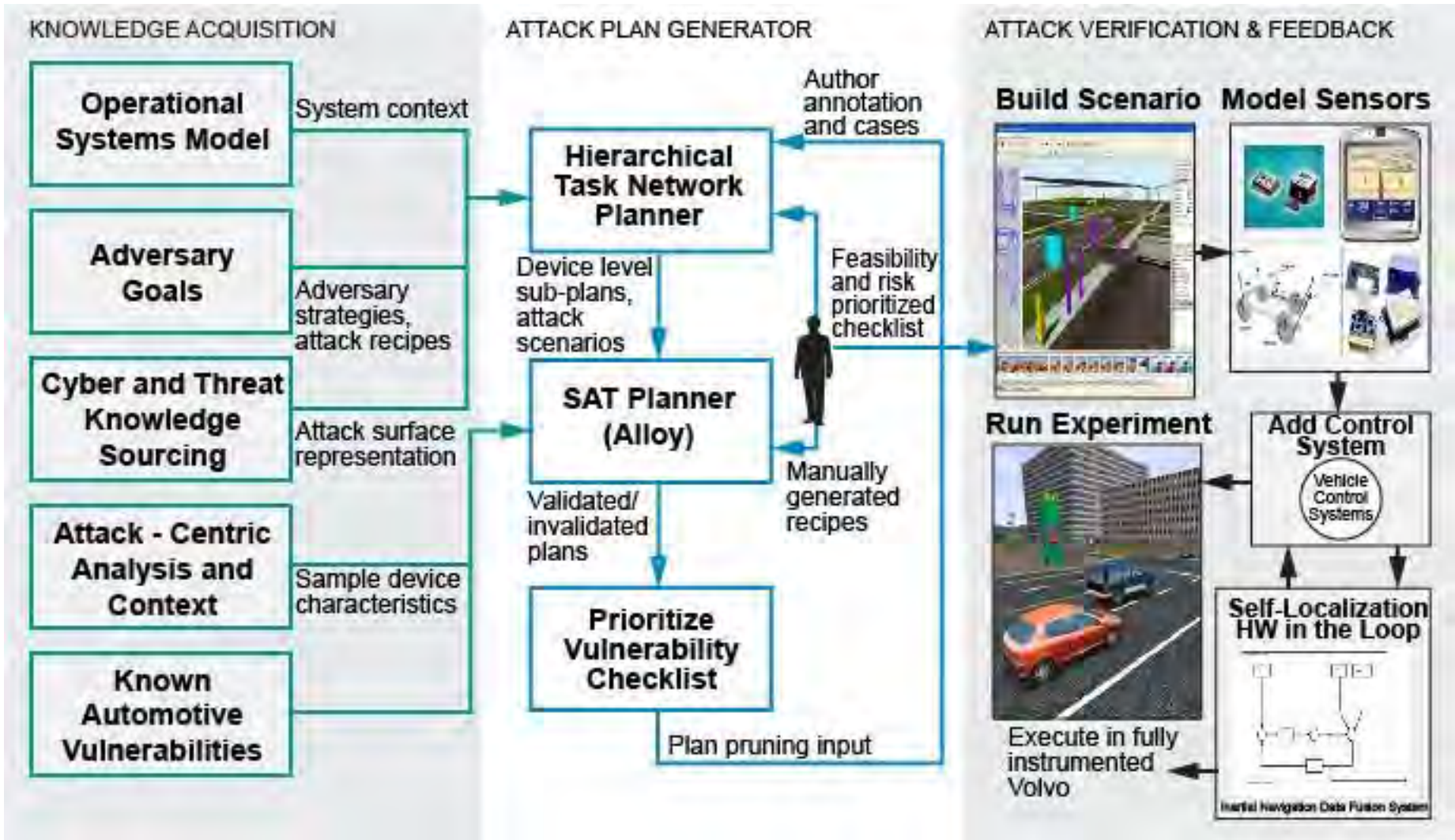


# Applied Vulnerability Detection System

J. Smith, B. Krikeles, D. Wittenberg and M. Taveniku

April 14, 2015

# Generate, verify, and prune to manage and prioritize possible attack point vulnerabilities



# Data from multiple external sources seeds a goal-oriented attack plan generation process

## KNOWLEDGE ACQUISITION

Characterizing systems based on their exposed services versus their marketed functions.

**Operational Systems Model**

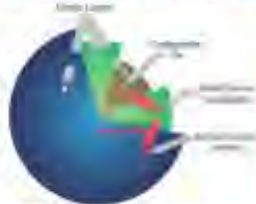
System context

How services and vulnerabilities relate to attack objectives.

