Assurance and Security for AI CPS

Presenter: Dionisio de Niz

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

Copyright 2022 Carnegie Mellon University.

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

Carnegie Mellon® is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

DM22-0140

Assurance for AI CPS (e.g., Autonomy)

Assurance of AI

- Enable ML to
  - Detect complex patterns (object recognition), handle uncertainty
- Interact with unknown environment

Cyber-Physical Systems (Most Systems in Field)

- React to physical environment
- Safe behavior: safe actions at right time (e.g., prevent crash)
- Security : ensure attacker does not make system crash

© 2020 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

Enforcement-based Verification

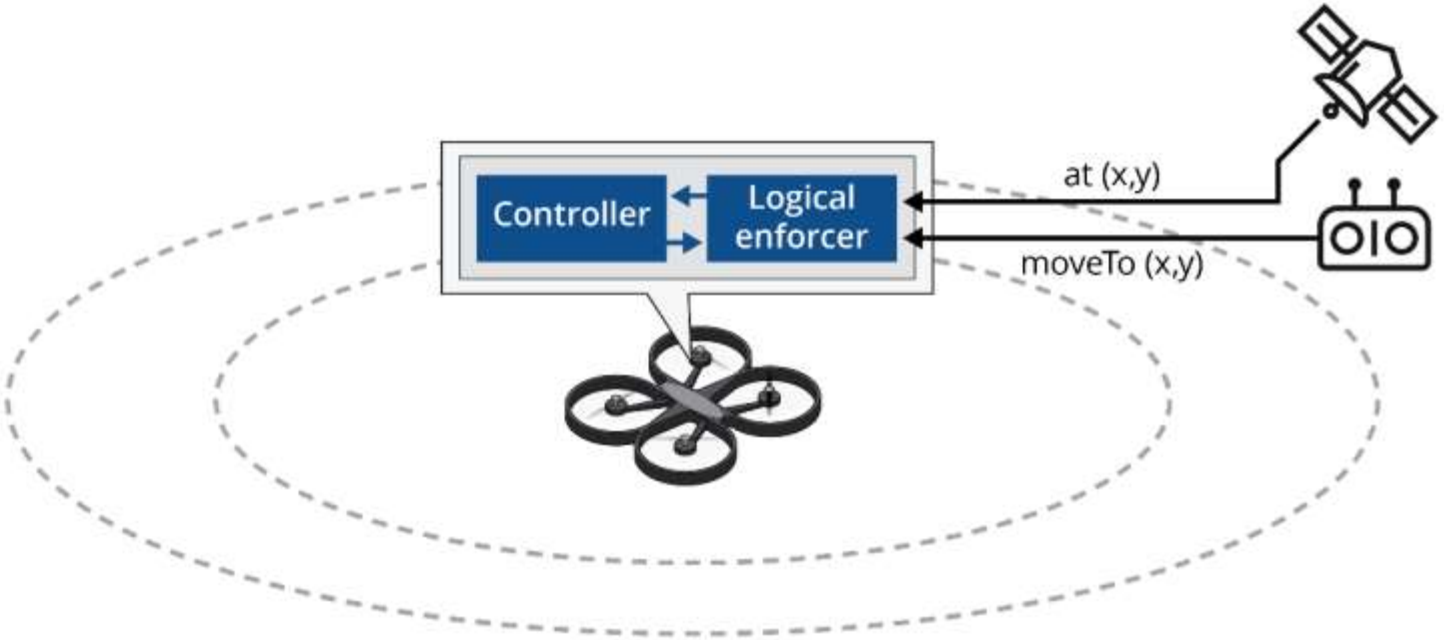
Add **simpler (verifiable)** runtime enforcer to make algorithms predictable

Formally: specify, verify, and compose multiple enforcers

- Logic: Enforcer **intercepts/ replaces** unsafe action
- Timing: at **right time**
- Physics: verified physical effects

Protect enforcers against

- Failures (Safety)
- Attacks (Security)



© 2020 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

Verifying Physics (Control Theory)

