
# Rapid Certifiable Trust

Dr. Dionisio deNiz

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.



DM19-1079

# Problem

New Technologies

- **Key for DoD Superiority**
- **Validation of behavior** essential for adoption
  - Non-deterministic algorithms, e.g., Machine Learning (ML)

Assured Autonomy

- 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)

# Trusting Rapid Capability Fielding

**Fast**

- **DoD Rapid Capability Offices** (Air Force, Army, Strategic Capability Office)
- Maximize reuse
  - Open source
  - Ever increasing complexity

**Multiply Human Capabilities**

- Learning Autonomy
  - Continuously adapting behavior

**BUT** Trustworthy

- Fast validation
- Safety-critical interactions with the physical world (Cyber-Physical System)
  - Physics
  - Timing
  - Logic

# Rapid Certifiable Trust

**Fast Trustworthy Validation**

- Automation with formal verification

**Complexity**

- Traditional Verification Does Not Scale

**Adapting Behavior**

- Cannot verify at design time

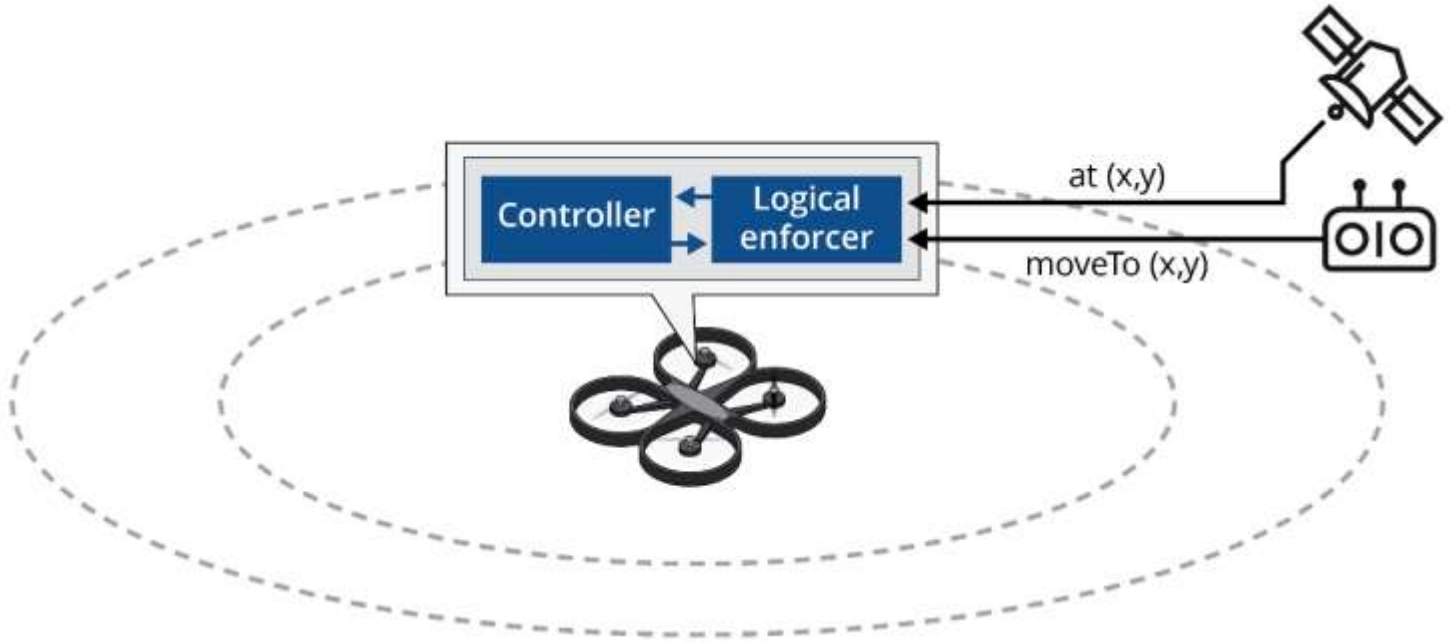
# Enforcement-based Verification

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

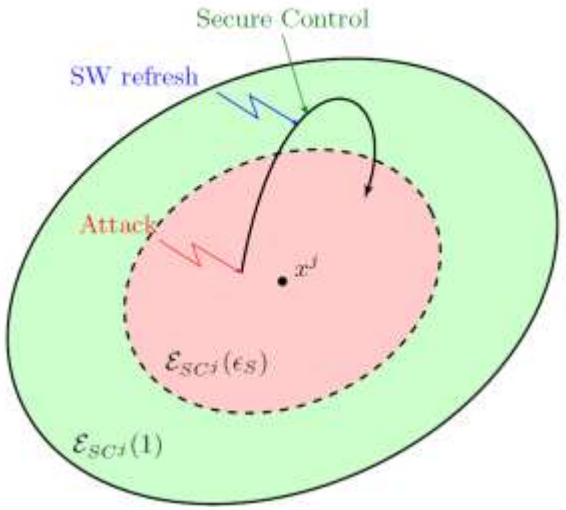
Protect enforcers against failures/attacks



# Verifying Physics (Control Theory)

Recoverable Set: $\varepsilon_{SCj}(1)$

Safety Set: $\varepsilon_{SCj}(\epsilon_s) \triangleq \epsilon_s\, \varepsilon_{SCj}(1)$




