Security Assessment of a CubeSat

Wouter Jehee (TU Delft) / Yohann Roiron (ESA)

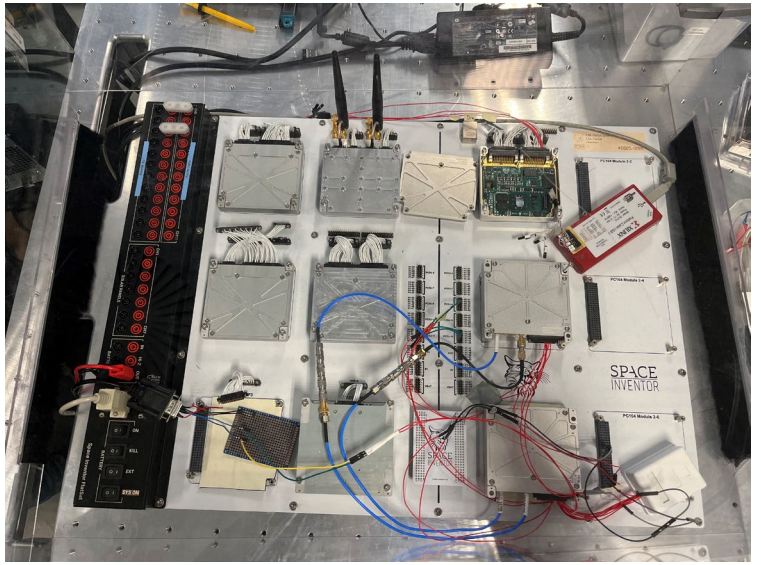
24/10/2024

Agenda

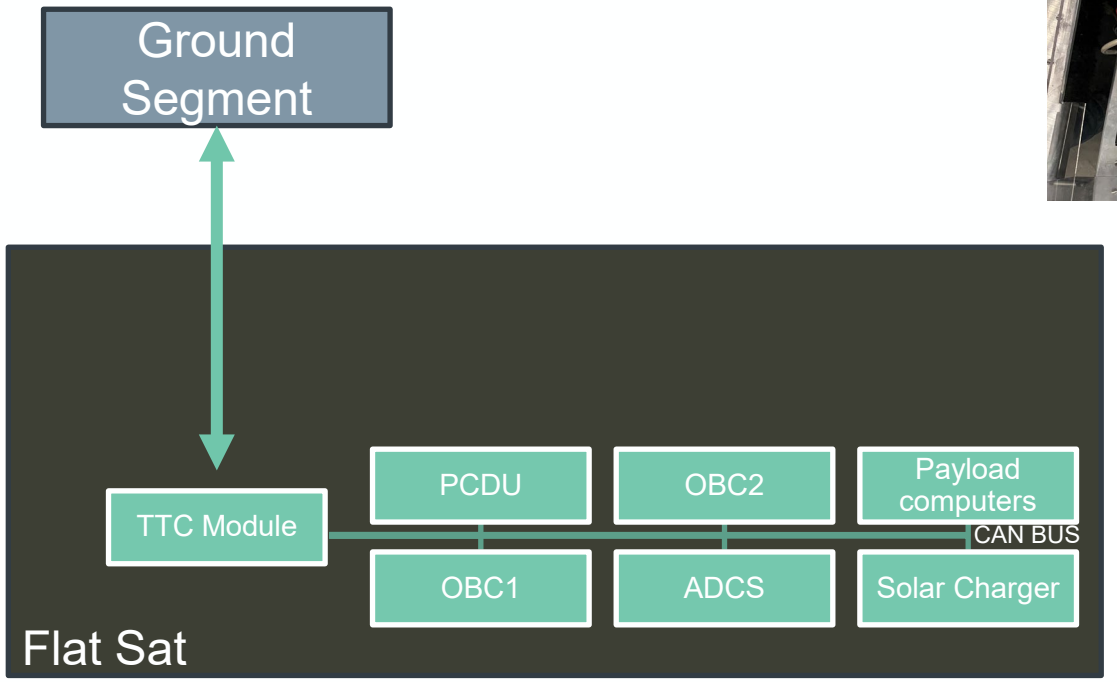
Introduction to Cubesat Security

- Threat Sources and Actors
- Risk Analysis and Scenarios
- Mitigation Strategies
    - Security Gateway
    - Authenticated Encryption (AEAD)
    - Extensions to Other Protocols
- Conclusion

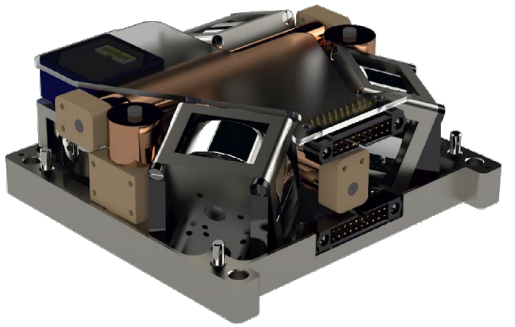
Setup presentation



7. Application layer	Application specific
6. Presentation layer	CSH Libparams
5. Session layer	
4. Transport layer	
3. Network layer	CSP
2. Data link layer	SDLS
	TC/TM Space Datalink Protocol
	TM Sync. And Channel Coding
1. Physical layer	RF and Modulation Systems



