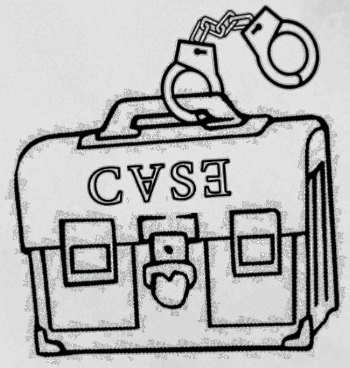

# BRIEFCASE FULL OF PROOFS:
## CYBER ASSURED SYSTEMS ENGINEERING

HIGH CONFIDENCE SOFTWARE & SYSTEMS
6 MAY 2021

DARREN COFER
TRUSTED SYSTEMS
DARREN.COFER@COLLINS.COM





Collins Aerospace

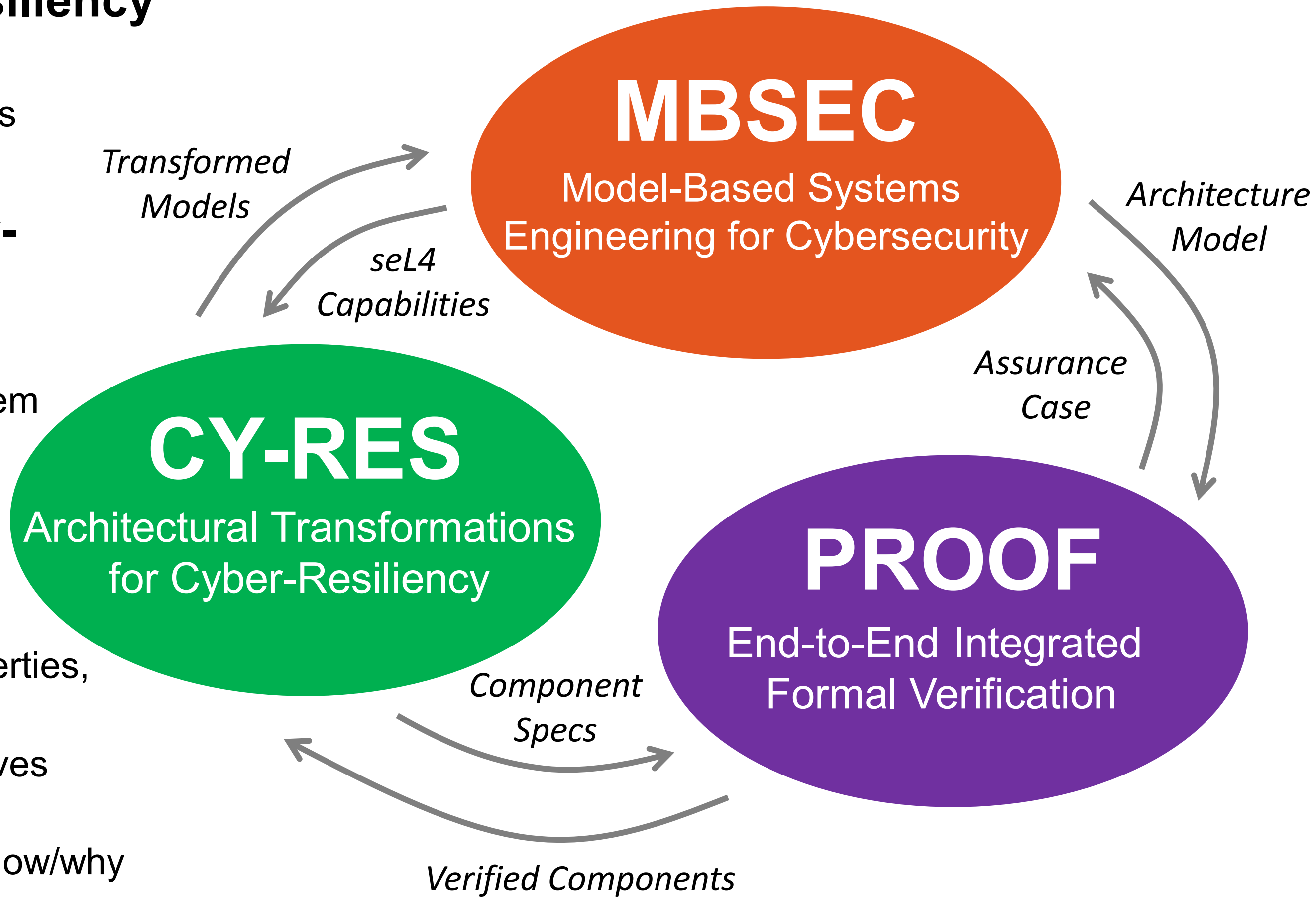
# TEAM

## COLLINS-LED DARPA CASE PROJECT

- Collins Aerospace
  - Architectural transformations for cyber-resilience
  - Component synthesis and proofs
  - Formal analysis and assurance case
  - Tool integration
- Data 61
  - seL4 formally verified secure microkernel for memory protection
  - Formally verified components (seL4, CakeML language)
- University of Kansas
  - Formally verified attestation for distributed computing platforms
- Adventium
  - Real-time scheduling
  - AADL modeling
- Kansas State University
  - Automatic code generation from architecture models with proof of equivalence
  - Information flow analysis

# 3 TECHNOLOGY PILLARS

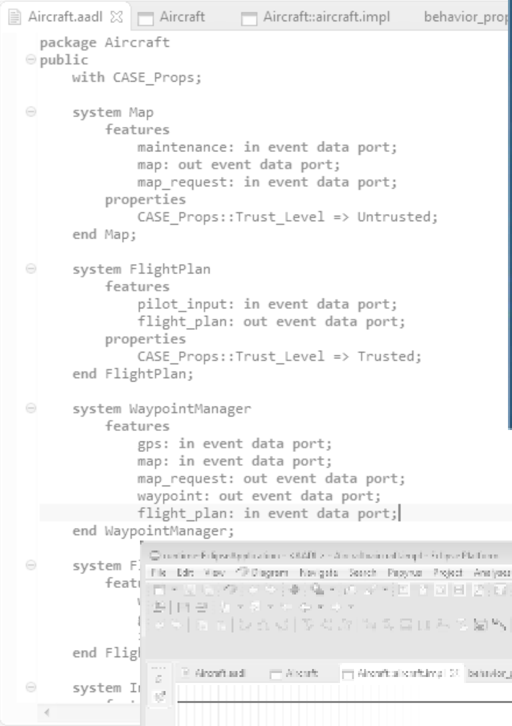
1. **Developer assistance to implement cyber-resiliency**
   - Automated architecture transforms for threat mitigation
   - High assurance components generated from specifications
   - Techniques to deal with legacy code (cyber retrofit)
2. **MBSE environment for high-assurance cyber-resilient system development**
   - Build system directly from detailed, verified AADL model
   - Makes the security guarantees of seL4 accessible to system developers
   - Ability to target different platforms to facilitate incremental debugging/development
3. **Integration of formal verification/proof**
   - Formal verification of functional and cyber-resiliency properties, information flow properties, component proofs
   - Code generation equivalence to model, seL4 build preserves properties
   - Integrate evidence as an assurance case demonstrating how/why requirements are satisfied

