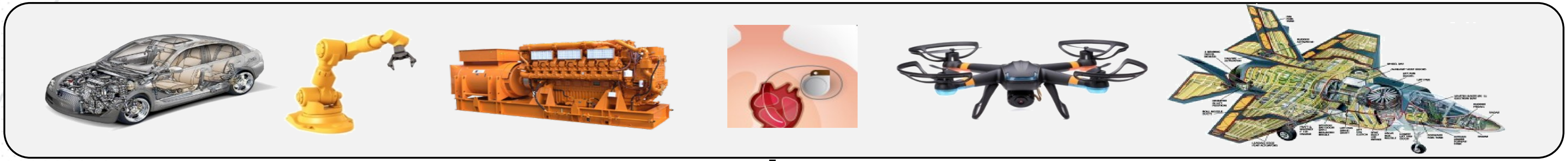
Security, Resilience and Artificial Intelligence in Cyber Physical Systems

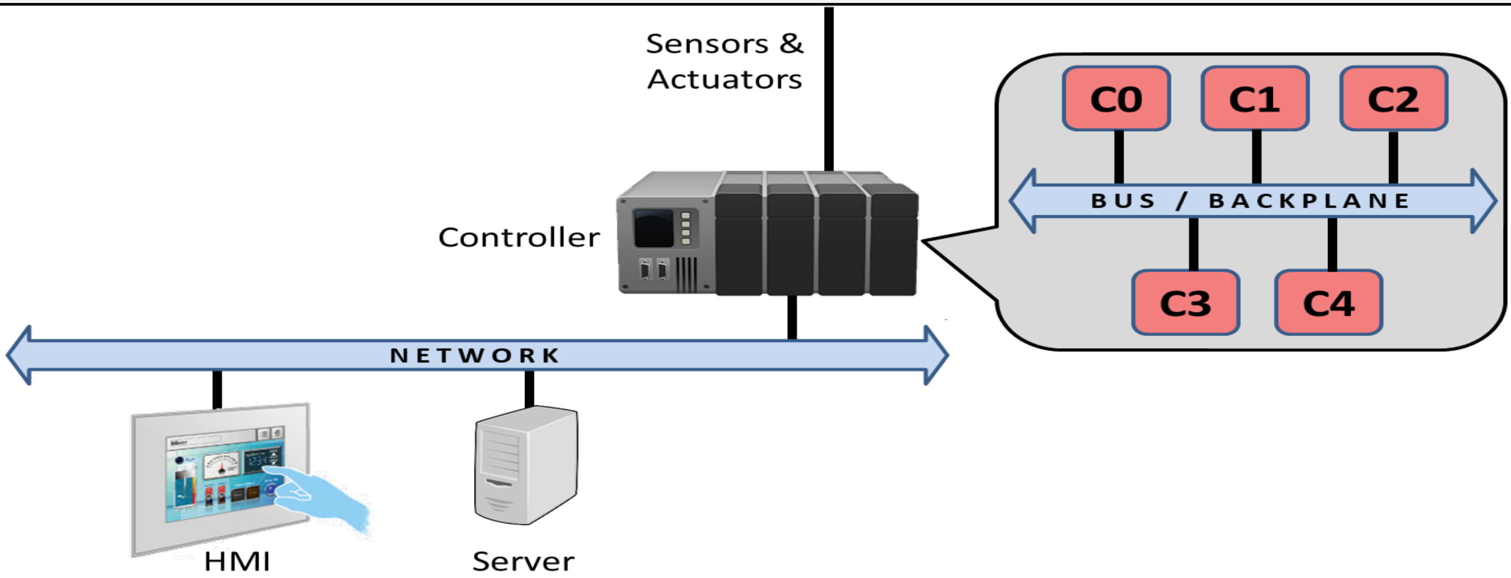
J. Sukarno Mertoguno

karno@gatech.edu

Georgia Tech College of Computing
School of Cybersecurity and Privacy

Cyber Physical Systems

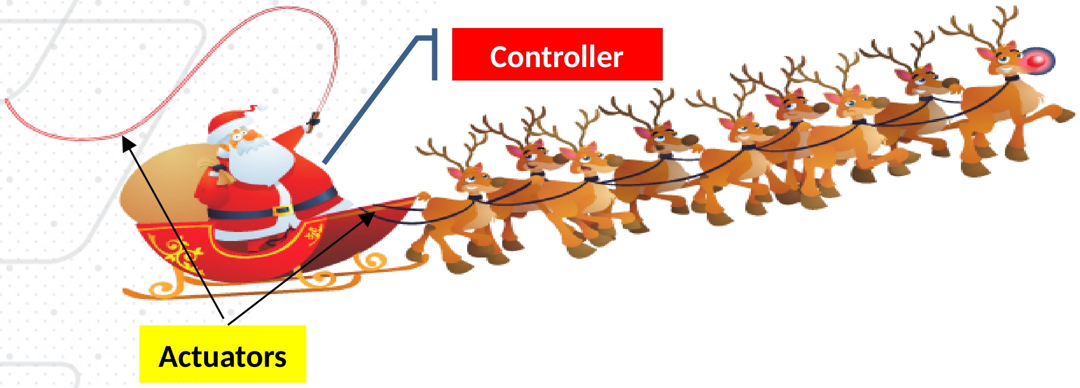




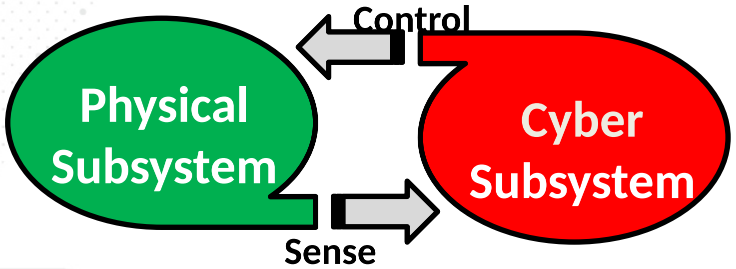
The class of computing systems that interact (i.e., observe & control) with physical processes

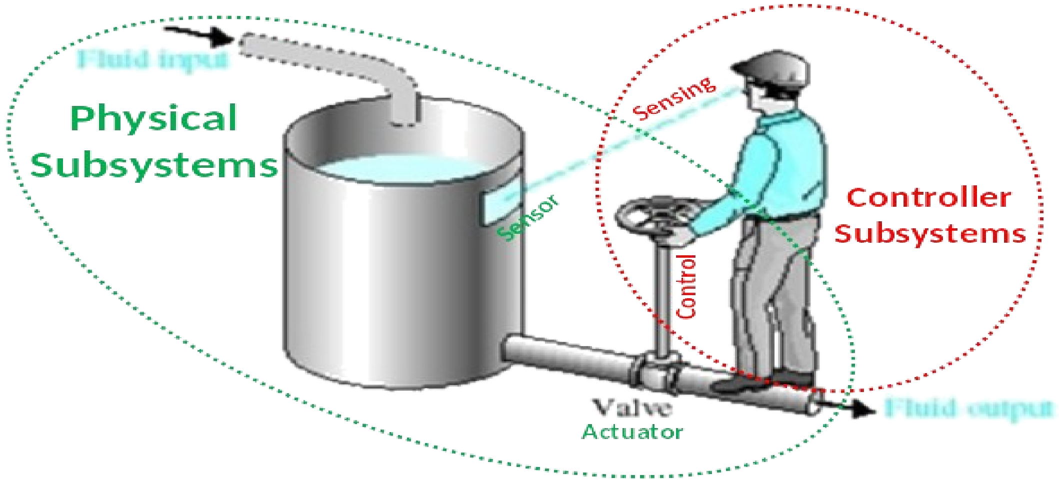
Georgia Tech College of Computing
School of Cybersecurity and Privacy

CPS Subsystems



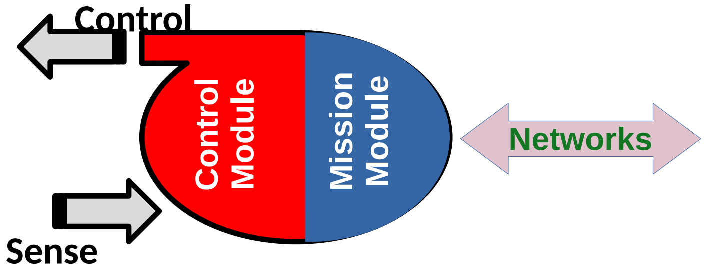
Cyber Physical System





A manual control system for regulating the level of fluid in a tank by adjusting the output valve. The operator views the level of fluid through a port in the side of the tank.

Cyber Subsystem



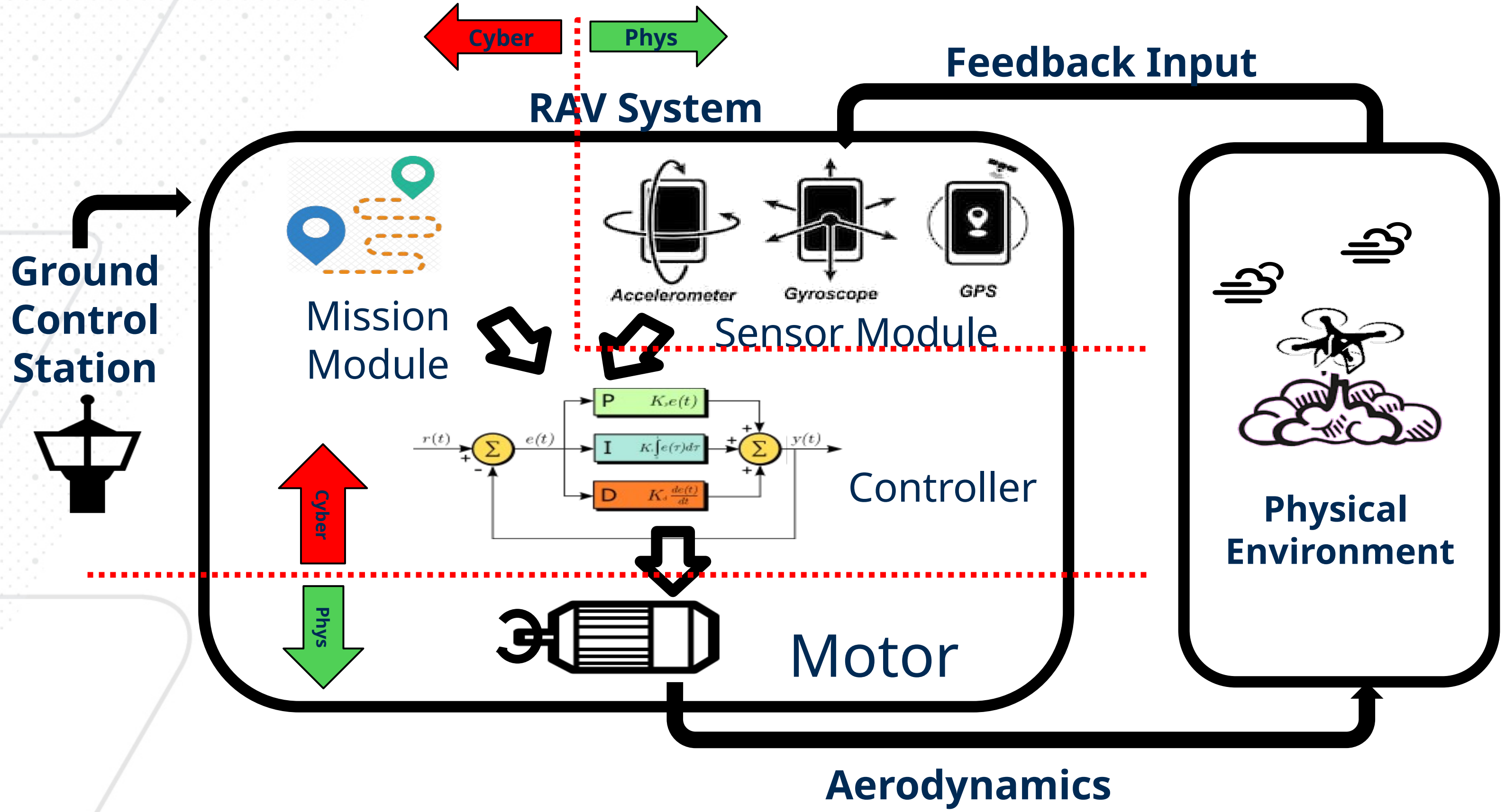
CPS: 2 loosely coupled subsystems

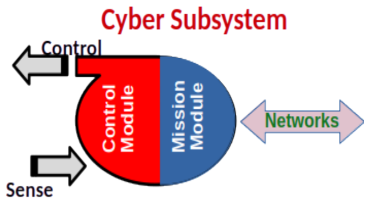
– **Physical** Subsystems, governed by physics

– Controller (**Cyber**) Subsystems, periodically sense/monitor & control Physical Subsystems

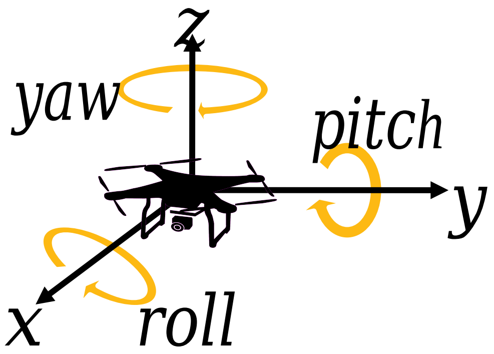
Goal: to have the **physical systems** behave properly and as expected, regardless of fault or disruption (cyber or otherwise).

An Illustration: Robotic Aerial Vehicle



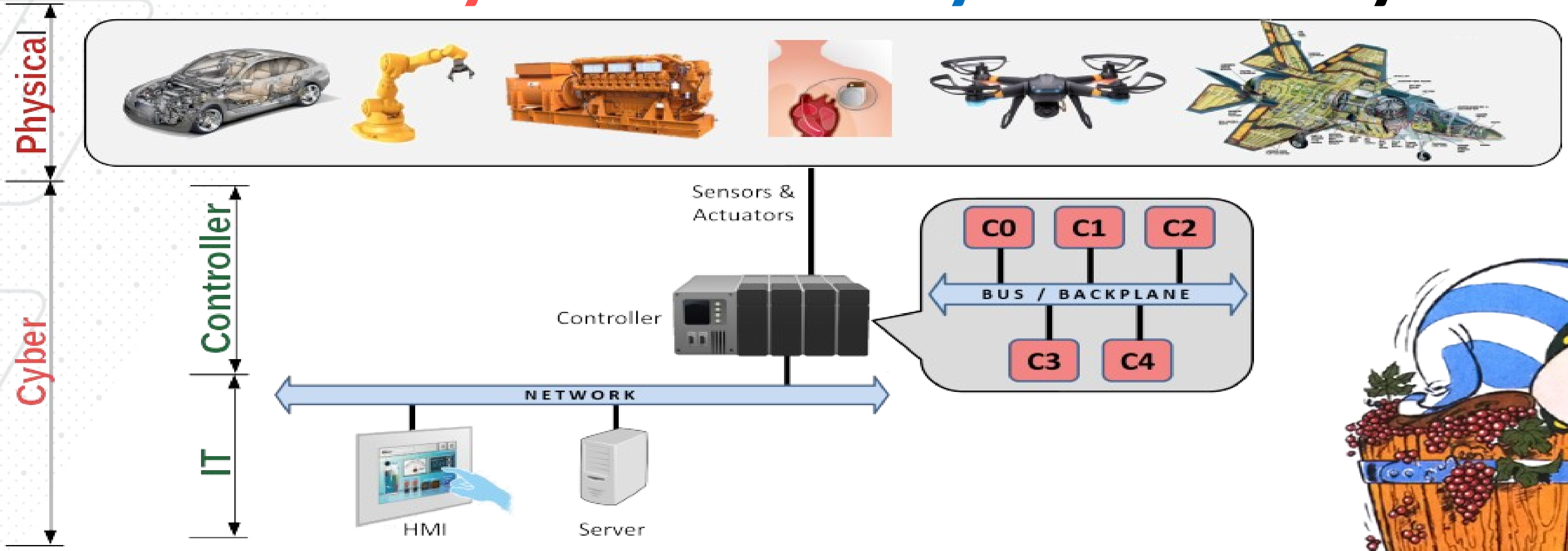


- Controller
  - Control vehicle movements & operations
  - Follow user commands
  -



Adapted from: Dongyan Xu, ACM AsiaCCS '19 Keynote:
"From Control Model to Control Program: A Cross-Layer Approach toRobotic Vehicle Security",

Cyber Physical Systems



Subsystems:

- Physical Subsystems
- Cyber Subsystems
  - IT Space
  - Controller Space (our focus)

CPS Security Space

IT (or some called it OT) Space :

- Monitoring & intrusion detection is relatively easier due to predictability of CPS operation
- Encryption & authentication

Controller Space :

- Knowledge/Model dependent (e.g. digital twin, intrusion detection at controller bus level, etc)
- Encryption & authentication
  - limited computing capacity,
  - integrity (I – assuring data correctness) is extremely important,
  - authentication (A – assuring sender identity) is very relevant,
  - but encryption (C – assuring no-information-leak) is less so in majority of applications (data is low-level, state dependent & temporal/short-lifetime)
- **Mechanism** (knowledge independent)

Impact on Solution Space

Cyber **P**hysical **S**ystems

Supporting
Is a "means to an end"

Primary Concern
"Stability & Integrity"

- Cyber-attack resilient solutions should be primarily defined and motivated by **physical requirements**

- Things we need to protect are not exactly the same as with protecting IT systems
  - Availability & Integrity are the utmost important
  - Confidentiality is less of a concern, no need for full heavyweight encryption, etc.

Different solution space to explore

Focus on Cyber v.s. Physical

a)

		Physical	
		Stable	Unstable
Cyber	Stable	P(S), C(S)	P(U), C(S)
	Un-stable	P(S), C(U)	P(U), C(U)

b)

		Physical	
		Stable	Unstable
Cyber	Stable	P(S), C(S)	P(U), C(S)
	Un-stable	P(S), C(U)	P(U), C(U)

Cyber centric

- Focusing on cyber stability/security
- However physical also need to be stable

Physical centric

- Focusing on Physical stability
- Due to time scale different, limited cyber in-stability can be tolerated

Physical-centric provides additional design space to explore

Properties of CPS

— All resulting from physical requirements —

Real-time	Safety-critical	Resource constrained	Inertia

- Execution must meet hard real-time deadlines
- Sensitive to **latency** variations (need predictability)
  - For example: we are guaranteed an output every 10ms…

Tolerant to INPUT disruption

- Periodic, often uses predefined task scheduling slots (time-wheel)
- Security solutions cannot disrupt real-time properties or it will severely impact reliability

Periodicity tolerates occasional disruption

Properties of CPS

— All resulting from physical requirements —

Real-time	**Safety-critical**	Resource constrained	Inertia

- Systems are often expensive and are designed for longevity
  - Especially true for safety-critical systems

- Often require extensive physical certifications: shock, vibe, interference, radiation hardening, etc.,

- Often too expensive or impractical to replace

  ➡ must focus on the **legacy** equipment and how to **retrofit**

Properties of CPS

— All resulting from physical requirements —

Real-time	Safety-critical	**Resource constrained**	Inertia

- CPS are not meant to be general purpose computers
  - Designed with just enough resources to get the job done
- Systems are often **resource constrained**:
  - Memory
  - Storage
  - CPU
- May lack many IT-style defenses (data execution prevention, ASLR, etc.)
  - Some embedded processors do not have MMU
  - There may not even be an OS!

Properties of CPS

— All resulting from physical requirements —

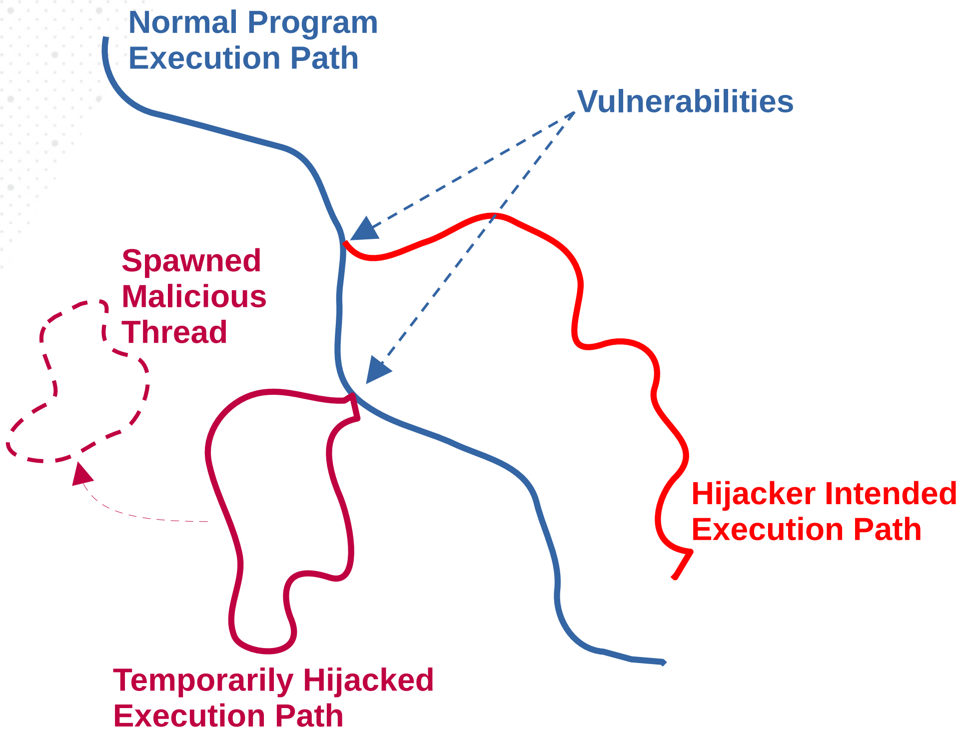
Real-time	Safety-critical	Resource constrained	Inertia

Tolerant to OUTPUT corruption

- Physical systems follow laws of physics and have ***inertia***
- Effect: physical systems can tolerate some small loss of signal and still maintain stability
- Order of magnitude difference between physical & cyber speeds
- While first 3 properties present constraints, the physical inertia property gives us leeway:
  - Can lose some state and still keep going

