Cybersecurity Considerations for Heavy Vehicle Event Data Recorders

Jeremy S. Daily
Department of Mechanical Engineering
The University of Tulsa
800 S. Tucker Dr.
Tulsa, OK 74104
jeremy- daily@utulsa.edu

Ben Gardiner
Irdeto Canada
2500 Solandt Rd.
Suite 300
Ottawa, ON K2K 3G5
ben.gardiner@irdeto.com

June 2018

Trust in the digital data from heavy vehicle event data recorders (HVEDRs) is paramount to using the data in legal contests. Ensuring the trust in the HVEDR data requires an examination of the ways the digital information can be attacked, both purposefully and inadvertently. The goal or objective of an attack on HVEDR data will be to have the data omitted in a case. To this end, we developed an attack tree and establish a model for violating the trust needed for HVEDR data. The attack tree provides context for mitigations and also for functional requirements. A trust model is introduced as well as a discussion on what constitutes forensically sound data. The main contribution of this paper is an attack-tree based model of both malicious and accidental events contributing to compromised EDR data. A comprehensive list of mitigations for HVEDR systems results from this analysis.

Introduction

Heavy vehicle event data recorders (HVEDRs) are computing applications running installed into heavy vehicles that record operational states of the vehicle when triggered. Triggers may include speed changes, final stops, diagnostic trouble codes (DTCs), manual triggers, and software/network triggers. EDRs could be standalone devices with access to the data on the vehicle's communication network or built into existing electronic control units (ECUs). EDR data has application in a forensic context, since recording triggers may be crash events. The western legal system uses the event data reported from these ECUs as forensic evidence. However, the utility of HVEDR data is not limited to the courts.

For this paper, we assume a potentially interesting digital record is present in the memory of an electronic control unit. A forensic investigator (FI) may be able to use that data when investigating the scenario surrounding the creation of that data.

There are unique considerations for digital data used in a forensic context. The end use of the data may be presented in a court of law many years after the incident. The data must be discoverable and shared among the litigants. Even if data is considered to be proprietary or sensitive, the court may order or compel production and interpretation of the evidence. Because of the unique application, the digital data must be trusted and verifiable by all parties. Establishing the trust of the data can be challenging and perhaps expensive. For

example, a heavy vehicle in a crash is often inspected after the event. Since the digital data may be challenging to recover and preserve, all parties attend the inspections– with some inspectors acting as observers of other inspectors. The reason for the additional labor stems from a lack of trust. Establishing trust in a forensic context is paramount to both the legal and engineering process.

Trust in the engineering opinions expressed by forensic investigators is established through scientific principles and evidence. Chain of custody rules and procedures for handling physical evidence are well established in the forensic community. With the proliferation of digital evidence, forensic processes and models for establishing trust in digital data becomes paramount. To establish trust we must understand how it can be violated. Towards the purposes of this paper, we explain digital forensic trust models for HVEDR data and examine potential threats to that trust using a cybersecurity tool known as an attack tree [1].

Attack trees can help us understand the cybersecurity posture of the HVEDR digital forensic process. Once the attack tree is generated, mitigation strategies can be proposed to bolster the forensic process and ensure higher levels of trust in the data from the HVEDR. This is a contemporary cybersecurity assessment process, where security requirements for the system are derived from a model of the threats to the system [2] . Furthermore, the modeling of impact can be combined with this to also assess the relative value of the security requirements. This method is found in [3].

For the purposes of this paper, we will not go into the detail needed to complete such an assessment of the relative value of the mitigations. We will consider only the attack model which will yield security requirements without ordering. If we were to also identify the stakeholders (those who stand to be impacted by attacks), model their interests in the process, and assess the impacts of failures of the process, then a risk assessment is also possible. Risk is quantified by combining the likelihood of malicious events occurring and the consequences of those events. Risk assessments are beyond the scope of the attack tree presented herein.

Cybersecurity and Digital Forensics

Establishing trust in digital evidence is a cybersecurity problem. Tampering with physical evidence is often detectable though classic detective work (i.e. photographic records, journal entries, and chain-of-custody procedures); however, detecting altered digital data requires sound mathematics and robust algorithms, like cryptographic hashing. As such, the principles of cybersecurity are inseparable from digital forensics.

Digital data used in a forensic context must be forensically sound. Forensic soundness, as originally enumerated by McKemmish [4] and applied to heavy vehicles in [5] makes use of the following tenets:

- **Meaning**: the digital evidence has an accurate representation of the event it is describing.
- **Error Detection**: a process for detecting and predicting evidence corruption.
- **Transparency**: algorithms and processes are known and verifiable.
- **Expertise**: the testifying expert can put the digital evidence into the proper context.
- **Tamper Resistance**: being able to demonstrate the original digital evidence was not altered.

The meaning of the digital forensic data in terms of forensic soundness is based on the assumption that the digital data produced by the originating source accurately reflects the physical event that is being recorded. It is important to realize that this assumption may still lead to challenges in interpreting the data. Therefore, third party independent testing of the HVEDR data is recommended to verify the digital records against external reference measurements.