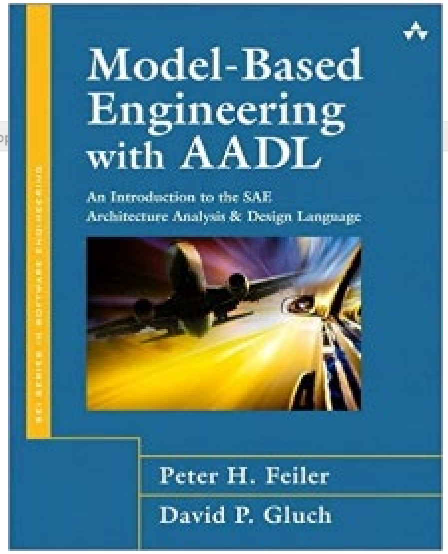



Collins Aerospace

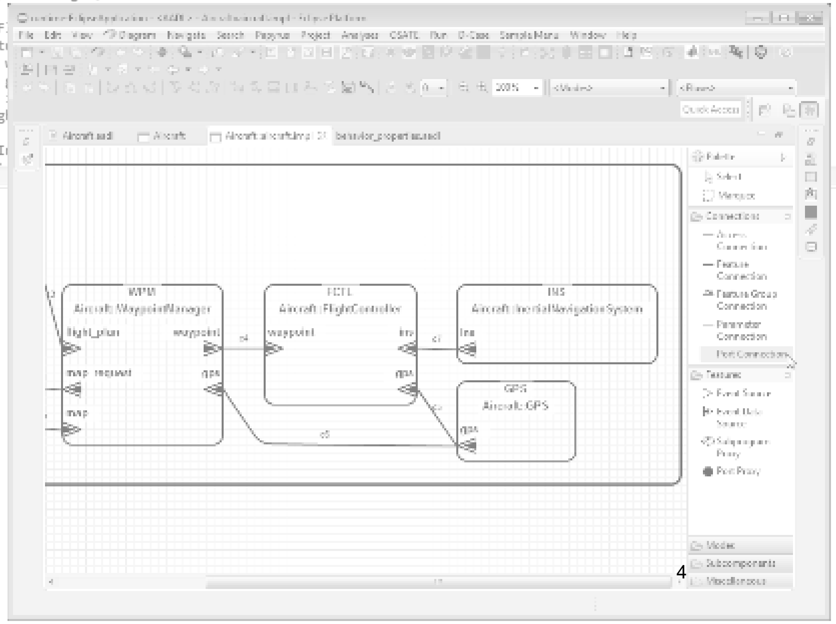
# BRIEFCASE TOOL CAPABILITIES

- Integrated **model-based systems engineering** tool suite based on Architecture Analysis & Design Language (AADL) models
- Transform system design to satisfy **cyber-resiliency** requirements
- Generate new **high-assurance components** from formal specifications
- Verify system design using **formal methods** and document evidence/compliance with assurance case
- Generate **software integration code** directly from verified architecture models, targeting multiple operating systems (including seL4)

