeof(foo_t)) != 0)
9                 return -1;
10
11             /* WRITE */
12             if (foo.type == 77)
13                 gTable[foo.idx] = foo.val;
14
15             /* CLEAR */
16             else if (foo.type == 78)
17                 kmemset(gTable, 0, sizeof(gTable));
18
19             else
20                 return -1;
21
22             break;
23
24             default:
25                 return -1;
26     }
27 }
```

Fuzzing Drivers

Inputs:

Cmd: 1337
Param: → [564645, 0...]

Cmd: 1337
Param: → [77, 321...]

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5     foo_t foo;
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&foo, (void *)param, sizeof(foo_t)) != 0)
9                 return -1;
10
11             /* WRITE */
12             if (foo.type == 77)
13                 gTable[foo.idx] = foo.val;
14
15             /* CLEAR */
16             else if (foo.type == 78)
17                 kmemset(gTable, 0, sizeof(gTable));
18
19             else
20                 return -1;
21
22             break;
23
24             default:
25                 return -1;
26     }
27 }
```

Fuzzing Drivers

Inputs:

Cmd: 1337
Param: → [564645, 0...]

Cmd: 1337
Param: → [77, 321...]

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5     foo_t foo;
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&foo, (void *)param, sizeof(foo_t)) != 0)
9                 return -1;
10
11             /* WRITE */
12             if (foo.type == 77)
13                 gTable[foo.idx] = foo.val;
14
15             /* CLEAR */
16             else if (foo.type == 78)
17                 kmemset(gTable, 0, sizeof(gTable));
18
19             else
20                 return -1;
21
22             break;
23
24             default:
25                 return -1;
26     }
27 }
```

Fuzzing Drivers

Inputs:

Cmd: 1337
Param: → [564645, 0...]

Cmd: 1337
Param: → [77, 321...]

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5     foo_t foo;
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&foo, (void *)param, sizeof(foo_t)) != 0)
9                 return -1;
10
11             /* WRITE */
12             if (foo.type == 77)
13                 gTable[foo.idx] = foo.val;
14
15             /* CLEAR */
16             else if (foo.type == 78)
17                 kmemset(gTable, 0, sizeof(gTable));
18
19             else
20                 return -1;
21
22             break;
23
24             default:
25                 return -1;
26     }
27 }
```

Fuzzing Drivers

Inputs:

Cmd: 1337

Param: → [564645, 0...]

Cmd: 1337

Param: → [77, 321...]

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5     foo_t foo;
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&foo, (void *)param, sizeof(foo_t)) != 0)
9                 return -1;
10
11             /* WRITE */
12             if (foo.type == 77)
13                 gTable[foo.idx] = foo.val;
14
15             /* CLEAR */
16             else if (foo.type == 78)
17                 kmemset(gTable, 0, sizeof(gTable));
18
19             else
20                 return -1;
21
22             break;
23
24             default:
25                 return -1;
26     }
27 }
```

Fuzzing Drivers

Inputs:

Cmd: 1337
Param: → [564645, 0...]

Cmd: 1337
Param: → [77, 321...]

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5     foo_t foo;
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&foo, (void *)param, sizeof(foo_t)) != 0)
9                 return -1;
10
11             /* WRITE */
12             if (foo.type == 77)
13                 gTable[foo.idx] = foo.val;
14
15             /* CLEAR */
16             else if (foo.type == 78)
17                 kmemset(gTable, 0, sizeof(gTable));
18
19             else
20                 return -1;
21
22             break;
23
24             default:
25                 return -1;
26     }
27 }
```

Fuzzing Drivers

Inputs:

Cmd: 1337

Param: → [564645] 0...]

Cmd: 1337

Param: → [77, 321...]

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5     foo_t foo;
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&foo, (void *)param, sizeof(foo_t)) != 0)
9                 return -1;
10
11             /* WRITE */
12             if (foo.type == 77)
13                 gTable[foo.idx] = foo.val;
14
15             /* CLEAR */
16             else if (foo.type == 78)
17                 kmemset(gTable, 0, sizeof(gTable));
18
19             else
20                 return -1;
21
22             break;
23
24             default:
25                 return -1;
26     }
27 }
```

Fuzzing Drivers

Inputs:

Cmd: 1337

Param: → [564645] 0...]

Cmd: 1337

Param: → [77, 321...]

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5     foo_t foo;
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&foo, (void *)param, sizeof(foo_t)) != 0)
9                 return -1;
10
11             /* WRITE */
12             if (foo.type == 77)
13                 gTable[foo.idx] = foo.val;
14
15             /* CLEAR */
16             else if (foo.type == 78)
17                 kmemset(gTable, 0, sizeof(gTable));
18
19             else
20                 return -1;
21
22             break;
23
24         default:
25             return -1;
26     }
27 }
```

Fuzzing Drivers

Inputs:

Cmd: 1337
Param: → [564645, 0...]

Cmd: 1337
Param: → [77, 321...]

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5     foo_t foo;
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&foo, (void *)param, sizeof(foo_t)) != 0)
9                 return -1;
10
11             /* WRITE */
12             if (foo.type == 77)
13                 gTable[foo.idx] = foo.val;
14
15             /* CLEAR */
16             else if (foo.type == 78)
17                 kmemset(gTable, 0, sizeof(gTable));
18
19             else
20                 return -1;
21
22             break;
23
24         default:
25             return -1;
26     }
27 }
```

Fuzzing Drivers

Inputs:

Cmd: 1337

Param: → [564645, 0...]

Cmd: 1337

Param: → [77, 321...]

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5     foo_t foo;
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&foo, (void *)param, sizeof(foo_t)) != 0)
9                 return -1;
10
11             /* WRITE */
12             if (foo.type == 77)
13                 gTable[foo.idx] = foo.val;
14
15             /* CLEAR */
16             else if (foo.type == 78)
17                 kmemset(gTable, 0, sizeof(gTable));
18
19             else
20                 return -1;
21
22             break;
23
24             default:
25                 return -1;
26     }
27 }
```

Fuzzing Drivers

Inputs:

Cmd: 1337

Param: → [564645, 0...]

Cmd: 1337

Param: → [77, 321...]

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5     foo_t foo;
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&foo, (void *)param, sizeof(foo_t)) != 0)
9                 return -1;
10
11             /* WRITE */
12             if (foo.type == 77)
13                 gTable[foo.idx] = foo.val;
14
15             /* CLEAR */
16             else if (foo.type == 78)
17                 kmemset(gTable, 0, sizeof(gTable));
18
19             else
20                 return -1;
21
22             break;
23
24             default:
25                 return -1;
26     }
27 }
```

Fuzzing Drivers

Inputs:

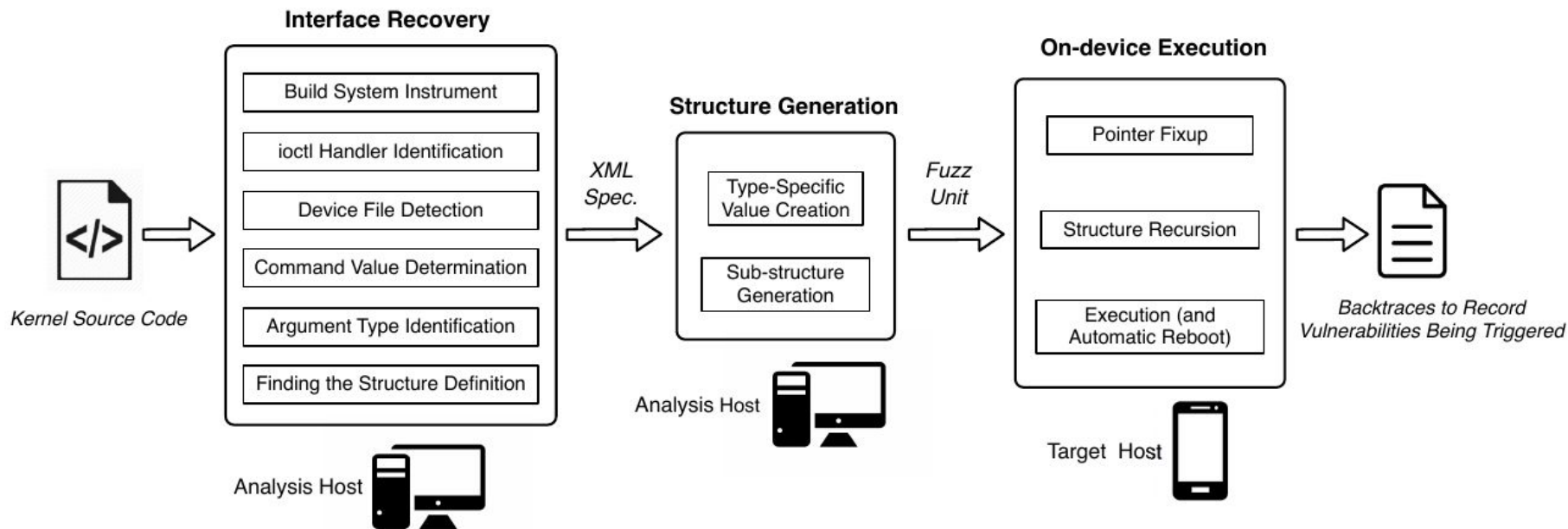
Cmd: 1337
Param: → [564645, 0...]

Cmd: 1337
Param: → [77, 321...]

