YOLO
Frequently Resetting CPS for Security

Miguel A. Arroyo, **M. Tarek Ibn Ziad**, Hidenori Kobayashi, Junfeng Yang, Simha Sethumadhavan

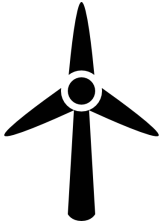
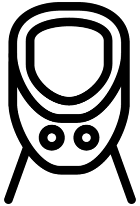
COLUMBIA UNIVERSITY
IN THE CITY OF NEW YORK

YOLO

You **O**nly **L**ive **O**nce

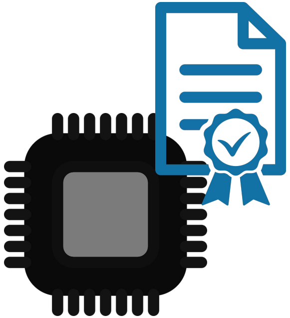


Cyber-Physical Systems = Cyber + Physical


CPS Characteristics (vs Cyber)

- More vulnerable to attacks
  - Not designed for security
  - Slow to no upgrades
- More difficult to recover from failures
  - Replacing hardware is non-trivial



CPS Characteristics (vs Cyber)

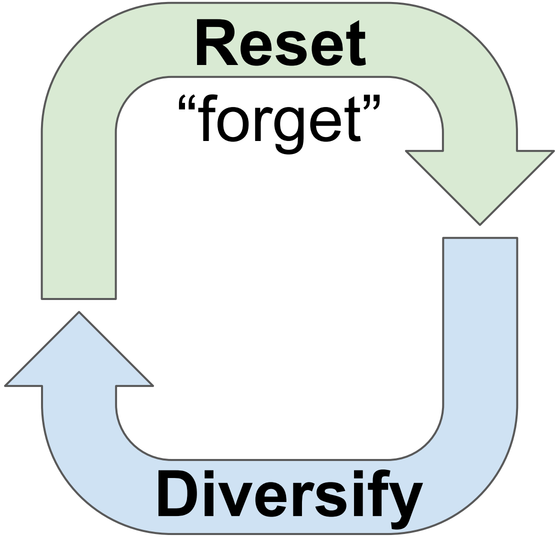
- Resilient by design
  - Redundancy against unintentional failures/faults

Key Research Question

Can we take advantage of unique CPS properties to protect them against security attacks?

YOLO in a nutshell

- Leverage ***physical*** characteristics of CPS to ensure ***cyber*** security.
- Flexible framework that can be integrated for a varying spectrum of systems.

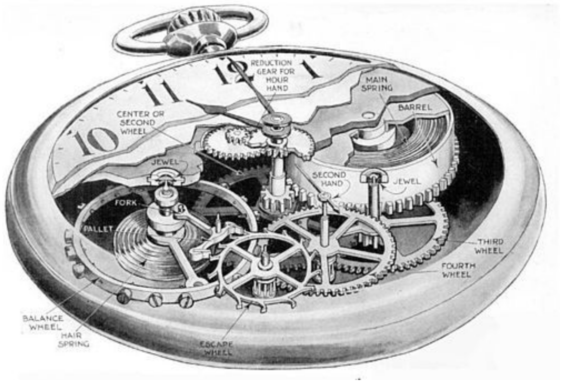


YOLO: Threat Model

- Attacker's intention is to gain a foothold into the system.

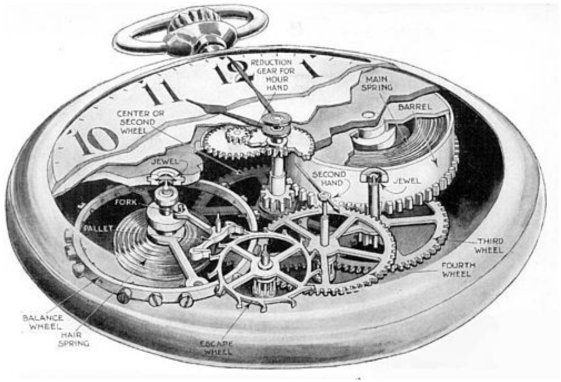
YOLO: Threat Model

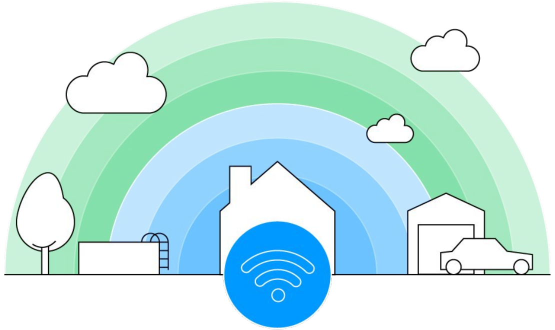
- Attacker's intention is to gain a foothold into the system.
- An attacker has complete knowledge of the system internals.



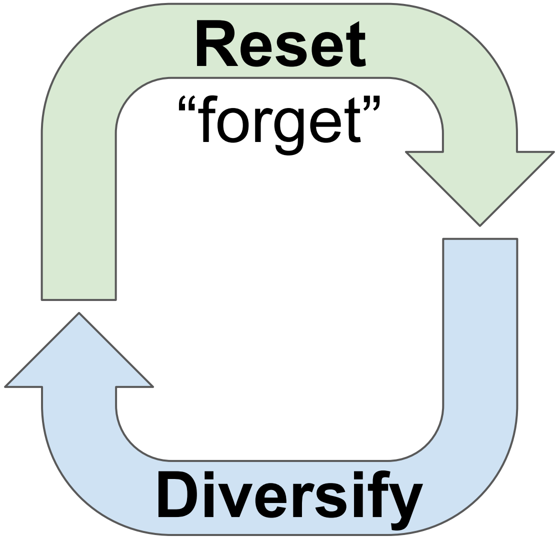
YOLO: Threat Model

- Attacker’s intention is to gain a foothold into the system.
- An attacker has complete knowledge of the system internals.
- An attacker’s sphere of influence is bounded.

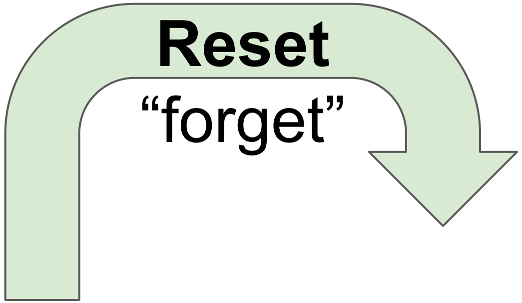




YOLO in a nutshell

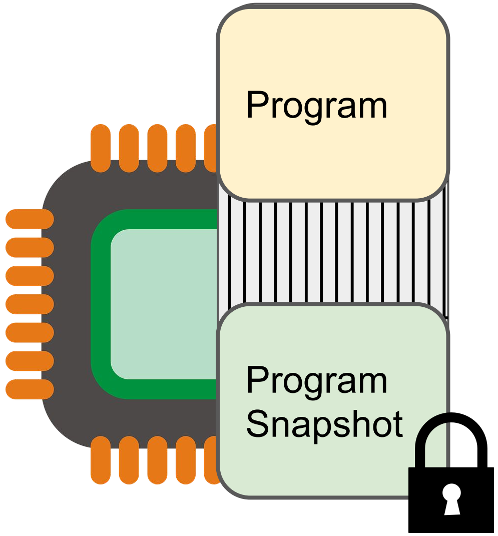


YOLO in a nutshell



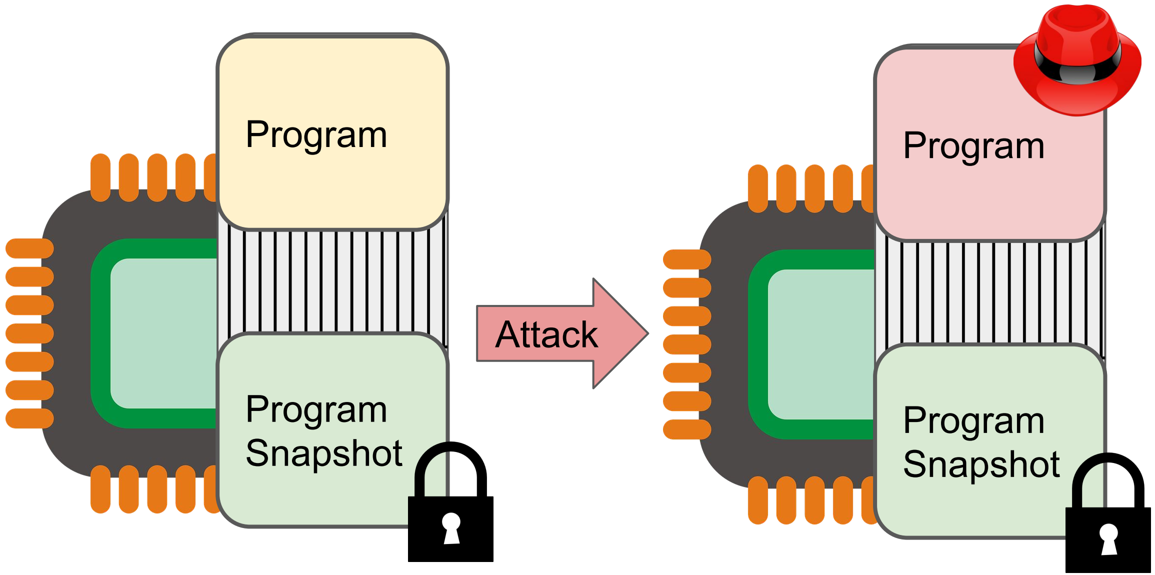
YOLO: You Only Live Once

- Why Reset?
  - Prevents an adversary's ability to corrupt the system.
    - Bounded time horizon over which an attacker can affect the system.