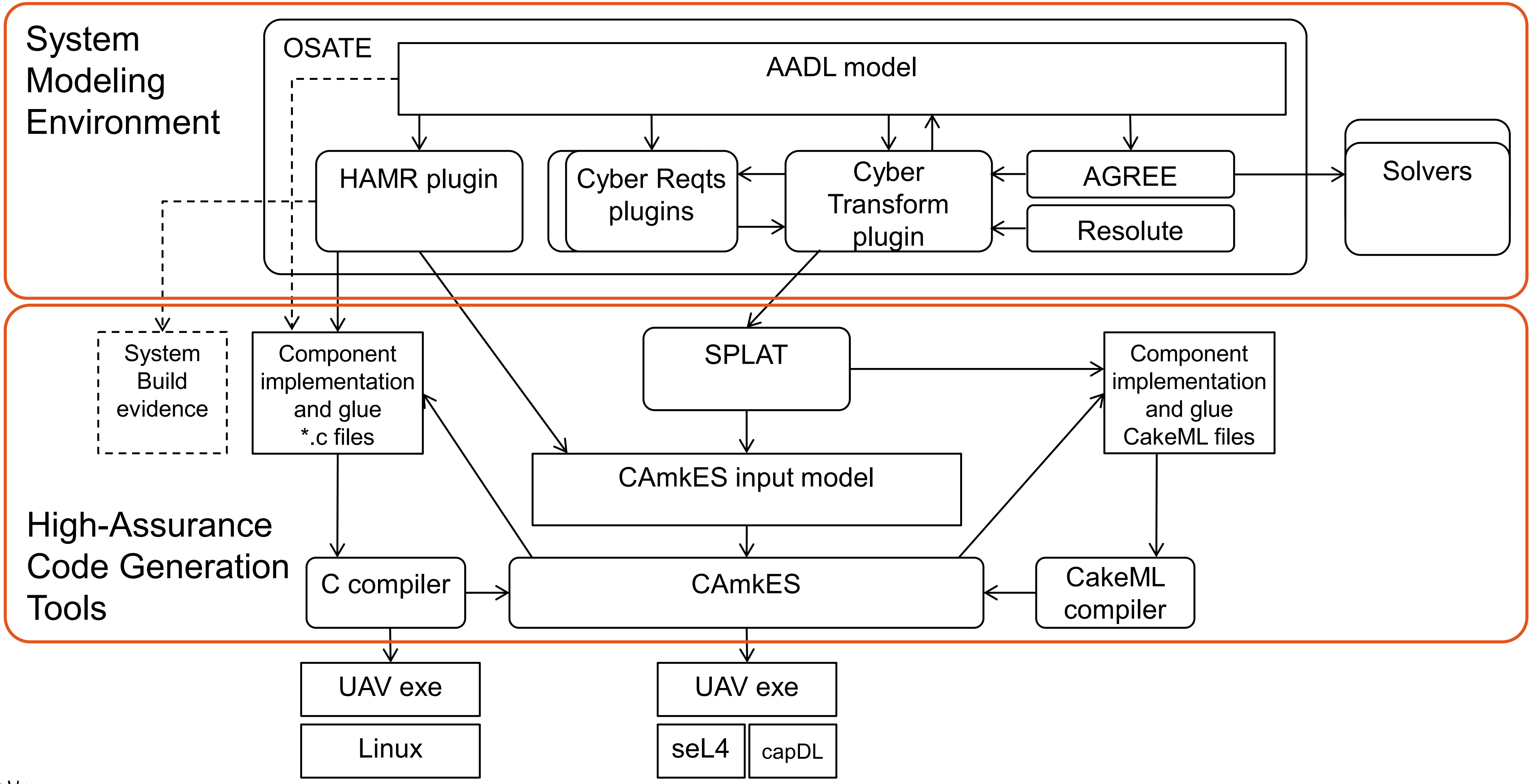



SAE AS5506 STANDARD

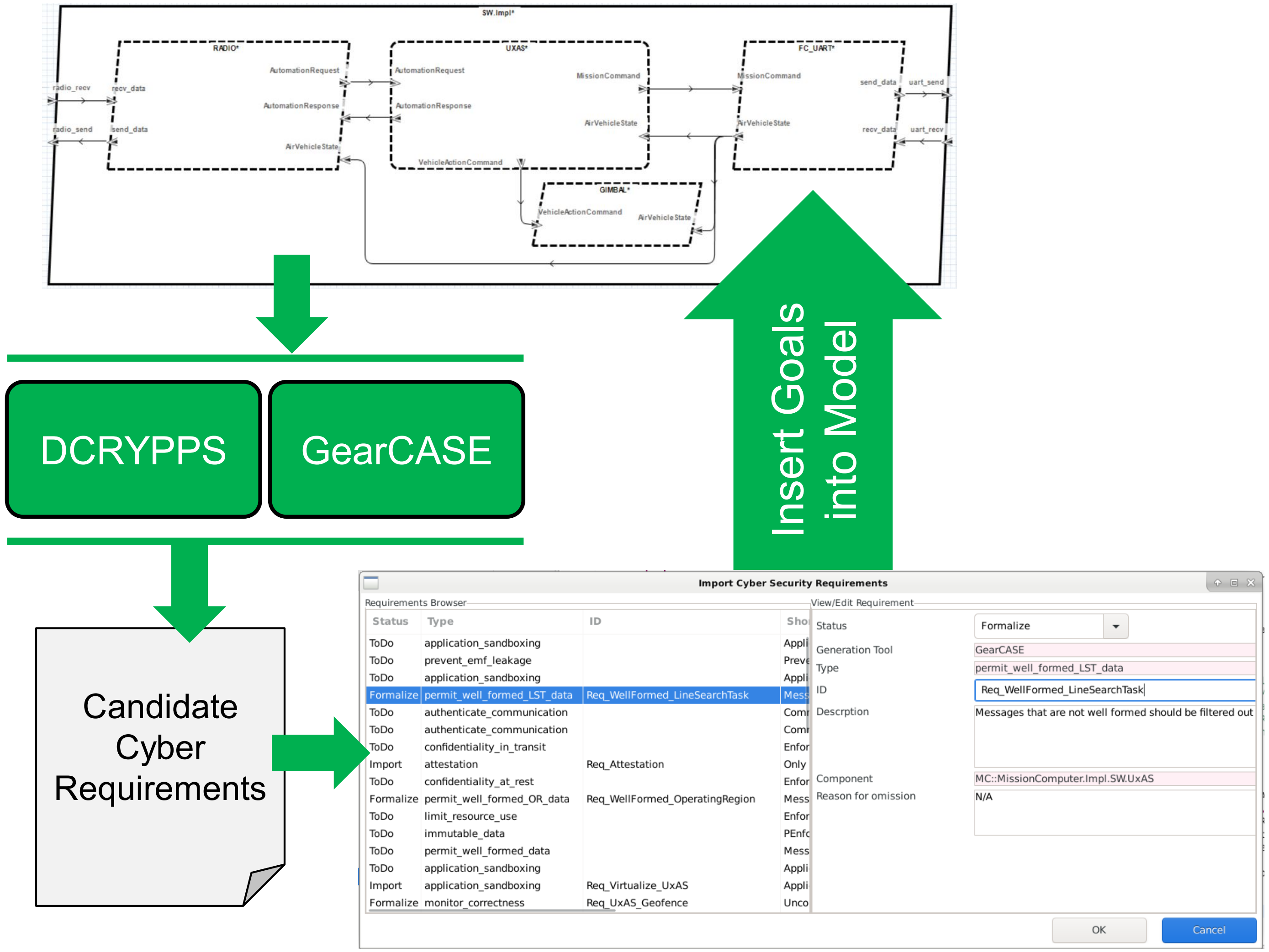


# BRIEFCASE TOOL ARCHITECTURE



# 1. GENERATE / IMPORT CYBER REQUIREMENTS

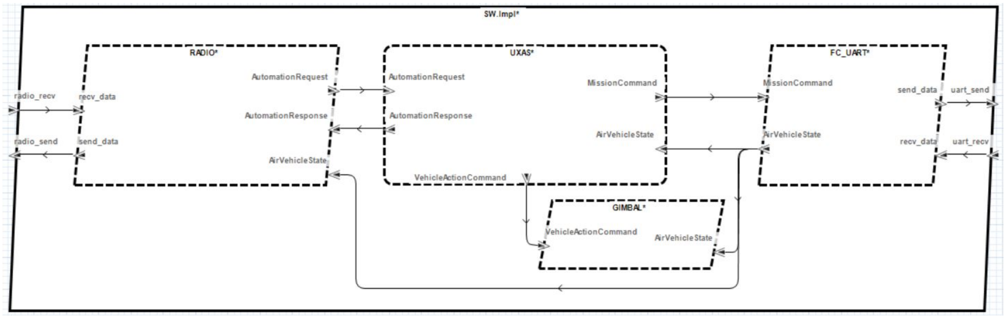
- Choose one of the Cyber Requirements generation tools
  - CRA GearCASE plugin
  - Vanderbilt/DOLL DCRYPPS plugin
- Initial model data is exported to selected tool
- Requirements import wizard manages the generated requirements
  - Select action
  - Naming/tagging
  - Associate with formal properties
- Requirements inserted into model as Resolute goals (GSN)
  - We will build an assurance argument to satisfy these goals

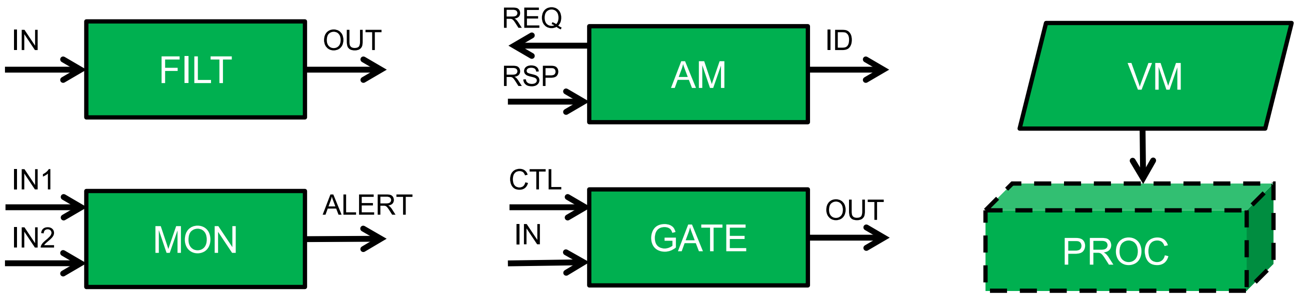


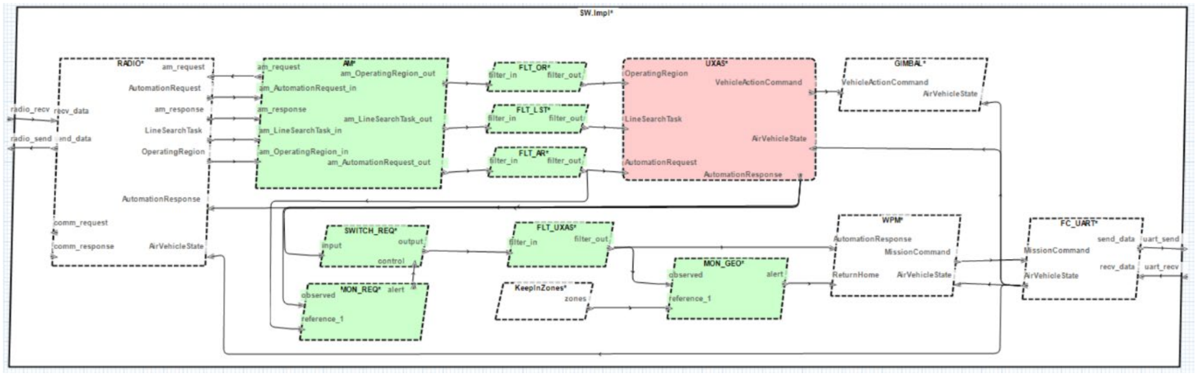
# 2. APPLY CYBER TRANSFORMATIONS

- Cyber requirements tools provide model context and sometimes suggested mitigation
- System engineer selects from available cyber-resiliency transformations
  - Filter
  - Monitor
  - Gate (controlled by  monitor)
  - Attestation
  - Virtualization
  - seL4 build prep
- Wizard interface collects needed configuration data
- Tool automatically transforms AADL model

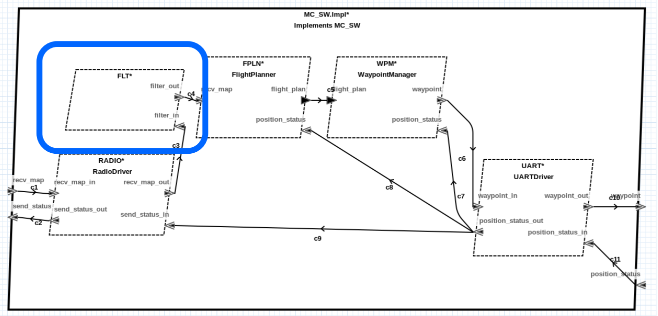
- Also adds Resolute assurance case strategy to show how the associated goal (requirement) is satisfied



# 2A. INSERT ASSURANCE CASE STRATEGY




- Resolute links cyber transform to goal as a GSN strategy
- Checks for violations/changes that impact correctness
- Collects evidence and generates assurance case

```
thread Filter
    features
        filter_in: in event data port
        filter_out: out event data po
    properties
        CASE::COMP_TYPE => FILTER;
        CASE::COMP_IMPL => "CakeML";
        CASE::COMP_SPEC => "(\\i{-90,
    annex agree {**
        guarantee "The Flight Planner
```

```
package CASE_Requirements
private

    annex Resolute {**

        goal Req_WellFormed(comp_
            ** "[permit_well_form
            context Generated_By
            context Generated_On
            context Req_Component
            context Formalized :
            agree_property_checke
```

```
annex Resolute {**

    ---------------------------
    -- MODEL TRANSFORMATIONS --
    ---------------------------

    -- Top-level claim for proper insertion of a filter
    goal add_filter(comp_context : component, filter : component, conn : connection, msg_type : data) <=
        ** "Filter " filter " is properly added to component " comp_context **
        filter_exists(filter, comp_context, conn) and component_not_bypassed(filter, comp_context, msg_type) and component_implemented(filter)

    -- Top-level claim for proper insertion of attestation manager
    goal add_attestation_manager(comm_driver : component, attestation_manager : component, attestation_gate : component) <=
        ** "Attestation Manager added for communications driver " comm_driver **
        attestation_manager_exists(comm_driver, attestation_manager) and attestation_manager_not_bypassed(comm_driver, attestation_manager, attes
```

