Off-Board Commercial Diagnostic Systems

Prepared by Sharika Kumar

SHARIKA KUMAR

Technical Advisor/ Cybersecurity Manager for Accelera by Cummins 2022 - **Current**

Ph.D. Candidate at The Ohio State University – 2020 - **Current**

Technical Specialist, Electronic Cybersecurity R&D, Cummins Inc. 2012 – 2022

Patents

- WO US CN US20210058777A1 - System, method, and apparatus for secure telematics communication
- WO EP US CN EP3938249A4, Method and system for detecting intrusion in a vehicle system
- WO EP US CN EP3938249A4 – Enhanced Cryptography systems and methods

Trade Secret

- Cryptographic Operations Performance Enhancement Techniques

Journal

- An embedded multichannel telemetry unit for bone strain monitoring, J Embedded Systems 2013, 14 (2013).

Papers

- SAE WCX 2023 - Cybersecurity Vulnerabilities for Off-Board Commercial Vehicle Diagnostics
- ESCAR 2023 – Securing Vehicle Diagnostic Communication

Awards

- Cummins Global Industry Impact Award – Brand Promise – Multi-Level Security for Cummins Autosar Based Software for next Generation ECM's
- Cummins Business Impact Award - Prototyping Cryptographic Features for Cummins using HSM Co-Processor
- Cummins Emission Solution Catalyst Award Nominee 2016

Agenda

01 Off-Board Commercial Vehicle Diagnostics Overview

02 Machine-In-The-Middle Attack Guided Exercises

03 ISO14229 Unified Diagnostic Services (UDS)

04 UDS Trace Interpretation Exercise

05 Cyber Defense for Diagnostic Interfaces

Training Goals

- Understand the need for off-board vehicle diagnostics
- Demonstration of Machine-In-The-Middle Attacks on a Diagnostic session
- Understand the challenges related to securing diagnostics sessions
- Interpret UDS network traffic using ISO 14229-1 standard

Cybersecurity Vulnerabilities for Off-Board Commercial Vehicle Diagnostic Sessions

01

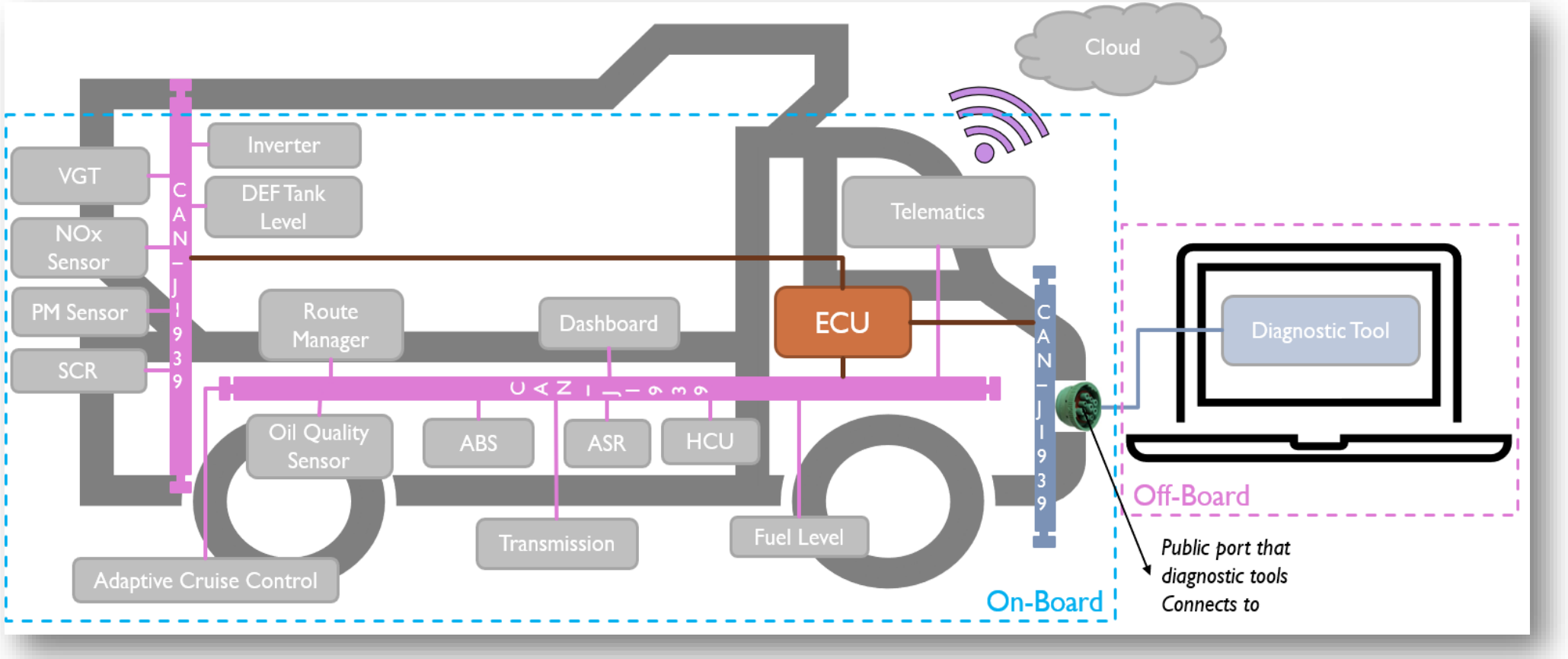
Author(s): Sharika Kumar, Jeremy Daily, Qadeer Ahmed, Anish Arora

Affiliated: Accelera by Cummins/Ohio State University, Colorado State University, Ohio State University

SAE Technical Paper 2023-01-0040, 2023, https://doi.org/10.4271/2023-01-0040

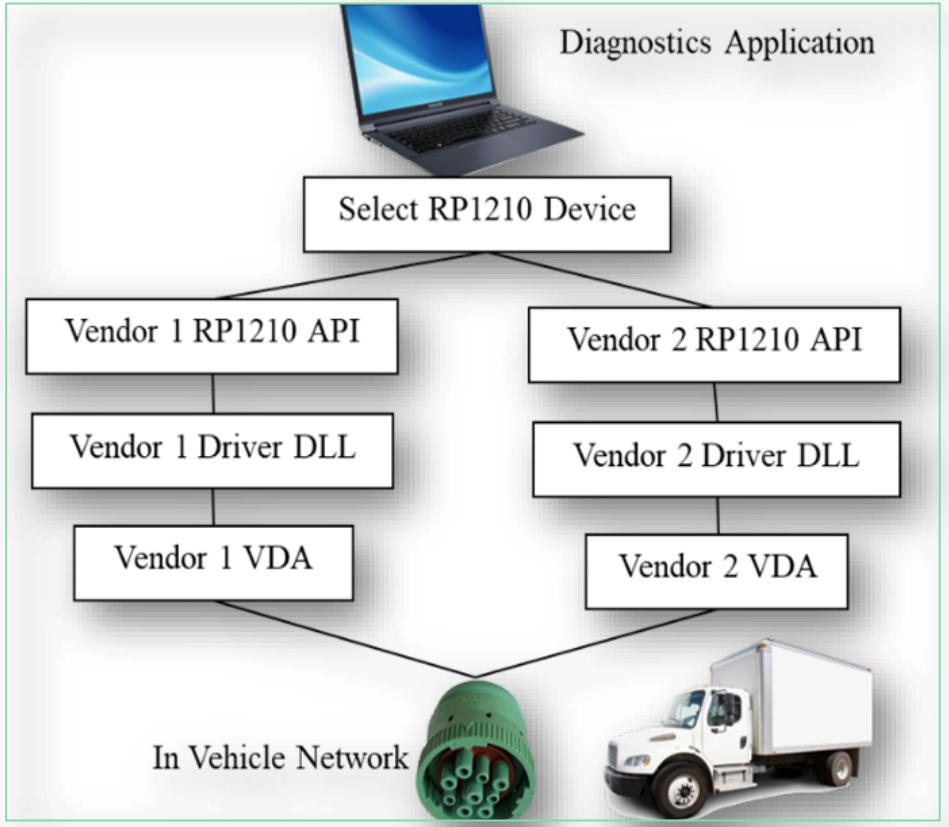
Background: Medium and Heavy Duty (MHD) Network Communication

- MHD networks are typically built on SAE J1939 over CAN 2.0b (Multi-master serial bus, features unicast and broadcast messages, transport fragmentation/reassembly)
- Diagnostic application often run on a Windows-based PC or laptop



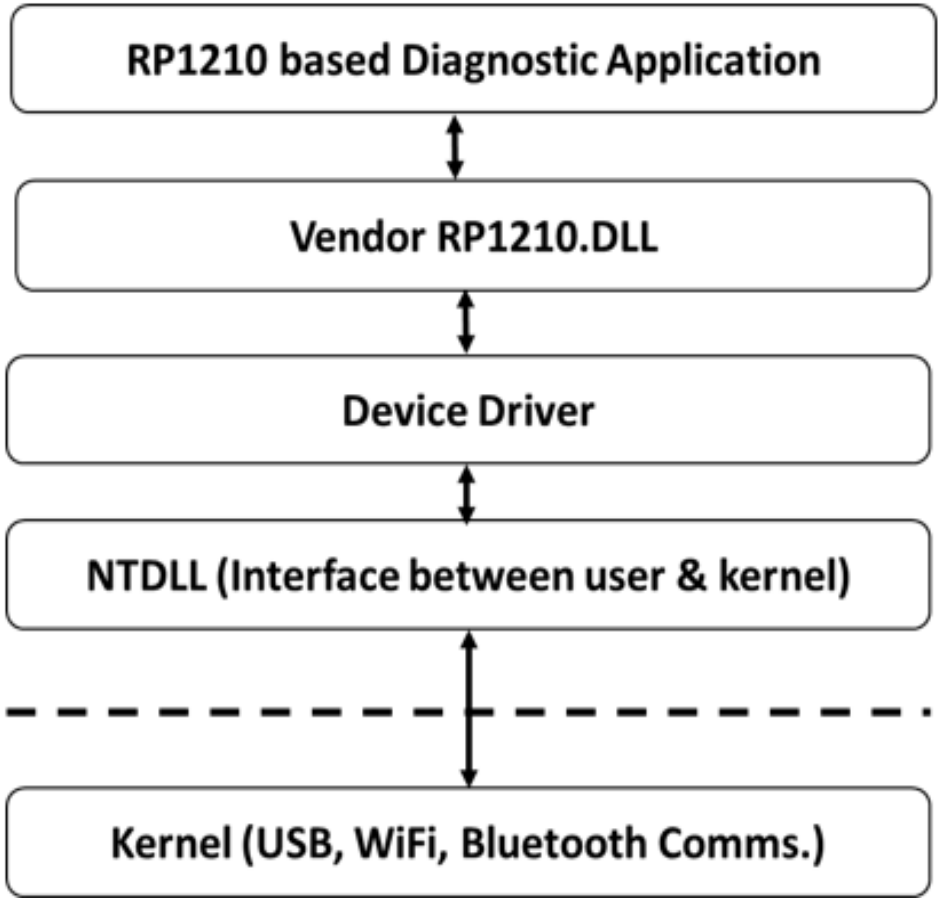
Vehicle Diagnostic Adapters (VDAs)

- VDAs translates vehicle communications to a diagnostic application
- American Trucking Association's (ATA) Technology and Maintenance Council (TMC) initiated RP1210 in the 1990's to manage VDAs
- RP1210 describes a standard API for a Windows PC application to communicate with the network
- A trusted maintenance technician is often granted access to connect a VDA to the diagnostic port to exercise the off-board communications

