Reverse Engineering the Tesla Battery Management System to Increase Power Available

By Patrick Kiley
Patrick Kiley – Principal Security Consultant - Rapid7

- Member of the Penetration Testing team at Rapid7
- Performed research in Avionics security
- Internet connected transportation platforms
- Experience in hardware hacking, IoT, Autonomous Vehicles, and CAN bus
Topics

• Architecture of the Model S and Battery Management System (BMS)
• Performance and Ludicrous timeline
• Hardware changes
• Data stored in toolbox
• Firmware changes
• Shunt modification
• Upgrade process
• Failure and what I learned
• Next steps
Model S Architecture

- Central Information Display (CID): Nvidia Tegra based
- Gateway: a security component, stores vehicle configuration, sits between the various CAN buses and the CID
- Powertrain (PT) CAN bus, contains the BMS, Drive units, charging, thermal control and other powertrain related controllers
- PT CAN runs at 500 kBit/sec and is a standard vehicle CAN bus (differential signaling, 11 bit arbitration ids, etc)
- PT CAN supports UDS standard
**BMS Overview**

- TI TMS320C2809 – Main microprocessor
- Altera CPLD – Hardware backup for TMS320
- Current Shunt with STM8, measures current coming from the battery
- Precharge Resistor, prevents inrush current damage
- BMB boards on each battery pack, these include bleed resistors to balance packs

All the firmware changes are on the TMS320

Some settings are changed on the shunt, in addition it has a small physical modification

Full reversing of all the components is an ongoing project, so if you want to help, I am lacking in some of the skill areas
Ludicrous History

• P85D announced on Oct 10, 2014
• Ludicrous announced on July 17, 2015
• 10K for new buyers, 5K for existing P85D owners
• Upgrade involved new contactors and pyro fuse.
• Many performance battery packs would come standard with new components
• They were “ludicrous capable”
• All 100kWh performance battery packs are “ludicrous capable”
• Ludicrous capable means add “performanceaddon 1” to internal.dat on the gateway
I upgraded a donor vehicle
Pack Dropped
Fuse and Contactor Bay
Shunt and Contactor Close up
What about firmware?

- For this we need to dig into some python
- Tesla makes a diagnostic tool called toolbox, runs on windows, uses encrypted and compiled python modules
- The important files are contained as individual plugins with the .scramble extension
- All of the information needed to decrypt the scramble files are on a machine that is running toolbox
- Some of these scramble files include firmware as well as many other useful items
- Once decrypted, we can use Uncompyle6 to give us source code
- Tesla left all the source code comments in place. Thank you!