Inertia provides natural tolerance

Fundamental of Cyber Exploit

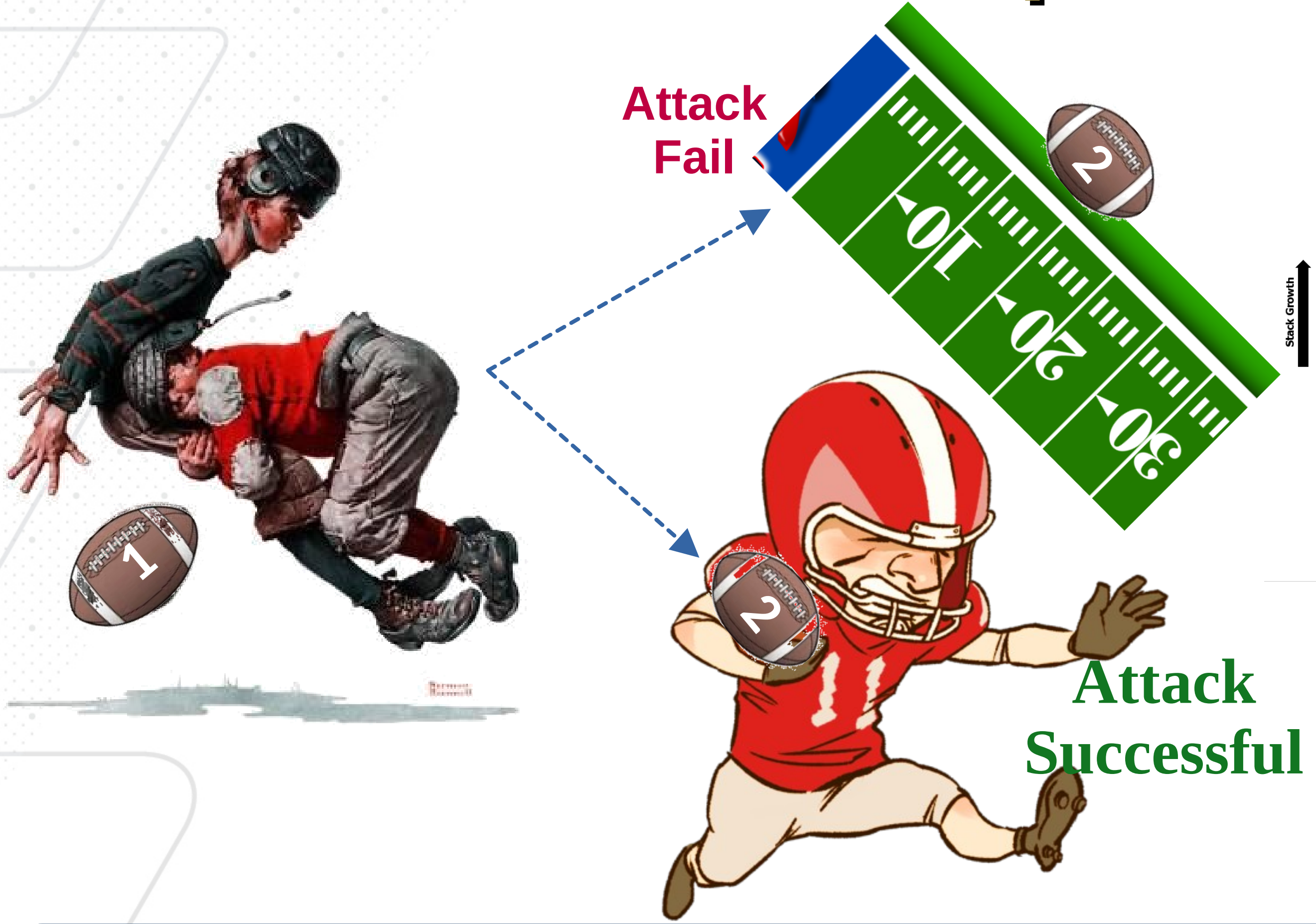


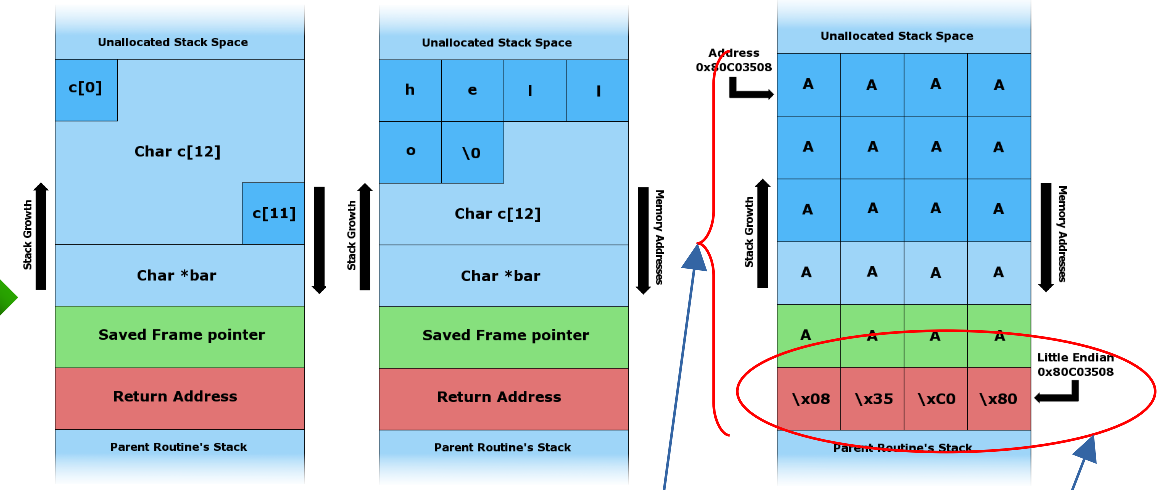
Malicious Intent

- Needs to execute its bad stuffs
- Hence, needs to hijack original or target program/execution
- Exploits **vulnerability** in target program to get an **opportunity** to run its bad stuff
- Needs to **own the process**, at least temporarily (short duration)

Vulnerabilities are essential for Bad things to do Bad stuffs

2 Fundamental Requirement of Cyber Exploit





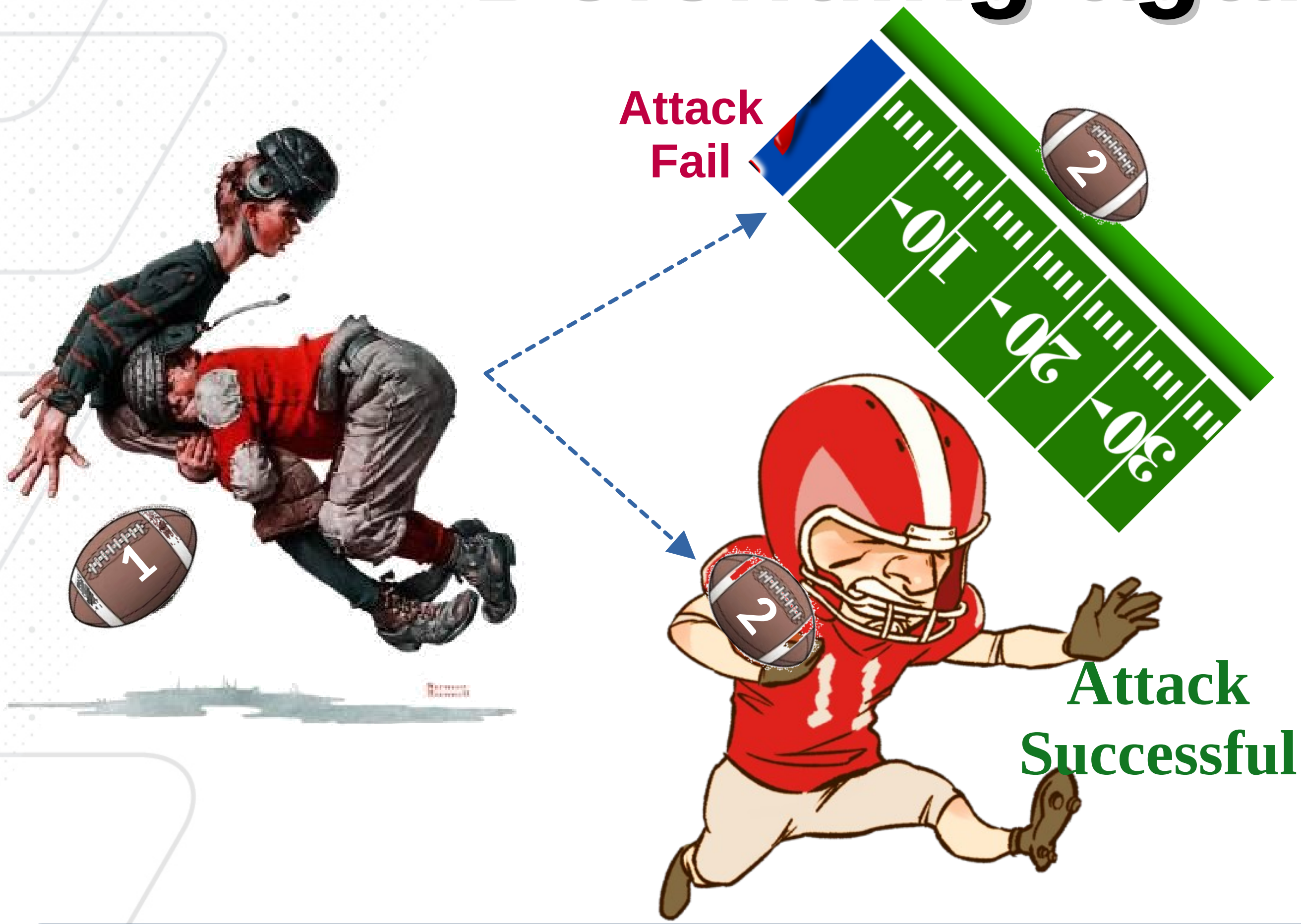
1. Tackle
Exploit Vulnerability

2. Recover
This Address must point to Malware intended code

Successful attack requires:

1. Success on derailing targeted program --> targeted program loses control
2. Success on capturing control --> attacker controls program execution

Defending against Stage 1



- Prevention requires:
  - No Cyber (software) Vulnerability,

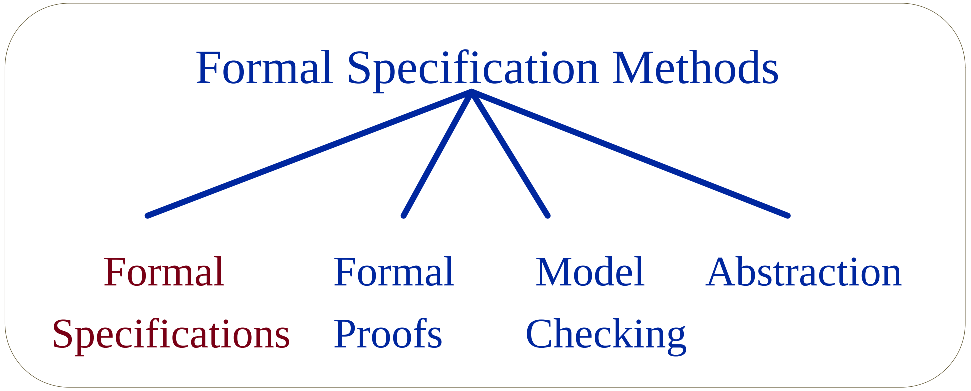
    or
  - Complete (adequate) Cyber Defense
- Hard to achieve and guarantee

Successful attack requires:

1. Success on derailing targeted program --> targeted program loses control
2. Success on capturing control --> attacker controls program execution

No Vulnerability

- Formal methods often used to provides guarantee for No-Vulnerability
- Formal Methods:
  - Rigorous Mathematics & Formal Logics based methods for modeling and analyzing (computer-based) systems
  - **Formal specification**
    - Build a mathematical model of the system → Spec of Sw, Spec of Exe Env
    - Express properties (requirements) → Spec of Assertion
  - **Formal verification**
    - Check that the model satisfies its requirements
  - For: Hardware, **Software**, Distributed Systems, etc.

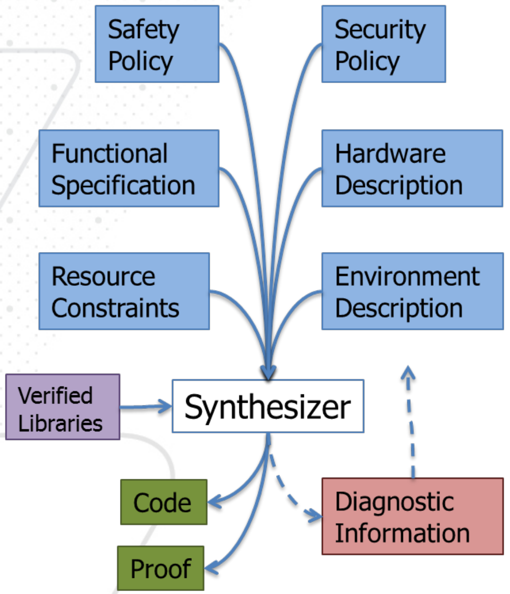


- Can provides coverage guarantee where testing cannot
- Generally an expensive proposition

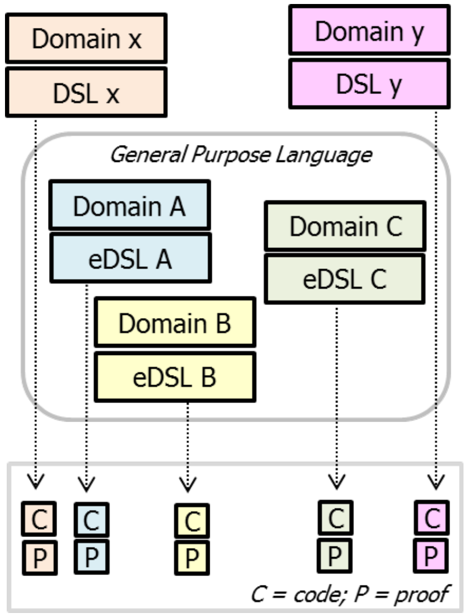
CPS is generally small enough for Formal Methods to be Viable

DARPA HACMS:
Clean-Slate Methods for High-Assurance Software

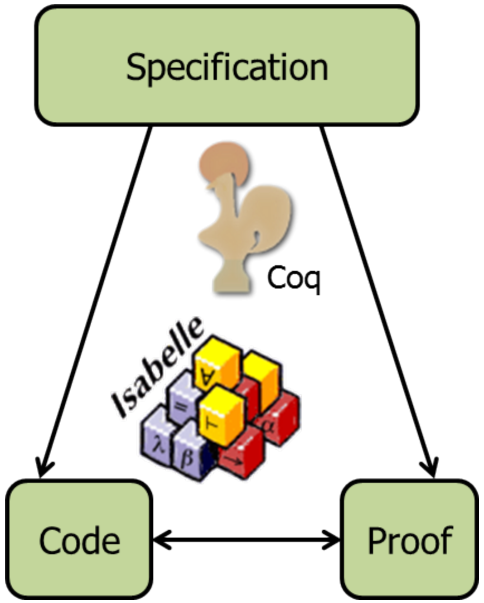
Code Synthesis



Domain Specific Languages (DSLs)



Interactive Theorem Prover as PL



High Assurance: Ensuring Correctness, Safety, Security

Formal Methods (Top Down)

Preservation of properties

Model ⟷ **Executable**

Formally Verified

Libraries Runtime

1) Formal Specification → develop src code,

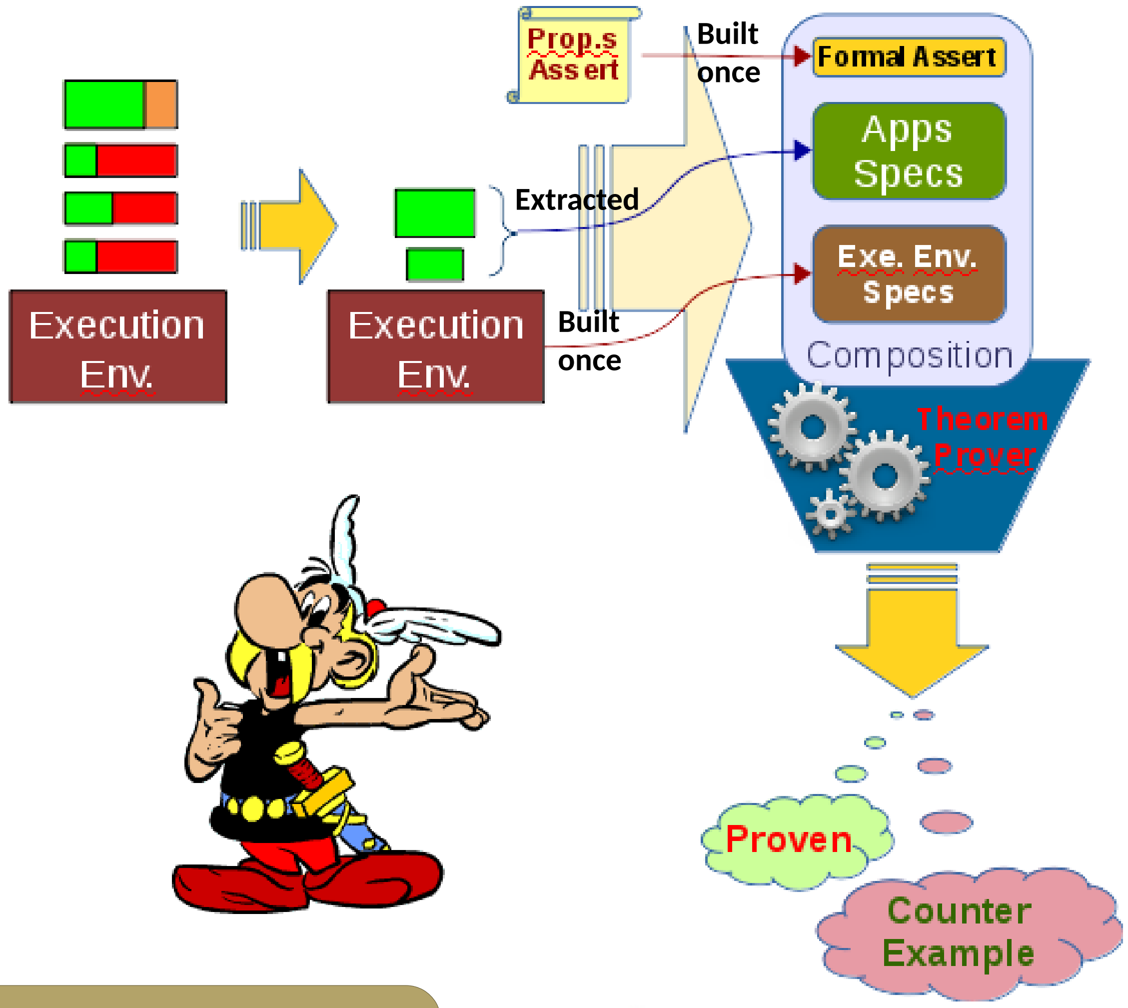
   or

   Src Code → develop formal model/spec
2) Includes assertions in the code or Assertion Spec
3) Formal **Model evaluated** with help of Proof Assistant (eg. COQ, Isabelle, EZ-Crypt (for crypto))
4) Compile w/ property preserving compiler → Binary

What do you means that you formally verify your code but not your libraries ???

Correctness and Fulfilling Requirements

- Functionality-preserving with respect to either full or reduced set of features
- Validation of functionality
- Verification of desired properties
- Formal assertions of (security) properties
  - Formal model of execution environment
  - Extracted formal model of the program/application
  - Formal specification of properties to assure



ONR's Bottom Up Formal Methods

Cyber Defenses

Existing security mechanisms: W⊕R, ASLR, CFI
→ Not hard to by pass

Protect all dangerous operation using **sanity checks**:
→ Auto-applied at compile time

```
void foo(T *a) {
    *a = 0x1234;
}
```

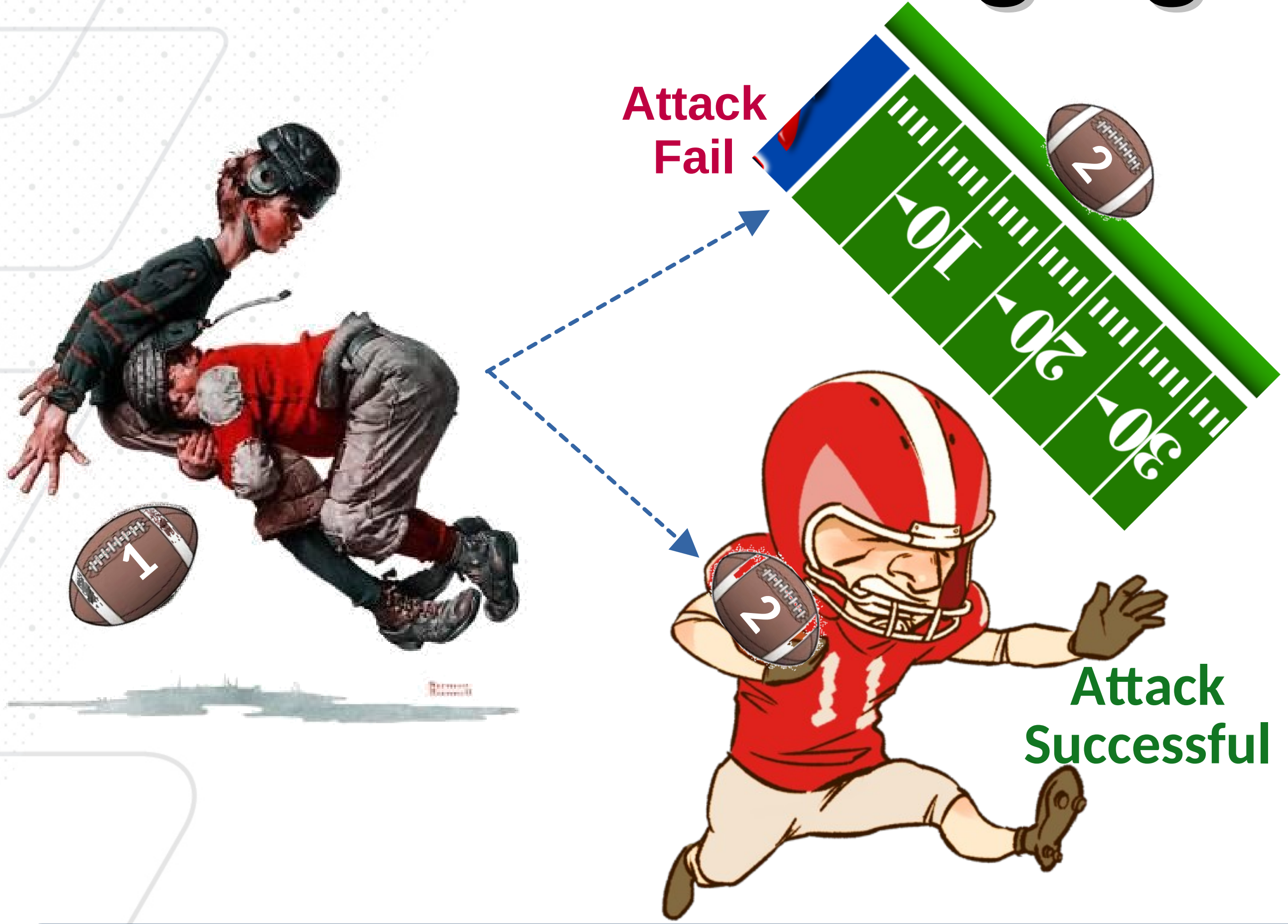
Sanitize →

```
void foo(T *a) {
    if(!is_valid_address(a) {
        report_and_abort();
    }
    *a = 0x1234;
}
```

Cyber Defenses

Memory Error	Main Causes	Defenses
Out-of-bound read/write	Lack of length check	Softbound AddressSanitizer
	Integer overflow	
	Format string bug	
	Bad type casting	
Use-after-free	Dangling pointer	CETS (Compiler-Enforced Temporal Safety) AddressSanitizer
	Double free	
Uninitialized read	Lack of initialization	MemorySanitizer
	Data structure alignment	
	Subword copying	
Undefined behaviors	Divide-by-zero	UndefinedBehaviorSanitizer
	Pointer misalignment	
	Null-pointer dereference	

