2024 NDIA MICHIGAN CHAPTER
GROUND VEHICLE SYSTEMS ENGINEERING
AND TECHNOLOGY SYMPOSIUM
MODULAR OPEN SYSTEMS APPROACH (MOSA) TECHNICAL SESSION
AUGUST 13-15, 2024 - NOVI, MICHIGAN

Towards Deployment of a Zero-Trust Architecture (ZTA) for Automated Vehicles (AV)

Victor Murray[1] CISSP®, Scott Lathrop[2] Ph.D., CISSP®, Dariusz Mikulski[3] Ph.D.,
[1]Southwest Research Institute, San Antonio, TX
[2]Raytheon BBN Technologies, Cambridge, MA
[3]US Army DEVCOM Ground Vehicle Systems Center, Warren, MI

ABSTRACT

Automated Vehicles (A)V development historically placed a significant focus on functionality and less on security. Programs such as Cybersecurity for Robotics and Autonomous Systems Hardening (CRASH) are addressing AV cybersecurity, strengthening security while simultaneously supporting the developer focus on functionality. This task is challenging due to continuous interaction by AVs with the environment through sensors and actuators, command and control, and remote connectivity. This paper presents an approach balancing functionality and security through an AV Zero-Trust Architecture (ZTA) which leverages authentication, cyber policy enforcement, and monitoring to detect and mitigate cyber-attacks. The AV ZTA approach is traceable to NIST 800-217 guidance for applying ZT concepts to Information Technology (IT) networks.

The presented AV architecture example begins with a non-self-driving baseline, adding sensors, actuators, command/control, and remote connectivity. NIST 800-207 principles are distilled into three (3) components: 1. Authentication 2. Policy Enforcement and 3. Monitoring. Authentication includes verifying software authenticity prior to booting, and use of a combination of public/private key encryption, symmetric key encryption, and Message Authentication Codes (MACs) to secure network communication. Policy Enforcement occurs at every AV network node and is overseen by a central gateway. The gateway also monitors traffic and logs issues. Together, these combine into an AV ZTA.

Several recent programs have partially implemented the outlined AV ZTA. For example, the CRASH program has implemented authentication for networked communication, policy enforcement, and monitoring. Other programs are tackling monitoring automotive CAN and ethernet busses and improving resiliency through sensor redundancy and fusion. There remain other unaddressed pieces to fully implement an AV ZTA.

1. INTRODUCTION

Modern ground vehicles rely on controlling physical access to protect in-vehicle networks. When building an AV, it is common to add autonomy on-top of an existing non-AV ground vehicle. For example, leading commercial autonomy companies such as Tesla, Waymo, General Motors, and Ford add autonomy to their existing, non-AV, vehicles.

The result of adding inherently vulnerable sensors, This paper presents a ground AV ZTA that inherently distrusts received data and communication. CRASH and other programs are addressing the AV vulnerability gap by implementing pieces of the AV ZTA. One of the focuses of these programs is to create cybersecurity solutions while minimizing the impact on autonomy developers.

2. Prior Research

Prior research focused on addressing specific vulnerabilities such as those found in Controller Area Network (CAN) bus [5] and firmware [6]. They outline how to properly implement

security mechanisms and protect systems from exploitation against specific threats. More recent publications apply Zero Trust to Automated Vehicle networks. Specifically, applying Zero Trust to Automotive Ethernet [7] and CAN bus [8].

This research forms a foundation that this paper builds upon. The ZTA leverages well published security mechanisms and takes a wholistic approach towards securing AV.

3. Introduction to Zero Trust Architecture (ZTA)

ZTA is a set of principles and a corresponding high-level reference architecture that the Department of Defense (DoD) is advocating to improve cybersecurity posture and counter ongoing adversarial cyberspace activity [9, 10]. ZTA moves cybersecurity mechanisms towards a focus on users, compute assets, and resources and away from purely static, perimeter-based defenses. ZTA principles are typically geared towards enterprise IT networks but can also be relevant to AVs, although differences in implementation vary significantly.

From a daily workflow perspective, users have observed zero trust approaches manifest as more frequent requirements to present one's credentials (i.e., username/password) coupled with multi-factor authentication. However, the goals and concepts of ZTA are much more encompassing than these visible manifestations.

As the name implies, zero-trust assumes that there should not be an implicit trust between an active *subject* and a passive *resource.* The subject requires access to a resource and that access is realized through a communication channel such as a network transport, file system access, or inter-process communication. In practical terms, a subject is an active entity, such as a person who is requesting access to a passive resource, such as a data source. Passive resources include objects in memory, files, database records, or a socket connection such as a web service. A subject can also be a non-person entity, such as an operating system process where that process, perhaps acting alone or on behalf of another subject, requests access to a resource. In ZTA, the system must decide whether to grant that access and establish the communication channel, such as enabling access to a memory location, opening a file, establishing a database connection, or opening a socket.

ZTA explicitly recognizes that there is no easily defined network perimeter because of trends such as remote users, off premise mobile and cloud computing, Internet of Things (IoT) devices, and insider threats. In the case of AVs, the platform itself is communicating internally between processes and externally through a command and control (C2) infrastructure. However, the vehicle itself is not operating physically or logically within that C2 infrastructure. Because of these trends, ZTA assumes that an adversary has already gained access to the environment, and to mitigate the risk of lateral movement, the system must assume that the subject has malicious intent. Therefore, subjects must continuously verify their legitimate need to access a specific resource before the system permits access. This process requires making access control as granular as possible by shrinking implicit trust zones. Table 1 summarizes the ZTA principles and their intent.