**Controlled System**: $\dot{x} = f_\varphi(x) \triangleq f(x, \varphi(x))$

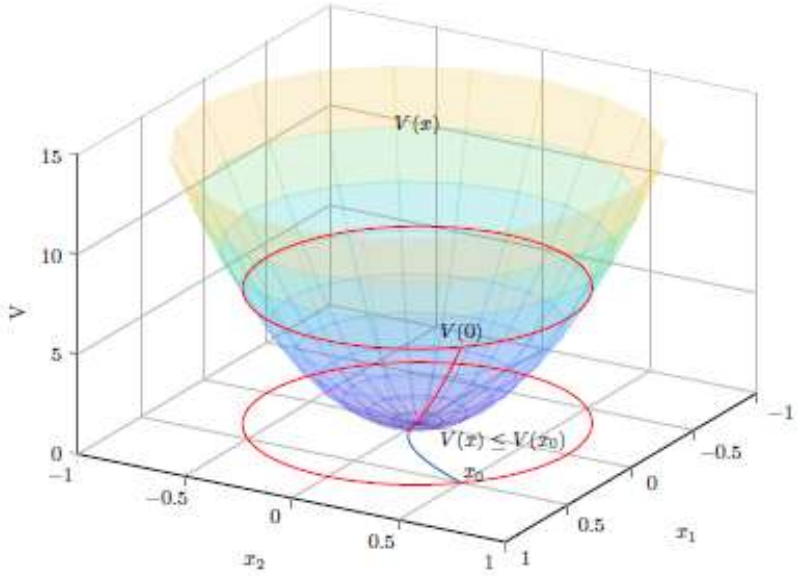
**Lyapunov Function**: $V_\varphi : \mathbb{R}^n \to \mathbb{R}$, $\mathcal{N}_{V_\varphi}(x_{eq}) \subseteq \mathcal{N}_\varphi(x_{eq})$,

$V_\phi(x_{eq}) = 0$ 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 $\epsilon > 0$,

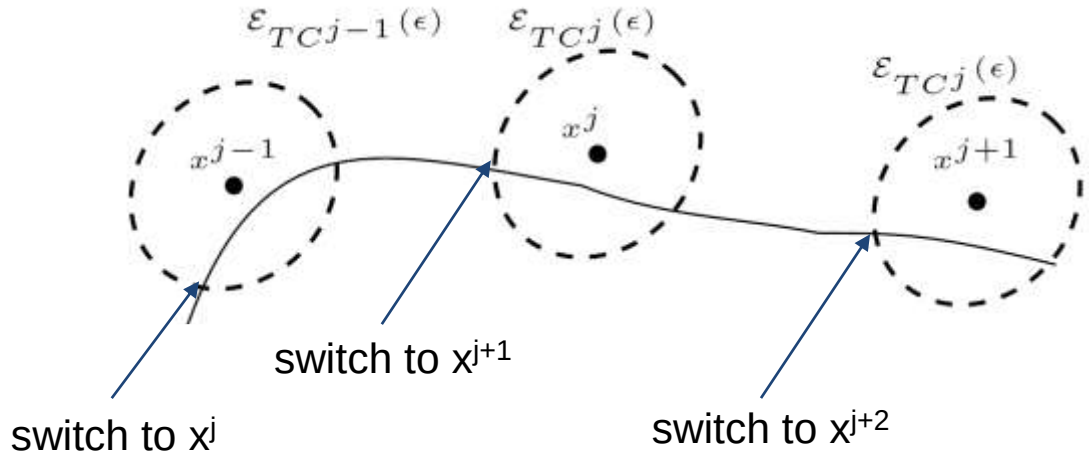
$$\mathcal{E}_\varphi(\epsilon) = \{x \in \mathcal{N}_{V_\varphi}(x_{eq}) | V_\varphi(x) \leq \epsilon\}. \qquad \epsilon \leq 1$$



# Analysis of Mission Progress

Idea:

Provide a sequence of waypoints that represent a sequence of equilibrium points around which we define the Safe Set.




Goal:

- Safely transition from one waypoint to the next
- Liveness (in the case of no errors)

# Analysis of Mission Progress Enforcing Unsafe Behavior

6 DOF $\Rightarrow$ 12 state variables

$$\ddot{p}_x = -\cos\phi \sin\theta \frac{F}{m}$$

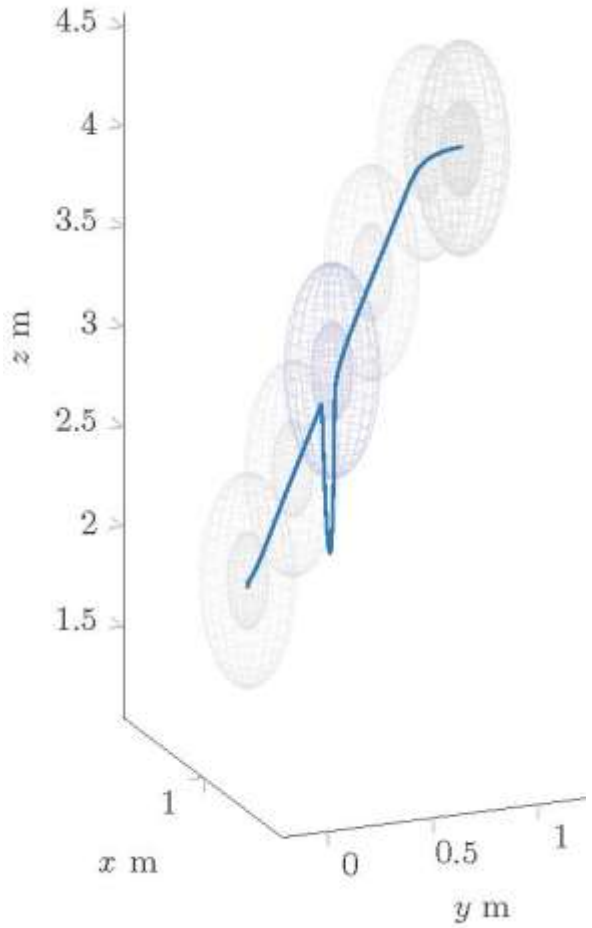
$$\ddot{p}_y = \sin\phi \frac{F}{m}$$

$$\ddot{p}_z = g - \cos\phi \cos\theta \frac{F}{m}$$

$$\ddot{\phi} = \frac{1}{J_x}\tau_\phi$$

$$\ddot{\theta} = \frac{1}{J_y}\tau_\theta$$

$$\ddot{\psi} = \frac{1}{J_z}\tau_\psi.$$




Linear design:

