AEROSPACE VILLAGE

Racing PX4: Memory Safety and Timing Vulnerabilities

DEF CON 34

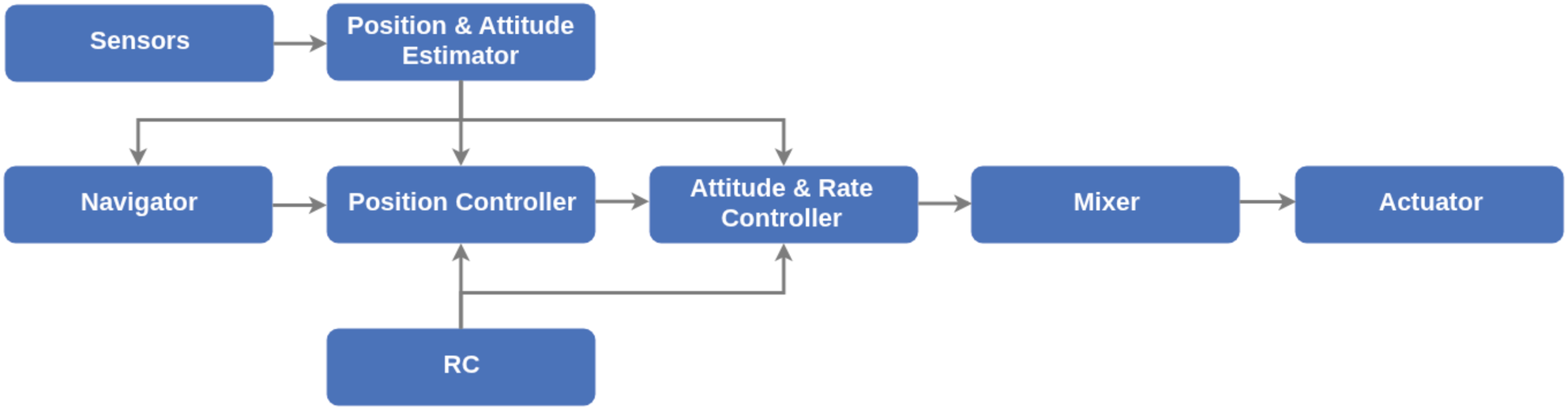
Nefeli Georgilas

Background

- Commercial off-the-shelf (**COTS**) autonomous vehicles are widespread
- These are **safety-critical environments** where real-time missions are done
- Performance and reliability are prioritized over security
- Failures can lead to mission failure, damage to the vehicle, loss of life

Background

- **PX4** is an autopilot commonly used in systems from delivery drones to military-grade UAVs
- Open-source
- Handles flight control, navigation, telemetry