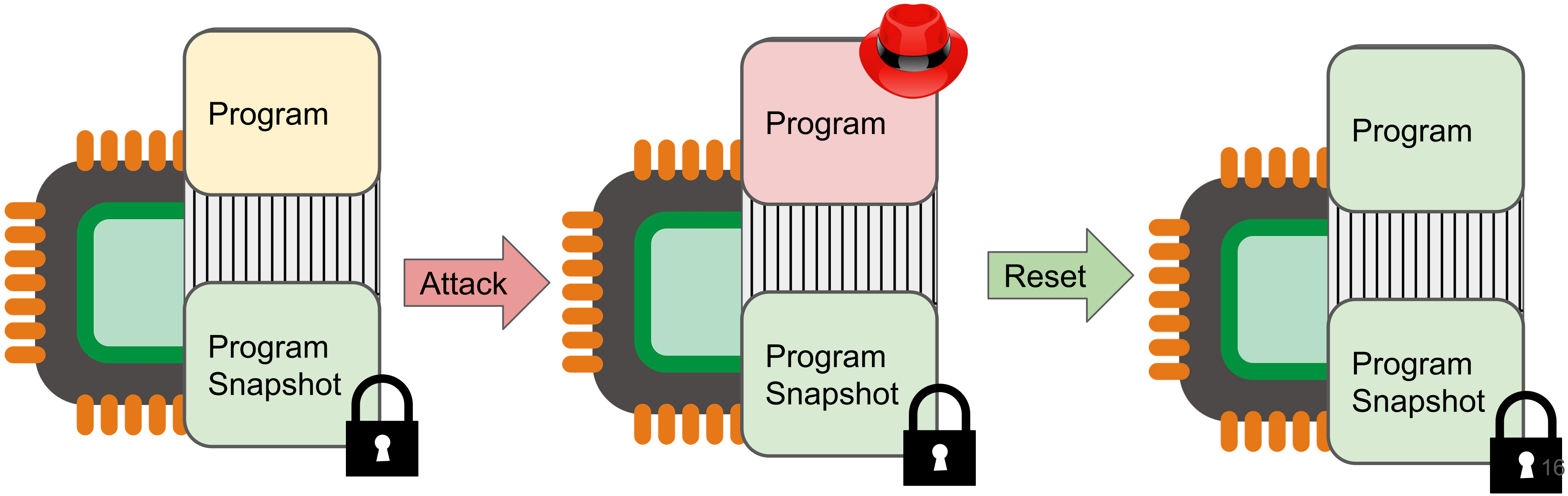
YOLO: You Only Live Once

- Why Reset?
  - Prevents an adversary's ability to corrupt the system.
    - Bounded time horizon over which an attacker can affect the system.

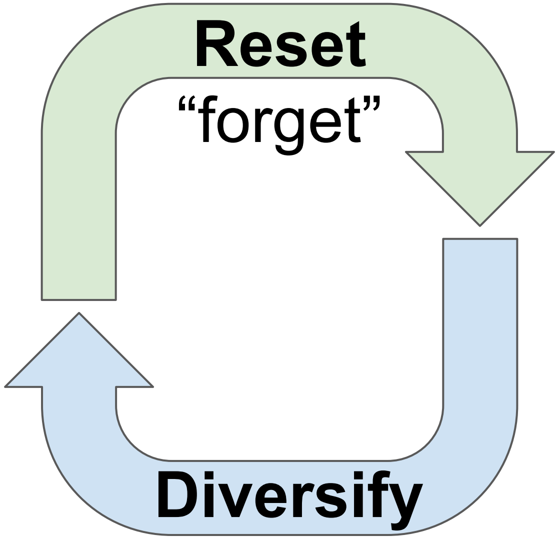


YOLO: You Only Live Once

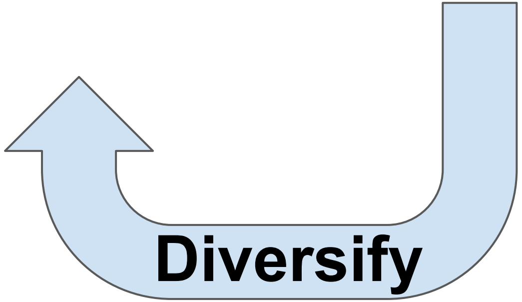
- Why Reset?
  - Prevents an adversary’s ability to corrupt the system.
    - Bounded time horizon over which an attacker can affect the system.



YOLO in a nutshell

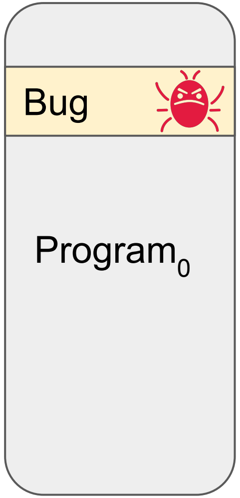


YOLO in a nutshell



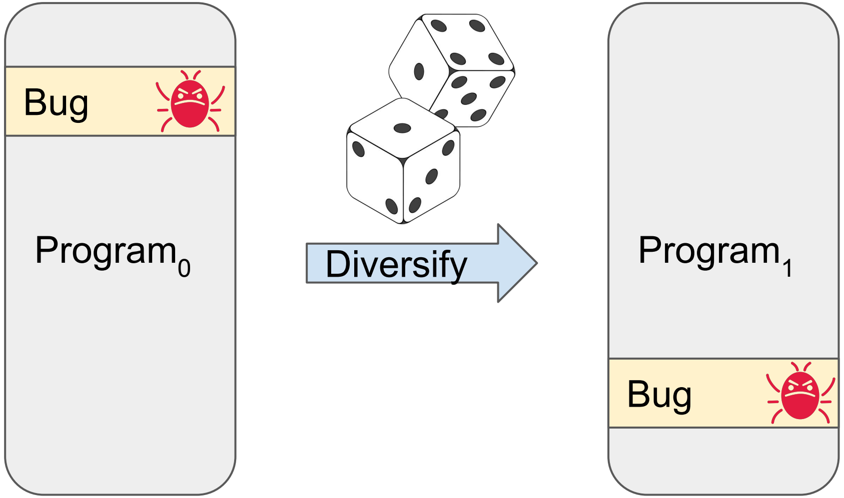
YOLO: You Only Live Once

- Why Diversify?
  - Introduce randomness to prevent the system from being compromised by the same method continuously.
    - Reduce chance of attacker success.



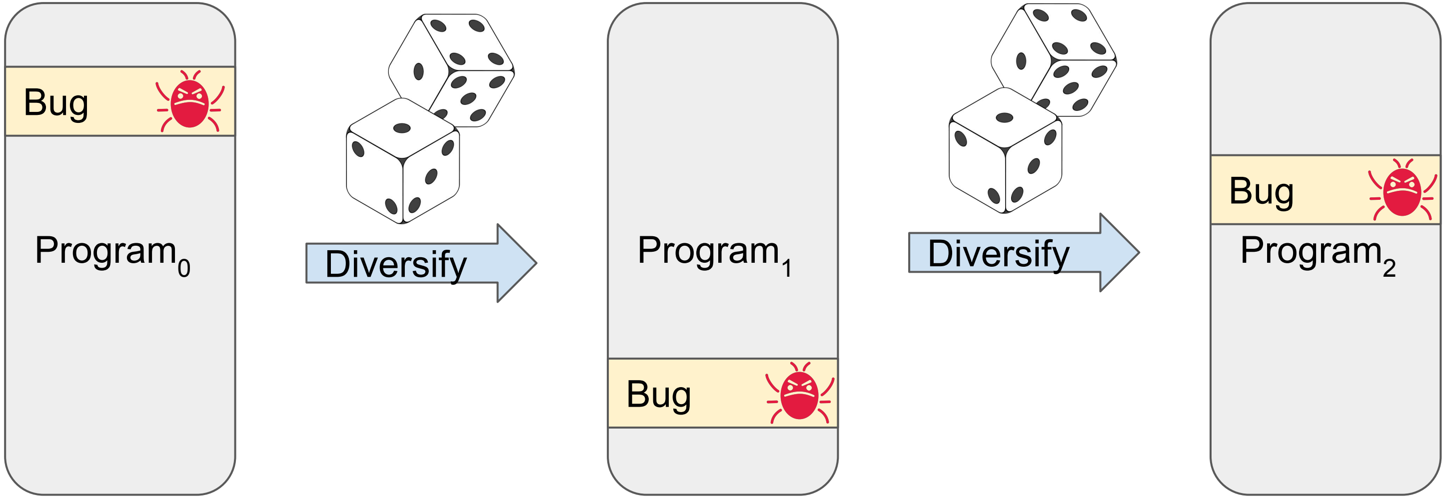
YOLO: You Only Live Once

- Why Diversify?
  - Introduce randomness to prevent the system from being compromised by the same method continuously.
    - Reduce chance of attacker success.



YOLO: You Only Live Once

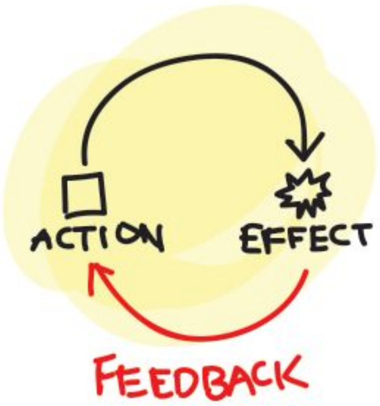
- Why Diversify?
  - Introduce randomness to prevent the system from being compromised by the same method continuously.
    - Reduce chance of attacker success.



YOLO: You Only Live Once

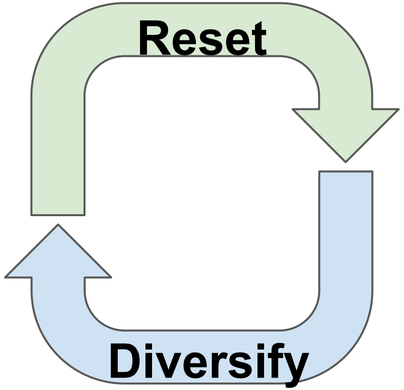
- Why does this work for CPS?

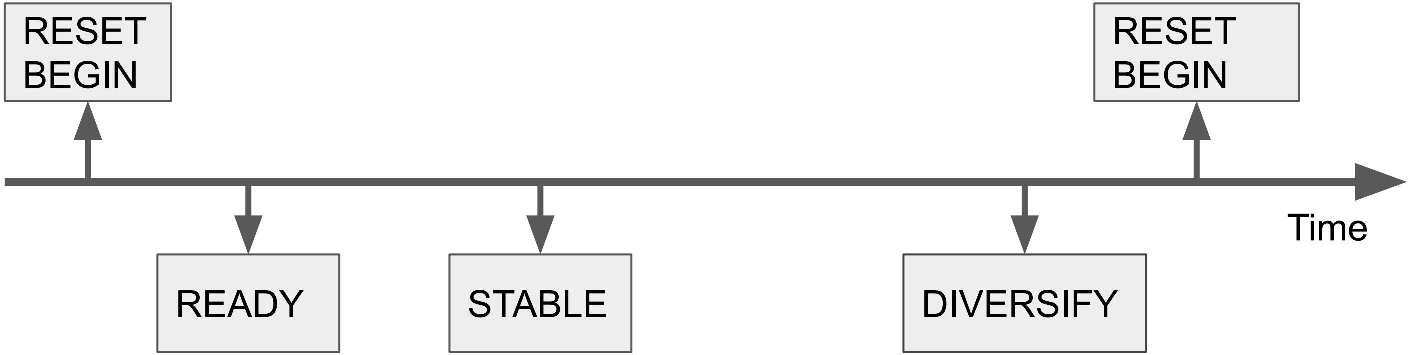
Inertia
Allows system to continue operation.