Table 1. ZTA Principles and Intent

Principle	Intent
Resource Protection Vice Perimeters	• Data and compute services require protection. • Least privilege, need to access
Continuous Subject Authentication	• Dynamic authentication • Multi-factor for end-users • Re-authentication as defined by policy (i.e., time-based, resource-based)
Conditional Authorization to Access Resource	• Approval based on *subject's need to know* and the subject's *compute asset trustworthiness.* • Access granted on per-session basis and determined by dynamic policy
Resource Integrity and Confidentiality	• Protect data at rest and transit • Secure communication regardless of network location • Data tagging
Continuous Auditing, Logging, and Monitoring	• Continuous asset inventory, diagnostics, and mitigation • Continuous event logging and monitoring • Visualization and analytics to provide situational understanding

The intent behind these principles is to protect resources at a granular level through traditional cybersecurity mechanisms such as authentication; authorization; data integrity and confidentiality; and auditing, logging, and monitoring. However, ZTA places an increased emphasis on continuous verification and runtime monitoring. Authentication occurs more frequently. Authorization includes verifying that the access control policy enables the subject's access to the resource and may also include an assessment and decision regarding the security posture of subject's compute asset. For example, is the asset up to date with patches, does threat intelligence indicate active threats against devices/operating systems of its type? The authorization policy is a set of access rules based on attributes assigned to subject and resource for which a policy engine checks at runtime through a "trustworthy algorithm." This algorithm, implemented by a *policy engine* requires inputs from threat intelligence, asset inventories, and auditing, logging, and monitoring systems

that provide the policy engine a sense of the current state of the system. Some assets (e.g., personal devices) may be treated differently than others (e.g., enterprise devices up to date with patches and not running unknown software), granting the subject more or less access to the resource depending on the situation. Of course, implementation of such a ZTA requires engineering a balance between performance, scalability, functionality, and security. This tradeoff is especially important in the case of an AV, as some functionality like safety (e.g., obstacle avoidance) cannot be denied by the ZTA's policy engine.

Several technologies are part of a ZTA solution for enterprise networks. These include virtualization, containerization, and network virtualization to support micro-segmentation. Segmentation is not new—firewalls, virtual local area networks (VLAN), and access control lists (ACL) have been around for years. The difference in a ZTA is that micro-segmentation policies are applied to individual workloads.

4. Automated Vehicle Architecture

To illustrate incrementally how ZTA applies to AV, this section presents an example AV architecture, without cybersecurity controls or ZTA. The foundation is built on a non-self-driving baseline; adding sensors, actuators, command/control; and then adding remote connectivity. Section 0 will then apply ZTA to this baseline.

4.1. Ground Vehicle Baseline (No Automation)

AVs are commonly built on top of non-automated ground vehicles. These non-automated vehicles use Electronic Control Units (ECUs) that are on one or more networks to control things like the engine, transmission, tires, and gauges. ECUs include hardware (processor, sensors, memory, interface) and firmware. The communication between ECUs uses CAN, Local Interconnect Network (LIN), and automotive Ethernet. An example ECU network and connection diagram is shown in Figure 1.

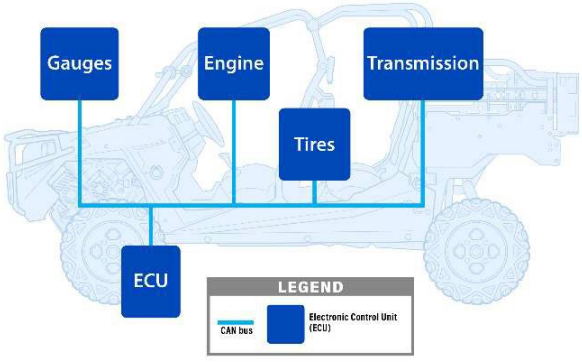

Figure 1. Example Ground Vehicle Baseline ECU Network and Connection Diagram

4.2. Add Control and Sensors

To convert a non-automated ground vehicle into an AV, several systems are added:

- **Sensors** – Sensors provide location and object avoidance information. Our example AV relies on LiDAR and Camera to track objects for avoidance and GPS for location.
- **Drive By Wire** – Provides actuators for steering, brakes, throttle, and shifter, as well as an electronically controlled interface. The example vehicle uses the CAN bus as the electrical control interface.
- **Control Computers** – The control computers (C2 Computer, Main and Localization Computer, Perception Computer) receive data from sensors, receive desired path from operator, and control the vehicle through the Drive by Wire ECU.

4.2.1 Control Software

The Army Robotic Common Software (ARCS) – previously known as the Robotic Technology Kernal (RTK) – is the Army's ground vehicle autonomy library and is built on Robot Operating System 2.0 (ROS2). Data Distribution Service (DDS) is the publish/subscribe middleware that implements the ROS2 topics, services, and actions as shown in Figure 2. A ROS2 middleware (RMW) component serves as an adapter between the ROS2 client and the specific DDS implementation, which the user must choose. Currently the options are FastRTPS, Cyclone DDS, and RTI's ConnextDDS. The software stack running on the control computers is a combined ARCS, ROS2, and DDS system.

