Master Thesis Project

vehicleLang's Assets & Attacks Overview

Firstly, a list of the assets that are relevant to security, in the domain of modern connected vehicles, will be presented.

Connected car/vehicle = is a car/vehicle that is equipped with Internet access, and usually also with a wireless local area network. Not to be confused with interconnected cars (V2X) or autonomous cars (level 4 or 5 of automation).

The high-level categories recognized are the following:

- **System** (including ECUs, GatewayECUs, embedded sensors, firmware and more)
- **Networking** (including the types of networks found on modern vehicles)
- **Communication** (including Dataflows, Data, EncryptedData and more)
- **Security** (including Vulnerabilities, Accounts, Credentials, IDPS and more)

Moving at a lower level, the child assets of the aforementioned parents are presented below. The ones that are in dark gray are assets that are not so important at the current phase, since **the initial target was to model as best as possible the internals of a vehicle** and if this was achieved then expand the coverage to other higher levels assets. Additionally, the assets that are underlined exist already in the current core language or need only minor modifications. Finally, the assets in light gray are related to V2X communications and are completely out of scope for this work.

- **Network**
  - Ethernet (normal Ethernet, 802.3)
  - VehicleNetwork
    - CANnetwork
      - J1939Network
    - FlexRayNetwork
    - LINnetwork
    - BroadR-Reach (100-BASE-T1)
  - WAN (Internet, needed for future expansions of my work)
  - Wi-Fi
  - Bluetooth
  - MOST (for infotainment system)
  - Power-line communication (CAN or LIN over power line DC-LIN and CD-BUS)
  - RF protocols (for keyless entry, DASH7 used for TPMS, key fobs)
  - Telematics

  Hardware ports & connections:
  - OBD-II port
  - Aftermarket dongles
  - Maintenance and diagnostic equipment

Inter-vehicle, or Vehicle-to-infrastructure wireless protocols (Ref. ENISA report):
- WAVE (Wireless Access in Vehicular Environments)/802.11p/ETSI ITS-G5
- DSRC (Dedicated Short-Range Communications)

- **Machine**
  - ECUs
    - Gateway ECUs
    - Ethernet Gateway ECUs

    Specific types of ECUs (But better not have so much detail here: generalization):
    - ARM platform, Power, SH, V850, Automotive MSP430 or TriCore)
  - Embedded sensors (sensors that are stand-alone not connected directly to an ECU and have their own firmware like on TPMS)
  - Infotainment System
  - Software (extends Machine)
    - Operating System (QNX, Linux, Windows CE, Green Hills or even Android for Infotainment System / OSEK or AUTOSAR for ECUs / BlackBerry QNX SDP 7.0 for the whole car!)
    - Firmware (running on either ECUs or sensors, could be based on OS asset but must be different asset)
    - Application
    - Service
      - NetworkService
      - TransmitterService
      - UDSService
    - Client
      - NetworkClient
      - VehicleNetworkReceiver

- **Communication**
  - Information
  - Data
    - CoreEncryptedData
  - Dataflow
    - ConnectionOrientedDataflow
    - ConnectionlessDataflow

- **Security**
  - Vulnerability
  - User
  - Account
  - Key/Credentials
  - MessageID
  - IDPS
  - Attacker (of course!)

❖ Covered Attacks List

Below follows a tabular overview **of all the attacks** and the corresponding assets that are currently modeled by **vehicleLang**.

Related Asset	Attack Name	Short Description	Possible Defense/Obstacle	Test Case	References
ECU	A1.1 Bypass Access Control	An attacker can bypass access control and authenticate to the machine through the diagnostics interface	None	TC 2	Core language
	A1.2 Denial of Service (DoS)	When an attacker performs a DoS attack to the ECU. This leads to DoS on the services running and deny of access to stored data. This can even lead to unresponsive ECU	None	TC 1, TC 2	Core language
	A1.3 Shutdown	When an attacker successfully switches off or takes offline a working ECU	None, but it requires significant effort	TC 10, TC 11	[1], [2]
	A1.4 Change operation mode	An attacker might put the ECU into diagnostics (if vehicle is moving slowly or is stopped) or even update mode (bootmode)	Prevent ECU from entering diagnostics mode after it started moving for first time. Allow diagnostics mode only after some physical change on car.	TC 18, TC 19, TC 20, TC 21	[1]
	A1.5 Bypass message confliction	An attacker can bypass message confliction protection mechanisms by changing ECU's operation mode (i.e. no conflicts) and achieve message injection	See A1.4	TC 21	[1]
	A1.6 (Service) Message injection	An attacker can inject forged messages, to the services that are running on this ECU, that could for example notify about vehicle's fault or report fake status (speed, operation mode, etc.). This can even lead to unresponsive ECU/DoS attack	Communication protection (authorization and/or authentication), message confliction mechanism, IDPS.	TC 7, TC 26	[3]