**Adversary Goals**

Adversary strategies, attack recipes

Externally available services reside on Device Layer 0.



Transforming existing data into attack-centric representations.

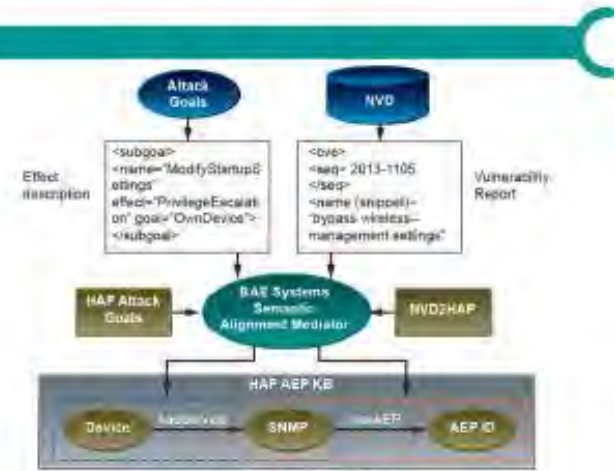
**Cyber and Threat Knowledge Sourcing**

Attack surface representation

- ID and auto-capture data sources
- Use a BAE Systems mediation solution to process data source records and attach new semantic "Weakness" and "Effect" descriptions
- Transform data sources into a standardized Attack and Exposure Point (AEP) record based on a new Attack Surface Ontology (ASO)

**Attack - Centric Analysis and Context**

- Construct a new AEP device view that aligns with how an adversary seeks to expose the device's attack surface
- Unify disparate device scanning techniques into an automated framework that populates a common Device Characterization Ontology for analysis by the planner
- Infer adversary goals from data collected on the attacks that adversaries have already conducted against other systems based on attack patterns derived from the Common Attack Pattern Enumeration and Classification (CAPEC)



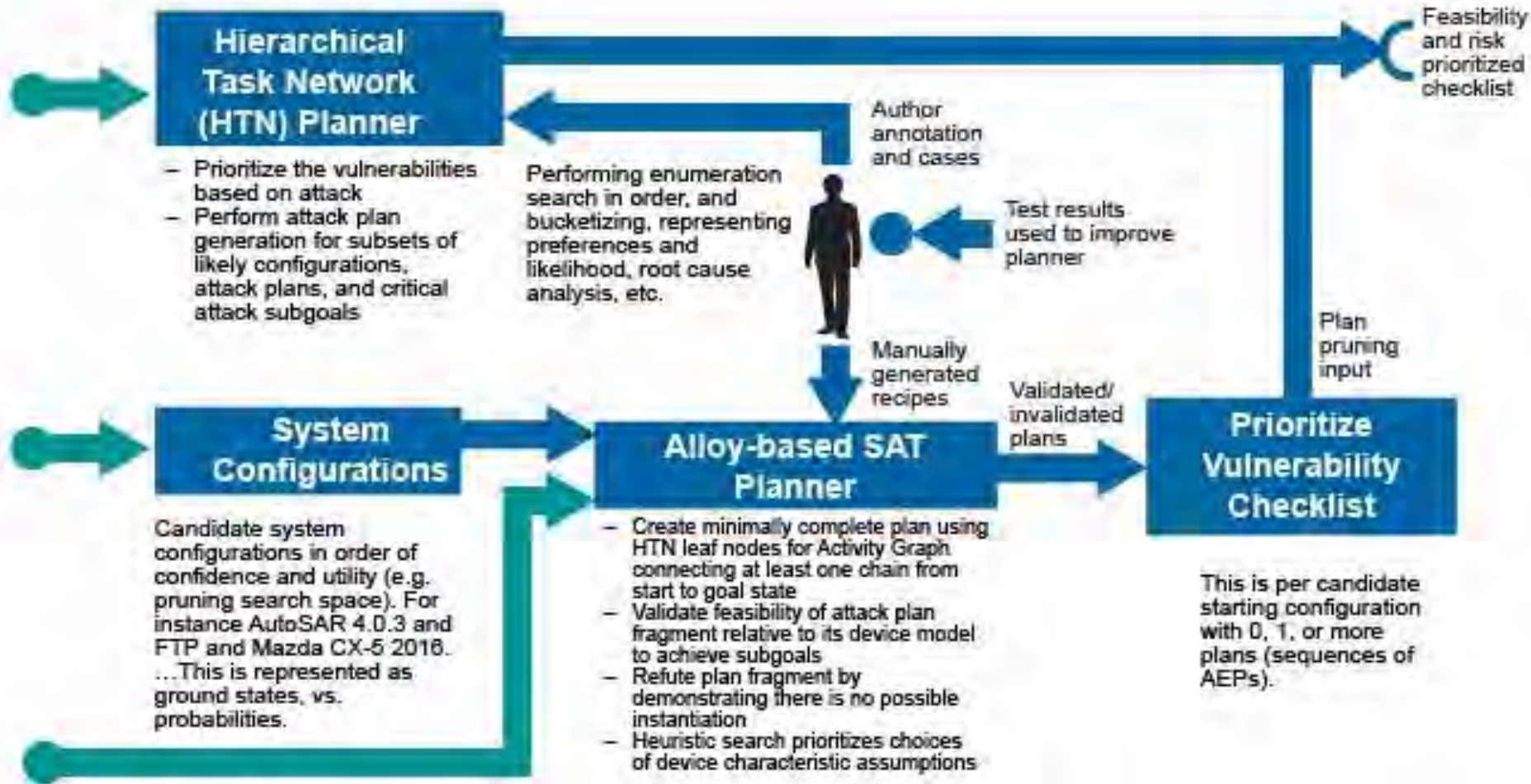
Unify attack subgoals to AEP "effects" by making declarative mappings to our ASO