```
1 int gTable[128];
2
3 ioctl_handler(struct file *pFile, unsigned int cmd, unsigned long param) {
4     int idx;
5     foo_t foo;
6     switch(cmd) {
7         case 1337:
8             if (copy_from_user(&foo, (void *)param, sizeof(foo_t)) != 0)
9                 return -1;
10
11             /* WRITE */
12             if (foo.type == 77)
13                 gTable[foo.idx] = foo.val;
14
15             /* CLEAR */
16             else if (foo.type == 78)
17                 kmemset(gTable, 0, sizeof(gTable));
18
19             else
20                 return -1;
21
22             break;
23
24             default:
25                 return -1;
26     }
27 }
```

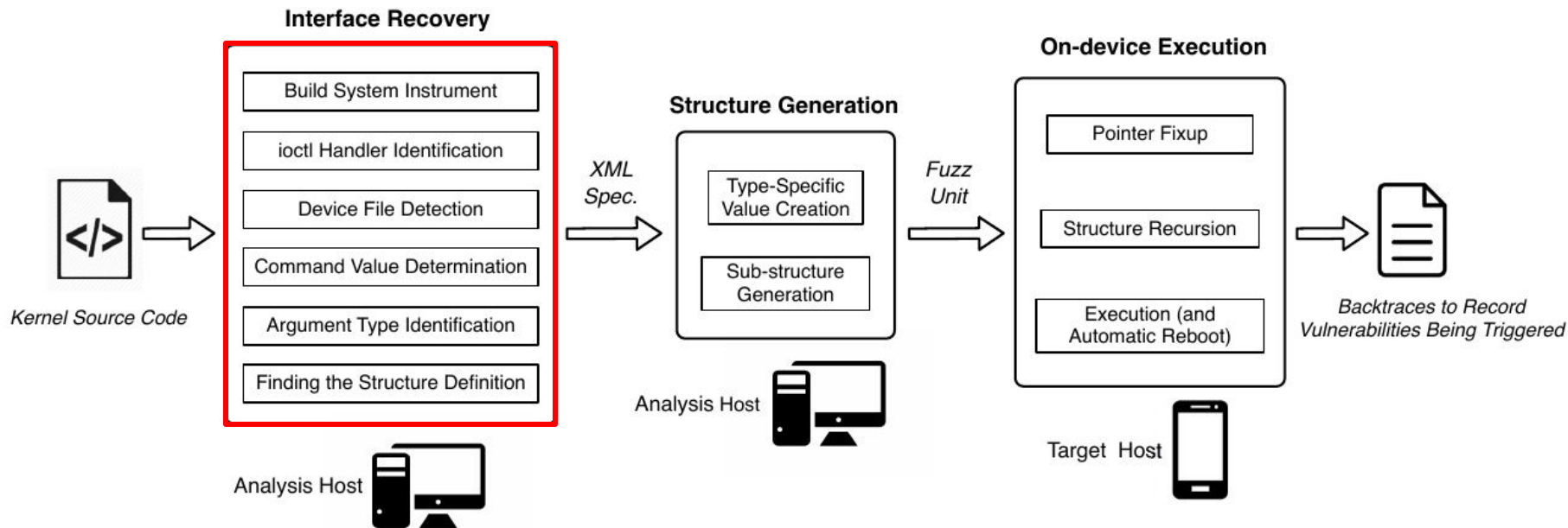
DIFUZE

- Our system that attempts to solve this problem *automatically*.



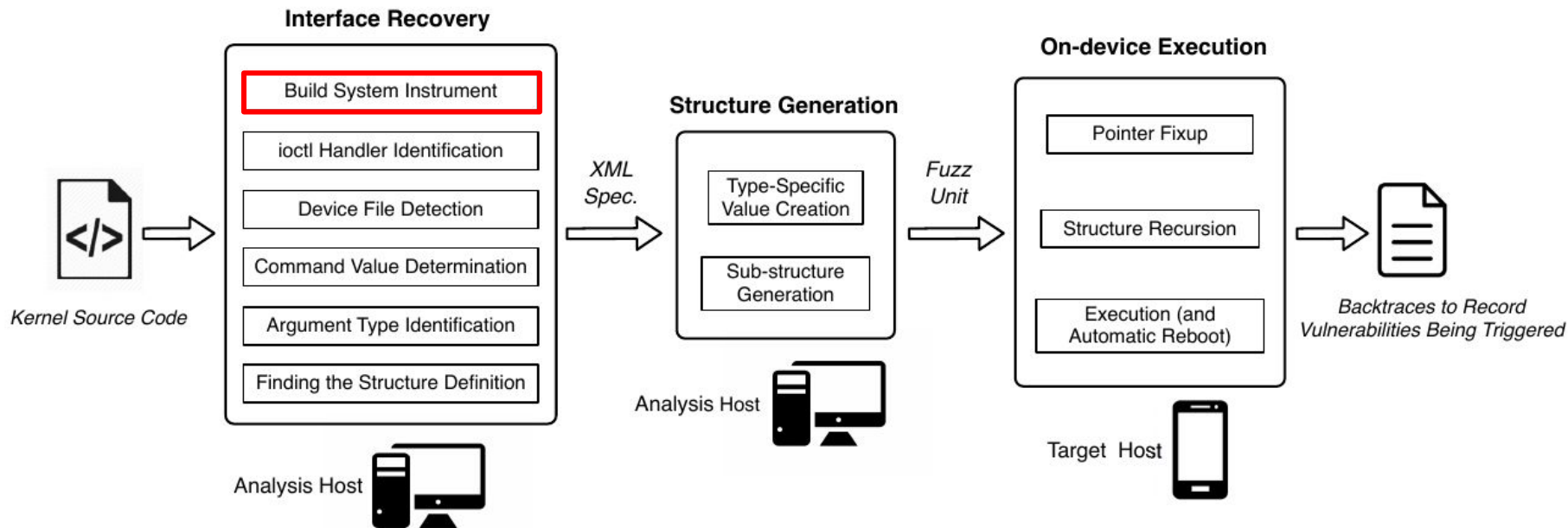
DIFUZE

- Our system that attempts to solve this problem *automatically*.



DIFUZE

- Our system that attempts to solve this problem *automatically*.

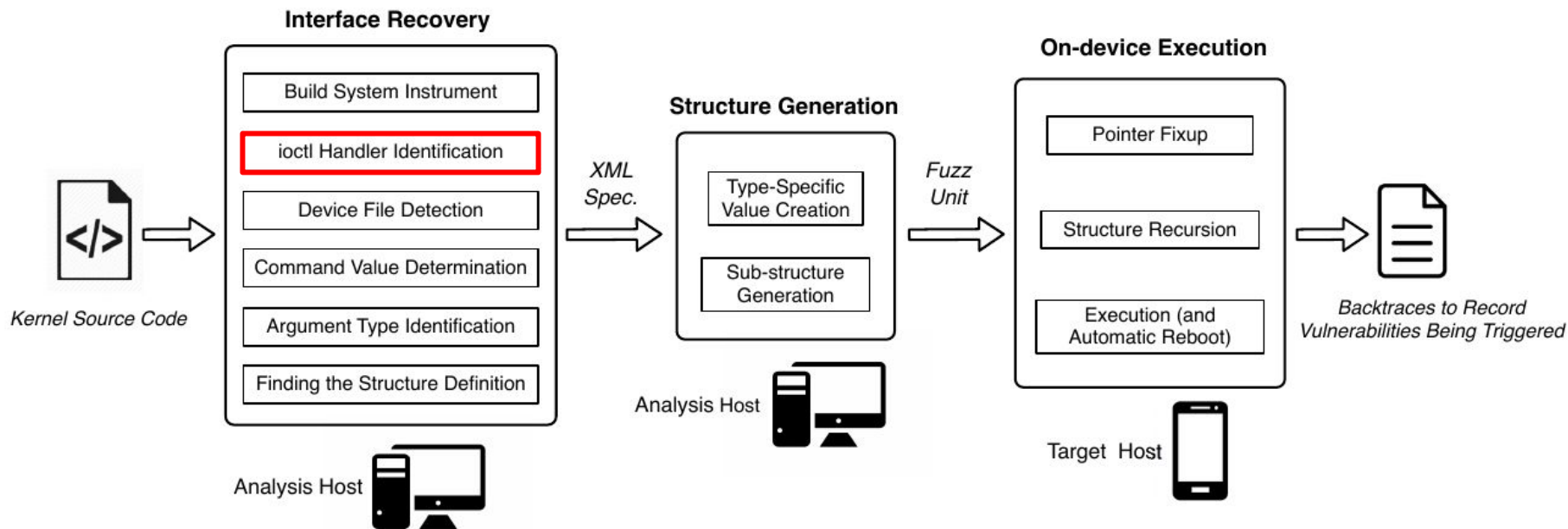


Build Process Instrumentation

- For the majority of our analyses we use LLVM
- Want LLVM bitcode
- Linux Kernel uses GCC
- Build the kernel normally with the provided makefile
- Capture output (build commands)
- Transform GCC commands to Clang commands
- Link bitcode files

DIFUZE

- Our system that attempts to solve this problem *automatically*.



“Operations” Structure Recovery

- List of “operation” structures
- Grep for definitions in includes
- Use c2xml on relevant header files
- Recover .unlocked_ioctl offset

ioctl Handler Recovery

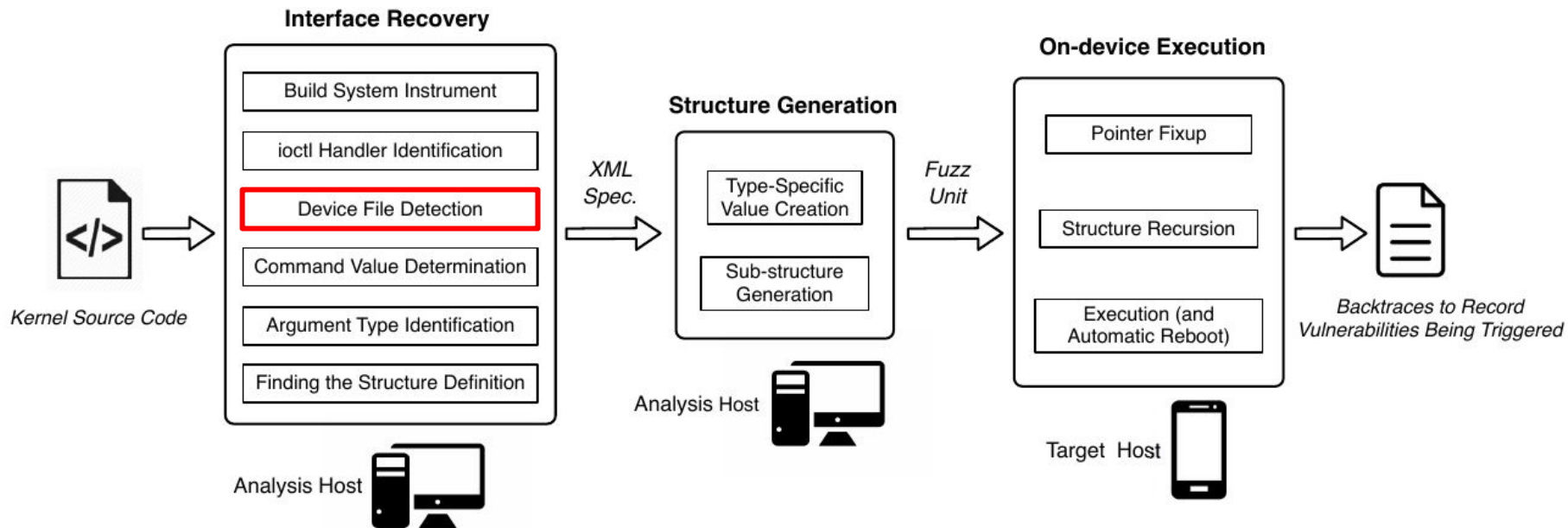
- Run analyses on the linked driver bitcode files and search for uses of the “operations” structures.
- Look for assignments at the offsets recovered
- If found, we’ve found the ioctl handler for that driver, and we store the name.