- linearize at equilibrium
- assume full state available
- LQ state feedback design
- reference points = equilibrium states

# Drone Experiment

# Are We Done Yet?

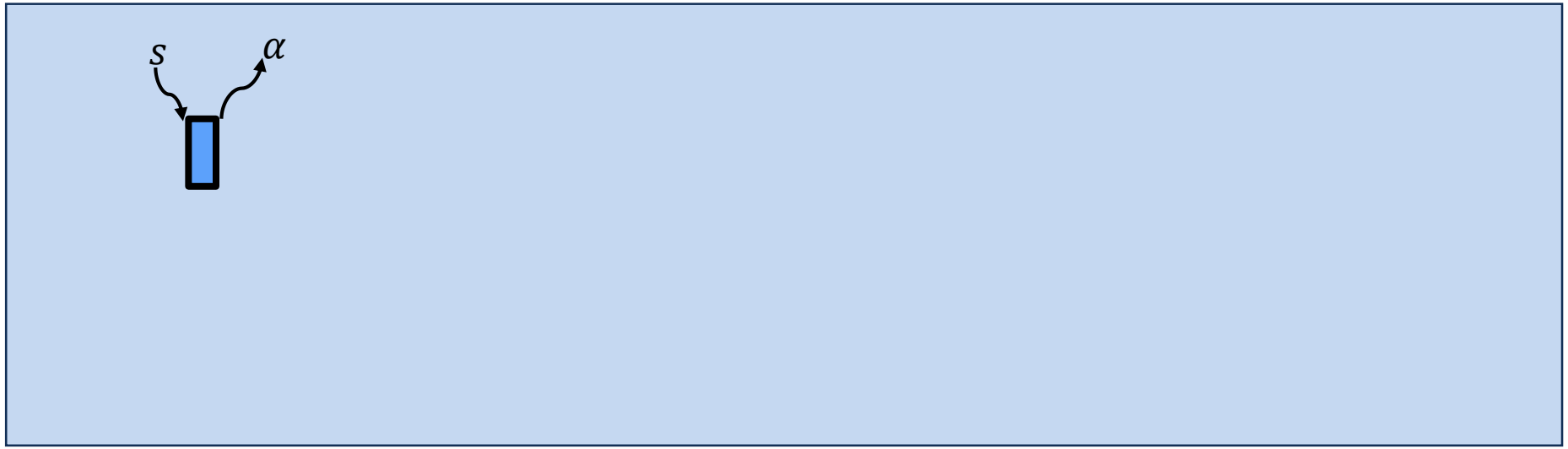
Scalable Verification

- Only verify safety-critical components
- Guarding unverified components

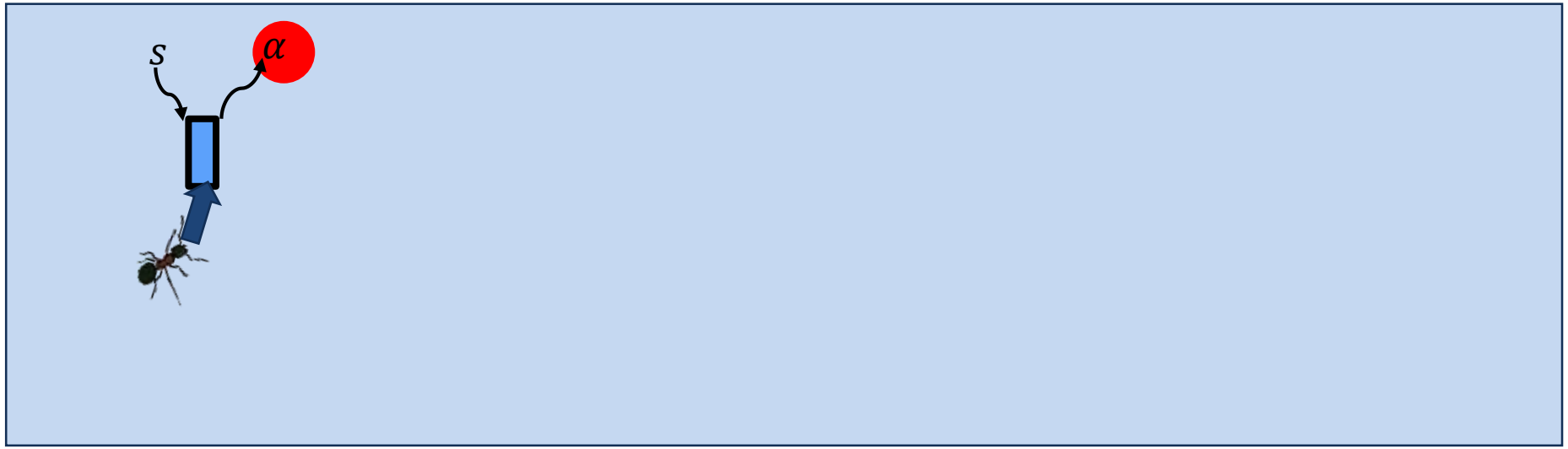
Trust

- Protect verified components
- Against attacks or bugs from unverified components