**Known Automotive Vulnerabilities**

Includes services that an attacker can misuse.

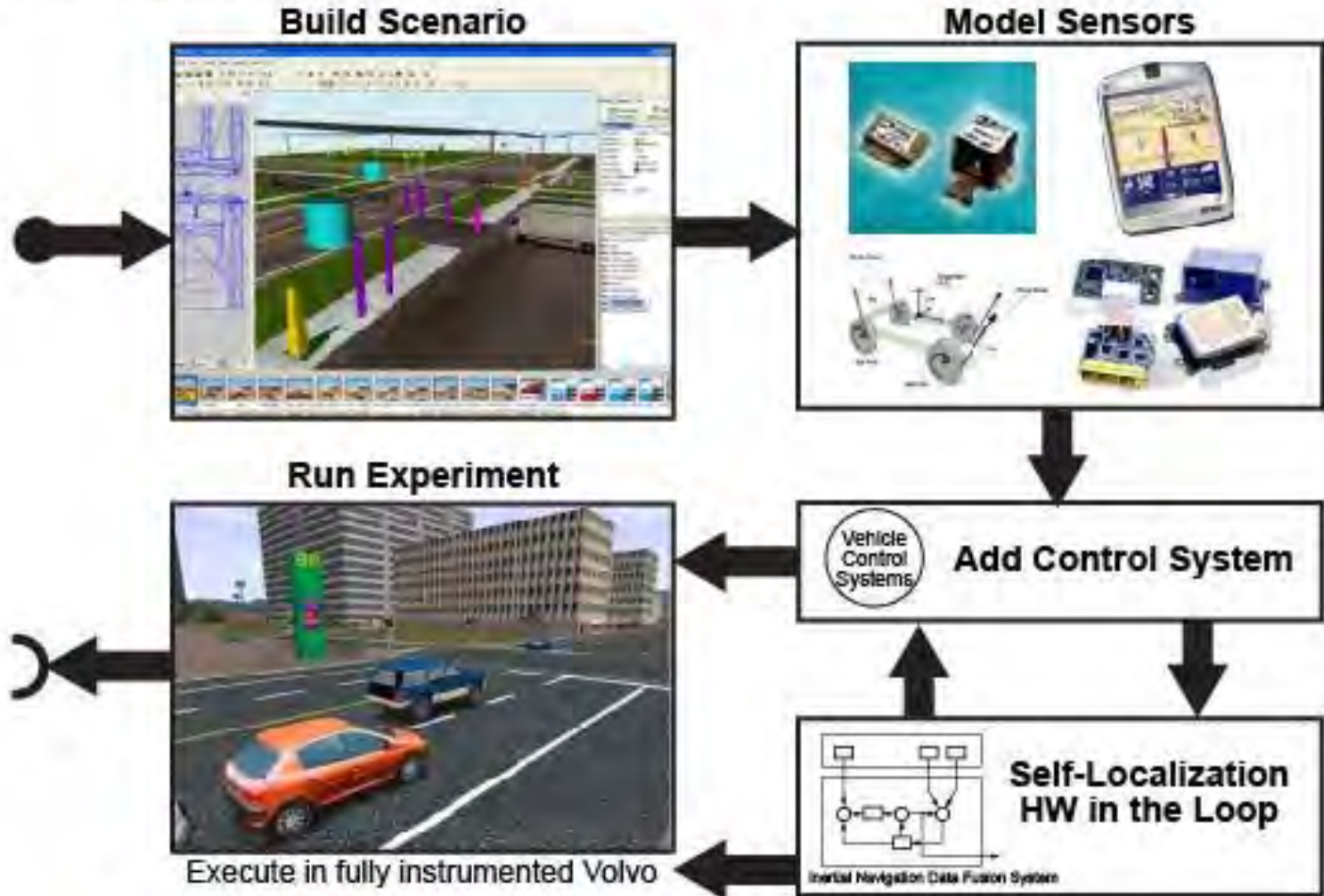
# The identified attack space is reduced by using a vulnerability-based feasibility analysis