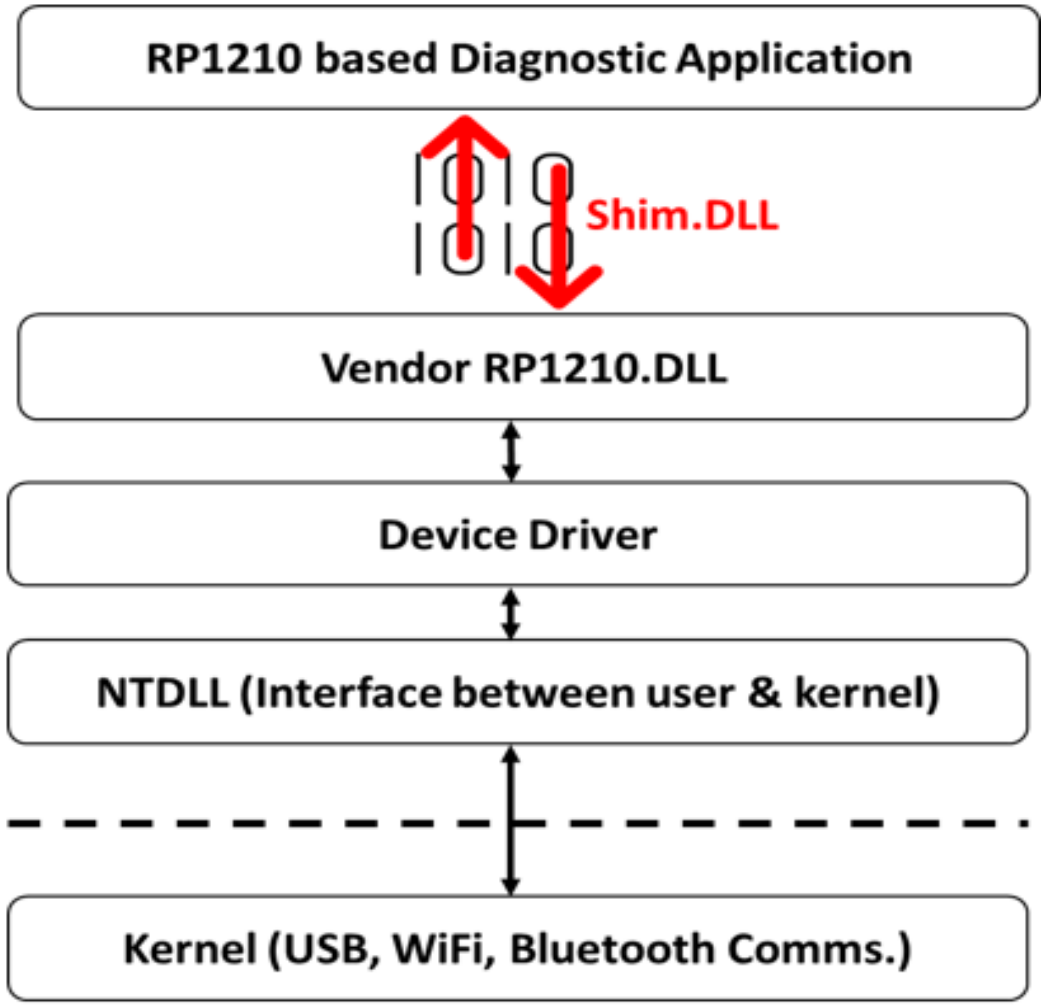


The concept of RP1210

Attacking Vehicle Diagnostic Adapter Drivers

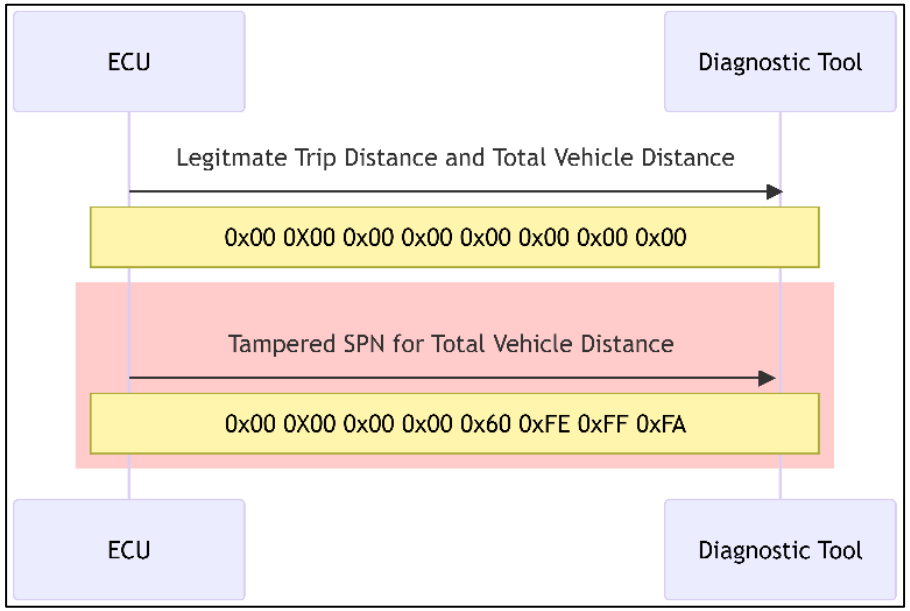


Communication stack within the PC/laptop



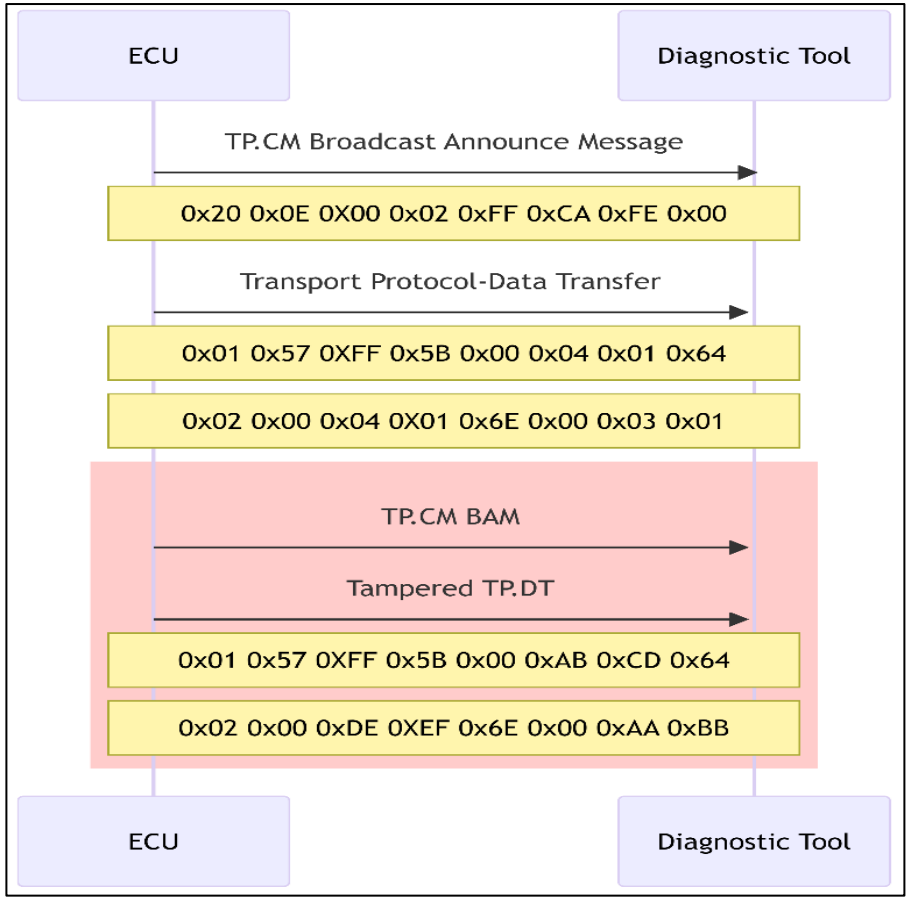
Attack uses inserted shim DLL to tamper RP1210 communications

Periodic, Single Frame Message Attack



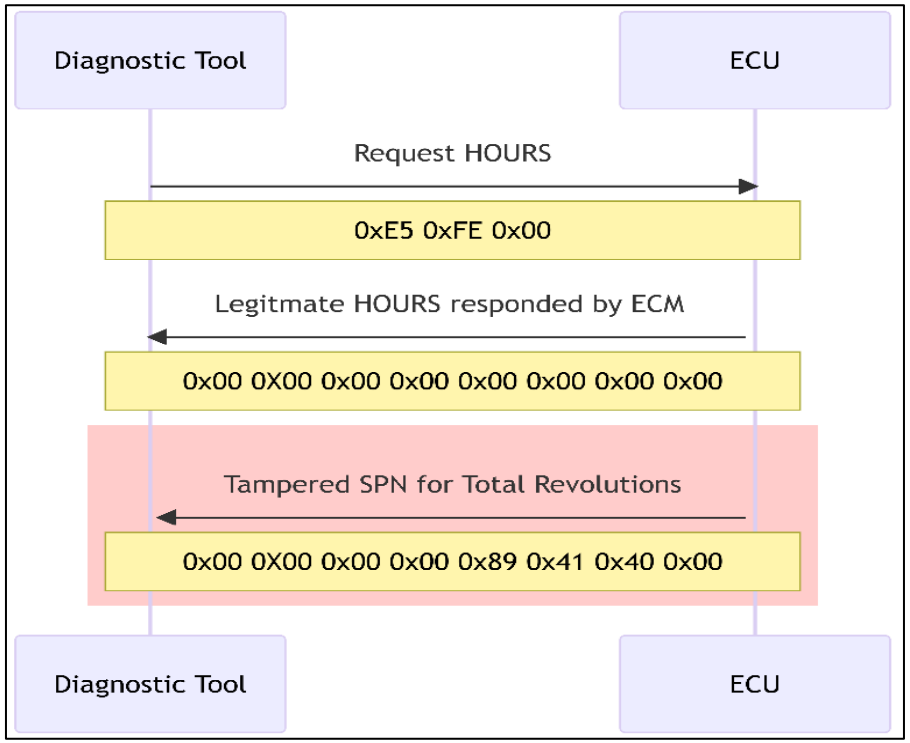
Manipulation of SPN 245, total vehicle distance. The legitimate message has all zeros as the ECM used was brand new

Periodic, Multi-Frame Message Attack



Demonstration of manipulating multi-frame messages in J1939 with the DM1 message as an example.

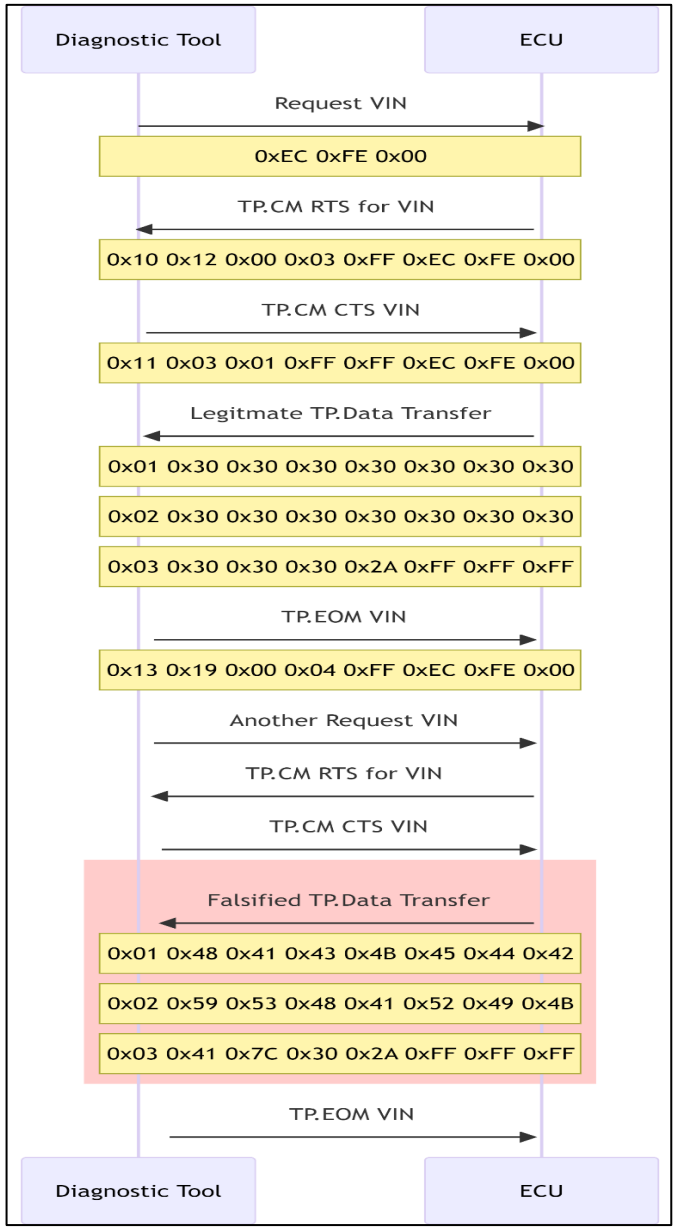
Request/Respond Single Frame Message Attack



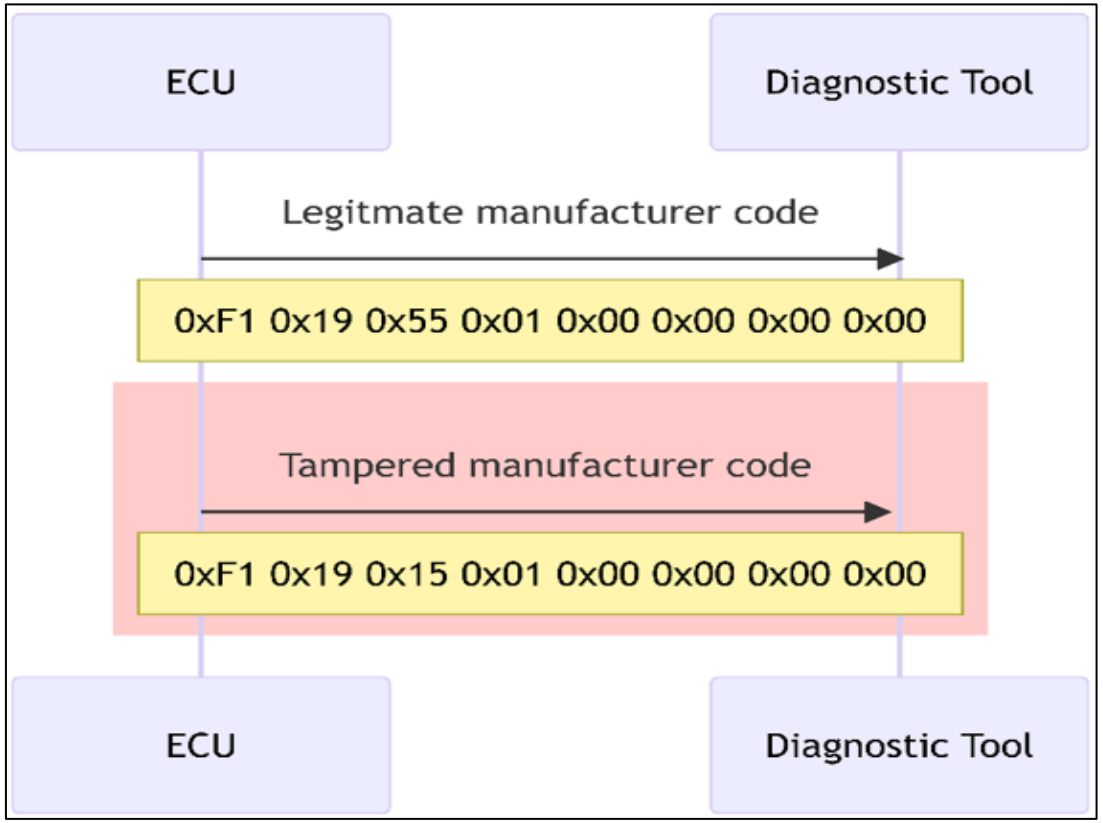
Manipulation an on-request message for engine revolutions. The legitimate message has all zeros as the ECM used was brand new.

Request/Respond Single and Multi-Frame Message Attack

Manipulation of Vehicle Identification Number (VIN), which is a requested multi-frame message.



On-Event Message Attack



Sequence diagram that reflects log files to change the data in the Address Claim in the NAME field defined in SAE J1939-81.

Prepare for the hands-on session

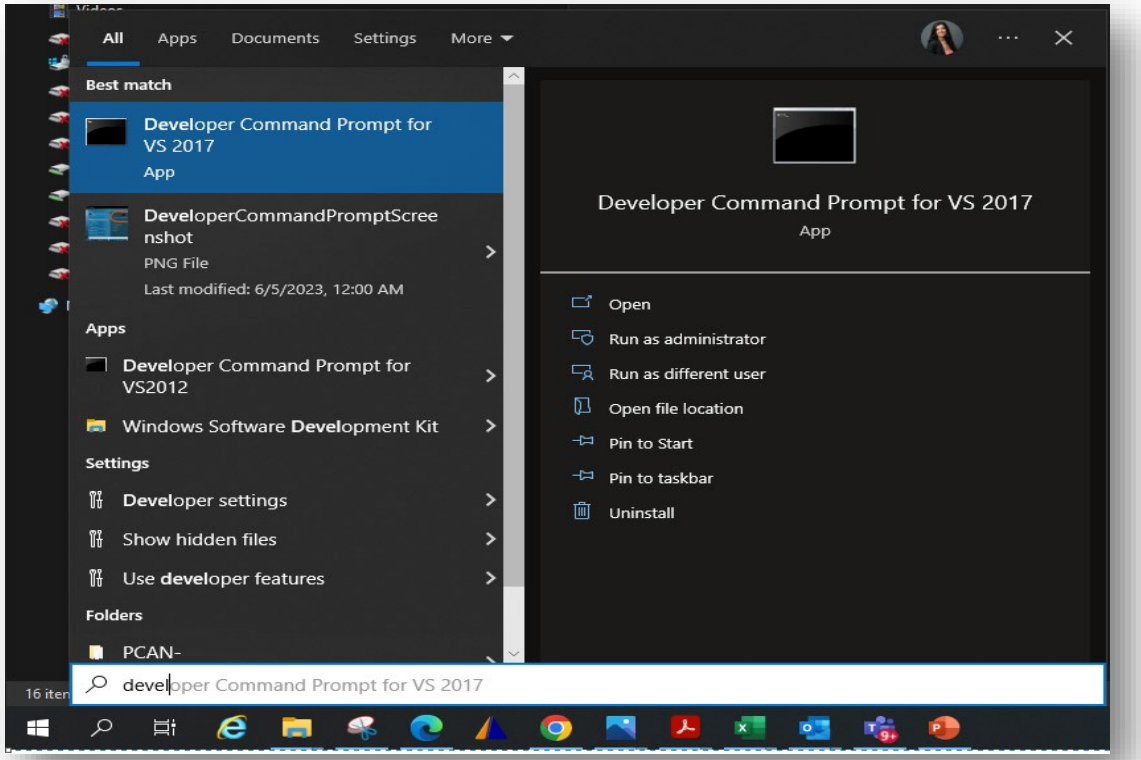
#1: Clone shim dll source code from https://github.com/SystemsCyber/ShimDLL

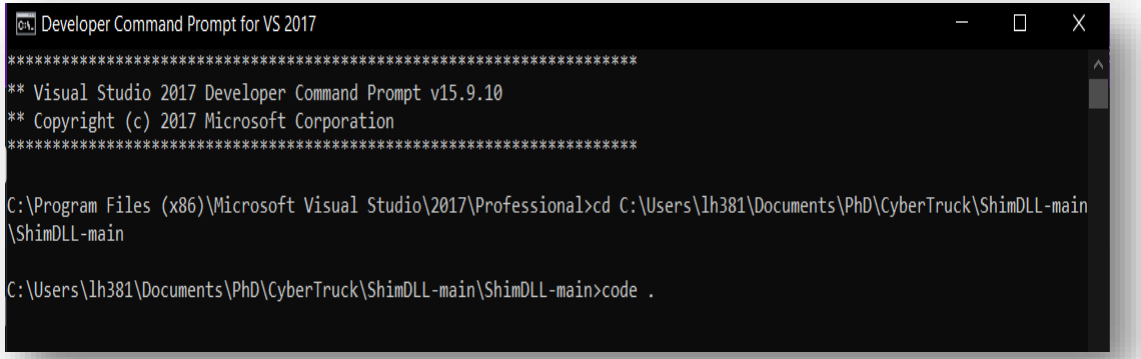
#2: Launch the Developer Command Prompt for VS