Related Asset	Attack Name	Short Description	Possible Defense/Obstacle	Test Case	References
ECU (cont.)	A1.7 Firmware modification	An attacker could upload a custom firmware on the target ECU so that he can gain full access on the ECU and maybe on the connected network	Firmware validation/signing mechanisms that prevent custom firmware uploads	TC 22, TC 23, TC 24, TC 31	[1]
Gateway ECU	A2.1 Bypass Firewall	If the firewall is disabled, then the attacker can bypass it	Enable firewall	TC 34	Core language
	A2.2 Bypass IDPS	If firewall is disabled but IDPS is not, then the attacker can attempt to bypass the IDPS by carefully injecting messages to the network	Enable IDPS, and in general bypass requires some effort from the attacker	TC 34	[3], [4], [5]
	A2.3 Denial of Service	An attacker can perform denial of service attack on the connected networks	Same as on ECU	TC 6	Core language
Vehicle Network	A3.1 Eavesdrop	An attacker can eavesdrop the network and the transmitted dataflows	None	TC 8	[6], [7]
	A3.2 Man in the Middle (MitM)	An attacker can intercept and tamper with the network's communications	None	TC 8	[6]
	A3.3 Denial of Service (DoS)	An attacker can perform a DoS attack to the network	Attacker must have compromised the Gateway ECU	TC 10, TC 11, TC 12, TC 25, TC 34	[2], [8]
	A3.4 Message Injection	An attacker can inject forged messages to the network, that for example could notify about vehicle's fault or report fake status (speed, operation mode, etc.)	Attacker must have physical access to the network under attack	TC 26, TC 26, TC 29, TC 32, TC 37	[1], [6], [9], [10]
CAN Network	A4.1 Exploit CAN's arbitration mechanism	By exploiting the arbitration mechanism for message prioritization in CAN bus, an attacker could achieve invalidation of legitimate messages and/or message tampering	Requires network access plus effort is needed from the attacker	TC 10	[11], [1], [12]

Related Asset	Attack Name	Short Description	Possible Defense/Obstacle	Test Case	References
	A4.2 Bus-off attack	An attacker can exploit the CAN's error-handling scheme to disconnect or shut down good/uncompromised ECUs	The defense proposed on literature, plus effort is needed from the attacker	TC 10	[2]
FlexRay Network	A5.1 Common time base attack	An attacker that sends more than needed SYNC messages within one communication cycle can make the whole network inoperable	None, but effort is needed from the attacker	TC 11	[13]
	A5.2 Exploit FlexRay's Bus Guardian	An attacker can utilize Bus Guardian to send well-directed faked error messages to deactivate controllers. However, Bus Guardian is hardened so much effort is needed	None, but effort is needed from the attacker	TC 11	[13], [14]
	A5.3 Sleep frame attack	An attacker can send well-directed forged sleep frames to deactivate power-saving capable FlexRay controllers	Power-saving features can be disabled, plus effort is needed from the attacker	TC 11	[13]
LIN (Sub)-Network	A6.1 Inject bogus sync bytes	An attacker that sends frames with bogus synchronization bytes within the SYNCH field can make the local LIN network inoperative or cause at least serious malfunctions	None, but effort is needed from the attacker	TC 12	[13]
	A6.2 Gain LIN access from CAN	An attacker can easily gain access to the LIN subnetwork through a CAN-bus node	The CAN-bus ECU must first be compromised	TC 12	[15], [16]
	A6.3 Inject header or timed response	An attacker can exploit the error handling mechanism of LIN bus to inject forged headers or messages to the network, but in general it is not so easy	The defense proposed in literature, plus effort is needed from the attacker	TC 12	[17]

Related Asset	Attack Name	Short Description	Possible Defense/Obstacle	Test Case	References
J1939 Network	A7.1 Message Injection	An attacker can make requests towards other J1939 nodes or PGNs (Parameter Group Number) and after effort to forge malicious responds	Attacker must have physical access to the network layer	TC 39	[11]
Ethernet Network	A8.1 Bypass port security / access control	An attacker can bypass the port security which restricts a port's ingress traffic by limiting the MAC addresses that can send traffic into that port. This is a well-known attack.	Enable port security	-	Core language
	A8.2 ARP cache poisoning	The well-known ARP spoofing attack	Usage of static ARP tables	-	Core language
Ethernet Network (cont.)	A8.3 Eavesdrop	An attacker can eavesdrop the network and the transmitted dataflows	None	-	Core language
	A8.4 Man in the Middle (MitM)	An attacker can intercept and tamper with the network's communications	None	-	Core language
	A8.5 Denial of Service (DoS)	An attacker can perform a DoS attack to the network	Attacker must have compromised the Ethernet Gateway ECU or the router	TC 40	Core language
Ethernet Gateway ECU	A9.1 Bypass Firewall	If the firewall is disabled, then the attacker can bypass it	Enable firewall	-	Core language
	A9.2 Denial of Service	An attacker can perform denial of service attack on the connected networks	Same as on ECU	-	Core language
UDS Service	A10.1 Access	An attacker that gains access on an UDS service can access stored ECU data, has the possibility to update the firmware and change the operation status of the ECU.	None, but access on the service must first be compromised	TC 45	[18], [19]
Message ID	A11.1 Read Message ID	An attacker that eavesdrops (a dataflow or the network) or knows the unique ID of a message on a bus network can use it to forge messages that	Only the encryption of the ID	TC 46	[1]