## ATTACK PLAN GENERATOR

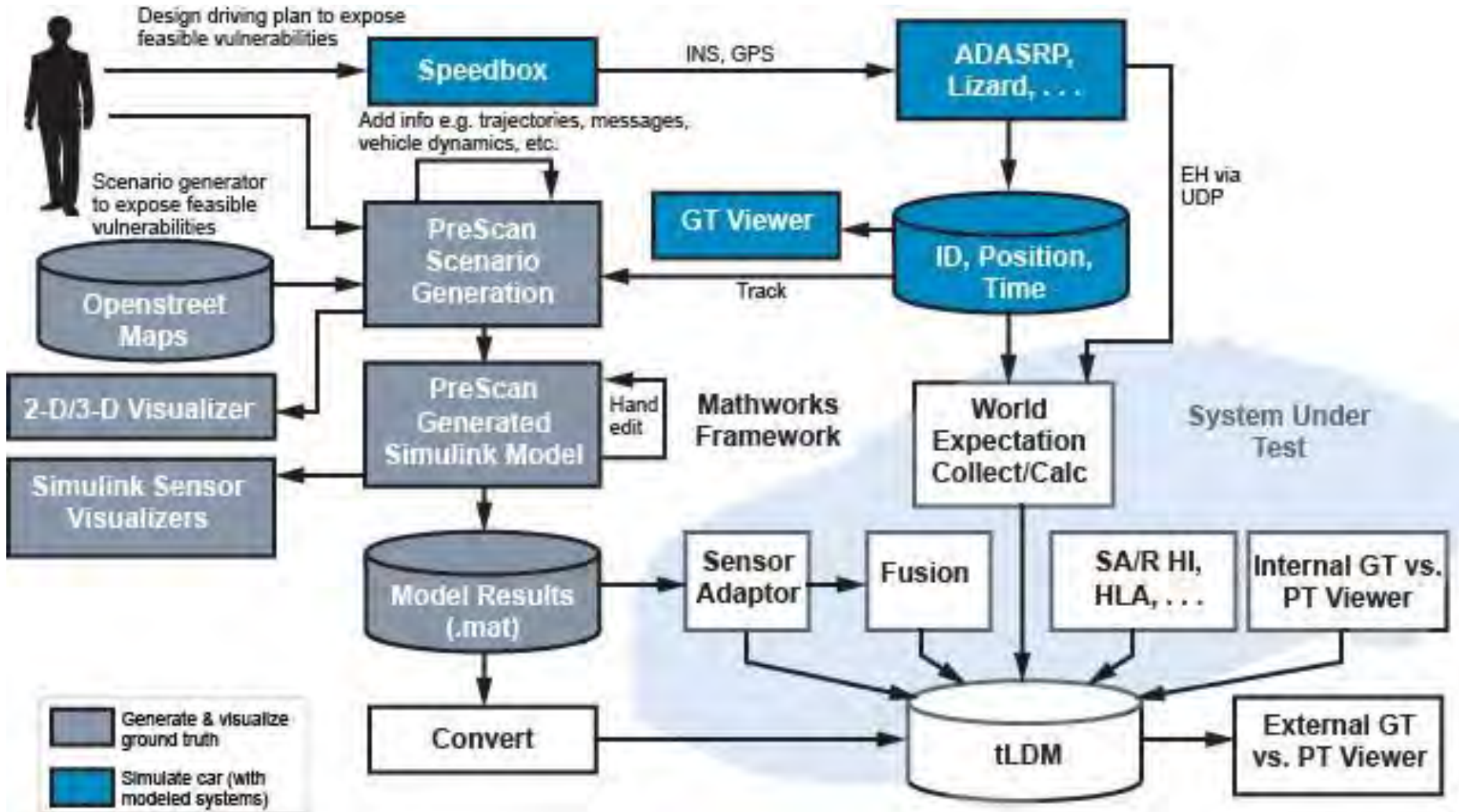


# PreScan simulations verify attack plans and identify exhibiting an emergent system-level vulnerability

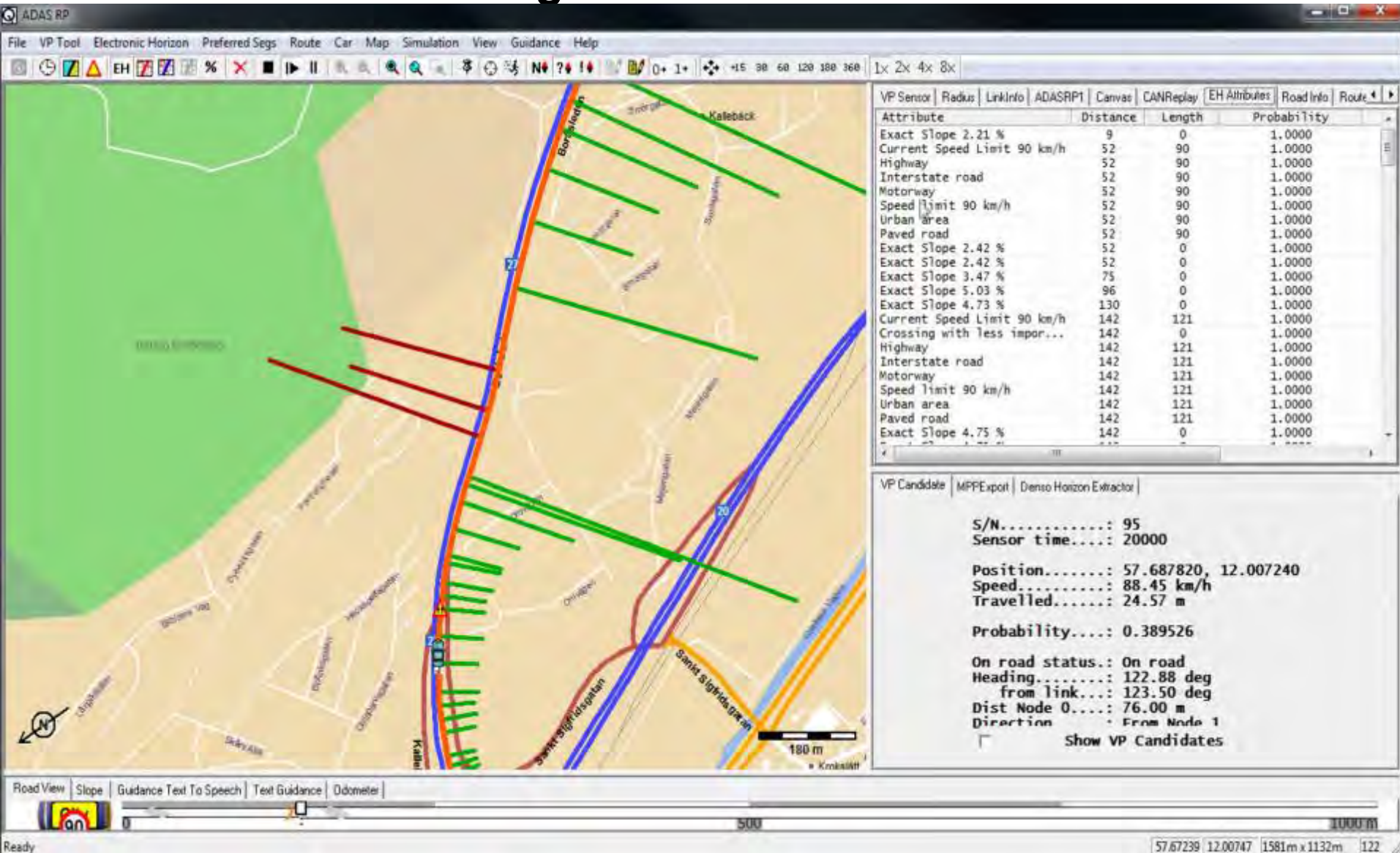
ATTACK VERIFICATION AND FEEDBACK



# Ground truth is used to calibrate PreScan simulations



# Generate and visualize ground truth, verify simulation scenario against actual



# Model car sensors and additional obstacles for vulnerability analysis

The screenshot displays the PreScan Experiment Editor interface. The main workspace shows a 3D model of a car with sensor beams (green and red) extending from its front and rear. A top-down view of the car is also visible. The interface includes a 'Positioning' panel with the following data:

