



ROSRV: Runtime Verification for Robots

Jeff Huang, Cansu Erdogan, Yi Zhang, Brandon Moore,
Qingzhou Luo, Aravind Sundaresan and Grigore Rosu

University of Illinois at Urbana-Champaign
SRI International, Menlo Park, CA

Robot Operating System

- Meta-operating system for robot software development
 - Not a traditional operating system:
 - No scheduling, process management
- Message passing interface based on a graph architecture
 - ROS processes (a.k.a. *nodes*)
 - Receive,
 - Post,
 - Multiplex messages

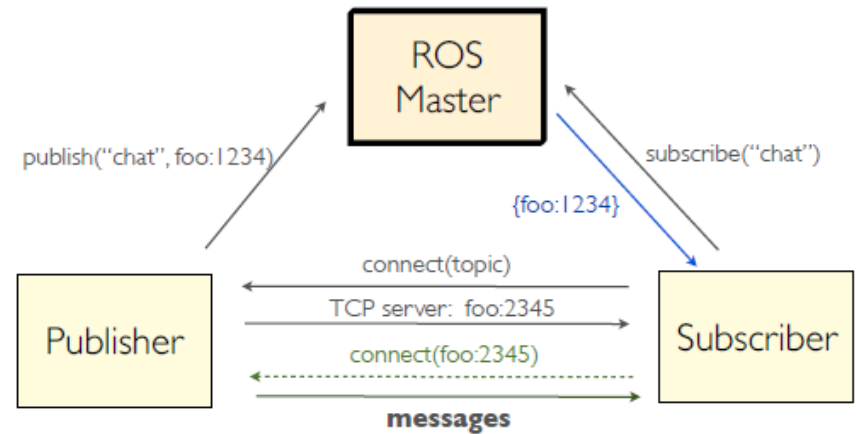


Fig. 1. ROS Communication Architecture

Motivation

Problem

- ROS is highly dynamic and distributed (and popular!)
 - No control over what nodes do
 - Lacks security protection mechanism

Solution

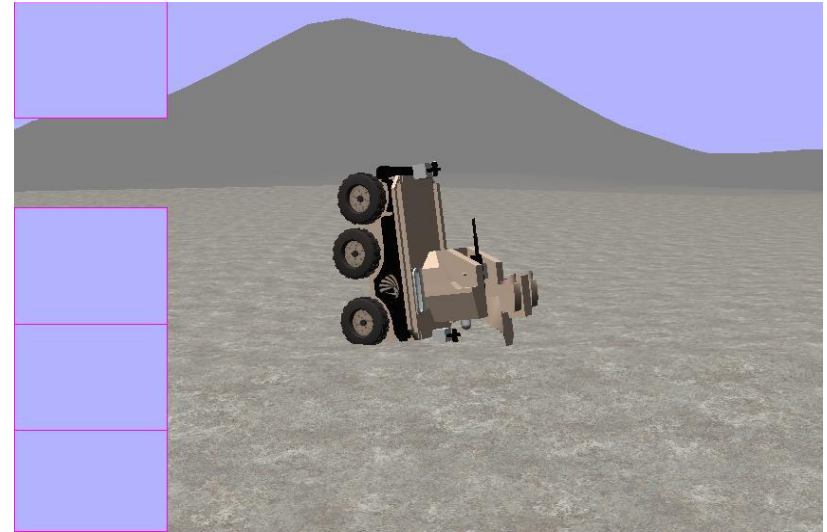
- Monitoring safety properties at runtime
- Enforcing security policies at runtime

LandShark

Shoots itself



Flips over



Stays in safe zone

The image displays a ROS simulation environment with a terminal window on the left and a 3D simulation window on the right.

Terminal Window:

```
[ WARN] [1390975712.522835013]: Monitor: Not allowed to trigger in this pose!  
[ WARN] [1390975712.522885631]: Monitor: Not allowed to trigger in this pose!  
[ WARN] [1390975712.522945178]: Monitor: Not allowed to trigger in this pose!  
[ WARN] [1390975712.523040149]: Monitor: Not allowed to trigger in this pose!  
[ WARN] [1390975712.623287275]: Monitor: Not allowed to trigger in this pose!  
[ WARN] [1390975712.623336592]: Monitor: Not allowed to trigger in this pose!  
user request from ip address 127.0.0.1 port 57495  
[ INFO] [1390975715.868178865]: Node rosrvc trying to monitorControl  
[ INFO] [1390975715.868220358]: disable monitor: safeTrigger  
[ INFO] [1390975715.868240779]: Node rosrvc successfully monitorControl from 127.0.0.1  
user request from ip address 127.0.0.1 port 57751  
[ INFO] [1390975724.966542249]: Node rosrvc trying to monitorControl  
[ INFO] [1390975724.966634999]: enable monitor: safeZone  
[ INFO] [1390975724.966678252]: Node rosrvc successfully monitorControl from 127.0.0.1
```

Simulation Window:

The simulation window is titled "/opt/hacms/landshark/share/landshark_sim/worlds/DesertWall.wbt - Webots PRO 7.3.0". It shows a 3D environment with a large green square on the ground, representing a safe zone. A robot is visible in the bottom right corner of the safe zone. The interface includes a menu bar (File, Edit, View, Simulation, Build, Robot, Tools, Wizards, Help), a toolbar with various icons, and a status bar showing the time "0:01:12:736" and zoom level "0.57x".