```
static const struct file_operations IspFile0per = {  
    .owner = THIS_MODULE,  
    .open = ISP_open,  
    .release = ISP_release,  
    .mmap = ISP_mmap,  
    .unlocked_ioctl = ISP_ioctl  
};
```

```
static const struct file_operations IspFileOper = {  
    .owner = THIS_MODULE,  
    .open = ISP_open,  
    .release = ISP_release,  
    .mmap = ISP_mmap,  
    .unlocked_ioctl = ISP_ioctl  
};
```

DIFUZE

- Our system that attempts to solve this problem *automatically*.



Device Name Recovery

```
root@F3116:/ (#) ls -l /dev/ | grep "camera-isp"  
crw-rw---- system= fcamera; 243, 0 2017-05-18 08:43 camera-isp
```

- Where is this device file and what is it called?
- Recall our journey through the registration of a device.
- If there's a device file created, it must be associated with the operations structure we've found!
- We use analysis to recover static names

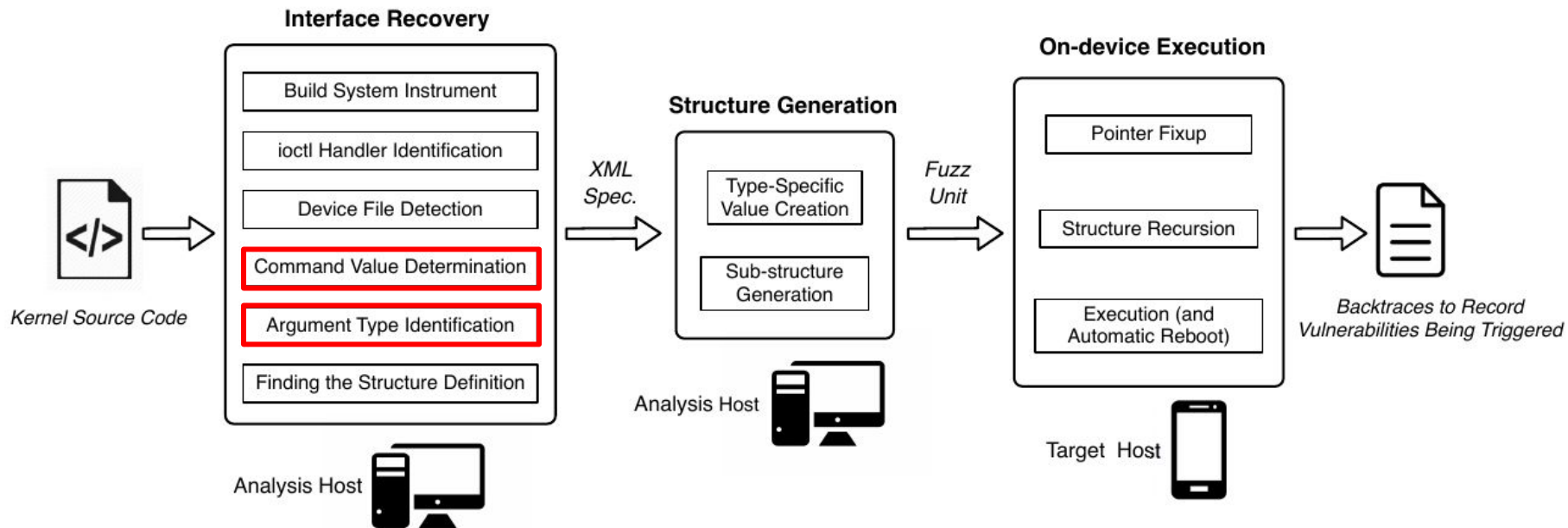
Dynamic Names

- Unfortunately the names aren't always static.
- We miss dynamic names and must fallback to manual analysis.