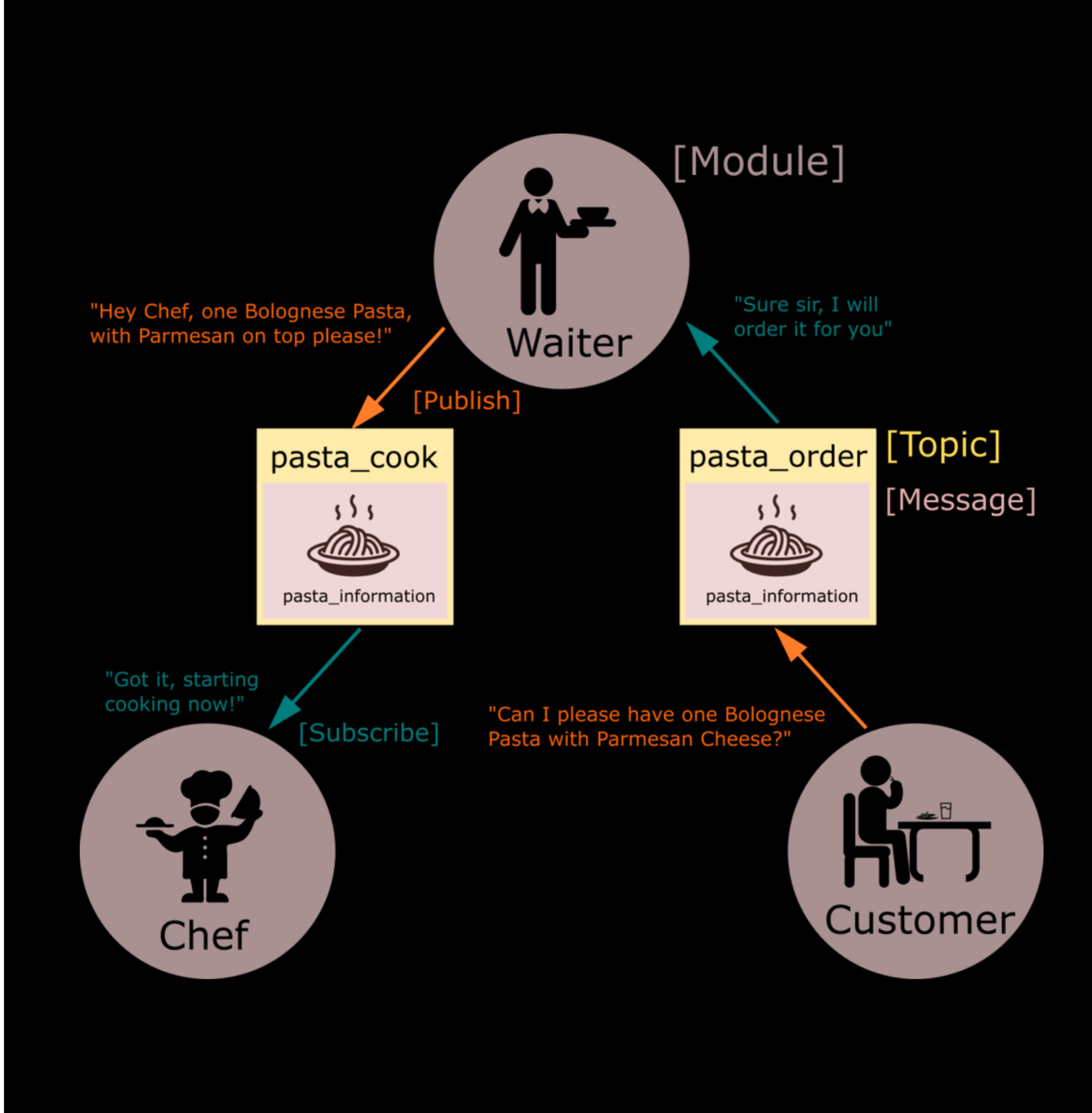


Stack

- PX4 has a **modular** design
- Examples of modules:
  - uORB Middleware
  - MAVLink
  - Logging subsystem (MavlinkUlog)

uORB

- **Micro Object Request Broker**
- Multiple can subscribe
- Publish/Subscribe Model



Share
86.9%
Legend
Filters
fw_virtual_attitude_setpoint
actuator_controls_virtual_fw
mc_virtual_attitude_setpoint
vehicle_roi
gnd_pos_control
actuator_controls_virtual_mc
rover_steering_control
home_position
ex_fixedwing_control
landing_gear
uuv_example_app
vmount
camera_trigger
vehicle_local_position_setpoint
mc_pos_control
vtol_att_control
fw_pos_control_l1
px4_simple_app
gnd_att_control
fw_att_control
camera_capture
vehicle_rates_setpoint
mc_att_control
sensor_bias
tecs_status
position_controller_status
camera_trigger_secondary
position_setpoint_triplet
test_motor
bottle_drop
vehicle_attitude_setpoint
ulog_stream
navigator
sensor_combined
vehicle_command_ack
manual_control_setpoint
logger
vehicle_global_position
vehicle_land_detected
uavcan_parameter_request
airspeed
actuator_controls_2
uavcan_parameter_value
vehicle_attitude
vehicle_command
vehicle_control_mode
gpssim
geofence_result
vehicle_air_data
mission
battery_status
gps
pmw3901
vehicle_magnetometer
vehicle_status
uavcan
vehicle_local_position
px4io
iridiumsbd_status
mavlink
estimator_status
vehicle_gps_position
sensor_baro
satellite_info
subsystem_info
sensors
atxxxx
frsky_telemetry
bst
blinkm
rc_input
commander
sensor_selection
optical_flow
ping
simulator
rc_receiver
linux_sbus
rpi_rc_in
spektrum_rc
offboard_control_mode
send_event
l3gd20
mkblctrl
hmc5883
rm3100
bmm150
lsm303agr
df_hmc5883_wrapper
qmc5883
df_ak8963_wrapper
lsm303d
lis3mdl
ist8310
telemetry_status
vehicle_status_flags
input_rc
sensor_gyro
obstacle_distance
rc_parameter_map
differential_pressure
sensor_mag

