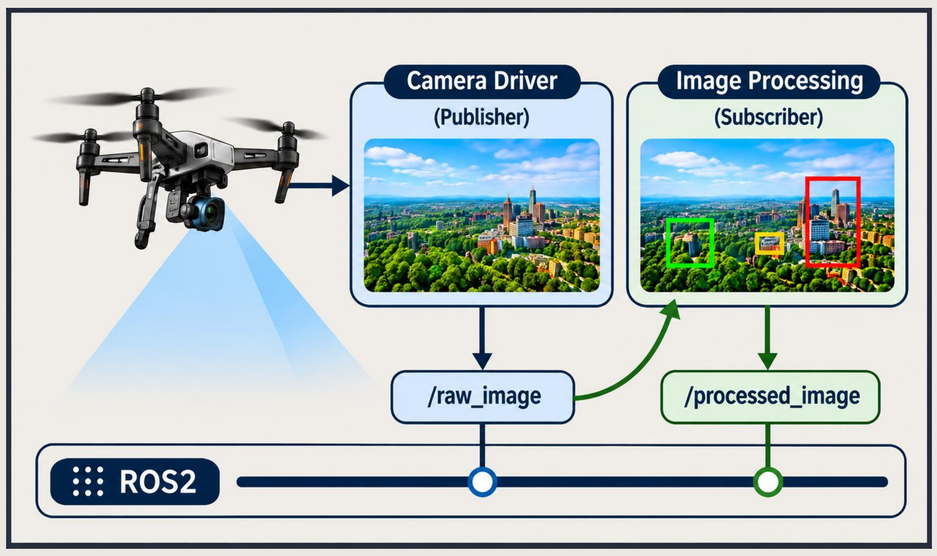
ESROS2: Application Sandboxes for Secure ROS2

Dev Tejas Gandhi, Vinod Ganapathy

Computer Systems Security Lab, Indian Institute of Science

1 Introduction

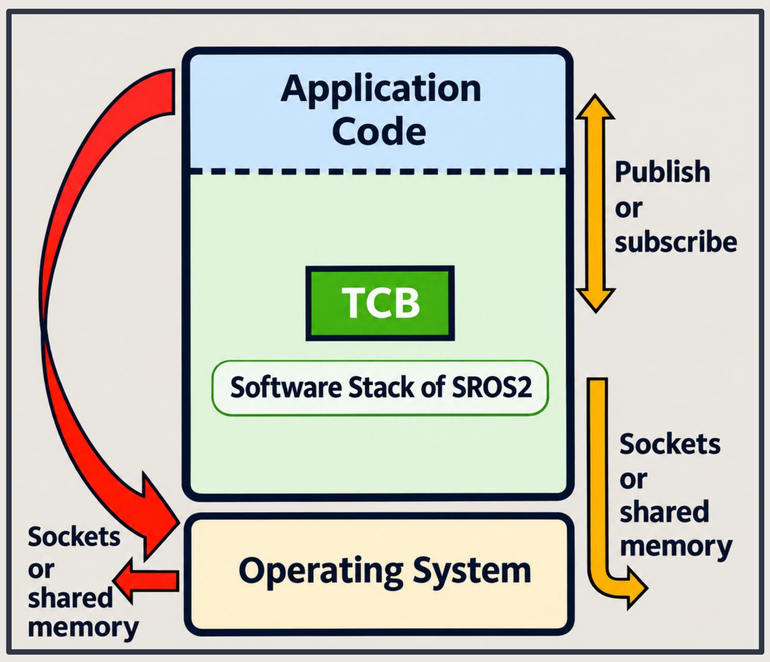
- **ROS2 uses a publish–subscribe-based middleware for robotics.**
- **ROS2 has some known security challenges.**



2 Bypassing SROS2 API

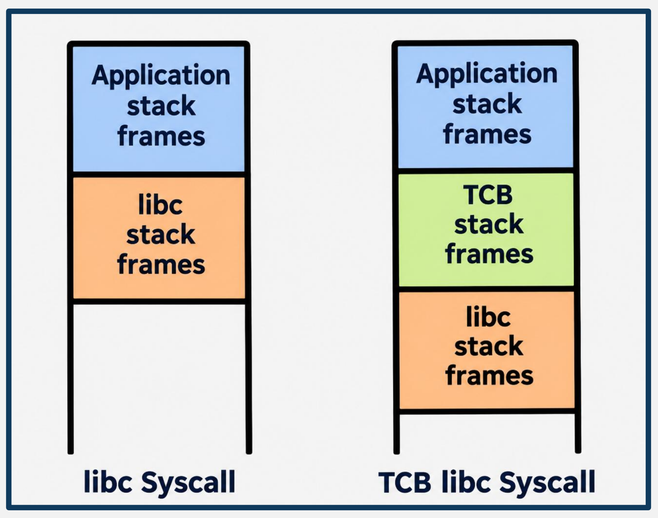
- **SROS2 addresses these security limitations in ROS2.**
- **Security guarantees hold only when applications use SROS2 APIs.**