2* Arm Cortex-M7



ADCS Module

Brief Risk analysis

Scope: Intentional disruption of the satellite

Security objective:

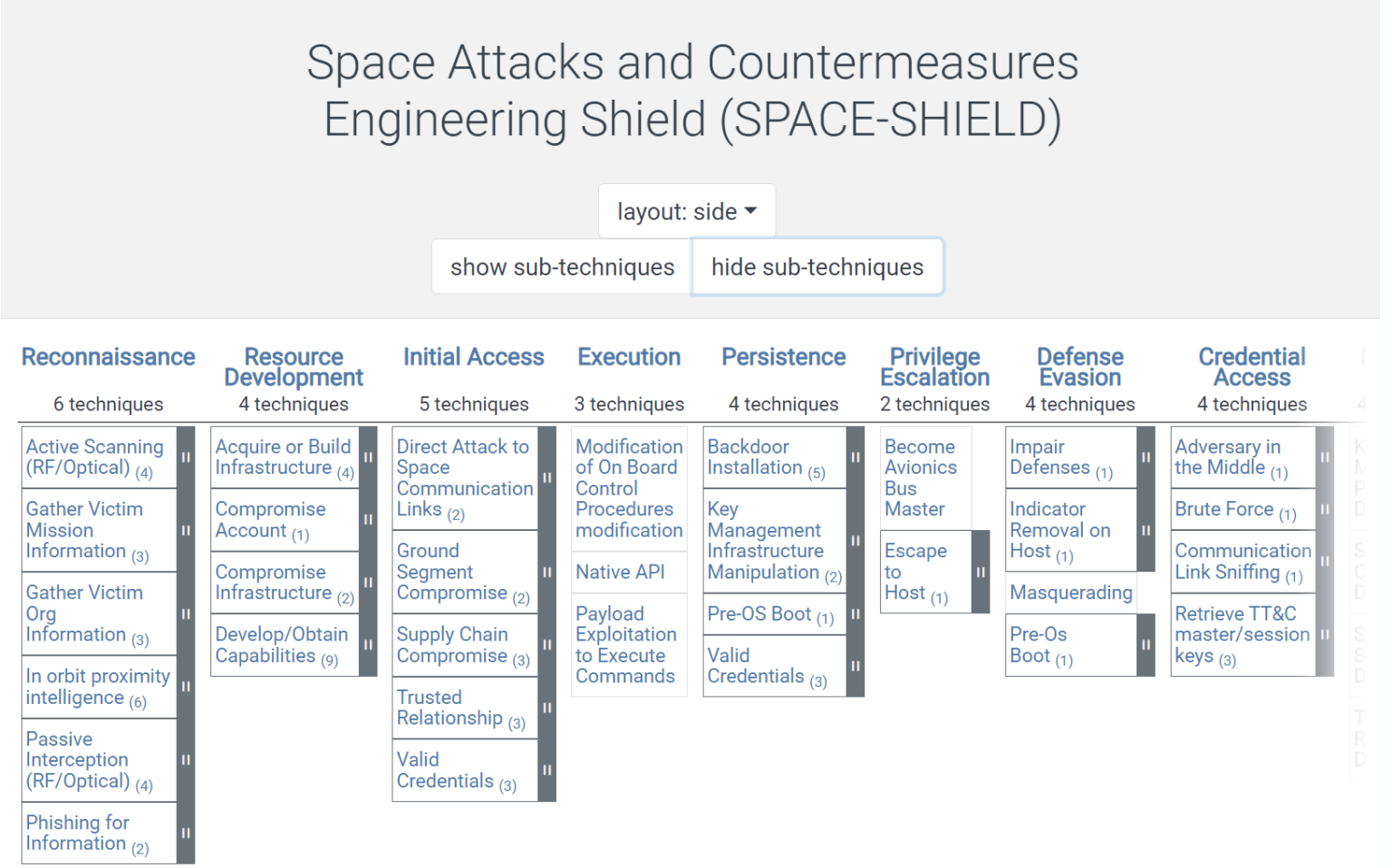
- Protect investments in space
- Protect satellite manufacturer & operator reputation, image & interests
- Ensure confidentiality of the mission data (limited time) and status of the space segment
- Preserve integrity of the mission data
- Maintain control over the on board components (OBC, payload, etc.)