Related Asset	Attack Name	Short Description	Possible Defense/Obstacle	Test Case	References
		can be used to gain access to receiving ECUs and their connected physical machines			
Firmware	A12.1 Bypass firmware validation	An attacker can bypass the validation of a firmware update easily if no validation functionality is present	Code signing and verification during upload, use of strong checksum functions and/or don't distribute the private keys for signing	TC 22, TC 23, TC 24, TC 31, TC 46	[1]
	A12.2 Crack firmware validation	If, however, firmware validation is enabled, the attacker can try to crack it. This of course will require a large amount of time	None	-	[1]
Connection Oriented Dataflow	A13.1 Eavesdrop	An attacker that eavesdrops on the dataflow, can access the contained data if they are unencrypted	Encryption of the data	-	Core language
	A13.2 Denial of Service (DoS)	An attacker that performs a denial-of-service-attack on the dataflow makes the contained data inaccessible	None	-	Core language
	A13.3 Man in the Middle (MitM)	An attacker that man-in-the-middles the dataflow, can control the contained data, perform requests and responds and even DoS	None	TC 14	Core language
	A13.4 Malicious Respond	An attacker can try to perform a malicious respond on a dataflow (on the clients connected to it). The confliction protection mechanism does not completely prevent malicious responds, but bypass takes time	None	TC 39	-
	A13.5 Malicious Request	An attacker can send malicious requests to the service connected to the dataflow	None	TC 41	Core language

Related Asset	Attack Name	Short Description	Possible Defense/Obstacle	Test Case	References
Connection-less Dataflow	A14.1 Same as A13.1-A13.3	-	-	TC 15	Core language
	A14.2 Malicious Transmit	An attacker can try to maliciously transmit a message. This can happen only when IDPS is not in place	The presence of an IDPS	TC 25, TC 26, TC 27, TC 28, TC 38	-
Infotainment System	A15.1 Gain Network Access	An attacker that has compromised the infotainment system can exploit the network access service to gain network access	None	TC 47	[20]
	A15.2 Engineer Network Access	An attacker can engineer a custom service and maybe a custom firmware that will allow network access via a compromised infotainment system	None, but effort is needed from the attacker	TC 48	[21]

End of table

It must be noted that child assets, such as CAN Network, FlexRay Network, LIN Network and J1939 Network, inherit also all the attacks that can happen on its parent assets, in this example the Vehicle Network asset.

Additionally, where “Core language” is used as a reference it means that those attacks were inspired or taken as is from the core modeling language on which the vehicle modeling language is building on top.

❖ Assets & Attack Steps List

Next, follows a complete tabular overview of the **all attack steps and their corresponding assets**, as they are modeled by the current version of the vehicle modeling language, called **vehicleLang**. All the attack steps in gray background are inherited from parent asset, while all the others are asset specific attack steps.

Related Asset	Attack Step Name	Short Description	Possible Defense/Obstacle	Test Case	References
ECU	(1.1) Connect	*Inherits from Machine asset*. An attempt to connect to an ECU. This allows additionally to an attacker to attempt change operation mode (1.8) and firmware modification (3.3)	None	TC 1, TC 18, TC 19	Core language
	(1.2) Authenticate	If connected (1.1) and has the proper credentials, an attacker can try to authenticate to the ECU	Attacker must first connect	TC 1	Core language
	(1.3) Bypass Access Control	An attacker can bypass access control and authenticate immediately to the machine	None	TC 2	Core language
	(1.4) Access	*Inherits from Machine asset*. The result of a successful authentication or bypass of authentication. This allows additionally to an attacker to access the connected sensors/actuators, change operation mode (1.7), gain LIN access from CAN (1.11), firmware modification (1.14) and bypass message confliction (1.9)	None	TC 1, TC 20, TC 21	Core language
	(1.5) Denial of Service (DoS)	When an attacker performs a DoS attack to the ECU. This leads to DoS on the services running and deny of access to stored data. This can even lead to unresponsive ECU	Attacker must have access on the target ECU	TC 1, TC 2	Core language