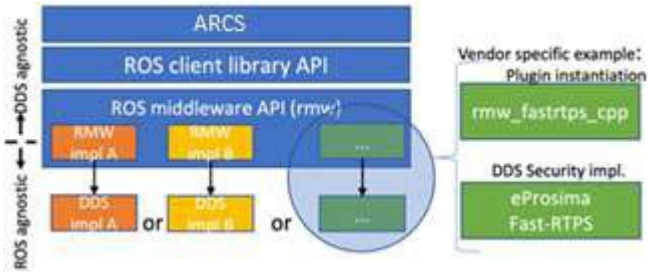

Figure 2. In ROS2, each DDS Provides the ROS Middleware to Interface Between ROS and DDS

Each DDS implementation provides a DDS-Security specification that implements authentication, access control, encryption for data in transit, data tagging, and security event logging—all key capabilities required for the ZTA. Access control is at the DDS topic level (i.e., message) and prevents unauthorized publishers from publishing to a topic and, similarly, unauthorized subscribers from reading that topic's messages. The configuration of this policy is static, however, so it does not enable updates to the policy at runtime.

In theory, DDS implementations are interoperable, but in practice that is not the case—specifically when using security plugins. As the RMW is a software adapter, certain assumptions are encoded, such as QoS and security settings, which provide the developer immediate out-of-box support but limits configuration flexibility. Understanding the specific DDS implementation is crucial in tuning ARCS behavior and performance.

No clear performance (speed, memory) advantage was found between DDS implementations, although there has been a lot of reported performance measurements within the ROS2 community. For now, performance is a race between DDS vendors with no clear leader.

4.3. Add Remote Connectivity

While connecting to an AV remotely is not a functional requirement, mission objectives often necessitate remote communication with the AV. Commercial automobiles have many wireless interfaces including Bluetooth, Telematics (cellular), Wi-Fi, and Tire Pressure Management Systems (TPMS) that can also add additional remote threat vectors. The full AV architecture shown in Figure 3 uses an ethernet radio to connect the Command and Control (C2) computer, remotely, to the vehicle network.

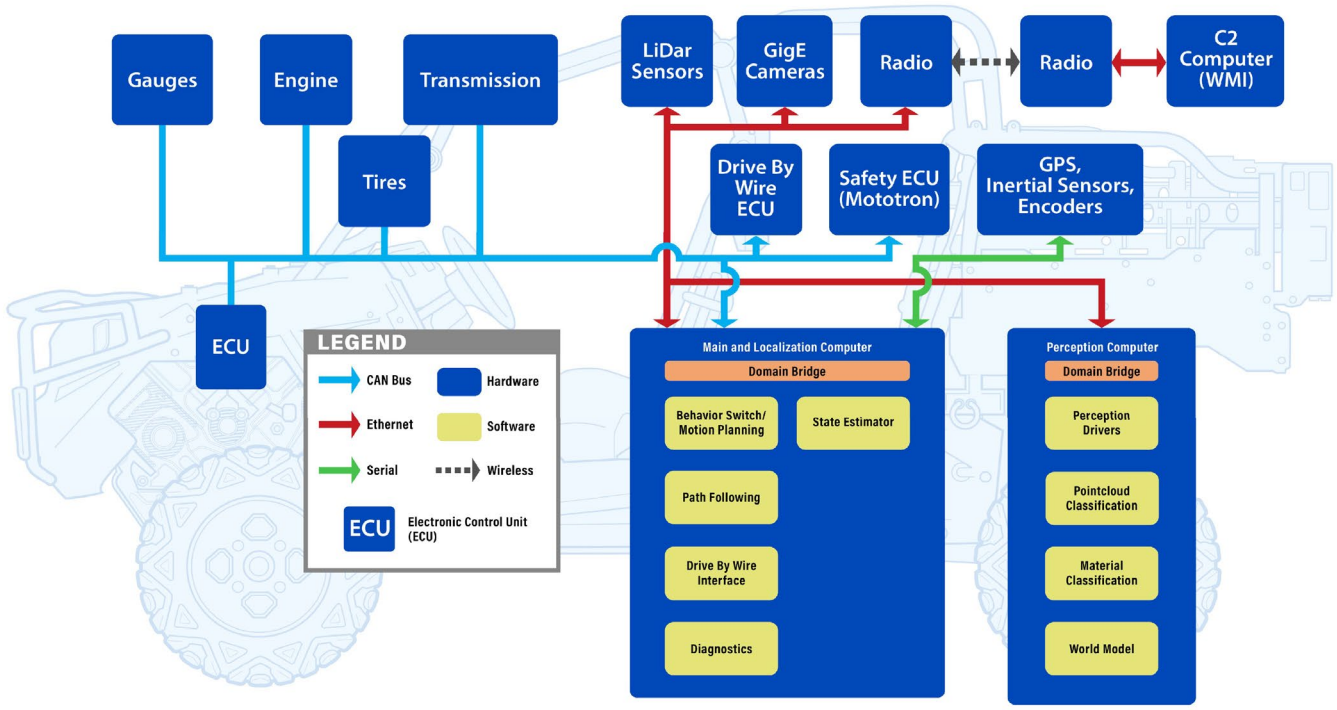

Figure 3. AV Architecture with Remote Connectivity; Sensors, Drive-by-Wire, and Control Computers; and Ground Vehicle Baseline ECU. This architecture is used to Outline the ZTA for AV

5. Zero-Trust Architecture (ZTA) for Automated Vehicles (AV)

This section presents a ZTA for AVs. At the time of publishing, the ZTA has been partially implemented. Section 6 details research performed on CRASH in support of applying ZTA to Avs, and section 7 details additional efforts. Section 8 details components of ZTA that are yet to be implemented and are planned for future efforts.

The ZTA for AV is based on the following principles outlined in NIST SP 800-207 [9]:

(1) All ROS2 nodes and ECUs are subjects.
(2) All ROS2/DDS topics and ECUs are resources.
(3) All communication is secured.
(4) Access is granted on per session basis.
(5) Access is determined by the policy engine.
(6) Network communication is monitored and tracks security posture of all resources.
(7) Authentication and authorization are enforced before access is allowed.
(8) Information is logged to improve security.

