Security & Assured Autonomy

Howie Shrobe

MIT CSAIL

Assured Autonomy

The expectation that a system entrusted to make decisions on its own, will with high probability:

- Make good enough decisions to achieve (most of) its goals without harm
- Be able to justify those decision

- A secure system is one that will behave as designed and implemented even when under attack
  - Normally thought of as Confidentiality, Integrity and Availability
  - For Cyber-Physical systems must also include predictable timing

- Security is necessary condition for Assured Autonomy

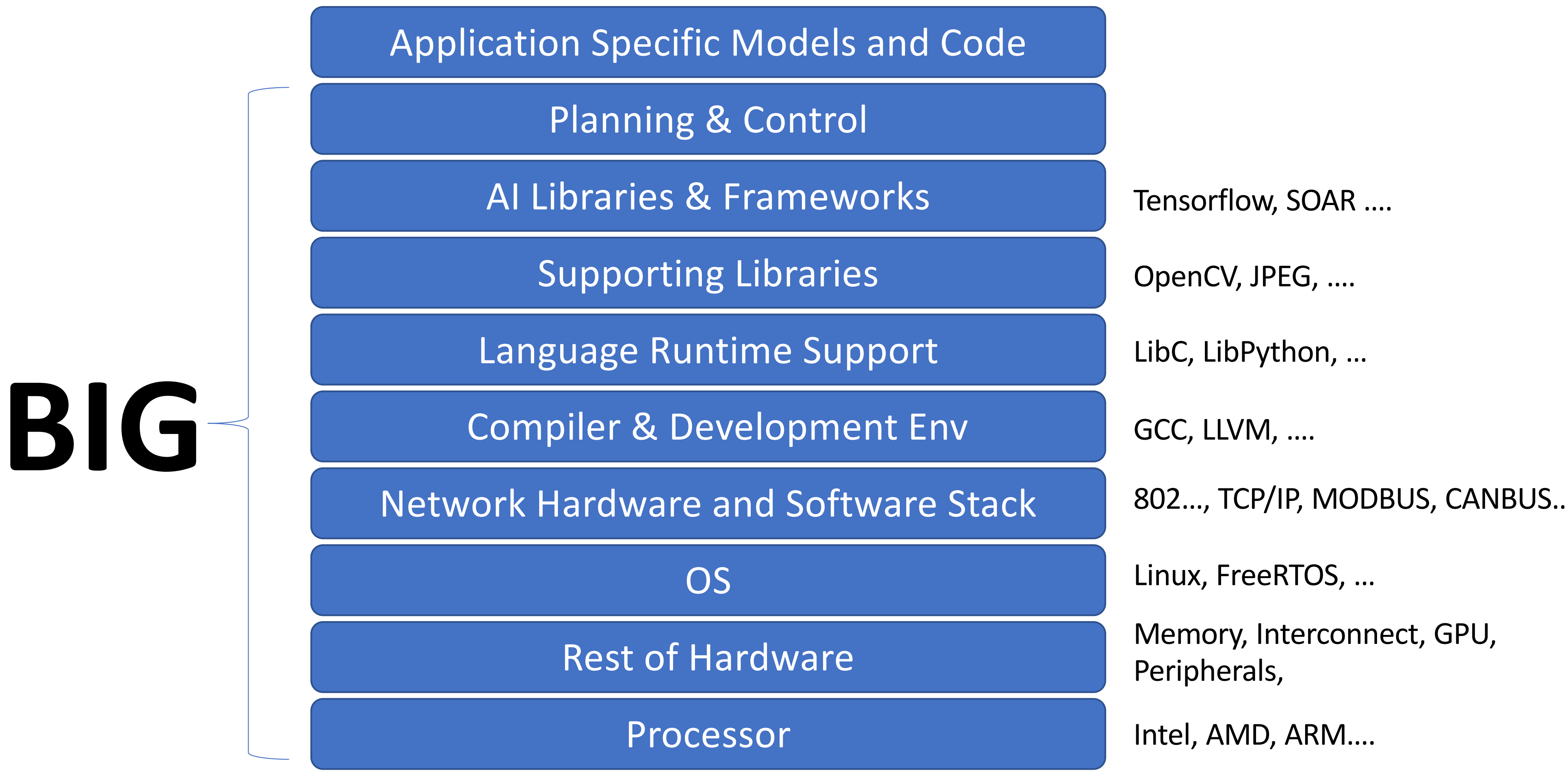
Cars are (unsafe) rolling computers

Full Stack Security is critical

- Vulnerabilities are exploited at many levels:
  - Processor
  - Software
  - Interconnect

- Interconnecting components can amplify the problem:
  - The CANBUS is insecure
  - It was connected to the entertainment/control panel systems for convenience
  - Vulnerabilities in the entertainment system now are vulnerabilities in the control system