Position	Y (m)	Z (m)
0.817673	0.400000	1.300000

Orientation: Roll (deg): 0.0, Pitch (deg): 0.0, Yaw (deg): 0.0

Generic sensor parameters:  
Range (m): 50.0  
Cone angle (deg): 45.0

Detection range:  
Range detection type: All  
Response distance before alarm is sounding: 10

Parent bounding box dimensions:  
Length (m): 4.425 (W)  
Width (m): 1.880 (H)  
Height (m): 1.314 (H)

Vehicle specific dimensions:  
Bumper Front (m): 3.400 (W)  
Bumper Rear (m): 3.000 (W)  
Rear Offset (m): 3.907 (H)

The interface also features a 'Hierarchy' panel on the right, showing the object configuration tree, and a 'Properties' panel at the bottom right for the selected 'AIR\_1' object.

Identification	Value
Name	AIR_1
Description	AIR
Type	AIR
Object specific info	
Beam color	Green
Beam length	50
Cone angle	45
Position	
Location	0.81767315954673, -0.4, 1.300000
Orientation	0, 180, 0

At the bottom of the interface, the 'Positioning' panel shows the following data:

Position	Y (m)	Z (m)
3.519000	0.000000	3.400000

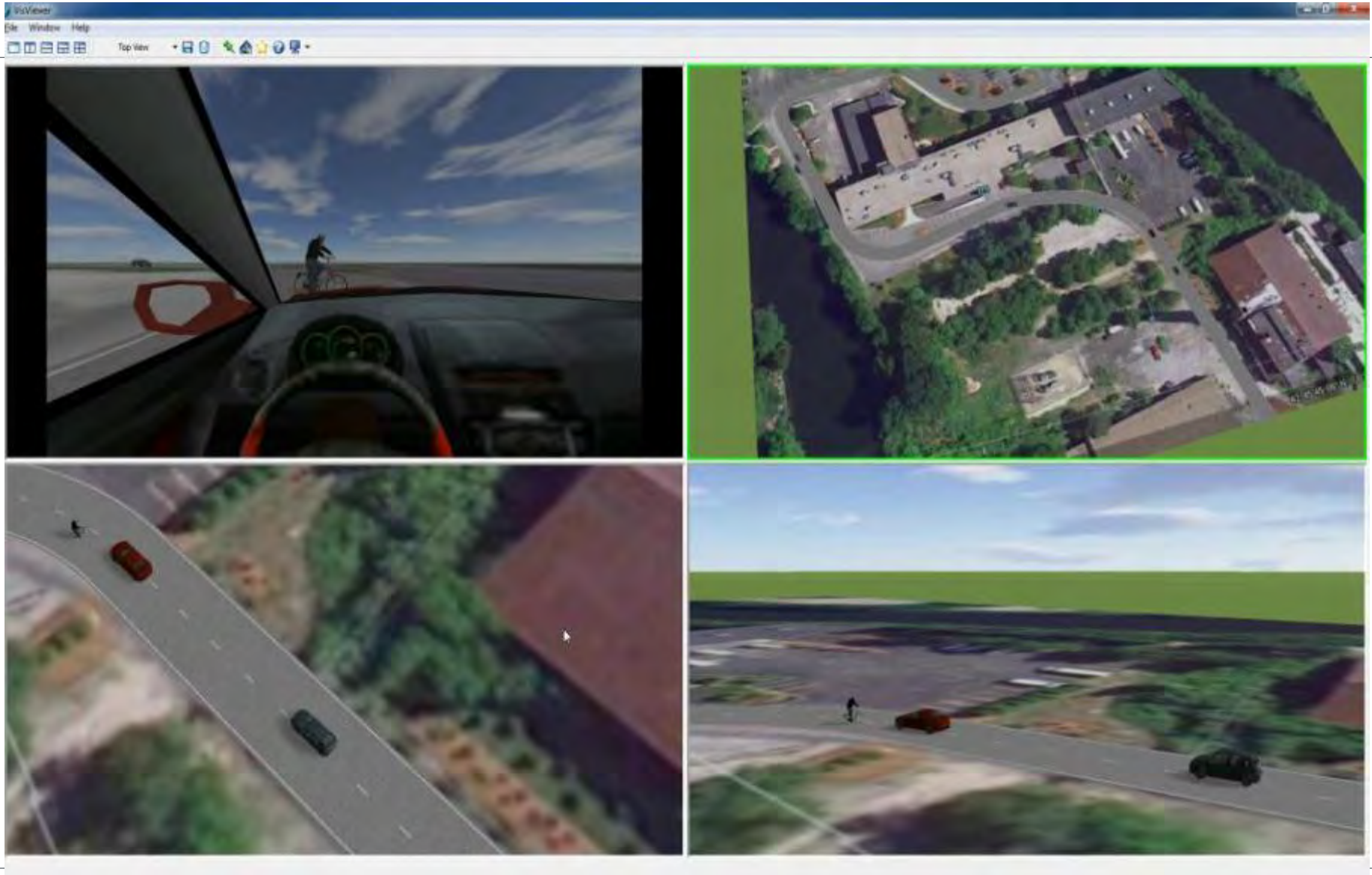
Orientation: Roll (deg): 0.0, Pitch (deg): 0.0, Yaw (deg): 0.0