# Enforcing Unverified Components



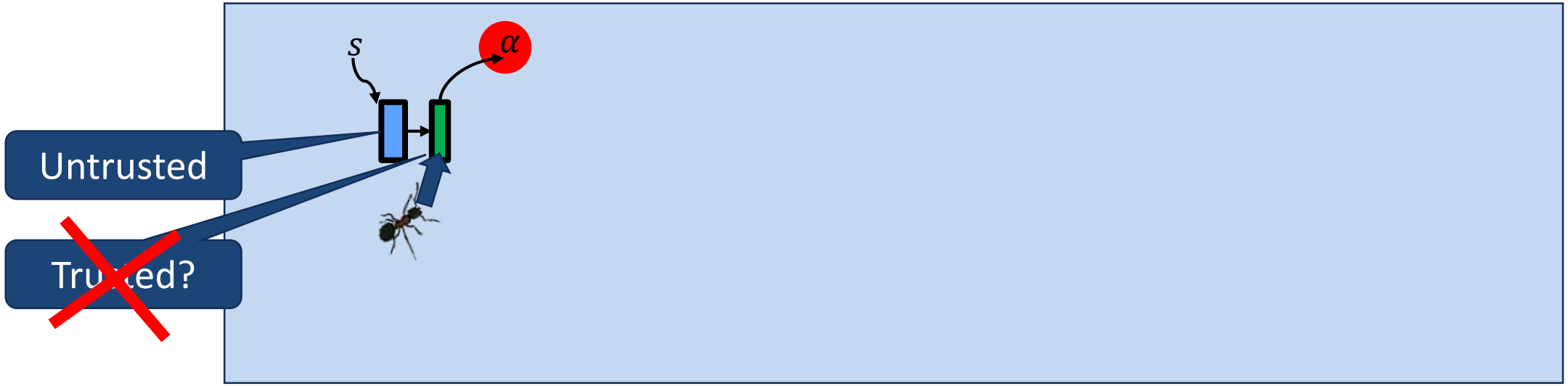
# Enforcing Unverified Components



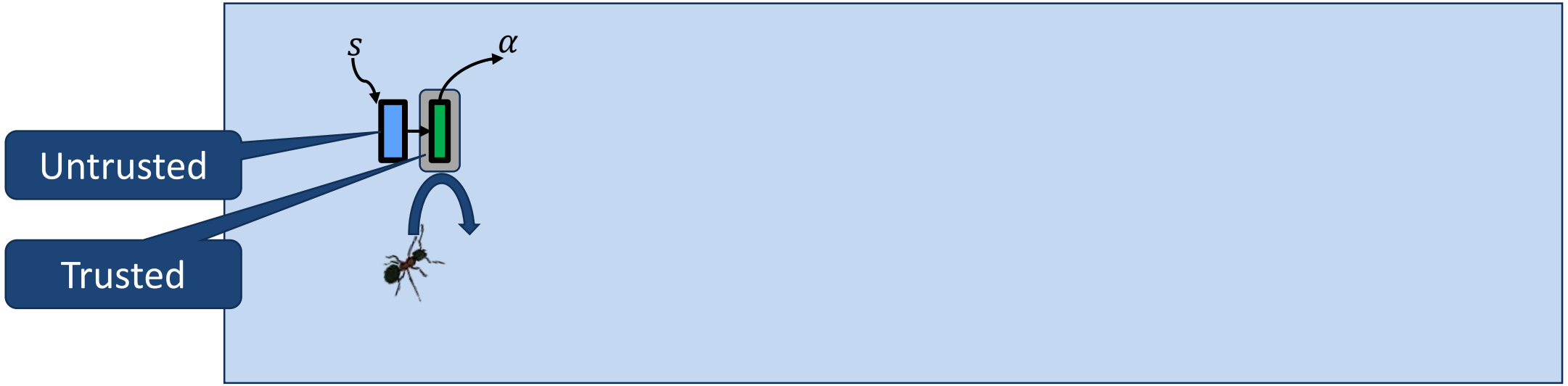

Ant illustration by Jan Gillbank, license by Creative Commons Attribution 3.0 Unported