#3: Change Directory to cloned directory

#4: Launch VS Code.

```
> code .
```

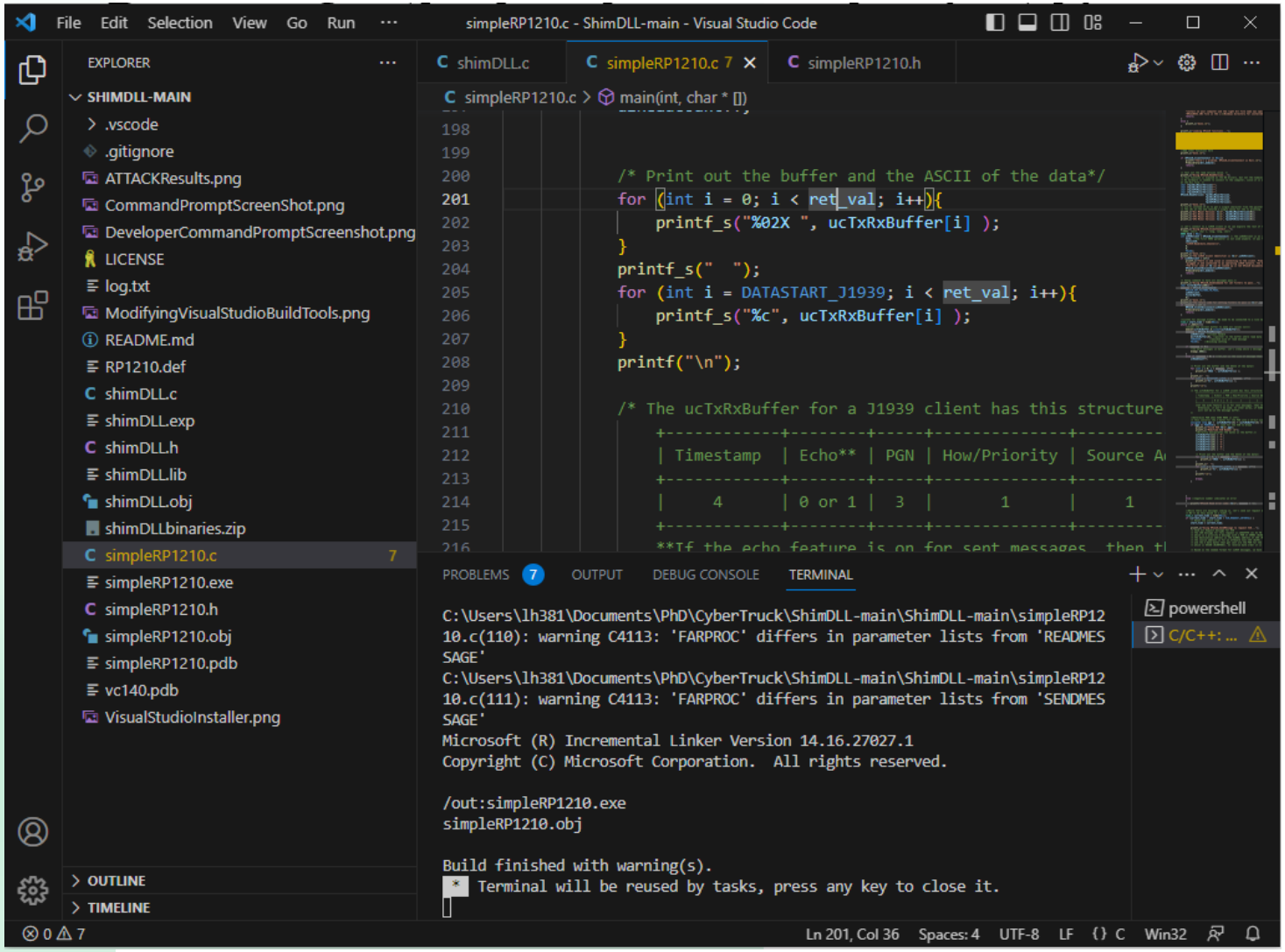




Prepare for the hands-on session (cntd.)

#5: Your Visual Studio Code window is pulled up

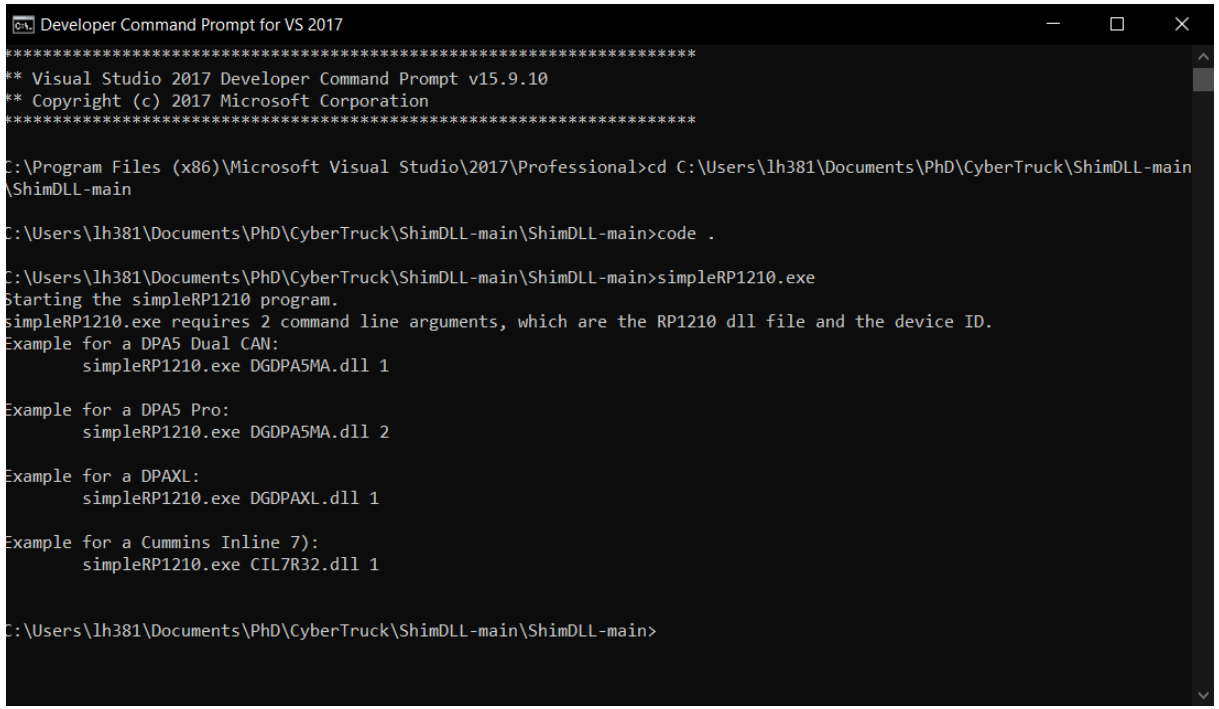
#6: Do **Control+Shift+B** to build



Prepare for the hands-on session (cntd.)

#7: Run **simpleRP1210.exe** and choose **Cummins Inline 7**

```
> simpleRP1210.exe CIL7R32.dll 1
```



Class Exercise #1

Using PowerSpec, simpleRP1210 and Inline 7:

1. On Ubuntu Computer: Connect and monitor CAN data from truck ECUs at CAN port#0.
    1. Look for VIN with SavvyCAN or candump.
    2. The data will bin from transport messages 0x0CEBFF00 and 0x0CECFF00

2. On Windows
    1. Run SimpleRP1210.dll
    2. Run Powerspec and get dataplate

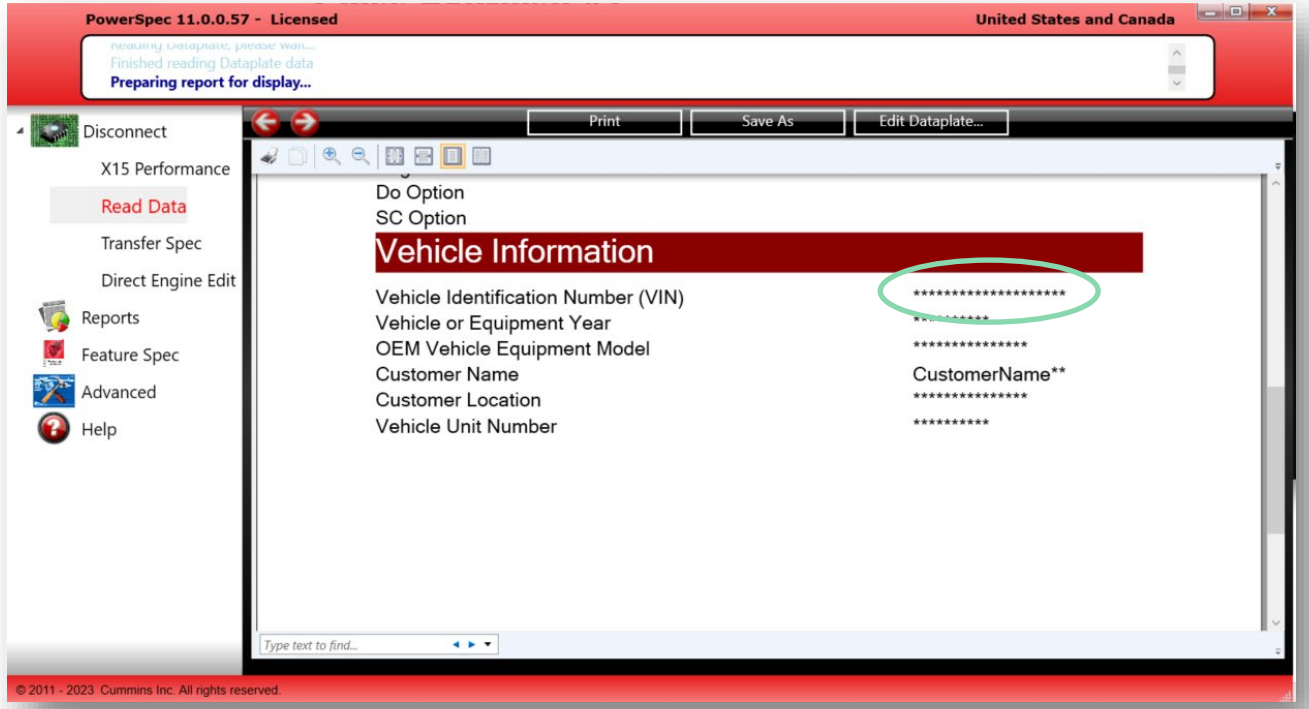
3. Compare value of **VIN**
    1. PowerSpec Dataplate
    2. simpleRP1210.exe

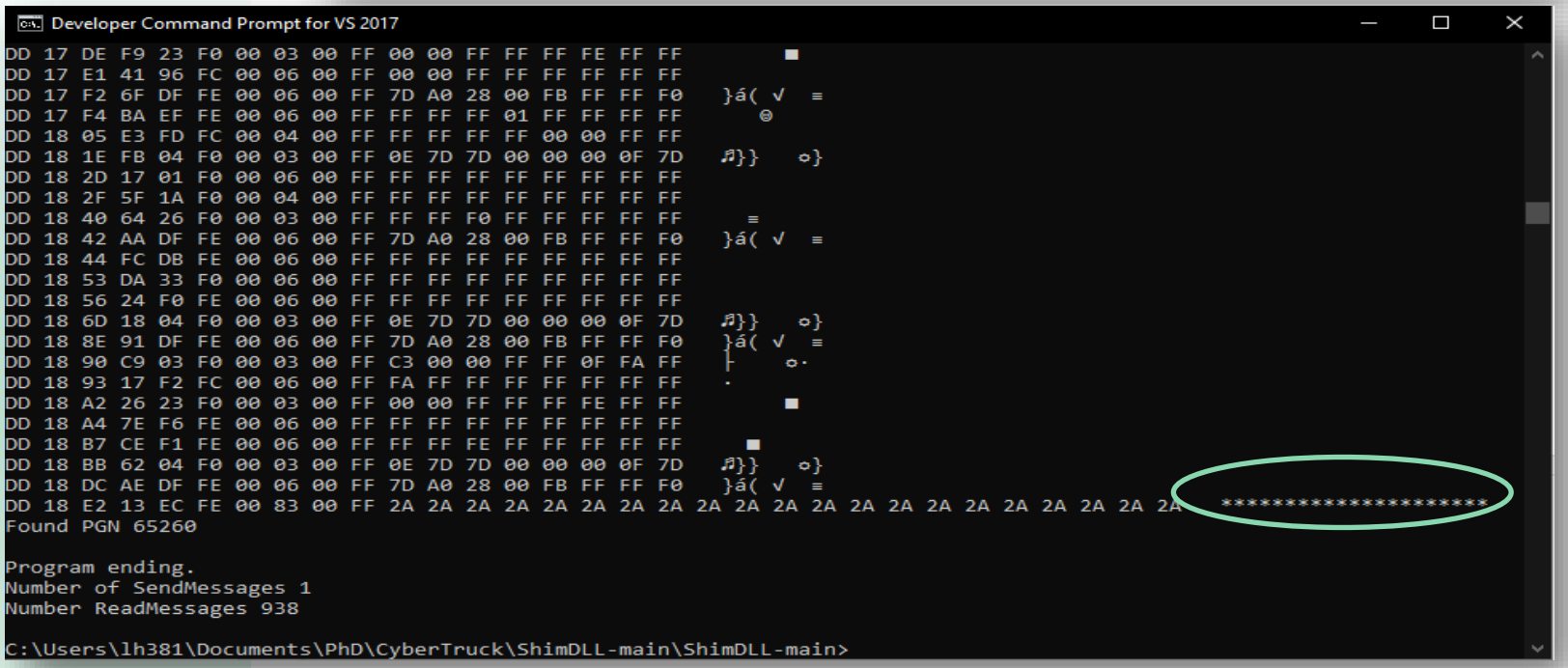
Class Exercise #1

Solution for **VIN:**

Both VINs should match

The data on the CAN is in the transport protocol





Class Exercise #2

Using PowerSpec, simpleRP1210, shimdll and Inline 7:

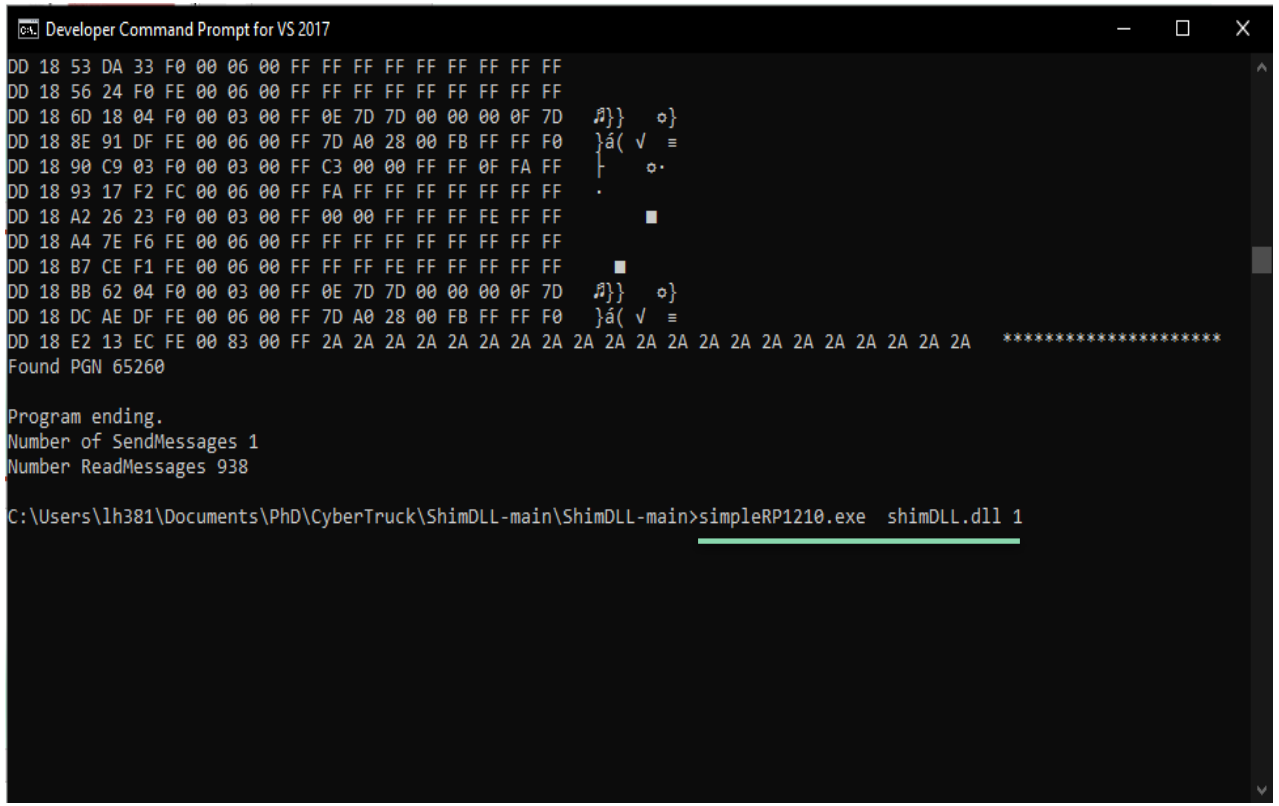
1. Compile shimdll.c using

cl.exe /nologo /LD /EHsc /W4 /D_USRDLL /D_WINDLL shimDLL.c /link /OUT:shimDLL.dll /DEF:RP1210.def

2. Enter the following command

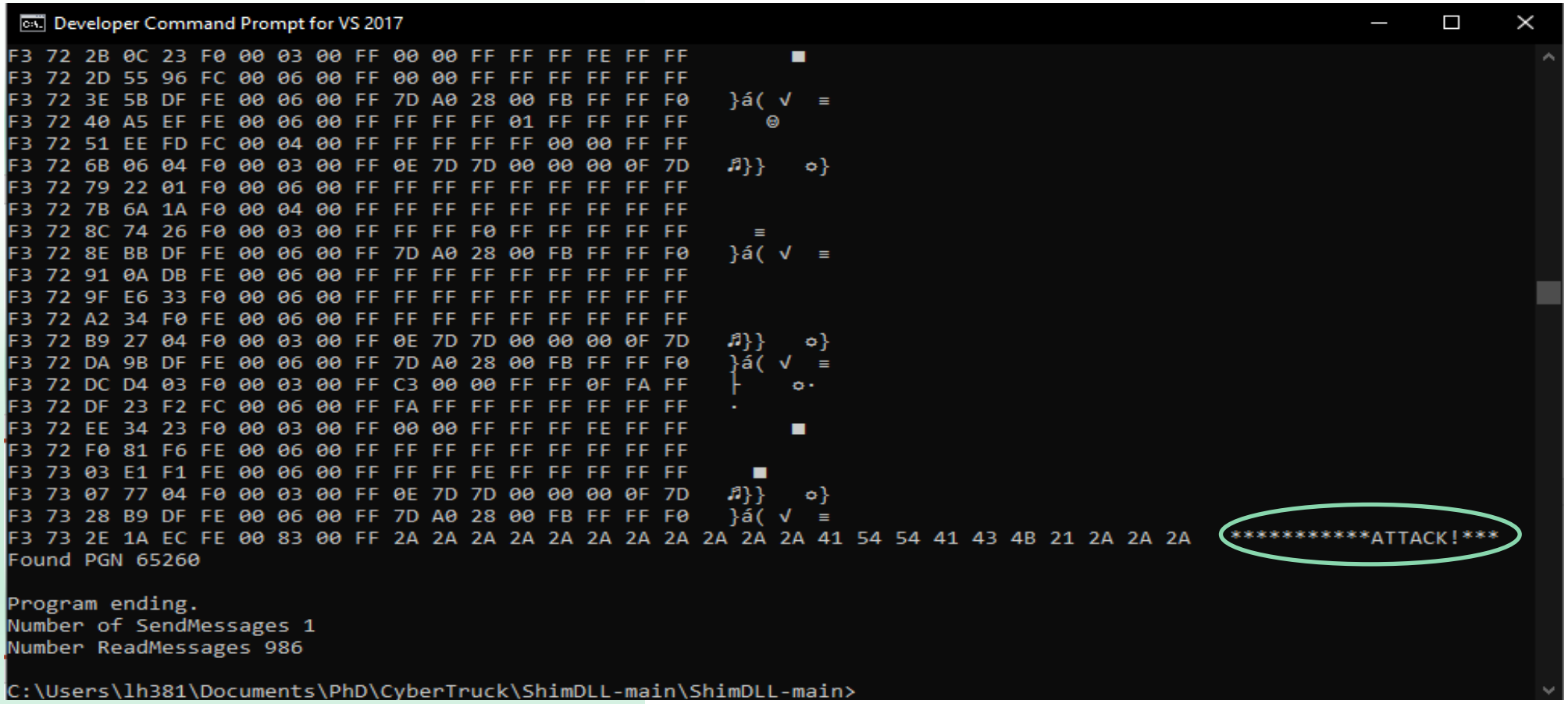
simpleRP1210.exe shimDLL.dll 1

3. Notice the new altered response for VIN



Class Exercise #2

Solution for tampered **VIN**



Class Exercise #3: Modifying SHIM for modifying HOURS

Using PowerSpec, shim dll and Inline 7:

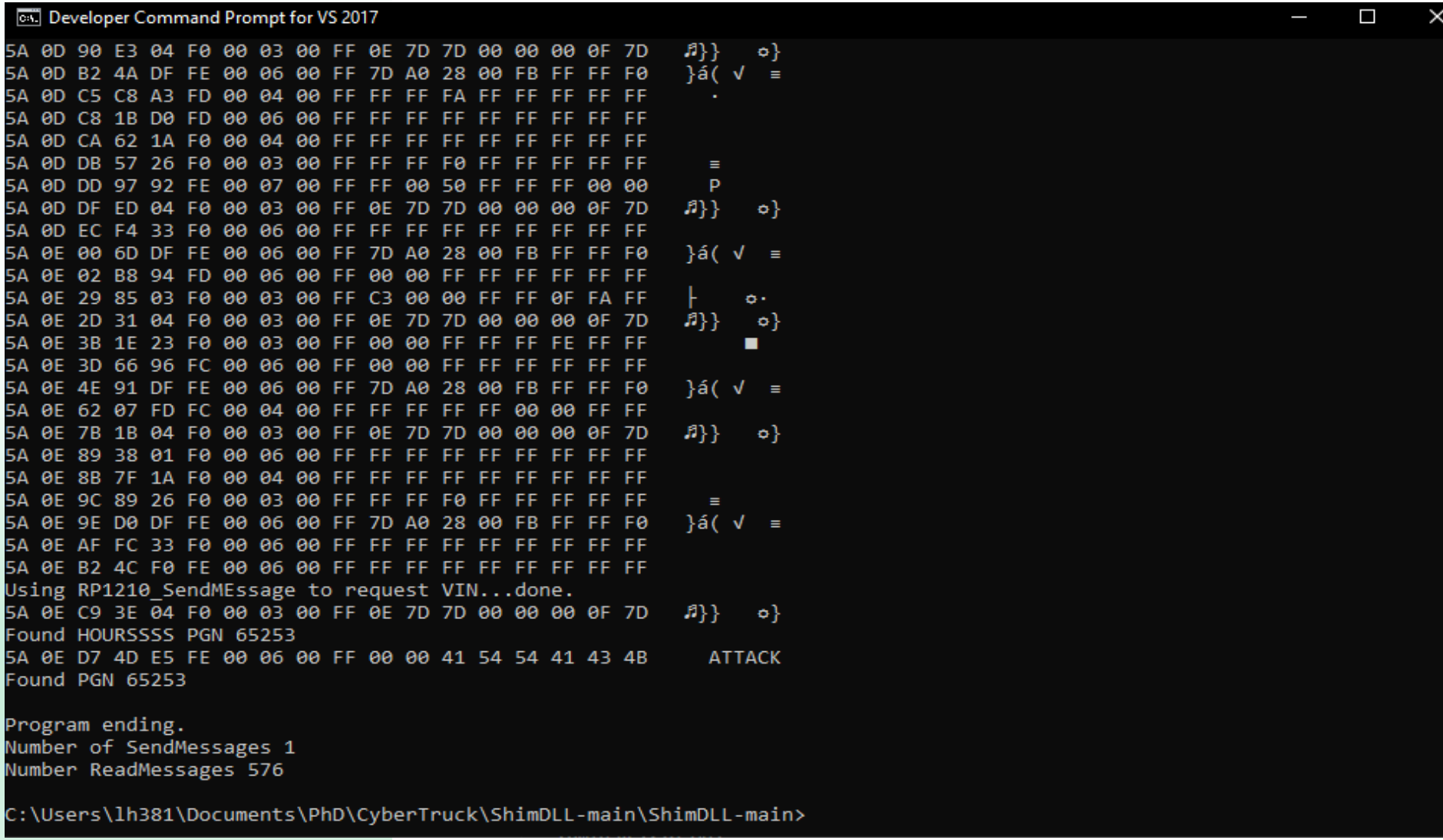
Develop code to demonstrate tampering with the engine hours.

Hints:

1. Look up the PGN for hours in the J1939 DA
2. Alter the simpleRP1210.c code to request engine hours.
3. Alter the simpleRP1210 code to exit once it finds HOURS.
4. Alter the shimDLL.c code to detect engine hours.
5. Alter the shimDLL.c code to tamper with the bytes of the hours message.

Class Exercise #3

Solution for **HOURS**



Challenge Exercise

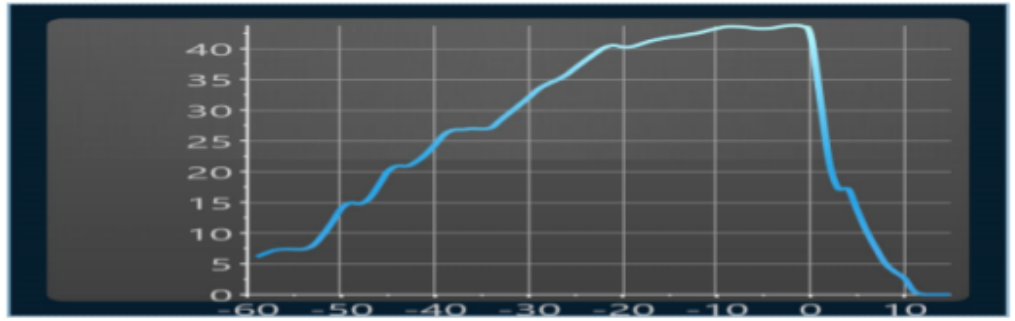
Build a shim for PowerSpec

Sudden Vehicle Speed Deceleration Report Record 1

Engine Type	ISX 2010	Ecm Code	CL10132.39
Engine Serial Number	60811136	Software Phase	7.70.0.71
Unit Number	0000000000	Extraction Date	04-03-2019 04:37:51
Sudden Decel Threshold Rate:	N/A	ECM Run time	8227:56:32

Occurrence Date: N\A
ECM Run Time at Occurrence: 2899:18:36
Air Temperature (°F) at Occurrence: 72
Occurrence Distance (mi): 109323.4

Vehicle Speed



Record 1

Time (Seconds)	Vehicle Speed (mph)	Engine Speed (rpm)	Engine Load (%)	Throttle (%)	Brake Status	Clutch Status	Cruise Status	Lamp Status
-59	6	683	34.8	30.1	-	-	-	-
-58	7	788	33.6	32.2	-	-	-	-
-57	7	813	21.9	27.5	-	-	-	-

Genuine Data

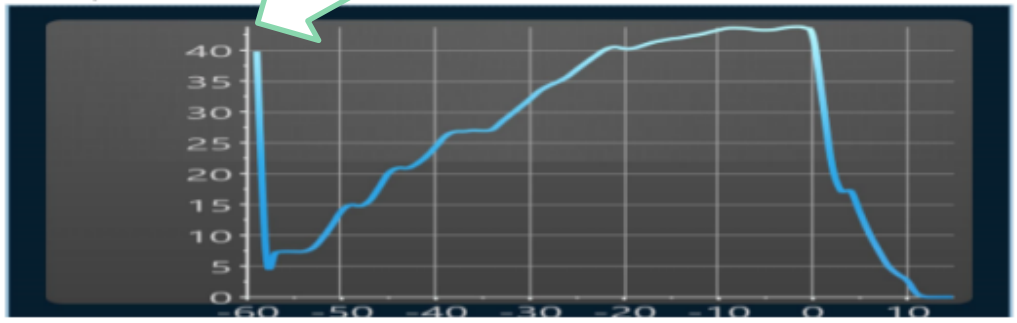
Sudden Vehicle Speed Deceleration Report Rec

Engine Type	ISX 2010	Ecm Code	10132.39
Engine Serial Number	60811136	Software Phase	7.70.0.71
Unit Number	0000000000	Extraction D	04-03-2019 02:08:09
Sudden Decel Threshold Rate:	N/A	ECM R	8225:27:34

Occurrence Date: N\A
un Time at Occurrence: 2899:18:36
Air Temperature (°F) at Occurrence:
ccurrence Distance (mi): 109323.4

Altered Speed Value

Vehicle Speed



Record 1

Time (Seconds)	Vehicle Speed (mph)	Engine Speed (rpm)	Engine Load (%)	Throttle (%)	Brake Status	Clutch Status	Cruise Status	Lamp Status
-59	40	683	34.8	30.1	-	-	-	-
-58	7	788	33.6	32.2	-	-	-	-
-57	7	813	21.9	27.5	-	-	-	-

Attack Data

Section Summary

- Understood RP1210 API's for off-board vehicle diagnostics
- Utilized Machine-in-the-Middle (MitM) attacks using a Shim DLL on RP1210 diagnostics communications
- Attacker does not need physical access to the network (Technicians do that for them).
- All exploits are executed on Windows
  - Truck security relies on the security of the Windows computers used by technicians.
  - Stealthy
    - Challenging to detect with intrusion detection systems
    - Most diagnostics messages seem legitimate.