Threats actors (from CCSDS 350.1-G-3: Security Threats against Space):

Threat actor	Type	Internal/External	Objective
Public	Group	External	Defeat
Hacker/script kiddie	Individual	External	Defeat
Disgruntled employee	Individual	Internal	Resist
Hacktivist/hacking group	Group	External	Resist
Insider helping other	Group	Internal	Deter
Foreign espionage	Organization	External	Deter
Unfunded terrorist	Individual	External	Deter
State sponsored	Group	External	Deter

Main risk and associated mitigations:

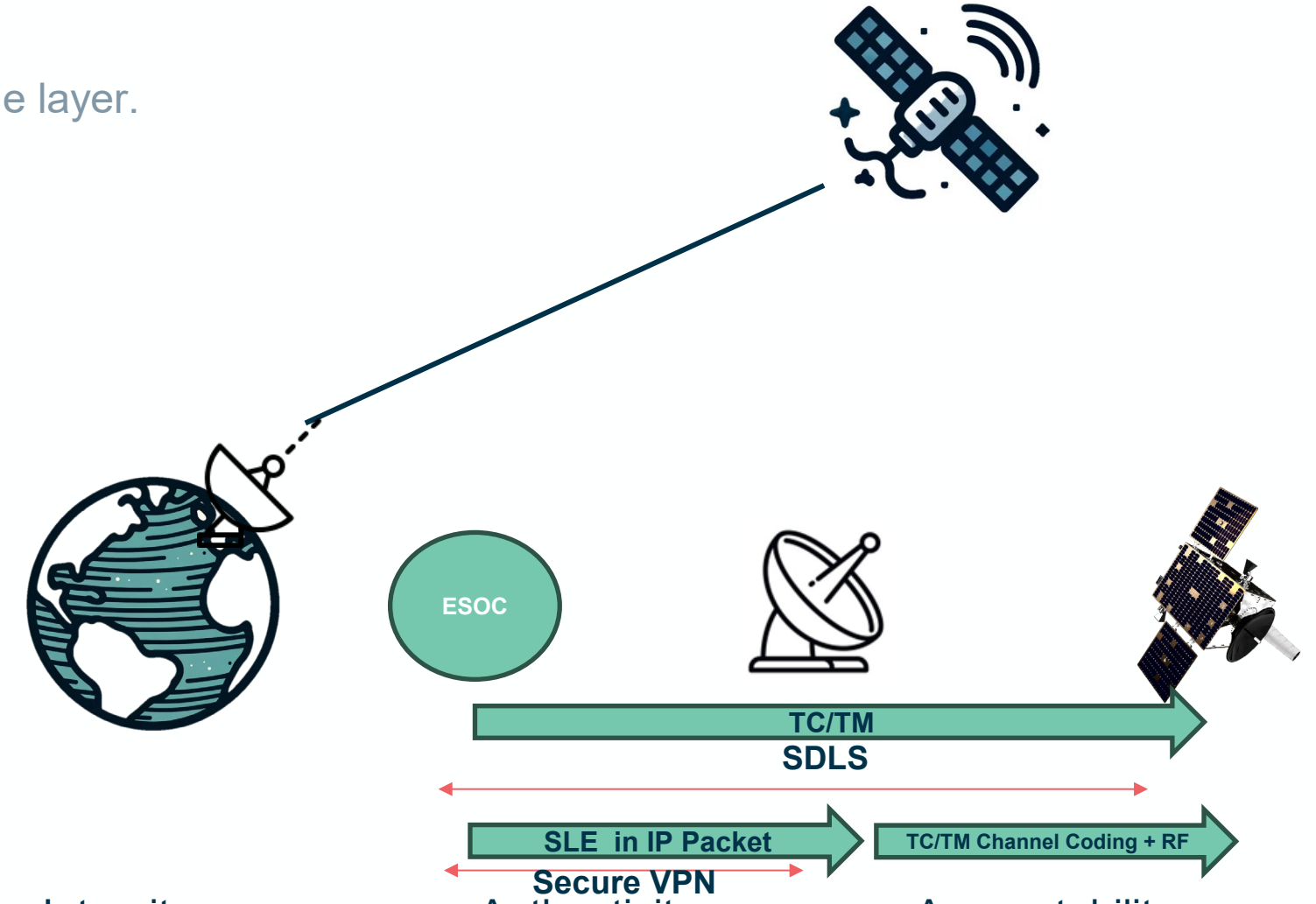
- Direct Attack on communication link
- Onboard Software Update Vulnerability
- Equipment Software Tampering
- Internal Bus compromission



Communication link security

Risk: Direct Attack on communication link

Mitigation: Authenticated encryption at frame layer.

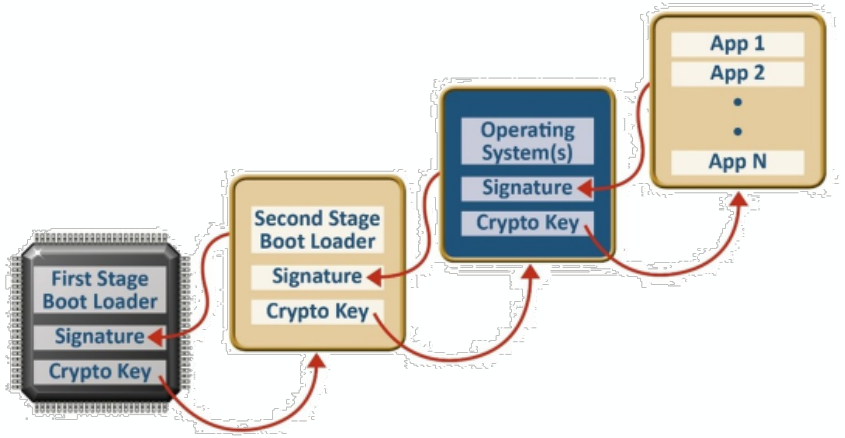


Confidentiality | Availability | Integrity | Authenticity | Accountability

Secure boot and software update

Risk: Onboard Software Update Vulnerability

Mitigation secure boot & secure software update



Each step of the bootloader is responsible for checking the next one

The first Stage is not updatable

From a security point of view:
Long term asymmetric keys to be able to sign the system
Algorithms that we can trust for a long time
Key lifetime : 10-20 years
Keys cannot be renewed at lower layer
Trust anchor for the rest of the system