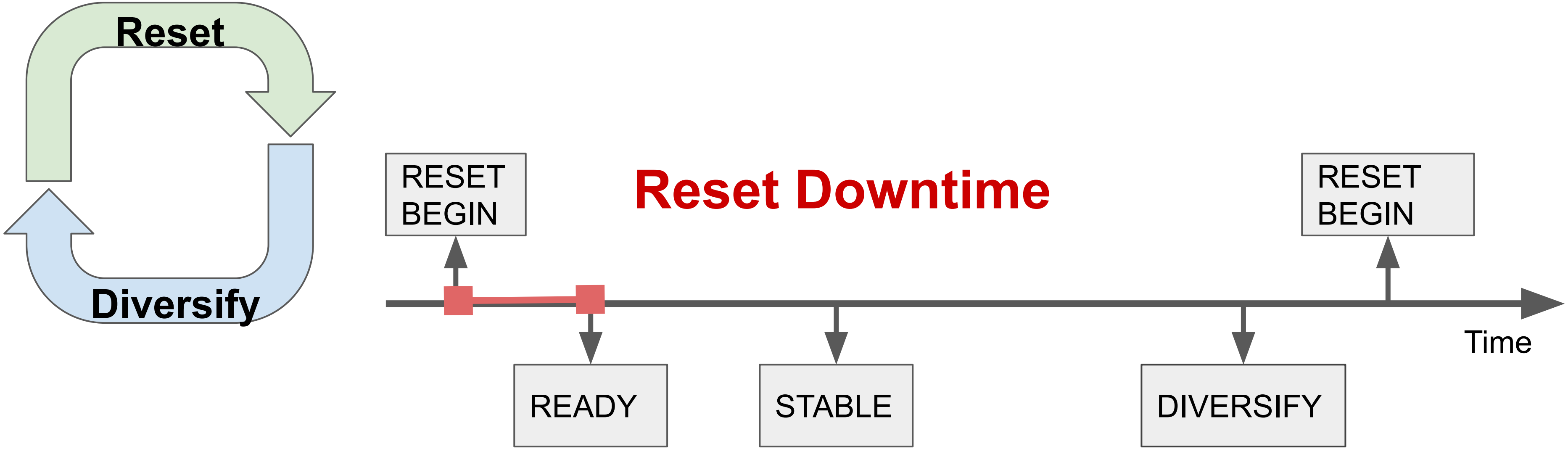
Feedback
The state of the system can be observed.

Why does **YOLO** provide protection?

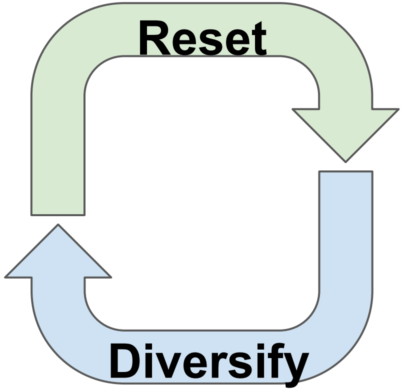


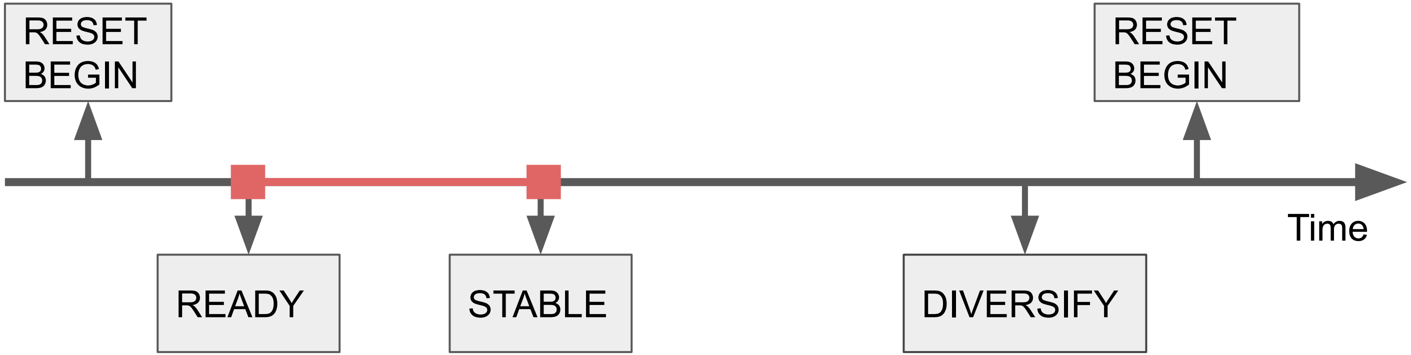


Why does **YOLO** provide protection?

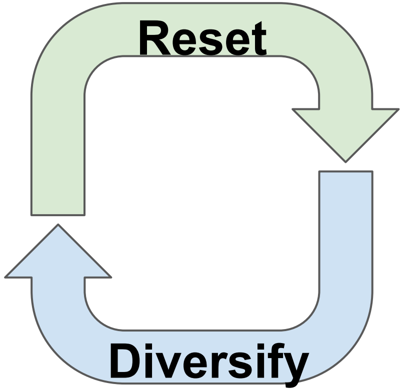


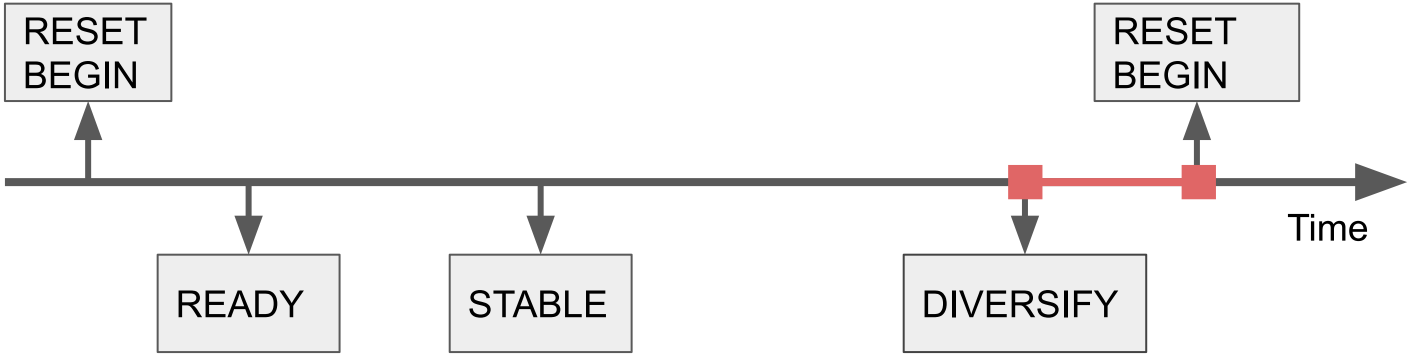
Why does **YOLO** provide protection?



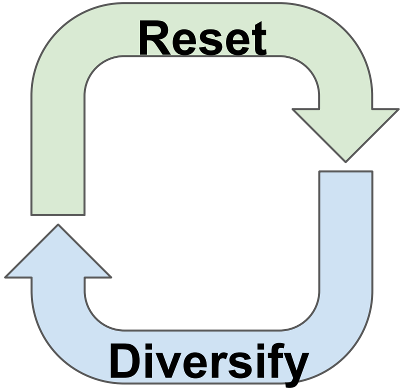


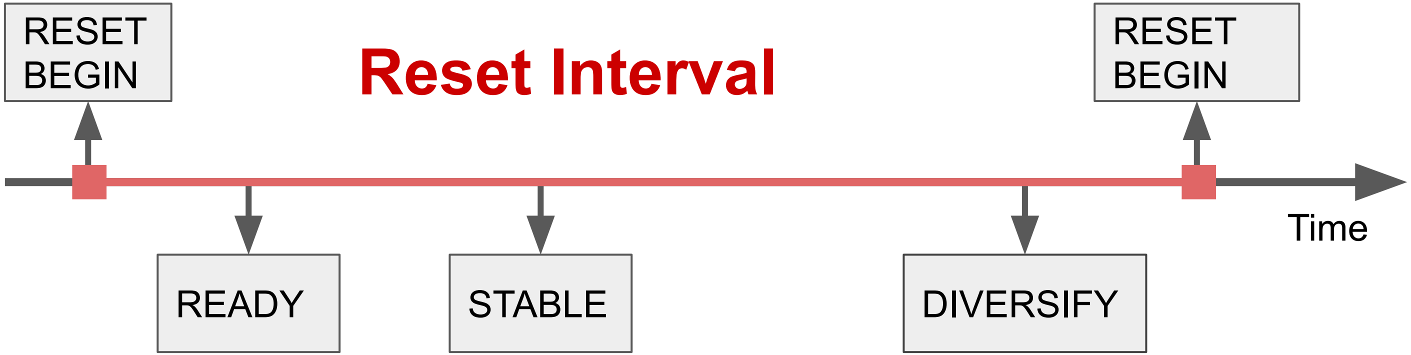
Why does **YOLO** provide protection?





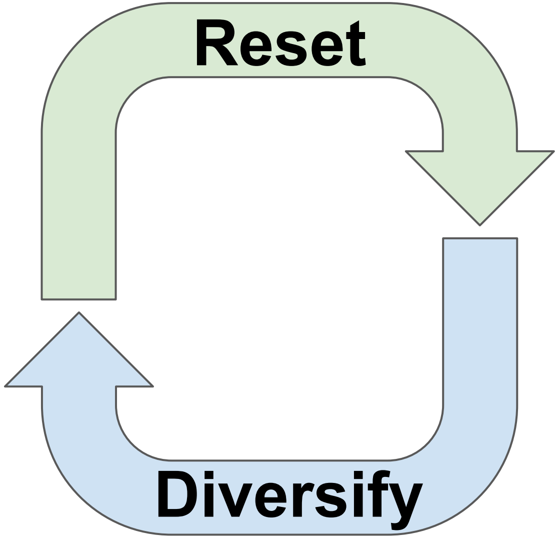
Why does **YOLO** provide protection?





- For YOLO to win: reset interval **<** time for an attacker’s effects to manifest.

Why does **YOLO** provide protection?

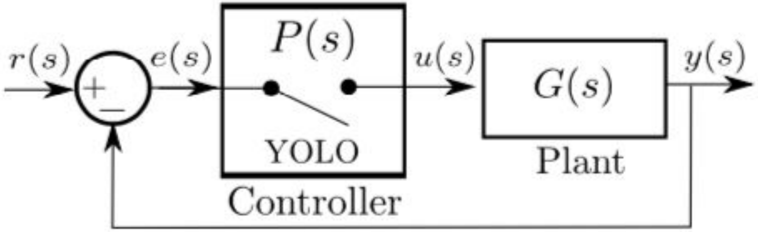


- Persistent malware is denied (**RESET** step)
    - Memory is wiped clean.