Problems | Properties | AADL Property Values | AGREE Results | Console | Progress | Assurance Case

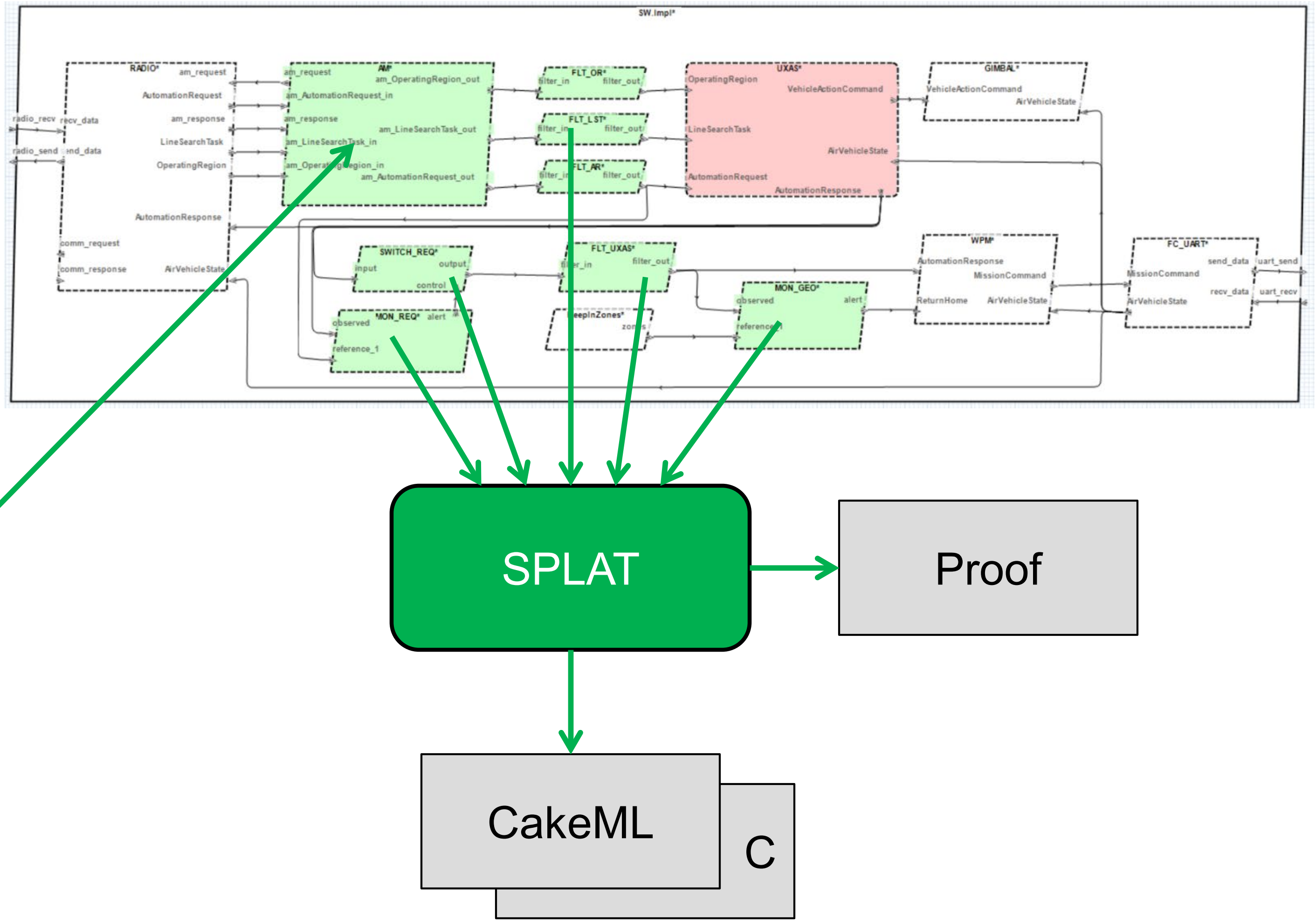
- well_formed(FPLN : SW::FlightPlanner, "good_gs_command")
  - FPLN : SW::FlightPlanner only receives well-formed messages
    - A filter exists on the communication pathway immediately before FPLN : SW::FlightPlanner
    - Filter cannot be bypassed
    - Filter property implemented by CakeML
    - AGREE property passed: [ good_gs_command ]

Collins Aerospace

# 3. GENERATE HIGH ASSURANCE COMPONENTS

- Some of the cyber transforms insert new high-assurance components into the model
- The behavior of the component (its contract) is specified in AGREE
- SPLAT generates component implementations from their specifications
- SPLAT also generates a proof showing that the component implements its specification

- Other components (such as the Attestation Manager) are pre-built pre-verified libraries
- Their implementations are essentially library functions that are added to the build, possibly with some configuration data from the model