Related Asset	Attack Step Name	Short Description	Possible Defense/Obstacle	Test Case	References
	(1.6) Shutdown	When the ECU is taken offline by some other attack step. This leads to bypass message confliction (1.9) and DoS (1.5)	Defenses on the attack steps that are parent to this attack step	TC 10, TC 11	[1], [2]
	(1.7) Change operation mode	Put the ECU into diagnostics (if vehicle is moving slowly or is stopped) or even update mode (bootmode). This leads to (1.9)	Operation Mode Protection: Prevent ECU from entering diagnostics mode after it started moving for first time. Allow diagnostics mode only after some physical change on car	TC 18, TC 19, TC 20, TC 21	[1]
	(1.8) Attempt change operation mode	Bypass enabled “operation mode protection” and achieve (1.9) after some effort	None, just luck because effort is needed from the attacker	TC 18, TC 19, TC 20, TC 21	[1]
	(1.9) Bypass message confliction	Bypass message confliction protection mechanisms by changing ECU's operation mode (i.e. no conflicts) which leads to service message injection (11.2)	Defenses on the attack steps that are parent to this attack step	TC 21	[1]
	(1.10) Service message injection	Moved to (11.2)			
	(1.11) Gain LIN access from CAN	This attack allows an attacker to easily gain access to the LIN bus (subnet) through a CAN-bus node	None	TC 18, TC 19, TC 20, TC 21	[15], [16]
	(1.12) Pass firmware validation	If the firmware validation key is stored on ECU, this attack step leads to firmware upload (1.14)	Avoid storing secrets in the ECU	TC 22, TC 23, TC 24	[1]
	(1.13) Malicious firmware upload	Maliciously upload a forged firmware leads to full access on the ECU and ability to inject messages on the previous running services and on network	If firmware validation is enabled, then it requires a huge amount of effort from the attacker	TC 22, TC 23, TC 24, TC 31	[1]
	(1.14) Upload firmware	Updating the firmware leads to the ability to inject messages on the network (4.8)	Happens only if (1.12) is compromised	TC 22, TC 23, TC 24	-

Related Asset	Attack Step Name	Short Description	Possible Defense/Obstacle	Test Case	References
	(1.15) Network service message injection	This is the same as (11.2) but this is only reached from an accessed vehicle network	Same as (11.2)	TC 7, TC 26	[3]
	(1.16) Access data	This helper attack step should work as an intermediate step to reach request access on data stored on a machine/ECU from UDSService	Attacker must have access to the ECU	TC 45	[18], [19]
	(1.17) Firmware Upload Network Access	This a helper attack step because both (1.13) and (1.14) are leading to the same attack step connections	Same as on (1.13) and (1.14)	TC 22, TC 23, TC 24, TC 31	-
	(1.18) ID Access	This attack step is reached after the ID is compromised from data/dataflow and allows an attacker to connect to the ECU's connected physical machines.	Read of the "secret" ID residing on the dataflow is needed first	TC 46	-
Gateway ECU	(2.1) The same attack steps as on ECU: (1.1) – (1.3)				
	(2.2) Access	Inherits from (1.4) but includes additionally the ability of the attacker to perform Man in the Middle (MitM) attack on the connected networks or to perform "forwarding" (2.3)	Attacker must first connect	TC 6	Core language
	(2.3) Forwarding	Forwarding is the lightest interaction with the gateway ECU, where the gateway simply retransmits received messages to other networks. This leads to (2.4).	Attacker must have access to the Gateway ECU	TC 6, TC 34	Core language
	(2.4) Bypass Firewall	If firewall is disabled, then attacker can bypass it and perform message injection on connected networks	"Firewall protection" defense	TC 34	Core language

Related Asset	Attack Step Name	Short Description	Possible Defense/Obstacle	Test Case	References
	(2.5) Denial of Service	Perform denial of service attack on the connected networks. This leads to (4.5)	None	TC 6	Core language
	(2.6) Gateway bypass IDPS	If Firewall is disabled and (2.4) is compromised, then try to bypass the IDPS system. This leads to (4.9)	"Firewall protection" defense	TC 34	[3], [4], [5]
	(2.7) Gateway no IDPS	If Firewall is disabled and (2.4) is compromised and IDPS is disabled, then the attacker can access the network unrestricted. This leads to (4.10)	"Firewall protection" defense	TC 34	[3], [4], [5]
Firmware	(3.1) The same attack steps as on ECU: (1.1) – (1.3)&(1.5)				
	(3.2) Malicious firmware modification	Validate a custom firmware update either by cracking (3.3) (brute forcing) the checksum or worse, by exploiting the absence of verification (3.4)	None, but cracking it will require a huge amount of time, see (3.3)	TC 22, TC 23, TC 24	[1]
	(3.3) Crack firmware validation	Crack firmware validation if it is enabled. This leads to (1.13)	None, but cracking it will require a huge amount of time	TC 22, TC 23, TC 24	-
	(3.4) Bypass firmware validation	Bypass firmware validation if it is not enabled. This leads to (1.13)	"Firmware validation" defense	TC 22, TC 23, TC 24	-
Vehicle Network	(4.1) Physical access	Attacker has physical access to the network. This allows him to access the network layer (4.2)	None	TC 10, TC 11, TC 12, TC 25, TC 34	Core language
	(4.2) Access network layer	*Inherits from Network asset*. Access implies the possibility to submit packets over the network. On a vehicle network, it allows the attacker to perform forwarding (2.3), DoS (4.5), message injection (4.8), network specific attacks (4.6),	None	TC 10, TC 11, TC 12, TC 25, TC 34	Core language