Recoverable Set: $\varepsilon_{SC^j}(1)$

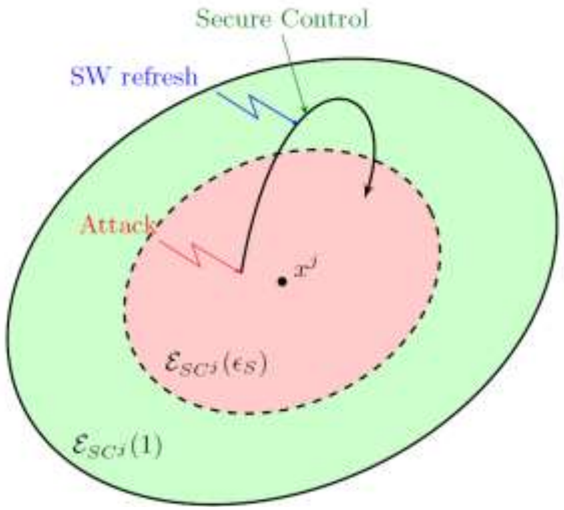
Safety Set: $\varepsilon_{SC^j}(\epsilon_s) \triangleq \epsilon_s\, \varepsilon_{SC^j}(1)$

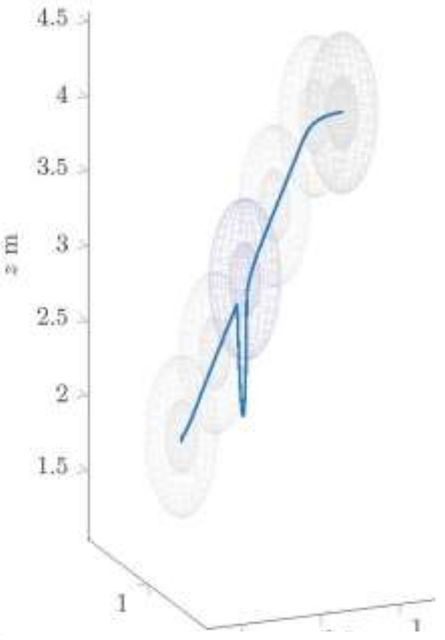
Controlled System: $\dot{x} = f_\varphi(x) \triangleq f(x, \varphi(x))$
Lyapunov Function: $V_\varphi : \mathbb{R}^n \to \mathbb{R}$, $\mathcal{N}_{V_\varphi}(x_{eq}) \subseteq \mathcal{N}_\varphi(x_{eq})$,
$V_\phi(x_{eq}) = 0$ and $\forall x \in \mathcal{N}_{V_\varphi}(x_{eq}) - \{x_{eq}\}$: (i) $V_\varphi(x) > 0$,

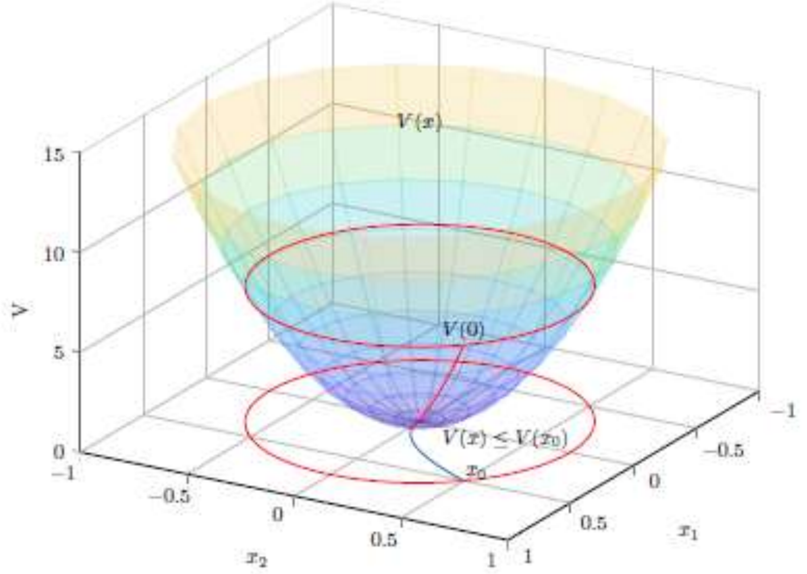
$$\dot{V}_\varphi(x) = \frac{\partial V}{\partial x} \cdot f_\varphi(x) < 0$$