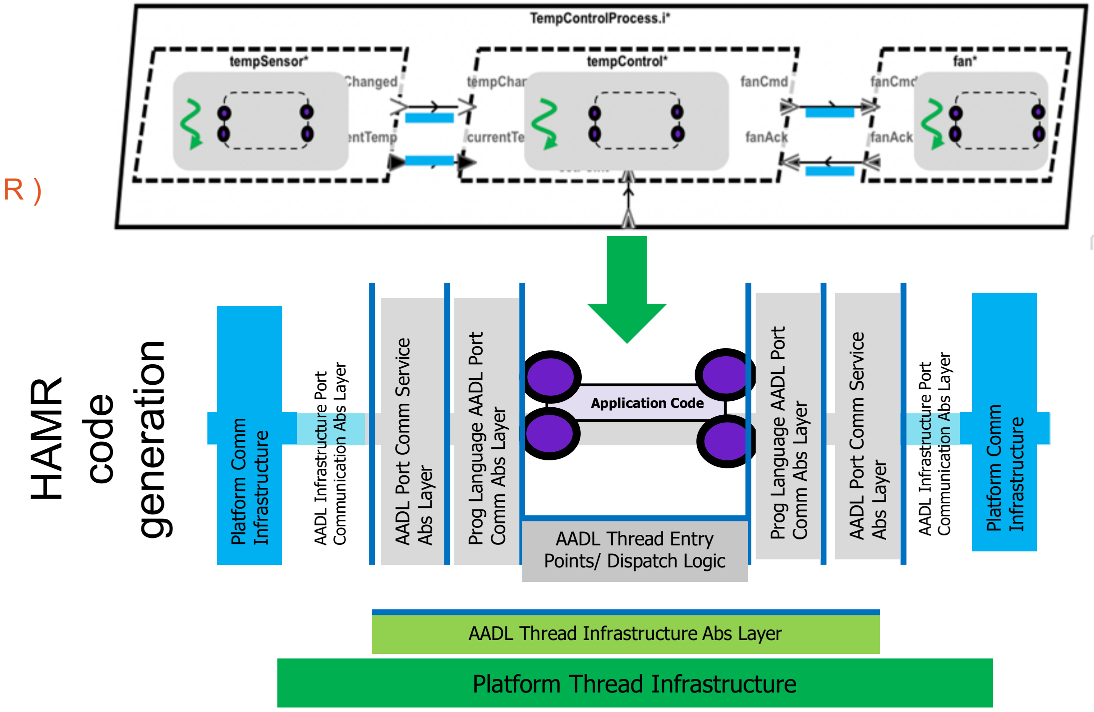

Collins Aerospace

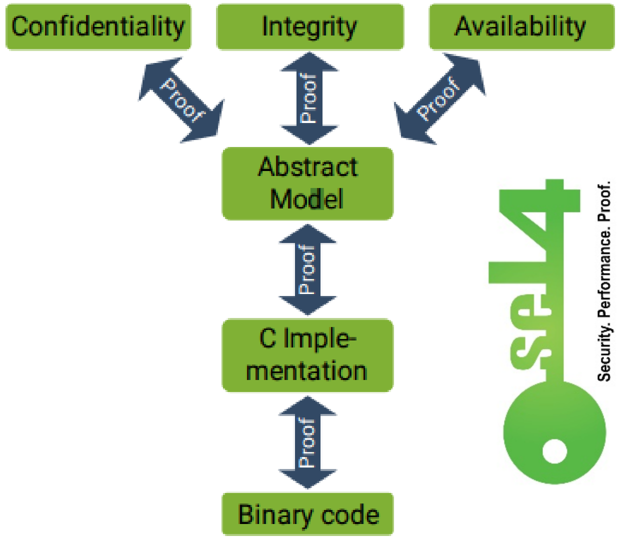
# 4. SOFTWARE INFRASTRUCTURE

## HIGH ASSURANCE MODEL-BASED RAPID ENGINEERING (HAMR)

- Multi-stage translation architecture to address CASE goals of component migration between platforms and information flow control
- Semantic consistency from model to execution
- Ensures model-level analysis applies to deployed code
- Same computational model across different platforms
- Build for multiple target platforms:
  - seL4 / Linux / Virtual Machine
  - Build for workstation / emulator / embedded platform
- Correspondence proof of dataflow preservation

- seL4 microkernel guarantees partitioning of components and communication, backed by computer-checked proofs
- seL4 guarantees no infiltration, exfiltration, eavesdropping, interference, and provides fault containment for untrusted code



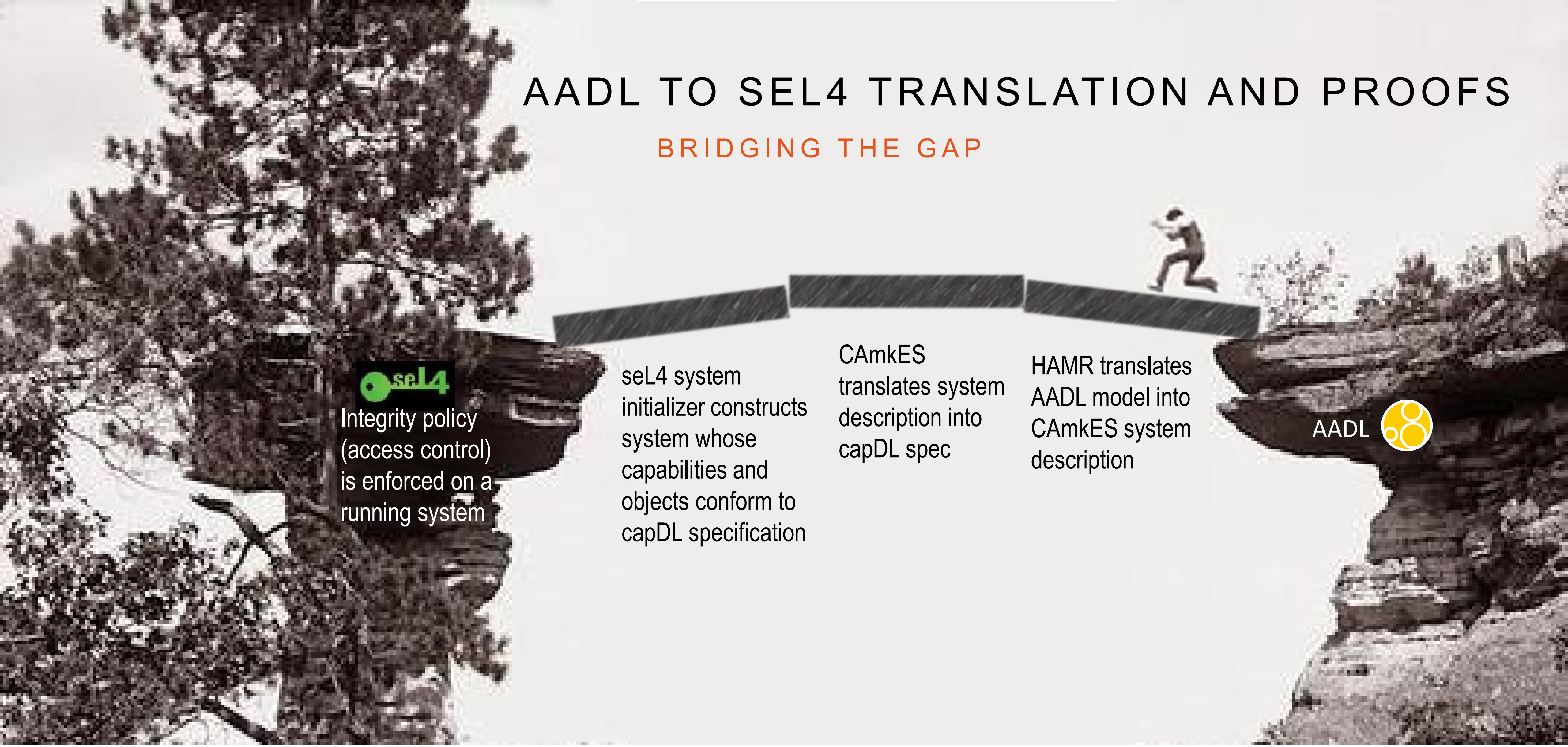

**seL4 is…**

- An operating system microkernel
- A hypervisor
- Proved correct
- Provably secure
- Fast

Collins Aerospace

# AADL TO SEL4 TRANSLATION AND PROOFS

## BRIDGING THE GAP



seL4

Integrity policy (access control) is enforced on a running system

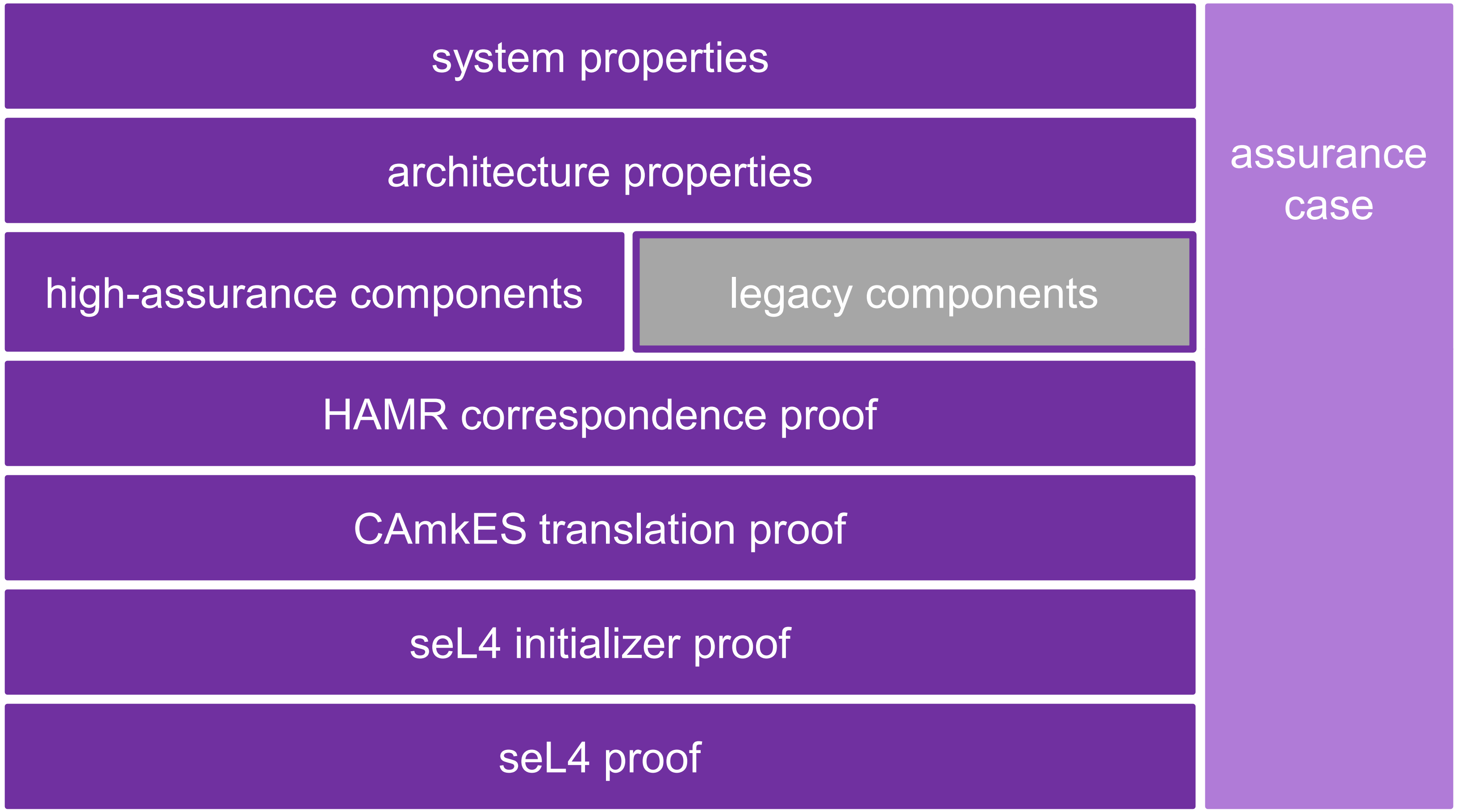
seL4 system initializer constructs system whose capabilities and objects conform to capDL specification