Defending against Stage 2



- Prevention includes:
  - Randomization
  - (Artificial) Diversity
- Easier, but
- Stage 1 have already occurred

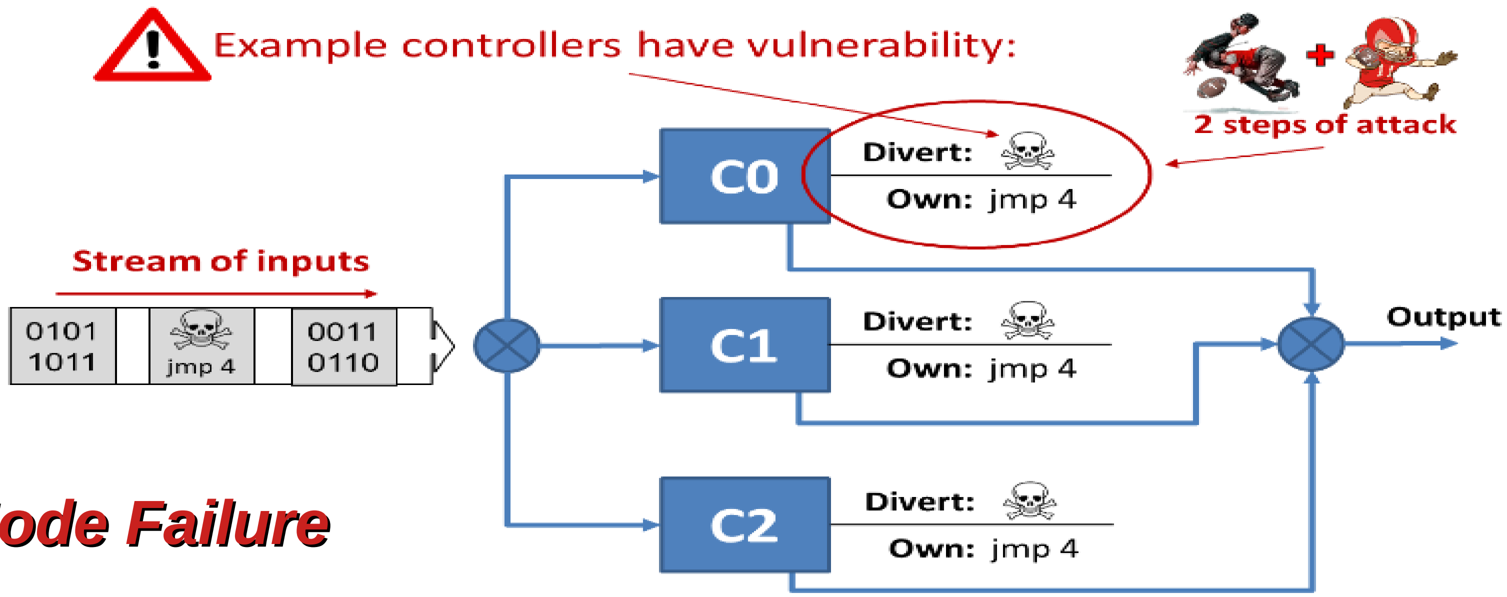
Successful attack requires:

1. Success on derailing targeted program --> targeted program loses control
2. Success on capturing control --> attacker controls program execution

Traditional Fault Tolerance

Many systems already employ some type of **fault tolerance** for **physical** and **random** failures:

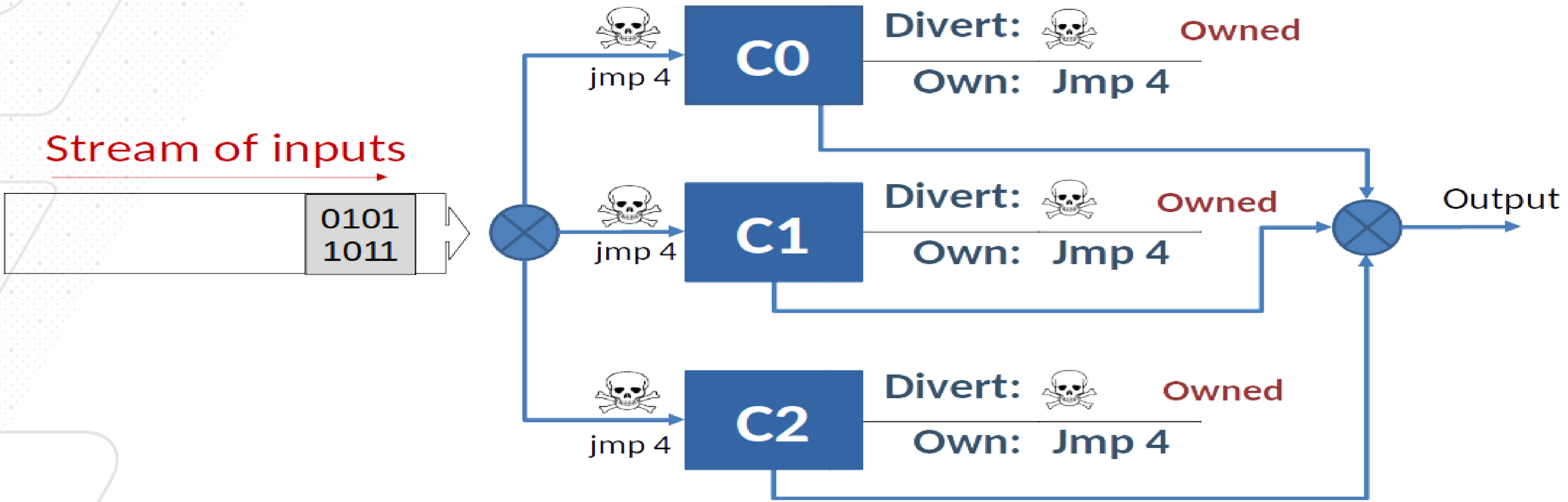
- Redundancy with voting/consensus
- Quad Redundant Control (QRC)
- Byzantine Fault Tolerance (BFT)



Cyber Attack → Common Mode Failure

How to transform Fault Tolerance into Cyber-Attack Tolerance ???

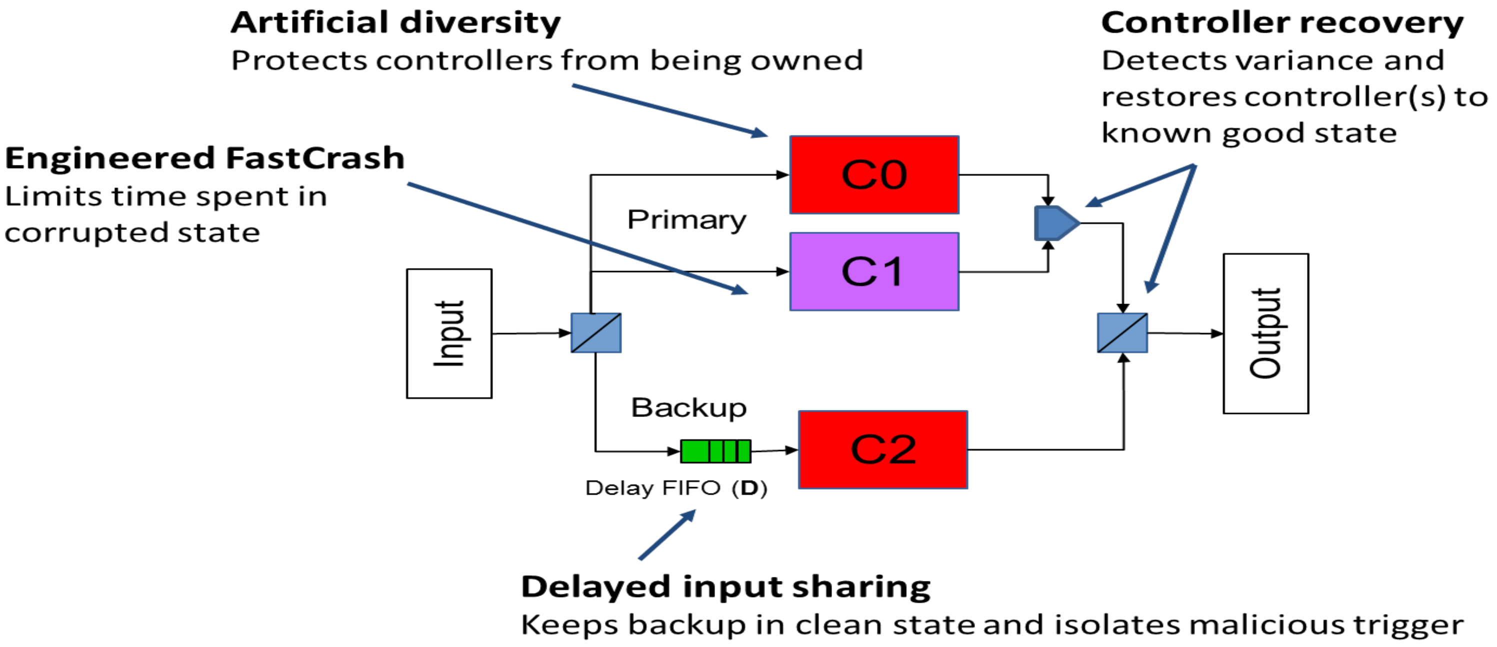
Common Mode Failure



Effect: All owned

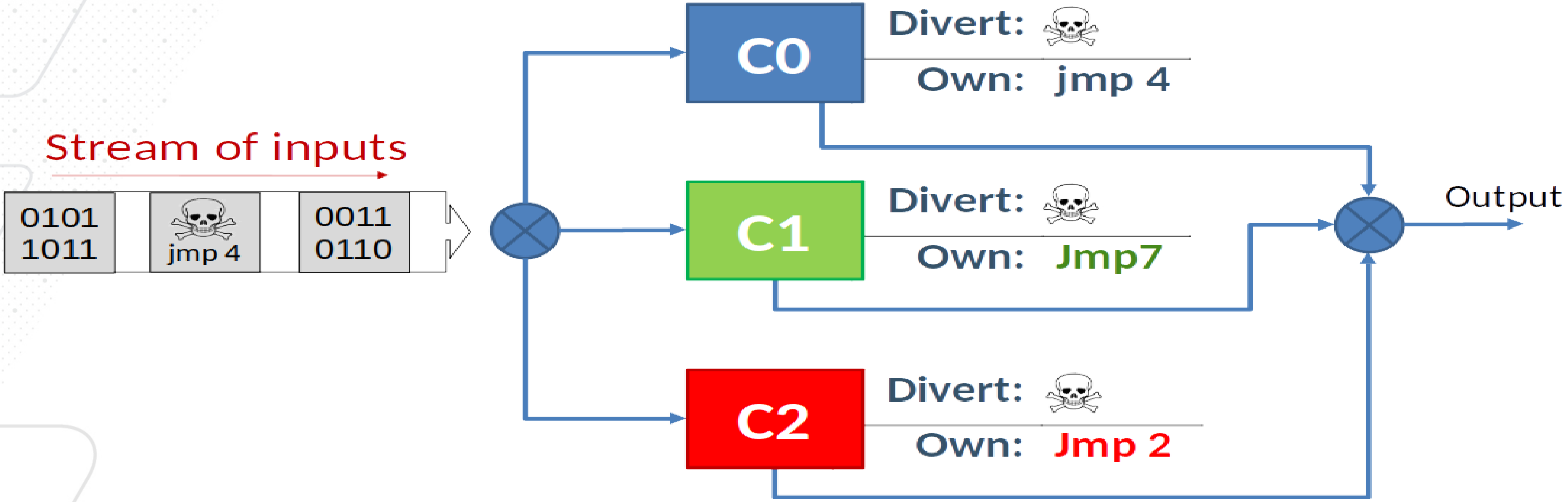
Key Elements of BFT++

- **Execution level diversity**
  - Same algorithm, same source code
  - Diversifying compiler (DARPA-CRASH)
  - Binary diversifying transformer (ONR, DARPA-CFAR)
- **Algorithmic diversity**
  - Different algorithm → different source code
  - Exp.: sort → quick sort, bubble sort, merge sort & all sort of sort stuffs.



BFT++ assumes Execution Level Diversity

with Diversification



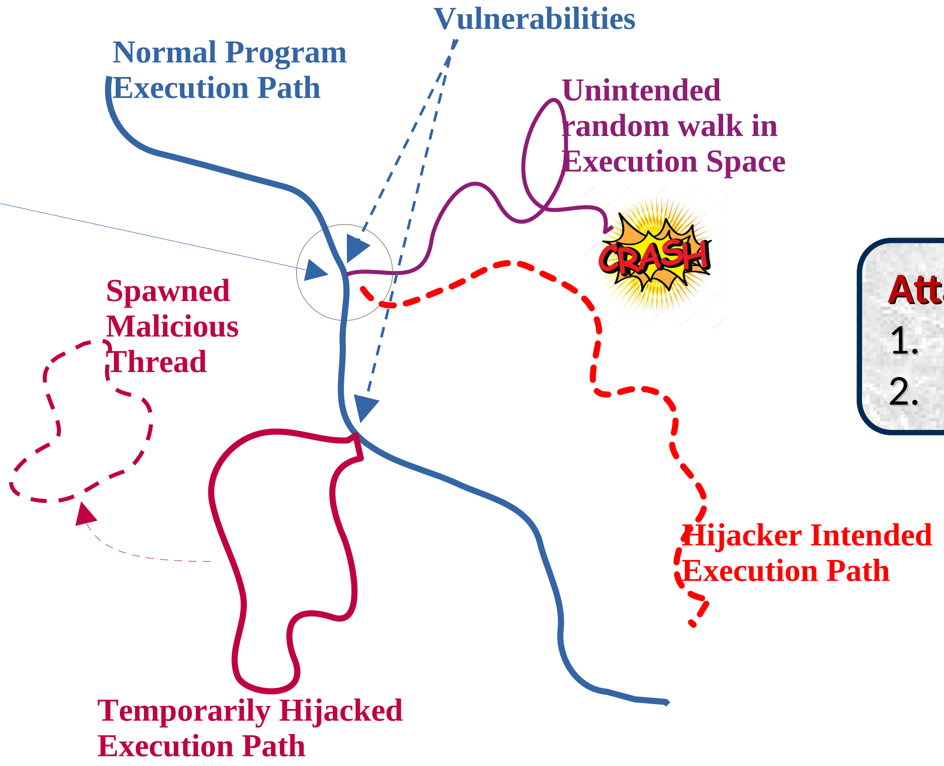
Successful attack requires:

1. Success on derailing targeted program --> targeted program loses control
2. Success on capturing control --> attacker controls program execution

Failure to Jump to Intended Instruction

Internal state of the processor & memory not compatible w/ the Unintended (garbage) Instruction

Not unlike
FUZZING



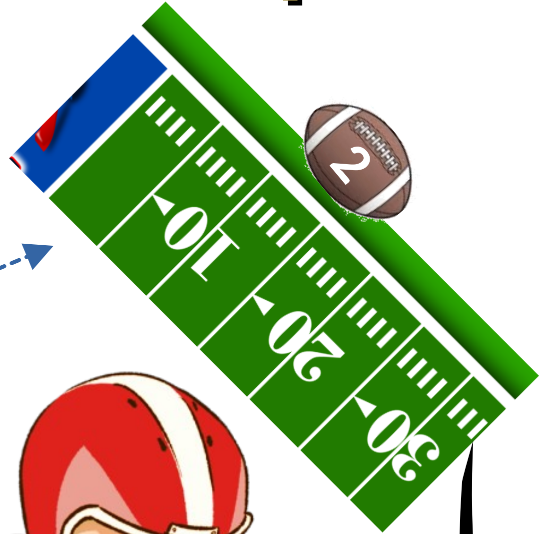
Attack Failure on 2nd phase:

1. targeted program loses control
2. attacker loses controls

CRASH is *practically* guaranteed

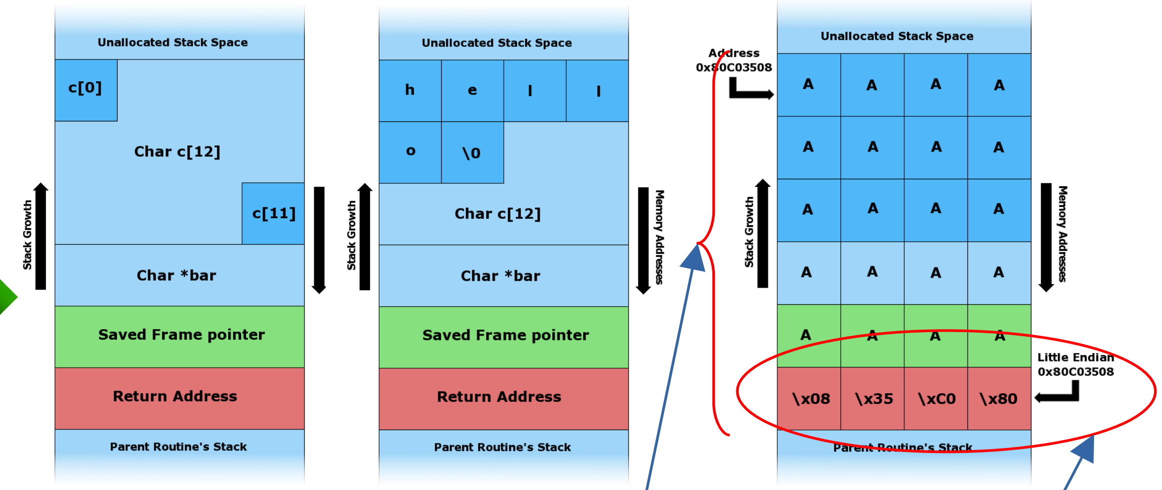
2 Fundamental Requirement of Cyber Exploit

Forcing → Attack Fail



Attack Successful

Preventing



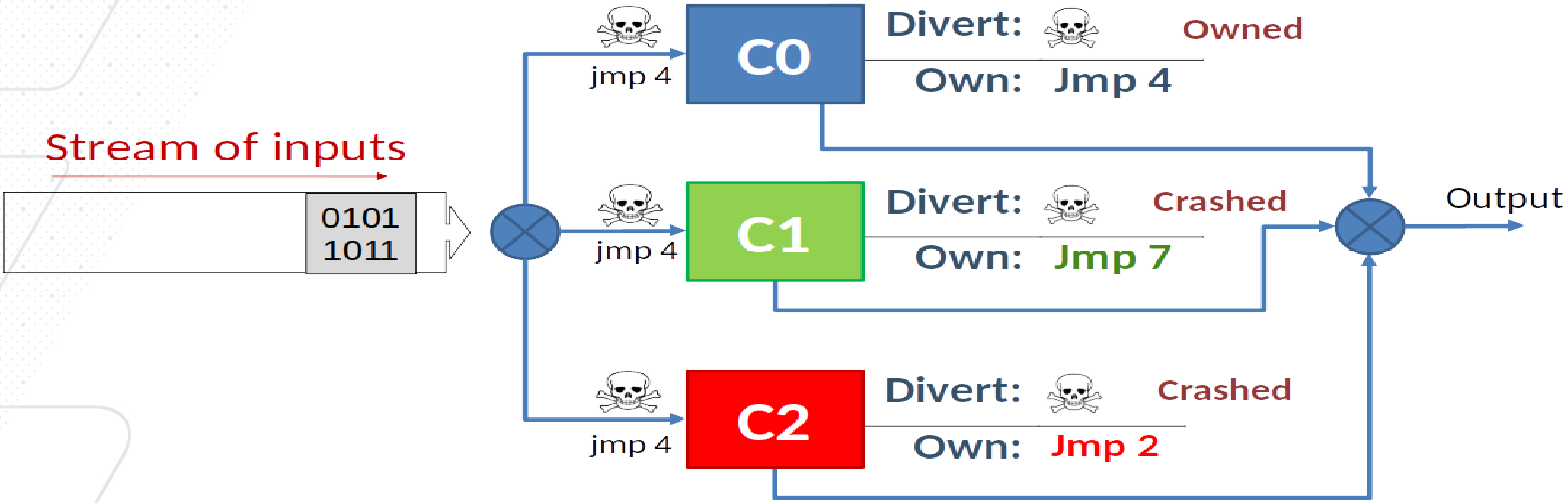
1. Tackle
Exploit Vulnerability

2. Recover
This Address must point to Malware intended code

Successful attack requires:

1. Success on derailing targeted program --> targeted program loses control
2. Success on capturing control --> attacker controls program execution

with Diversification



Effect: 1 owned, others crashed

Controller Recovery

- If we do not need to save controller state:

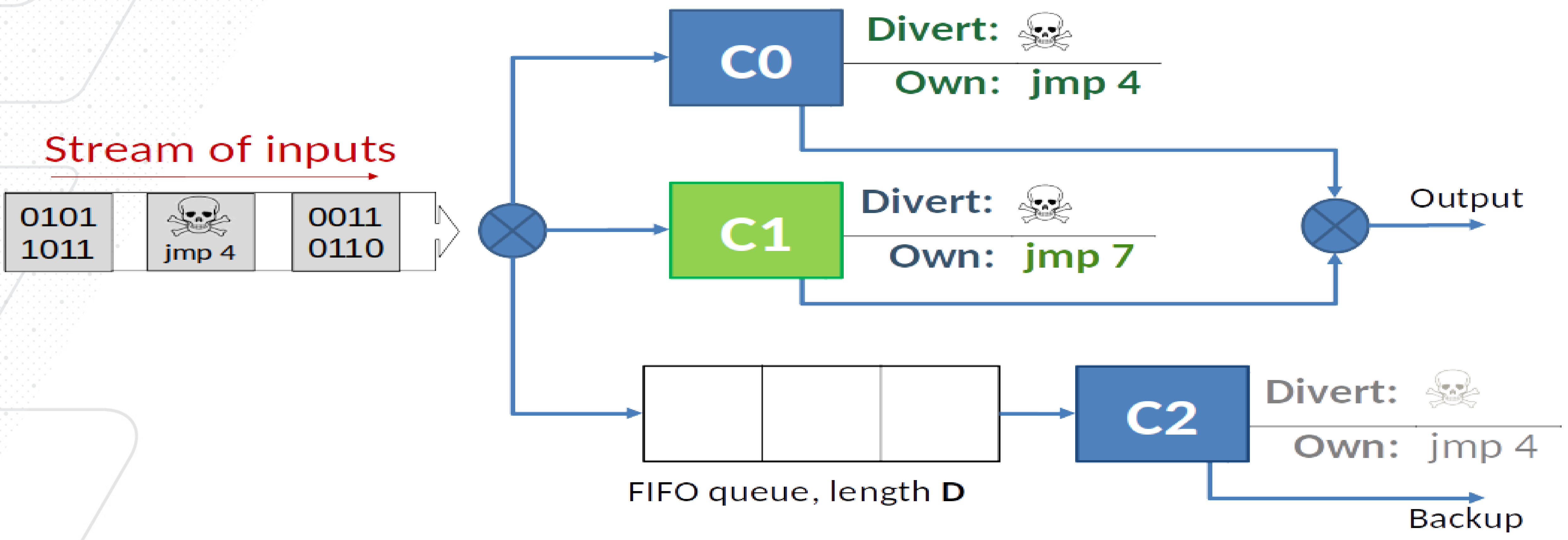
➡ Restore from a cold backup

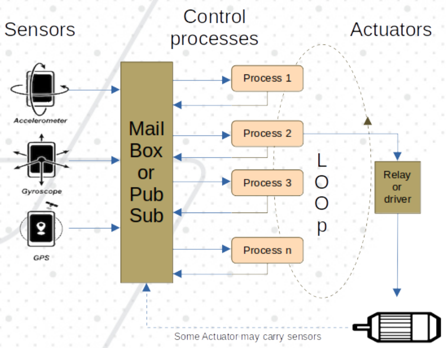
- If we need to restore with state, need a hot/warm backup

➡ But how can we keep a hot backup that does not crash or get owned?