Confidentiality	Availability	Integrity	Authenticity	Accountability
□	□	■	■	□

Software fuzzing

Risk: Security vulnerability not detected in the code

Mitigations: Perform fuzzing during

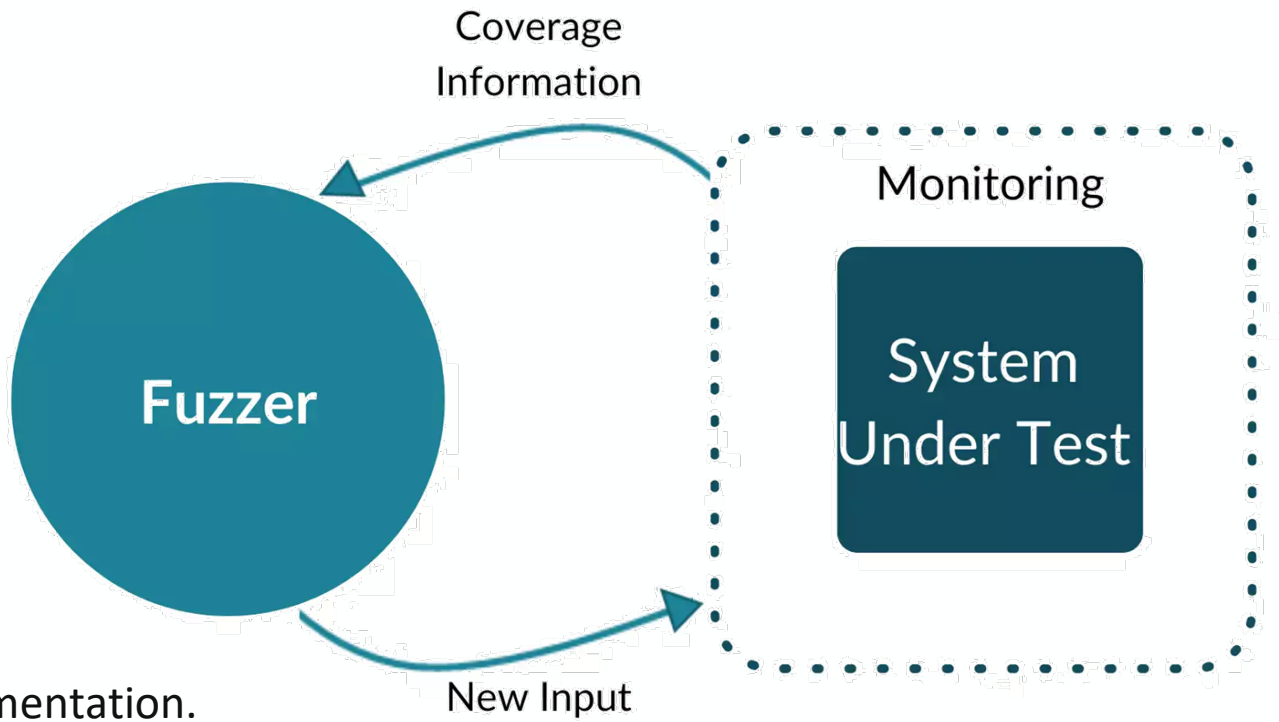
Definition: Fuzz testing is a software testing technique that uses random inputs to find security vulnerabilities or bugs by causing crashes in the application.

In our case:

Implemented a fuzzer on the OBC software communication stack (libcsp/libparam).

Found few bugs, some intentional security vulns:

Peak/Poke in the memory



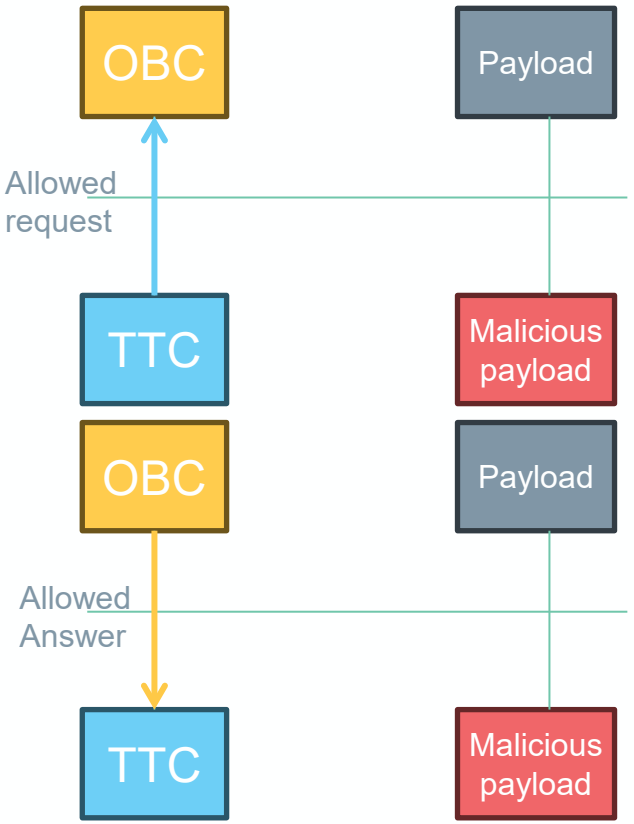
Note: It does not replace proper standard implementation.

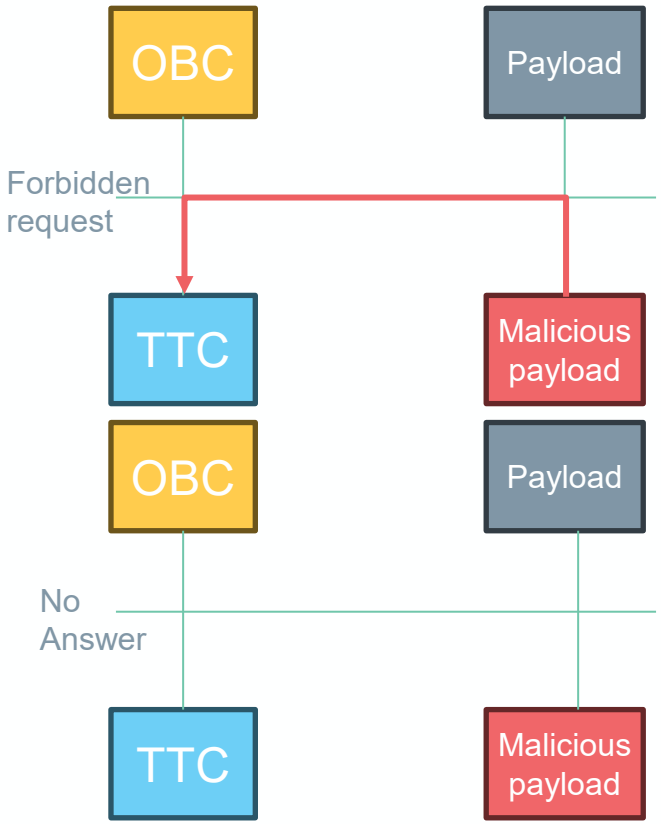
Confidentiality | Availability | Integrity | Authenticity | Accountability

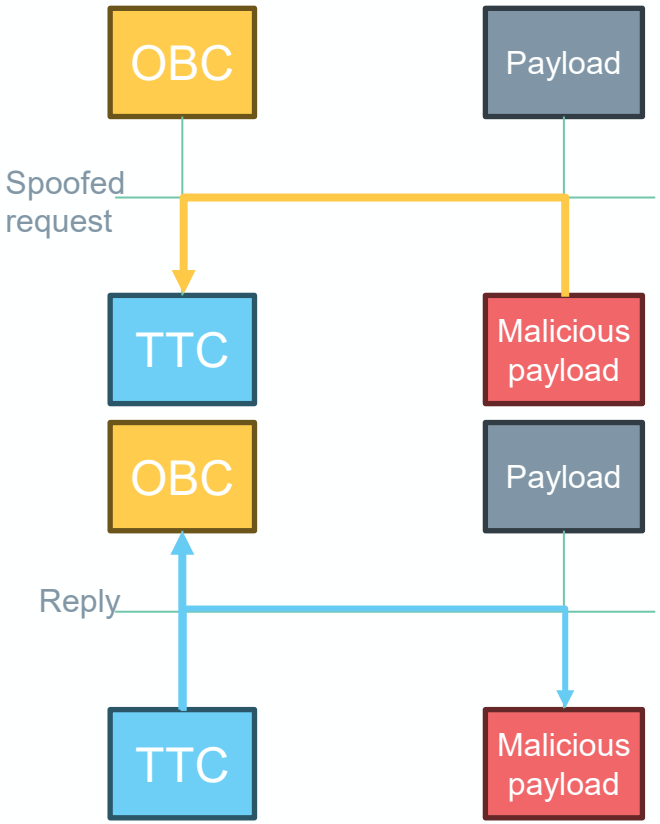
Network segregation and firewalls (1/2)

Risk: Bus compromission and Lateral Movement via common Avionics Bus

Mitigation: Network segregation on the shared bus.