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>hci-2018.0.6-win32.scramble</td>
<td>SCRAMBLE File</td>
<td>548 KB</td>
</tr>
<tr>
<td>tbox_chodemo-2018.0.1-win32.scramble</td>
<td>SCRAMBLE File</td>
<td>311 KB</td>
</tr>
<tr>
<td>tbox_coyote_carr-2018.0.1-win32.scramble</td>
<td>SCRAMBLE File</td>
<td>140,060 KB</td>
</tr>
<tr>
<td>tbox_dev-2018.0.1-win32.scramble</td>
<td>SCRAMBLE File</td>
<td>184 KB</td>
</tr>
<tr>
<td>tbox_driver_assist-2018.0.3-win32.scramble</td>
<td>SCRAMBLE File</td>
<td>142 KB</td>
</tr>
<tr>
<td>tbox_engineering-2018.0.1-win32.scramble</td>
<td>SCRAMBLE File</td>
<td>232 KB</td>
</tr>
<tr>
<td>tbox_fw_update_06-2018.0.1-win32.scramble</td>
<td>SCRAMBLE File</td>
<td>458 KB</td>
</tr>
<tr>
<td>tbox_fw_update_07-2018.0.2-win32.scramble</td>
<td>SCRAMBLE File</td>
<td>19 KB</td>
</tr>
<tr>
<td>tbox_gen2_firmware-2018.0.1-win32.scramble</td>
<td>SCRAMBLE File</td>
<td>22,107 KB</td>
</tr>
<tr>
<td>tbox_gen2_meta-2018.0.1-win32.scramble</td>
<td>SCRAMBLE File</td>
<td>3,415 KB</td>
</tr>
<tr>
<td>tbox_gen2-2018.36.26-win32.scramble</td>
<td>SCRAMBLE File</td>
<td>2,780 KB</td>
</tr>
<tr>
<td>tbox_key_pairing-2018.0.4-win32.scramble</td>
<td>SCRAMBLE File</td>
<td>1,102 KB</td>
</tr>
<tr>
<td>tbox_meta_18_2_23-18.2.23-win32.scramble</td>
<td>SCRAMBLE File</td>
<td>13,573 KB</td>
</tr>
<tr>
<td>tbox_rest-2018.0.3-win32.scramble</td>
<td>SCRAMBLE File</td>
<td>1,205 KB</td>
</tr>
<tr>
<td>tbox_restraint-2018.0.1-win32.scramble</td>
<td>SCRAMBLE File</td>
<td>1,341 KB</td>
</tr>
<tr>
<td>tbox_security-2018.0.3-win32.scramble</td>
<td>SCRAMBLE File</td>
<td>102 KB</td>
</tr>
<tr>
<td>tbox_service-2018.33.4-win32.scramble</td>
<td>SCRAMBLE File</td>
<td>1,257 KB</td>
</tr>
<tr>
<td>tbox_steering-2018.0.1-win32.scramble</td>
<td>SCRAMBLE File</td>
<td>75 KB</td>
</tr>
<tr>
<td>tbox_suspension-2018.0.1-win32.scramble</td>
<td>SCRAMBLE File</td>
<td>38 KB</td>
</tr>
<tr>
<td>tbox_testing-2018.35.1-win32.scramble</td>
<td>SCRAMBLE File</td>
<td>48 KB</td>
</tr>
<tr>
<td>tbox_third_party-2018.0.2-win32.scramble</td>
<td>SCRAMBLE File</td>
<td>5,381 KB</td>
</tr>
<tr>
<td>tbox_usb-2018.0.1-win32.scramble</td>
<td>SCRAMBLE File</td>
<td>92 KB</td>
</tr>
</tbody>
</table>
Toolbox Uncompyled

```python
# uncompyle6 version 3.3.2
# Python bytecode 2.7 (62211)
# [GCC 5.4.0 20160609]
# Embedded file name: build\bdist.win32\egg\vehicle\core\uds\data.py
# Compiled at: 2017-07-27 11:08:07
__author__ = 'Eric Hulser'
__email__ = 'ehulser@teslamotors.com'
__copyright__ = 'Copyright Tesla Motors Inc. 2013'
import logging
from PyQt import QtCore
from .object import UdsObject
from . import errors
log = logging.getLogger(__name__)

class UdsData(UdsObject):
```

#BHUSA  @BLACKHATEVENTS
Defines the a VehicleTest to change the performance addon config.

```python
__authors__ = [
    'Otto Chiu']
__author__ = (',').join(__authors__)
__credits__ = []
__copyright__ = 'Copyright Tesla Motors Inc. 2015'
from tux_gen2.test import Gen2VehicleTest
import logging
log = logging.getLogger(__name__)

class PerformanceAddonConfig(Gen2VehicleTest):
    def exec(self):
        """
        First verify that a vehicle can be configured in the desired config.
        For Standard mode, there is no checks; for Ludicrous mode, the vehicle
        needs to be AWD and has a battery pack config that supports 1500A+ current discharge.
        """
```
Data Structures – Extract and Binwalk

# uncompyle6 version 3.3.2
# Python bytecode 2.7 (62211)
# Decompiled from: Python 2.7.12 (default, Nov 12 2018, 14:36:49)
# [GCC 5.4.0 20160609]
# Embedded file name: build\bdist.win32\egg\tbx_gen2_firmware\resources\pyside_tbx_gen2_firmware_rc.py
# Compiled at: 2018-01-23 14:15:28
from xqt import QtCore
qt_resource_data = '
qt_resource_name = '
qt_resource_struct = '
def qInitResources():
    QtCore.qRegisterResourceData(1, qt_resource_struct, qt_resource_name, qt_resource_data)
def qCleanupResources():
    QtCore.qUnregisterResourceData(1, qt_resource_struct, qt_resource_name, qt_resource_data)
qInitResources()
Bootloader