Definitions

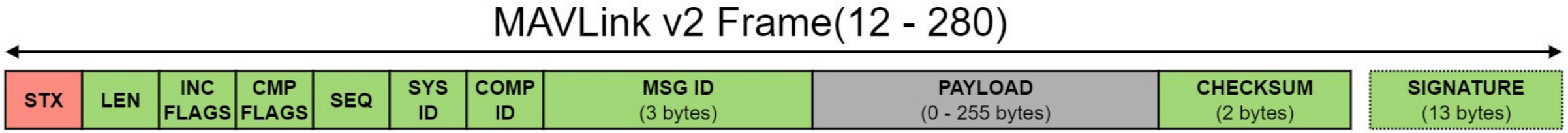
Information	Parameter	Format
ver_sw: 4cfba248 **ver_hw:** PX4_SITL **ver_sw_branch:** uorb_explained_pasta ⋮	**RC_MAP_ROLL: 2** (Channel 2) **BAT1_CELLS: 4** (4S Battery) ⋮	pasta_cook uint64 timestamp uint16 customer_table_id uint8 table_name uint8 cooked_texture uint8 pasta_type float32 pasta_temperature pasta_order uint64 timestamp uint16 customer_table_id uint8 table_name uint8 cooked_texture uint8 pasta_type float32 pasta_temperature • • •

```
uorb status
TOPIC NAME                          INST #SUB #Q SIZE PATH
actuator_armed                         0   16  1   16 /obj/actuator_armed0
actuator_controls_0                    0   10  1   48 /obj/actuator_controls_00
actuator_controls_status_0             0    0  1   24 /obj/actuator_controls_status_00
actuator_motors                        0    1  1   56 /obj/actuator_motors0
actuator_outputs                       0    4  1   80 /obj/actuator_outputs0
actuator_outputs                       1    2  1   80 /obj/actuator_outputs1
actuator_outputs                       2    1  1   80 /obj/actuator_outputs2
actuator_outputs                       3    0  1   80 /obj/actuator_outputs3
actuator_servos                        0    1  1   48 /obj/actuator_servos0
actuator_servos_trim                   0    0  1   40 /obj/actuator_servos_trim0
adc_report                             0    1  1   96 /obj/adc_report0
autotune_attitude_control_status       0    5  1  104 /obj/autotune_attitude_control_status0
battery_status                         0   10  1  168 /obj/battery_status0
commander_state                        0    1  1   16 /obj/commander_state0
control_allocator_status               0    2  1   80 /obj/control_allocator_status0
cpuload                                0    5  1   16 /obj/cpuload0
ekf2_timestamps                        0    0  1   24 /obj/ekf2_timestamps0
ekf2_timestamps                        1    0  1   24 /obj/ekf2_timestamps1
ekf2_timestamps                        2    0  1   24 /obj/ekf2_timestamps2
estimator_attitude                     0    2  1   56 /obj/estimator_attitude0
estimator_attitude                     1    2  1   56 /obj/estimator_attitude1
estimator_attitude                     2    1  1   56 /obj/estimator_attitude2
estimator_baro_bias                    0    1  1   40 /obj/estimator_baro_bias0
estimator_baro_bias                    1    1  1   40 /obj/estimator_baro_bias1
estimator_baro_bias                    2    1  1   40 /obj/estimator_baro_bias2
estimator_event_flags                  0    1  1   56 /obj/estimator_event_flags0
estimator_event_flags                  1    1  1   56 /obj/estimator_event_flags1
estimator_event_flags                  2    1  1   56 /obj/estimator_event_flags2
```