- Increased work for the attacker (**DIVERSIFY** step)
    - Inputs have to be crafted to exploit each variant.

Rest of the talk…

A. Theoretical Analysis



B. Case Studies

1. Engine Control Unit (ECU)
2. Flight Controller (FCU)

Theoretical Analysis

Key Research Question

Can a system be stable with YOLO?

YOLO: Theoretical Analysis

- Stability.

Under Ideal Conditions	***Under Adversarial Conditions***
Does YOLO maintain regular stability?	Can YOLO limit the attacker’s effect on stability?

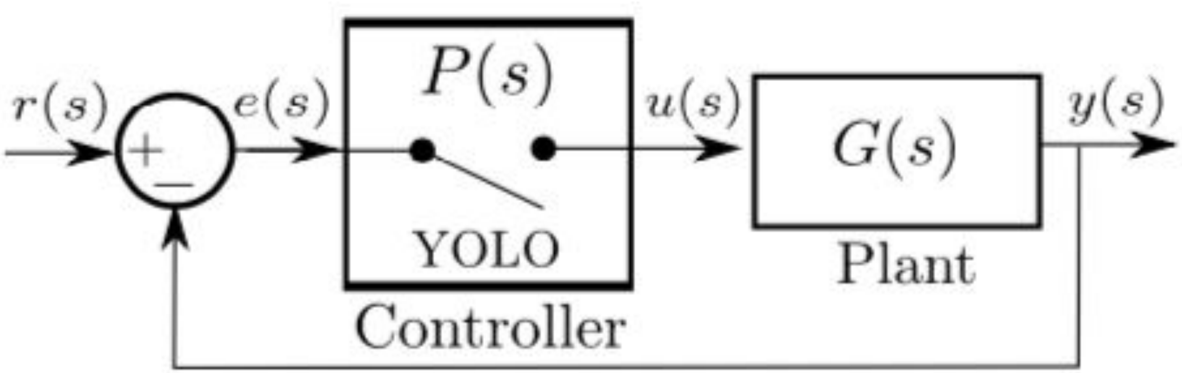
YOLO: Theoretical Analysis

- Stability.

Under Ideal Conditions	***Under Adversarial Conditions***
Does YOLO maintain regular stability?	Can YOLO limit the attacker’s effect on stability?
Yes, various combinations of reset periods possible.	Yes, frequent resetting limits the attacker’s ability to construct solid attacks.

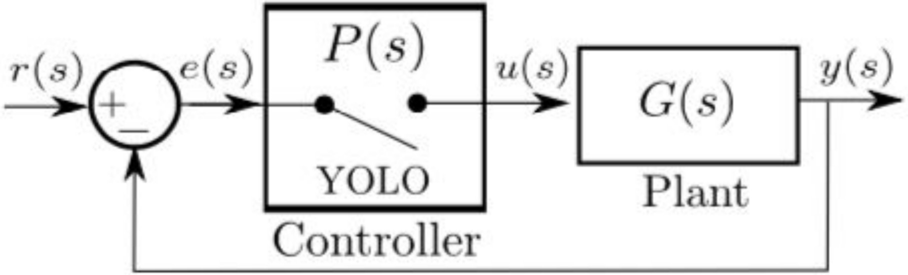
YOLO: Theoretical Analysis

- Problem Formulation.



YOLO: Theoretical Analysis

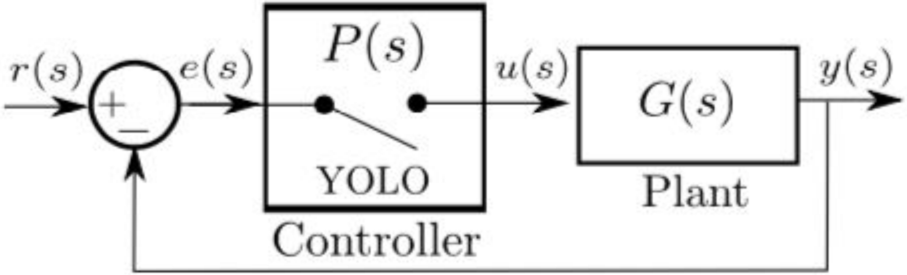
- Problem Formulation.



- YOLO acts as an ON/OFF switch with period `Tr`.
- `Tr = Tu + Td`
  where `Tu` is the controller up-time and `Td` is the controller down-time

YOLO: Theoretical Analysis

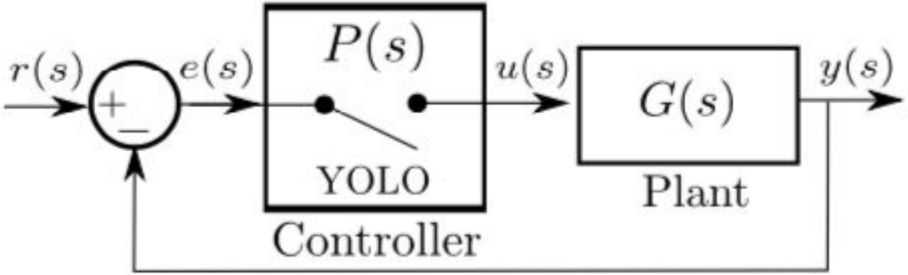
- Problem Formulation.



$$F_{inactive}(s) = \frac{y(s)}{r(s)} = \frac{P(s)G(s)}{1 + P(s)G(s)}$$

YOLO: Theoretical Analysis

- Problem Formulation.

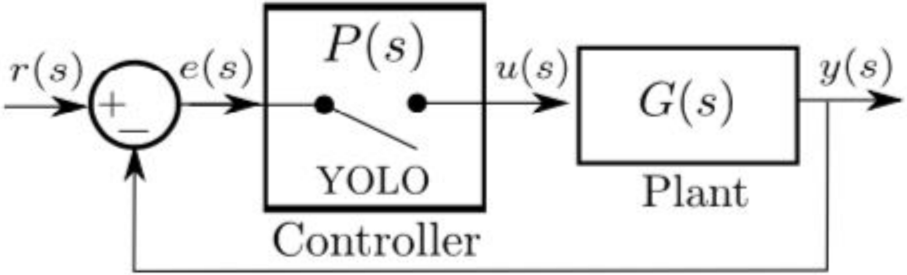


$$F_{inactive}(s) = \frac{y(s)}{r(s)} = \frac{P(s)G(s)}{1 + P(s)G(s)}$$

$$F_{active}(s) = \frac{y(s)}{u(s)} = G(s)$$

YOLO: Theoretical Analysis

- Problem Formulation.



$$\begin{cases} \dot{x} = A_i x + B_i r, \\ y = C_i x + D_i r \end{cases}$$

YOLO: Theoretical Analysis

- Stability Analysis.
  - Prior work uses **Lyapunov** functions.
  - They prove the stability of dynamic systems **without** requiring the actual solution of the system's ODEs to be available.

YOLO: Theoretical Analysis

- Stability Analysis.
  - Prior work uses **Lyapunov** functions.
  - They prove the stability of dynamic systems **without** requiring the actual solution of the system's ODEs to be available.
  - We adopt the average "dwell time", $\boldsymbol{\tau}$, approach proposed by Zhai et al. [1].

[1] Zhai, G., Hu, B., Yasuda, K., and Michel, A. N., "Stability analysis of switched systems with stable and unstable subsystems: an average dwell time approach," in [American Control Conference], 1, 200–204 (Sep 2000).

YOLO: Theoretical Analysis

- Stability Analysis.
  - We adopt the average “dwell time”, τ, approach proposed by Zhai et al. [1].
  - Stability conditions:

$$\tau \geq \frac{a}{\lambda^* - \lambda}$$

$$\frac{T_u}{T_d} \geq \frac{\lambda^+ + \lambda^*}{\lambda^- - \lambda^*}$$

[1] Zhai, G., Hu, B., Yasuda, K., and Michel, A. N., “Stability analysis of switched systems with stable and unstable subsystems: an average dwell time approach,” in [American Control Conference], 1, 200–204 (Sep 2000).

YOLO: Theoretical Analysis

- Stability Analysis.
  - We adopt the average "dwell time", τ, approach proposed by Zhai et al. [1].
  - Stability conditions:

$$\tau \geq \frac{a}{\lambda^* - \lambda}$$

$$\frac{T_u}{T_d} \geq \frac{\lambda^+ + \lambda^*}{\lambda^- - \lambda^*}$$

$$\begin{cases} \|e^{A_i t}\| \leq e^{a_i - \lambda_i t} & i \in \mathbb{S} \\ \|e^{A_i t}\| \leq e^{a_i + \lambda_i t} & i \in \mathbb{U} \end{cases}$$

[1] Zhai, G., Hu, B., Yasuda, K., and Michel, A. N., "Stability analysis of switched systems with stable and unstable subsystems: an average dwell time approach," in [American Control Conference], 1, 200–204 (Sep 2000).

YOLO: Theoretical Analysis

- Stability Analysis.
  - We adopt the average “dwell time”, τ, approach proposed by Zhai et al. [1].
  - We use piecewise Lyapunov function, $V(x) = x^T P_i x$

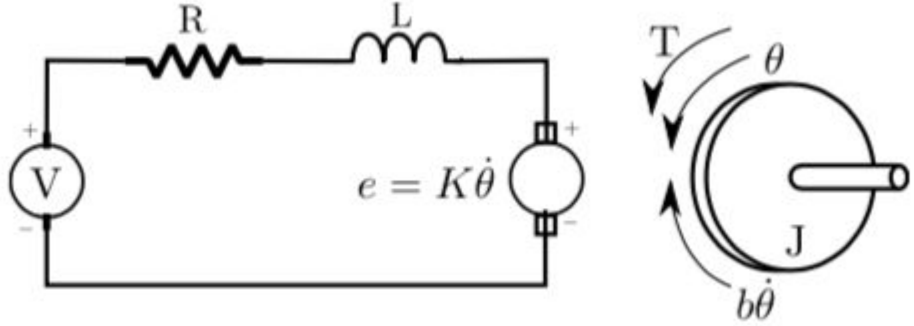
    where P_i are positive definite symmetric matrices, $P_i \in R_n$, which are directly obtainable by solving the linear matrix inequalities (LMIs)

$$\begin{cases} (A_i + \lambda_i I)^T P_i + P_i(A_i + \lambda_i I) < 0 & i \in \mathbb{S} \\ (A_i - \lambda_i I)^T P_i + P_i(A_i - \lambda_i I) < 0 & i \in \mathbb{U} \end{cases}$$

[1] Zhai, G., Hu, B., Yasuda, K., and Michel, A. N., “Stability analysis of switched systems with stable and unstable subsystems: an average dwell time approach,” in [American Control Conference], 1, 200–204 (Sep 2000).