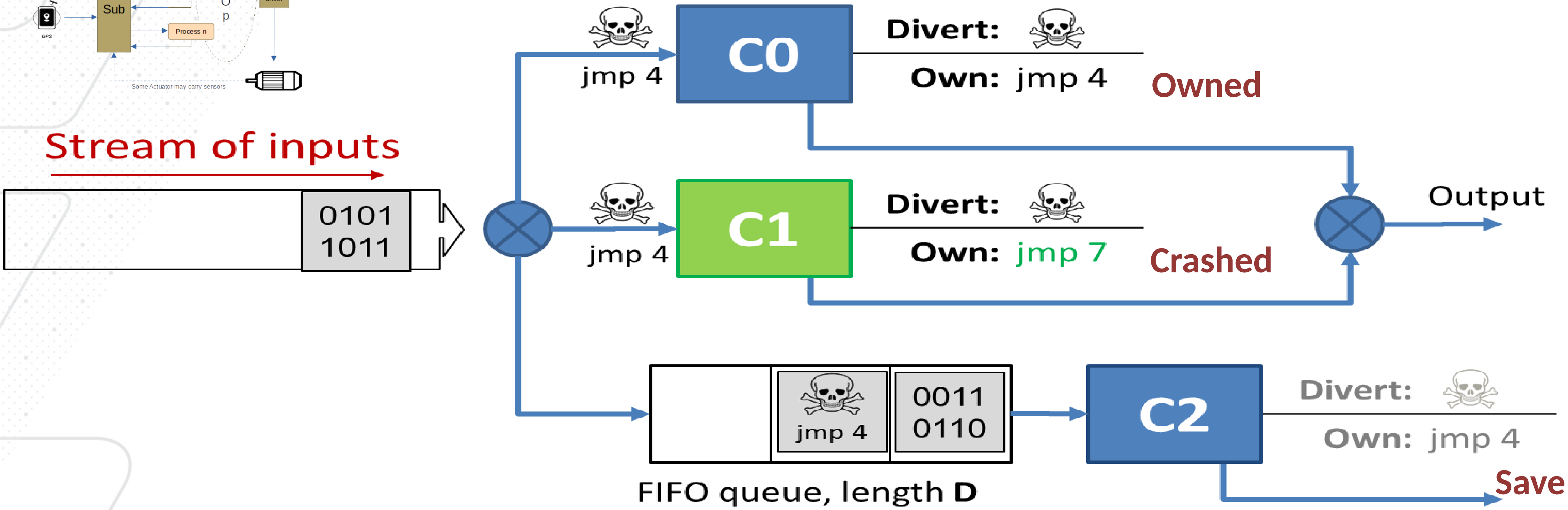
    - Must maintain a known good state,
    - check-pointing ???,
    - or may be not for LEGACY stuffs

Delayed Input Sharing





Delayed Input Sharing



Effect: 1 owned / 1 crashed, but C1's crash trigger is sitting in FIFO queue for C2

Yields cyber resilience if: * C1 crashes in time * Can safely flush C2's queue

Applicability of BFT++

BFT++ is applicable when:

$$T_{crash} \leq D * T_{SC} \leq T_d - T_r$$

(system dependent)

T_{crash}	=	Time/latency for engineered crash once corrupted (freq: GHz)
T_{SC}	=	Scan Cycle Period (1 epoch, freq: ~1 -300 Hz)
D	=	Time delay for backup system (length of FIFO queue, unit = # of epoch or scan-cycle, 1 or 2 epochs)
T_d	=	Maximum control loss tolerable by physical system (~ large # of epochs)
T_r	=	Recovery latency (one or more epoch)

Quicker system crashes → Shorter erroneous period → Less system disruption

Brittle is Better !!!

Does it work for system X?

Recall it depends on: $T_{crash} \leq D^*T_{SC} \leq T_d - T_r$

Cyber Allowances

- Cyber cycles within an epoch
- Slacks in an epoch

Physical Tolerance

- Tolerable # of failed epochs

Example:

- Medium class embedded processor = 1 GHz
- Control loop frequency = 100 Hz (ex: 747 inner loop)

length of an epoch = 10ms, meaning **10^7 cycles** available **per epoch**

X original processing	slack
	Assume 10%

Likely divert here (near where controller samples input)

Worst case divert here (means at least 10^6 slack cycles left)

- **1 delay slot buys at least 10^6 cycles, n delay slots buy (n-1) * 10^7 + 10^6 cycles**
- A single delay slot is often sufficient to crash and recover (2 slots at most)
- Recovery time calculating control for time t+2, from state t-D+1, within ~**1**+ epoch

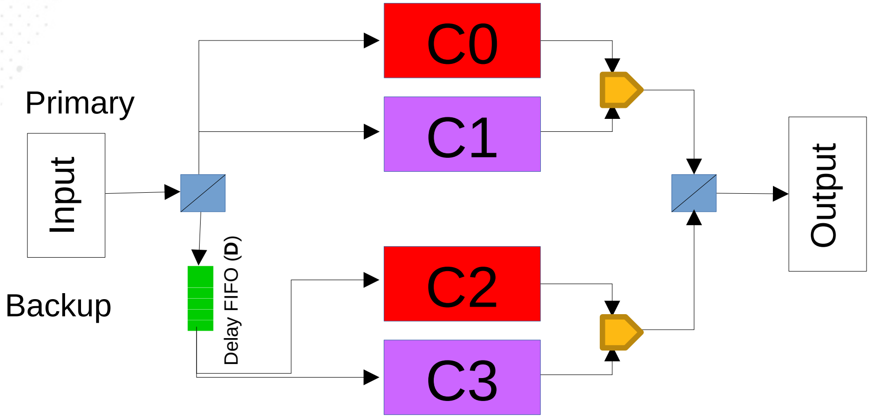
Does the physical system's inertia allow 1 or 2 cycles of non-control?

we believe so

Initial ONR Efforts

BFT++ v1 (Vanilla) – NRL

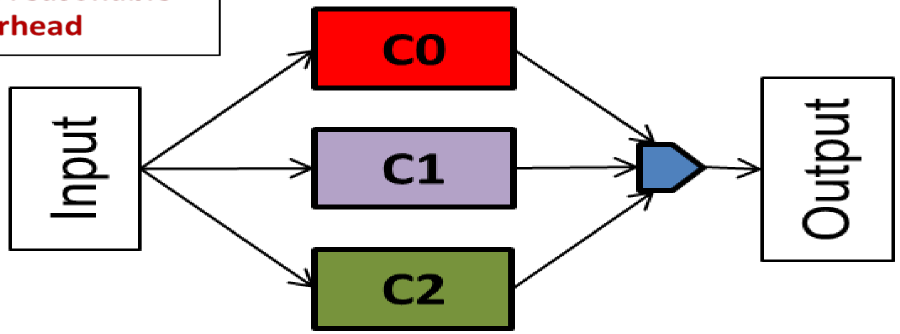
Original: all elements as depicted



BFT++ v2 – Georgia Tech

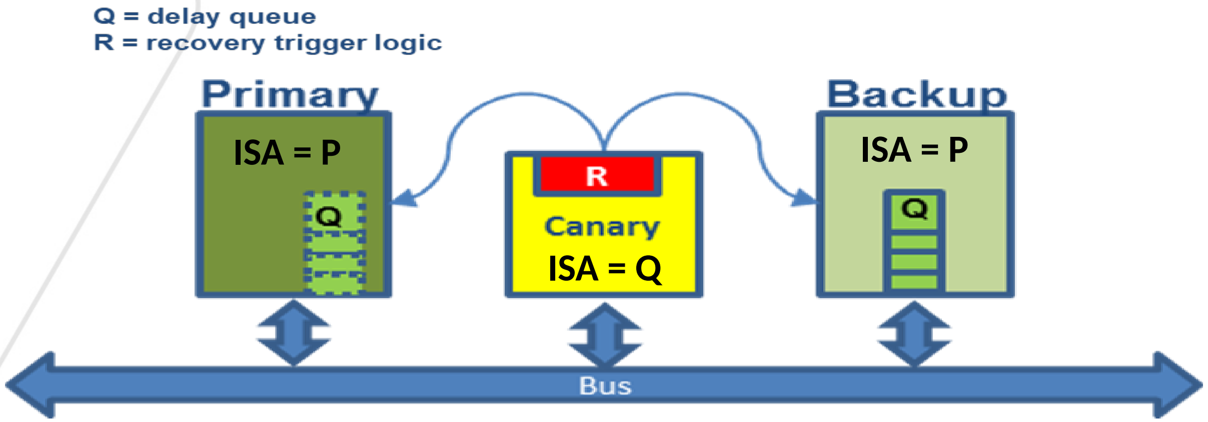
More robust, more costly: protection is diversified

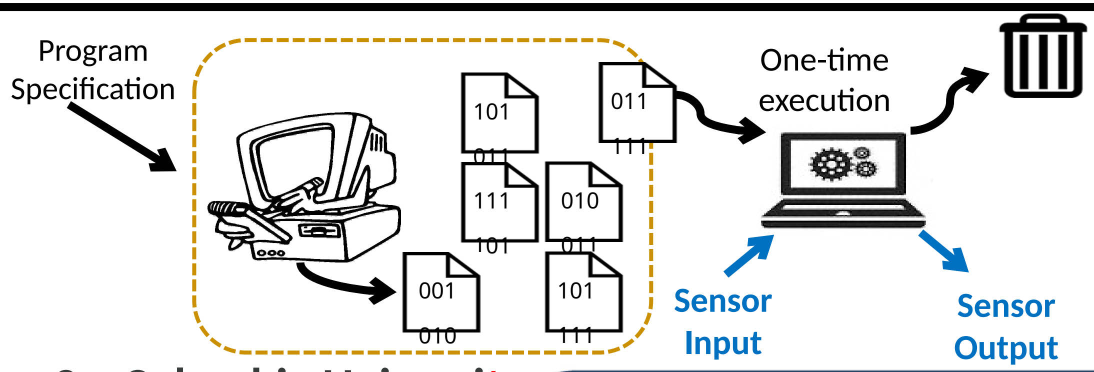
Diversified protection sets provides comprehensive protection with reasonable distributed overhead



BFT++ v4 (Rum Raisin)

Variation of the original: ISA diversity

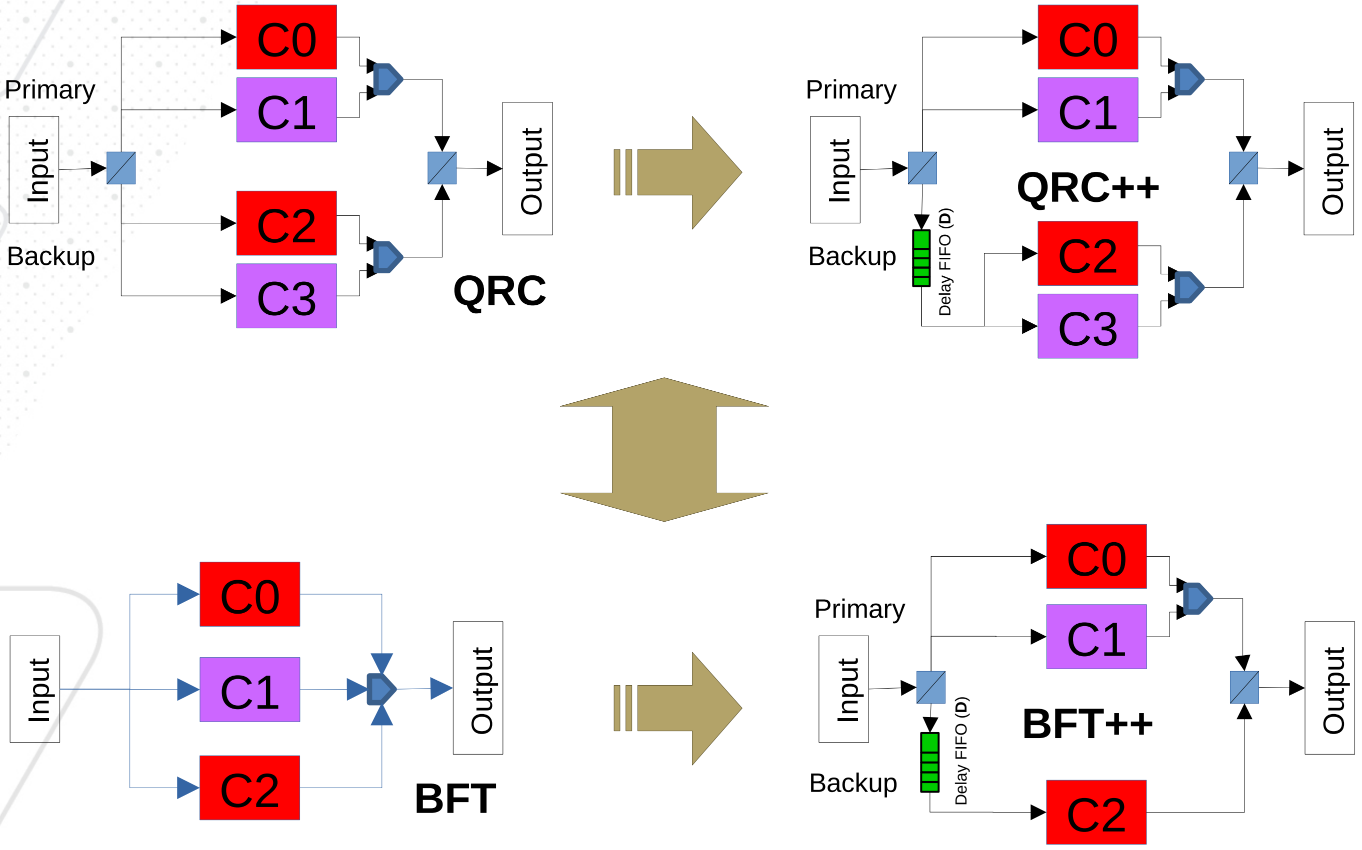




BFT++ v3 – Columbia University

Lightweight,
probabilistic guarantee:
requires no redundancy

BFT++ & QRC++

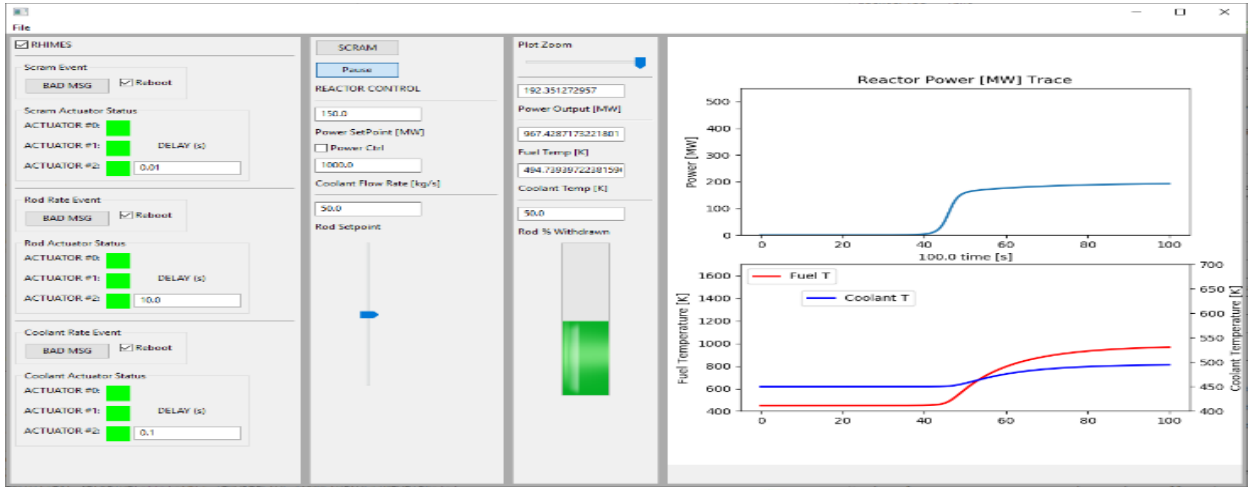


- Quad Redundant Controller (QRC) is often used in critical systems, e.g. flight control system (QFCS)
- also called Double Double

(not to be mistaken w/ In-N-Out Burger menu)

MITRE's RHIMES Laboratory Experiment

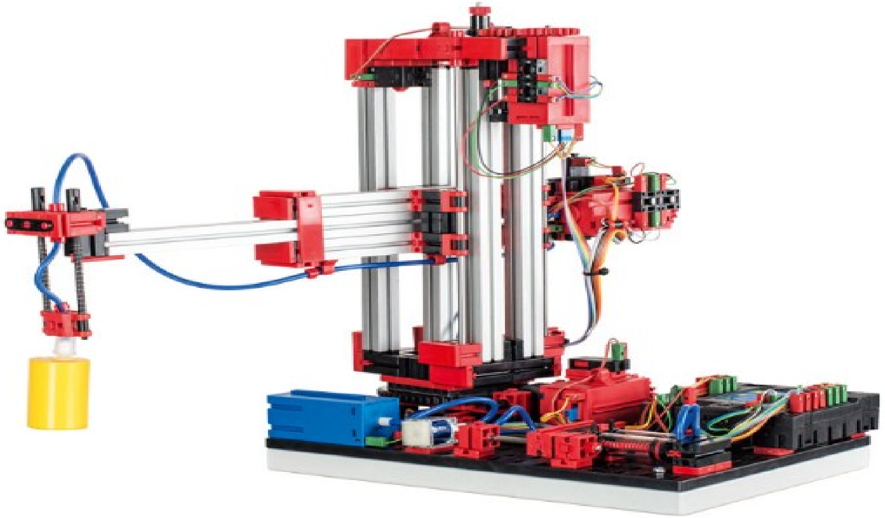
contributed by: Matt Mickelson, MITRE



Hypothesis: The time it takes to detect a crash and switch to a hot backup PLC is less than the time it takes to lose a "puck" due to inertia of the gripper losing grip.

Full recovery is acquired if the first 2 PLCs can be rebooted and reassume control.