```
VOS_INT __init RNIC_InitNetCard(VOS_VOID)
{
    ...
    snprintf(pstDev->name, sizeof(pstDev->name),
        "%S%S",
        RNIC_DEV_NAME_PREFIX,
        g_astRnicManageTbl[ucIndex].pucRnicNetCardName);
    ...
}
```

DIFUZE

- Our system that attempts to solve this problem *automatically*.



Command Value + Type Recovery

- Found the ioctl handler function, we run LLVM analyses on the function.
- Know the arguments of interest.
- We search for equality comparisons on “command”, and keep track of the constraints on a given path.
- Search for copy_from_user using “param”
- If found, we find the type associated with the first argument.
- Follow functions that are passed “param” and/or “command”

```
1 static long ISP_ioctl(struct file *pFile, unsigned int Cmd, unsigned long Param)
2 {
3     ...
4     switch (Cmd) {
5         ...
6         case ISP_BUFFER_CTRL:
7             Ret = ISP_Buf_CTRL_FUNC(Param);
8             break;
9         ...
10    }
11 ...
12 }
13
14 static long ISP_Buf_CTRL_FUNC(unsigned long Param)
15 {
16     ...
17     ISP_BUFFER_CTRL_STRUCT rt_buf_ctrl;
18     ...
19     if (copy_from_user(&rt_buf_ctrl, (void __user *)Param, sizeof(ISP_BUFFER_CTRL_STRUCT)) == 0) {
20         ...
21     }
22     ...
23 }
```

```
1 static long ISP_ioctl(struct file *pFile, unsigned int Cmd, unsigned long Param)
2 {
3     ...
4     switch (Cmd) {
5         ...
6         case ISP_BUFFER_CTRL:
7             Ret = ISP_Buf_CTRL_FUNC(Param);
8             break;
9         ...
10    }
11 ...
12 }
13
14 static long ISP_Buf_CTRL_FUNC(unsigned long Param)
15 {
16     ...
17     ISP_BUFFER_CTRL_STRUCT rt_buf_ctrl;
18     ...
19     if (copy_from_user(&rt_buf_ctrl, (void __user *)Param, sizeof(ISP_BUFFER_CTRL_STRUCT)) == 0) {
20         ...
21     }
22     ...
23 }
```

```
1 static long ISP_ioctl(struct file *pFile, unsigned int Cmd, unsigned long Param)
2 {
3     ...
4     switch (Cmd) {
5         ...
6         case ISP_BUFFER_CTRL:
7             Ret = ISP_Buf_CTRL_FUNC(Param);
8             break;
9         ...
10    }
11 ...
12 }
13
14 static long ISP_Buf_CTRL_FUNC(unsigned long Param)
15 {
16     ...
17     ISP_BUFFER_CTRL_STRUCT rt_buf_ctrl;
18     ...
19     if (copy_from_user(&rt_buf_ctrl, (void __user *)Param, sizeof(ISP_BUFFER_CTRL_STRUCT)) == 0) {
20         ...
21     }
22     ...
23 }
```

```
1 static long ISP_ioctl(struct file *pFile, unsigned int Cmd, unsigned long Param)
2 {
3     ...
4     switch (Cmd) {
5         ...
6         case ISP_BUFFER_CTRL:
7             Ret = ISP_Buf_CTRL_FUNC(Param);
8             break;
9         ...
10    }
11 ...
12 }
13
14 static long ISP_Buf_CTRL_FUNC(unsigned long Param)
15 {
16     ...
17     ISP_BUFFER_CTRL_STRUCT rt_buf_ctrl;
18     ...
19     if (copy_from_user(&rt_buf_ctrl, (void __user *)Param, sizeof(ISP_BUFFER_CTRL_STRUCT)) == 0) {
20         ...
21     }
22     ...
23 }
```

Command Value: **ISP_BUFFER_CTRL**

```
1 static long ISP_ioctl(struct file *pFile, unsigned int Cmd, unsigned long Param)
2 {
3     ...
4     switch (Cmd) {
5         ...
6         case ISP_BUFFER_CTRL:
7             Ret = ISP_Buf_CTRL_FUNC(Param);
8             break;
9         ...
10    }
11 ...
12 }
13
14 static long ISP_Buf_CTRL_FUNC(unsigned long Param)
15 {
16     ...
17     ISP_BUFFER_CTRL_STRUCT rt_buf_ctrl;
18     ...
19     if (copy_from_user(&rt_buf_ctrl, (void __user *)Param, sizeof(ISP_BUFFER_CTRL_STRUCT)) == 0) {
20         ...
21     }
22     ...
23 }
```

Command Value: **ISP_BUFFER_CTRL**

```
1 static long ISP_ioctl(struct file *pFile, unsigned int Cmd, unsigned long Param)
2 {
3     ...
4     switch (Cmd) {
5         ...
6         case ISP_BUFFER_CTRL:
7             Ret = ISP_Buf_CTRL_FUNC(Param);
8             break;
9         ...
10    }
11 ...
12 }
13
14 static long ISP_Buf_CTRL_FUNC(unsigned long Param)
15 {
16     ...
17     ISP_BUFFER_CTRL_STRUCT rt_buf_ctrl;
18     ...
19     if (copy_from_user(&rt_buf_ctrl, (void __user *)Param, sizeof(ISP_BUFFER_CTRL_STRUCT)) == 0) {
20         ...
21     }
22     ...
23 }
```

Command Value: **ISP_BUFFER_CTRL**

```

1 static long ISP_ioctl(struct file *pFile, unsigned int Cmd, unsigned long Param)
2 {
3     ...
4     switch (Cmd) {
5         ...
6         case ISP_BUFFER_CTRL:
7             Ret = ISP_Buf_CTRL_FUNC(Param);
8             break;
9         ...
10    }
11 ...
12 }
13
14 static long ISP_Buf_CTRL_FUNC(unsigned long Param)
15 {
16     ...
17     ISP_BUFFER_CTRL_STRUCT rt_buf_ctrl;
18     ...
19     if (copy_from_user(&rt_buf_ctrl, (void __user *)Param, sizeof(ISP_BUFFER_CTRL_STRUCT)) == 0) {
20         ...
21     }
22     ...
23 }

```

Command Value: **ISP_BUFFER_CTRL**

```
1 static long ISP_ioctl(struct file *pFile, unsigned int Cmd, unsigned long Param)
2 {
3     ...
4     switch (Cmd) {
5         ...
6         case ISP_BUFFER_CTRL:
7             Ret = ISP_Buf_CTRL_FUNC(Param);
8             break;
9         ...
10    }
11 ...
12 }
13
14 static long ISP_Buf_CTRL_FUNC(unsigned long Param)
15 {
16     ...
17     ISP_BUFFER_CTRL_STRUCT rt_buf_ctrl;
18     ...
19     if (copy_from_user(&rt_buf_ctrl, (void __user *)Param, sizeof(ISP_BUFFER_CTRL_STRUCT)) == 0) {
20         ...
21     }
22     ...
23 }
```

Command Value: **ISP_BUFFER_CTRL**

```
1 static long ISP_ioctl(struct file *pFile, unsigned int Cmd, unsigned long Param)
2 {
3     ...
4     switch (Cmd) {
5         ...
6         case ISP_BUFFER_CTRL:
7             Ret = ISP_Buf_CTRL_FUNC(Param);
8             break;
9         ...
10    }
11 ...
12 }
13
14 static long ISP_Buf_CTRL_FUNC(unsigned long Param)
15 {
16     ...
17     ISP_BUFFER_CTRL_STRUCT rt_buf_ctrl;
18     ...
19     if (copy_from_user(&rt_buf_ctrl, (void __user *)Param, sizeof(ISP_BUFFER_CTRL_STRUCT)) == 0) {
20         ...
21     }
22     ...
23 }
```

Command Value: **ISP_BUFFER_CTRL**

```
1 static long ISP_ioctl(struct file *pFile, unsigned int Cmd, unsigned long Param)
2 {
3     ...
4     switch (Cmd) {
5         ...
6         case ISP_BUFFER_CTRL:
7             Ret = ISP_Buf_CTRL_FUNC(Param);
8             break;
9         ...
10    }
11 ...
12 }
13
14 static long ISP_Buf_CTRL_FUNC(unsigned long Param)
15 {
16     ...
17     ISP_BUFFER_CTRL_STRUCT rt_buf_ctrl;
18     ...
19     if (copy_from_user(&rt_buf_ctrl, (void __user *)Param, sizeof(ISP_BUFFER_CTRL_STRUCT)) == 0) {
20         ...
21     }
22     ...
23 }
```

Command Value: **ISP_BUFFER_CTRL**