YOLO: Theoretical Analysis

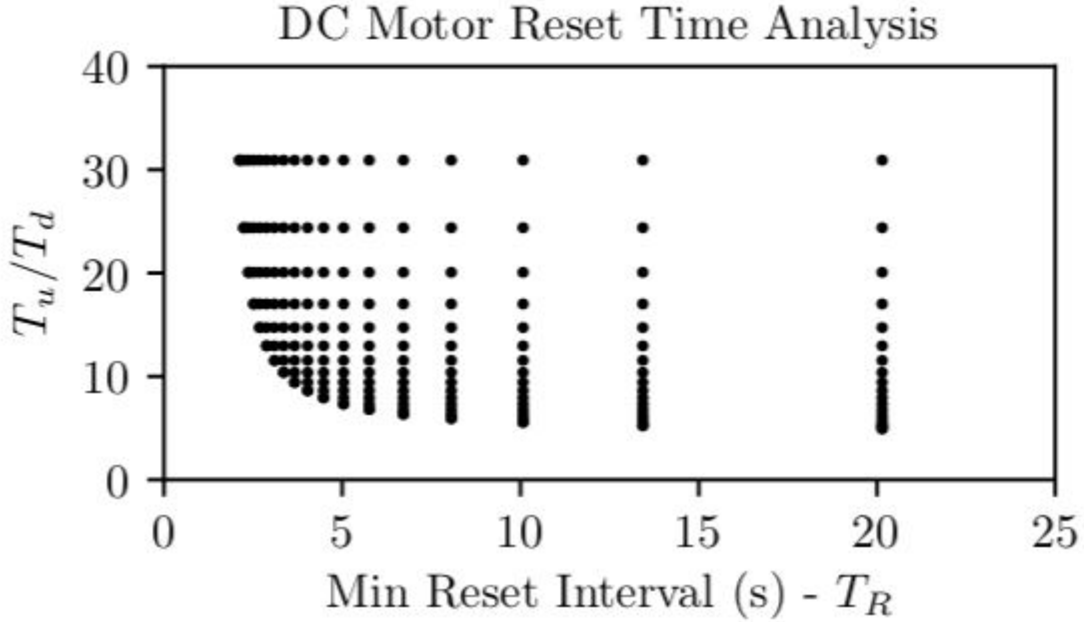
- Case Study.
  DC motor with PID controller.



$$F_{inactive}(s) = \frac{K_oK_ds^2 + K_oK_ps + K_oK_i}{(T_o + K_oK_d)s^2 + (1 + K_oK_p)s + K_oK_i} \qquad F_{active}(s) = \frac{K_o}{T_os + 1}$$

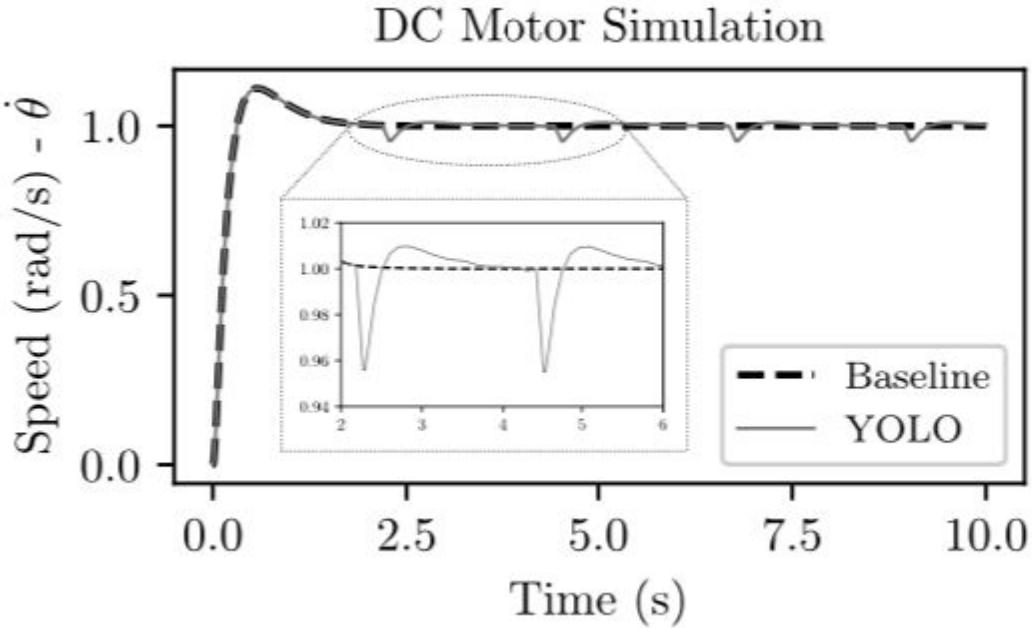
YOLO: Theoretical Analysis

- Case Study: MATLAB Simulation Results



YOLO: Theoretical Analysis

- Case Study: MATLAB Simulation Results



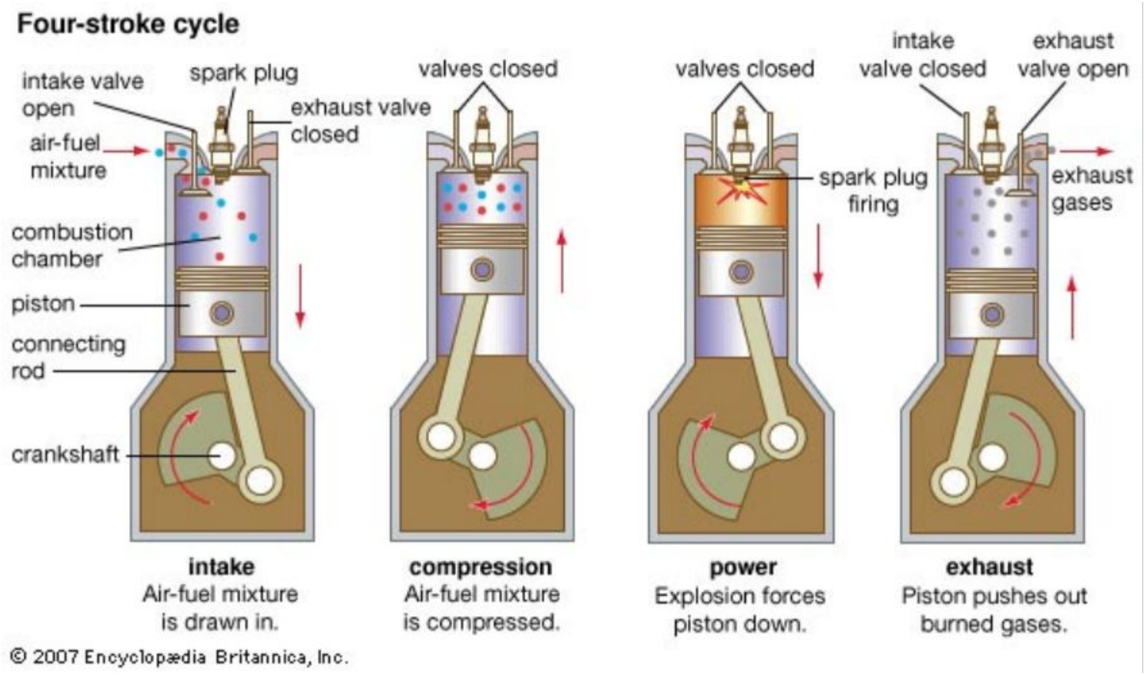

LOOKS GOOD
WILL IT WORK?
memegenerator.net

THE REAL WORLD

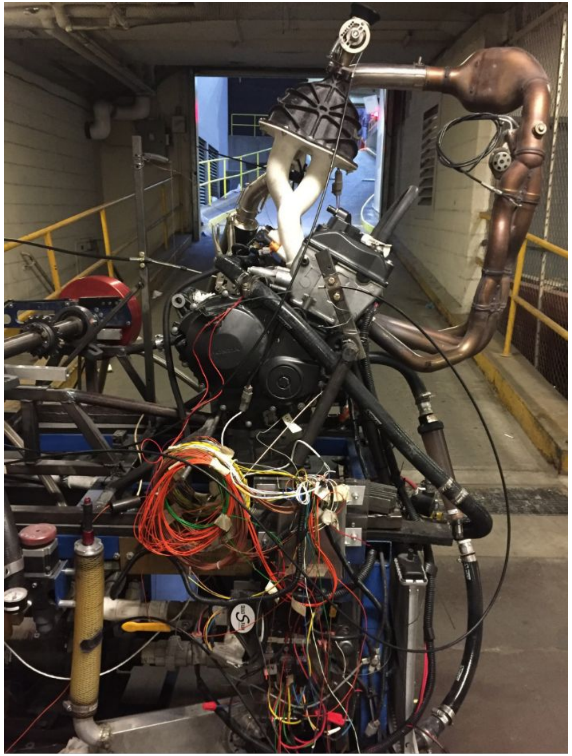
Case Study - **ECU**

Case Study - **ECU**

How it works



Case Study - **ECU**



- rusEFI: Open Source ECU
    - C/C++
- Honda CBR600RR Engine
- Cortex M4 @168 MHz
    - 192 KB SRAM
    - 1 MB Flash

Case Study - **ECU**

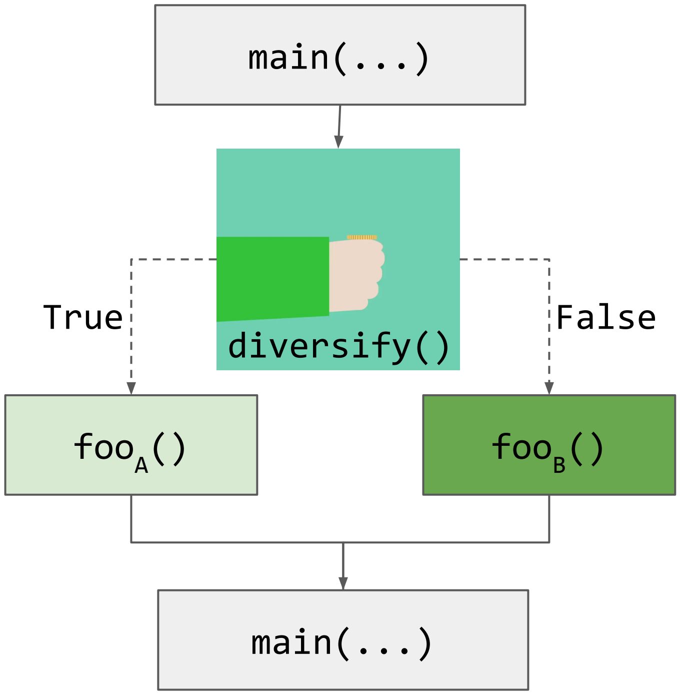
Reset Strategy

- Power cycle.
  - Externally triggerable.
  - Clears RAM & peripheral state.