CAmkES translates system description into capDL spec

HAMR translates AADL model into CAmkES system description

AADL

# END-TO-END INTEGRATED FORMAL VERIFICATION

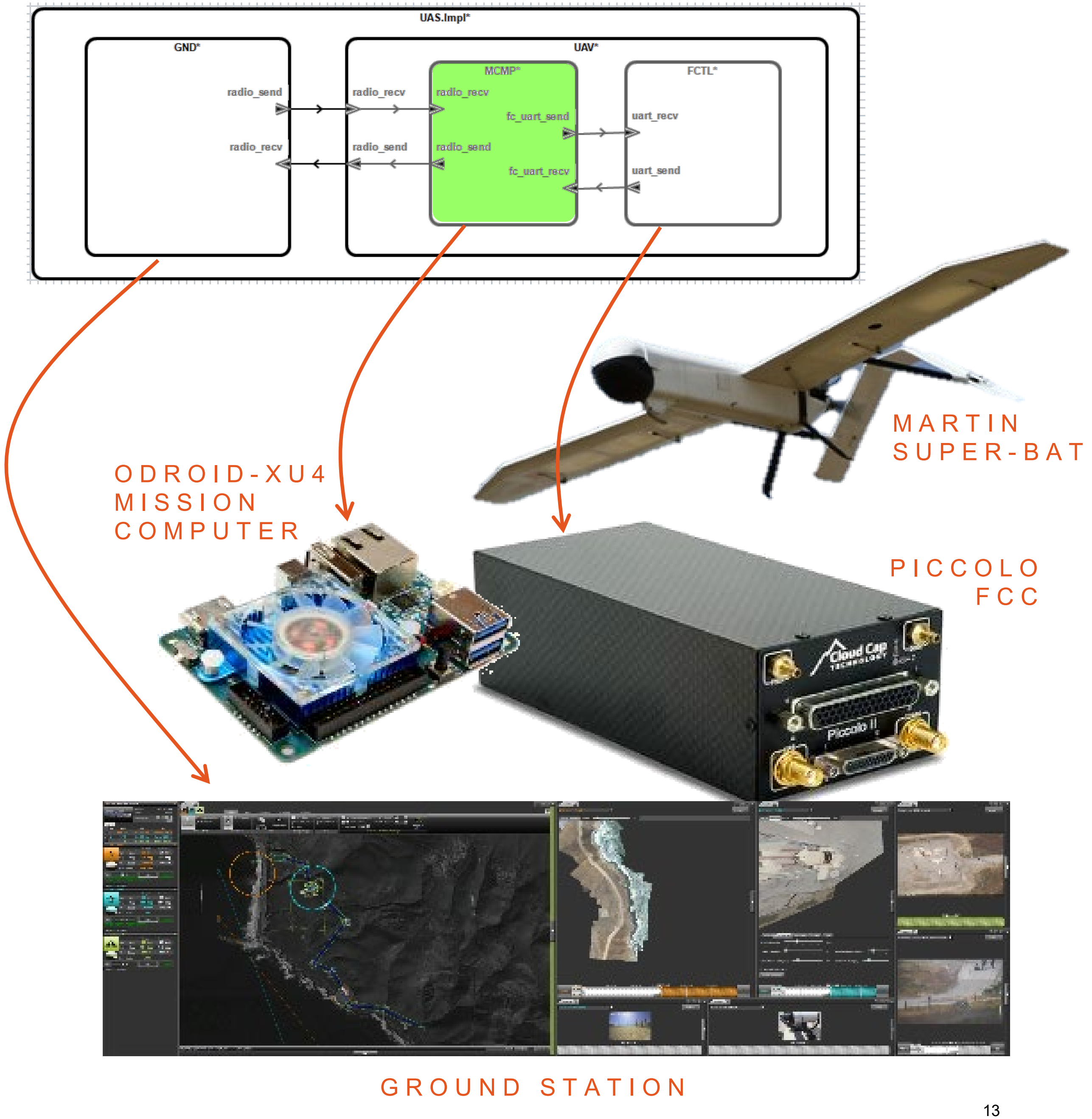


# DEMONSTRATION

## AFRL UXAS AUTONOMOUS MISSION PLANNER

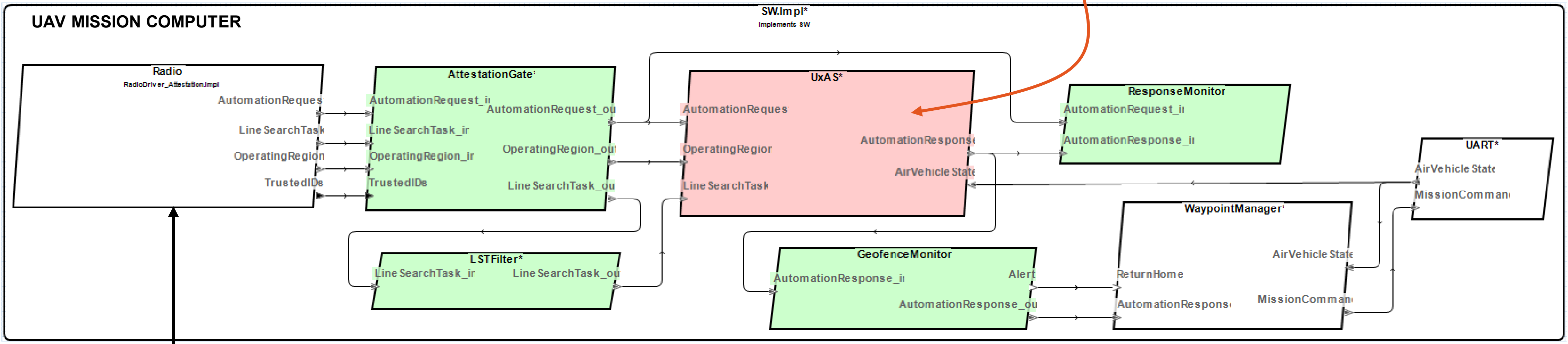
- Ground Station sends automation requests to UAV
- UAV Mission Computer processes requests and generates flight plans
- UAV Flight Control Computer computes guidance commands to follow current segment of flight plan

- CASE evaluation team developed cyber attacks on baseline platform
- Collins team hardened platform using BriefCASE tools
- Evaluation team attacks ineffective against hardened platform



# ATTACKS / MITIGATION

2 Trojans added to UxAS PlanBuilderService
Replaces AutomationResponse with plan that violates KeepOutZone




1 Malformed ground station messages

**GND STN**

3 Code added to ground station software to generate malicious requests

All UAV software runs on formally verified seL4 secure kernel

UxAS software isolated in Linux virtual machine (cyber retrofit)