The ZTA for AV distils these National Institute of Standards and Technology (NIST) principles into the following components, and the corresponding NIST principles are defined in parenthesis:

- **Authentication** (1) (2) (3) (4) (7) – Requires devices or processes to prove their identity, commonly using public/private key encryption, digital signatures, Message Authentication Codes (MACs), or a username/password.
- **Policy Enforcement** (5) – Policy engine clearly defines which devices need access to which resources. This policy is enforced prior to allowing access and is updated to ensure resilience to the latest threats.
- **Monitoring** (6) (8) – Enables the detection and logging of devices or processes. For example, attempt to incorrectly access networked resources and system errors are logged.

The ZTA for AV shown in Figure 4 applies these components through security features (e.g., Authentication via DDS Security) that are detailed in following sections.

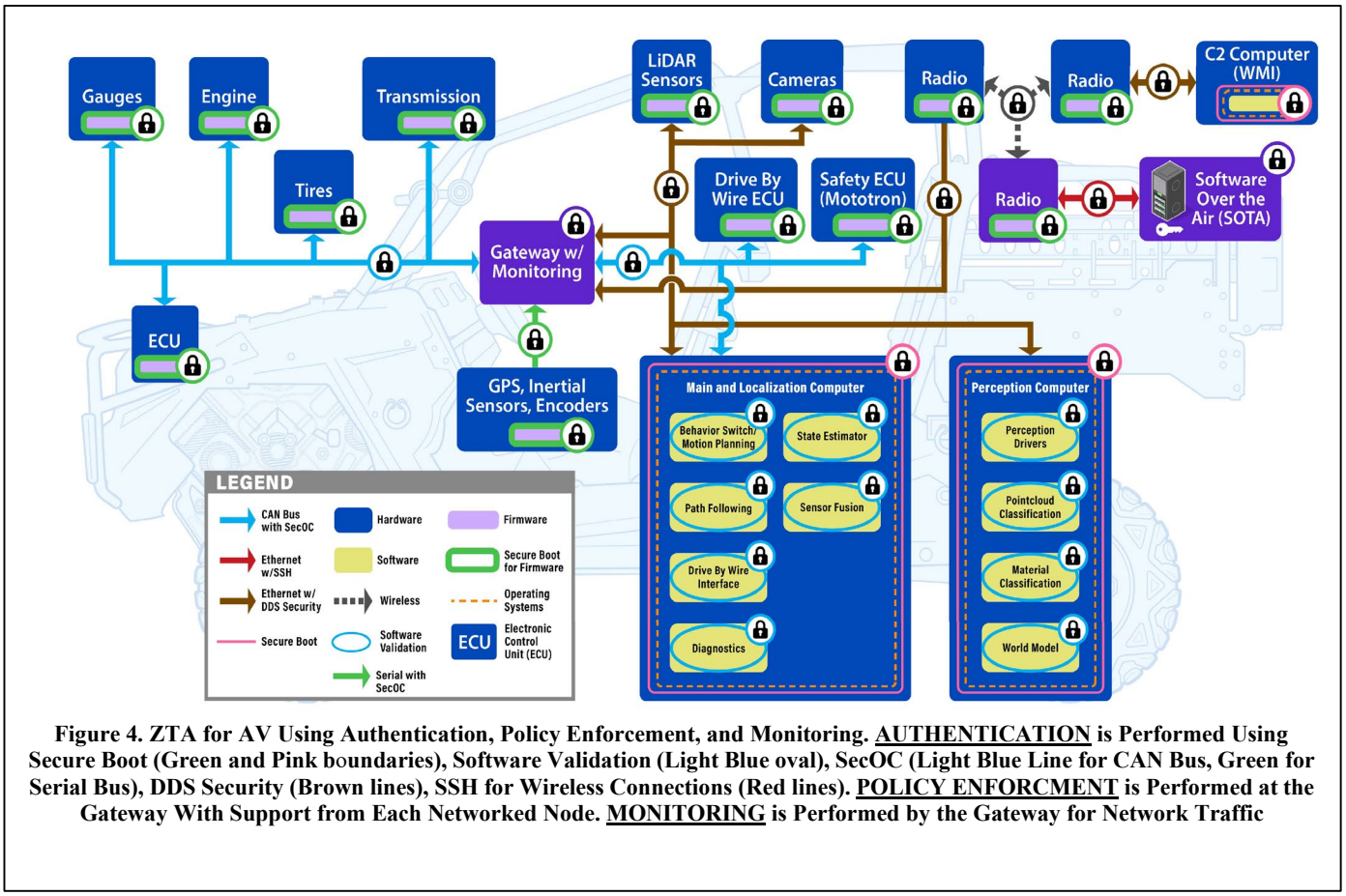

Figure 4. ZTA for AV Using Authentication, Policy Enforcement, and Monitoring. AUTHENTICATION is Performed Using Secure Boot (Green and Pink boundaries), Software Validation (Light Blue oval), SecOC (Light Blue Line for CAN Bus, Green for Serial Bus), DDS Security (Brown lines), SSH for Wireless Connections (Red lines). POLICY ENFORCMENT is Performed at the Gateway With Support from Each Networked Node. MONITORING is Performed by the Gateway for Network Traffic

5.1. Authentication NIST Principles (1) (2) (3) (4) (7)

Authentication requires subjects to prove their identity prior to accessing a resource. These requirements will vary depending on the communication bus and include the following:

- **Ethernet Communication – Authentication via DDS Security** [11]

In AVs, data is transmitted from one software process or electronic device to another typically using an ethernet connection. This enables the data throughput necessary for controlling the AV. To secure this communication, ZTA applies the DDS security model [11] to ROS2 messages. These messages are primarily implemented on the control computers. With DDS security enabled, ROS2 messages are encrypted and authenticated.