Case Study - **ECU**

Diversify Strategy

- Build off technique called *Isomeron* [1].
  - Execution-path randomization.
  - Compile-time implementation.

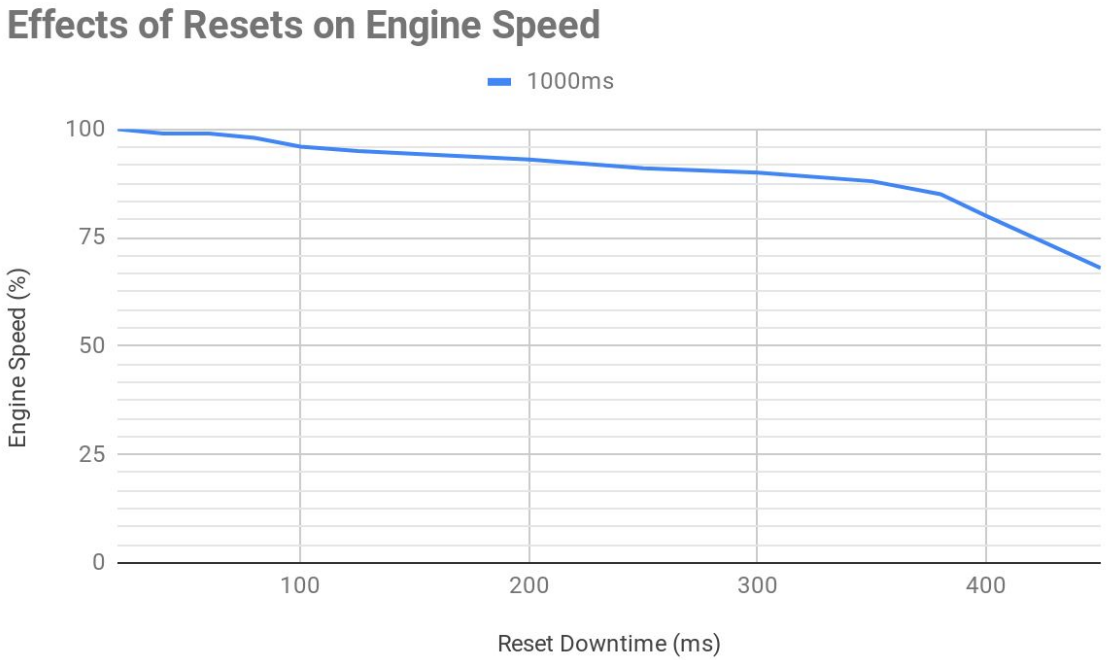


Program Control Flow Graph

[1] Davi, Lucas, et al. "Isomeron: Code Randomization Resilient to (Just-In-Time) Return-Oriented Programming." *NDSS*. 2015.