Terminal Window (Bottom):

```
zy@ubuntu: ~  
zy@ubuntu:~$ rosrvc -enable safeZone  
zy@ubuntu:~$
```

ROS...

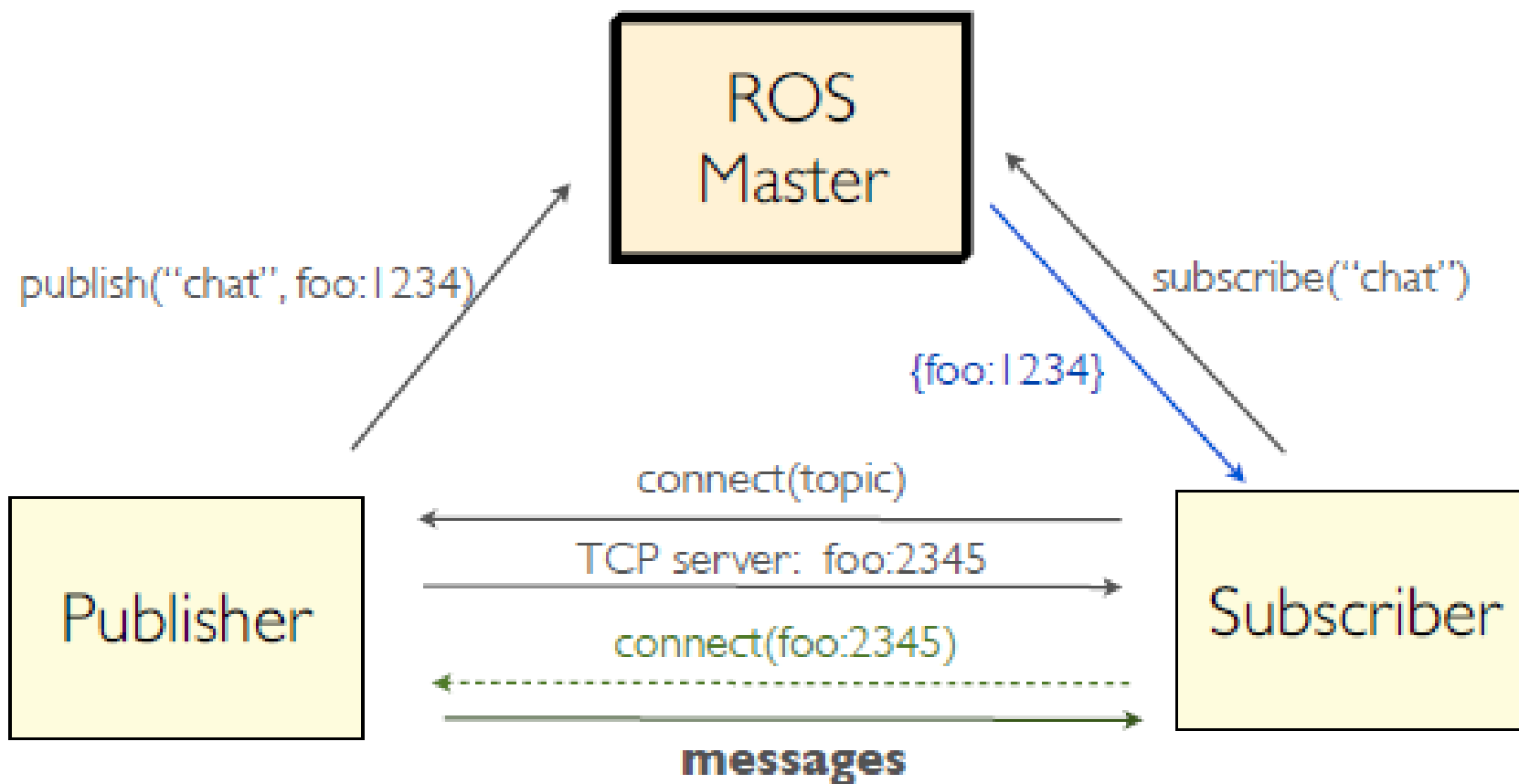
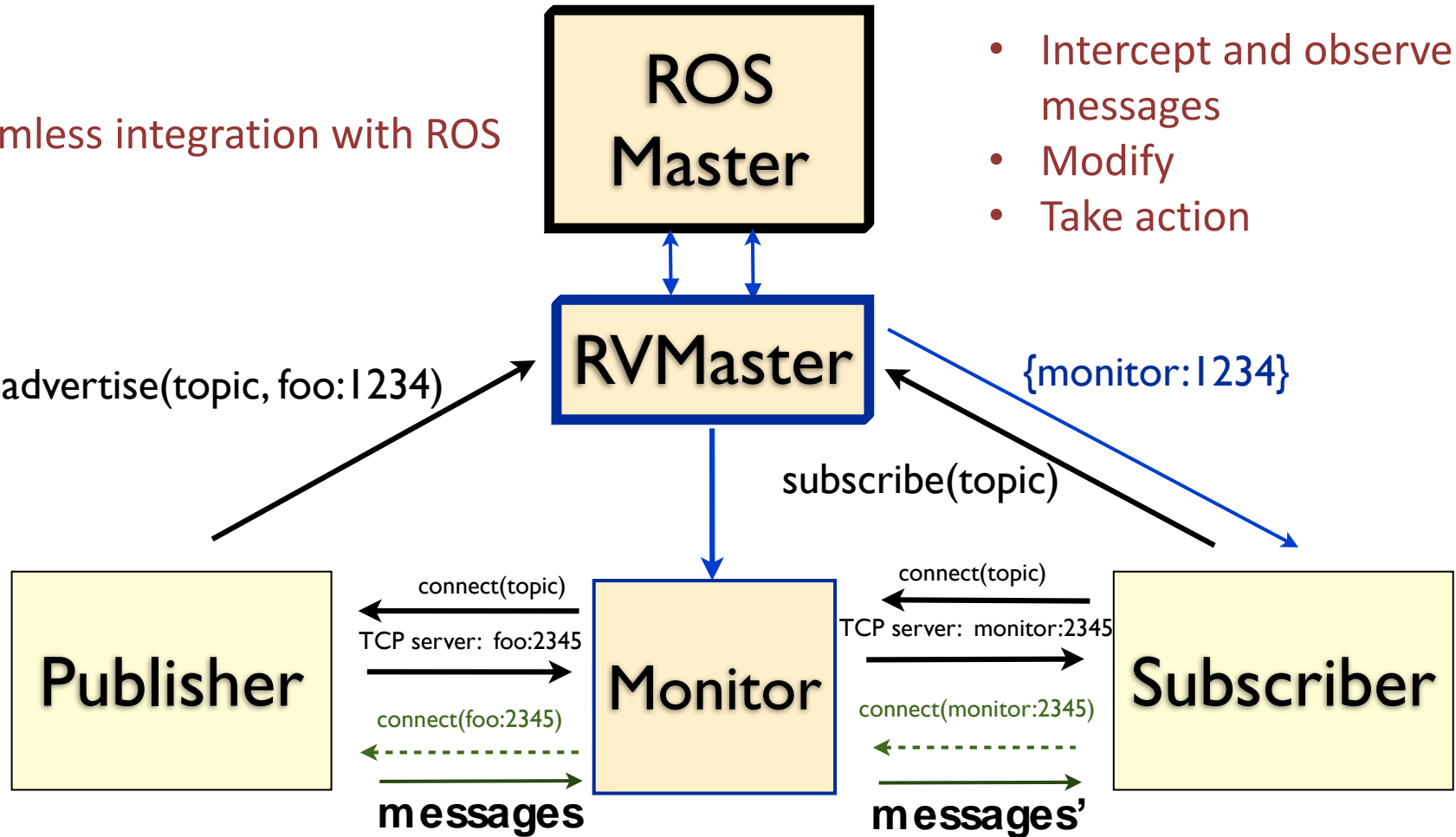