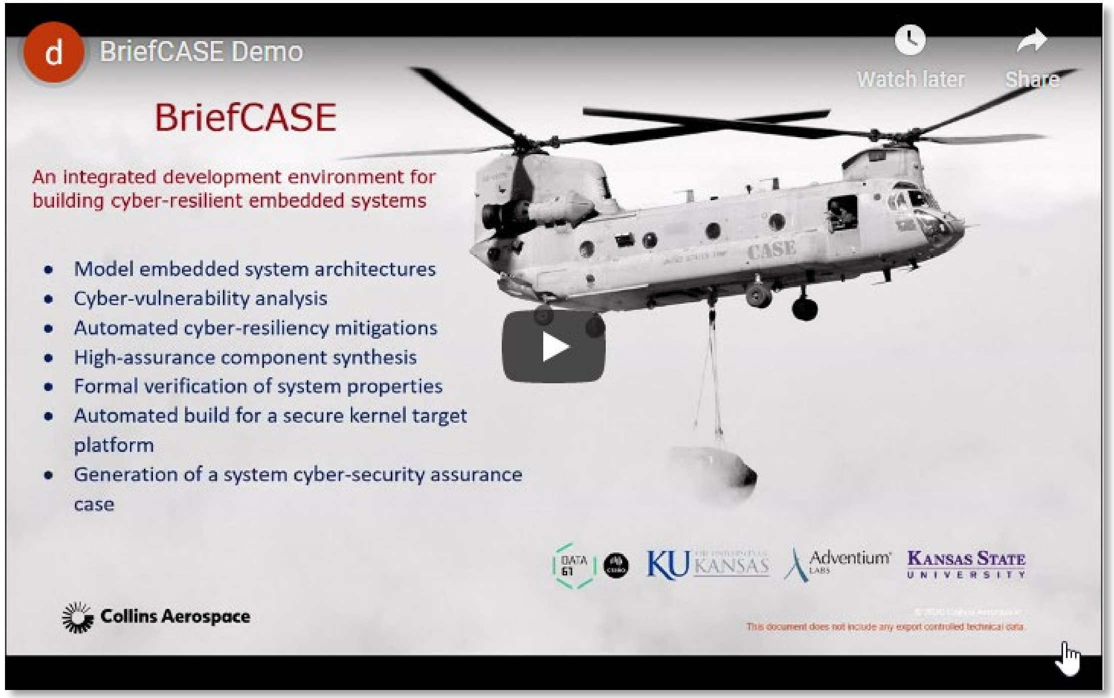
Specific attacks from evaluation team mitigated:

1. Malformed messages blocked by high-assurance filter
2. High-assurance geofence monitor detects plan that violates KeepOutZone and sends alert to WaypointManager to trigger return to base

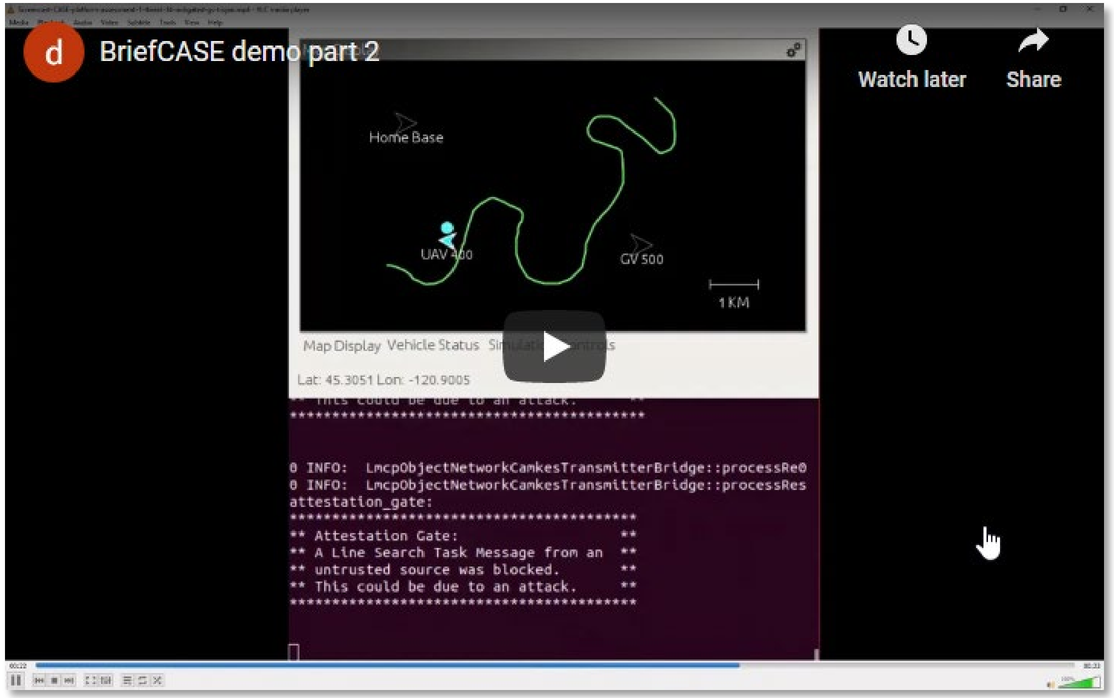
   Response monitor detects crashed UxAS planner and alerts operator
3. Remote attestation measures ground station software and detects modified code

# CASE PHASE 2 DEMO VIDEOS

## HTTP://LOONWERKS.COM/PROJECTS/CASE



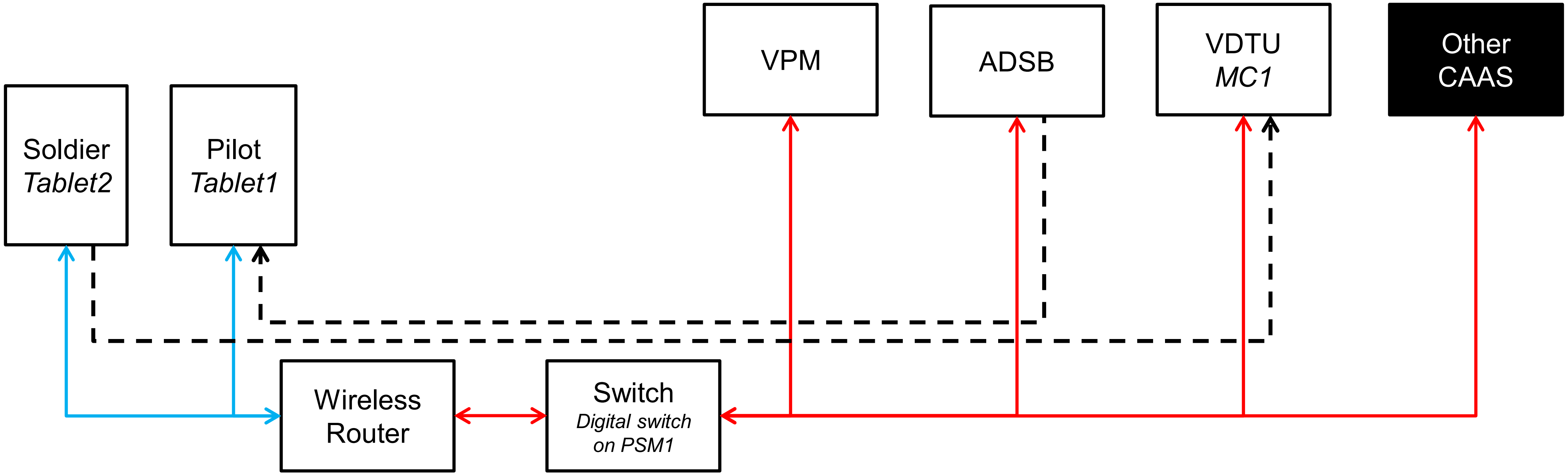

This video provides a demonstration of the BriefCASE tool environment, showing how to use the tools to address multiple cyber-resiliency requirements for a UAV mission computing system (22:35).




In part two, we run the hardened UAV mission computing system built in the first video and test it against several cyber attacks to show the effectiveness of the approach (10:13).