- Possible extensions:
  - Session hijacking
  - Forensic write-blocking (or denial of service)
  - Changing governed speed limits (65 -> 15 mph)

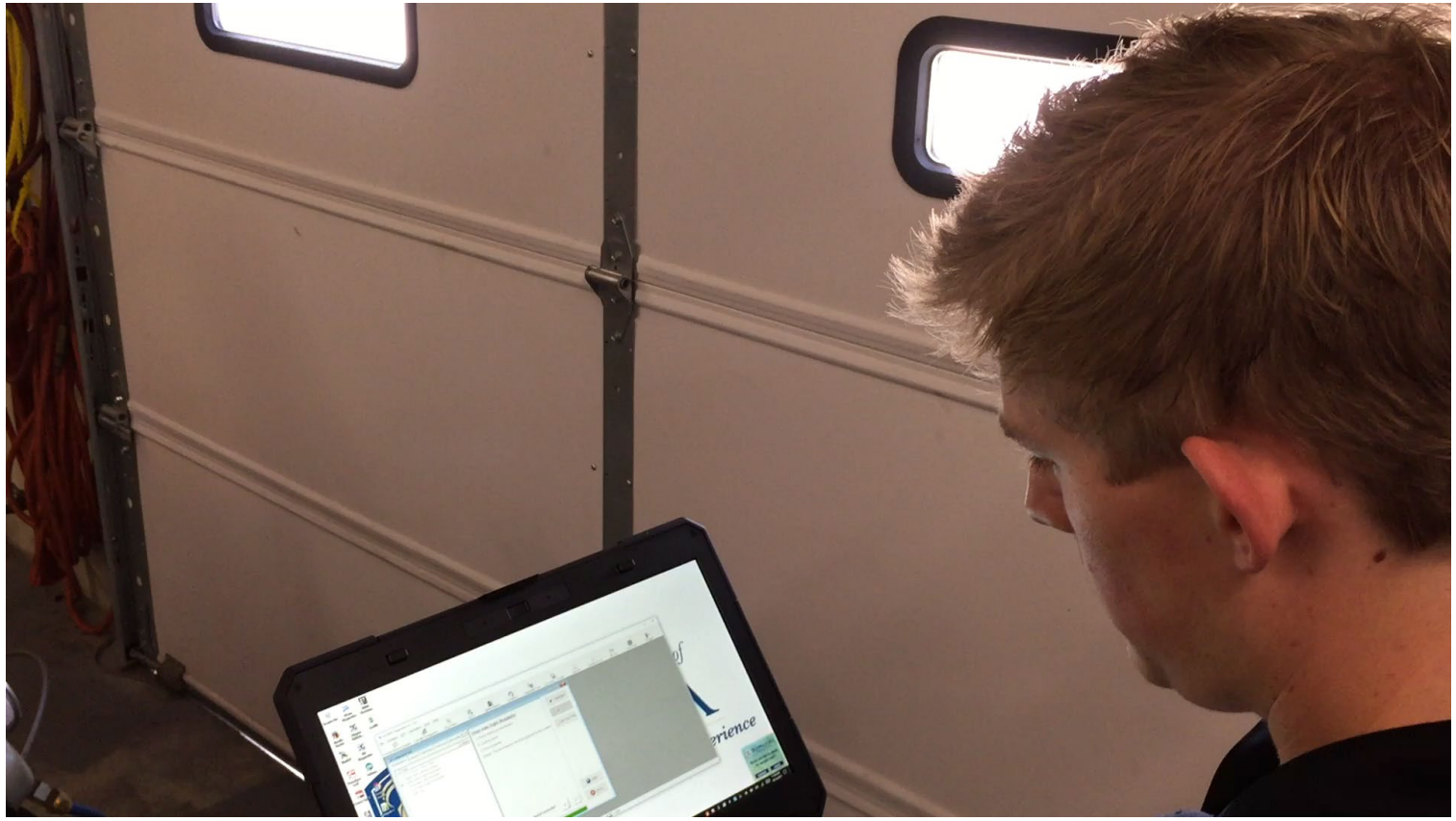
Break Time

Overview of Unified Diagnostics Protocol (UDS)

02

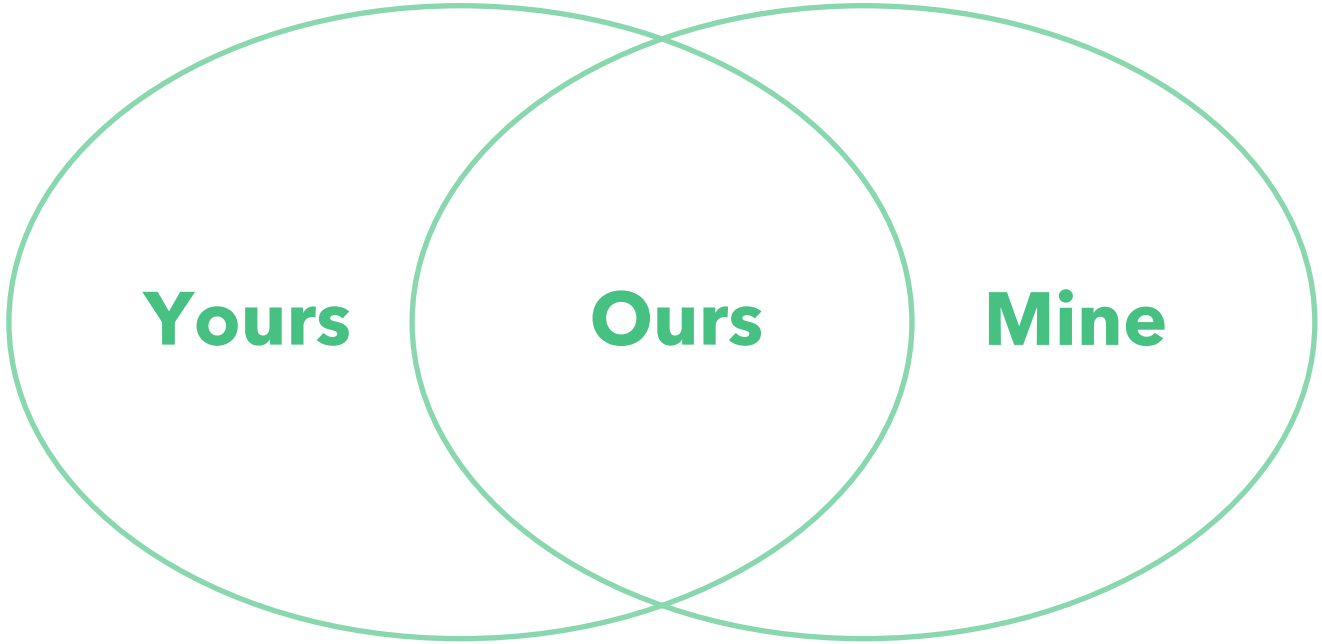
Example UDS Session for Brake Controls

- A session is established for brake controller diagnostics
- Students commanded a brake chuff test
- All communications went over UDS
- The brake controller trusts the UDS commands



Why Unified Diagnostic Services?

- **Interoperability** with various vendors for engine, controls, tools
- Public standards encourage interoperation
- Increased productivity
- Reduced errors
- Reduced cost



Accelera

Evolution of Diagnostic Protocols

1989

ISO 9141 *Requirements for interchange of digital info (K-Line)*

1994

ISO 9141-2 *CARB Requirements for interchange of digital information*

ISO 14230 -1\|2\|3\|4 Keyword Protocol 2000 (KWP2000) *– Physical layer, Datalink layer, Application layer, Requirements for emission related systems, VDS guideline*

1998

ISO 14229 *Diagnostic Service Specification (UDS)*

1999

2001

ISO 15031 *Communication between vehicle and external equipment for emissions related diagnostics*

2004

Diagnostics on CAN

2004

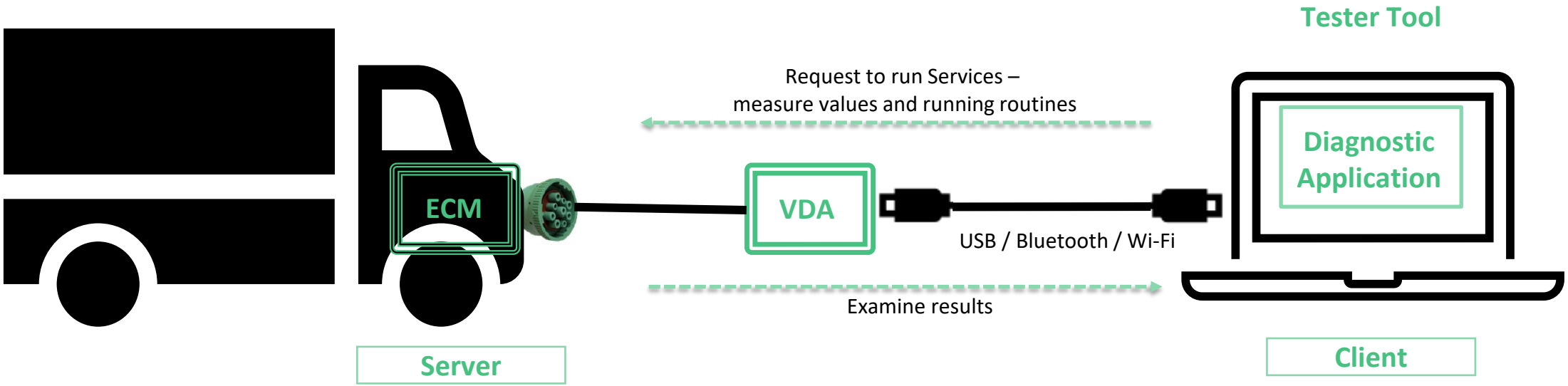
UDS

ISO 14229 based on OSI Model

OSI Layer	UDS Enhanced Diagnostic Services
Diagnostic Application	User defined
7. Application Layer	**ISO 14229-1 UDSonCAN** ISO 14229-4 UDSonFR ISO 14229-5 UDSonIP ISO 14229-6 UDSonK-Line ISO 14229-7 UDSonLIN ISO 14229-81 UDSonCXPI ISO 27145-3 VOBD
6. Presentation Layer	ISO 27145-2 VOBD, **NA for CAN**
5. Session Layer	**ISO 14229-2**
4. Transport Layer Defines client-server system (ECU being server)	**ISO 15765-2 DoCAN** ISO 10681-2 Communication on FlexRay ISO 13400-2 DoIP ISO 17987-2 LIN ISO 20794-32 CXPI ISO 27145-4 VOBD
3. Network Layer	**ISO 15765-2 DoCAN** ISO 10681-2 Communication on FlexRay ISO 13400-2 DoIP ISO 17987-2 LIN ISO 20794-3 CXPI ISO 27145-4 VOBD
2. Data Link Layer	**ISO 11898-1** ISO 11898-2 ISO 17458-2 ISO 13400-3 IEEE 802.3 ISO 14230-2 ISO 17987-3 LIN ISO 20794-43 CXPI ISO 27145-4 VOBD
1. Physical Layer	ISO 11898-1 **ISO 11898-2** ISO 17458-4 **ISO 13400-3** IEEE 802.3 ISO 14230-1 ISO 17987-4 LIN ISO 20794-4 CXPI ISO 27145-4 VOBD

Note: Bolded are standards that apply to CAN

Diagnostic “Services”



CAN Frame Structure

29-bit CAN Frame Format														
128 bits														
	29-bit Identifier					Control field			Data field	CRC		ACK		
SOF	ID	SRR	IDE	Extended ID	RTR	r1	r0	DLC	**Data Payload**	CRC	Delimiter	ACK	Delimiter	EOF
1 bit	11 bit	1 bit	1 bit	18 bit	1 bit	1 bit	1 bit	4 bits	**64 bits**	15 bits	1 bit	1 bit	1 bit	7 bits

CAN ID for UDS in J1939
uses PGN 0xDA00

UDS tunnel here

Accelera

Normal Addressing per ISO 15765-2 on 29-bit CAN Identifiers

29-bit CAN Frame Format

128 bits

	29-bit Identifier																	
SOF	ID					SRR	IDE	Extended ID		RTR	r1	r0	DLC	**Data Payload**	CRC	Delimiter	ACK	Delimiter	EOF
1 bit	11 bit					1 bit	1 bit	18 bits		1 bit	1 bit	1 bit	4 bits	**64 bits**	15 bits	1 bit	1 bit	1 bit	7 bits
	Priority	Reserved/ Extended	Data Page	PDU Format	PDU-Specific (Destination or PDU format)	Source Address													
	3 bits	1 bit	1 bit	8 bits	8 bits	8 bits													
	Default 110_2	**0**	**0**	**218 (0xDA)**	**N_TA (Network Target Address)** *J1939 Destination Address (DA)*	**N_SA (Network SA)**								N_PCI (Network Protocol Control Information)	N_Data				

J1939 PGN for UDS is 0xDA00

Source Address is F1 per ISO 15765-4

Ex: 18DAXXYY

XX – N_TA

YY – N_SA

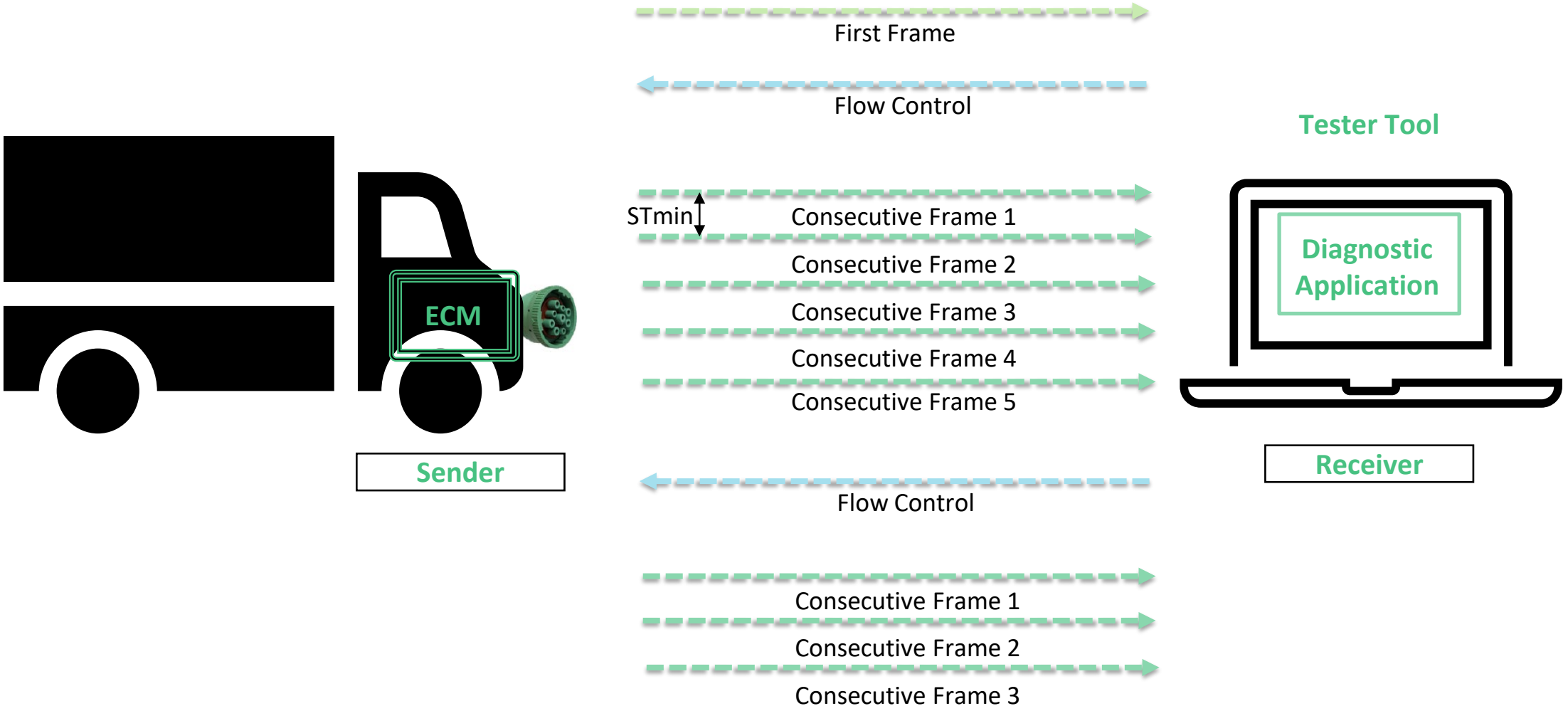
Accelera

Simplified Generic Structure of UDS

29-bit CAN Frame Format							
128 bits							
29-bit Identifier						Data field	
Priority	Reserved/ Extended	Data Page	PDU Format	PDU-Specific (Destination or PDU format)	Source Address		
3 bits	1 bit	1 bit	8 bits	8 bits	8 bits		
Default 110_2	0	0	219	N_TA	N_SA	**N_PCI (Protocol Control Information)**	N_Data

UDS used multiplexed messages

Segmentation per ISO 15765-2 Transport Protocol (TP)



Summary of N_PCI Bytes

Note: First nibble is either 0, 1, 2 or 3

N_PDU name	N_PCI bytes						
	Byte #1		Byte #2	Byte #3	Byte #4	Byte #5	Byte #6
	Bits 7 – 4	Bits 3 – 0					
SingleFrame (SF) (CAN_DL ≤ 8)	0000_2	SF_DL	—	—	—	—	—
SingleFrame (SF) (CAN_DL > 8)[a]	0000_2	0000_2	SF_DL	—	—	—	—
FirstFrame (FF) (FF_DL ≤ 4 095)	0001_2	FF_DL		—	—	—	—
FirstFrame (FF) (FF_DL > 4 095)[b]	0001_2	0000_2	$0000\ 0000_2$	FF_DL			
ConsecutiveFrame (CF)	0010_2	SN	—	—	—	—	—
FlowControl (FC)	0011_2	FS	BS	ST_{min}	N/A	N/A	N/A

[a] Messages with CAN_DL > 8 shall use an escape sequence where the lower nibble of Byte #1 is set to 0 (invalid length). This signifies to the network layer that the value of SF_DL is determined based on the next byte in the frame (Byte #2). As CAN_DL is defined to be greater than 8, this definition is only valid for CAN FD type frames.