# Enforcing Unverified Components

# But enforcer can be corrupted (bug or cyber attack)



# Add Memory Protection




Trusted = Verified & Protected

# Are We Done Yet?

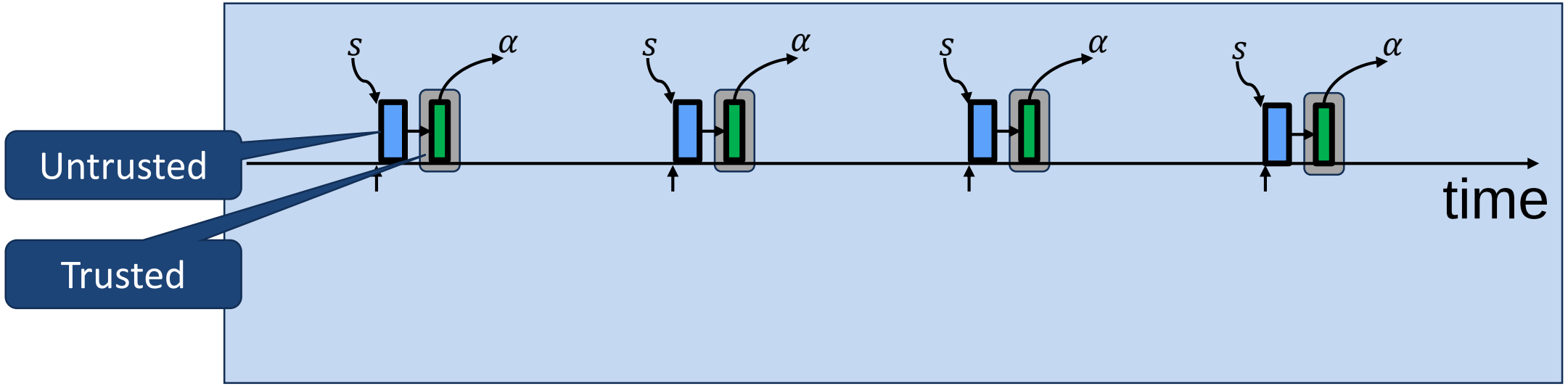
Timing can still be corrupted

- Guaranteed correct value
- BUT potentially at wrong time

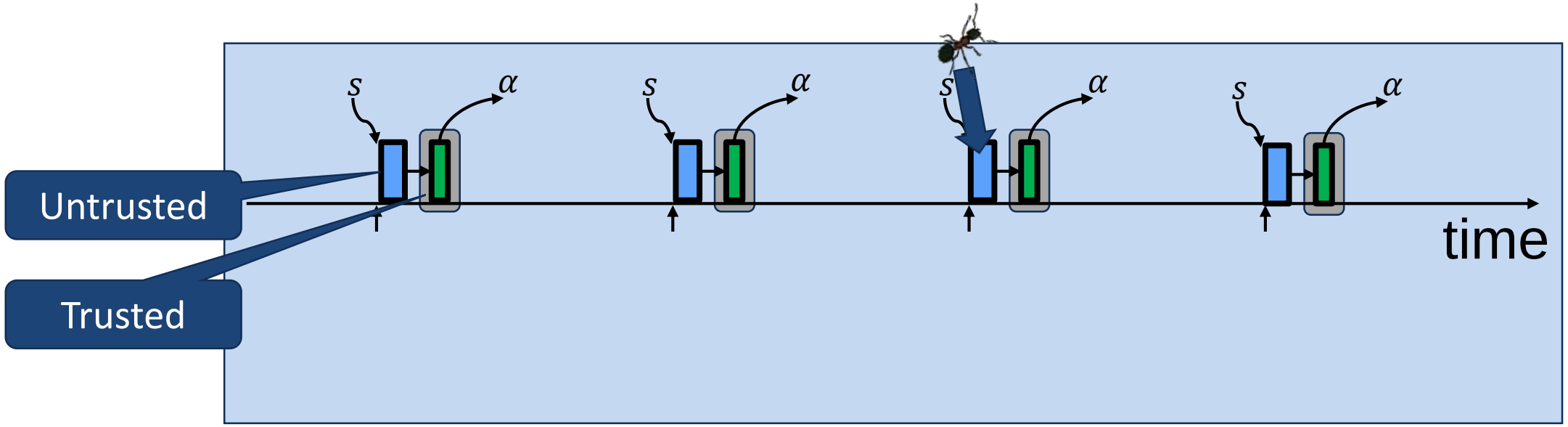
Trusted timely actuation

- Tamper-proof time-triggering mechanism
- In sync with periodic controller
- In sync with expected untrusted

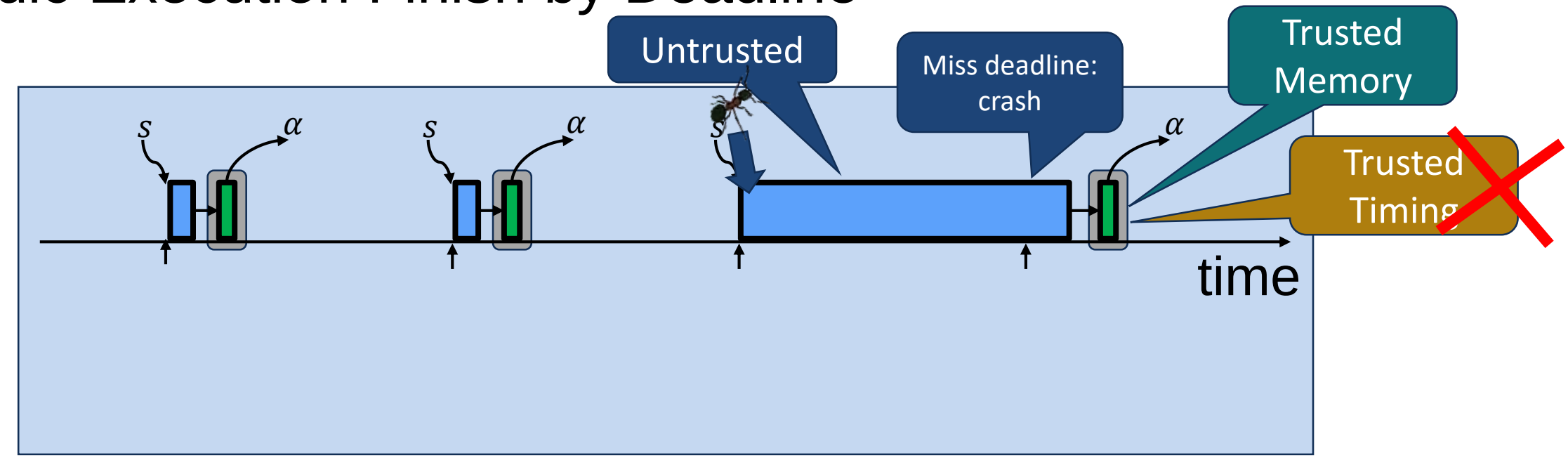
# Periodic Execution Must Finish by Deadline