# PH3 DEMO PLATFORM : BASELINE (UNHARDENED)

## COLLINS COMMON AVIONICS ARCHITECTURE SYSTEM (CAAS)

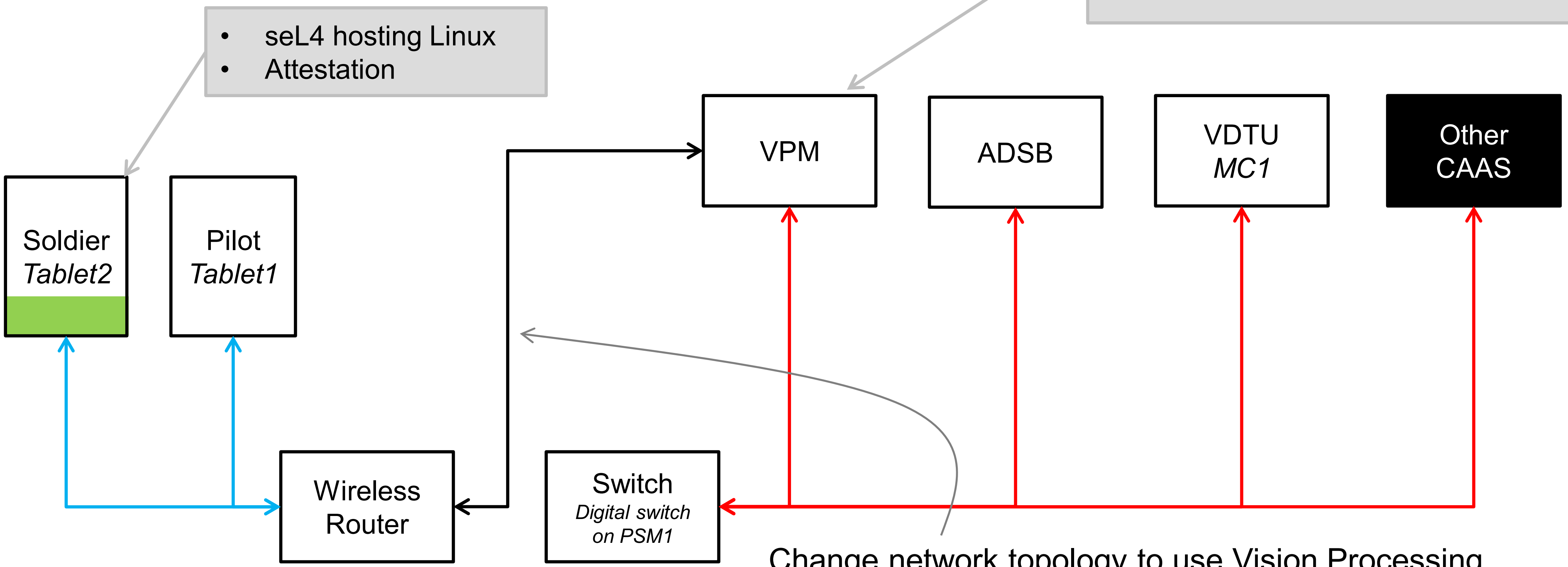



Wireless device access

# PH3 DEMO PLATFORM : HARDENED

## COLLINS COMMON AVIONICS ARCHITECTURE SYSTEM

- Filter messages to/from tablets
- Attestation of tablet(s)
- Monitor ADSB traffic for spoofing
- Virtualization for legacy SW (optional)

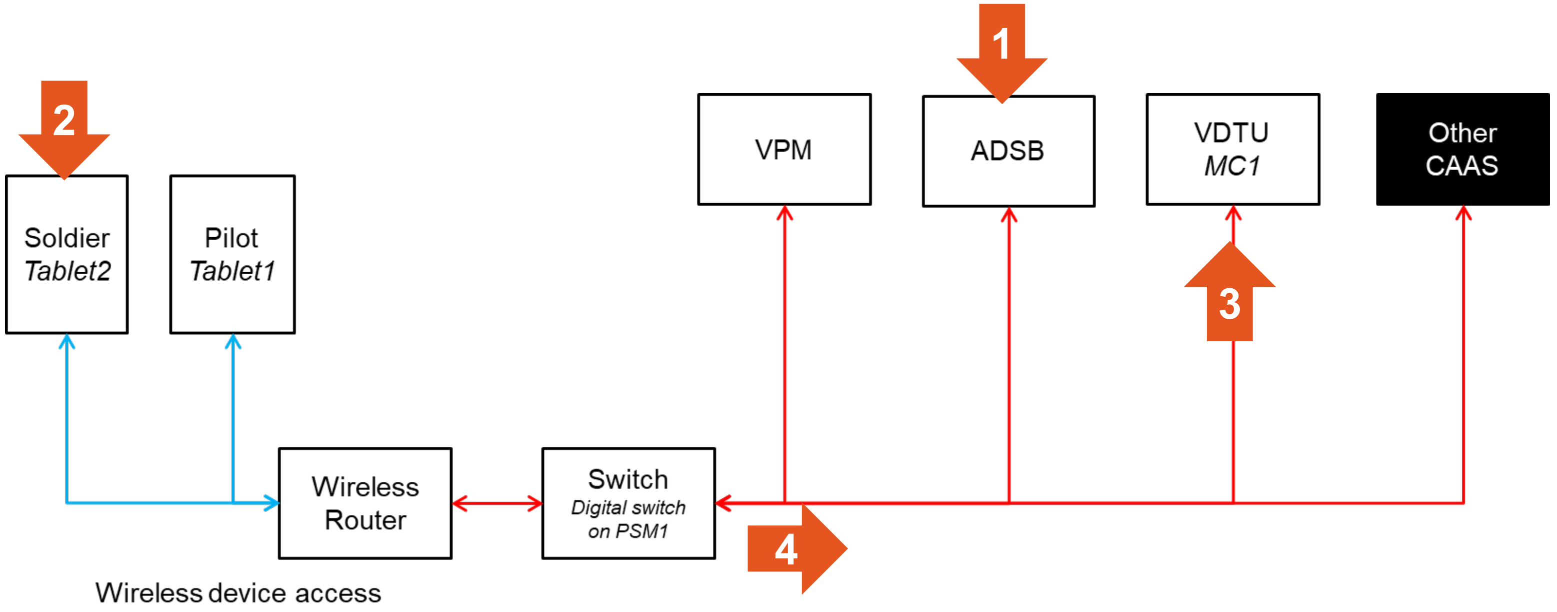



Wireless device access

Change network topology to use Vision Processing Module (VPM) as guard between lower assurance wireless network/components and rest of CAAS

# TA7 PLATFORM EVALUATION ATTACKS

1. ADS-B Spoofing vs. Anomaly Detection **Monitor** (malicious traffic displayed on Pilot Tablet)
2. Infected Soldier Tablet vs. **Attestation** (malicious software runs on Soldier Tablet)
3. Infected Solider Tablet vs. **Filter** (malformed message, code injection on VDTU)
4. Infected Soldier Tablet vs. **Monitor**/Guard (denial of service on CAAS network)




Collins Aerospace

# PHASE 3 DEMONSTRATION PLATFORM

## COMMON AVIONICS ARCHITECTURE SYSTEM (CAAS)


https://www.youtube.com/watch?v=77xCISlJpkk

U.S. Air Force photo by Staff Sgt. Elizabeth Rissmiller/Released

# CONCLUSION

- **Developer assistance to implement cyber-resiliency**
  - Automated architecture transforms for threat mitigation
  - High assurance components generated from specifications
  - Techniques to deal with legacy code (cyber retrofit)
- **MBSE environment for high-assurance cyber-resilient system development**
  - Build system directly from detailed, verified AADL model
  - Makes the security guarantees of seL4 accessible to system developers
  - Ability to target different platforms to facilitate incremental debugging/development
- **Integration of formal verification/proof**
  - Formal verification of functional and cyber-resiliency properties, information flow properties, component proofs
  - Code generation equivalence to model, seL4 build preserves properties
  - Integrate evidence as an assurance case demonstrating how/why requirements are satisfied

**Open source tools : GitHub.com/Loonwerks/formal-methods-workbench**
**Demo videos : Loonwerk.com/projects/case.html**

**CY-RES**
Architectural Transformations for Cyber-Resiliency

**MBSEC**
Model-Based Systems Engineering for Cybersecurity

**PROOF**
End-to-End Integrated Formal Verification