- **CAN bus and Serial Communication – Authentication via Secure Onboard Communication (SecOC)** [12]

SecOC [12] provides a framework for securing CAN bus communication. At its core, it relies on the use of freshness value (e.g., counter) and MACs to validate communication. This framework is used for securing CAN and serial busses.

- **Wireless Communication - Authentication via Secure Shell (SSH)**

SSH allows for secure, remote access of ethernet via Transmission Control Protocol (TCP). This protocol protects system access by requiring a strong username and password or securely generated certificates to authenticate remote connections. This is applied on top of DDS security.

- **Secure Boot and Software Validation – Authentication Using Hashing Algorithms SHA-2 or SHA-3**

When the AV is started, secure boot [13] ensures that software, firmware, and operating systems are verified. If any change is detected, the secure boot will prevent the system from running and communicating on the network.

- **Key Distribution, Management, and Revocation – Supports Secure Implementation of Authentication**

The use of keys, usernames, and password for authentication, encryption, MACs, and hashing rely on keeping keys and passwords secret. Keys are stored in protected memory locations and not stored as clear text.

5.2. Policy Enforcement Engine NIST Principle (5)

To control access, the policy explicitly defines which users need access to resources. The policy is enforced at the gateway and each node on the network.

- **Policy**. The policy clearly defines:
  a. Network users (subjects) and resources (topics and data sources).
  b. Expected network traffic including packet info, who sends it, who receives it, and allowable bounds as applicable.
- **Policy Enforcement**. The policy is actively enforced at each node and at the central gateway and is determined and enforced at runtime.
- **Gateway**. A network gateway provides a central hub for monitoring and enforcing the policy.

5.3. Monitoring NIST Principles (6) (8)

ZTA networks are monitored by the Gateway to ensure devices maintain ZT policy compliance. If a device is not in compliance with the policy, the network will reject data and requests associated with the non-compliance and log the circumstances for future improvement.

6. Implementing Zero Trust on CRASH

The CRASH program started in FY 2021 and, at the time this paper is written, is on-going. CRASH focuses on improving cybersecurity for AVs and hardening Robotic Autonomous Systems (RAS) to be resilient to cyber-attacks. CRASH builds on prior autonomy efforts and supports developers by simplifying the application of security. CRASH's five (5) focus areas cover:

Focus Area 1. Hardened Communication Interfaces. Defends against the threat of unwanted behavioral changes in RAS through hardened communication interfaces at radio touch points. While physical security is important, it does not directly apply to the ZTA for AV and is therefore not detailed further in this paper.

Focus Area 2. Robust RAS Access Control. This implements DDS security for ARCS which defends against unauthorized control of or service denial to RAS using robust mission-tailored access control.

Focus Area 3. Anomaly Detection Engine. Detects and responds to cyber-attacks using anomaly monitoring and the policy engine.

Focus Area 4. Secure-ARCS on seL4. Enhances content controls and process isolation using a secure implementation of ARCS on seL4.

Focus Area 5. Secure Software Update. Harden RAS against code and policy tampering through the automotive-grade secure software update mechanism.

Figure 5 shows how Focus Areas 1 – 5 overlays onto the AV architecture.

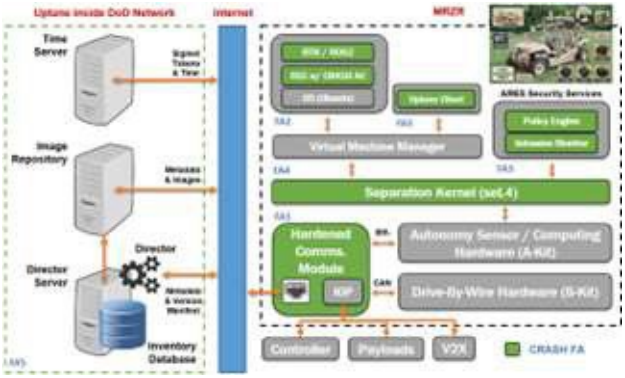

Figure 5. Overview of five (5) focus areas for CRASH.

Each focus area 2-5 improves AV security and supports implementation of ZTA for AVs. In addition to implementing components of the ZTA for AV, CRASH provides an extensible architecture that supports developers by giving them a place to put their code in a secure space. So rather than worrying about applying cybersecurity to the ARCS code, the developer writes code inside the hardened cybersecure infrastructure.

6.1. Robust RAS Access Control, Secure Communication Using DDS (Focus Area 2)

The Focus Area 2 team implemented DDS security mechanisms and containerization of code for portions of ARCS. The code is configured to support teleop of the AV. DDS-Security access control provides authentication of the network communication. This prevents subjects, ARCS publishers/subscribers from communicating on topics (i.e., resources) for which they are unauthorized.

The DDS access controls were implemented through an XML policy file and a ROS2 launch system design pattern. The XML policy declares the subject (e.g., gui_monitor, a ROS2 node/process) along with the topics (resources) for which that subject is allowed to publish and/or subscribe. In Figure 6, the subject node, *gui_monitor*, is only allowed to subscribe to the *planner_command* and *planner_pose* topics and is not allowed to publish or subscribe to any other topic in the system.

This policy provides a solid foundation for ZTA but is not sufficient as the access control policy cannot be changed dynamically at run-time based on factors such as threat intelligence, the compute asset's security posture, or the node/subject's current behavior.