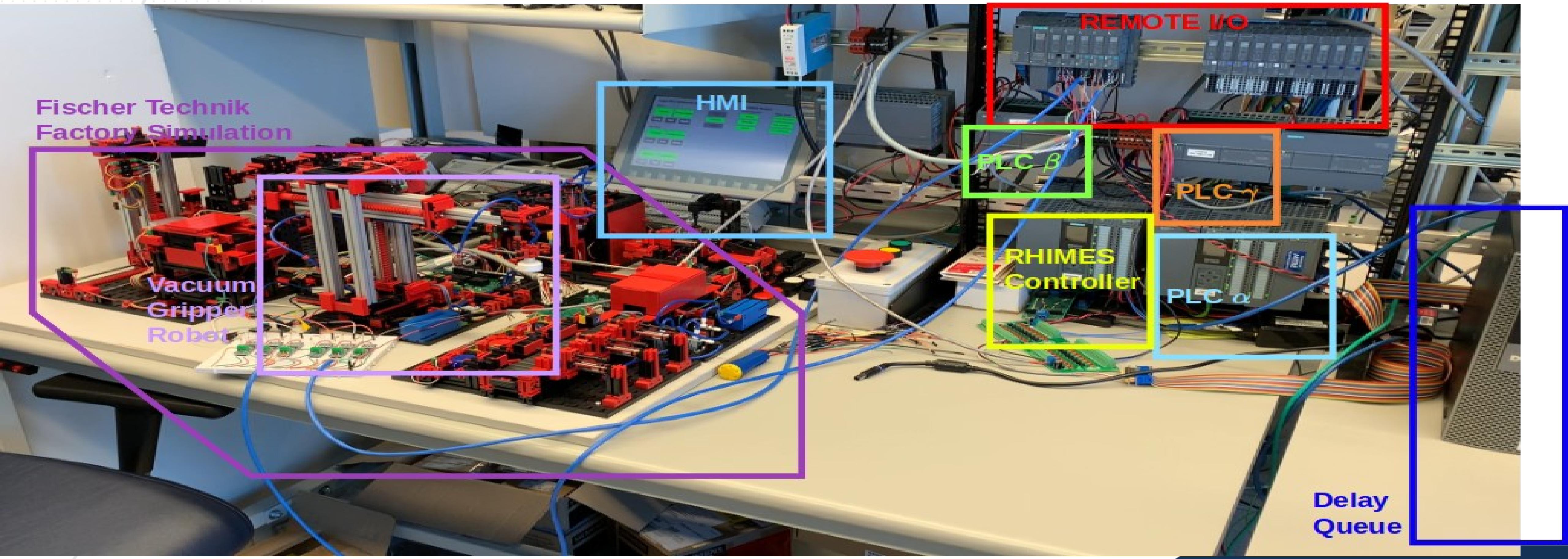
Hypothesis Confirmed



Georgia Tech College of Computing
School of Cybersecurity and Privacy

MITRE

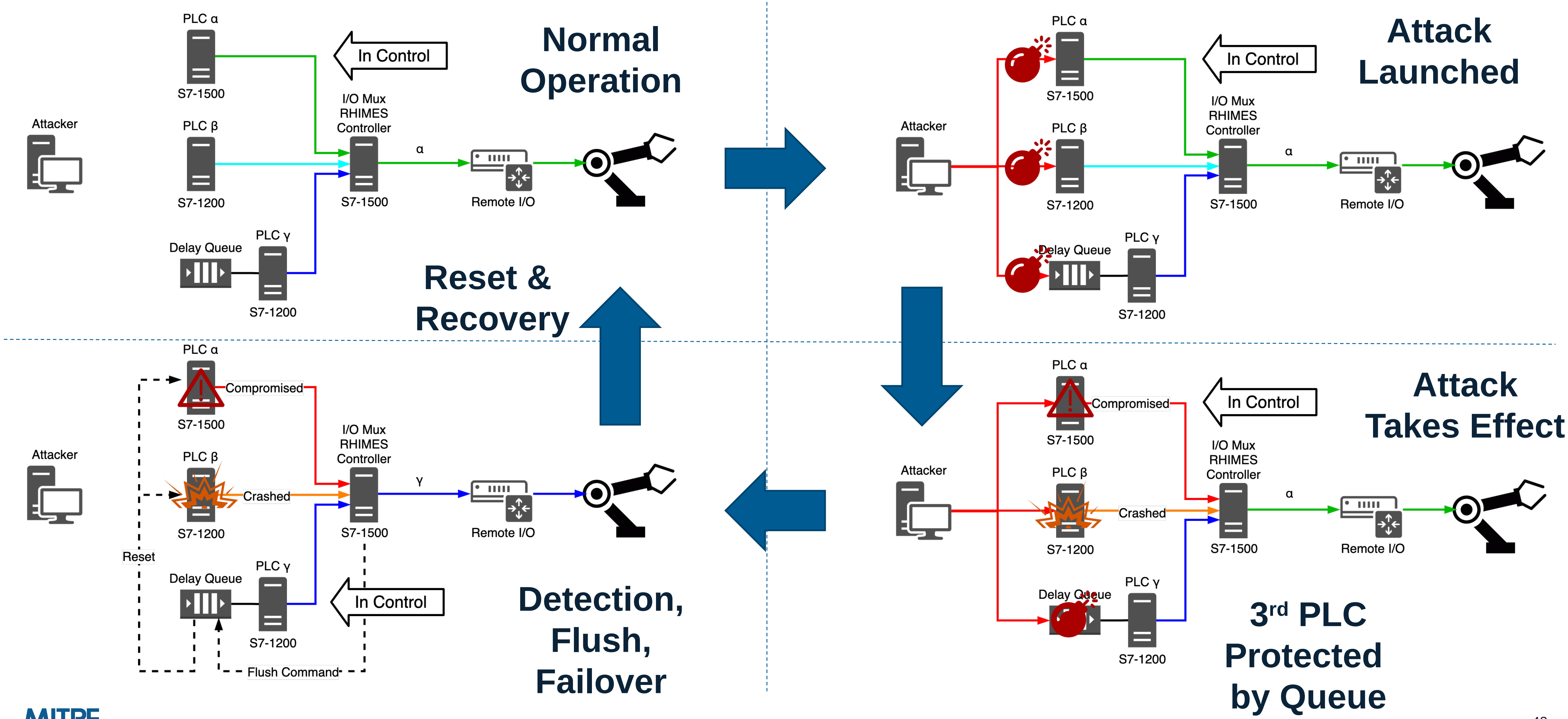
Reconfigurable RHIMES Implementation



MITRE

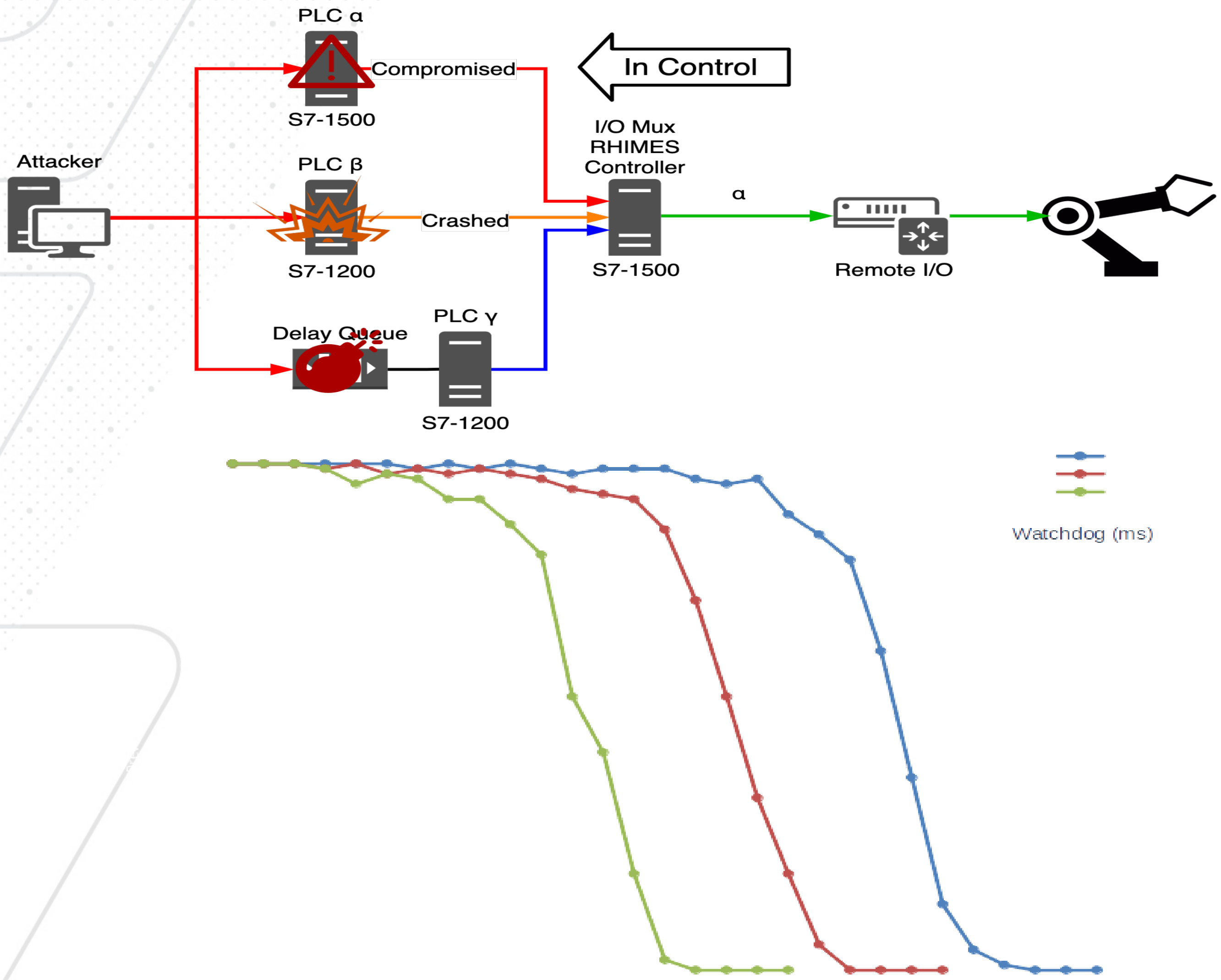
Georgia Tech College of Computing
School of Cybersecurity and Privacy

RHIMES Operational Sequence



MITRE

SCRAM inspired experiment result



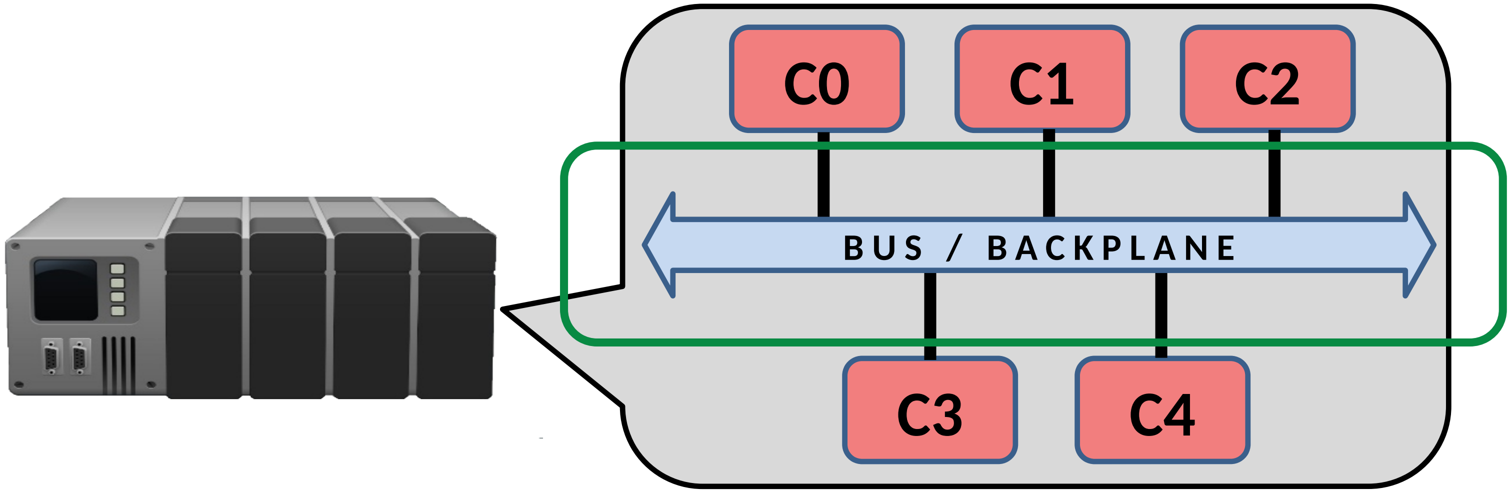
- Demonstrated continued operation of the gripper throughout the duration of the synthetic cyber attack.
- Fully characterized upper and lower limits of resiliency for the vacuum gripper
- Demonstrated recovery of non-protected PLCs to safe state.
- Demonstrated continuous operation despite repeated cyber exploit.

Georgia Tech College of Computing
School of Cybersecurity and Privacy

MITRE

Attack Resilient Communications

- Controllers could now withstand an attack, but recovery cycle is still triggered -> potential DoS



BUS LEVEL

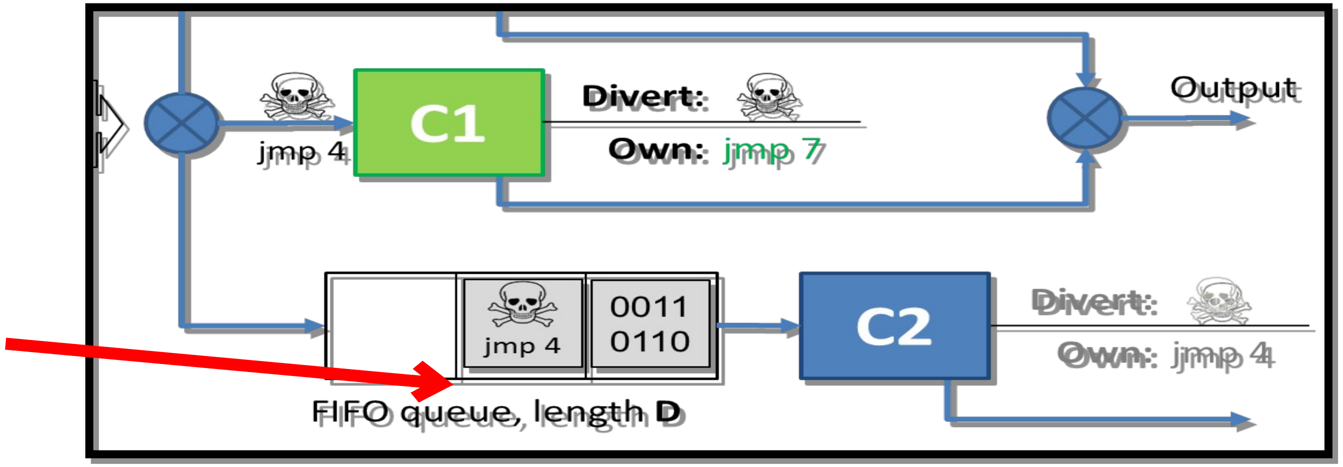
There is limits what inertia can tolerate

If Attack Keep on Coming

Malicious Input Filtering

Controllers could now withstand an attack, but recovery cycle is still triggered -> potential DoS

- Recall we isolated the malicious input with BFT++:



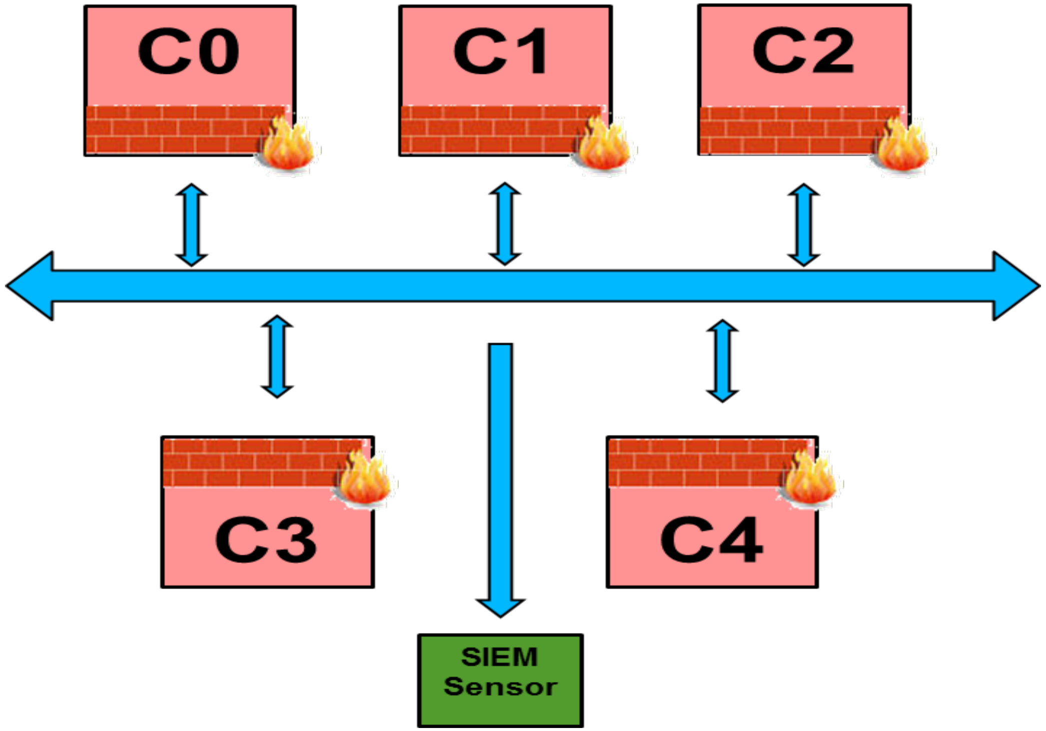
- We could drop/filter it at the bus if we had a filtering capability…

Attack Artifact is captured

Technical Approach: Bus

Retrofit binary code into input processing path (i.e., bus driver) of PLC firmware

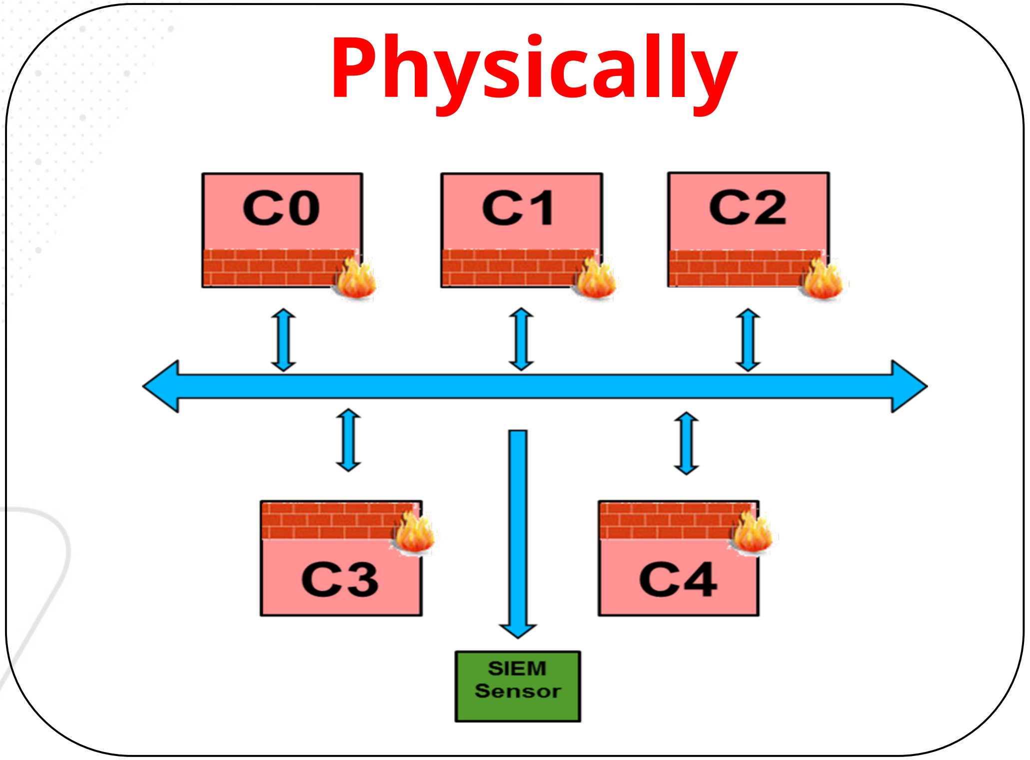
- Meet real-time requirements
- Gives ability to filter out malicious commands (and stop DoS on BFT++)
- Could also support:
  - Setting a mode of operation (strong crypto)
  - Mode-dependent whitelist
  - Expected range of operation and tolerances



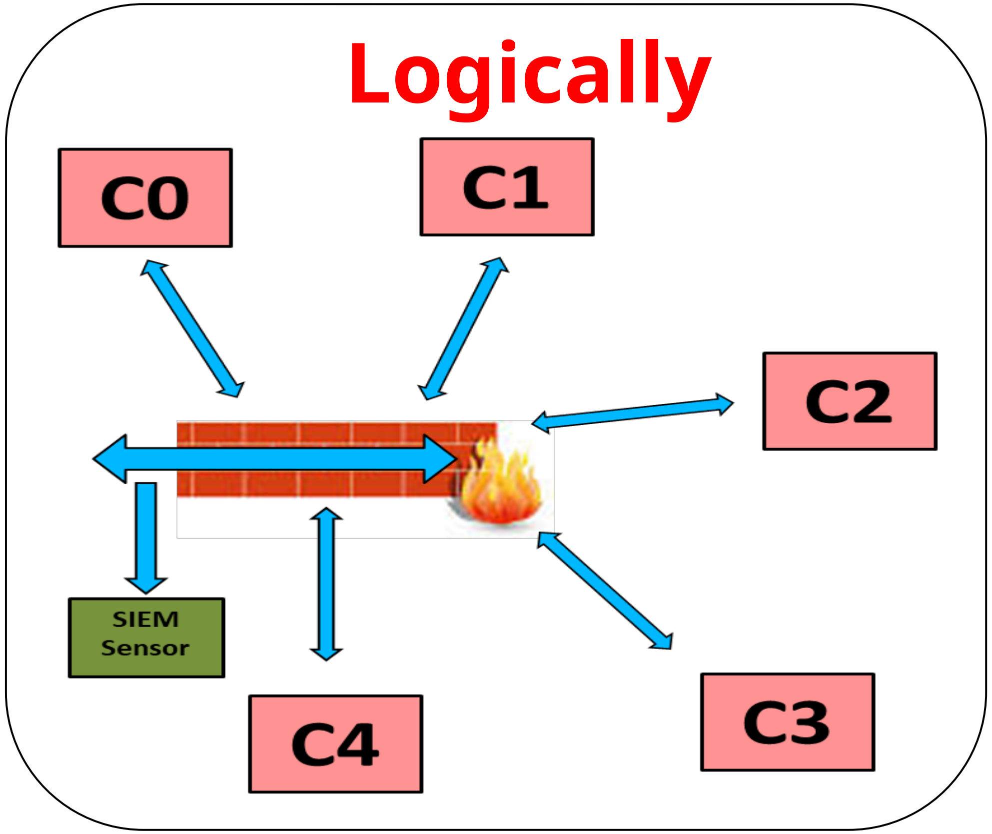
Transform Broadcast into Hub & Spoke

Or, *logically*…

Turns uncontrolled (broadcast-type) bus into a ‘hub-and-spoke’ topology with a central control point



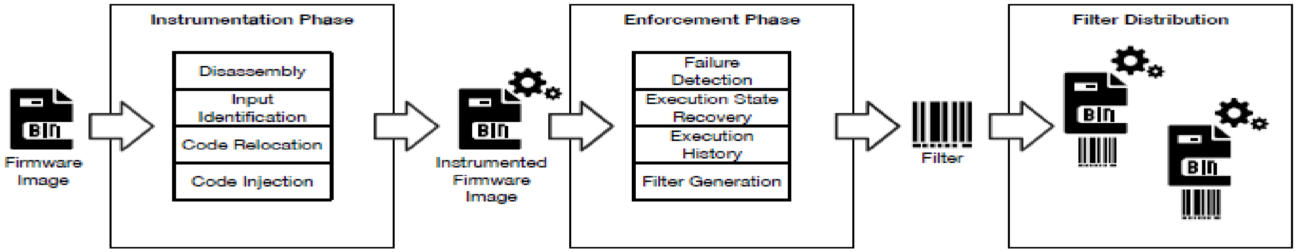
=



Initial ONR & ASD(R&E)-funded Efforts

Software Shim – UCSB + Boston Univ.