```
1 static long ISP_ioctl(struct file *pFile, unsigned int Cmd, unsigned long Param)
2 {
3     ...
4     switch (Cmd) {
5         ...
6         case ISP_BUFFER_CTRL:
7             Ret = ISP_Buf_CTRL_FUNC(Param);
8             break;
9         ...
10    }
11 ...
12 }
13
14 static long ISP_Buf_CTRL_FUNC(unsigned long Param)
15 {
16     ...
17     ISP_BUFFER_CTRL_STRUCT rt_buf_ctrl;
18     ...
19     if (copy_from_user(&rt_buf_ctrl, (void __user *)Param, sizeof(ISP_BUFFER_CTRL_STRUCT)) == 0) {
20         ...
21     }
22     ...
23 }
```

Command Value: **ISP_BUFFER_CTRL**

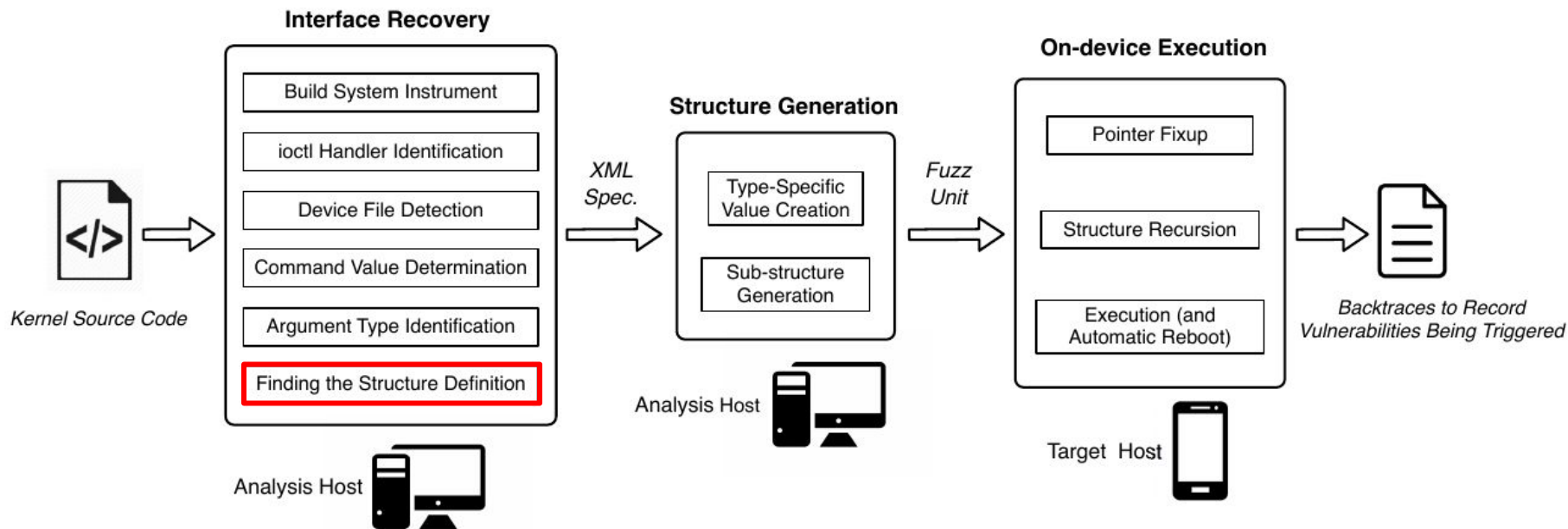
```
1 static long ISP_ioctl(struct file *pFile, unsigned int Cmd, unsigned long Param)
2 {
3     ...
4     switch (Cmd) {
5         ...
6         case ISP_BUFFER_CTRL:
7             Ret = ISP_Buf_CTRL_FUNC(Param);
8             break;
9         ...
10    }
11 ...
12 }
13
14 static long ISP_Buf_CTRL_FUNC(unsigned long Param)
15 {
16     ...
17     ISP_BUFFER_CTRL_STRUCT rt_buf_ctrl;
18     ...
19     if (copy_from_user(&rt_buf_ctrl, (void __user *)Param, sizeof(ISP_BUFFER_CTRL_STRUCT)) == 0) {
20         ...
21     }
22     ...
23 }
```

Command Value: **ISP_BUFFER_CTRL**

Type: **ISP_BUFFER_CTRL_STRUCT**

DIFUZE

- Our system that attempts to solve this problem *automatically*.



Structure Definition Recovery

- Have the GCC build commands
- Know the file we're looking at
- Use GCC command with -E
- Get a HUGE (40k+ line) file which **somewhere** will include the structure definition.
- Run our good friend c2xml on the file and get an equally massive xml file.
- Python passes to extract the struct + struct dependencies, account for padding, recover enum values, etc.

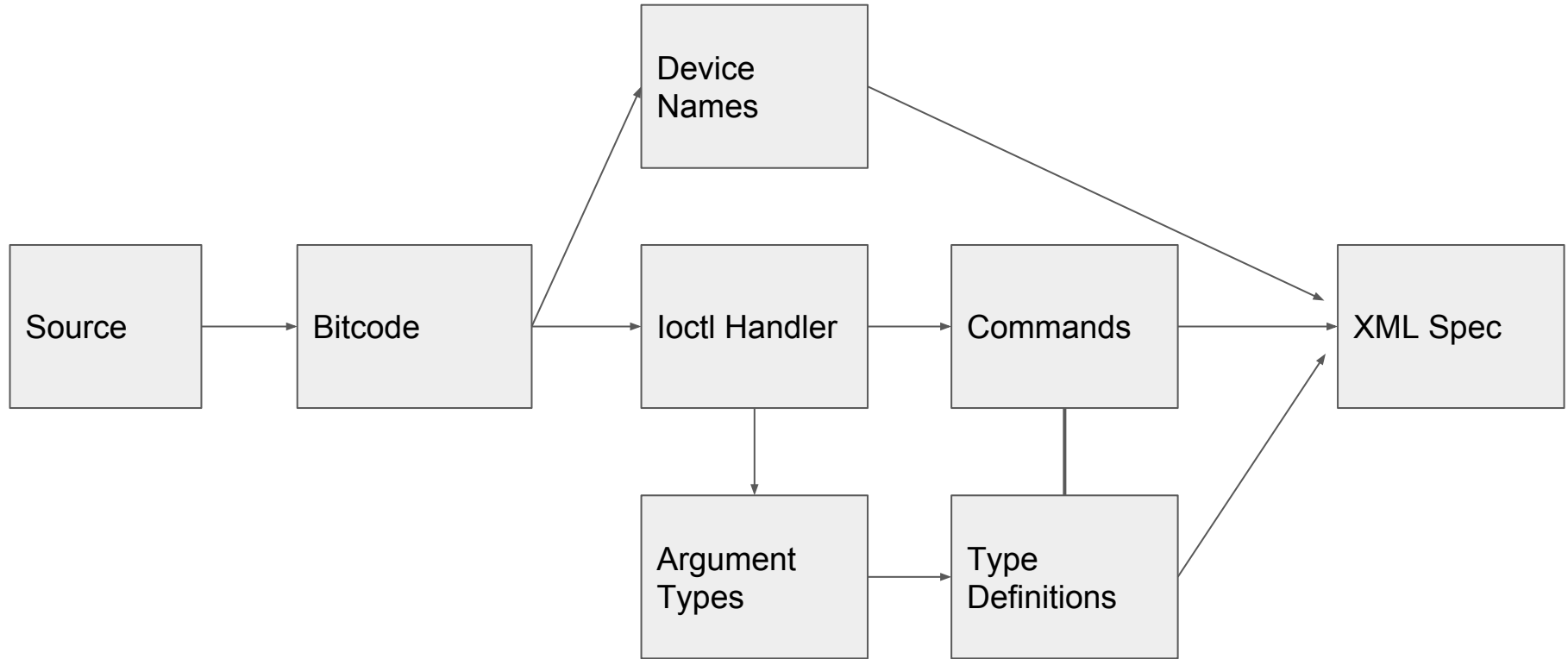
```