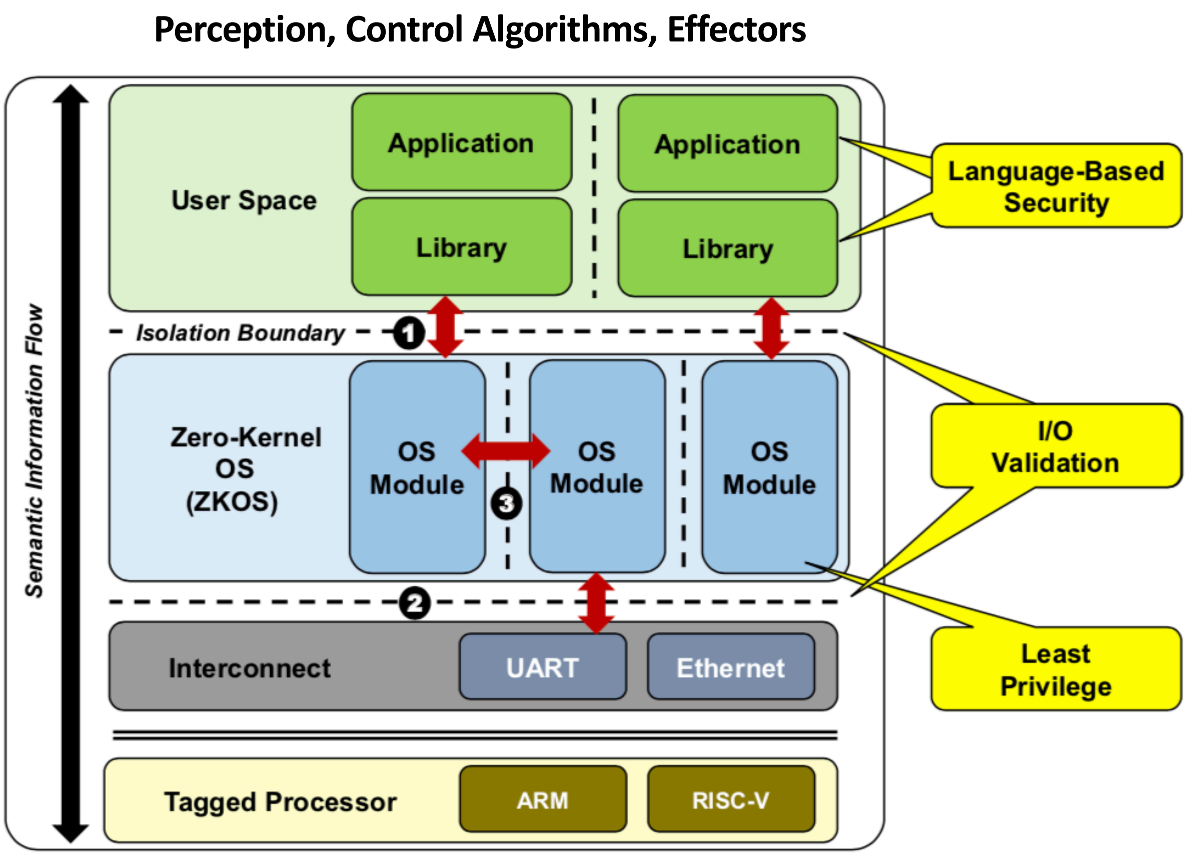
An Autonomous System Stack



Tensorflow vulnerability list

Advisory Number	Type	Versions affected	Reported by	Additional Information
TFSA-2019-001	Null Pointer Dereference Error in Decoding GIF Files	<= 1.12	Baidu Security Lab	
TFSA-2018-006	Crafted Configuration File results in Invalid Memory Access	<= 1.7	Blade Team of Tencent	
TFSA-2018-005	Old Snappy Library Usage Resulting in Memcpy Parameter Overlap	<= 1.7	Blade Team of Tencent	
TFSA-2018-004	Checkpoint Meta File Out-of-Bounds Read	<= 1.7	Blade Team of Tencent	
TFSA-2018-003	TensorFlow Lite TOCO FlatBuffer Parsing Vulnerability	<= 1.7	Blade Team of Tencent	
TFSA-2018-002	GIF File Parsing Null Pointer Dereference Error	<= 1.5	Blade Team of Tencent	
TFSA-2018-001	BMP File Parser Out-of-bounds Read	<= 1.6	Blade Team of Tencent	
-	Out Of Bounds Read	<= 1.4	Blade Team of Tencent	issue report

Full Stack Security (CSAIL, Draper, MIT-LL)

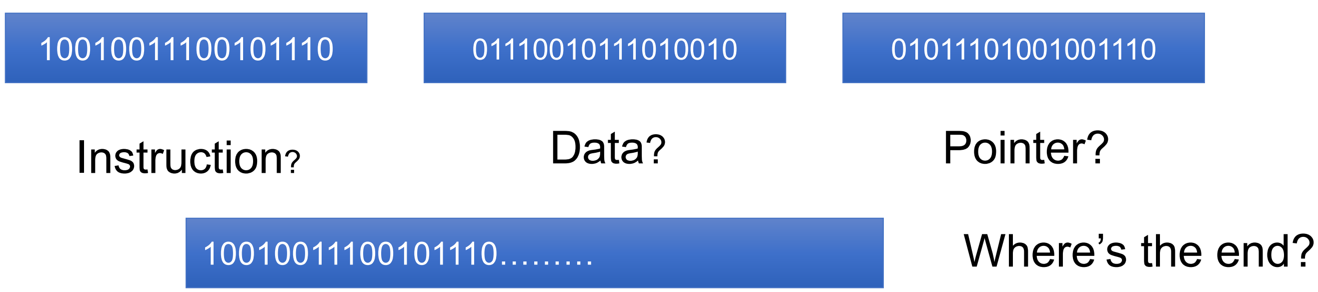


Memory and Type Safe:
E.g. RUST, …

Hardware enforced compartmentalization into minimal sized components

The Underlying Problem:

- Computer hardware and many language runtimes don’t represent or properly manage important semantic distinctions
- At runtime, there’s nothing but “Raw Seething Bits”



The solution

- Tag all data with meta-data
- State policies in the form of rules referencing the meta-data
- Enforce the policies at most efficient & lowest level possible to guarantee complete mediation
- Example 1: Memory Safety
  - All pointers includes bounds information
  - Attempts to reference outside the bounds are trapped
- Example 2: Type safety
  - All objects are typed
  - Operations can only be performed on relevant object types (e.g. you can't execute a string)

Summary

- If we want to have any confidence in an autonomous system then its necessary (but not sufficient) to guarantee that vulnerabilities cannot be exploited to change the reasoning of the autonomy software
  - All data must carry meta-data
  - Policies related to the meta-data must be systematically enforced
  - Enforcement must be done across all levels
  - A “belt and suspenders” approach is necessary
  - Prevention + Containment