[b] Messages larger than 4 095 bytes shall use an escape sequence where the lower nibble of Byte #1 and all bits in Byte #2 are set to 0 (invalid length). This signifies to the network layer that the value of FF_DL is determined based on the next 32 bits in the frame (Byte #3 is the MSB and Byte #6 the LSB).

NOTE Dash lines are not utilized for PCI information, but depending on the PDU, they might be utilized for payload data.

Ref: Technical Committee ISO/TC 22/SC 31,
"ISO 15765-2 : 2016, Road Vehicles – Diagnostic Communication over CAN (DoCAN)-Part 2 Transport Protocol and Network Layer Services"

Accelera

Security handshake that authenticates a session

Enhanced authentication recently added

Secures session

Standardized UDS Services

Enables tester to execute operation

SID in Hex	Mnemonic	Service Name	Functional Class
10	DSC	Diagnostic Session Control	Diagnostic and Communication Management
11	ER	ECU Reset	
27	SA	Security Access	
28	CC	Communication Control	
29	ARS	Authentication	
3E	TP	Tester Present	
84	SDT	Secured Data Transmission	
85	CDTCS	Control DTC Setting	
86	ROE	Response on Event	
87	LC	Link Control	
22	RDBI	Read Data by Identifier	Data Transmission
23	RMBA	Read Memory by Address	
24	RSDBI	Read Scaling Data by Identifier	
2C	RDBPI	Dynamic Define Data Identifier	
2E	DDDI	Write Memory by Identifier	
3D	WDBI	Write Memory by Address	
14	CDTCI	Clear Diagnostic Information	Stored Data Transmission
19	RDTCI	Read DTC Information	
2F	IOCBI	Input Output Control by Identifier	Input Output Control
31	RC	Routine Control	Remote Activation of Routines
34	RD	Request Download	Upload Download
35	RU	Request Upload	
36	TD	Transfer Data	
37	RTE	Request Transfer Exit	
38	RFT	Request File Transfer	

UDS Log file Class Exercise #1

Inspect the CAN trace on the paper sheet and address these questions:

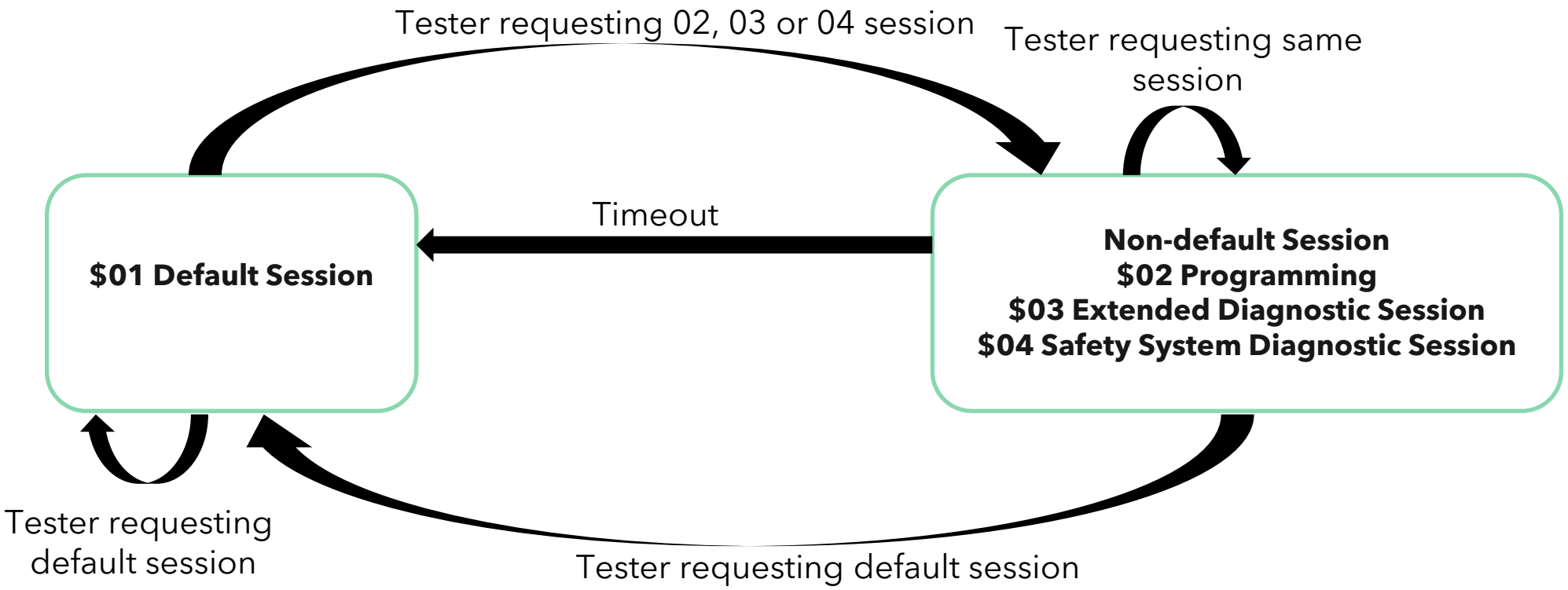
1. What kind of diagnostics session is being used?
2. What is the hex code for the first DID requested by ID?
3. What was the line number for the message that requested a seed for the security service?
4. What is the length of the seed for the security service?
5. What is the value (in hex bytes) of the seed?
6. What is the key value?
7. After examining the values of the seed and key, can you identify any security issues?
8. What is the Routine ID (RID) for the Service Routine in the trace?
9. What is the reason for the negative acknowledgement for the Service Routine Request?
10. What line number started the Service Routine response?

UDS Trace A:

```
Line Num:        @timestamp:   PGN (3)   Pri  SA  DA  Eight Data Bytes
00000001: @00:20:15.187759: 00 df 00 06 f1 ff 3f ff ff ff ff ff ff ff
00000002: @00:20:15.201856: 00 da 00 06 f1 00 03 22 fd 02 00 00 00 00
00000003: @00:20:15.202148: 00 da 00 06 00 f1 04 62 fd 02 ff 8a 20 f0
00000004: @00:20:15.545797: 00 da 00 06 f1 00 03 22 fd 00 00 00 00 00
00000005: @00:20:15.546090: 00 da 00 06 00 f1 07 62 fd 00 00 00 04 1b
00000006: @00:20:15.548736: 00 da 00 06 f1 00 03 22 fd 02 00 00 00 00
00000007: @00:20:15.549032: 00 da 00 06 00 f1 04 62 fd 02 ff 8a 20 f0
00000008: @00:20:15.551770: 00 da 00 06 f1 00 02 10 03 00 00 00 00 00
00000009: @00:20:15.552064: 00 da 00 06 00 f1 06 50 03 00 32 01 f4 f0
00000010: @00:20:15.556743: 00 da 00 06 f1 00 02 10 02 00 00 00 00 00
00000011: @00:20:15.557041: 00 da 00 06 00 f1 03 7f 10 78 50 00 20 f0
00000012: @00:20:15.587414: 00 da 00 06 00 f1 06 50 02 00 32 01 f4 00
…
```

Session State Machine

Service ID $10



Accelera

Seed/Key Security

Service ID $27

Server

Request seed

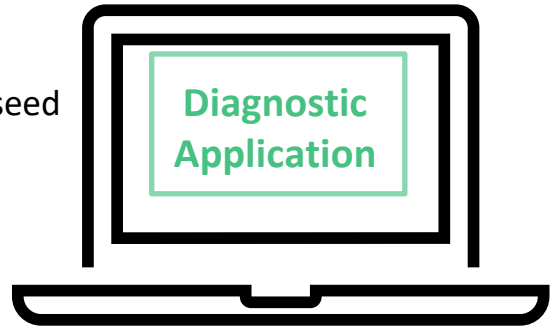
Send seed

Encrypt seed

Session key = Encrypted seed

Decrypt and compare seed

Tester Tool



Client

Authorized tool - Access permitted

Read data by identifier

Data value by identifier

.
.
.

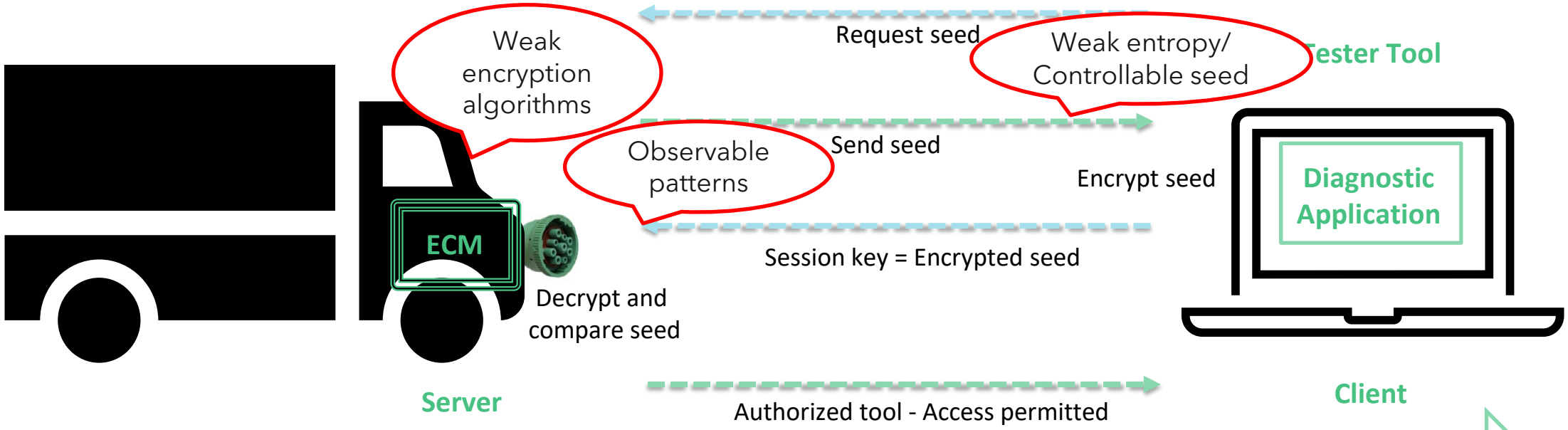
Start routine

Routine control respond

Unencrypted session communication

Security Challenge for Seed/Key Exchange

Evolution of Service ID $27 to $29



Challenge response-based security access **27$** (symmetric algo)

1. Challenge response-based algorithm-based security access (symmetric/asymmetric) **29$**
2. Certificate based authentication **29$**
3. Bidirectional authentication **29$**

Trace 1
ISO 15765-2 normal fixed addressing

18	DA	00	F1	3	02	3E	00				
18	DA	F1	00	8	02	7E	00	*00*	*00*	*00*	*00*	*00*
18	DA	00	F1	4	03	22	86	CA			
18	DA	F1	00	8	07	62	86	CA	00	00	03	F6
18	DA	00	F1	8	10	29	22	38	67	38	69	2F
18	DA	F1	00	8	30	00	05	01	00	00	00	01
18	DA	00	F1	8	21	4F	2D	A8	43	C6	43	C8
18	DA	00	F1	8	22	30	8D	30	8A	38	0D	38
18	DA	00	F1	8	23	0E	38	1C	38	01	2A	93
18	DA	00	F1	8	24	25	85	37	FF	38	00	39
18	DA	00	F1	8	25	7E	39	7F	3C	6C	3C	6E
18	DA	F1	00	8	10	3D	62	38	67	00	38	69
18	DA	00	F1	3	30	03	00					
18	DA	F1	00	8	21	01	2F	4F	01	2D	A8	01
18	DA	F1	00	8	22	43	C6	01	43	C8	00	30
18	DA	F1	00	8	23	8D	00	30	8A	01	38	0D
18	DA	00	F1	3	30	03	00					
18	DA	F1	00	8	24	01	38	0E	01	38	1C	00
18	DA	F1	00	8	25	38	01	01	2A	93	01	25
18	DA	F1	00	8	26	85	01	37	FF	00	38	00
18	DA	00	F1	3	30	03	00					
18	DA	F1	00	8	27	00	39	7E	00	39	7F	00
18	DA	F1	00	8	28	3C	6C	01	3C	6E	01	07

18 = 110b priority 00b=edp/dp, DA00=15765 PGN 00=ECU, F1=tool, 3=datalen

8=datalen (note extra bytes ignored, see next slide)

Accelera

Trace 1
ISO 15765-2 network layer

18	DA	00	F1	3	**02**	3E	00					0=SF 2=SF_DL
18	DA	F1	00	8	**02**	7E	00	00	00	00	00	0=SF 2=SF_DL
18	DA	00	F1	4	**03**	22	86	CA				0=SF 3=SF_DL
18	DA	F1	00	8	**07**	62	86	CA	00	00	03	F6 0=SF 7=SF_DL
18	DA	00	F1	8	**10**	**29**	22	38	67	38	69	2F 1=FF 029=FF_DL
18	DA	F1	00	8	**30**	**00**	**05**	01	00	00	00	01 3=FC 0=FS (0continue/1wait/2overflow) 00=BS(00=all) 05=ST_min
18	DA	00	F1	8	**21**	4F	2D	A8	43	C6	43	C8 2=CF 1=SN
18	DA	00	F1	8	**22**	30	8D	30	8A	38	0D	38 2=CF 2=SN
18	DA	00	F1	8	**23**	0E	38	1C	38	01	2A	93 2=CF 3=SN
18	DA	00	F1	8	**24**	25	85	37	FF	38	00	39 2=CF 4=SN
18	DA	00	F1	8	**25**	7E	39	7F	3C	6C	3C	6E 2=CF 5=SN
18	DA	F1	00	8	**10**	**3D**	62	38	67	00	38	69 1=FF 03D=FF_DL
18	DA	00	F1	3	**30**	**03**	**00**					3=FC 0=FS (0continue/1wait/2overflow) 03=BS 00=ST_min
18	DA	F1	00	8	**21**	01	2F	4F	01	2D	A8	01 2=CF 1=SN
18	DA	F1	00	8	**22**	43	C6	01	43	C8	00	30 2=CF 2=SN
18	DA	F1	00	8	**23**	8D	00	30	8A	01	38	0D 2=CF 3=SN
18	DA	00	F1	3	**30**	**03**	**00**					3=FC 0=FS (0continue/1wait/2overflow) 03=BS 00=ST_min
18	DA	F1	00	8	**24**	01	38	0E	01	38	1C	00 2=CF 4=SN
18	DA	F1	00	8	**25**	38	01	01	2A	93	01	25 2=CF 5=SN
18	DA	F1	00	8	**26**	85	01	37	FF	00	38	00 2=CF 6=SN
18	DA	00	F1	3	**30**	**03**	**00**					3=FC 0=FS (0continue/1wait/2overflow) 03=BS 00=ST_min
18	DA	F1	00	8	**27**	00	39	7E	00	39	7F	00 2=CF 7=SN
18	DA	F1	00	8	**28**	3C	6C	01	3C	6E	01	07 2=CF 8=SN