Fig. 1. ROS Communication Architecture

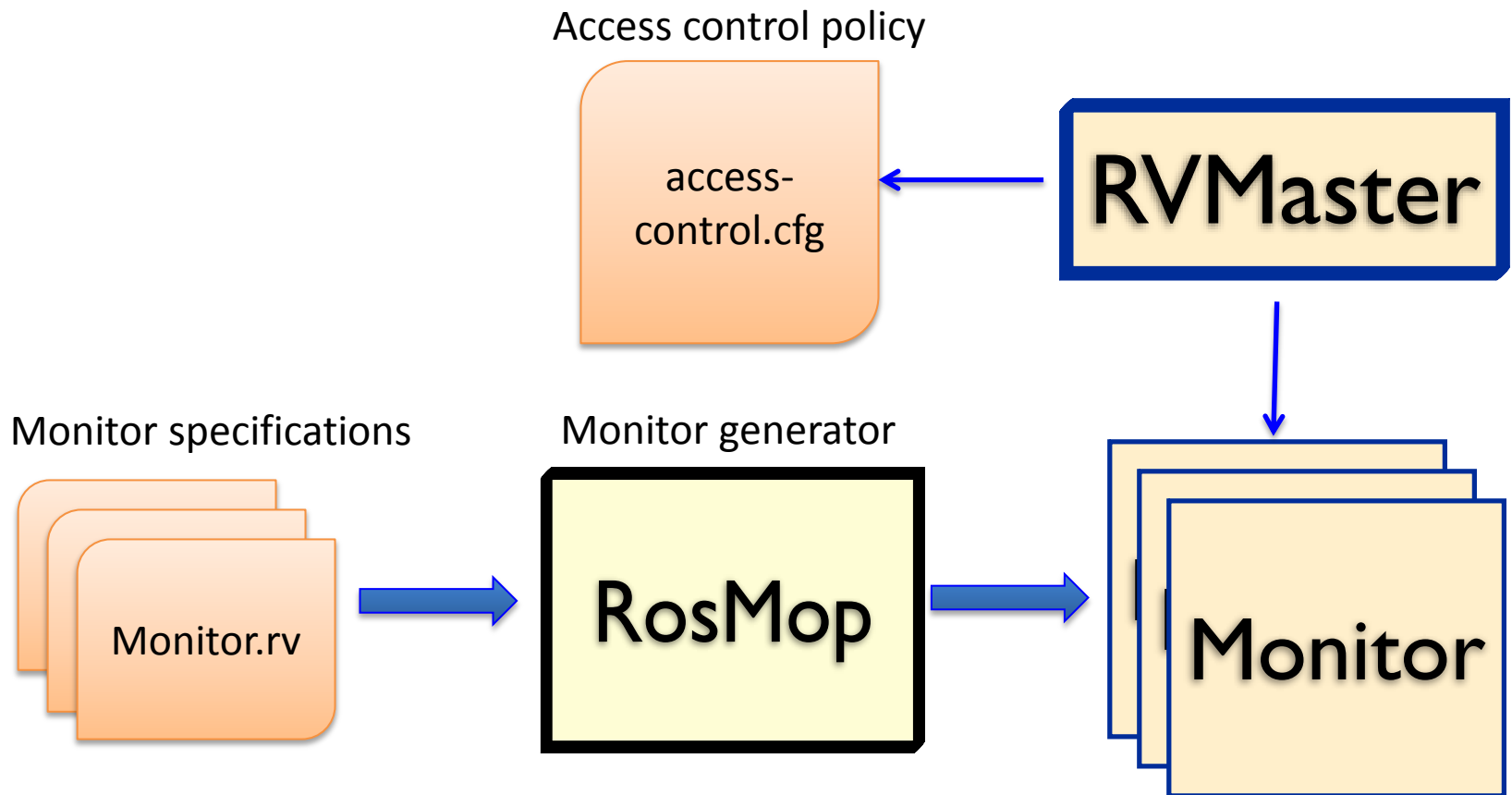
... vs. ROSRV

Seamless integration with ROS

- Intercept and observe messages
- Modify
- Take action



System Overview



Monitor Specifications

```
safeBehavior() {
  bool isSafeTrigger = false;
  event checkPosition(string N, double P)
    /landshark/joint_states sensor_msgs/JointState
    '{name[1]:N, position[1]:P}' {
    ...
    //check gun position
    ...
  }
  event safeTrigger() /landshark_control/trigger
    landshark_msgs/PaintballTrigger '{} ' {
    ...
    //drop trigger message if not safe
    ...
  }
  ...
}
```

Access Control Policy

[Groups]

```
localhost = 127.0.0.1  
certikos = ip1 ip2 ip3 ip4  
ocu = ip5 ip6 ip7 ip8
```

[Nodes]

```
default=localhost  
/landshark_radar=certikos
```

[Publishers]

```
default=localhost certikos  
/landshark_control/trigger= ocu
```

[Subscribers]

```
default = localhost certikos  
/landshark/gps = ocu
```

[Commands]

```
# Commands: full access  
getSystemState = localhost certikos ocu  
# Commands: limited access  
lookupNode = localhost certikos  
# Commands: local access only  
shutdown = localhost
```

shutdown
command cannot
be sent by any
node other than
localhost

Future Work

- Formal verification
 - Monitors are correct
 - Nodes cannot bypass monitoring
 - No irrelevant events yield bogus errors
- Scalability
 - Decentralized approach

Conclusion

- Safer and more secure robots
- Automatically generated monitors
- User defined formal safety and security specifications



Thank you!

Questions?

Check our demo at:

<http://fsl.cs.illinois.edu/ROSRV>