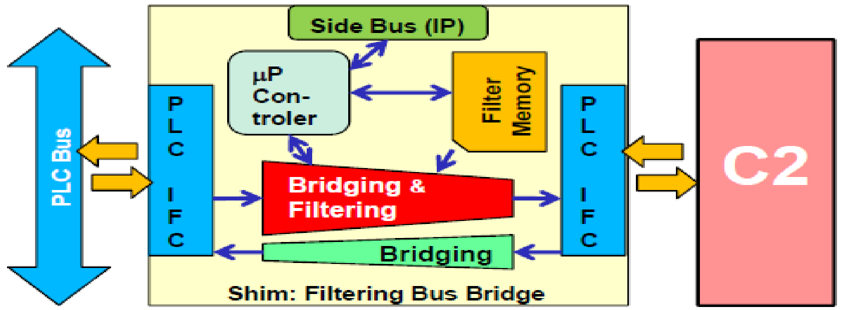
- Modify binary firmware automatically to insert shim
- Assume no access to source code
- Shim decouples firmware from inputs via a flexible, programmable input filtering capability
- Must not interfere with ability to meet real-time performance deadlines



Hardware Shim – PSU ARL

- Physical shim that sits in between card and backplane
- Uses FPGA for speed to keep real-time deadlines
- Shim decouples firmware from inputs via a flexible, programmable input filtering capability

Typical diagram for Shim's circuit

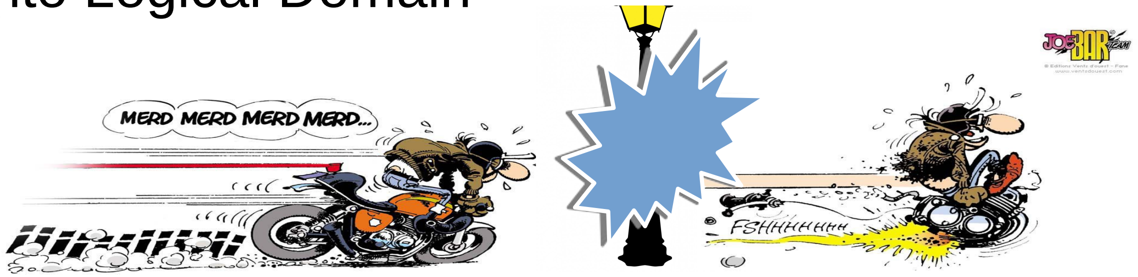


Machine Learning and CPS

- Contemporary Robot & Robotic Vehicle
  - heavily use ML (especially w/ RL)
  - for controlling its low level operation
  - for fault recovery
  - for adverse state recovery

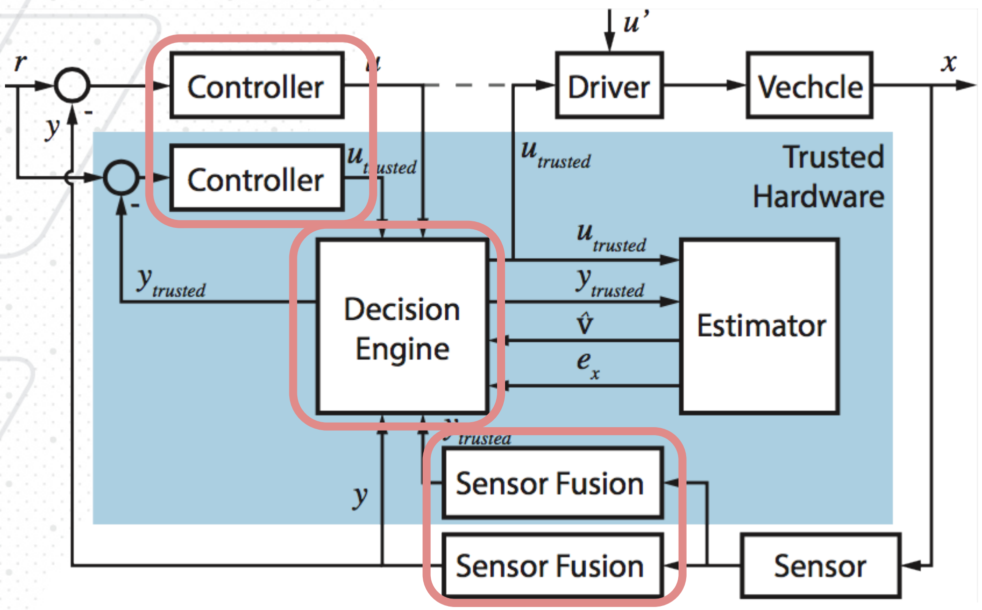
    e.g. Purdue's Learn2Recover
- ML has been very successful here
- It not unlike human's muscle memory & reflexes

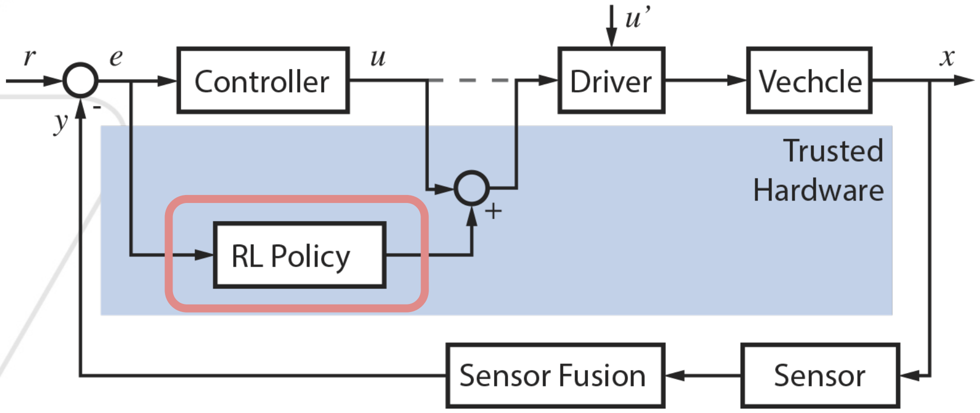
- ML is being used in Network level
- ML is also being used in mission level, which drives low-level
  - Object recognition & identification
  - Obstacle avoidance
  - Etc.
- It also use to assist planning – bridging into Logical Domain



Machine Learning is Statistical Machinery.
It inherits the strengths and limitations of Statistics.

Enhancing CPS robustness with Machine Learning





BlueBox Strategy

- Rely on redundancy to detect sensor fault
- Use estimator to detect actuator fault
- Recovery trigger by decision engine

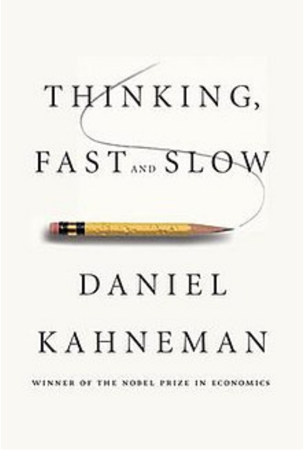
Reinforcement Learning

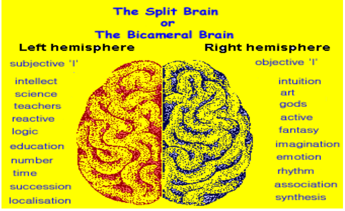
- No detection needed, always engaging
- Recover from sensor and actuator fault/attack
- Can retrofit existing controller
- Optimized to minimize position error

Purdue Univ.: Dongyan Xu, Xinyan Deng

Learn 2 Reason

Artificial Stuffs





Formal reasoning:

- Relatively slower inference
- Constructed knowledge w/ symbolic /semantic abstraction of the world, --> difficult to be complete
- Complex, hierarchical, logical structure
- Relatively rigid, does not handle uncertainties well (Fuzzy logic & probability help w/ uncertainty)
- Analyzable results, intermediate results & dominant inputs are traceable
- Can be **taught** and **trained**
- Less suitable for sensing, due to semantic gap

Deliberative Thinking

Statistical learning:

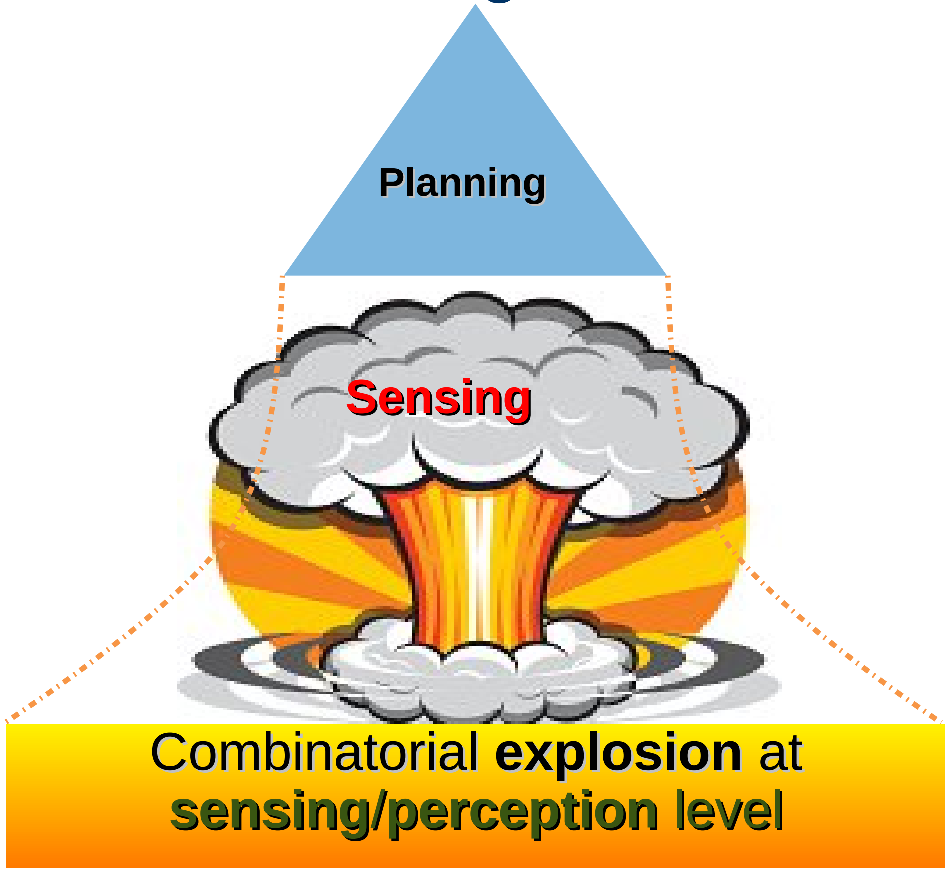
- Faster forward inference
- Knowledge acquired by sampling the world over time, --> quality depends on training samples
- Flat or simple structure with probabilistic representation
- Can deal with uncertainty really well
- Single point result, difficult to further analyze how result was reached
- Can **only** be **trained**
- Less suitable for planning

Gut Feeling

Learn 2 Reason

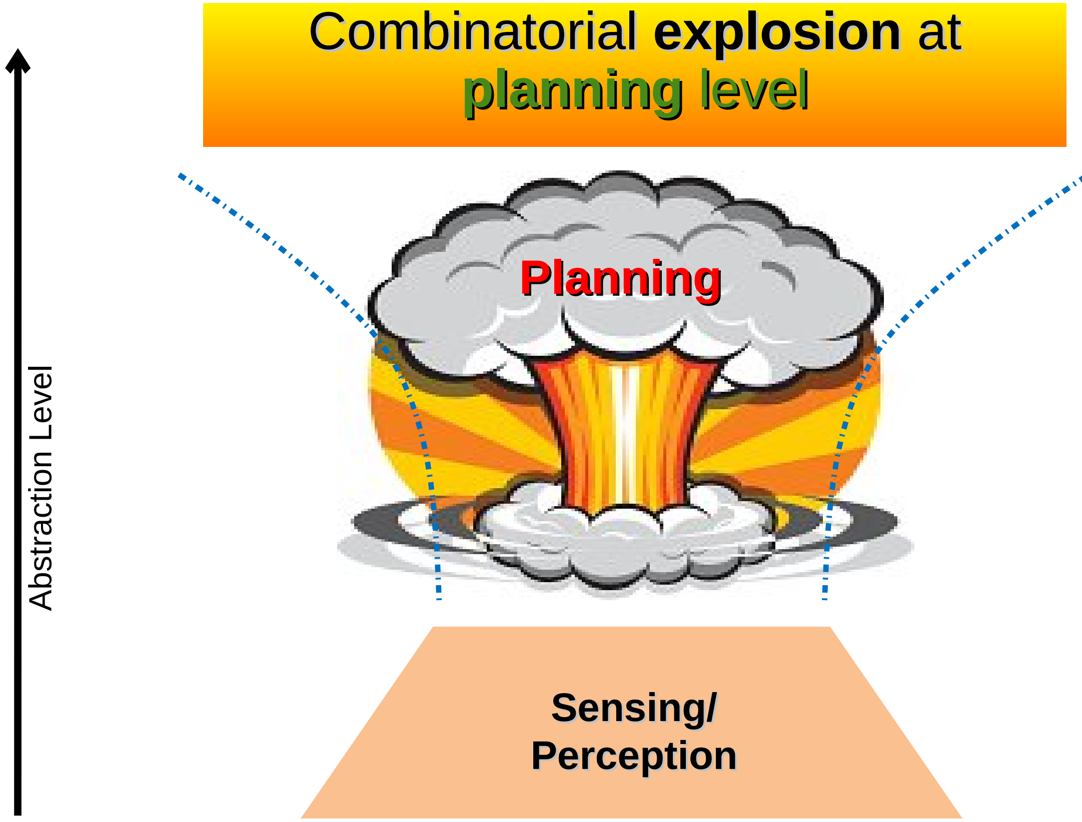
Artificial Stuffs… , again

Formal reasoning:



Overwhelming # of cases at sensing level;
reasonable completeness can be elusive

Statistical learning:



Every single path in the plan KB needs separate recognition

Formal --> knowledge representation = Set of Rules , probabilistic or not
Statistical --> knowledge representation = Set of Numbers

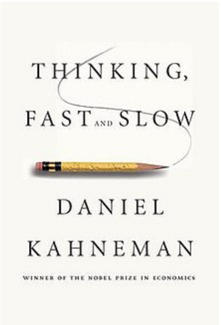
Statistic : Logic ~ Train : Teach

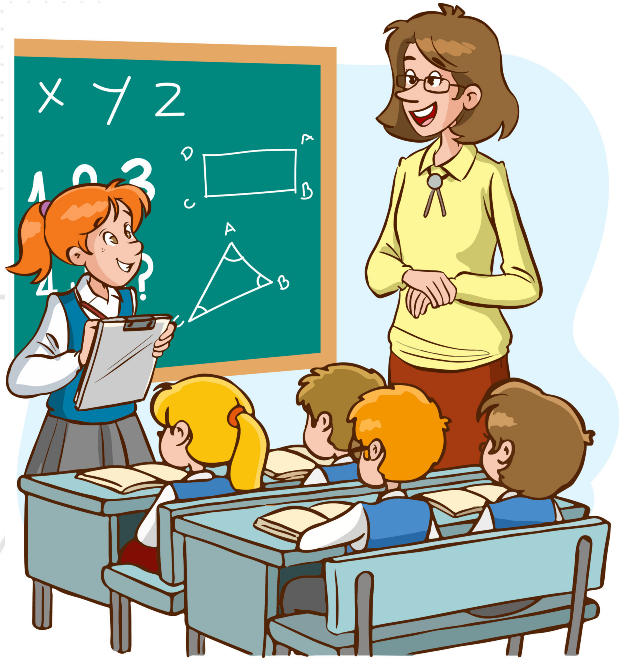
Slow – Logic – Teach

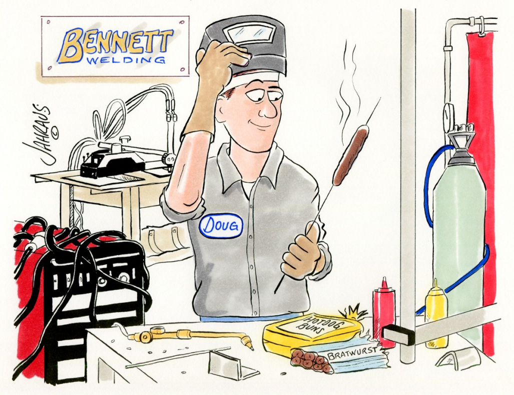
- Enable education
- Knowledge

Fast – Statistics – Train

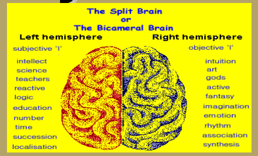
- Enable a lot of stuffs
- Skill, Gut-feel, Reflex







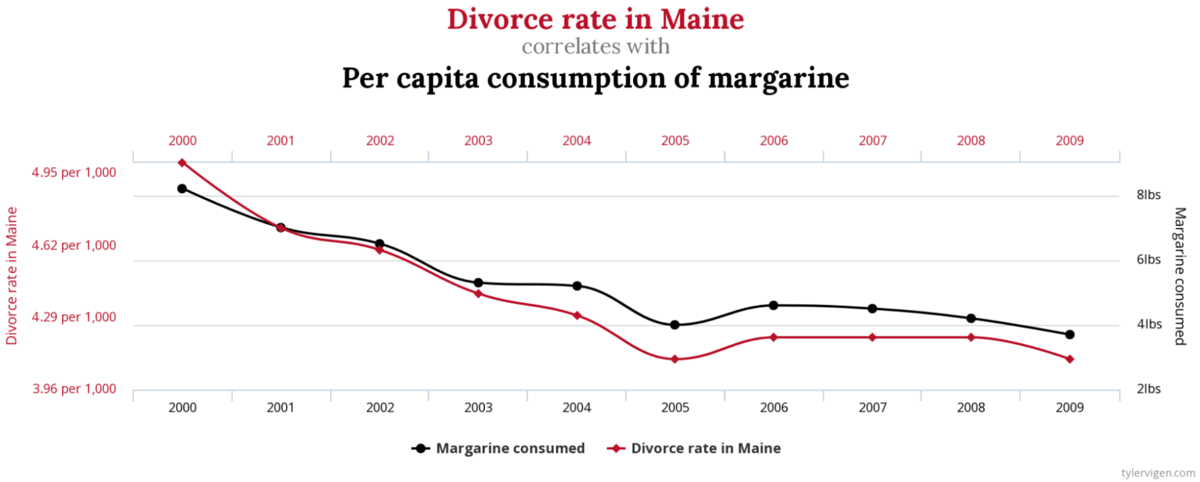
Many human activities are coaction between Knowledge & Skill

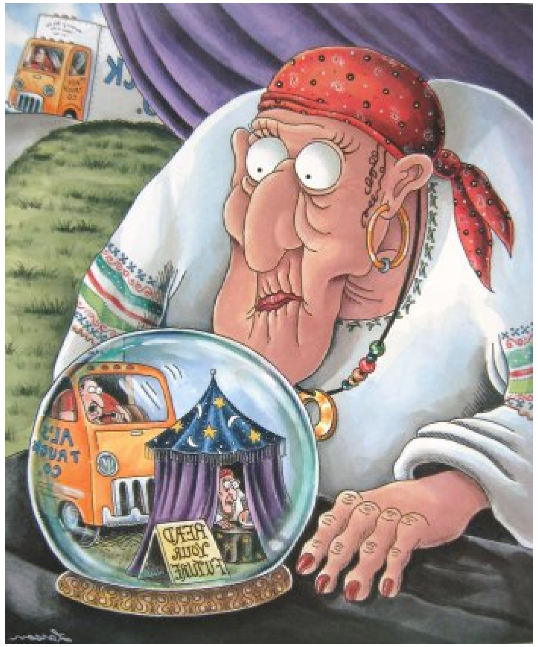


Recent AI success in the Industry

Success: Large Language Model – humongous worldwide data, what statistics do best

- Trained base on sentence completion (by masking) on world wide texts (& other)
- Learns correlated words and by association, objects & relations
- Problem: correlation vs. causation
  - Similar to big data fallacies





Can large data statistics reliably emulates logic???

Roles & Pitfall of AI in CPS

It is important to:

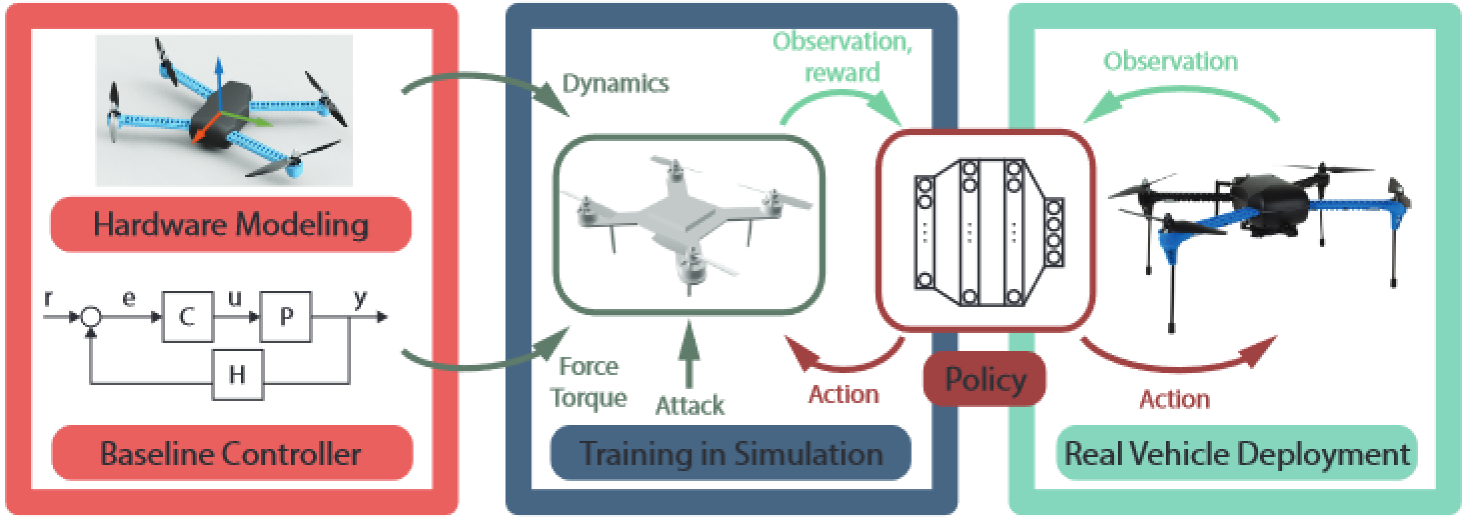
- Analyze & understand the problems, context, requirements and limitations
- Select ML/AI technique that match the problem/sub-problems &
- Appropriate strategy for acquiring training data

Example GAN (generator--discriminator):

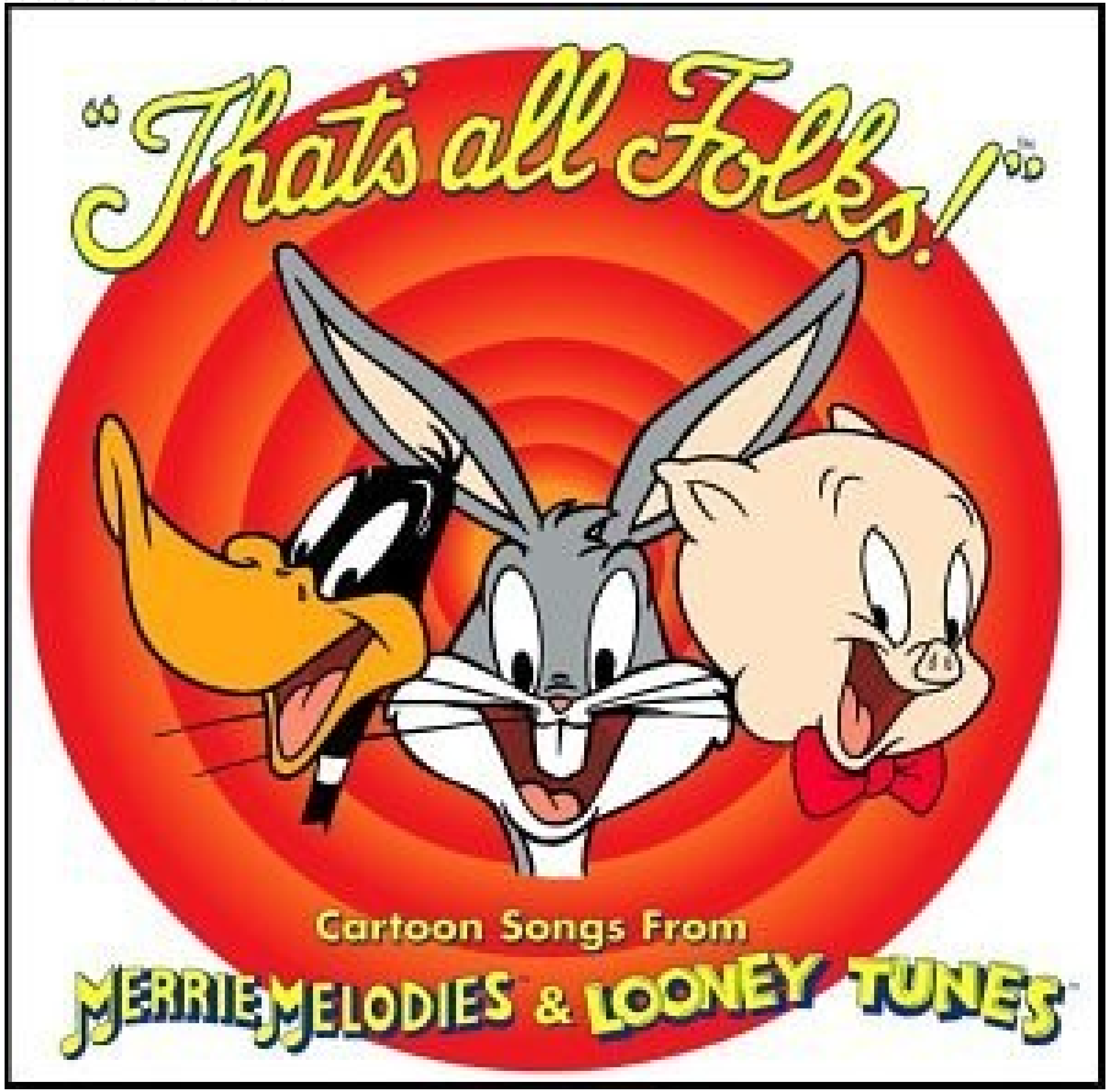
- ✔ GAN to to enhance robustness of ML-based anti-Malware
- ? GAN to generate Malware that can bypass Virus-Total (*inappropriate discriminator*)
- ? GAN to directly generate raw physical signals to enhance robustness of an ML-based CPS fault detection (*inappropriate generator*)
- ✔ GAN to vary parameters of physics model to enhance robustness of an ML-based CPS fault detection