Accelera

Trace 1
ISO 14229 UDS payload

18	DA	00	F1	3	02	3E	00						3E=TesterPresent 00=subfunction0
18	DA	F1	00	8	02	7E	00	00	00	00	00	00	7E=ACK 00=subfunction0
18	DA	00	F1	4	03	22	86	CA					22=ReadDataByIdentifier 86CA="ECU_VARIANT_ID"
18	DA	F1	00	8	07	62	86	CA	00	00	03	F6	62=ACK 86CA="ECU_VARIANT_ID" 000003F6=value
18	DA	00	F1	8	10	29	22	38	67	38	69	2F	22=ReadDataByIdentifier 3867=param1 3869=param2...
18	DA	F1	00	8	30	00	05	01	00	00	00	01	
18	DA	00	F1	8	21	4F	2D	A8	43	C6	43	C8	2F4F=param4 ...
18	DA	00	F1	8	22	30	8D	30	8A	38	0D	38	...
18	DA	00	F1	8	23	0E	38	1C	38	01	2A	93	...
18	DA	00	F1	8	24	25	85	37	FF	38	00	39	...
18	DA	00	F1	8	25	7E	39	7F	3C	6C	3C	6E	... 3C6E=paramN
18	DA	F1	00	8	10	3D	62	38	67	00	38	69	62=ACK 3867=param1 00=value1 3869=param2
18	DA	00	F1	3	30	03	00						
18	DA	F1	00	8	21	01	2F	4F	01	2D	A8	01	01=value2 2F4F=param3 ...
18	DA	F1	00	8	22	43	C6	01	43	C8	00	30	...
18	DA	F1	00	8	23	8D	00	30	8A	01	38	0D	...
18	DA	00	F1	3	30	03	00						
18	DA	F1	00	8	24	01	38	0E	01	38	1C	00	...
18	DA	F1	00	8	25	38	01	01	2A	93	01	25	...
18	DA	F1	00	8	26	85	01	37	FF	00	38	00	...
18	DA	00	F1	3	30	03	00						
18	DA	F1	00	8	27	00	39	7E	00	39	7F	00	...
18	DA	F1	00	8	28	3C	6C	01	3C	6E	01	07	...

Negative Response

NACK	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
Description	Negative Response SID -7F	Requested SID	Negative Response Code (NRC)	NA				

UDS Log file Class Exercise #1

Inspect the CAN trace on the paper sheet and address these questions:

1. What kind of diagnostics session is being used?
2. What is the hex code for the first DID requested by ID?
3. What was the line number for the message that requested a seed for the security service?
4. What is the length of the seed for the security service?
5. What is the value (in hex bytes) of the seed?
6. What is the key value?
7. After examining the values of the seed and key, can you identify any security issues?
8. What is the Routine ID (RID) for the Service Routine in the trace?
9. What is the reason for the negative acknowledgement for the Service Routine Request?
10. What line number started the Service Routine response?

UDS Trace A:

```
Line Num:        @timestamp:   PGN (3)   Pri  SA  DA  Eight Data Bytes
00000001: @00:20:15.187759: 00 df 00 06 f1 ff 3f ff ff ff ff ff ff ff
00000002: @00:20:15.201856: 00 da 00 06 f1 00 03 22 fd 02 00 00 00 00
00000003: @00:20:15.202148: 00 da 00 06 00 f1 04 62 fd 02 ff 8a 20 f0
00000004: @00:20:15.545797: 00 da 00 06 f1 00 03 22 fd 00 00 00 00 00
00000005: @00:20:15.546090: 00 da 00 06 00 f1 07 62 fd 00 00 00 04 1b
00000006: @00:20:15.548736: 00 da 00 06 f1 00 03 22 fd 02 00 00 00 00
00000007: @00:20:15.549032: 00 da 00 06 00 f1 04 62 fd 02 ff 8a 20 f0
00000008: @00:20:15.551770: 00 da 00 06 f1 00 02 10 03 00 00 00 00 00
00000009: @00:20:15.552064: 00 da 00 06 00 f1 06 50 03 00 32 01 f4 f0
00000010: @00:20:15.556743: 00 da 00 06 f1 00 02 10 02 00 00 00 00 00
00000011: @00:20:15.557041: 00 da 00 06 00 f1 03 7f 10 78 50 00 20 f0
00000012: @00:20:15.587414: 00 da 00 06 00 f1 06 50 02 00 32 01 f4 00
…
```

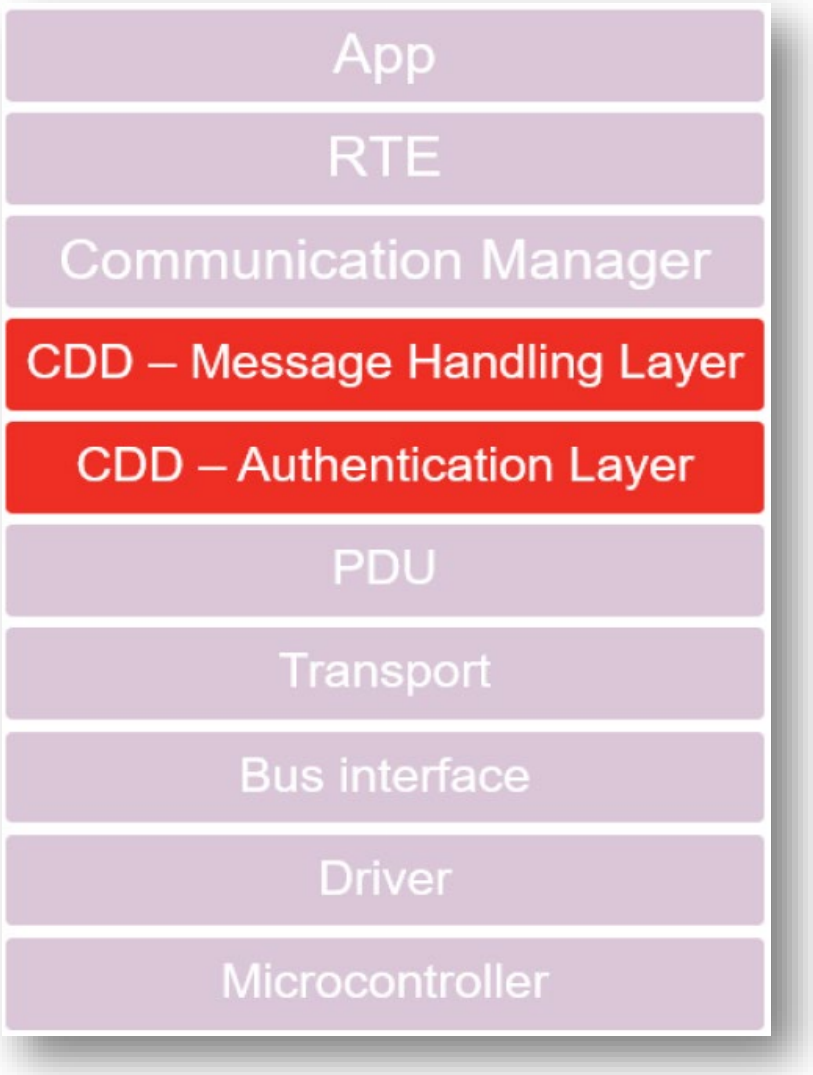
Securing Vehicle Diagnostic Communication

03

Author(s): Sharika Kumar, Jeremy Daily
Affiliated: Accelera by Cummins/Ohio State University, Colorado State University, Ohio State University

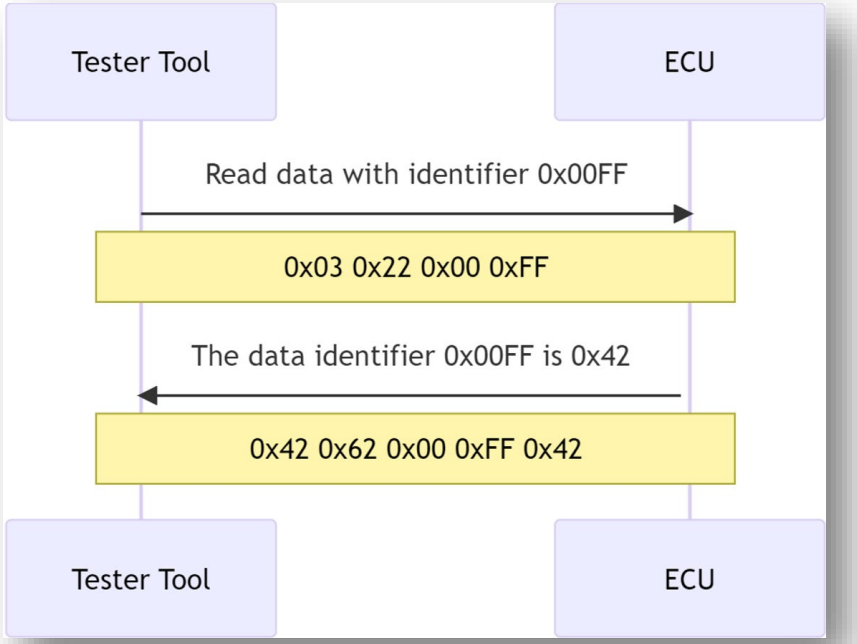
ESCAR USA 2023

Complex Device Drivers based Security Sublayer for UDS Security

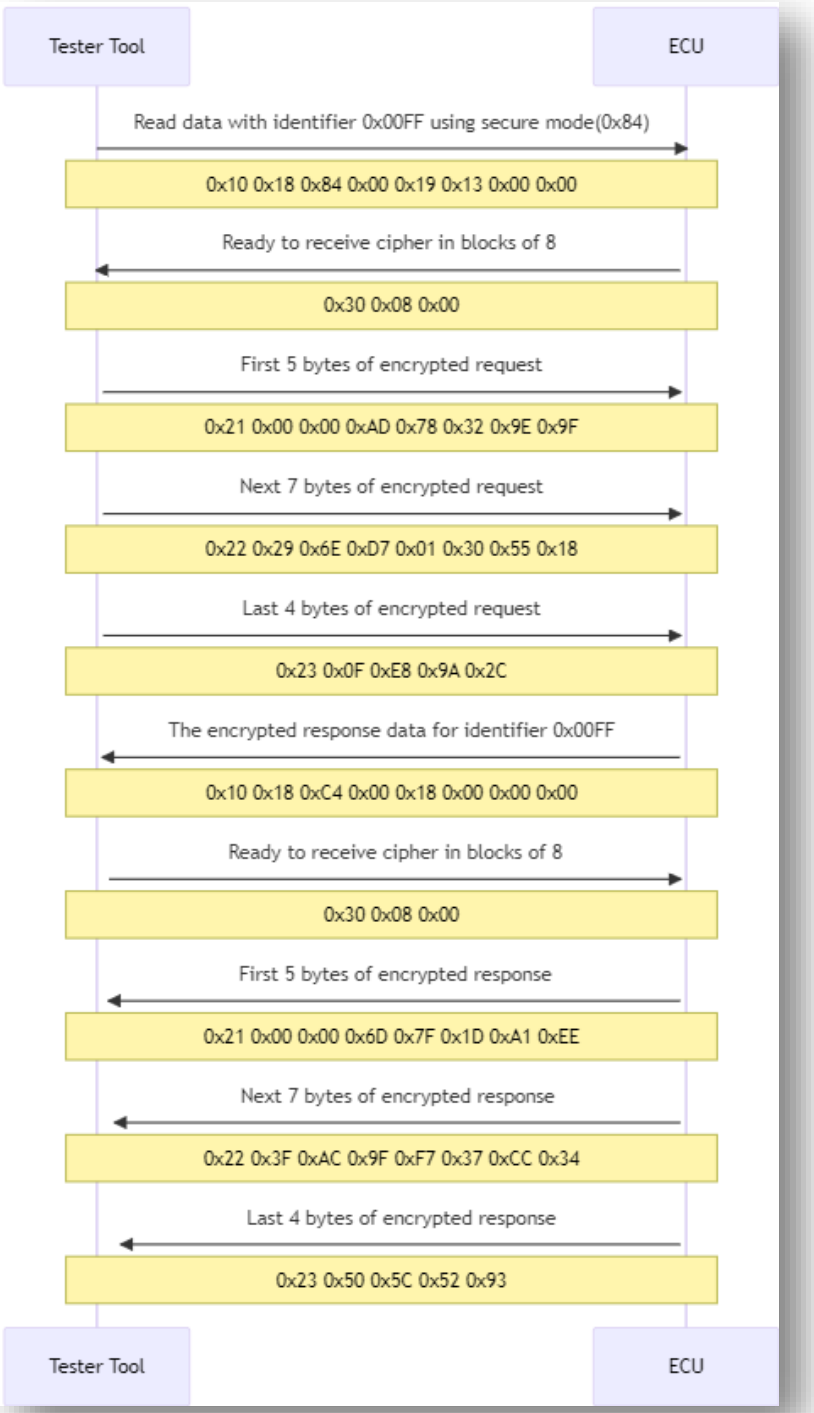


Accelera

Service 84$ Secure Data Transmission



Unsecured UDS Read Data by Identifier Service



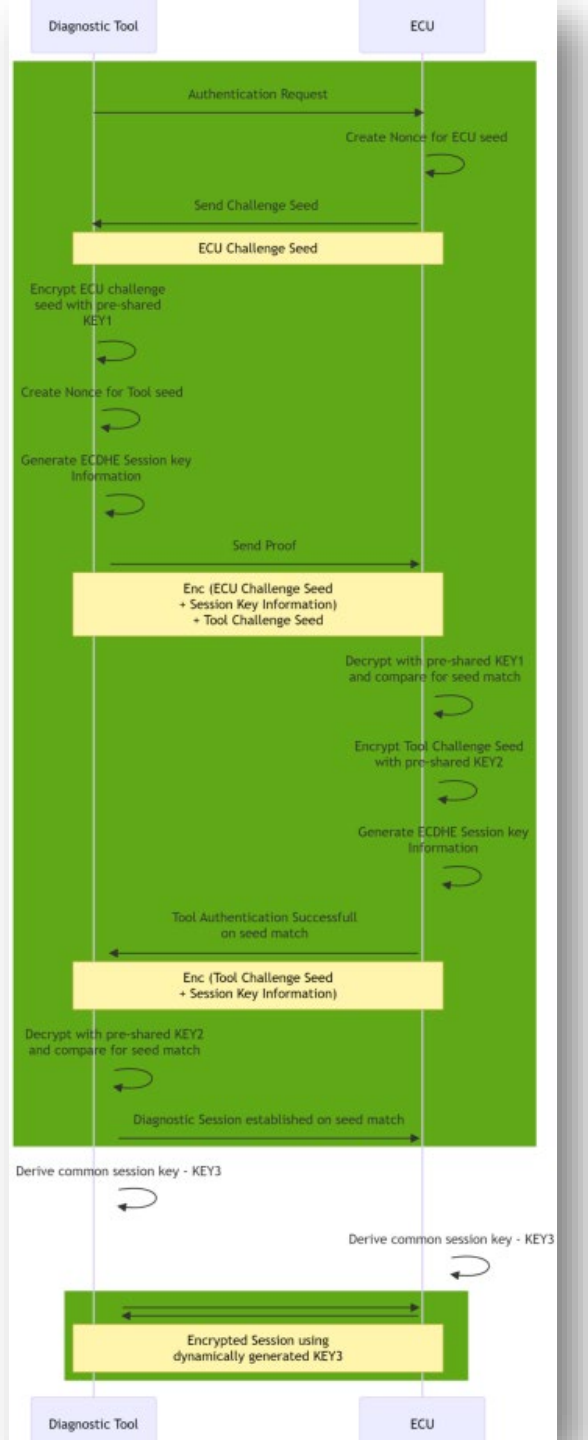
UDS Read Data by Identifier Service secured using UDS Secured Data Transmission Service

