The newest version of this slide deck and other related stuff can be found at

http://truckhacking.github.io/

Cheap Tools for Hacking Heavy Truck

By Haystack and Six Volts

What we are going to talk about

- Heavy Trucks: similarities and differences from cars
- R&D Problems: Trucks are expensive and the workaround
- Networking Protocols and Standards
- Adventures in truck hacking
- New Hardware Tools

Some Quick Notes

- We assume that you are familiar with basic vehicle networking concepts – e.g. there are computers in cars and they use a network

- We also assume you are familiar with the idea that you can do bad things once you are on those networks
- We are leaving out LOTS of details for time reasons
  - Check out our github
- Safety Disclaimer: Moving vehicles are dangerous. Do not fuzz a rental vehicle while driving, or do anything else stupid

Trucks vs. Cars

- “Trucks” are really any heavy vehicle including but not limited to Over-the-road Semis, Vocation Trucks, Fire Engines, Busses, some Armored Personnel Carriers, Ambulances, Armored cars, boats, diesel generators and agricultural equipment.
  - Exception: Diesel Pickup Trucks (these act more like cars)
- Nearly all heavy vehicles are Diesel engines.
- Different On-board Diagnostic and Networking Standards (J1939/J1708)
  - RP1210 governs workstation->adapter interface

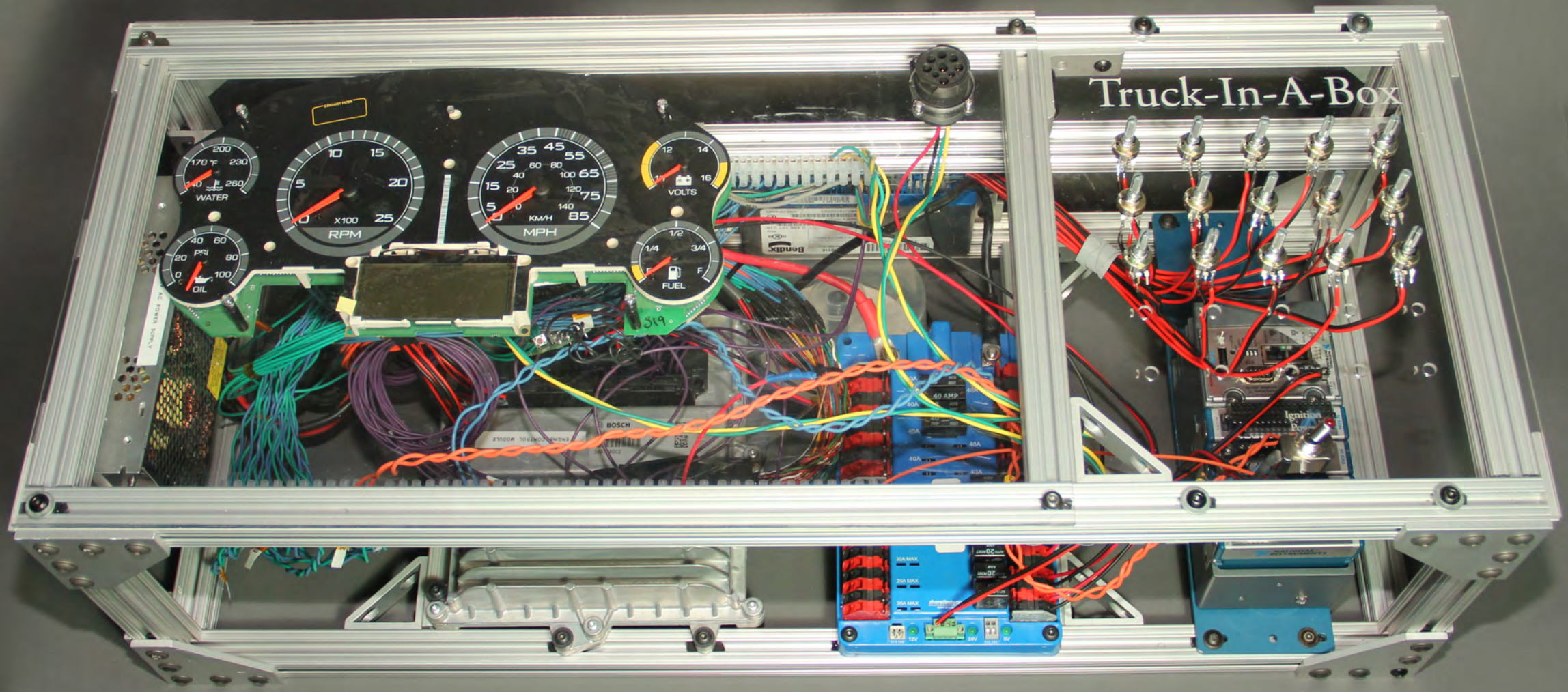
Truck Economics

- Many components from different manufacturers are interchangeable (engine, brakes, etc)
  - Example: Navistar/International Truck can be purchased with either a Cummins or International/Navistar Engine (and Previously CAT also)
  - This means that products from different manufacturers have to be interoperable
- Many trucks operate in Fleets, typically as homogenous as possible
- The industry is incredibly data hungry, lots of data are stored and transmitted
- Data hungry industry + lots of miles = trucks spend (comparatively) more time connected to diagnostic computers

Trucks are EXPENSIVE

- A new Truck can cost over $100,000. Ouch.
- For the aspiring hacker - They are big, hard to store, hard to drive and expensive to operate.
- So we didn’t have one (and still don’t)…
- …so how do we experiment? We built a thing.