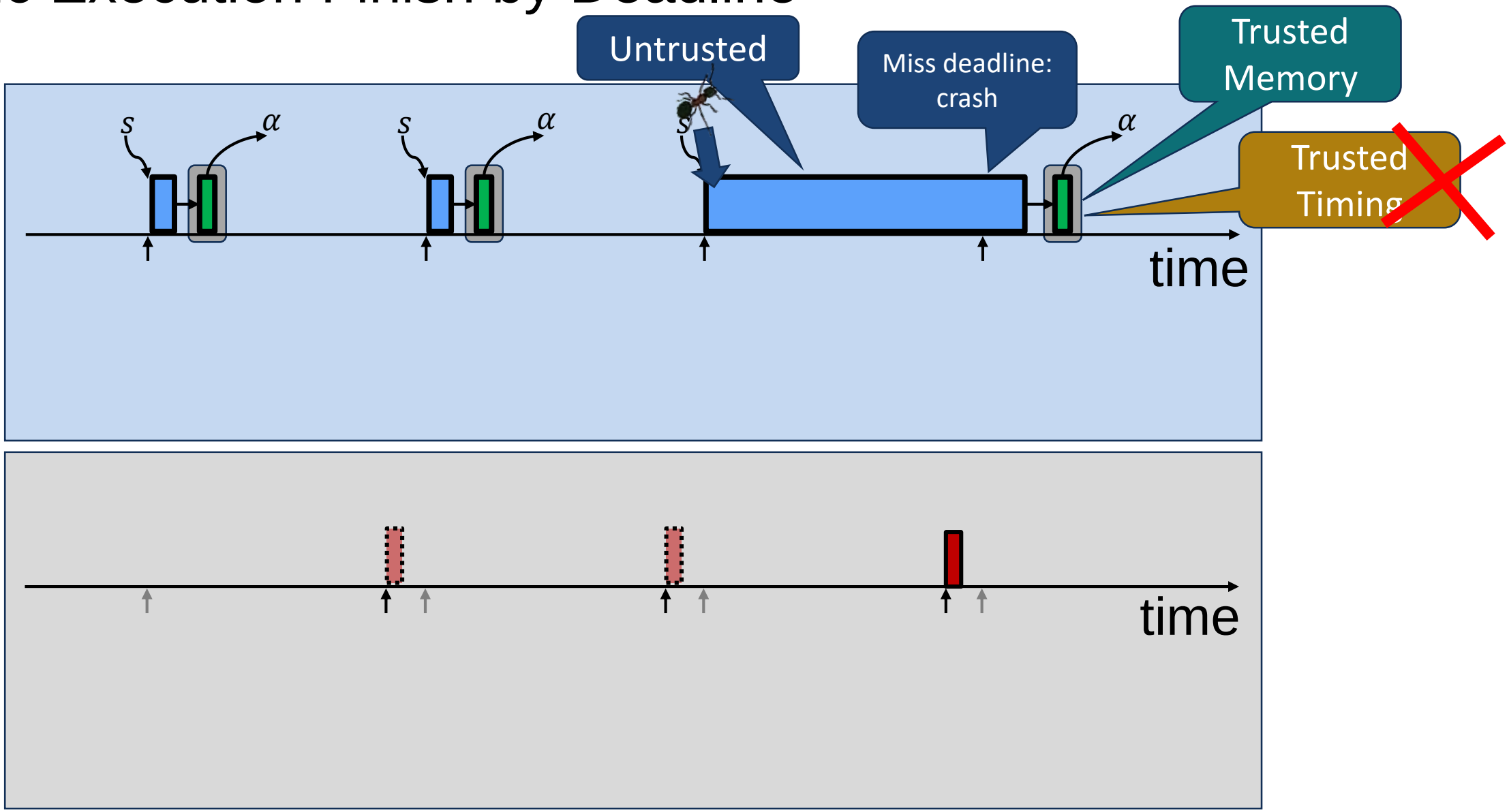
# Periodic Execution Must Finish by Deadline