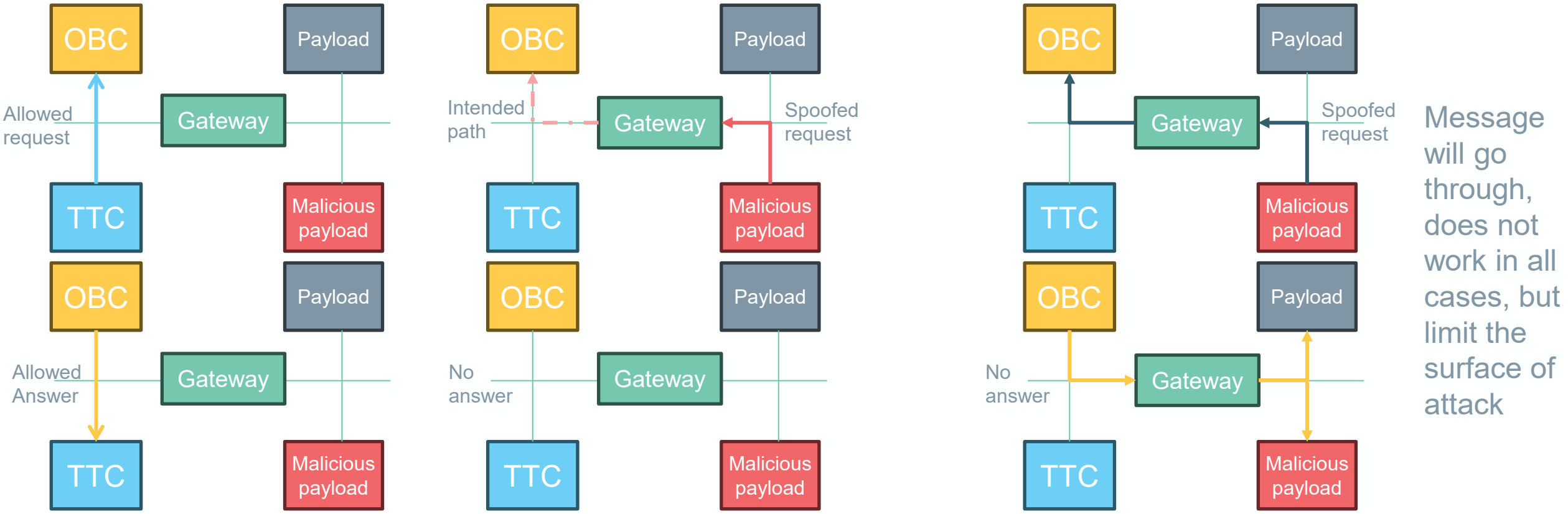






Network segregation and firewalls (2/2)

Mitigation: Network segregation on the shared bus.