Accelera

UDS Session Authentication and Encryption

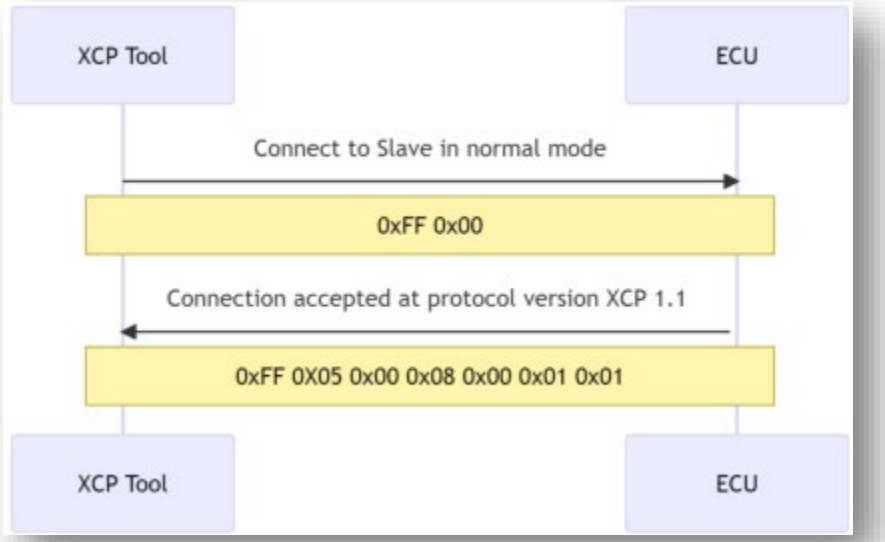
- Bi-Authentication
  - Example shows based on symmetric algorithm
  - Can also be performed using certificate based/asymmetric algorithms
  - Challenge response scheme

- Dynamic Session Key Derivation
  - Unique shared session key is derived using Diffie-Helman key exchange
  - Unique key per session



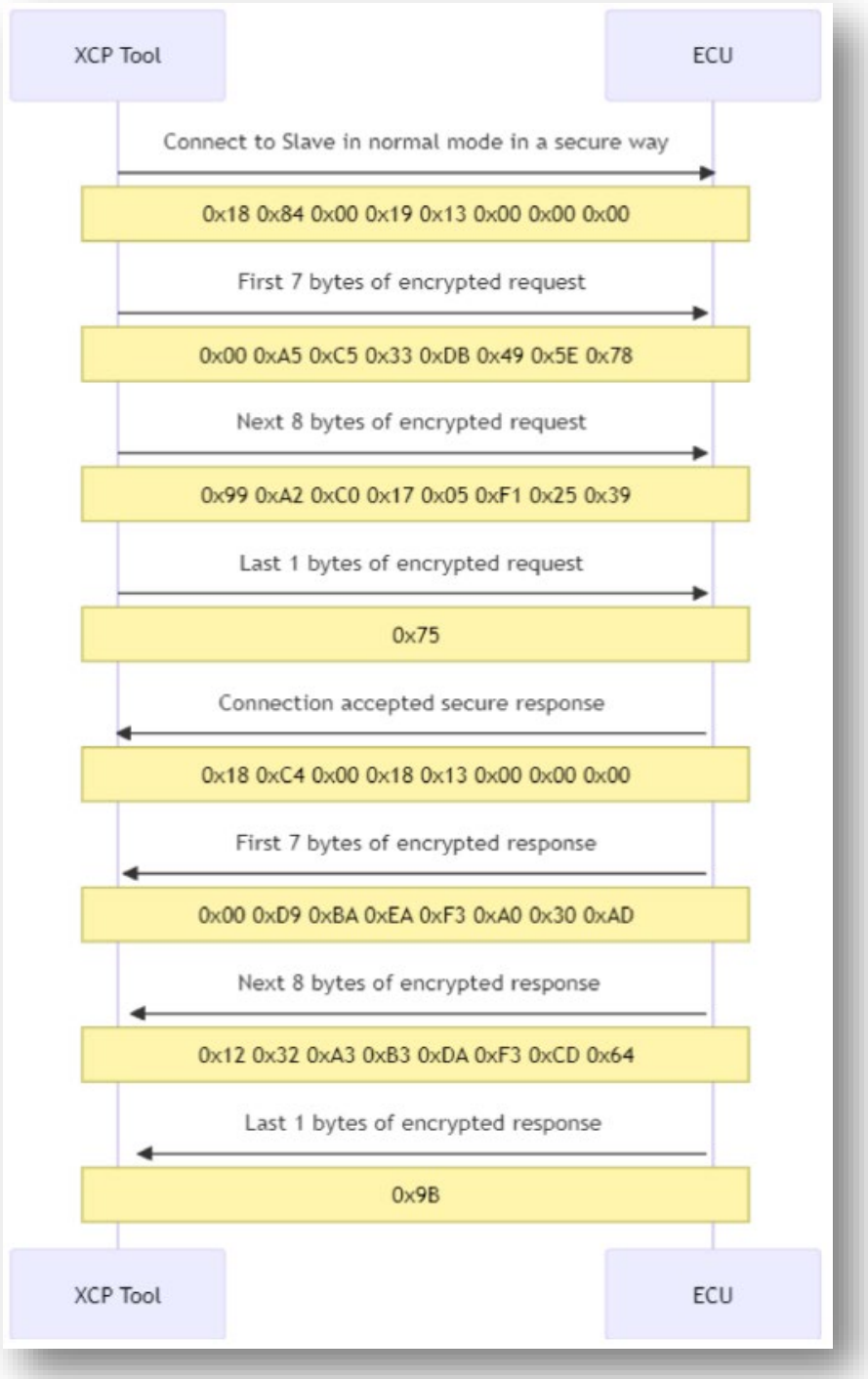
Accelera

Securing XCP Session



Unsecured XCP Connect Command

XCP Connect Command secured by expanding
UDS Secured Data Transmission Service



Cyber Defense for SAE J1939 Messages

- Security aspect that is compromised is integrity and confidentiality of SAE J1939 messages
- The basic idea of our defense proposal is to transmit a security validation message that the receiver can use to verify if the legitimate message is tampered with or not
- The receiver can simply discard the received frame if verification fails
- In the simplest form the security message could contain a MAC of the freshest or latest message transmitted out

Mitigating undetected message manipulation

Accelera

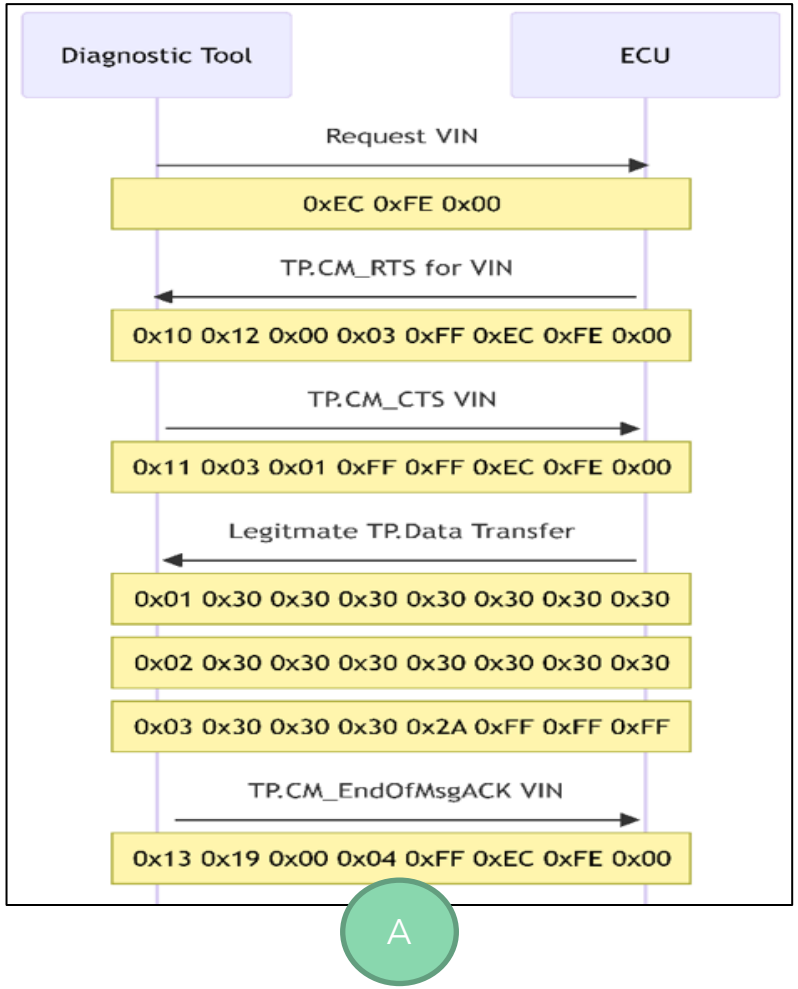
Cyber Defense for Diagnostic Interfaces

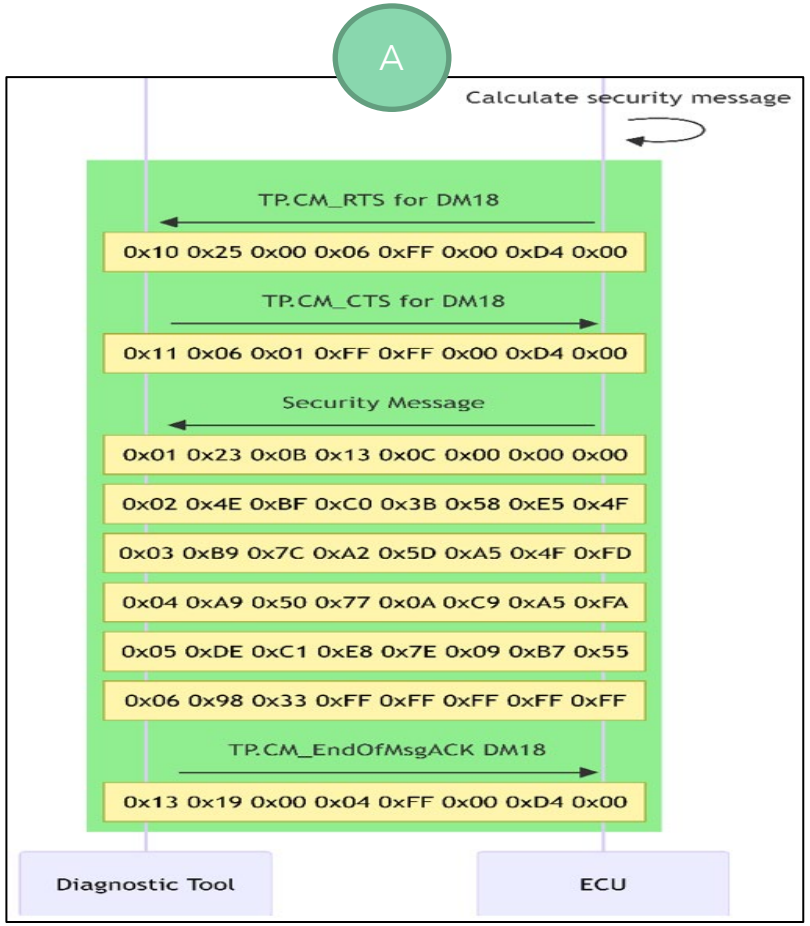
Byte Pos.	Bits	Definition (Existing in the SAE J1939-73)	Updates to existing definition
1	8-1(LSB)	Security Entity Length – Length of the data security parameter	
2	8-5(MSB)		
2	4-1	Security Entity Type – Indicating type of usage 0000 – Data is long seed 0001 – Data is long key 0010 – Data is a session key 0011 – Data is a certificate 0100 – 1111 - Reserved	1000 – Data is encrypted with pre-shared key 1001 – Data is signed with pre-shared key 1011 – Data is encrypted and signed with pre-shared key 1100 – Data is encrypted with dynamically derived key 1101 – Data is signed with dynamically derived key 1111 – Data is encrypted and signed with dynamically derived key
3	8-1	Data Security Parameter	Signature/Encryption Calculation – Contains an algorithm identifier
4-5	8-1		Signature Length – Length of signature portion of the message
6-7	8-1		Anti-replay Counter – Incrementing counter to prevent replay attack
8- n*	8-1		Message/Cipher
n+1 – m** n+ Signature Length	8-1		Signature

Data Security Message (Dm18) Updates for Defense

Accelera

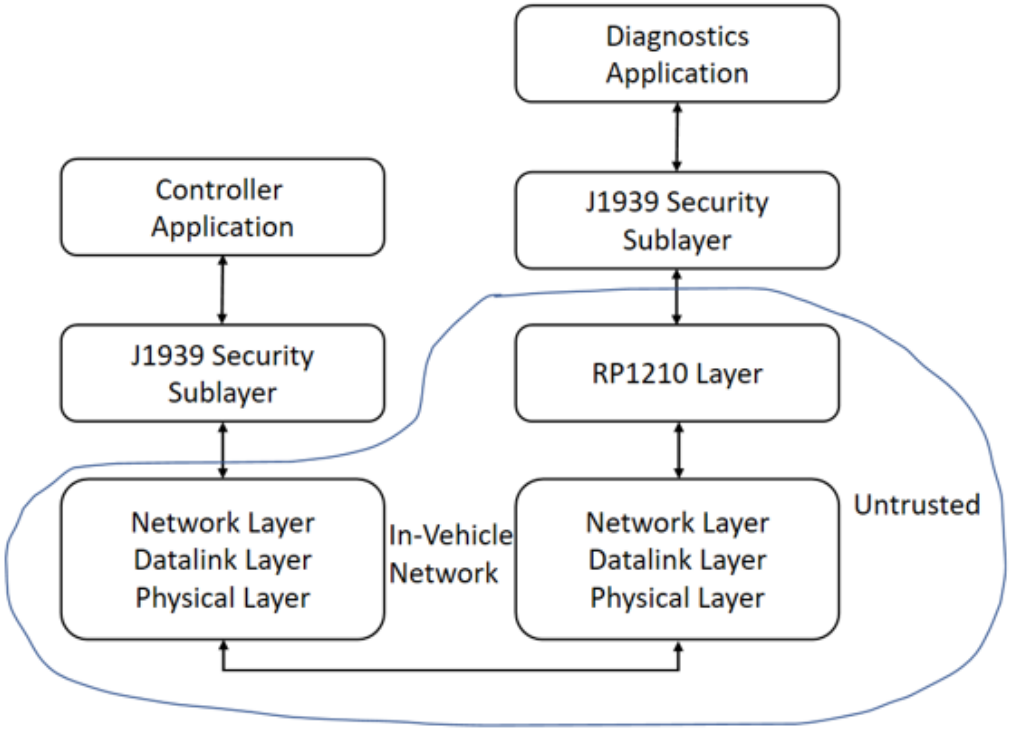
Cyber Defense using DM18





Sequence diagram that reflect log files showing the utility of DM18 to send secure messages over SAE J1939

Cyber Defense for Diagnostic Interfaces



Proposed security architecture were external layers are untrusted

Contact Info

Thank you

Sharika Kumar
Accelera by Cummins and Ohio State University
7018 Stoney Ridge Drive, Columbus, IN -47201
+1-812-341-0190
sharika.kumar@cummins.com
kumar.918@buckeyemail.osu.edu
sharikakkumar@gmail.com