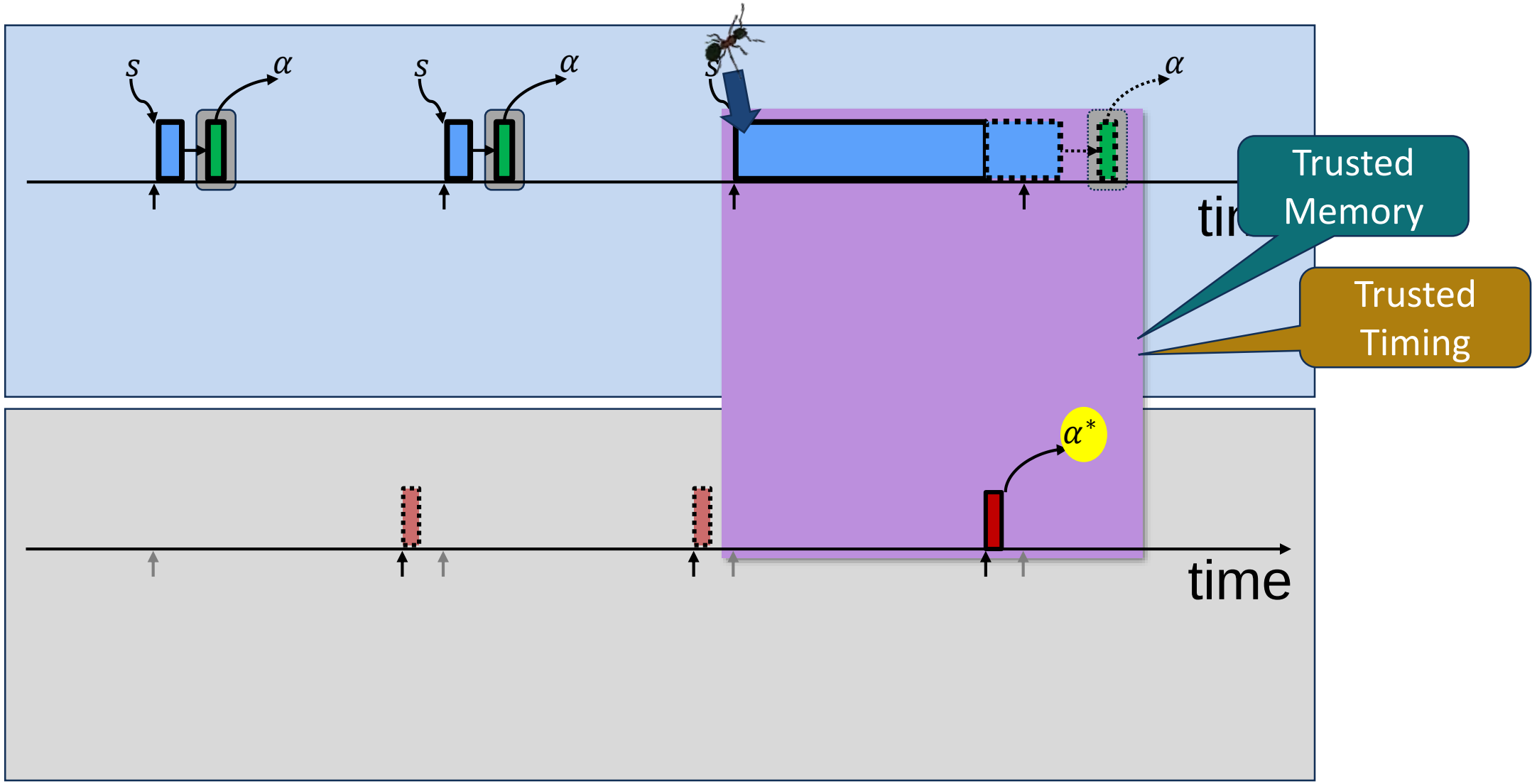
# Periodic Execution Finish by Deadline



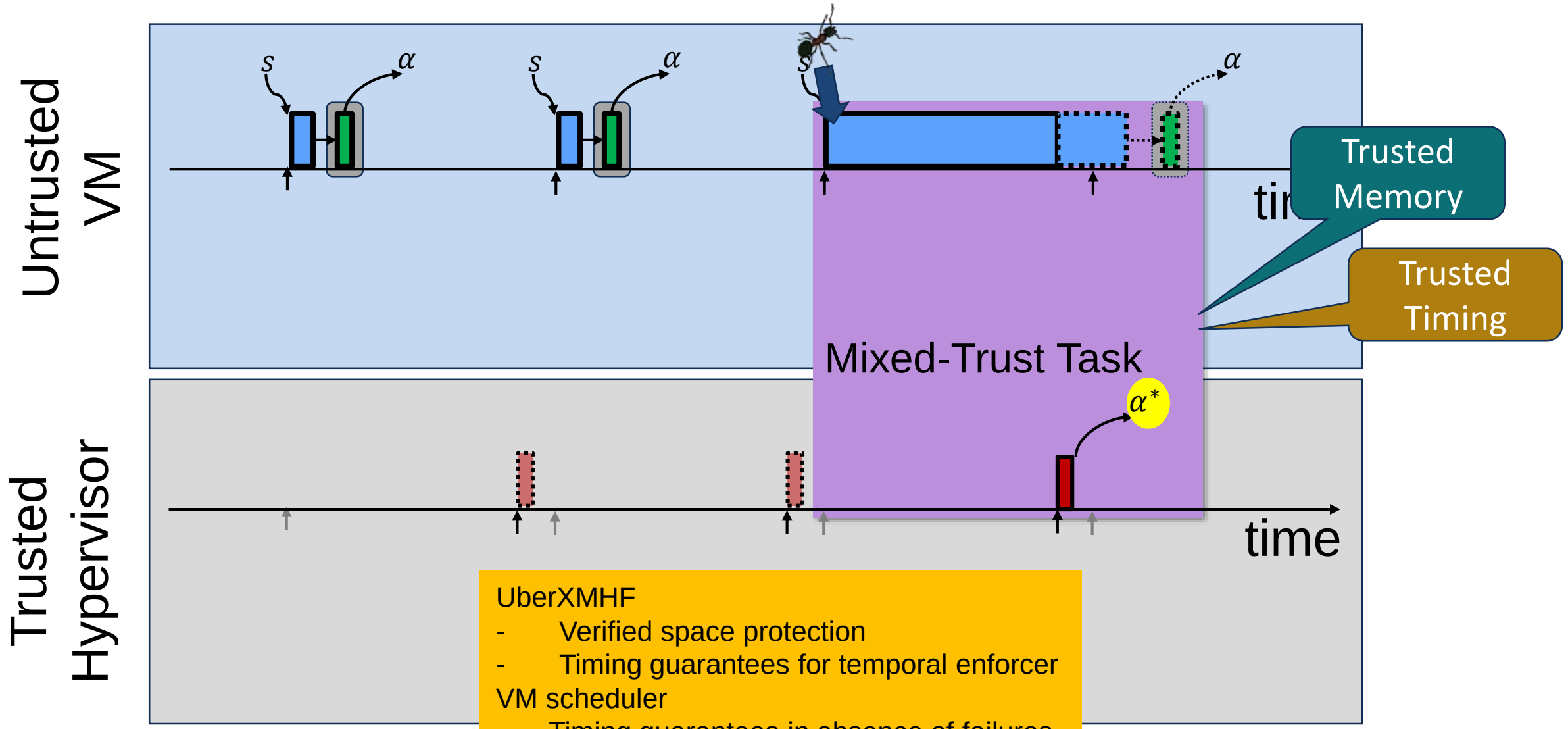
# Periodic Execution Finish by Deadline



# Periodic Execution Finish by Deadline



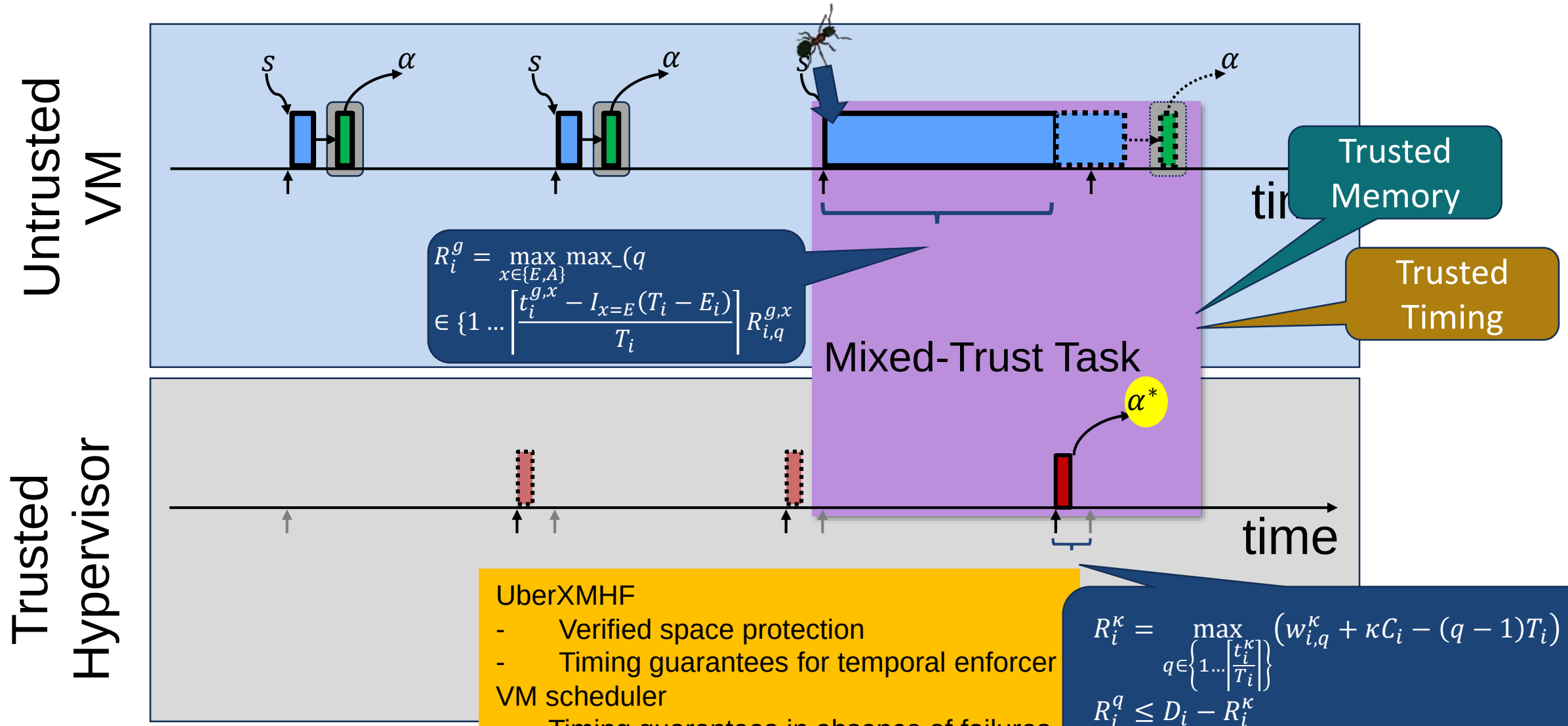
# Real-Time Mixed-Trust Computation




UberXMHF
- Verified space protection
- Timing guarantees for temporal enforcer

VM scheduler
- Timing guarantees in absence of failures
- In sync with hypervisor scheduler

# Real-Time Mixed-Trust Computation




UberXMHF
- Verified space protection
- Timing guarantees for temporal enforcer

VM scheduler
- Timing guarantees in absence of failures
- In sync with hypervisor scheduler

$$R_i^\kappa = \max_{q\in\left\{1\dots\left\lceil\frac{t_i^\kappa}{T_i}\right\rceil\right\}} \left(w_{i,q}^\kappa + \kappa C_i - (q-1)T_i\right)$$
$$R_i^q \le D_i - R_i^\kappa$$

# Results So Far

Physics verification

- Lyapunov-based analysis of enforcement

Temporal verification

- Guaranteed timing even in presence of bugs/attacks

Logical verification

- Verified hypervisor with space and time protection

# Demos

Lyapunov-enforced controller in open source drone code (PX4)

- Running in Hardware in the Loop
- Coded in DIY drone

Real-Time Mixed-Trust Framework in DIY drone with Raspberry Pi-3 + PX4

- UberXMHF hypervisor + Mixed-Trust HV Scheduler
- VM with Linux + Mixed-Trust VM Scheduler

# Publications

R. Romagnoli, B.H. Krogh, and B. Sinopoli. Design of Software Rejuvenation for CPS Security Using Invariant Sets. *American Control Conference (ACC).*July, 2019.