Related Asset	Attack Step Name	Short Description	Possible Defense/Obstacle	Test Case	References
		connect to network ECUs (1.1) and network service message injection (1.15)			
	(4.3) Eavesdrop	Attacker can eavesdrop the network and the transmitted dataflows	None	TC 8	[6]
	(4.4) Man in the Middle (MitM)	Attacker can intercept and tamper with the network's communications (dataflows) and also access the network layer. Additionally, on a vehicle network, the attacker can perform DoS on the gateway ECU (2.5)	None	TC 8	[6]
	(4.5) Denial of Service (DoS)	When an attacker performs a DoS attack to the network. This leads to DoS on the dataflows transmitted over the network	Attacker must have compromised the Gateway ECU	TC 10, TC 11, TC 12, TC 25, TC 34	[2], [8]
	(4.6) Network specific attack	This helper attack step should work as an intermediate step to reach network specific attacks.	None	-	-
	(4.7) Network forwarding	An attacker that has access (1.4) to a network connected ECU can also perform forwarding (2.3) on that network using the connected GatewayECU	None	TC 26	-
	(4.8) Message Injection	Inject forged messages to dataflows, that for example could notify about vehicle's fault or report fake status (speed, operation mode, etc.)	Attacker must have access to the network layer	TC 26, TC 27, TC 29, TC 31, TC 32, TC 37	[1], [6], [9], [10]
	(4.9) Bypass IDPS	Bypass IDPS on dataflows when the GatewayECU has IDPS enabled (2.6). This is a helper attack step	Defenses on the attack steps that are parent to this attack step. See (2.6)	TC 34	-

Related Asset	Attack Step Name	Short Description	Possible Defense/Obstacle	Test Case	References
	(4.10) No IDPS	This is reached when no IDPS is connected/present on the GatewayECU (2.7). This is a helper attack step	Defenses on the attack steps that are parent to this attack step. See (2.7)	TC 34	-
	(4.11) Gain LIN access from CAN	This is an empty attack that will only be implemented on LINnetwork (7.4) and it will be invoked by the ECU (1.11)	Happens only if (1.11) on ECU is compromised	TC 37	[15], [16]
CAN Network	(5.1) The same as on Vehicle Network				
	(5.2) Network specific attack	*Inherits from (4.6)* and includes all the following attacks	-	-	-
	(5.3) Exploit (CAN's) arbitration (block messages)	Exploiting the arbitration mechanism for message prioritization in CAN bus can lead to invalidation of legitimate messages and allow message/dataflow tampering	Requires network access plus effort is needed from the attacker	TC 10	[11], [1], [12], [22]
	(5.4) Bus-off attack	Exploits the error-handling scheme of in-vehicle networks to disconnect or shut down good/uncompromised ECUs (1.6) or to cause a DoS attack on the network (4.5).	"Bus off protection" defense plus little effort is needed from the attacker	TC 10	[2]
FlexRay Network	(6.1) The same as on Vehicle Network				
	(6.2) Network specific attack	*Inherits from (4.6)* and includes all the following attacks	-	-	-
	(6.3) Common time base attack	Send more than needed SYNC messages within one communication cycle to make the whole network inoperable. This leads to (4.5)	None, but effort is needed from the attacker	TC 11	[13]