Future Direction for AI in CPS

- Careful & appropriate deployment of large statistical model (e.g. LLM) can help in generating interesting v&v cases
- Cooperation of logic/symbolic & statistical models/algorithms will significantly enhance robustness & security



- Controller level: will continue to be the prominent role of ML
- Mission level: will play increasing roles, may need collaboration of symbolic for critical functionality
- Logical pipeline of ML will make it more interpretable



CPS → Physics Rules

References

Highlights

Pre-TPCP

- Purdue University
  - 7 conferences publications
  - 3 best papers
    (NDSS'16, FSE'16, Usenix Security'17)
- George Washington University
  - 2 refereed publications

Apply the technique to **18** Python projects on Github with the largest one having **54k LOC**

- Comparing with PySonar2 (by **Google**)
  - PySonar2 cannot type **51%** of the variables
  - **Purdue**'s tool can type **96.8%** of these variables with **79%** recall and **82.9%** precision
- Comparing with using learning only
  - Purdue's tool precision is **112%** better and recall is **68%** better

Best Artifact Award

ACM Foundation of Software Engineering 2016

Within TPCP

- Purdue University
  - More refereed publication
  - At least 1 best paper (OOPSLA'19)
- George Washington University
  - More refereed publication
  - 1 best paper (SecureComm'19)

GWU's StatSym

	KLEE w/ *StatSym*		Pure Sym. Exec. w/ KLEE	
Benchmark	#paths	time(sec)	#paths	time(sec)
polymorph	63	214.6	8368	3252.0
CTree	112	45.6	17575	Failed
thttpd	5168	1691.0	17882	Failed
Grep	11462	563.0	38708	Failed

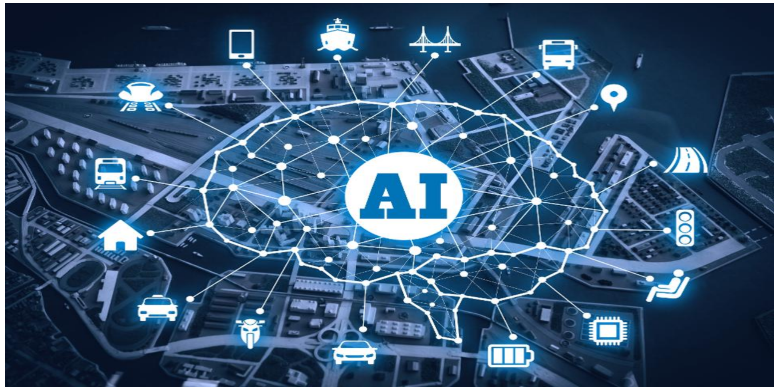
KLEE failed for 3 out of the four applications.
For smallest application, polymorph, StatSym speeds up KLEE by 15X

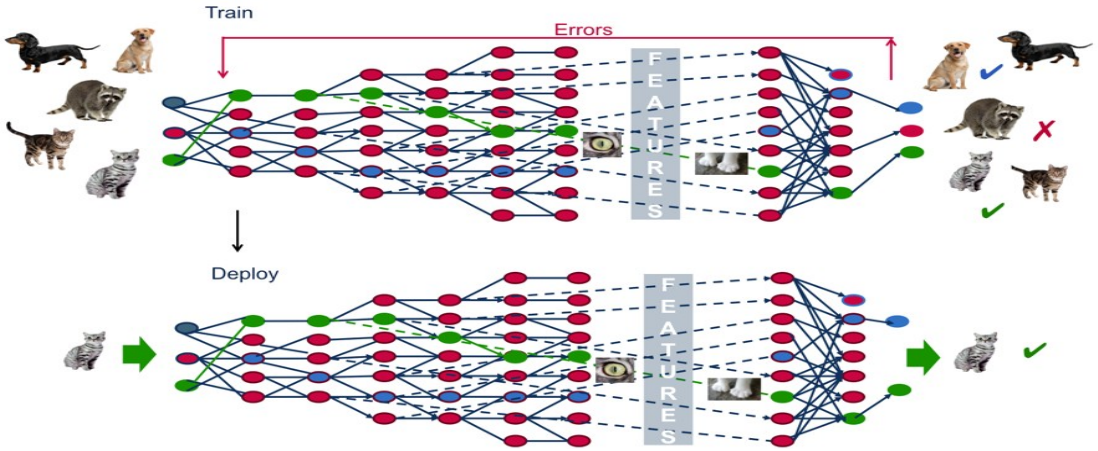
Performance Improvements have been Significant when Statistical Learning were Integrated with Formal Reasoning

AI success in the Industry

- Statistics based AI (NeuralNets, DeepLearning, Convolutional-NN, SVM, Recurrent-NN etc.) is the current rage in the commercial world & dominating the tech news.

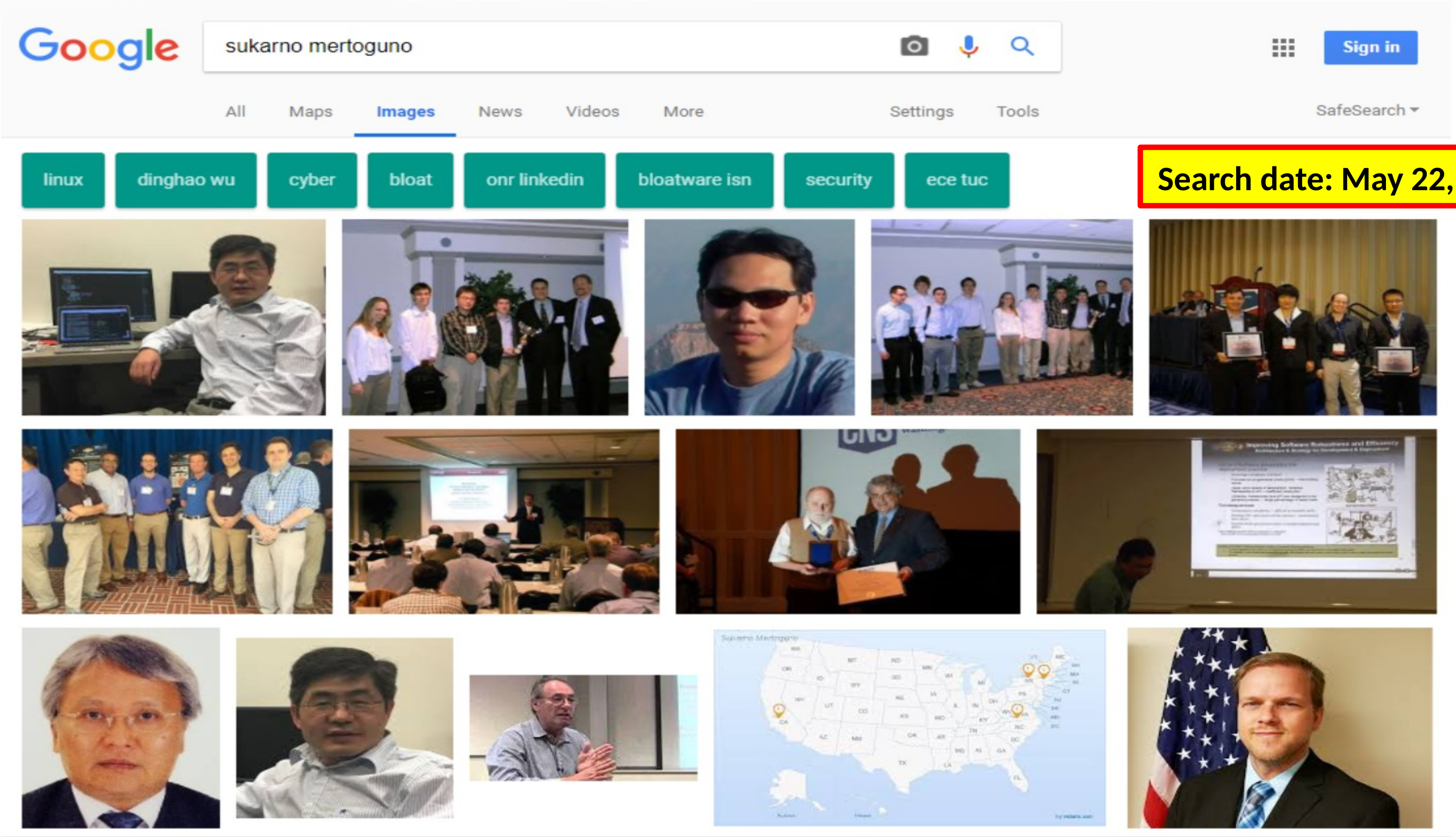
- Major 'success' of AI in commercial world
  - Face recognition – Facebook, google, Apple, …
  - Shopping recommendation – Amazon, Google, Facebook, …
  - Sending Advertisement – Google, Facebook, Amazon, …
  - Voice recognition/NL – Google, Amazon, Apple, …
  - Game playing – Google (AlphaGo, AlphaZero)
  - Etc.





Successes have been made in the area where mistakes have little consequences

AI success in the Industry ?



Search date: May 22, 2018

- Only 3 pictures are correct
- 3 Correct pictures & not rank #1 …. (I thought I have a unique name)
- Are we putting too much credit to our Industry Leaders ?
- Financial Success ≠ Technological Superiority

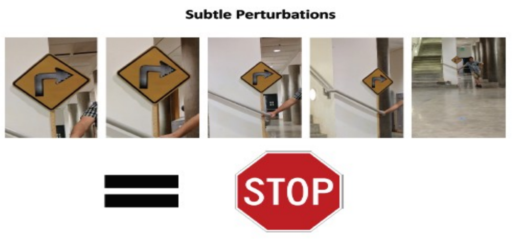
Note: This is more of Big-Data (statistics) than AI (machine learning), just to illustrate that some of the commercial world stuffs isn't very robust

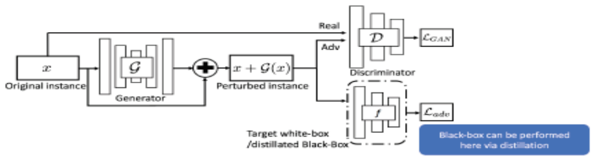
AI success in the Industry ??

Adversarial AI provides **systematic methods** for fooling machine learning (AI)

Attack on ML Algorithm

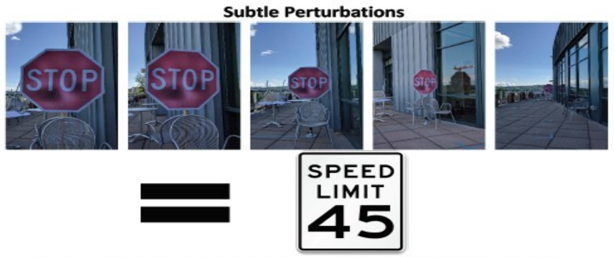
- Explaining and harnessing adversarial examples, J Goodfellow, J Shlens, C Szegedy, arXiv:1412.6572 (2014)
- Ensemble Adversarial Training: Attacks and Defenses. Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, Patrick McDaniel. 6th International Conference on Learning Representations (2018)
- Adversarial Examples for Malware Detection. Kathrin Grosse, Nicolas Papernot, Praveen Manoharan, Michael Backes, and Patrick McDaniel. European Symposium on Research in Computer Security (2017)
- Robust Physical-World Attacks on Machine Learning Models, Evtimov, Ivan, Kevin Eykholt, Earlence Fernandes, Tadayoshi Kohno, Bo Li, Atul Prakash, Amir Rahmati, and Dawn Song, arXiv:1707.08945 (2017).
- Adversarial examples for generative models, Jernej Kos, Ian Fischer, Dawn Song, arXiv:1702.06832 (2017)
- Delving into adversarial attacks on deep policies, Jernej Kos, Dawn Song, arXiv:1705.06452 (2017)
- Fooling Vision and Language Models Despite Localization and Attention Mechanism, Xiaojun Xu, Xinyun Chen, Chang Liu, Anna Rohrbach, Trevor Darrell, Dawn Song, arXiv:1709.08693 (2017)
- Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods, Nicholas Carlini and David Wagner, ACM Workshop on Artificial Intelligence and Security, 2017
- Towards Evaluating the Robustness of Neural Networks, , Nicholas Carlini and David Wagner, IEEE Symposium on Security and Privacy, 2017
- Audio Adversarial Examples: Targeted Attacks on Speech-to-Text, Nicholas Carlini and David Wagner, Deep Learning and Security Workshop, 2018
- …. and many more





$$\mathcal{L}_{GAN} = \mathbb{E}_{x \sim \mathcal{P}_{\text{data}}(x)} \log \mathcal{D}(x) + \mathbb{E}_{x \sim \mathcal{P}_{\text{data}}(x)} \log(1 - \mathcal{D}(x + \mathcal{G}(x)))$$

$$\mathcal{L} = \mathcal{L}^{f}_{adv} + \alpha \mathcal{L}_{GAN} + \beta \mathcal{L}_{hinge}$$



Evtimov, Ivan, Kevin Eykholt, Earlence Fernandes, Tadayoshi Kohno, Bo Li, Atul Prakash, Amir Rahmati, and Dawn Song. "Robust Physical-World Attacks on Machine Learning Models." *arXiv preprint arXiv:1707.08945* (2017).

Current statistics based machine learning is extremely brittle

AI success in the Industry ???

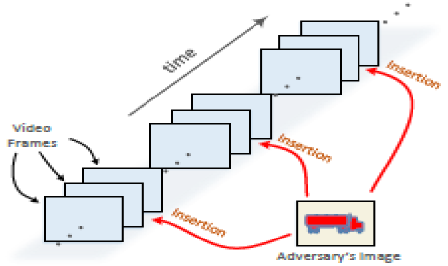
Attack on the seam of Problem Space & Approach

Fooling Google AI Services w/ **simple** stuffs

- Google Perspective – Toxicity score easily fooled w/ misspelling, extra character (space, dot, hyphenation), & can't deal w/ negation ('not', 'no'), arXiv:1702.08138 (2017)
- Google Cloud Video Intelligence – Inserting clear image periodically within a video stream dominates classification, arXiv:1703.09793 (2017)

Original Phrase (Toxicity Score)	Modified Phrase (Toxicity Score)
Climate change is happening and it's not changing in our favor. If you think differently you're an **idiot**. (84%)	Climate change is happening and it's not changing in our favor. If you think differently you're an **idiiot**. (20%)
They're **stupid**, it's getting warmer, we should enjoy it while it lasts (86%)	They're **st.upid**, it's getting warmer, we should enjoy it while it lasts (2%)
They are liberal **idiots** who are **uneducated** (90%)	They are liberal **i.diots** who are **un.educated** (15%)
idiots. backward thinking people. **nationalists**. not accepting facts. susceptible to **lies**. (80%)	**idiiots**. backward thinking people. **nationaalists**. not accepting facts. susceptible to **l.ies**. (17%)
They are **stupid** and **ignorant** with no class (91%)	They are **st.upid** and **ig.norant** with no class (11%)
It's **stupid** and wrong (89%)	It's **stuipd** and wrong (17%)
If they voted for Hilary they are **idiots** (90%)	If they voted for Hilary they are **id.iots** (12%)
Anyone who voted for Trump is a **moron** (80%)	Anyone who voted for Trump is a **mo.ron** (13%)
Screw you trump supporters (79%)	**S c r e w** you trump supporters (17%)

Original Phrase (Toxicity Score)	Modified Phrase (Toxicity Score)
Climate change is happening and it's not changing in our favor. If you think differently you're an idiot (84%)	Climate change is happening and it's not changing in our favor. If you think differently you're **not** an idiot (73%)
They're stupid, it's getting warmer, we should enjoy it while it lasts (86%)	They're **not** stupid, it's getting warmer, we should enjoy it while it lasts (74%)
They are liberal idiots who are uneducated. (90%)	They are **not** liberal idiots who are uneducated. (83%)
idiots. backward thinking people. nationalists. not accepting facts. susceptible to lies. (80%)	**not** idiots. **not** backward thinking people. **not** nationalists. accepting facts. **not** susceptible to lies. (74%)
They are stupid and ignorant with no class (91%)	They are **not** stupid and ignorant with no class (84%)
It's stupid and wrong (89%)	It's **not** stupid and wrong (83%)
If they voted for Hilary they are idiots (90%)	If they voted for Hilary they are **not** idiots (81%)
Anyone who voted for Trump is a moron (80%)	Anyone who voted for Trump is **not** a moron (65%)
Screw you trump supporters (79%)	**Will not** screw you trump supporters (68%)

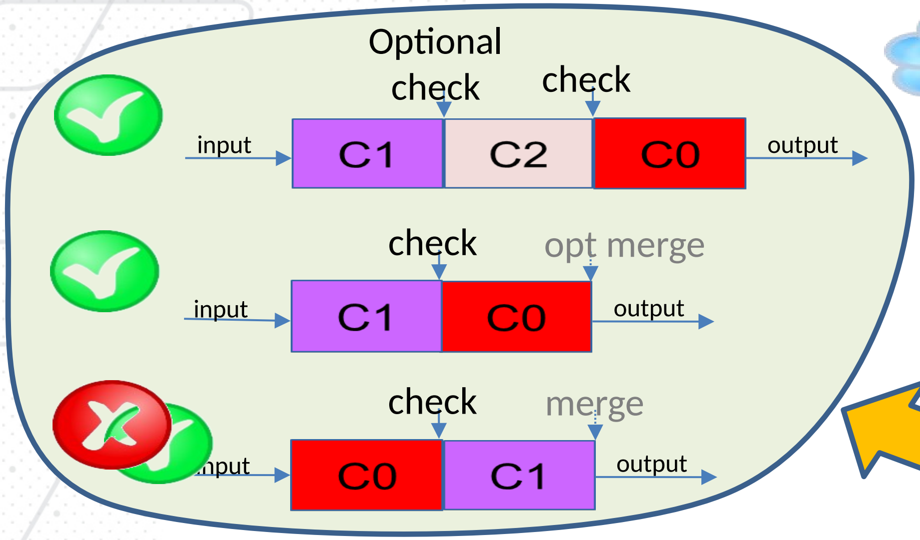


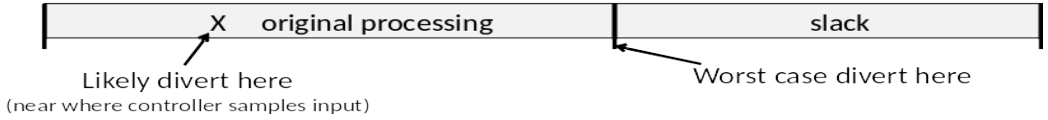
Video Name	Inserted Image	Video Label Returned by API (Confidence Score)
"Animals.mp4"	"Car"	Audi (98%)
	"Building"	Building (89%)
	"Food Plate"	Pasta (99%)
	"Laptop"	Laptop (91%)
"GoogleFiber.mp4"	"Car"	Audi (98%)
	"Building"	Classical architecture (95%)
	"Food Plate"	Noodle (99%)
	"Laptop"	Laptop (91%)
"JaneGoodall.mp4"	"Car"	Audi (98%)
	"Building"	Classical architecture (95%)
	"Food Plate"	Pasta (99%)
	"Laptop"	Laptop (91%)

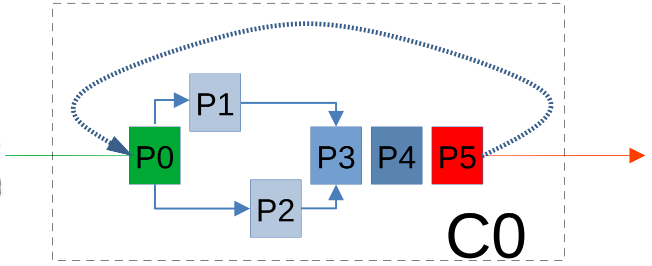
Current statistics based AI/machine-learning is extremely brittle & …. dumb ???