We already know from the donor vehicle’s config that it had a pack id of “57”

These are the files we need from the extracted firmware

Pack id 57 becomes pack id 70 after the changes

```python
# Changes HWID from 57 to 70
# http://artifacts.teslamotors.com/jenkins-job/bootloaders/git/21c44cbe0713a1beaa105b00c1

- name: 57 Gateway Application
description: HWID 57 Gateway Application for Shunt Calibration
filename: firmware/bms/withSecondaryBoot_UDSBoot_BMS_GATEWAY_APP_HWID-57.hex
ludicrousable: True
hwid: 57
calibrateShunt: True
linkedPackageName: 57 Updater

- name: 57 Updater
description: HWID 57 Updater File
filename: firmware/bms/withSecondaryBoot_UDSBoot_BMS-R57-CSM_UPDATER_SVN-68454.hex
flashBootUpdater: True
linkedPackageName: 70 Bootloader
ludicrousable: True

- name: 70 Bootloader
description: HWID 70 Bootloader
filename: firmware/bms/withSecondaryBoot_UDSBoot_BMS-R70-CSM_SVN-71214.hex
changeBootHWID: True
module: UDS_FLASH_BOOTLOADER
linkedPackageName: 70 Application
```
Firmware Upgrade

- All the instructions and files needed for the upgrade process were stored in Toolbox files.
- DBC files to help understand signals on the PT CAN bus.
- ODX files that defined how to calibrate the shunt, grant security access and upgrade the firmware.
- Files that stored calibration data and firmware.
- Text comments and text data structures that offered clues on the process.
CAN and UDS

Sitting on top of the CAN network stack is a protocol called UDS, or “Unified Diagnostic Services”, this protocol can be used to help technicians:
- Diagnose problems
- Read values from sensors
- Update firmware

CAN networks use a descriptor file called a DBC file

UDS networks can use a scripting file called ODX or GMD

Used commercial tool Vehicle Spy to assist in the research

ARBS 7E2 and 202 from BMS identify max current as a static value. 232 (BMS), 266 (DI) and 2E5 (DIS), identify max power in watts, which varies based on SOC, temp, and power recently used.
DBC Turns this