bit-size="16" alignment="2" offset="0" base-type="builtin:signed short"/>
14 <symbol type="node" id="_11" ident="u16" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="78" start-col="24" end-line="78" end-col="27"
bit-size="16" alignment="2" offset="0" base-type="builtin:unsigned short"/>
15 <symbol type="node" id="_12" ident="s32" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="80" start-col="20" end-line="80" end-col="23"
bit-size="32" alignment="4" offset="0" base-type="builtin:signed int"/>
16 <symbol type="node" id="_13" ident="u32" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="81" start-col="22" end-line="81" end-col="25"
bit-size="32" alignment="4" offset="0" base-type="builtin:unsigned int"/>
17 <symbol type="node" id="_14" ident="s64" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="83" start-col="26" end-line="83" end-col="29"
bit-size="64" alignment="8" offset="0" base-type="builtin:signed long long"/>
18 <symbol type="node" id="_15" ident="u64" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="84" start-col="28" end-line="84" end-col="31"
bit-size="64" alignment="8" offset="0" base-type="builtin:unsigned long long"/>
19 <symbol type="struct" id="_16" ident="ftrace_branch_data" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="123" start-col="8" end-
line="138" end-col="2" bit-size="320" alignment="8" offset="0">
20 <symbol type="node" id="_17" ident="func" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="124" start-col="13" end-line="124" end-
col="18" bit-size="64" alignment="8" offset="0" base-type="18"/>
21 <symbol type="node" id="_19" ident="file" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="125" start-col="13" end-line="125" end-
col="18" bit-size="64" alignment="8" offset="8" base-type="20"/>
22 <symbol type="node" id="_21" ident="line" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="126" start-col="11" end-line="126" end-
col="15" bit-size="32" alignment="4" offset="16" base-type="builtin:unsigned int"/>
23 <symbol type="node" id="_22" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="137" start-col="3" end-line="137" end-col="3" bit-
size="128" alignment="8" offset="24" base-type="_23"/>
24 </symbol>
25 <symbol type="pointer" id="_18" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="124" start-col="13" end-line="124" end-col="14" bit-
size="64" alignment="8" offset="0" base-type="builtin:char"/>
26 <symbol type="pointer" id="_20" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="125" start-col="13" end-line="125" end-col="14" bit-
size="64" alignment="8" offset="0" base-type="builtin:char"/>
27 <symbol type="union" id="_23" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="127" start-col="8" end-line="137" end-col="3" bit-
size="128" alignment="8" offset="0">
28 <symbol type="node" id="_24" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="131" start-col="4" end-line="131" end-col="4" bit-
size="128" alignment="8" offset="0" base-type="25"/>
29 <symbol type="node" id="_28" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="135" start-col="4" end-line="135" end-col="4" bit-
size="128" alignment="8" offset="8" base-type="29"/>
30 <symbol type="node" id="_32" ident="miss_hit" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="136" start-col="17" end-line="136" end-
col="28" bit-size="128" alignment="8" offset="0" base-type="_33"/>
31 </symbol>
32 <symbol type="struct" id="_25" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="128" start-col="10" end-line="131" end-col="4" bit-
size="128" alignment="8" offset="0" base-type="0">
33 <symbol type="node" id="_26" ident="correct" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="129" start-col="18" end-line="129" end-
col="25" bit-size="64" alignment="8" offset="0" base-type="builtin:unsigned long"/>
34 <symbol type="node" id="_27" ident="incorrect" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="130" start-col="18" end-line="130" end-
col="27" bit-size="64" alignment="8" offset="8" base-type="builtin:unsigned long"/>
35 </symbol>
36 <symbol type="struct" id="_29" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="132" start-col="10" end-line="135" end-col="4" bit-
size="128" alignment="8" offset="0" base-type="0">
37 <symbol type="node" id="_30" ident="miss" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="133" start-col="18" end-line="133" end-
col="22" bit-size="64" alignment="8" offset="0" base-type="builtin:unsigned long"/>
38 <symbol type="node" id="_31" ident="hit" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="134" start-col="18" end-line="134" end-
col="21" bit-size="64" alignment="8" offset="8" base-type="builtin:unsigned long"/>
39 </symbol>
40 <symbol type="array" id="_33" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="136" start-col="25" end-line="136" end-col="28" bit-
size="128" alignment="8" offset="0" base-type="builtin:unsigned long" array-size="2"/>
41 <symbol type="node" id="_34" ident="data_access_exceeds_word_size" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="143" start-col="122"
end-line="147" end-col="1" static="1" inline="1" toplevel="1" bit-size="1" alignment="0" offset="0" base-type="_35"/>
42 <symbol type="function" id="_35" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="143" start-col="151" end-line="147" end-col="1" bit-
size="1" alignment="0" offset="0" base-type="builtin:void"/>
43 <symbol type="node" id="_36" ident="data_access_exceeds_word_size" file="/home/jay/timing/xa/llvm_out/drivers/misc/mediatek/cameraisp/src/mt6755/camera_isp.preprocessed" start-line="149" start-col="122"
end-line="150" end-col="1" static="1" inline="1" toplevel="1" bit-size="1" alignment="0" offset="0" base-type="_37"/>

```

```
1  <DataModel byte_size="136" name="ISP_RT_BUF_INFO_STRUCT" type="struct">
2    <Number name="memID" size="32"/>
3    <Number name="size" size="32"/>
4    <Number name="base_vAddr" size="64"/>
5    <Number name="base_pAddr" size="32"/>
6    <Number name="timeStampS" size="32"/>
7    <Number name="timeStampUs" size="32"/>
8    <Number name="bFilled" size="32"/>
9    <Number name="bProcessRaw" size="32"/>
10   <Block name="image" offset="36" ref="ISP_RT_IMAGE_INFO_STRUCT"/>
11   <Block name="rrzInfo" offset="88" ref="ISP_RT_RRZ_INFO_STRUCT"/>
12   <Block name="dmaoCrop" offset="112" ref="ISP_RT_DMAO_CROPPING_STRUCT"/>
13   <Number name="bDequeued" size="32"/>
14   <Number name="bufIdx" size="32"/>
15 </DataModel>
```

Summary

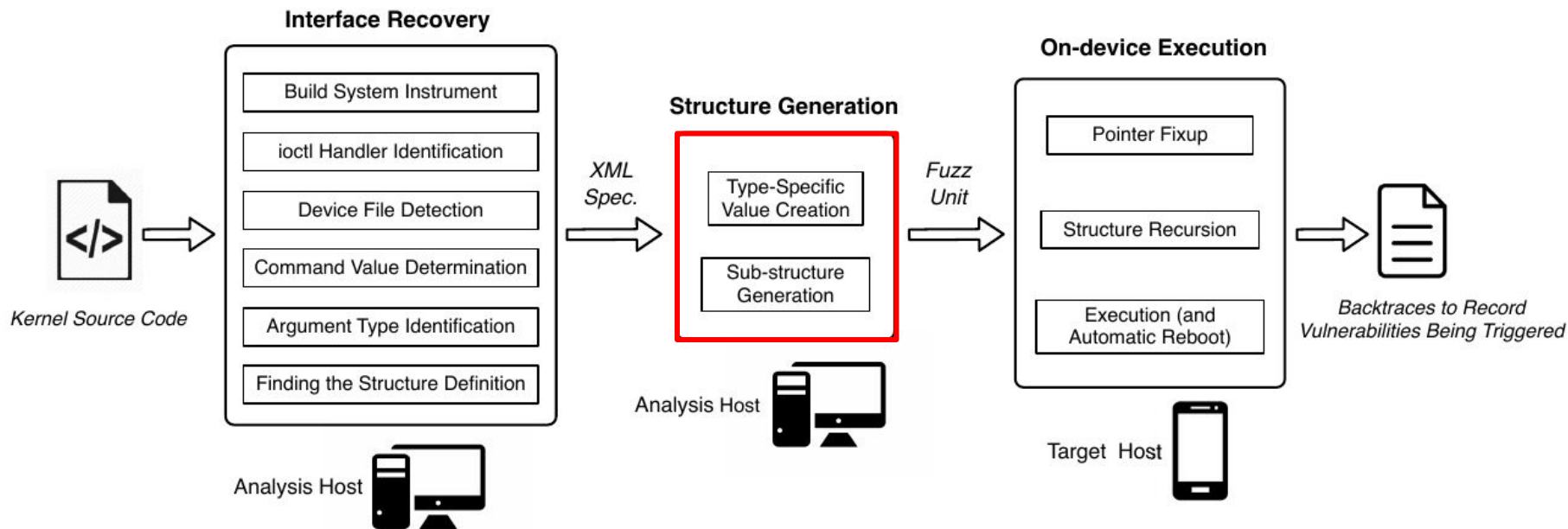


Awesome!

- Now we're entirely **interface aware**.
- We know where the device file is, we know what commands we can pass it, and we know what arguments those commands take.
- Need instances of those structures so that we can actually exercise the behaviour of the device.

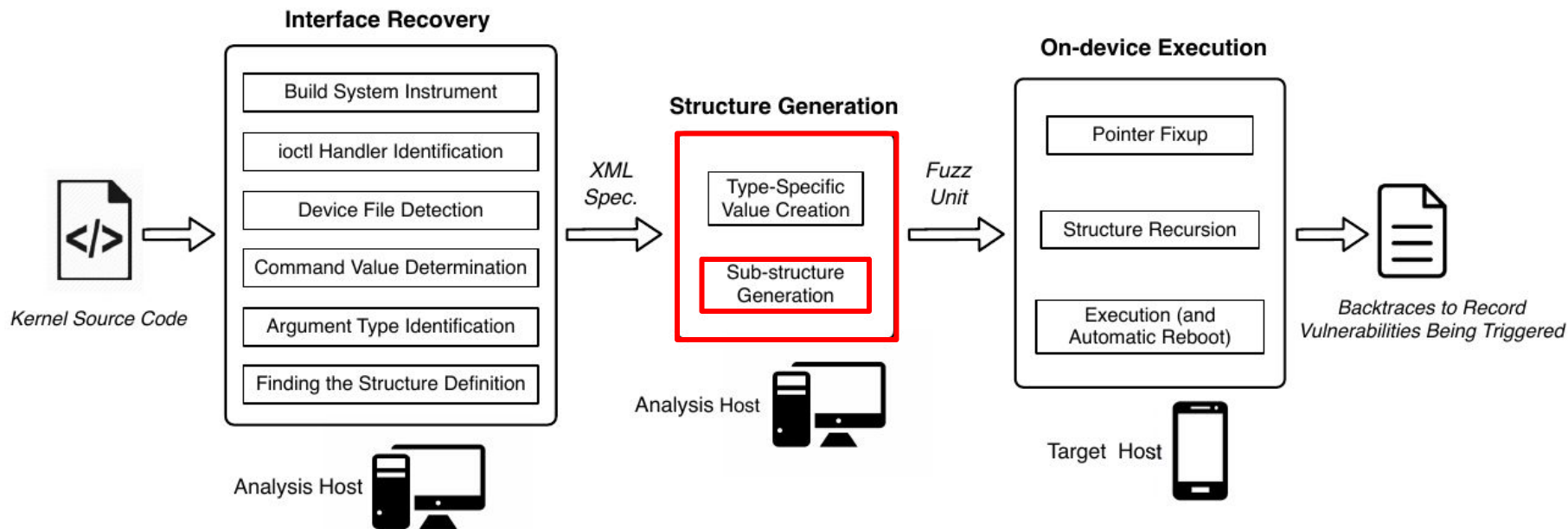
DIFUZE

- Our system that attempts to solve this problem *automatically*.



DIFUZE

- Our system that attempts to solve this problem *automatically*.



Structure Generation

- Harder than it seems
- Embedded structures and pointers

1

0