R. Romagnoli, B. H. Krogh and B. Sinopoli. Safety and Liveness of Software Rejuvenation for Secure Tracking Control. *2019 18th European Control Conference (ECC).* June, 2019.

D. de Niz, B. Andersson, M. Klein, J. Lehoczky, A. Vasudevan, H. Kim, and G. Moreno. Mixed-Trust Computing for Real-Time Systems. *25th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA).* August, 2019.

# Road ahead

Optimization of mission performance

- Absolute safety guarantees (worst case)
- Long-term mission performance (average)

Optimize cross-domain assumptions

- Control theory analysis of deadline-miss tolerance
  - Improve utilization with timing guarantees

Optimize inter-component assumptions

- Identify inter-component assumption conflicts
- Identify or eliminate inactive assumptions and conflicts
  - Not required for system-wide guarantees

# Transition

| NEAR | MID | FAR |
| --- | --- | --- |
| Apply mixed-trust architecture to Navy system | Demonstrate proof-of-concept to Navy (FY 2021)<br>• Physics verification<br>• Timing verification<br>Explore application in other CPS defense systems<br>• e.g., MDA | Evaluate transition to running system (FY 2023-24)<br>Extension & application to Autonomous systems |

# Team and Collaborators

Dr. Bruce Krogh

Dr. Gabriel Moreno

Dr. Bjorn Andersson

Dr. Amit Vasudevan

Dr. Jeffery Hansen

Anton Hristozov

Mark Klein

Dr. Dionisio de Niz

Dr. Raffaele Romagnoli (CMU / ECE)

Prof. John Lehoczky (CMU / Statistics)

Prof. Bruno Sinopoli (WUSL / ECE)