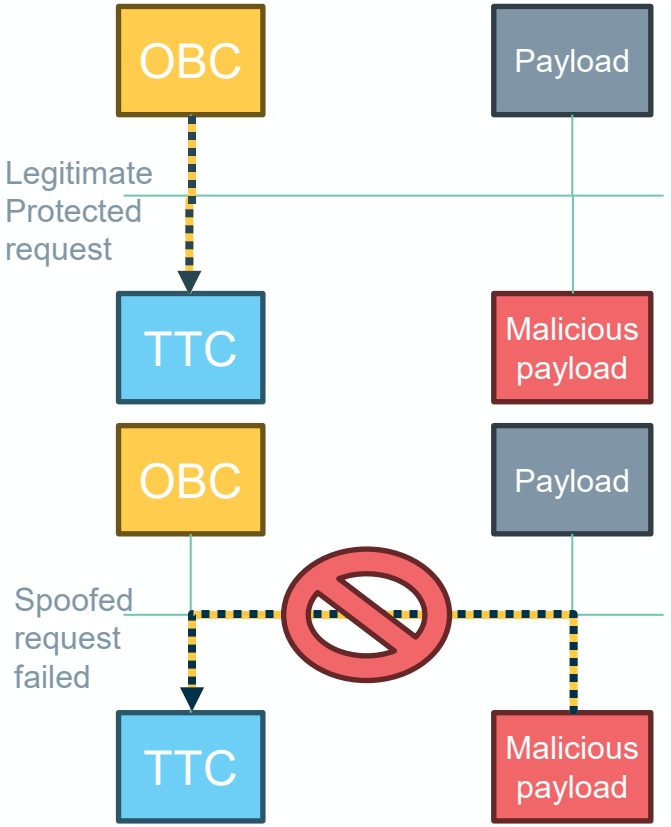


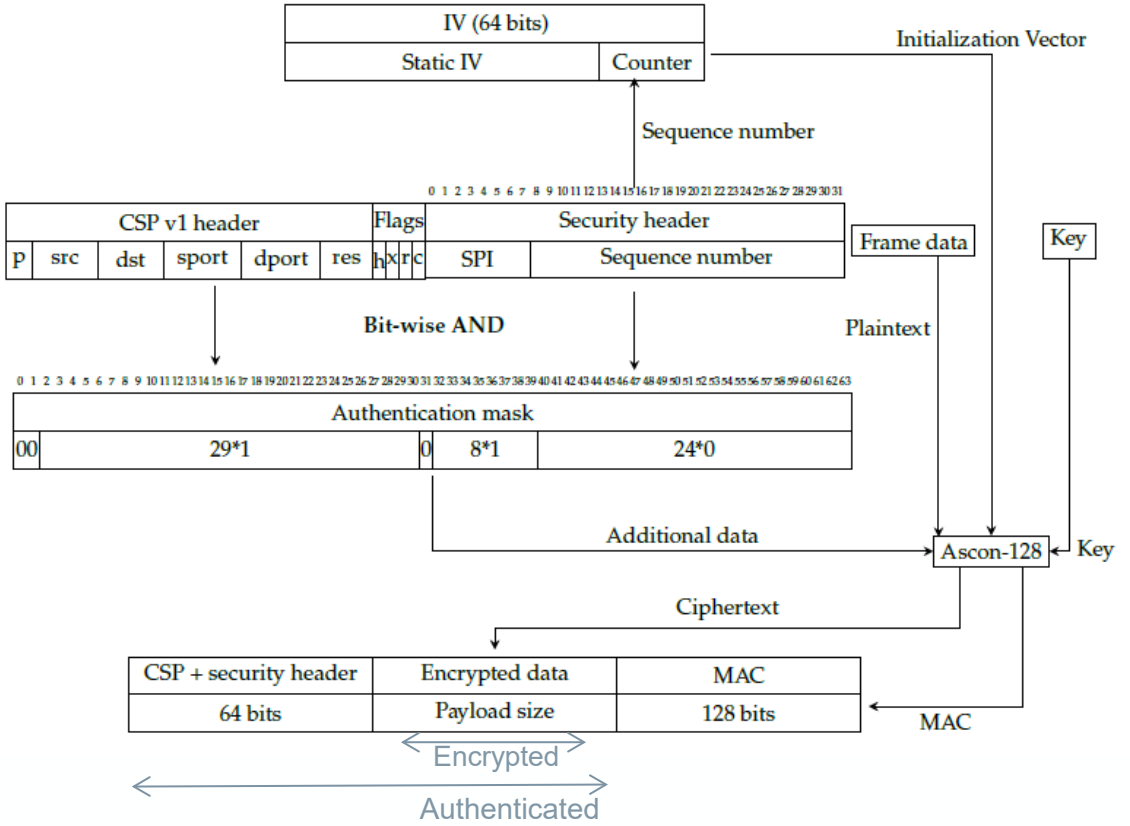
Message will go through, does not work in all cases, but limit the surface of attack

AEAD implementation

Risk: Bus compromission and Lateral Movement via common Avionics Bus

Mitigation: Bus layer security



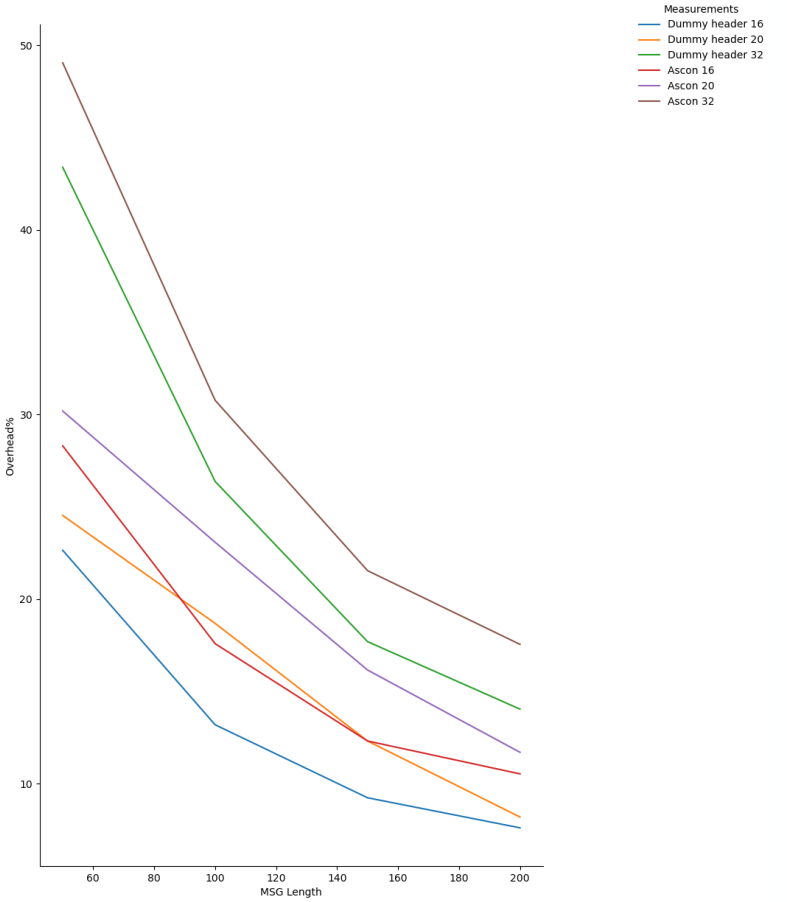


Confidentiality | Availability | Integrity | Authenticity | Accountability

Performance

Performance overhead using ASCON reference implementation on Cortex M7 (RTT), for 200 messages