MAVLink

- Vehicles use this protocol to communicate with the Ground Control Station, and vice versa
- For sending commands, telemetry, logging
- Can be a serial connection or by UDP

MAVLink


MAVLink v2 Frame(12 - 280)
STX
LEN
INC FLAGS
CMP FLAGS
SEQ
SYS ID
COMP ID
MSG ID (3 bytes)
PAYLOAD (0 - 255 bytes)
CHECKSUM (2 bytes)
SIGNATURE (13 bytes)

Vectors

- Real-time constraints (middleware)
- Network communication issues
- FTP subsystem issues
- The good ol’ memory corruption

Stack buffer overflow in MavlinkLogHandler

- Analysed CVE-2026-32743 by @zhangteng0526
- A buffer is overflown with more information than it can hold
- Overflows data into other parts of memory (i.e, rbp, rip, etc.)
- LogEntry struct describes a log entry

```
struct LogEntry {
    uint16_t id{0xffff};
    uint32_t time_utc{};
    uint32_t size_bytes{};
    FILE *fp{nullptr};
    char filepath[60];
    uint32_t offset{};
};
```

sscanf()

- **Does not provide bounds-checking**
- Will read however many bytes of input into this buffer

```
if (sscanf(line, "%" PRIu32 " %" PRIu32 " %s", &time_utc, &size_bytes, filepath) != 3) {
    PX4_DEBUG("sscanf failed");
    continue;
}
```

Exploit Methodology

- Craft a log with a filepath > 60 bytes via **MAVLink FTP** (CreateDirectory)
- Send a **LOG_REQUEST_LIST** command
- When the log handler checks /log for metadata, the overflow occurs

Impact

- **Loss of heartbeat from GCS**
- **Console crashing**
- Unresponsive log handler task in tasklist

```
=================================================================
==66954==ERROR: AddressSanitizer: stack-overflow on address 0x7fc2b854ffd0 (pc 0
    ↪ x7fc2bbe447d9 bp 0x7fc2b8550820 sp 0x7fc2b854ffc0 T121)
    #0 0x7fc2bbe447d8 in __sanitizer::StackTrace::StackTrace(unsigned long const*,
        ↪  unsigned int) ../../../../src/libsanitizer/sanitizer_common/
        ↪ sanitizer_stacktrace.h:50
    #1 0x7fc2bbe447d8 in __sanitizer::BufferedStackTrace::BufferedStackTrace()
        ↪ ../../../../src/libsanitizer/sanitizer_common/sanitizer_stacktrace.h:95
    #2 0x7fc2bbe447d8 in __interceptor_malloc ../../../../src/libsanitizer/asan/
        ↪ asan_malloc_linux.cc:144
    #3 0x7fc2bb8a1f18 in __alloc_dir ../sysdeps/posix/opendir.c:118
    #4 0x7fc2bb8a1f18 in opendir_tail ../sysdeps/posix/opendir.c:69
    #5 0x7fc2bb8a1f18 in __opendir ../sysdeps/posix/opendir.c:92
    #6 0x7fc2bbd86342 in __interceptor_opendir ../../../../src/libsanitizer/
        ↪ sanitizer_common/sanitizer_common_interceptors.inc:3101
```

Remediation

- Limit string width to **59** (append 1 byte for null-terminator)
- Use appropriate string format specifier (%59s)
- Avoid vulnerable functions such as gets() for reading user-inputted data
- For more complex parsing of strings into tokens, use **strtok()**

Zenoh uORB Subscriber Allows Arbitrary Stack Allocation

- Analysed CVE-2026-32708 by @max-r-b and @enitmar
- A **stack variable-length array (VLA)** is allocated for incoming Zenoh payloads

- How is the payload determined?
  - **payload_len** field
  - This length has no bounds checking before the allocation

Data Handling

Every time a message is received, the payload must be validated and a new stack frame is created for data_handler()

- Runs on each subscription update

```
const size_t max_payload_size = _uorb_meta->o_size + 4;

if (len > max_payload_size || len < 4) {
    return;
}
```