Parent bounding box dimensions:  
Length (m): 4.425 (W)  
Width (m): 1.880 (H)  
Height (m): 1.314 (H)

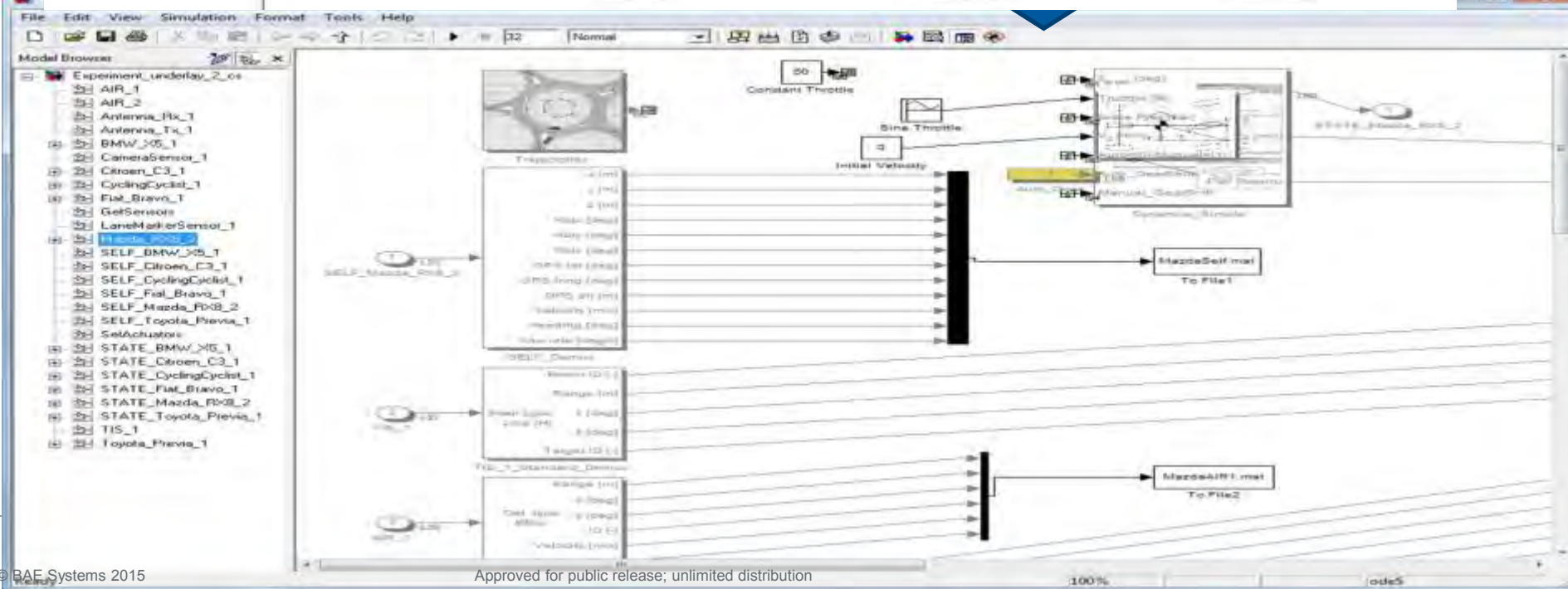
Vehicle specific dimensions:  
Bumper Front (m): 3.400 (W)  
Bumper Rear (m): 3.000 (W)  
Rear Offset (m): 3.907 (H)