Truck-In-A-Box, Version 1.0



Truck-In-A-Box (TIB)

- We bought an ECM (Engine Control Module) and built the electronics around it such that it functioned enough for analysis (Key-on, engine off)
- The first one took 6+ months and cost over $10,000
  - However, that's less than the cost of a truck

- Since then, we've built over a dozen of these full-size versions
- Later, we compressed the concept into small box with one or two PCBs that hook up to the ECM for each make/model

Truck-In-A-Box Concepts

- Recreate the Vehicle Networks, J1939 (CAN), J1708 (RS485-ish)
- Fake Passive sensor signals (usually just a set voltage or resistance)
- Fake Simple Active Signals (PWM for Accelerator Pedal)
- Generate Complex Analog Signals (Vehicle Speed)


Agilent
Acquisition
Normal
200MSa/s
Channels
DC 10.0:1
DC 1.00:1
DC 1.00:1
DC 1.00:1
Measurements
Freq(1):
903.44Hz
Pk-Pk(1):
33V
Measure Current Mean Min Max Std Dev Count
Freq(1): 903.44Hz 894.69Hz 293.07Hz 913.41Hz 43.599Hz 886
Pk-Pk(1): 33V 31.108V 14V 62V 10.407V 887
Measurement Statistics Menu
Display On
Reset Statistics
Transparent
Relative σ
Max Count
∞

Networking Protocols and Standards

- 2 main protocols: SAE J1939 and J1708
- J1708 is the old one (1985)
  - Based on 9600 baud UART
  - J1587 operates on top of it (transport layer)
- J1939 is the new one (?)
  - Physical & data link layers are 250K CAN
  - Addressing, transport, etc
- ISO15765 also used, but only for diagnostics comms
- (details in whitepaper)

J1708 basics

- 9600 baud serial
  - Can be read with a tty with a little work
- Messages are time delimited
- MIDs and PIDs
- Mostly older trucks will have only J1708
  - Some newer ones will have components using it
  - Also, gliders
- Data link escape for proprietary comms (PID 0xFE)
- Message fragmentation & reliable delivery (J1587)

J1939 Basics

- 250k CAN (500k in the near-ish future)
- Extended CAN ID broken into source, (maybe) destination, etc
- Address management, transport, message fragmentation
  - There's a bajillion different J1939 standards
- Also a PGN or two reserved for proprietary comms

VDA basics

- Vehicle diagnostics adapters
  - Similar in purpose to OBD-II scan tools
  - Basically USB/Serial/Ethernet -> J1939/J1708 brid
- Governs functions exposed by vehicle diagnostic adapters (VDAs)
- Best VDAs for RE are Dearborn Group DPA
  - Robust logging facilities allow for easy dynamic analysis
  - For now; we want to write a RP1210 driver for…

Truck Hacking Tools: Truck Duck

- Cape for a BeagleBone
  - Hardware for CAN and J1708
  - 2 of each for potential filtering/modification purposes
- We also have a software stack for doing comms
  - J1939 kernel extensions (plus J1939-enabled Python build)
  - Homegrown J1708 implementation using AM335x PRU (it is ugly)


DEF CON24
Truck Hacking Cape
@Six_Volts
LINK-PP
LPJ0011BNL
China M 1426
5V
P1
P2
P3
P4
J1
VR1
PTC1
TVS1
U1
U2
U3
U4
U6
U7
U8
R1
R2
R3
R4
R5
R6
R7
R8
R9
R10
R11
R12
R13
R14
R15
R16
C1
C2
C3
C4
C5
C6
C7
D1
Q1
S1
TP4
TP5
TP6

Adventures in Truck Hacking

Screwing with engine parameters

- Most engine parameter configuration is done over proprietary protocol extensions
- Pretty easy to reverse
  - Most OEM software is un-obfuscated .NET linked to some legacy C
- We super promised not to give *too* many specifics
- Demonstration of what is possible with TruckDuck

Engine parameter modification demo

- <demo goes here>

ECM impersonation

- Useful for reversing proprietary comms parameters
  - (details later)

Bad Crypto A Go Go

- (disclosed at con)

- More demos to come probably!

Heads up:

There is a ton of related material on our github including a white paper, schematics, assembly instructions, code, and embedded OS image.

truckhacking.github.io