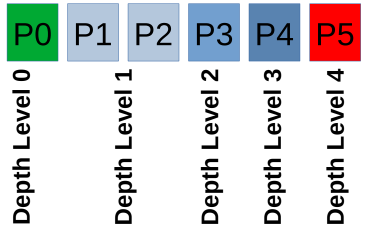
Diversified Redundancy on Single Processor

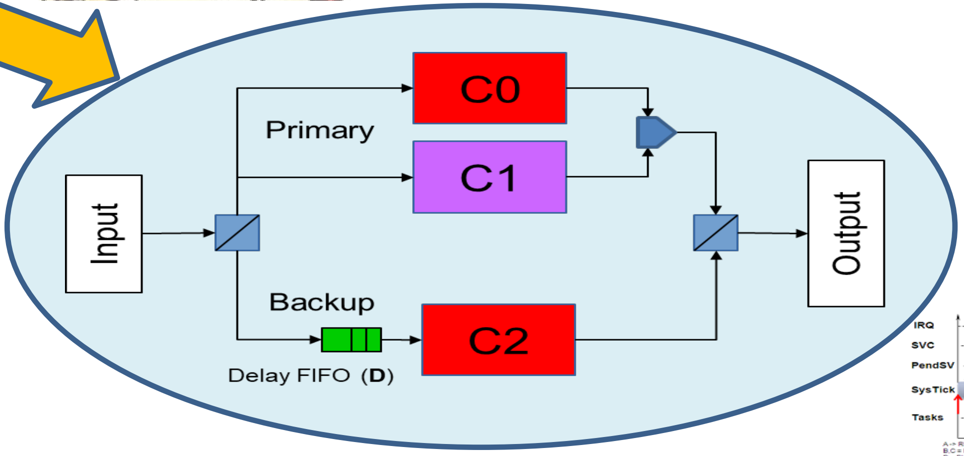
Parallel vs. Serial





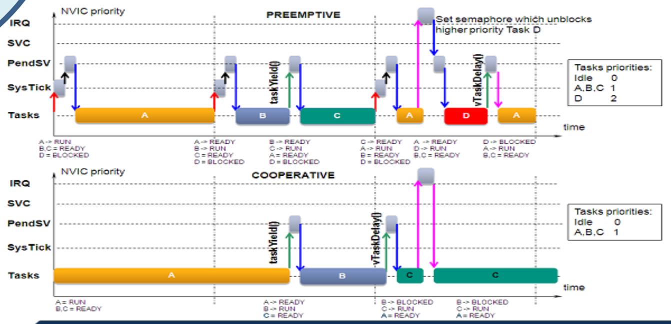






Legends:

- **C0** Original control program
- **C1** & **C2** Diversified programs
- If **check** fails, drop input

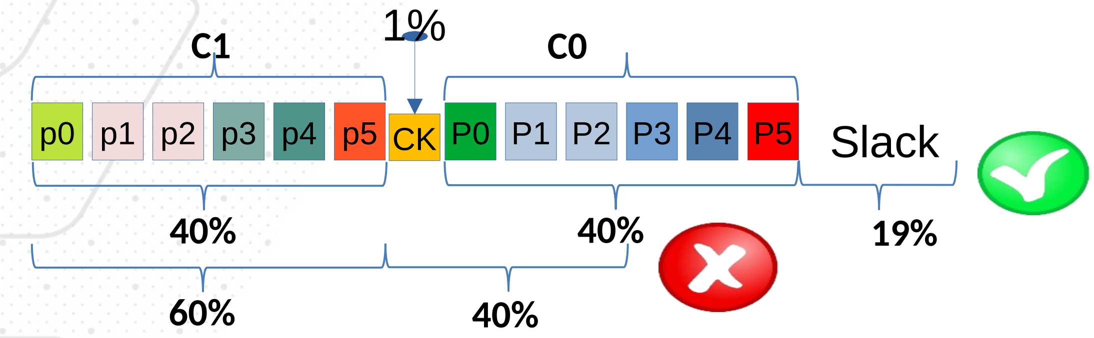


Design parameters:

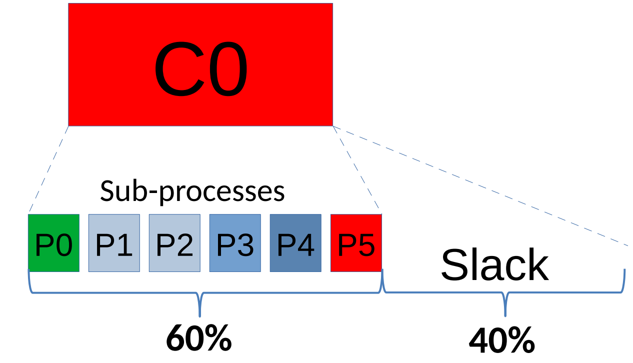
- Slack availability
- Number of sensitive processes
  – Depth relative to input
- Pre-emptive vs. Co-operative scheduling

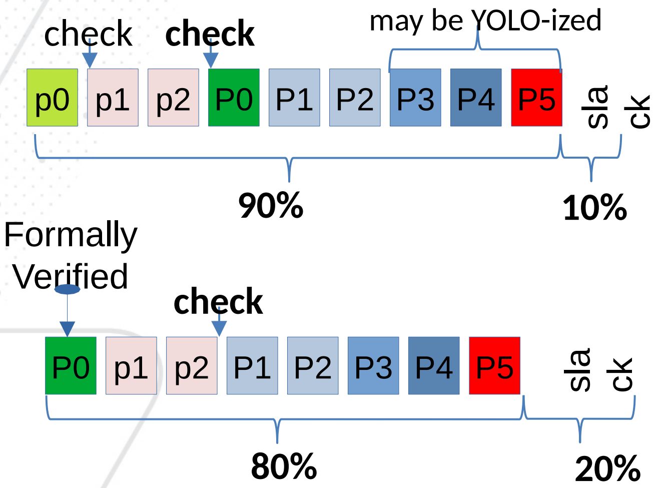
Diversified Redundancy on Single Processor

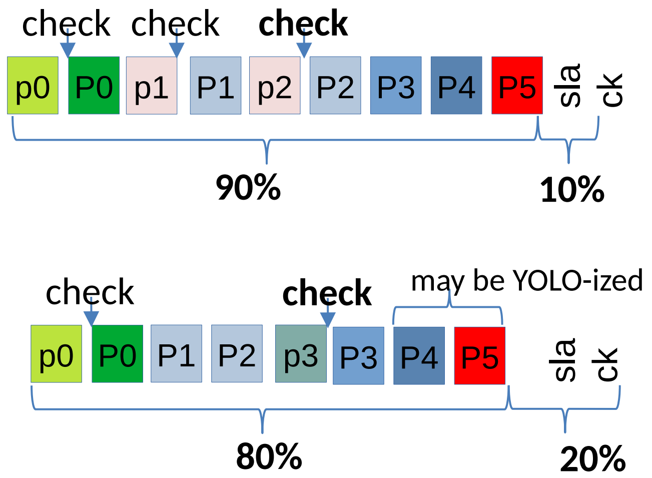
Serial in Finer Granularity

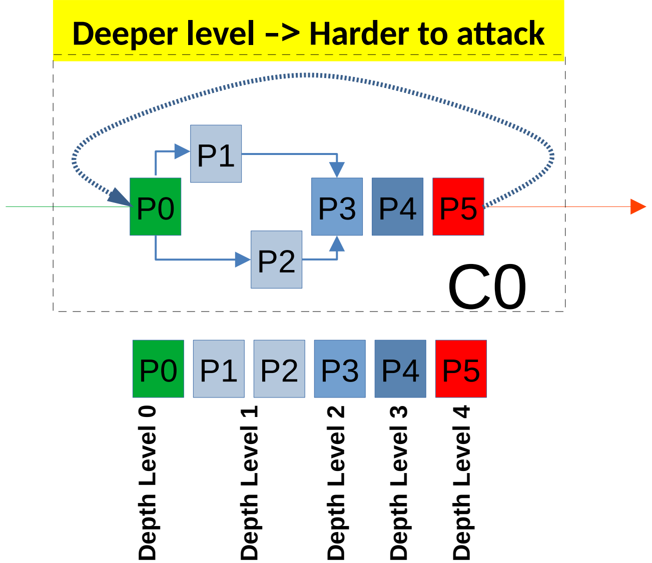


Legends:
- **C0** Original program
- **C1** Diversified program
- If **check** fails, drop input



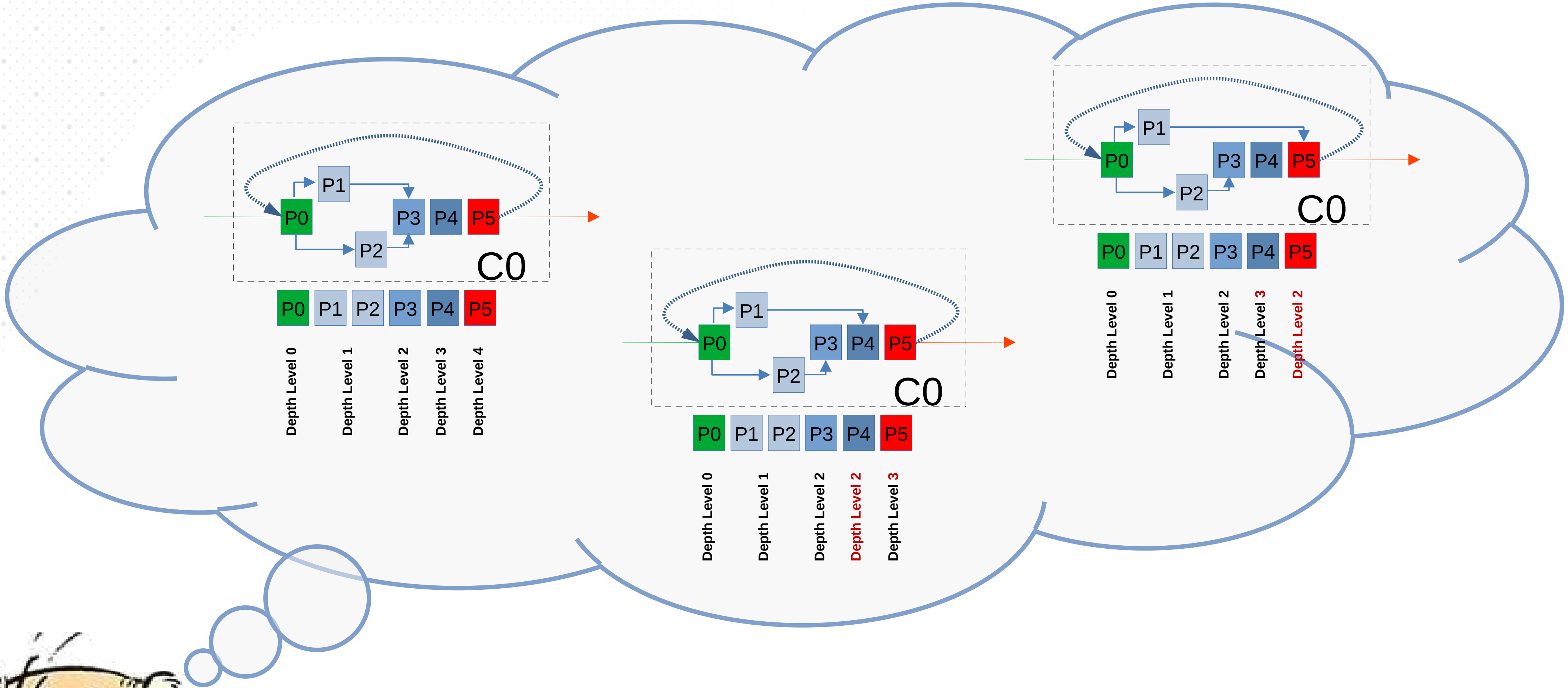






1001 ways to implement BFT++ concept w/ sub-process replication;
- Diversified replication can co-exist w/ Formal-Methods, Protections & YOLO
- Engineering for sub-processes replication depends on:
  - Available Slack & Desired Slack,
  - Sub-processes' Depth Level,
  - & particular sub-process' properties

Georgia Tech College of Computing
School of Cybersecurity and Privacy



Sensitivity Level: Minimum Distance to Inputs

BFT++ beyond CPS

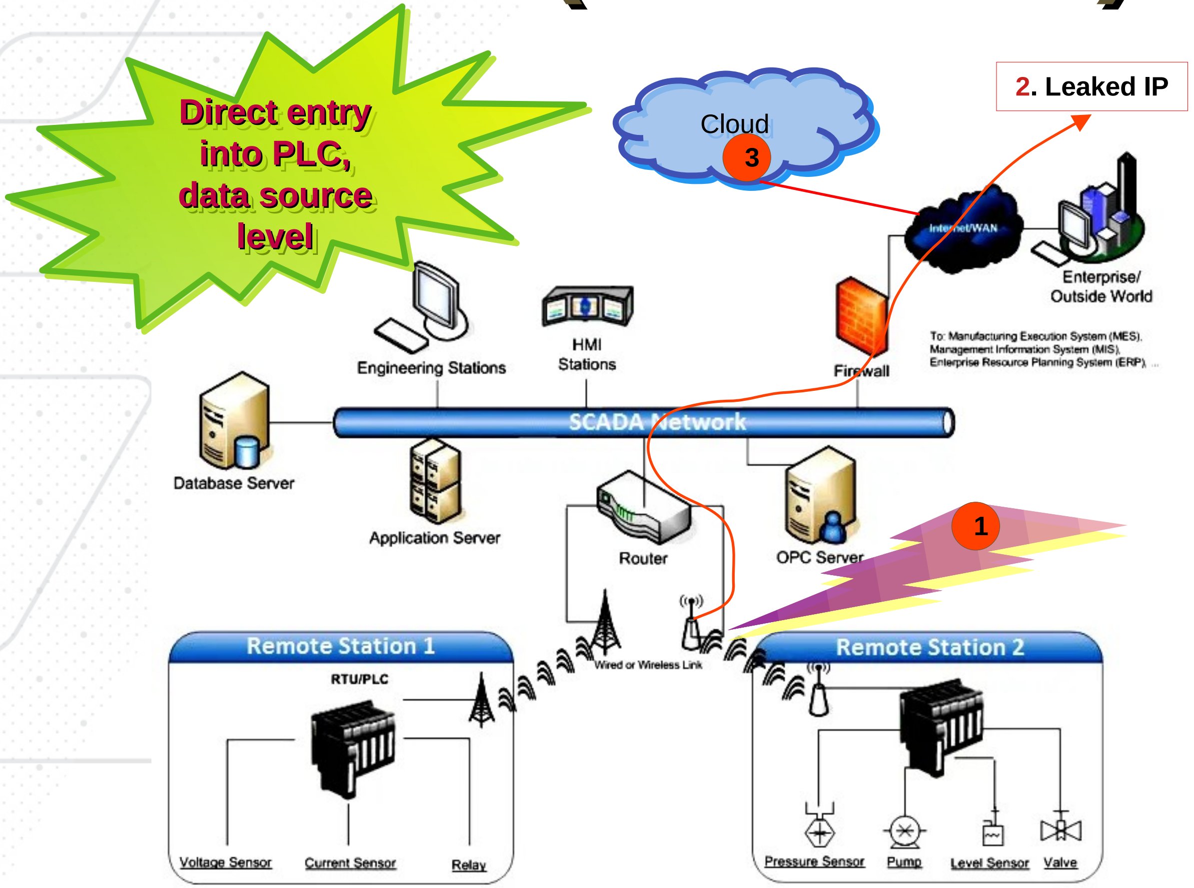
- BFT++ also applicable to:
  - Scanning radar –> Target has inertia, Scanner has periodicity
  - Many stateless & streaming transport
    - UDP
    - Streaming videos, audio, VOIP
    - Etc.
  - Anything that can tolerate ***small*** disruption or loss of data.

BFT++ is also applicable to application with Virtual Inertia

Comprehensive Protection with Bunshin

- Accumulated execution slowdown
  - Example: Softbound + CETS → **110%** slowdown
  - Bunshin: Reduce to 60% or 40% (depends on the config)
- Implementation conflicts
  - Example: AddressSanitizer and MemorySanitizer
  - Bunshin: Seamlessly enforce conflicting sanitizers

Recent (white-hat) Hacks



OpenControl SCADA Network Architecture

Otorio:

- The team discovered relatively simple ways for an attack to **hack** industrial **Wi-Fi access points** and **cellular gateways** in many ways:
  1. The researchers armed with a laptop could find and drive to a plant location and connect to the operational network.
  2. They also could reach the plant wireless devices via oft-exposed IP addresses inadvertently open to the public Internet.
  3. They could reach the OT networks via blatantly insecure cloud-based management interfaces on the wireless access points.
- and wage **man-in-the-middle** attacks to manipulate or sabotage physical machinery in production sites.

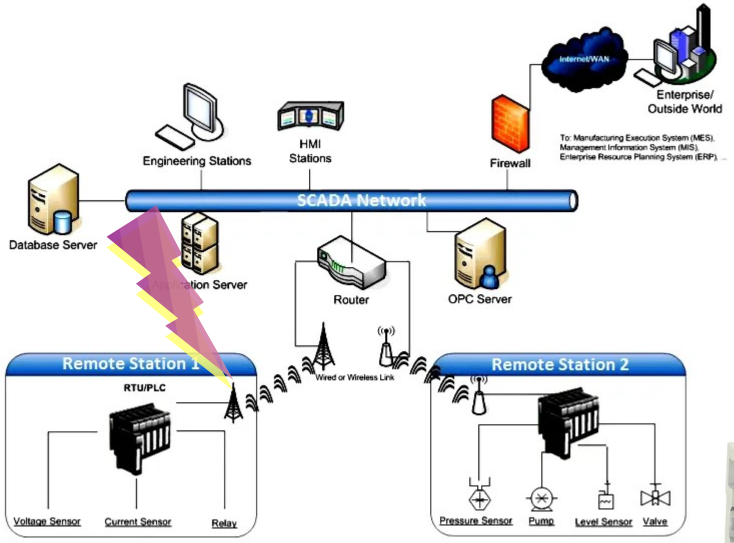
Dispelling conventional wisdom about the security of network segmentation & Highlighting vulnerability from third-party connections to the network

https://www.darkreading.com/ics-ot/ot-network-security-myths-busted-in-a-pair-of-hacks 02/2023

Recent (white-hat) Hacks

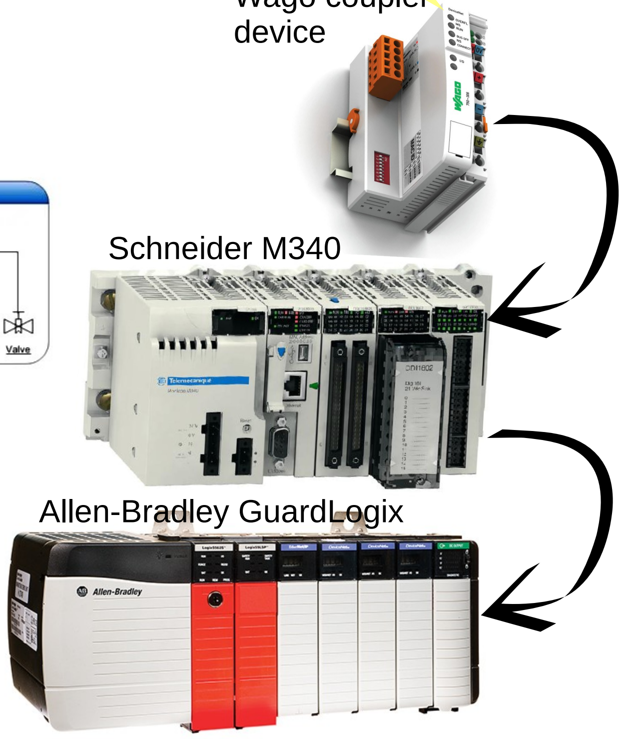
Forescout:

- Hacked a Wago coupler device:
  - connects ETHERNET to the modular I/O System,
  - detects all connected I/O modules and creates a local process image
  - supports a wide variety of standard ETHERNET protocols (e.g., HTTP(S), BootP, DHCP, DNS, SNMP, (S)FTP). An integrated Webserver provides user configuration options, while displaying the coupler's status information
- Get to Schneider M340 PLC:
  - Vulnerabilities: CVE-2022-45788 (remote code execution), CVE-2022-45789 (authentication bypass)
  - bypass the PLC's internal authentication protocol and
- move through the PLC to other connected devices, incl: an Allen-Bradley GuardLogix safety control system that protects plant systems by ensuring they operate in a safe physical state.
  - able to manipulate the safety systems on the GuardLogix backplane.
- Forescout, didn't just hack a PLC via an inherent vulnerability. They instead pivoted from the PLC to other systems connected to it in order to **bypass the security and physical safety checks** within the OT systems.



OpenControl SCADA Network Architecture

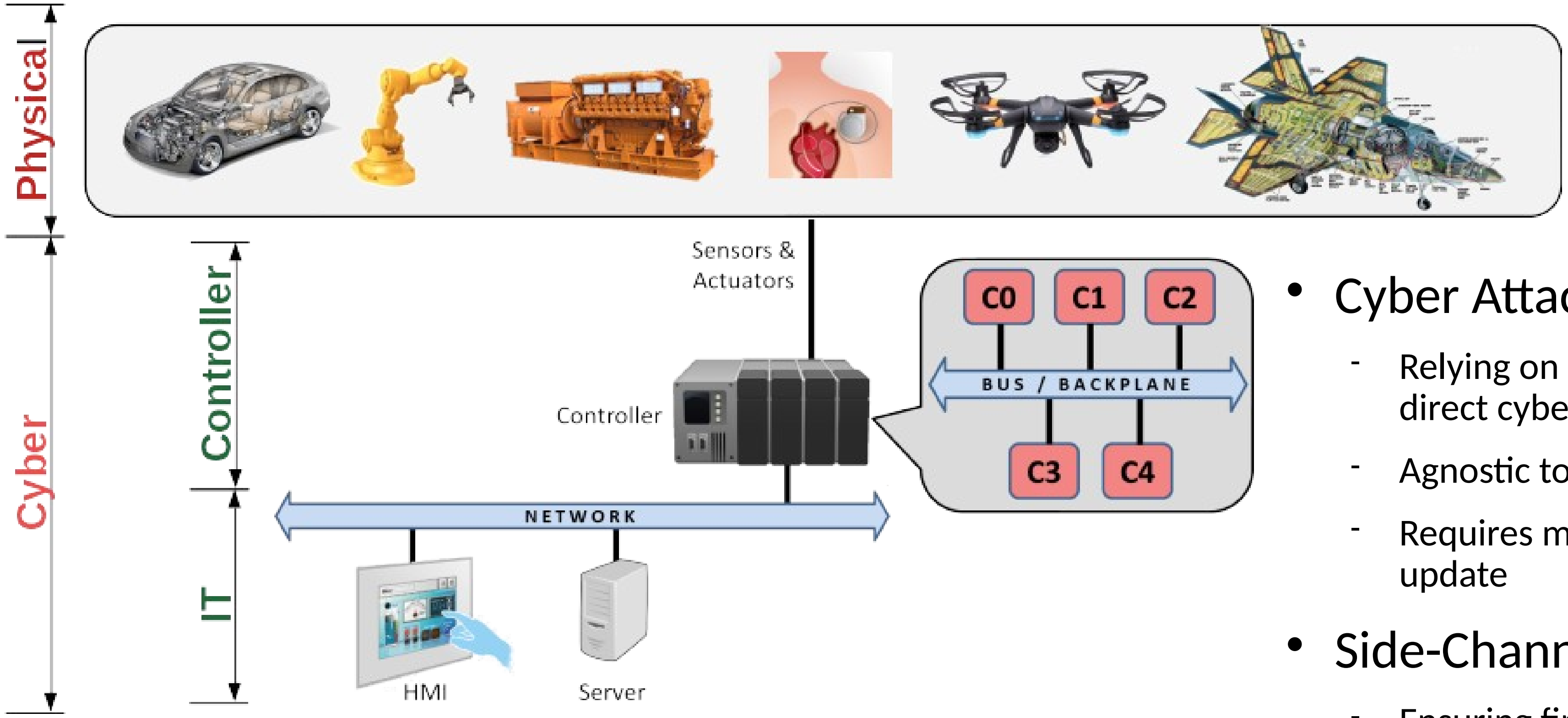
Direct entry into PLC, data source level



Dispelling conventional wisdom about the security of network segmentation & Highlighting vulnerability from third-party connections to the network

https://www.darkreading.com/ics-ot/ot-network-security-myths-busted-in-a-pair-of-hacks 02/2023

Device Level Security: Robustness from the Ground Up



- Effect of Compromised Device:
  - Lie to monitors – doing one thing, reporting another (e.g. Stuxnet)
  - Transport layer (communication) security **irrelevan**t – protecting the attacker

- Cyber Attack Resilience
  - Relying on CPS (controller) properties to tolerate direct cyber attack
  - Agnostic to the specificity of the attack (malware)
  - Requires multi-factor authentication for firmware update
- Side-Channels Monitoring
  - Ensuring firmware/software performing as expected
  - Cannot easily be circumvented by attacker (malware)

Building Resilience System from Resilient Components

Cyber Security Triad – CIA

- **Confidentiality**
  - protection of information from unauthorized access.
  - CPS: no-information leaks
  - Common techniques: Encryption
- **Integrity**
  - information is kept accurate and consistent unless authorized changes are made
  - CPS: provides correct and proper operation/service (as expected)
  - Common technique: Authentication, Hash/integrity checking
- **Availability**
  - information is available when and where it is rightly needed
  - CPS: Service availability
  - Common technique: Robust & Resilience operation

The Importance of C, I & A can be evaluated from the type of data/information, physical dynamics and needs/requirements