Lyapunov level set: For $\epsilon > 0$,

$$\mathcal{E}_\varphi(\epsilon) = \{x \in \mathcal{N}_{V_\varphi}(x_{eq}) | V_\varphi(x) \leq \epsilon\}. \qquad \epsilon \leq 1$$







© 2020 Carnegie Mellon University

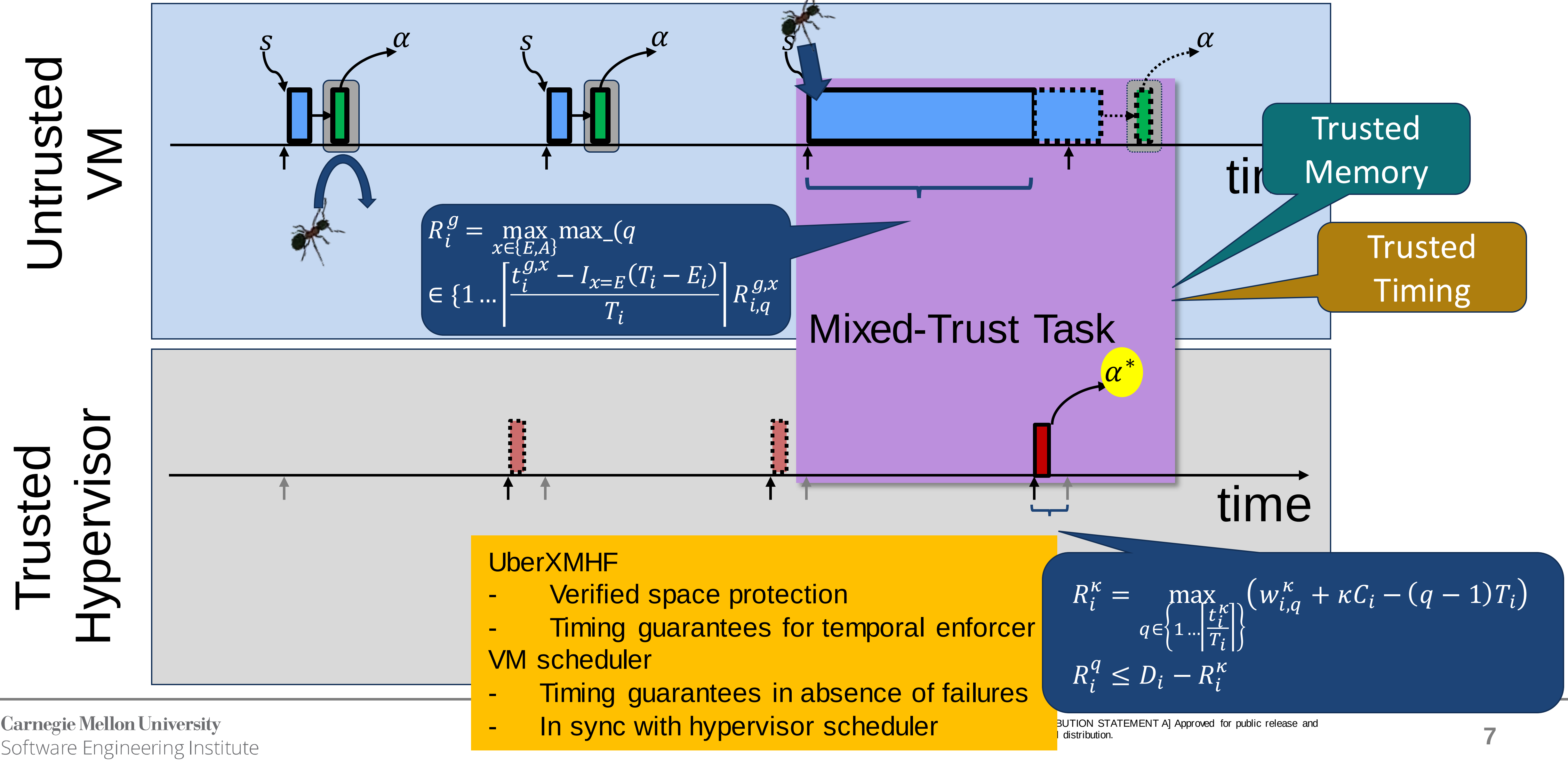
[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

Drone Experiment

© 2020 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

Real-Time Mixed-Trust Computation (Enforcement Protection)



BUTION STATEMENT A] Approved for public release and
distribution.