Other forms of secure communication are implemented in the ARCS system. This includes DDS-Security encryption, which provides data integrity and confidentiality in transit.

```
<?xml version="1.0" encoding="UTF-8"?>
<policy version="0.2.0"
 xmlns:xi="http://www.w3.org/2001/XInclude">
 <enclaves>
  <enclave path="/lesson_4/gui_monitor">
   <profiles>
    <profile ns="/cyber_training" node="gui_monitor">
     <xi:include href="common/node.xml"
      xpointer="xpointer(/profile/*)"/>
     <topics subscribe="ALLOW">
      <topic>planner_command</topic>
      <topic>planner_pose</topic>
     </topics>
    </profile>
   </profiles>
  </enclave>
 </enclaves>
</policy>
```

Figure 6. Example DDS Security Access Control Policy. In this Example, gui_monitor is Only Allowed to Subscribe to planner_command and planner_pose

6.2. Anomaly Detection Engine (Focus Area 3)

Focus Area 3 notifies an operator when anomalous activity. The anomaly detection engine is rule based and violation of any rule, such as a process attempting to subscribe to data that it is not allowed to subscribe to, is flagged as anomalous. By continuously monitoring the AV network activity, it provides a sense normal data flow based on operational parameters. This combats the threat of attacks because the CRASH architecture considers both the user credentials and the user behavior to determine validity.

6.3. Process Isolation Using seL4 (Focus Area 4)

Focus Area 4 provides process isolation between virtual machines through the seL4 microkernel so that the effects of a cyber-attack do not propagate throughout the system. By limiting the maneuverability of an attacker on the AV, the system has better security assurances.

seL4 achieves process isolation by taking a radical approach to resource management: it never allocates memory. In other words, there is no heap. Rather, it makes resource management the responsibility of user-level processes (outside of the kernel). If a process requires additional kernel level memory, it must provide this memory explicitly to the kernel.

6.4. Secure Software Updates (Focus Area 5)

Focus Area 5 of CRASH delivers software updates securely, using a combination of public/private keys and meta data from servers to ensure software is authentic prior to installation. Updates to control computers are signed by developers, manufacturers, and the Army; packaged with the server's meta data including time and versioning information sent to the AV, verified, and then deployed. The different roles are visualized in Figure 7. The ability to update software, including policy, is required by the ZTA for AV.

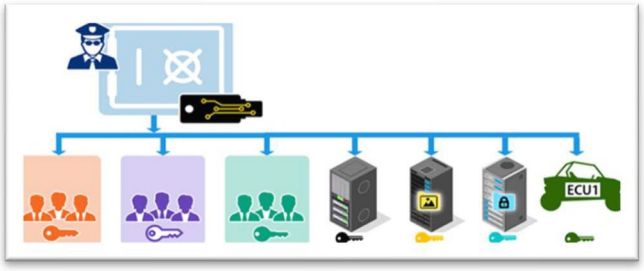

Figure 7. Different Roles in the Supply Chain use Keys to Approve Use of Software and Updates to the Security Policy. These Approvals are all Confirmed Prior to Installing Updates On-Vehicle

7. Related Efforts

Some development efforts, that are independent from CRASH, that relate to the ZTA were demonstrated including securing remote connectivity to ARCS, anomaly detection on automotive networks, and monitoring of AV sensors.

7.1. Remote Connectivity

When leveraging remote connectivity for ARCS, the radios are configured to implement these security features [15]:

1. **Encryption** - Symmetric Key encryption using 256-bit Advanced Encryption Standard (AES).
2. **Hashing** - Secure Hash Algorithm (SHA) – 256 for confirming data sent matches data received.
3. **Rekeying** - Supports wireless secure rekeying.

These security features are applied on top of others including DDS-Security.

7.2. Anomaly Detection

Anomaly detection systems have been built and demonstrated on CAN and ethernet networks.

7.2.1 CAN bus and Ethernet Anomaly Detection

While there are key differences in CAN bus and automotive ethernet communication, similar approaches were used to monitor these networks. Communication over both CAN and ethernet is looking for similar anomalies including packet injection, event simulation, and denial of service. Ethernet commonly has more bus traffic where CAN communication is highly reliable. Algorithms monitor vehicle networks using the following [16]:

- **Signature-based detection** - Uses characteristics of previously identified malicious packets to uncover anomalies. For example, unexpected packets, such as flooding a low Arb ID like 0x000 to cause denial of service leveraging CAN's packet prioritization, are flagged.
- **Anomaly-based** – Normal traffic is characterized and anything that does not match 'normal' is flagged. For example, timing of purely periodic packets are characterized and deviations are flagged.
- **State-based** – Adjusts anomaly detection based on the state of the vehicle. For example, a vehicle's communication profiles may change substantially from starting the engine to putting it into 'Drive.'

7.2.2 Physical Layer Monitoring

Through monitoring physical layer characteristics such as voltage levels, transitions, and bit spacing, broadcasting sources are identified. After characterizing broadcasting sources for each packet, nodes sending packets that are not allowed, for example when packets are spoofed by a compromised ECU, they are flagged as anomalous [17].

7.3. Sensors [18]

Input from sensors is key for localization, path following, object tracking and collision avoidance. Remote exploits have been demonstrated for all commonly used sensors including cameras, LiDAR, radar, ultrasonic, and GPS. Two approaches for improving resiliency of sensors were demonstrated:

- **Sensor Redundancy** – using multiple sensors for tracking similar areas. For example, using a long-range camera and long-range radar in front of the vehicle while having overlapped mid-range LiDAR and cameras. This way, if a single sensor has an error or is attacked, the remaining sensors can compensate. An example AV outline showing sensor redundancy is in Figure 8.
- **Sensor Fusion** [18] – Extracting similar features from different sensors, such as position, velocity, and acceleration of the vehicle and external objects, and comparing with other sensors in real time provides the ability to quickly identify issues with sensors. When a sensor has a problem, its readings can be removed at the software level, and the system continues to operate in a degraded state that allows the mission to complete.