Exploit Methodology

No validation for non-contiguous payloads

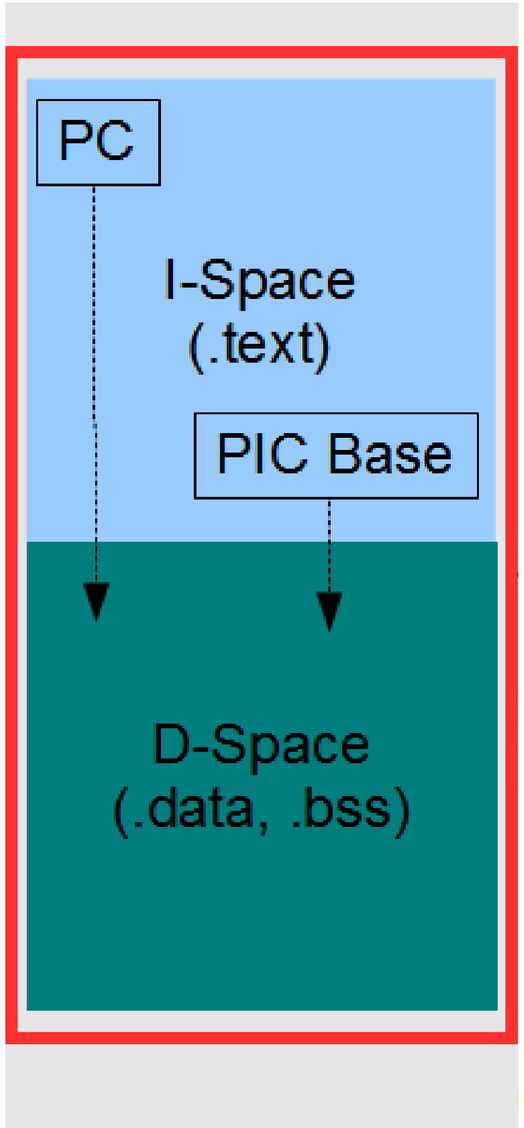
- Simple stack-based buffer overflow yet again

Craft a packet with a larger size than 64 bytes

- Will hit the stack VLA

Impact

- **Overflows other parts of memory**
- May crash the Zenoh bridge task



```
AddressSanitizer:DEADLYSIGNAL
=================================================================
==23755==ERROR: AddressSanitizer: stack-overflow on address 0x7ffdd55e7c58 (pc 0
    ↪ x5f8c5bb26e31 bp 0x7ffdd5de4ef0 sp 0x7ffdd55e6c60 T0)
    #0 0x5f8c5bb26e31 in uORB_Zenoh_Subscriber::data_handler(z_loaned_sample_t
        ↪ const*) ../src/modules/zenoh/subscribers/uorb_subscriber.hpp:97
    #1 0x5f8c5bb2630e in main /home/maxime/Projects/vuln_research/PX4-Autopilot/
        ↪ zenoh-vla-stack-overflow-poc/poc.cpp:184
    #2 0x7a28c6e29d8f in __libc_start_call_main ../sysdeps/nptl/
        ↪ libc_start_call_main.h:58
    #3 0x7a28c6e29e3f in __libc_start_main_impl ../csu/libc-start.c:392
    #4 0x5f8c5bb25404 in _start (/home/nefeli/scripts/zenoh-vla-stack-overflow-poc
        ↪ /build-make/zenoh_vla_stack_overflow_poc+0x2404)

SUMMARY: AddressSanitizer: stack-overflow ../src/modules/zenoh/subscribers/
    ↪ uorb_subscriber.hpp:97 in uORB_Zenoh_Subscriber::data_handler(
    ↪ z_loaned_sample_t const*)
==23755==ABORTING
```

Stack Overflow error from ASan run when testing the PoC

Remediation

- Implement the **maximum length check** to validate the payload against a limit
- Consider using fixed-length arrays instead of VLAs if possible

Race Conditions

- Cppcheck suggests a potential Use-After-Free (UAF). But this might be a false positive.