<table>
<thead>
<tr>
<th>Filter</th>
<th>Count</th>
<th>Time (abs/el)</th>
<th>Tx</th>
<th>Er</th>
<th>△ Description</th>
<th>ArbId/Itx</th>
<th>Len</th>
<th>DataBytes</th>
<th>Network</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>10.000 ms</td>
<td></td>
<td></td>
<td>HS CAN $102</td>
<td>102</td>
<td>8</td>
<td>00 00 32 A5 64 4C 08 00</td>
<td>HS CAN</td>
</tr>
<tr>
<td></td>
<td>360</td>
<td>100.006 ms</td>
<td></td>
<td></td>
<td>HS CAN $202</td>
<td>202</td>
<td>8</td>
<td>80 9D 1E 98 00 00 00 00</td>
<td>HS CAN</td>
</tr>
<tr>
<td></td>
<td>360</td>
<td>100.004 ms</td>
<td></td>
<td></td>
<td>HS CAN $212</td>
<td>212</td>
<td>5</td>
<td>40 47 70 FF 00</td>
<td>HS CAN</td>
</tr>
<tr>
<td></td>
<td>36</td>
<td>1.00002 s</td>
<td></td>
<td></td>
<td>HS CAN $218</td>
<td>218</td>
<td>8</td>
<td>7D 00 FF 27 FF DF FF 37</td>
<td>HS CAN</td>
</tr>
<tr>
<td></td>
<td>360</td>
<td>100.238 ms</td>
<td></td>
<td></td>
<td>HS CAN $222</td>
<td>222</td>
<td>6</td>
<td>00 00 00 00 00 00 00 10</td>
<td>HS CAN</td>
</tr>
<tr>
<td></td>
<td>360</td>
<td>100.474 ms</td>
<td></td>
<td></td>
<td>HS CAN $232</td>
<td>232</td>
<td>4</td>
<td>88 26 3E 07</td>
<td>HS CAN</td>
</tr>
<tr>
<td></td>
<td>36</td>
<td>1.00020 s</td>
<td></td>
<td></td>
<td>HS CAN $242</td>
<td>242</td>
<td>6</td>
<td>00 00 00 00 00 00 00 00</td>
<td>HS CAN</td>
</tr>
<tr>
<td></td>
<td>35</td>
<td>646 µs</td>
<td></td>
<td></td>
<td>HS CAN $246</td>
<td>246</td>
<td>5</td>
<td>DE 00 F5 03 49</td>
<td>HS CAN</td>
</tr>
<tr>
<td></td>
<td>360</td>
<td>99.986 ms</td>
<td></td>
<td></td>
<td>HS CAN $248</td>
<td>248</td>
<td>7</td>
<td>2B 2B 02 02 02 4B 01 31</td>
<td>HS CAN</td>
</tr>
<tr>
<td></td>
<td>360</td>
<td>101.181 ms</td>
<td></td>
<td></td>
<td>HS CAN $252</td>
<td>252</td>
<td>7</td>
<td>00 00 00 00 02 2B 00</td>
<td>HS CAN</td>
</tr>
<tr>
<td></td>
<td>360</td>
<td>101.417 ms</td>
<td></td>
<td></td>
<td>HS CAN $262</td>
<td>262</td>
<td>8</td>
<td>00 00 00 00 00 00 00 00</td>
<td>HS CAN</td>
</tr>
<tr>
<td></td>
<td>360</td>
<td>101.655 ms</td>
<td></td>
<td></td>
<td>HS CAN $272</td>
<td>272</td>
<td>8</td>
<td>00 E0 43 FF 00 00 00 00</td>
<td>HS CAN</td>
</tr>
<tr>
<td></td>
<td>362</td>
<td>40.006 ms</td>
<td></td>
<td></td>
<td>HS CAN $28F</td>
<td>28F</td>
<td>8</td>
<td>F0 FF FF FF FF FF 6F 61</td>
<td>HS CAN</td>
</tr>
<tr>
<td></td>
<td>360</td>
<td>100.002 ms</td>
<td></td>
<td></td>
<td>HS CAN $2C8</td>
<td>2C8</td>
<td>8</td>
<td>60 80 03 FF 20 24 20 40</td>
<td>HS CAN</td>
</tr>
<tr>
<td></td>
<td>360</td>
<td>101.664 ms</td>
<td></td>
<td></td>
<td>HS CAN $2D2</td>
<td>2D2</td>
<td>7</td>
<td>00 00 00 00 00 00 00 00</td>
<td>HS CAN</td>
</tr>
<tr>
<td></td>
<td>36</td>
<td>1.000022 s</td>
<td></td>
<td></td>
<td>HS CAN $302</td>
<td>302</td>
<td>8</td>
<td>00 00 10 00 FF 67 40 00</td>
<td>HS CAN</td>
</tr>
</tbody>
</table>
## Into This

<table>
<thead>
<tr>
<th>Filter</th>
<th>Count</th>
<th>Time (abs/rel)</th>
<th>Tx</th>
<th>Er</th>
<th>Description</th>
<th>ArbId/He...</th>
<th>Len</th>
<th>DataBytes</th>
<th>Network</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>48</td>
<td>1.000973 s</td>
<td></td>
<td></td>
<td>BMS_energyStatus</td>
<td>382</td>
<td>8</td>
<td>5C 00 00 00 00 00 A0 10</td>
<td>HS CAN</td>
</tr>
<tr>
<td></td>
<td>4326</td>
<td>10.000 ms</td>
<td></td>
<td></td>
<td>BMS_hvBusStatus</td>
<td>102</td>
<td>8</td>
<td>00 00 32 A6 64 4C 08 00</td>
<td>HS CAN</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>10.001053 s</td>
<td></td>
<td></td>
<td>BMS_iSensorInfo</td>
<td>532</td>
<td>8</td>
<td>00 00 00 00 00 00 00 00</td>
<td>HS CAN</td>
</tr>
<tr>
<td></td>
<td>31</td>
<td>1.000975 s</td>
<td></td>
<td></td>
<td>BMS_info</td>
<td>5D2</td>
<td>8</td>
<td>DD 00 00 00 A3 2A A2 39</td>
<td>HS CAN</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>1.001201 s</td>
<td></td>
<td></td>
<td>BMS_kWhCounter</td>
<td>3D2</td>
<td>8</td>
<td>17 6E 68 00 31 DB 61 00</td>
<td>HS CAN</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>10.001059 s</td>
<td></td>
<td></td>
<td>BMS_odometerStatus</td>
<td>562</td>
<td>4</td>
<td>DC D4 45 02</td>
<td>HS CAN</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>100.003 ms</td>
<td></td>
<td></td>
<td>BMS_powerAvailable</td>
<td>232</td>
<td>4</td>
<td>88 26 5E 07</td>
<td>HS CAN</td>
</tr>
</tbody>
</table>