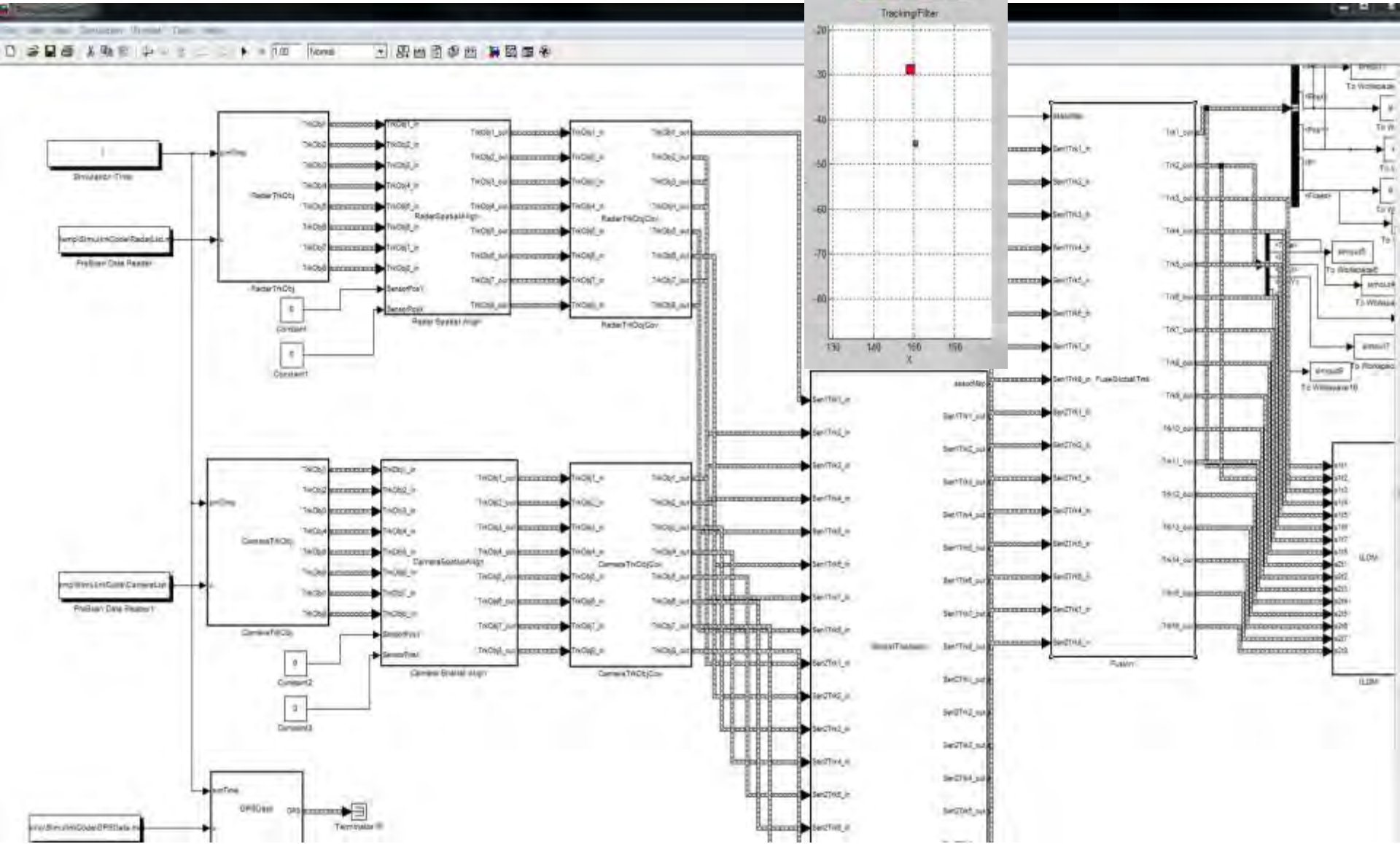
# Run simulation to visualize and generate Simulink



# Edit Simulink adding fault injection and dynamic model changes



# Run Simulink and analyze system under test



## Summary: The system equips users to...

- Automatically detect emergent vulnerabilities in complex CPS
- Approach from the attacker's point of view
- Automatically ingest public and private lists of vulnerabilities and organize them into a standardized Attack Point database
- Generate the set of composite device-level attack scenarios
- Reduce the cardinality of the generated attack space by evaluating the feasibility of each attack against device characteristics and constraints
- Discard unlikely or infeasible attacks and prioritizes the remainder by risk size and likelihood of occurrence
- Use prioritized vulnerability list to confirm or rule out each candidate attack and feeding back to refine planning