Q: Is the function vulnerable?
A: Maybe!

```
/home/nefi/PX4-Autopilot/src/modules/mavlink/mavlink_ulog.cpp:242:2: warning: Calling method 'unlock()' when 'this' might
be invalid [thisUseAfterFree]
 unlock();
 ^
```

```
// lines 46-48
bool MavlinkULog::_init = false;
MavlinkULog *MavlinkULog::_instance = nullptr;
px4_sem_t MavlinkULog::_lock;
```

Polling System

Each MAVLink instance's receiver thread accesses the **poll()** function. The function takes the following arguments:

- &fds[0] - a pointer to the first element of the struct called pollfd
- 1 - the number of descriptors it should track
- timeout - the maximum wait time (ms)

When MAVLINK_UDP is used, all incoming UDP packets are read in this function. For each packet, there is an acknowledgement that is handled too.

```
// lines 245-257
void MavlinkULog::handle_ack(mavlink_logging_ack_t ack)
{
  lock();

  if (_instance) { // make sure stop() was not called right before
    if (_wait_for_ack_sequence == ack.sequence) {
      _ack_received = true;
      publish_ack(ack.sequence);
    }
  }
  unlock();
}
```

Handling Acknowledgements

stop() and handle_ack() create a subtle race condition:

- **_instance** is a global static pointer
- stop() and handle_ack() use the same lock (so no UAF)

Steps:

1. stop() is called (delete _instance)
2. try_start() creates a new instance
3. new session starts
4. old ack arrives???
5. _instance check passes

Exploit Methodology

- Send packets to stop & start the logger via pymavlink link
- UDP port 18570
- Old ACKs are accepted as valid in a new session

Commands:

- MAV CMD **LOGGING START** (ID 2510)
- MAV CMD **LOGGING STOP** (ID 2511)

Impact

- Corrupt or unreliable telemetry
  - Missed retransmission
  - False delivery confirmation
- MAVLink desynchronization

Metric	Baseline	Attacked
RX Loss	0.0%	254.0%
Publication Rate (msgs/s)	~8.8K	~14.8K

Table 2.1: Telemetry and Performance Metrics Comparison

```
WARN  [mavlink] no ack from logger (is it running?)
WARN  [mavlink] no ack from logger (is it running?)
WARN  [mavlink] no ack from logger (is it running?)
WARN  [mavlink] no ack from logger (is it running?)
ERROR [logger] Ack timeout. Stopping mavlink log
INFO  [logger] Start mavlink log
WARN  [mavlink] no ack from logger (is it running?)
WARN  [mavlink] no ack from logger (is it running?)
WARN  [mavlink] no ack from logger (is it running?)
WARN  [mavlink] no ack from logger (is it running?)
WARN  [mavlink] no ack from logger (is it running?)
WARN  [mavlink] no ack from logger (is it running?)
WARN  [mavlink] no ack from logger (is it running?)
WARN  [mavlink] no ack from logger (is it running?)
ERROR [commander] vehicle_command lost, generation 18927 -> 18931
ERROR [navigator] vehicle_command lost, generation 18927 -> 18931
ERROR [commander] vehicle_command lost, generation 18941 -> 18944
ERROR [navigator] vehicle_command lost, generation 18941 -> 18944
ERROR [logger] Ack timeout. Stopping mavlink log
INFO  [logger] Start mavlink log
WARN  [mavlink] no ack from logger (is it running?)
WARN  [mavlink] no ack from logger (is it running?)
WARN  [mavlink] no ack from logger (is it running?)
WARN  [mavlink] no ack from logger (is it running?)
WARN  [mavlink] no ack from logger (is it running?)
pxh> 5FWARN  [mavlink] no ack from logger (is it running?)
WARN  [mavlink] no ack from logger (is it running?)
WARN  [mavlink] no ack from logger (is it running?)
ERROR [logger] Ack timeout. Stopping mavlink log
INFO  [logger] Start mavlink log
```

Remediations

- Create unique sessions for each initialization of the logger, so that stale acknowledgments aren’t used in future sessions
- Implement rate-limiting for logger toggling commands.
- Add a check within handle ack() to ensure that an active session exists **before** the periodic poll

Conclusion and Q&A

- Tradeoff between performance and security
- Even small vulnerabilities can compromise a vehicle in a safety-critical mission