BMS_maxRegenPower = 98.640 kW [2688]
BMS_maxDischargePower = 18.860 kW [7SE]

#BHUSA  @BLACKHATEVENTS
ODX routines for shunt calibration
Shunt Modification

- Shunt also needed a hardware modification
- Single wire connecting the shunt to the CPLD
- If this wire remained connected after the firmware update then the BMS would generate an alert and refuse to close the contactors
- Discovered ran through the upgrade process on a bench version of the components
- Made a breakout board to monitor the signals from the shunt
- This also meant that the hardware and firmware both had to be updated before the car was driven
Upgrade Process

- Had access to garage and lift in Southern California
- Drove there to do upgrade, arrive with low SOC
- Drop pack, do hardware stuff
- Reinstall pack, carefully (image is from borescope)
- Flash BMS with special firmware for shunt modification
- Flash BMS to new packID
- Update internal.dat to add ludicrous and change packID
- Redeploy firmware due to changed battery packID
- Drive away and enjoy the ridiculous amount of torque?
Final Steps

• Using known techniques that I have used before, I tried to redeploy the firmware, also tried to upgrade since I had access to several versions
• The car failed using every method I tried
• Had to Tow the car from Rancho Cucamonga to Las Vegas so I could continue to work on it
• Cost me $360 or 3.6 hundred dollars, not great, not terrible right?
Learned something cool

- Gateway uses a file called firmware.rc
- Gateway uses this as a validation check for the components
- Calculated during upgrade/redeploy
- When the BMS changed, so did its CRC
- Changed the CRC based on CAN and value from "signed_metadata_map.tsv"
- Final CRC line is a JAMCRC based on overall file
- Car woke up, errors cleared and car could be driven
- Eventually figured out the reason for the earlier failure
Power Before and After Upgrade

Before Upgrade
1300 Amps

After Upgrade
1500 Amps

Actual
Available
Why Lower?
Further Research

- TMS320F2809 is supported in IDA Pro
- ARBS 7E2 and 202 define max current
- Seems possible to increase speed beyond ludicrous, it has been done by others (1000 HP RWD P85)
- Just need to find the variables and “bump them up a bit”, also might need to modify DU firmware
- Would be extremely dangerous to do so
- Could end up blowing up the Drive unit or battery pack, or worse, cause a fire and injury
- Still it would be interesting to reverse engineer, hit me up if you would like to assist, I have a dug a lot deeper than the information I am presenting here
- Would like to understand shunt parameters CAU1, CGI1
Referenced Material, Acknowledgements

Spaceballs movie, inspiration for Tesla Ludicrous https://www.imdb.com/title/tt0094012/
Ludicrous announcement and P85D upgrade offer  https://www.tesla.com/blog/three-dog-day
What is a current shunt? https://youtu.be/j4u8fl31sgQ (electroboom)
TMS320 datasheet https://www.ti.com/product/TMS320F2809
Intrepid Control Systems, makers of Vehicle Spy software https://intrepidcs.com/
Bitbuster, for allowing use of lift and garage
The people who helped with the Toolbox reversing, you know who you are
Tesla security team for letting me do this talk