Handling Failures / Degraded Modes

LIDAR

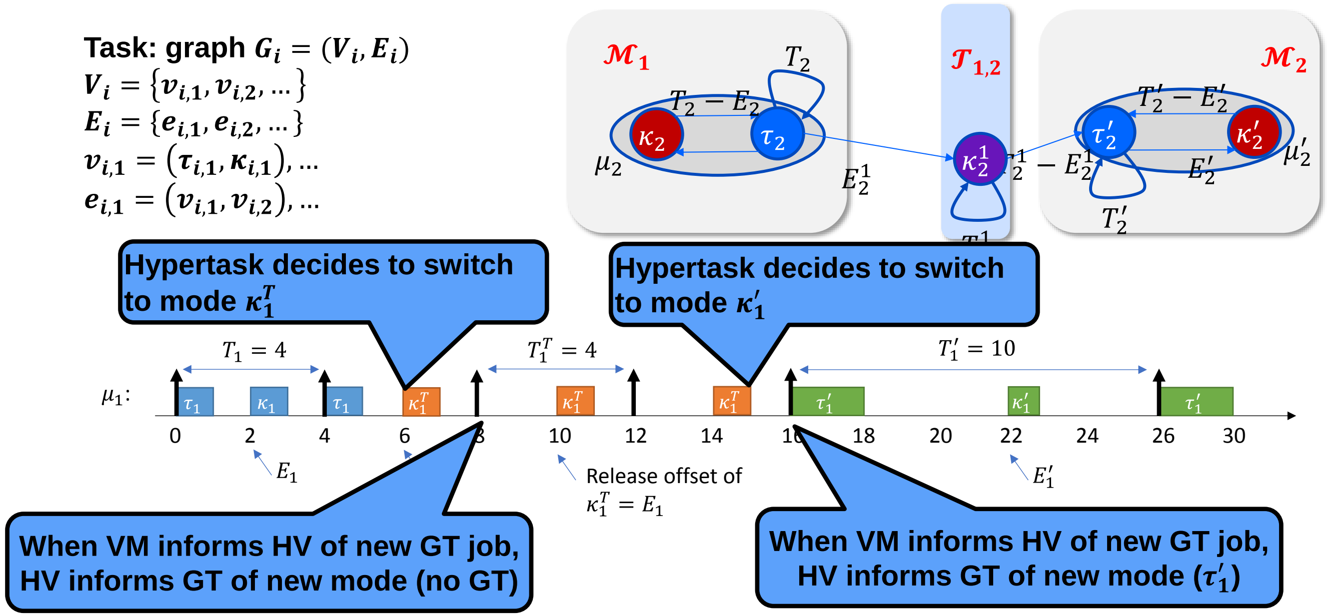
- Detection distance $20\ m$
- Max braking: $-10 \frac{m}{s^2}$
- Max speed: $20 \frac{m}{s}$

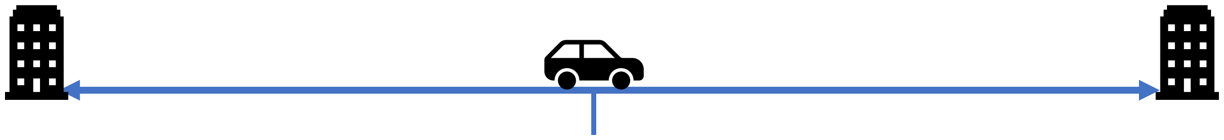
SONAR

- Detection distance $5\ m$
- Max braking: $-10\ m/s^2$
- Max speed: $10 \frac{m}{s}$

LIDAR Failure Transitioning Enforcer

- Upon failure: start braking at $-10 \frac{m}{s^2}$
- Once speed < $10 \frac{m}{s}$ Transition to SONAR enforcer





© 2020 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.