Modeling the Forensic Process for HVEDRs

Data Life Cycle and Timeline

The forensic process for HVEDRs starts with data creation and ends with case adjudication. This process can last for years and involve many stakeholders with life altering consequences (i.e. incarceration or financial impact) compelled by the courts. Since trust of the data is critical in the final presentation, modeling the forensic process for HVEDR data enables identification of potential vulnerabilities of the digital records during the life of the data, which is depicted in the timeline and flow diagram shown in Figure 1. This timeline models the majority of the process towards the beginning of the EDR record, but does not attempt to model the utility of the data in the courts.

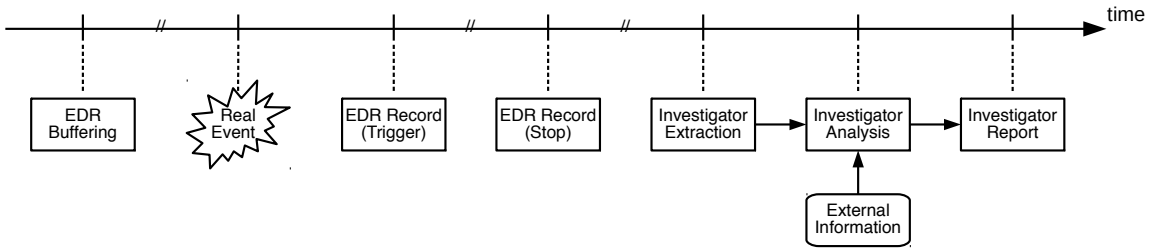


Figure 1: Timeline for the creation of the forensic investigator's report used in court.

The left side of the diagram in Figure 1 shows the data regarding the vehicle operation is continuously being produced. This operational data is usually available on the vehicle CAN bus or it is being produced directly from the processors on board the ECU with the HVEDR logic. The data is continuously being added to a circular buffer that retains some time history of operation. Once a real event happens, the HVEDR recording algorithm is triggered. Depending on the algorithm, an HVEDR may freeze current buffer to memory at the time of the trigger, or it may continue to record data for a specified time and then write the data to memory.

Once the HVEDR has completed the recording of the event, it is held in a memory location until it is overwritten. If the vehicle with event data continues to operate after the event, the existing data may be overwritten, which is another potential attack on the data. The digital data is extracted with an extraction tool.

Once an investigator acquires the data with an extraction tool, they perform an analysis on the data using external information. Since an extraction tool typically gives data from multiple events, the first goal is to attribute the correct digital event data set to the real event of interest. For example, if there were three events from sudden speed changes, then the investigator must attribute the correct event to the incident. Typically there are three pieces of information used to attribute digital event data to a physical incident: 1) real-time clocks, 2) engine hours, and 3) vehicle distance records (odometer readings). Some of these attribution data are usually available at the time of the HVEDR extraction as well as within the metadata for the event record. Visual inspections, data reports from the back office, and physics based reconstructions all make use of external information and the HVEDR data.

After an investigator performs an analysis, he or she may write a report, aggregate all the data used, and archive the information. This data is discoverable when legal proceedings ensue. Not only will the written opinions from the analysis be shared, the original source material will be shared also. This original file format containing the HVEDR record is part of the data used as a basis for fact in adjudicating the case. As such, it needs to be verified as not having been tampered with. However, some common file formats for HVEDR data do not have tamper resistant file structures and they can be easily manipulated without detection [5].

This overview of the life span of digital data from HVEDRs has introduced a few attack vectors by way of discussion. Additional threat models will be discussed, then a taxonomy of threats will be generated as a tree for visualization. It is important to note that the analysis performed herein is for typical HVEDRs and may not directly apply to every recording system on the road. However, the process of performing the threat assessment should prove to be useful for other systems that record HVEDR like data, including passenger car EDRs.

Trust Model for HVEDR Data

Courts trust and rely on HVEDR data. To establish the trust, there are layers and sequences of trust that build on one another. At the foundation of it all is reliable sensor data. This trust is verified through engineering experiments and quality assurance of the operational parts of vehicle systems. The trust in this data can be violated from things like faulty sensors, intermittent electrical problems, and calibration data that does not match the installed hardware (i.e. tire size changes). The sensor inputs also are used to trigger events. Sometimes, sensor data is transmitted to the HVEDR over a vehicle network.

Once an event triggers the recording algorithm, the data is stored in the device memory. The trust model says the data present during extraction was the data recorded. The record should be robust and resistant to alterations.

The data extraction process should be resilient and repeatable to establish trust. The trust model establishes how the data that was extracted can be trusted to be the same data that was stored on the ECU – it was not altered during the extraction process.

Once the data is extracted, it can either be stored or interpreted. Often times, data is interpreted in the same session of the extraction, so only the interpreted data is stored. This, however, requires a level of trust that the interpretation of the stored ECU binary record was correct. Establishing trust regarding the interpretation of the data may require physical testing with external reference measurements of similar make and model vehicles.