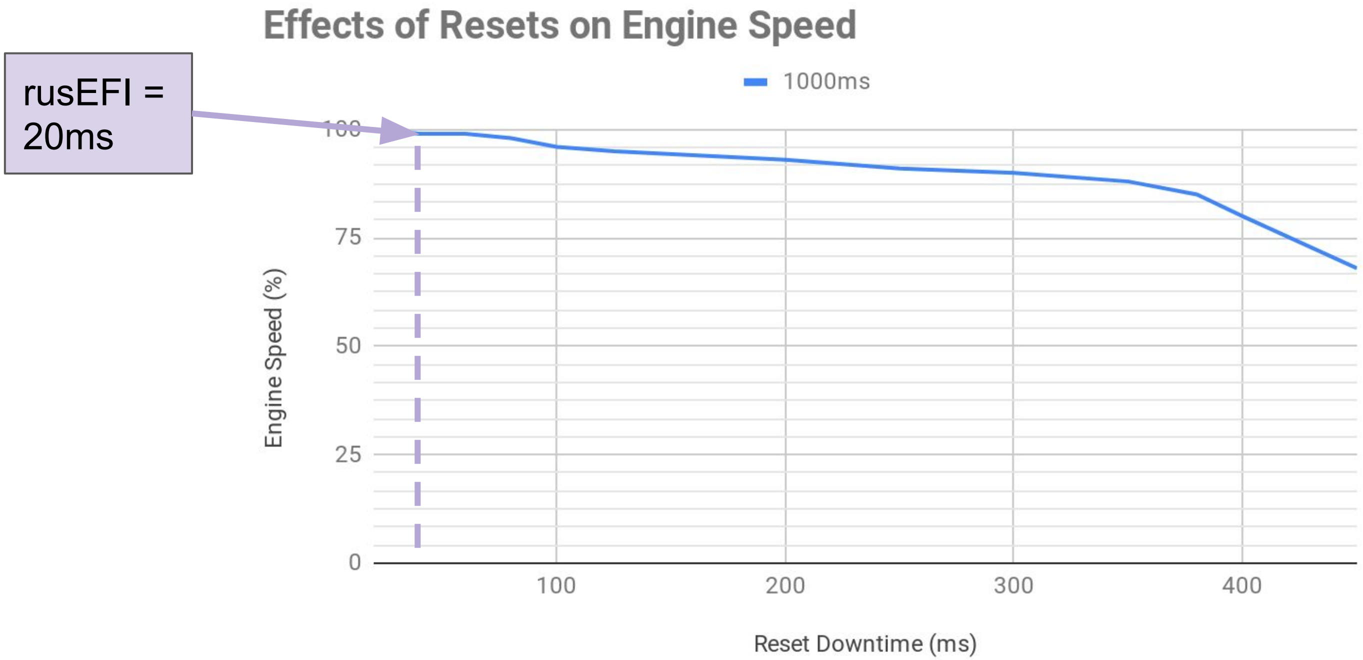
Case Study - **ECU**

YOLO Performance

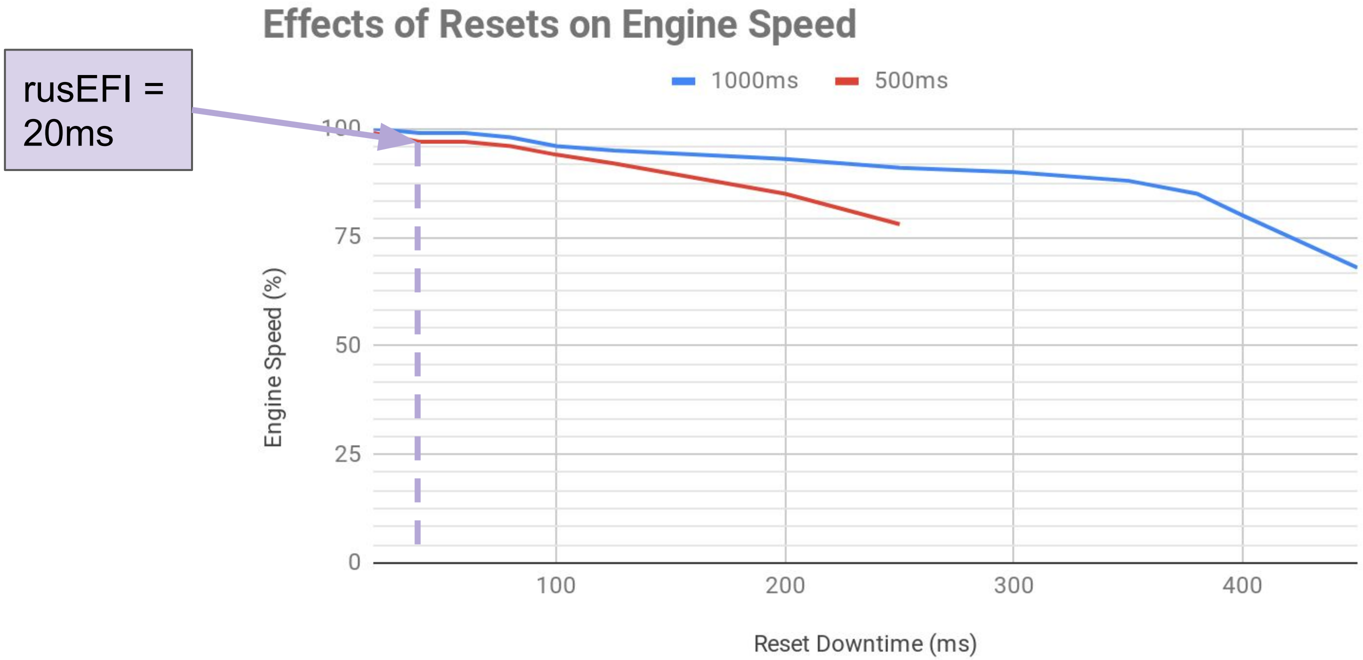


Case Study - **ECU**

YOLO Performance



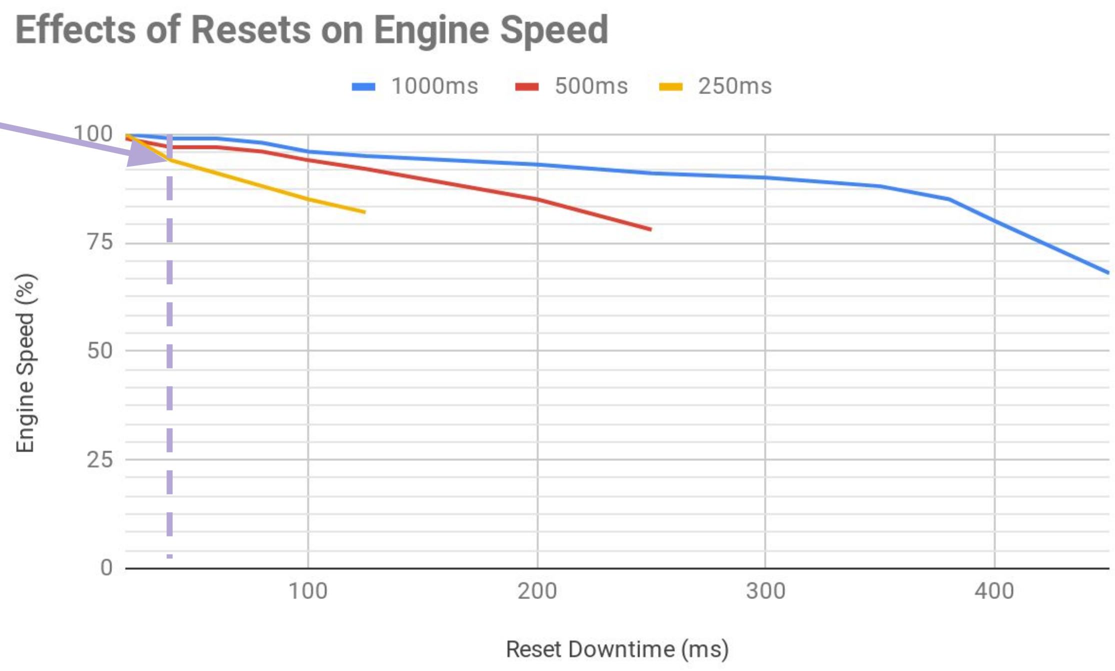
Case Study - **ECU**
YOLO Performance



Case Study - **ECU**

YOLO Performance

rusEFI =
20ms



Case Study - **ECU**

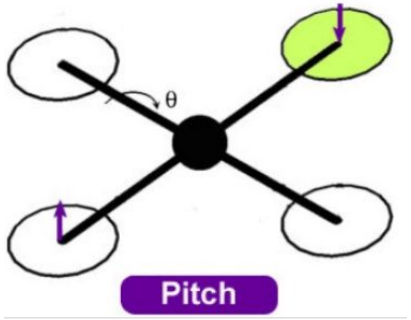
YOLO Performance

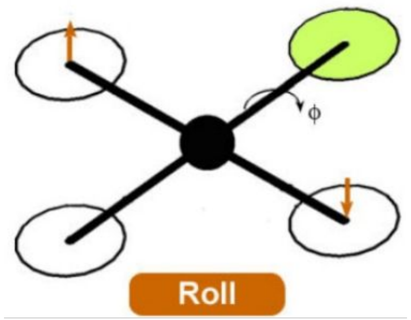
rusEFI = 20ms

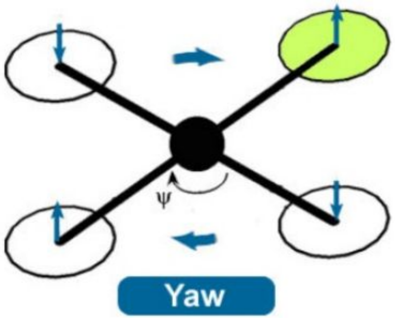
Case Study - **Flight Controller**

Case Study - **Flight Controller**

How it works







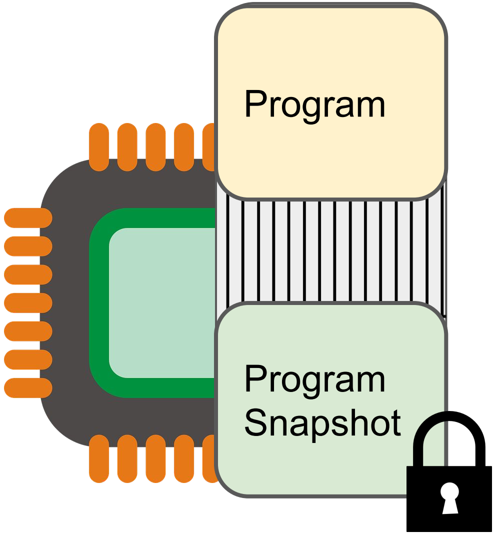
Case Study - **Flight Controller**

- PX4: Open Source FC
  - C/C++
- DJI F450 Flamewheel
- Cortex M4 @168 MHz
  - 192 KB SRAM
  - 1 MB Flash

Case Study - **Flight Controller**

Reset Strategy

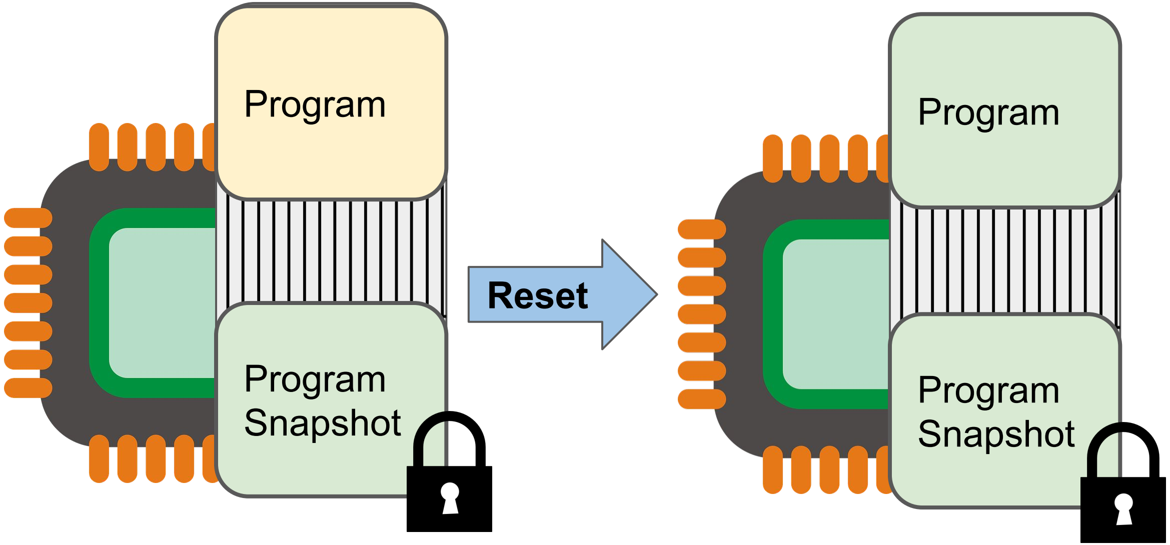
- Snapshot & Restore
    - Pre-initialized state for fast startup



Case Study - **Flight Controller**

Reset Strategy

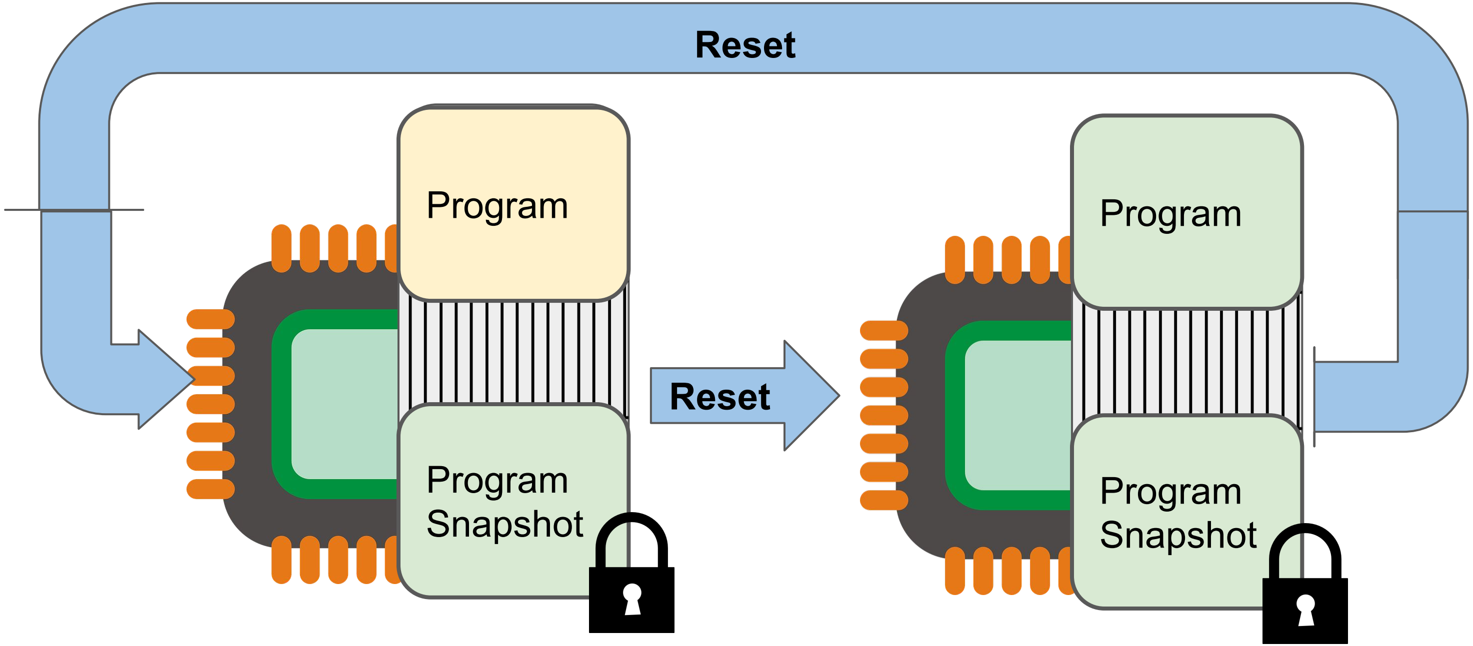
- Snapshot & Restore
    - Pre-initialized state for fast startup