```
10 typedef enum {
11     ISP_RT_BUF_CTRL_ENQUE, /* 0 */
12     ISP_RT_BUF_CTRL_EXCHANGE_ENQUE, /* 1 */
13     ISP_RT_BUF_CTRL_DEQUE, /* 2 */
14     ISP_RT_BUF_CTRL_IS_RDY, /* 3 */
15     ISP_RT_BUF_CTRL_DMA_EN, /* 4 */
16     ISP_RT_BUF_CTRL_GET_SIZE, /* 5 */
17     ISP_RT_BUF_CTRL_CLEAR, /* 6 */
18     ISP_RT_BUF_CTRL_CUR_STATUS, /* 7 */
19     ISP_RT_BUF_CTRL_MAX /* 8 */
20 } ISP_RT_BUF_CTRL_ENUM;

51 typedef struct {
52     unsigned int memID;
53     unsigned int size;
54     long long base_vAddr;
55     unsigned int base_pAddr;
56     unsigned int timeStamps;
57     unsigned int timeStampUs;
58     unsigned int bFilled;
59     unsigned int bProcessRaw;
60     ISP_RT_IMAGE_INFO_STRUCT image;
61     ISP_RT_RRZ_INFO_STRUCT rrzInfo;
62     ISP_RT_DMAO_CROPPING_STRUCT dmaoCrop;
63     unsigned int bDequeued;
64     signed int bufIdx;
65 } ISP_RT_BUF_INFO_STRUCT;
```

3

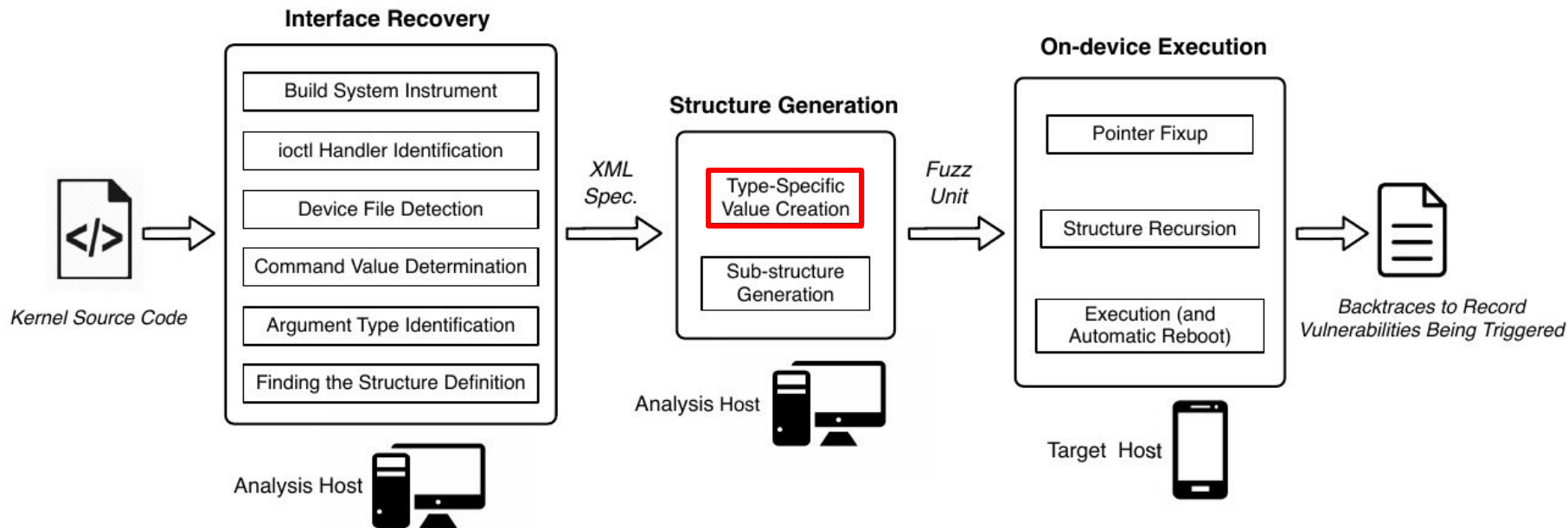
4

2

