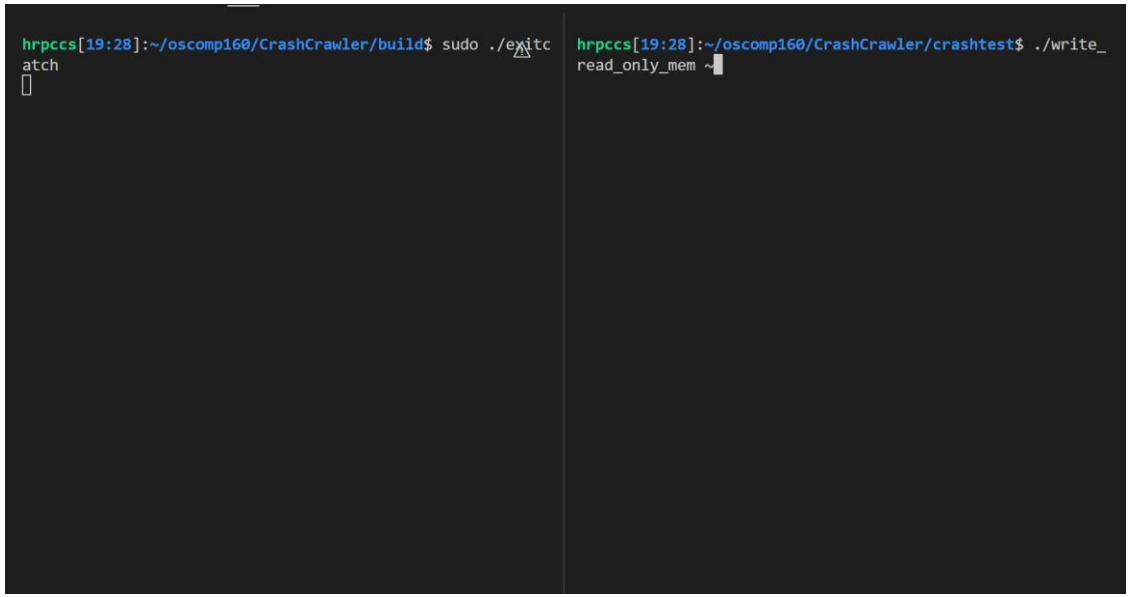

组件部署阶段 → 进程崩溃信息收集 → 信息整合与处理

加载eBPF程序

```
exitcatch_bpf__open();
exitcatch_bpf__load();
exitcatch_bpf__attach();
```

从perf event或ringbuffer中获取信息

```
static int handle_event(void *ctx, void *data, size_t data_sz){
    const struct event *e = data;
    /*...*/
    //get stack from maps
    int ret = bpf_map__lookup_elem(
        skel->maps.map_stack_traces,
        &e->stack_id,
        sizeof(unsigned int),
        &stacks,
        VALUESIZE,
        0);
    /*...*/
    //event handling
    /*...*/
    return 0;
}
```

用户态

Helper Function

内核态

sched_process_exit

挂载点触发bpf程序

```
// the path in the debugfs, and debugfs need to be mounted
SEC("tp/sched/sched_process_exit")
int trace_event_raw_sched_process_exit(struct trace_event_raw_sched_process *ctx)
{
    struct task_struct *task;
    struct event *e;

    u64 pid = bpf_get_current_pid_tgid();
    u32 tid = (u32)pid;
    pid = pid >> 32;
    // get information in task_struct
    task = (struct task_struct *)bpf_get_current_task();
    int exitcode = BPF_CORE_READ(task,exit_code);

    if(exitcode == 0){ //exit normally
        return 0;
    }
    //Log event to ringbuffer to be read by userspace
    e = bpf_ringbuf_reserve(&rb,sizeof(*e), 0);
    if(e){
        e->stack_id = bpf_get_stackid(ctx,&map_stack_traces,0);
        e->exit_code = exitcode >> 8;
        e->sig = exitcode & 0xff;
        e->ppid = BPF_CORE_READ(task,parent,pid);
        bpf_get_current_comm(&(e->comm),TASK_COMM_LEN);
        bpf_ringbuf_submit(e,0);
    }

    return 0;
}
```

BPF程序 过滤，追踪内核数据

```
8   //Use a ringbuffer Map to send data down to userspace
9   struct {
10      __uint(type,BPF_MAP_TYPE_RINGBUF);
11      __uint(max_entries,256 * 1024);
12  }rb SEC(".maps");
13
14  //Use to store stacktrace and will be used by bpf_get_stackid()
15  struct {
16      __uint(type,BPF_MAP_TYPE_STACK_TRACE);
17      __uint(key_size,sizeof(u32));
18      __uint(value_size,VALUESIZE);
19      __uint(max_entries,8192);
20  } map_stack_traces SEC(".maps");
21
```

BPF maps
存储数据
用户程序与内核交换数据

eBPF功能强，安全，但难移植

- Libbpf + CO-RE
- BTFGen + BTFHub

可移植版本分发