Case Study - **Flight Controller**

Reset Strategy

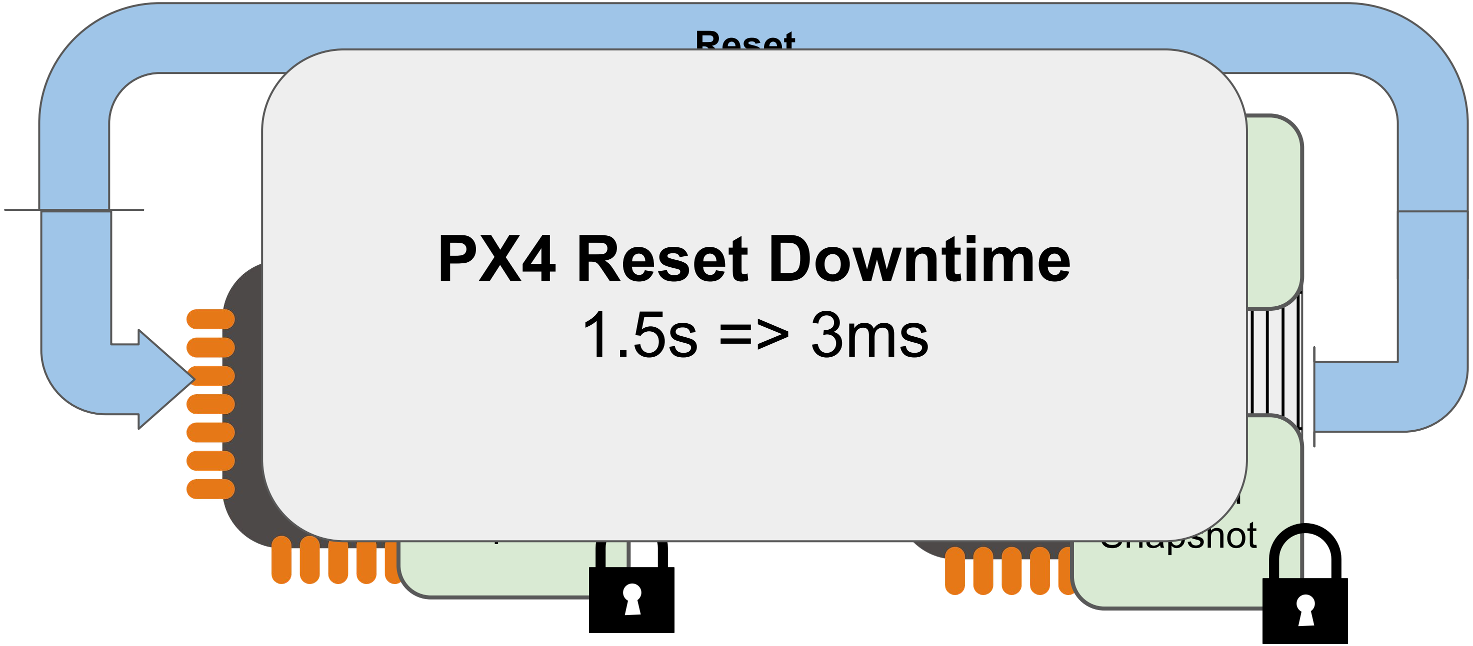
- Snapshot & Restore



Case Study - **Flight Controller**

Reset Strategy

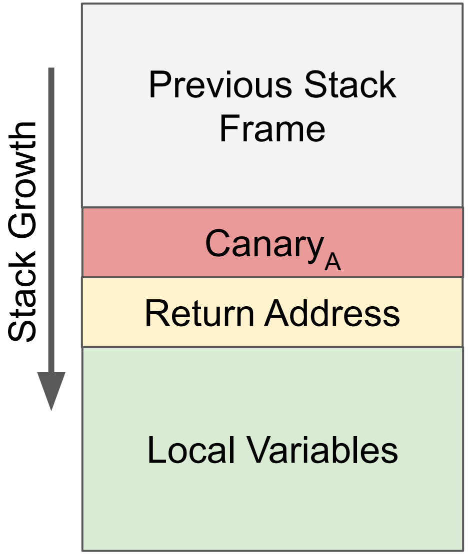
- Snapshot & Restore



Case Study - **Flight Controller**

Diversify Strategy

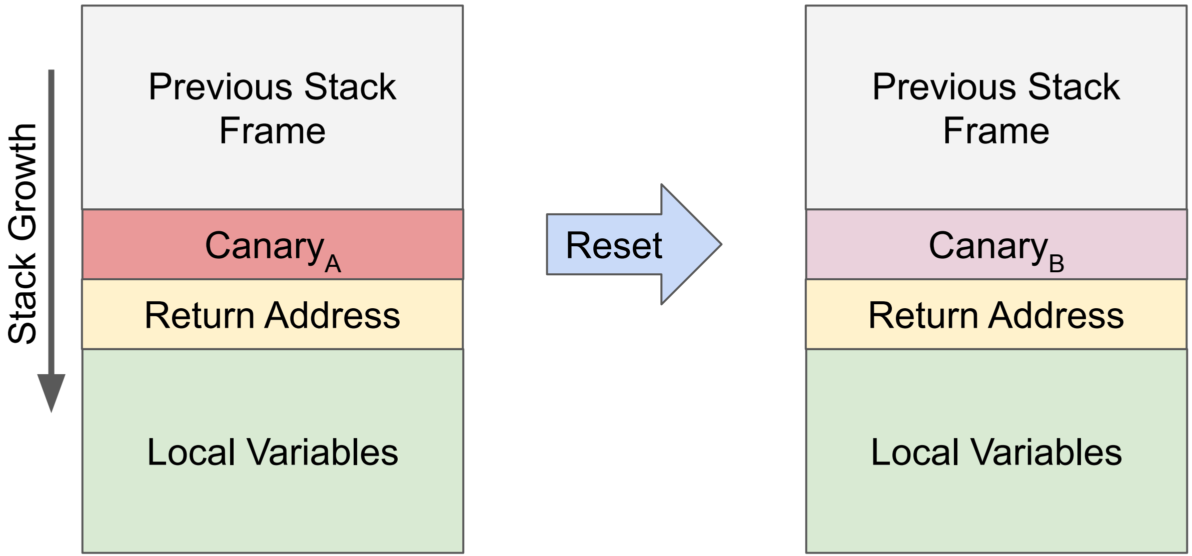
- Randomized Stack Canaries



Case Study - **Flight Controller**

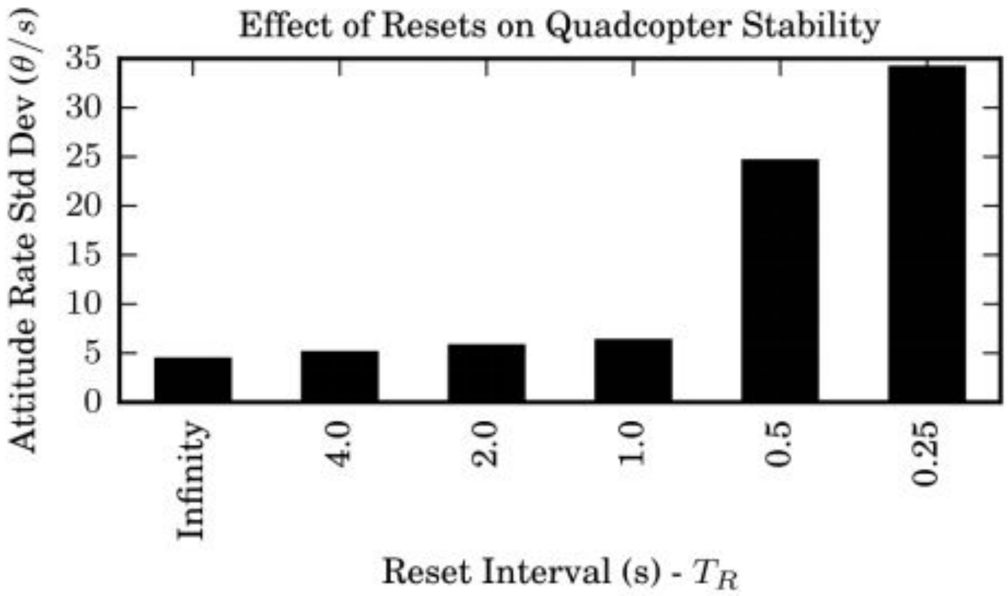
Diversify Strategy

- Randomized Stack Canaries

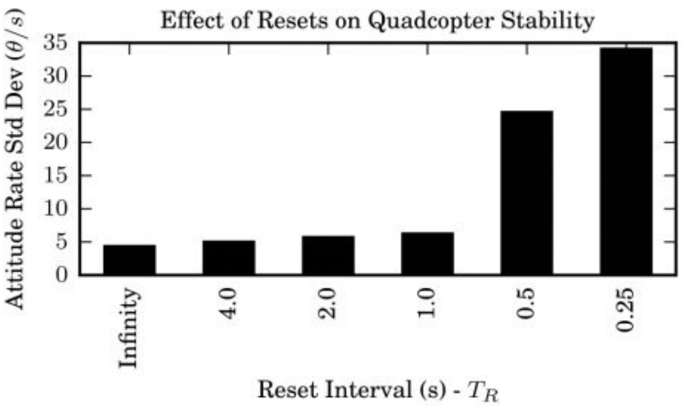


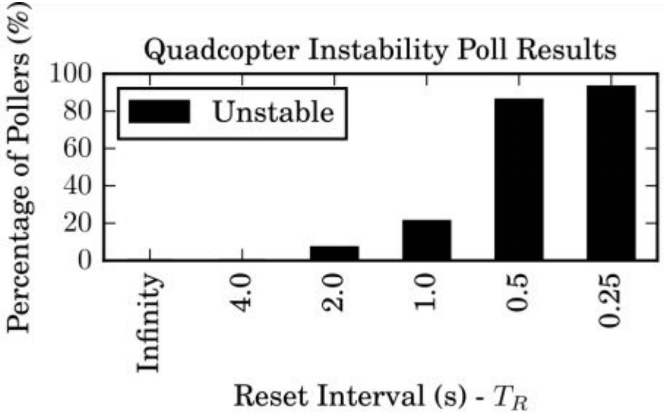
Case Study - **Flight Controller**

YOLO Performance



Case Study - **Flight Controller**
YOLO Performance





Lessons Learned
&
Open Questions

Lessons Learned

- Interdisciplinary research is challenging.
    - Catering to multiple audiences is a juggling act.
    - Find a collaborator across the disciplines you'll touch.
- Experimenting with physical systems takes time.
    - Lots of bureaucracy involved getting approval to do experiments.

Open Questions

- What is the community consensus on evaluating interdisciplinary work?
- What are appropriate venues for interdisciplinary work?

YOLO - Summary

- CPS properties can strengthen security.
- Eliminates malware from a system (RESET step).
- Increased work for an attacker (DIVERSIFY step).

Full Paper: **SPIE Defense & Commercial Sensing 2019** - (DOI: 10.1117/12.2518909)

Intentionally Left Blank

YOLO: Limitations & Mitigations

- Multiple Interacting Components
  - Timing and communications challenges may be mitigated by a microreboot like approach [2].
- Temporary loss of control
  - Replication & Interleaved resets can help alleviate this issue.
- Orthogonal Concerns
  - Spoofed inputs, algorithm stability, etc solutions can be layered with YOLO.

YOLO: Theoretical Analysis

- Controllable Canonical Form.

$$F(s) = \frac{b_0 S^n + b_1 S^{n-1} + \cdots + b_{n-1} S + b_n}{S^n + a_1 S^{n-1} + \cdots + a_{n-1} S + a_n}$$

$$A_i = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ -a_n & -a_{n-1} & -a_{n-2} & \cdots & -a_1 \end{bmatrix} \qquad B_i = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$$

$$C_i = \begin{bmatrix} (b_n - a_n b_0) & (b_{n-1} - a_{n-1} b_0) & \cdots & (b_1 - a_1 b_0) \end{bmatrix} \qquad D_i = b_0$$

YOLO: Theoretical Analysis

- Case Study

$$A_1 = \begin{bmatrix} 0 & 1 \\ \frac{-K_o K_i}{T_o + K_o K_d} & \frac{-(1+K_o K_p)}{T_o + K_o K_d} \end{bmatrix} \qquad B_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$C_1 = \begin{bmatrix} \frac{K_o K_i T_o}{(T_o + K_o K_d)^2} & \frac{K_o (K_p T_o - K_d)}{(T_o + K_o K_d)^2} \end{bmatrix} \qquad D_1 = \frac{K_o K_d}{T_o + K_o K_d}$$

$$A_2 = \begin{bmatrix} \frac{-1}{T_o} & 0 \\ 0 & 0 \end{bmatrix} \qquad B_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad C_2 = \begin{bmatrix} \frac{K_o}{T_o} & 0 \end{bmatrix} \qquad D_2 = 0$$