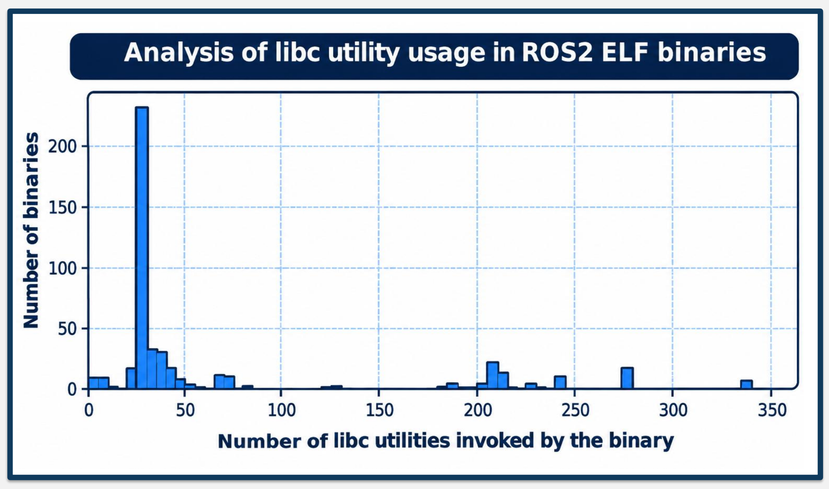
3 ESROS2-Invariant and ESROS2-SCGate

A ROS2 application can invoke system calls in several ways. Two common examples are:

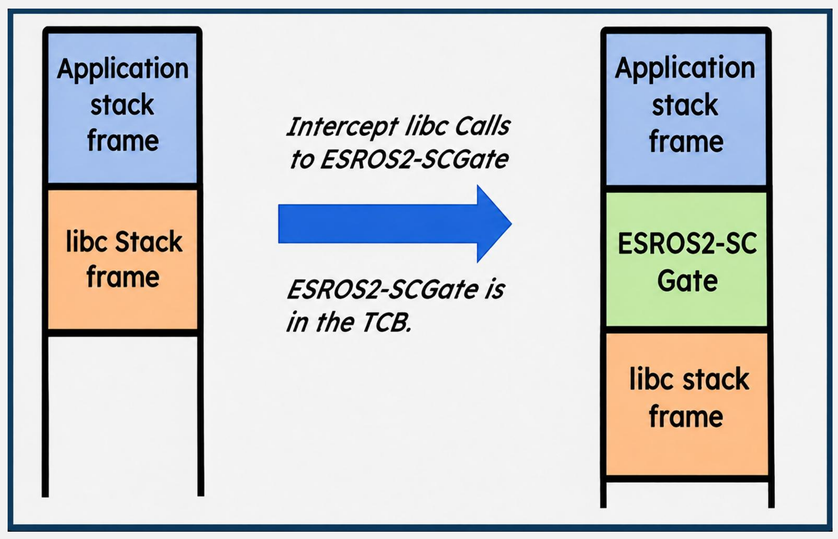


ESROS2-Invariant : When a system call is issued it must be meditated by the TCB.

- **`libc` Syscalls are common in practice but violate the ESROS2-Invariant.**
- **Analyzed 456 ROS2 apps to determine how they are using `libc` APIs.**
- **Observed a median of 27 `libc` API per app.**

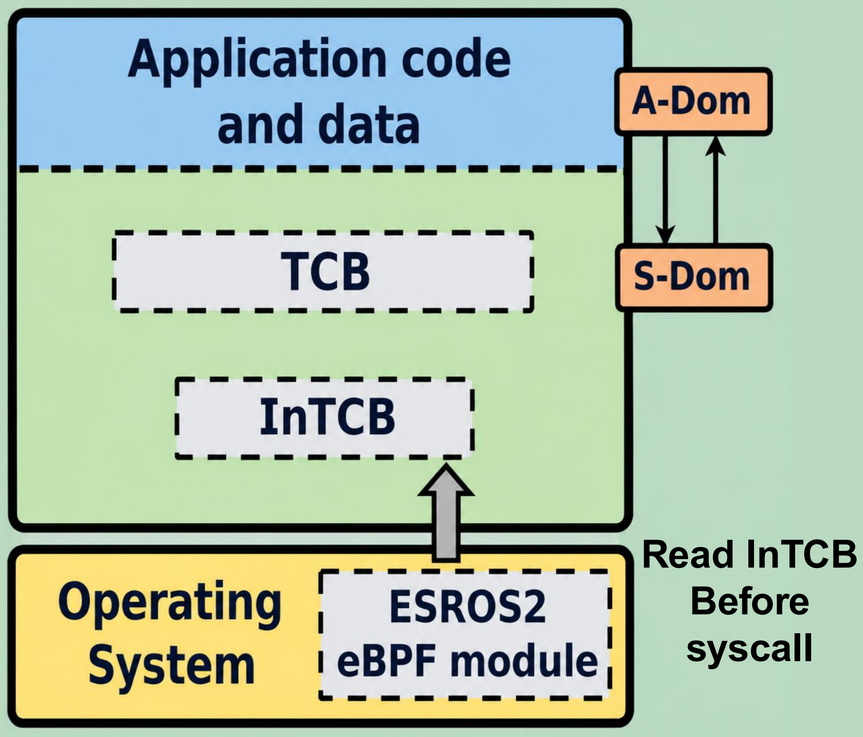


- **Developed library wrappers for `libc` APIs that violate the ESROS2-Invariant.**
- **Implemented app-specific policy checks within these wrappers.**
- **Wrappers are implemented as ESROS2-SCGates.**



4 Enforcing ESROS2-Invariant

- **Introduce an InTCB variable to track the app's stack state in user-space.**
- **Divide the application's address space into two domains using Intel MPK to isolate InTCB.**



5 Case Study Evaluation

The goal of the case study evaluation is to measure runtime overhead and assess the practicality of generating `libc` policies for real-world applications.

Case Study	Number of Applications	Average Policies per Application (Ceiling)	Runtime Overhead (%)
Robot Arm Manipulation	4	6	4.16%
Night Vision Image Processing	3	10	6.78%
Autonomous Navigation using SLAM and Nav2	9	5	5.25%

6 Summary

We identified a limitation in SROS2, introduced a new invariant, and enforced it with only 5–7% runtime overhead. Our evaluation also demonstrates the practicality of `libc` policy generation.

ESROS2-Paper