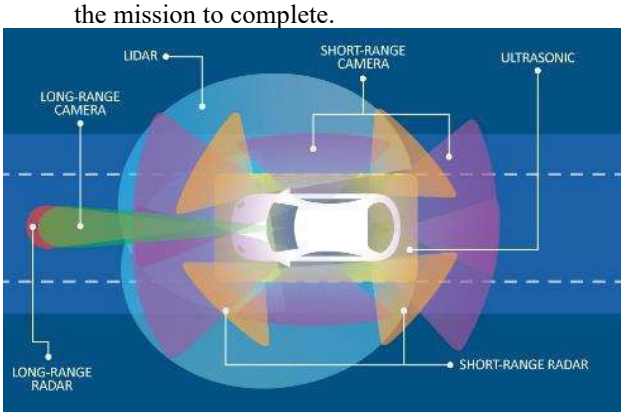

Figure 8. Sensor Redundancy Showing Overlapping of Similar and Different Sensor Types Tracking Similar Areas. This Greatly Improves Resiliency to a Single Sensor Fault and Remote Attacks

8. FUTURE WORK: IMPLEMENTING ZTA

Section 0 presented the ZTA for AV. Sections 6 and 7 detailed progress made towards implementing it, and this section addresses the question: What else is required for realizing the ZTA for AVs? Work is planned to expand current implementations of Authentication, Policy Enforcement, and Monitoring for the ZTA.

8.1. Authentication Updates

Authentication has been applied to portions of the AV environment; however, these mechanisms need to be extended and improved to align with the ZTA for AV.

8.1.1 DDS Security to Full AV

At the time this paper is being written, DDS security has been implemented on a portion of ARCS for teleop, or remote control operation. DDS security will be extended to the full ARCS code base.

8.1.2 Secure Boot and Software Validation

Secure boot is present in some automotive components and ensures authentic versions of software are running. However, secure boot is not commonly found on electronics used for autonomy such as the sensors, drive by wire, or control computers, and is not present on all automotive ECUs. Future efforts plan to implement secure boot on embedded devices and at the operating system level for control computers. Further, software validation via hashing will be added to the control software to prevent tampering.

8.1.3 Applying SecOC to CAN Bus

SecOC is being adopted by some commercial automotive companies to secure critical communication on the CAN bus and automotive ethernet. Currently, vehicles that are being automated commonly use unsecure forms of communication.

8.1.4 Key Distribution and Management

Some efforts, such as Secure Updates for CRASH focus area 5, have implemented key management processes. However, similar approaches need to be applied to DDS communication, software validation, SecOC communication, wireless communication, and secure boot processes.

8.1.5 Third Party ECUs

Security of third-party ECUs needs to be addressed and requirements clearly defined and implemented. This includes supporting secure boot, secure communication, and key distribution and management. Additionally, third party ECUs must monitor traffic and report anomalies and errors in communication when present.

8.2. Policy Enforcement and Updates

Static security policies have been implemented by CRASH Focus Area 2 (Access Control) with an initial policy monitoring system implemented by Focus Area 3 (Anomaly Detection) team members; however, a dynamic security policy for ground AV needs to be created. The policy will account for conditional authorization in the context of the ZTA for AV detailed in section 0. This includes:

- Defining how access control policies are verified and enforced at runtime.
- What access controls are non-negotiable and cannot be denied for safety and mission functionality reasons.
- Determine how threat intelligence, asset inventories and other types of data required for the policy engine's trustworthy algorithm are necessary to realize a runtime policy decision engine.

8.2.1 DDS Architecture for ARCS

Moving forward, the existing ARCS trust boundary analysis should be revisited to provide a granular view of topics across subsystems. Research and development to architect ARCS with multiple DDS domains and partitions coupled with the development of a dynamic policy engine to move ARCS towards the ZTA.

Remote Interfaces

The current approach to remote interfaces relies on SSH to secure the wireless network connection as well as symmetric key to further encrypt communication between radios. Part of the DDS architecture for ARCS must include process isolation of the C2 interface for AV control and apply a similar approach to software and policy updates. This strictly limits what information is passed to and from the wireless connection.

Gateways

Gateways will strictly control information that flows between different segments of the network (e.g., data packets moving from ethernet to CAN bus). Packets that are not explicitly allowed to pass are blocked.

8.3. Monitoring Updates

On-going projects handle monitoring of software and network traffic to varying degrees for individual pieces of the AV system. These disparate processes need to be unified and applied across the full vehicle network and reviewed and updated routinely to reduce the AV risk profile.

8.3.1 Full AV Monitoring

The control computers record ROS traffic in bag files which are helpful for debugging and troubleshooting. A higher-level monitoring process needs to be implemented that records the following for each bus (CAN, Serial, and Ethernet):

- **Errors** – Including errors generated by busses, software, and third party ECUs.
- **Anomalies** – When specific events of interest occur, log data is used to improve the AV performance.

8.3.2 Risk Management

A key tenant of cybersecurity is managing risk. Part of implementing the ZTA includes tracking the following:

- **Threats** – Entities, processes, or people that attempt to manipulate the AV.
- **Vulnerabilities** – A weakness in security that may be exploited by a threat.
- **Risks** – Risks occur when a threat exploits a vulnerability. Risks should be rated relative to each other to help with prioritizing efforts.