Related Asset	Attack Step Name	Short Description	Possible Defense/Obstacle	Test Case	References
	(6.4) Exploit (FlexRay's) Bus Guardian	Utilize Bus Guardian for sending well-directed faked error messages to deactivate controllers. Bus Guardian is hardened so much effort is needed. This leads to (1.6)	None, but effort is needed from the attacker	TC 11	[13], [14]
	(6.5) Sleep frame attack	Send well-directed forged sleep frames to deactivate power-saving capable FlexRay controller. This leads to (1.6)	"Power saving incapable nodes" defense plus effort is needed from the attacker	TC 11	[13]
LIN Network	(7.1) The same as on Vehicle Network				
	(7.2) Network specific attack	*Inherits from (4.6)* and includes all the following attacks	-	-	-
	(7.3) Inject bogus sync bytes	Sending frames with bogus synchronization bytes within the SYNCH field makes the local LIN network inoperative or causes at least serious malfunctions. This leads to (4.5)	None, but effort is needed from the attacker	TC 12	[13]
	(7.4) Gain LIN access from CAN	*Inherits from (4.11)*. There are techniques that make it easy to gain access to the LIN bus through a CAN-bus node. This leads to (4.2)	Happens only if (1.11) on ECU is compromised	TC 12	[15], [16]
	(7.5) Inject header or timed response	This is a specific attack that can happen on LIN bus exploiting the error handling mechanism, but it is not so easy. This leads to tampering of dataflows	"Header or timed response protection" defense plus effort is needed from the attacker	TC 12	[17]
J1939 Network	(8.1) The same as on CAN Network				
	(8.2) Access network layer	*Overriding from VehicleNetwork asset*. Access implies the possibility to submit packets over the	None	TC 39	[11]

Related Asset	Attack Step Name	Short Description	Possible Defense/Obstacle	Test Case	References
		network. On a J1939 network, it allows the attacker to perform DoS (4.5), eavesdrop (4.3), message injection (4.8), and to connect to network ECUs (1.1)			
	(8.3) J1939 message injection	Inject messages to a J1939 network means that the attacker can make requests towards other J1939 nodes or PGNs (Parameter Group Number) and after effort to maliciously respond	Same as (8.5)	TC 39	[11]
	(8.4) J1939 attacks	This attack step should work as an intermediate step to reach J1939 network specific attacks from (4.2)	-	TC 39	[11]
	(8.5) Advanced J1939 attacks	If only limited parts of the J1939 protocol are used, then the J1939 specific attacks might not work. For example, requests might not be supported. This is a helper attack step	This attack can only happen if all the operations of the J1939 network are supported. See “noFullJ1939Support” defense.	TC 39	[11]
Connection Oriented Dataflow	(9.1) Eavesdrop	An attacker that eavesdrops on the dataflow, can read the contained data if they are unencrypted	Encryption of the data	-	Core language
	(9.2) Denial of Service (DoS)	An attacker that performs a denial-of-service-attack on the dataflow makes the contained data inaccessible (delete)	None	-	Core language
	(9.3) Man in the Middle (MitM)	An attacker that man-in-the-middles the dataflow, can eavesdrop (9.1), control (read, write or delete) the contained data, perform requests (9.4) and responds (9.5) and even DoS attacks (9.2)	None	TC 14	Core language
	(9.4) Request	When a typical/direct request to the connected service is performed	None	-	Core language

Related Asset	Attack Step Name	Short Description	Possible Defense/Obstacle	Test Case	References
	(9.5) Respond	When a typical/direct respond to the connected clients is performed	None	-	Core language
	(9.6) Malicious Respond	When an attacker maliciously responds to the connected clients. The confliction protection mechanism does not prevent malicious responds, but it typically takes time for the attacker to bypass it	None	TC 39	-
Connection-less Dataflow	(10.1) Same as 9.1 – 9.3				
	(10.2) Transmit	When a typical/direct transmission is performed from a transmitter service	None	-	-
	(10.3) Malicious Transmit No IDPS	Perform a malicious transmission when IDPS is disabled on Gateway ECU. However even when IDPS is disabled effort is needed to bypass message confliction mechanism	IDPS protection	TC 34	-
	(10.4) Malicious Transmit Bypass Confliction Protection	Confliction protection mechanism does not prevent malicious transmissions, but it typically takes time for the attacker to bypass it	None	TC 25, TC 31	-
	(10.5) Malicious Transmit Bypass IDPS	IDPS does not prevent all malicious transmissions, and it typically takes time for the attacker to bypass it	Firewall protection	TC 34	-
	(10.6) Malicious Transmit	The act of trying to maliciously transmit. This happens when IDPS is not in place, so the attacker can make malicious transmissions unobstructed	IDPS protection	TC 26, TC 27, TC 38	-

Related Asset	Attack Step Name	Short Description	Possible Defense/Obstacle	Test Case	References
Transmitter Service	(11.1) The same attack steps as on ECU: (1.1) – (1.3)&(1.5)				
	(11.2) Service message injection	Inject forged messages, to the services that are running on this ECU, that could for example notify about vehicle's fault or report fake status (speed, operation mode, etc.)	Communication protection (authorization, authentication ref. paper), Message confliction mechanisms. Some message injection defense mechanisms are: "message confliction protection" on ECU and IDPS on gatewayECU	TC 7, TC 26, TC 27	[3]
Ethernet Network	(12.1) Physical access	Attacker has physical access to the network. This allows him to bypass the port security (9.2)	None	-	Core language
	(12.2) Bypass port security	Port security can be used to restrict a port's ingress traffic by limiting the MAC addresses that can send traffic into the port. An attacker that has physical access can bypass it if disabled.	"portSecurity" defense	-	Core language
	(12.3) Bypass access control		Same as on (9.2)	-	Core language
	(12.4) Access data link layer	If access control is bypassed, then access on the data link layer (OSI layer 2) is gained. This leads to access on the network layer (9.5) (OSI layer 3) and to ARP cache poisoning attacks (9.X).	Same as on (9.2)	-	Core language
	(12.5) Access network layer	*Inherits from Network asset*. Access implies the possibility to submit packets over the network. On a vehicle network, it allows the attacker to perform forwarding (2.3), DoS (4.5), message injection (4.8), network specific attacks (4.6),	None	-	Core language