MSG Length	No header	Dummy header 16	Dummy header 20	Dummy header 32	Ascon 16	Ascon 20	Ascon 32
50 bytes	530	650	660	760	680	690	790
100 bytes	910	1030	1080	1150	1070	1120	1190
150 bytes	1300	1420	1460	1530	1460	1510	1580
200 bytes	1710	1840	1850	1950	1890	1910	2010

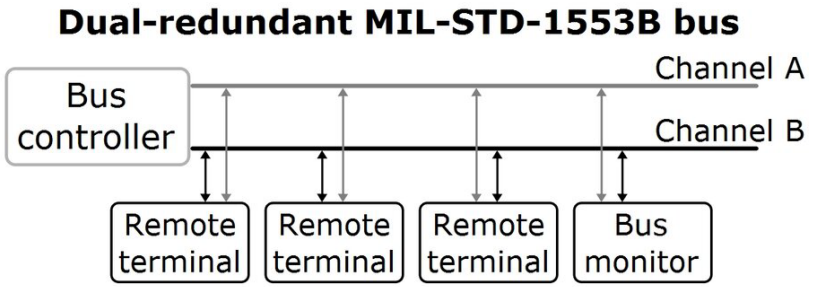


=> 20% overhead in our case, due to both Cryptographic processing but also additional data transmission

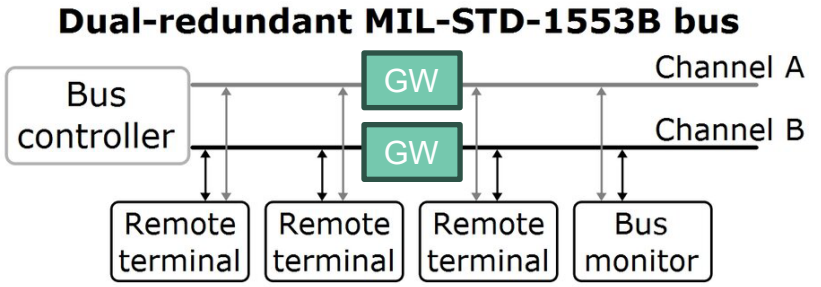
Extension to other Bus Protocols (1/2)

Milbus Protocol
Speed: 1Mb/s
Message size: Up to 32 data words (64 bytes)

Could implement Security gateway, but this will require specialized hardware.
What is the impact on reliability of this additional hardware?



AEAD implementation is expensive, overhead is 30%/50% using a 20B/32B security part.
Could be implemented if longer messages are required.
Could be also implemented at higher layer (network level).



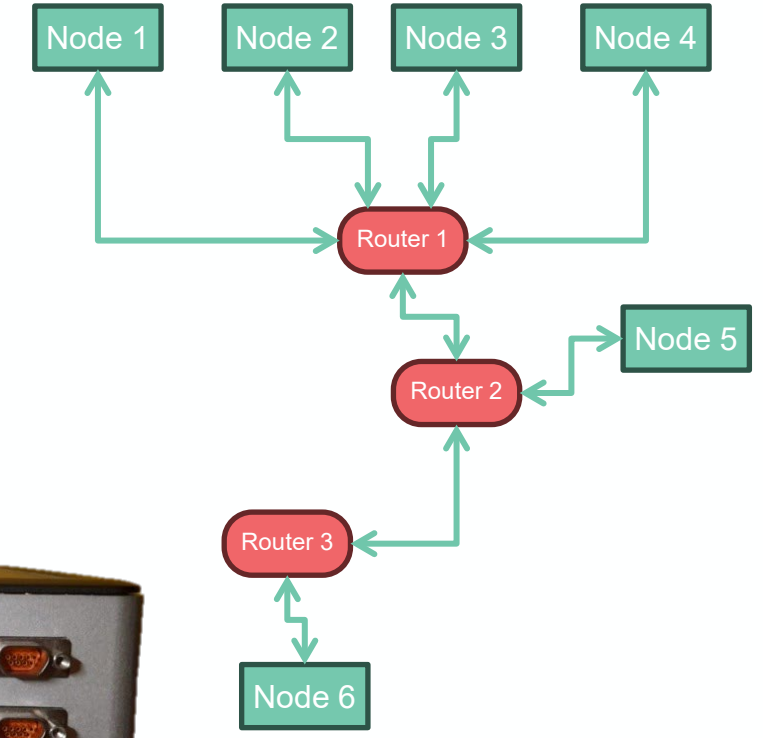
Extension to other Bus Protocols (2/2)

Space Wire Protocol

- Speed: 2 to 400Mb/s
- No packet size limit, (implementation specific)

Security Gateway can be iimplemented at router level.
Network dataflow can be fix for a specific mission.
No eavesdropping issue, as this is a point-to-point scenario.
Recure secure management of the routers.

AEAD overhead is small at transmission level (<1%)
Computation cost is still there.



Source: SpaceWire User's Guide

Conclusion

Structured risk methodology ensures comprehensive security measures, tailored to the needs.

Practical measures for security, such as fuzz testing, security gateways, and authenticated encryption

Enhancing mission security by focusing on practical and effective solutions