```
23 typedef enum {
24     _imgi_ = 0,
25     _vipi_, /* 1 */
26     _vip2i_, /* 2 */
27     _vip3i_, /* 3 */
28     _imgo_, /* 4 */
29     _ufdi_, /* 5 */
30     _lcei_, /* 6 */
31     _ufeo_, /* 7 */
32     _rrzo_, /* 8 */
33     _imgo_d_, /* 9 */
34     _rrzo_d_, /* 10 */
35     _img2o_, /* 11 */
36     _img3o_, /* 12 */
37     _img3bo_, /* 13 */
38     _img3co_, /* 14 */
39     _camsv_imgo_, /* 15 */
40     _camsv2_imgo_, /* 16 */
41     _mfbo_, /* 17 */
42     _feo_, /* 18 */
43     _wrot_, /* 19 */
44     _wdma_, /* 20 */
45     _jpeg_, /* 21 */
46     _venc_stream_, /* 21 */
47     _rt_dma_max_ /* 22 */
48 } _isp_dma_enum_;
```

DIFUZE

- Our system that attempts to solve this problem *automatically*.

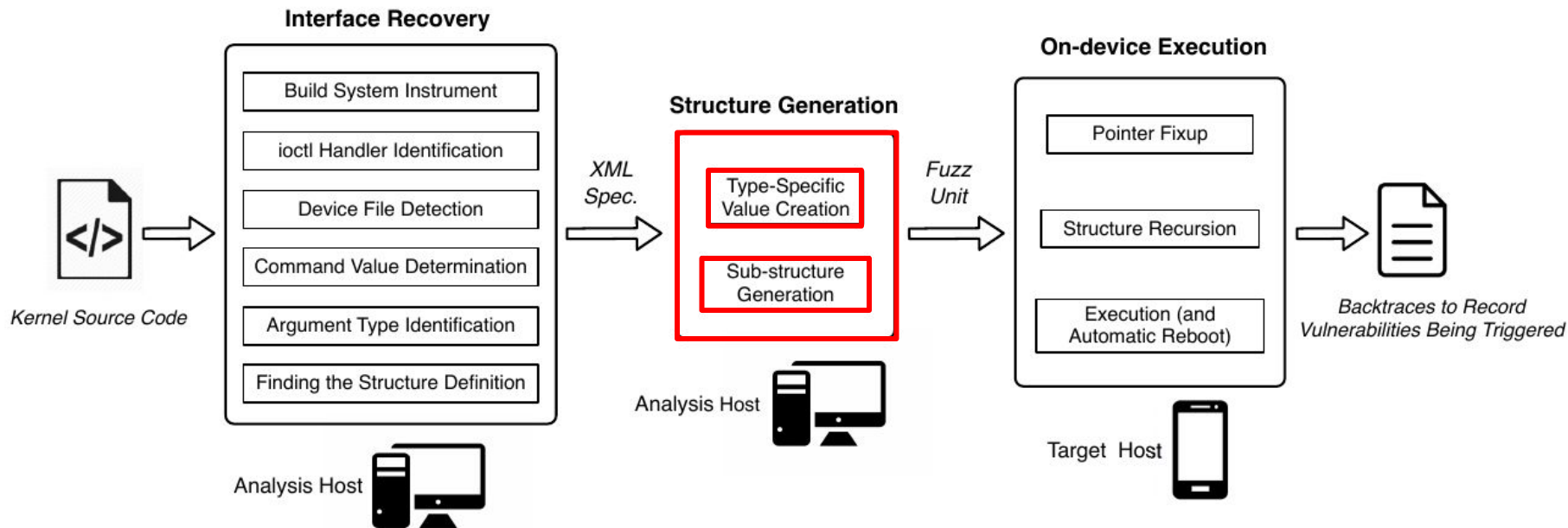


Type Specific Value Generation

- Generate value(s) for each field in the structure
 - Note that since we now have the type information, we can be intelligent about this!

DIFUZE

- Our system that attempts to solve this problem *automatically*.



MangoFuzz

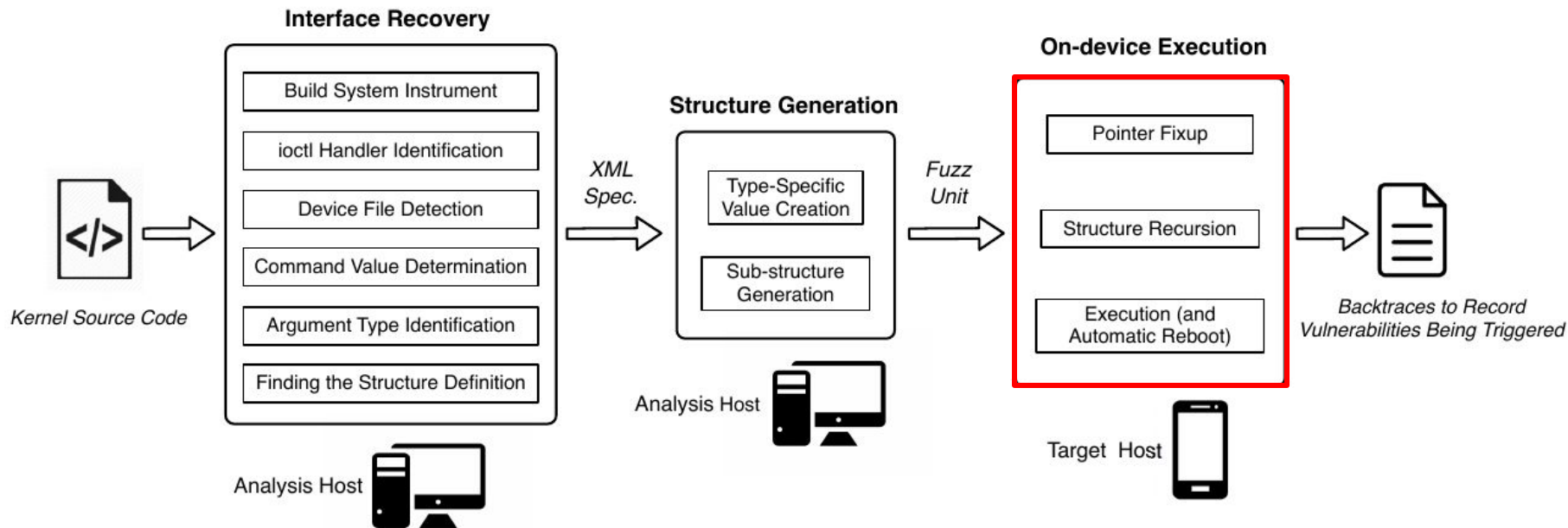
- Written in Python
- Consumes XML spec file(s)
- XML spec files detail the interface information (device path, ioctl command ID, target argument, and argument definition)
- Outputs binary blobs, an array of mappings (if needed), and interface info

XML Spec (jpit)

```
1 <Mango author="jay` bot" description="autogenerated jpit" version="1.0">
2
3   <Config>
4     <devname value="/dev/camera-isp"/>
5     <ioctl_id value="2148559647"/>
6     <target_struct value="ISP_REGISTER_USERKEY_STRUCT"/>
7   </Config>
8
9   <DataModel byte_size="16" name="ISP_REGISTER_USERKEY_STRUCT" type="struct">
10     <Number name="userKey" size="32"/>
11     <String length="4" name="padding256"/>
12     <Pointer base="char" elem_size="1" length="8" name="userName" offset="8" ptr_depth="1" ptr_to="String"/>
13   </DataModel>
14
15 </Mango>
```

DIFUZE

- Our system that attempts to solve this problem *automatically*.



On Device Execution

- When it comes time to actually call `ioctl()`, it needs to be done on the device itself.
- Must be an actual execution component running on the device.
- We connect the analysis host and the device via ADB (Android Debug Bridge).

On Device Execution

- The executor runs on the device and listens for data that will be sent by the fuzzer component
- At this point, it will map the binary data into memory, and do the necessary pointer fixups.
- It will then open the device file specified, and call `ioctl()` with the command value sent, and the now memory mapped argument/structure

How do we detect a bug?

- Device reboots
- the kernel backtrace/oops is saved in
/sys/fs/pstore/console-ramoops
- We use this to triage crashes

Time To Test!

<u>Manufacturer</u>	<u>Device</u>	<u>Chipset</u>
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

Device Name Recovery

<u>ioctl Handlers</u>	<u>Device Names Automatically Identified</u>
789	469

- We automatically identify device names for roughly 60% of the ioctl handlers.
- Most of these misses come from mainline drivers, which have a tendency to use dynamic names.

Type + Command ID Recovery

<u>Command ID's Recovered</u>	<u>User Argument Types</u>			
	Long/CTU	Scalar	Struct	Struct w/ Pointers
3565	1688	526	961	390

- For 47% of the commands, the user argument is used either as C Long, or is used as an address for copy_to_user, in these cases, no type recovery is needed.
- For the rest (53%), the user argument should be a pointer.
- Command ID recovery. Random sample verification of 5 ioctls for each phone (35 total handlers). 90% accuracy.

Fuzzing Evaluation

- 4 variants
- Syzkaller with:
 - Extracted Device Path (PATH)
 - Extracted Device Path and ioctl Numbers (PATH+NUM)
 - Extracted Device Path, ioctl Numbers, and Structures (DIFUZE^S)
- MangoFuzz
 - Extracted Device Path, ioctl Numbers, and Structures (DIFUZE^M)

Fuzzing Results

	PATH	PATH+NUM	DIFUZE ^s	DIFUZE ^m	Total Unique
E9 Plus	0	4	6	6	6
Galaxy S6	-	-	-	0	0
Honor 8	0	1	2	2	2
One M9	0	3	3	2	3
P9 Lite	0	2	5	5	6
Pixel	1	2	5	3	5
Xperia XA	2	10	13	12	14
Total	3	22	34	30	36

Some Fun Bugs



M4U Out Of Bounds Write

```
1 int config_mau(M4U_MAU_STRUCT mau)
2 {
3     int free_id = -1;
4     for (i = 0; i < M4U0_MAU_NR; i++) {
5         if (0 != gM4u0_mau[i].Enabled) {
6             // Start Monitor
7         } else {
8             free_id = i;
9         }
10    }
11    ...
12    if (free_id == -1) {
13        if (mau.force == 0)
14            return -1;
15    }
16    else {
17        free_id = gMAU_candidate_id;
18        ...
19    }
20
21    gM4u0_mau[free_id].Enabled = 1;
22    gM4u0_mau[free_id].MVASStart = MVASStart;
23    gM4u0_mau[free_id].MVAEnd = MVAEnd;
24    gM4u0_mau[free_id].port = mau.port;
25 }
```

free_id set to -1

Incorrect else and free_id is still -1

Out of Bounds write to index -1

BUG() - CVE-2017-0636

Saw the following in the last kmsg:

```
"Unable to handle kernel paging request at virtual  
address 0000dead"
```

BUG() - CVE-2017-0636

```
1 #define BUG() do { \  
2     printk("BUG: failure at %s:%d/%s()!\n", \  
3         __FILE__, __LINE__, __func__); \  
4     *((unsigned *)0xdead) = 0x0aee; \  
5     unreachable(); \  
6 } while (0)
```

BUG() - CVE-2017-0636

`mmap(0xd000, ...) = 0xd000`

Bypass assert and trigger memory corruption

CVE-2017-15307

- Suddenly, fuzzer could no longer find device
- Serial Number had changed to “^Ri£jDO>I”

Conclusion

- Driver fuzzing can yield a lot of bugs
- Modelling the interface correctly is important
- DIFUZE automatically extracts this info to make fuzzing easy

Questions?