Related Asset	Attack Step Name	Short Description	Possible Defense/Obstacle	Test Case	References
		connect to network ECUs (1.1) and network service message injection (1.15)			
	(12.6) ARP cache poisoning	ARP spoofing works on all common IPv4 networks, both wirebound and wireless if the don't have static ARP tables	The "staticARPTables" defense should be disabled	-	Core language
	(12.7) Eavesdrop	Attacker can eavesdrop the network and the transmitted dataflows	None	-	Core language
	(12.8) Man in the Middle (MitM)	Attacker can intercept and tamper with the network's communications (dataflows) and also access the data link layer. Additionally, on an Ethernet network, the attacker can perform DoS on the routers and on the Ethernet gateway ECUs	None	TC 44	Core language
	(12.9) Denial of Service (DoS)	When an attacker performs a DoS attack to the network. This leads to DoS on the dataflows transmitted over the network	Attacker must have compromised the Gateway ECU	-	Core language
Ethernet Gateway ECU	(13.1) The same attack steps as on ECU: (1.1) – (1.3)				
	(13.2) Access	Inherits from (1.4) but includes additionally the ability of the attacker to perform Man in the Middle (MitM) attack on both the connected vehicle and ethernet networks or to perform "forwarding" (13.3)	Attacker must first connect	TC 35	Core language
	(13.3) Forwarding	Forwarding is the lightest interaction with the gateway ECU, where the gateway simply retransmits received messages to other networks. This leads to (13.4).	Attacker must have access to the Ethernet Gateway ECU	TC 35	Core language

Related Asset	Attack Step Name	Short Description	Possible Defense/Obstacle	Test Case	References
	(13.4) Bypass Firewall	If firewall is disabled, then attacker can bypass it and gain network layer access on connected networks	"Firewall protection" defense	TC 35	Core language
	(13.5) Denial of Service	Perform denial of service attack on both the connected vehicle and ethernet networks	None	TC 35	Core language

End of table

Implemented test (use) cases list

Test Class “CoreMachineTest”:

TC 1. testMachineAccess
TC 2. testBypassMachineAccess
TC 3. testSoftwareHostToGuest
TC 4. testSoftwareGuestToHost
TC 5. testMachineAccountDataRWD

Test Class “CoreVehicleNetworkTest”:

TC 6. testGatewayECUAccess
TC 7. simpleServiceMessageInjection
TC 8. testMitmNetwork
TC 9. testPhysicalAccess
TC 10. testCANnetworkSpecificAttacks
TC 11. testFlexNetworkSpecificAttacks
TC 12. testLINnetworkSpecificAttacks

Test Class “CoreDataTest”:

TC 13. testDataAccess
TC 14. testDataflow1DataAccess
TC 15. testDataflow2DataAccess

Test Class “CoreVulnerabilityTest”:

TC 16. testVulnerability
TC 17. testSoftware

Test Class “CoreEcuTest”:

TC 18. testConnectEcuAttacks
TC 19. testConnectEcuAttacks2
TC 20. testAccessEcuAttacks
TC 21. testAccessEcuAttacks2

Test Class “CoreFirmwareTest”:

TC 22. testFirmwareValidation
TC 23. testFirmwareValidation2
TC 24. testBypassFirmwareValidation

Test Class “MessageInjectionTest”:

TC 25. testNetworkMessageInjection
TC 26. testServiceMessageInjectionConflictProtect
TC 27. testServiceMessageInjectionNoConflictProtect
TC 28. testServicekMessageInjectionFromECU

TC 29. testNetworkMessageInjectionAfterVuln
TC 30. testNetworkMessageInjectionAfterConnectionVuln
TC 31. testNetworkMessageInjectionAfterFirmwareUpload
TC 32. testProtectedNetworkMessageInjection
TC 33. testSeeminglyProtectedNetworkMessageInjection

Test Class "AdvancedNetworkTest":

TC 34. testDataflowWithFirewallAndIDPS
TC 35. testAccessEthGatewayECU
TC 36. testNetworkAccessWithEthGatewayECU
TC 37. testGainLINaccess
TC 38. paperTest
TC 39. testJ1393Network1