While a threat assessment has been performed on the AV, a process needs to be created that accounts for handling new threat information incorporated into and managed by the security framework. New threat information is often provided by outside entities; for example a new zero-day vulnerability is distributed to a system administrator.

9. CONCLUSION

This paper presents an approach to balance developer focused functionality with security through a ZTA for AV. This architecture leverages authentication, cyber policy enforcement, and monitoring to detect and mitigate cyber-attacks. It is traceable to NIST 800-207, and programs have implemented portions of the proposed ZTA. For example, the CRASH program has implemented authentication for ethernet communication through DDS, security policy enforcement for ARCS code base, process separation, monitoring communication, and securely updating software. Other programs have demonstrated monitoring automotive CAN and ethernet busses and improving resiliency through sensor redundancy and fusion. To fully implement the ZTA for AV, there are pieces that need to be addressed in future efforts including applying secure communication to the full AV, incorporating secure boot, developing/applying cyber policy to the full AV, and monitoring the full vehicle.

Realizing the ZTA for AV will take collaboration across many entities, broken down here into generalized groups:

1. **ARCS Developers** – Create and add functionality to the Army's autonomy software. Focus should be continuing to mature the ZTA for developers to incorporate security features more easily with minimal impact to coding of functionality. To implement the ZTA, the following are needed:
    a. Secure boot and software validation for all software components.
    b. Secure key distribution and management.
    c. Security policy development and enforcement.
    d. Update DDS architecture to use multiple nodes rather than having all communication on a single node. This step promises performance improvement as well as improved security posture.
    e. Use gateways to enforce security policy.
    f. Expand monitoring to full vehicle.
2. **Third Party Suppliers** – Build and manufacture systems that are used by Ground Vehicle Manufacturers and ARCS developers. For the reference system, this includes companies that make LiDAR, cameras, drive by wire, safety ECU, and GPS/IMU. To implement the ZTA, the following is needed:
    a. Secure boot for ECUs.
    b. Provide support for secure communications, including DDS Security and SecOC as applicable.
    c. Provide support for secure key distribution and management.
    d. Properly flag errors and notify the network.
3. **Ground Vehicle Manufacturers** – Design and build ground vehicles. To implement the ZTA, the following is needed:
    a. Secure boot for ECUs.
    b. Secure communication over CAN and automotive ethernet via SecOC.
    c. Secure key distribution and management.
    d. Security policy development and enforcement.

While implementing all the security features is recommended, it is prudent to apply a risk-based approach rather than all or nothing. In terms of priority, focus should be placed first on remote connectivity including the radios and C2 computer. Next priority will be everything the radio connects to, including sensors and control computers. Next, systems that directly impact the vehicle, including the safety, transmission, and engine ECUs. And so on.

Features implemented to date have substantially improved the security posture of AV. The ZTA for AV provides increased resiliency and minimizes the impact of an adversary.

REFERENCES

[1] ö. D. M. M. E. S. Sıddıka Berna örs Yalçın, "Designing and Implementing Secure Automotive Network for Autonomous Cars," in 29th Signal Procesing and Communications Applications Conference, 2021.

[2] S. K. L. S. M. a. B. M. Sanwald, "Secure Boot Revisited: Challenges for Secure Implementations in the Automotive Domain," in SAE International Journal of Transportation Cybersecurity and Privacy-V128-11EJ, 2020.

[3] J. B. Kristoffersson, "Zero Trust in Autonomous Vehicle Networks Utilizing Automotive Ethernet," in Chalmers University of Technology, Gothenburg, Sweden, 2022.

[4] Q. H. L. C. John Anderson, "A Zero-Trust Architecture for Connected and Autonomous Vehicles," in IEEE Internet Computing (Volume: 27, Issue: 5, Sept.-Oct. 2023), 2023.

[5] S. Rose, O. Borchert, S. Mitchell and S. Connely, "Zero Trust Architecture," NIST Special Publication 800-27, 2020.

[6] R. Freter, "Zero Trust Reference Architecture, Version 2.0.," Defense Information Sytem Agency (DISA) and the National Security Agency (NSA), 2022.

[7] Object Management Group, Inc. (OMG), "DDS Security Model," Object Management Group, Inc. (OMG), 2018.

[8] AUTOSAR, "Specification of Secure Onboard Communication," AUTOSAR, 2019.

[9] Microsoft , "Secure Boot," Microsoft Learn, 8 February 2023. [Online]. Available: https://learn.microsoft.com/en-us/windows-hardware/design/device-experiences/oem-secure-boot. [Accessed 2023].

[10] Persistent Systems, "MPU5 WR-5100," 2022. [Online]. Available: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://www.persistentsystems.com/site/wp-content/themes/persistensystems/pdf/mpu5/03EN070_MPU5_Spec_Sheet_Rev_Q.pdf.

[11] NDIA Michigan Chapter, "Automotive Ethernet Cyberattack Defense in Ground Vehicles," GVSETS, 2022.

[12] J. Wolford, C. Westrick and P. Moldenhauer, "Cyberattack Defense Through Digital Fingerprinting, Detection Algorithms, and Bus Segmentation in Ground Vehicles," NDIA Ground Vehicle Systems Engineering and Technology Symposium, 2021.

[13] R. McBee, J. Wolford and A. Garza, "Detection and Mitigation of Erroneous and Malicious Data in Vehicle Sensor Networks," NDIA Michigan Chapter Ground Vehicle Systems Engineering and Technology Symposium, 2022.