Test Class "CoreEthernetNetworkTest":

TC 40. testRouterAccess
TC 41. testDataflow1
TC 42. testMultiDataflowRequest
TC 43. testMultiDataflowResponse
TC 44. testMitmNetwork1

Test Class "AdvancedVulnerabilityTest":

TC 45. testVulnerabilityOnUDS

Test Class "newTest":

TC 46. acceleratorTest

Test Class "InfotainmentTest":

TC 47. NetworkAccessFromInfotainmentTest
TC 48. EngineerNetworkAccessFromInfotainmentTest

Assumptions done for the above design:

For **ECU.serviceMessageInjection**: The assumption here is that the message confliction protection resides on the ECU that offers the respective service/transmitter and not on the receiving ECUs. This is done because it is easier to model and configure it that way. So, if for example the ABS ECU has message confliction protection enabled, then the maliciousTransmit for the ABS Service will be harder to perform! Additionally, this protection can be considered, conceptually, as a host-based IDPS system. When the message confliction protection is disabled, an attacker can exploit it also by having access on the network, through the _networkServiceMessageInjection attack step.

IDPS is modeled like a centralized inline IDPS located on the gateway ECU.

For **LIN.injectHeaderOrTimedResponse**: This can happen either by injecting response after header is sent OR by injecting false header and then response.

Where the term "**effort**" is used, a probabilistic distribution is assumed.

Finally, "**helper attack steps**" are attack steps that will not be visible to the end user (modeler) and are only there to help in the connection of attacks over different assets.

References

[1] Charlie Miller and Chris Valasek, «Remote Exploitation of an Unaltered Passenger Vehicle,» 2015.

[2] Kyong-Tak Cho and Kang G. Shin, «Error Handling of In-vehicle Networks Makes Them Vulnerable,» 2016.

[3] Pierre Kleberger, Tomas Olovsson, and Erland Jonsson, «Security Aspects of the In-Vehicle Network in the Connected Car,» 2011.

[4] Jeremy Daily et al., «Towards a Cyber Assurance Testbed for Heavy Vehicle Electronic Controls,» 2016.

[5] U. E. Larson and D. K. Nilsson, «Securing vehicles against cyber attacks,» 2008.

[6] Jonathan Petit and Steven E. Shladover, «Potential Cyberattacks on Automated Vehicles,» 2015.

[7] Luca Dariz, Massimiliano Ruggeri, Gianpiero Costantino and Fabio Martinelli, «A Survey over Low-Level Security Issues in Heavy Duty Vehicles,» 2016.

[8] Subhojeet Mukherjee et al., «Practical DoS Attacks on Embedded Networks in Commercial Vehicles,» 2016.

[9] Charlie Miller and Chris Valasek, «CAN message injection,» 2016.

[10] C. Smith, The Car Hacker's Handbook: A Guide for the Penetration Tester, No Starch Press, 2016.

[11] Pal-Stefan Murvay and Bogdan Groza, «Security shortcomings and countermeasures for the SAE J1939 commercial vehicle bus protocol,» 2017.

[12] Kerstin Lemke, Christof Paar and Marko Wolf, Embedded Security in Cars, Springer, 2006.

[13] M. Wolf, Security Engineering for Vehicular IT Systems, Vieweg+Teubner, 2009.

[14] Philipp Mundhenk, Sebastian Steinhorst and Suhaib A. Fahmy, «Security Analysis of Automotive Architectures using Probabilistic Model Checking,» 2015.

[15] Karl Koscher et al., «Experimental Security Analysis of a Modern Automobile,» 2010.

[16] E. Rippel, «Embedded security challenges in automotive designs,» 2008.

[17] Junko Takahashi et al., «Automotive Attacks and Countermeasures on LIN-Bus,» 2017.

[18] Johan Lindberg, «Security Analysis of Vehicle Diagnostics using DoIP,» Chalmers University of Technology, 2011.

[19] Pierre Kleberger, «On Securing the Connected Car: Methods and Protocols for Secure Vehicle Diagnostics,» Chalmers University of Technology, 2015.

[20] Tencent Keen Security Lab, «Experimental Security Assessment of BMW Cars: A Summary Report,» 2018.

[21] Dr. Charlie Miller and Chris Valasek, «Remote Exploitation of an Unaltered Passenger Vehicle,» 2015.

[22] H. Ueda et al., «Security Authentication System for In-Vehicle Network,» 2015.

[23] Charlie Miller,Chris Valasek, «CAN message injection,» 2016.

[24] Pal-Stefan Murvay and Bogdan Groza, «Security shortcomings and countermeasures for the SAE J1939 commercial vehicle bus protocol,» 2017.

Last revision of this document on: 28 May 2018