Off-board storage is the long-term preservation of the HVEDR record. The trust model says that the data in the long-term preserved storage is representative of the original recordings. Establishing this may require being able verify the original contents of the HVEDR data files.

Event attribution is a method to establish trust in the fact that the HVEDR data represents the real event in question. Attribution data includes serial numbers, mileage, real-time clocks, and diagnostic clocks. These attribution data should be recorded at the time of the event trigger and again at the time of extraction. If an HVEDR records multiple events related to speed change, then the attribution data is used to figure out which one is the most recent or relevant to the event in question. Timely recovery of HVEDR data helps secure and establish trust for the attribution data.

Data verification is the process of comparing the data to the context of the event and ensuring the physical evidence and digital evidence agree with each other. It can also involve cryptographically verifying the original contents of the HVEDR records using signature verification techniques. In the end, this verification process may be needed for the courts to trust the HVEDR data. In building this model of trust, we have not addressed methods to attack these layers of trust. To do this, we invoke the Attack Tree tool [1].

Modeling Non-Malicious Events

The usual attack tree analysis considers sub-events arising from malicious action which contribute to events. However, it is important to also model events that occur accidentally or as a result of non-malicious behavior. In the usual case of malicious events, estimates of attack difficulty (which is a principal factor in modeling likelihood of attack) are included which are not applicable to non-malicious events. For our purposes, a trivial difficulty will be ascribed to non-malicious events. This is for the purposes of 1) ensuring that the mitigations we

[1] https://github.com/BenGardiner/mindmup-as-attack-trees

list focus first on the 'classic' reasons for the forensic process and also 2) so that we don't bog-down the analysis in this paper with additional experimental modeling. There is an opportunity for future work in attack trees to mix events whose probabilities are modeled by attack difficulty and also by empirical likelihood simultaneously.

Attack Tree Analysis for HVEDRs

In this section we will examine the attack trees that represent the model of the threats on the forensic investigation process and HVEDR systems using a graphic formatting and notation known as an attack tree [3]. The subsections which follow will each consider a small subsection of the larger tree each in turn. The complete tree is presented in the Appendix.

The most abstract event to which all others in the attack tree model contribute – the *attacker goal* on which our analysis focuses – is an occurrence of any discrepancy in an EDR record that violates the trust in its fidelity. If this event occurs then the stakeholders will be impacted negatively.

Violating trust in the data can be intentional or accidental. At first, one may consider these two descriptions as mutually exclusive, but a malicious act may be masked as accidental, which makes distinguishing between the two challenging. Instead, we approach the taxonomy of violating trust through different events, as opposed to intents. This attack tree model can be considered to be the *dual* of the trust model. This means the tree includes events and mitigations that may affect the hardware, software, or forensic investigation procedure. The tree includes an exhaustive list of mitigations; however, we will not strive to discuss each and every one, nor will we list an example for each mitigation in what follows. The section *Mitigations* will enumerate the exhaustive list and also discuss some selected mitigations. Also note, that the events in the attack tree model could occur before or after the *Real Event* (in reference to figure 1); the attack tree model focuses on the relationship of faults or attack events contributing to the *goal* and not on the chronology of said events.

Attacker's Goal: Compromised HVEDR Data

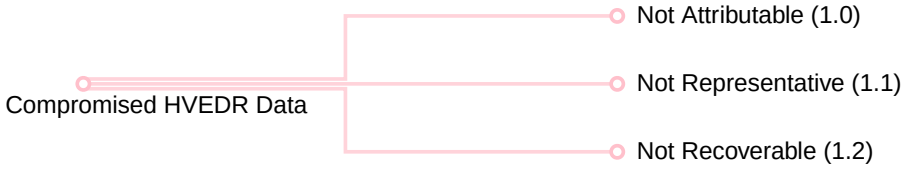


Figure 2: *Compromised HVEDR Data Subtree*

The root of the attack tree as mentioned above and as shown in Figure 2, is the discarding or omitting of the digital evidence due to lack of trust. If an attack is successful, then the digital data that was created by the HVEDR is not useful. Trust is built in layers around the data and requires procedural and mathematical methods to establish trust. There are three general cases that could render an HVEDR untrusted: the HVEDR data is not recoverable, not representative, or not attributable.

The remaining subsections will traverse the tree and provide descriptions, examples where appropriate, and mitigations.

Attack Tree Notation

In what follows, we will present several diagrammatic renderings of subsections of the complete attack tree model (see the appendix); these are referred to as *subtrees*. In the diagrams we will use a visual notation for

several purposes. First, to distinguish between nodes of the attack tree model which are concrete: having no further expansion in the attack tree model and hence also being ascribed with estimates of attack difficulty; or those which are abstract: defined by children nodes in conjunctive and disjunctive relations, whose difficulty estimates are derived from compound relationships of their concrete children.

Abstract nodes are represented by a hollow-fill (all four nodes in figure 2 are abstract); whereas concrete nodes are represented by a solid fill. We include both attacks/faults and also mitigations in the attack tree model; the mitigations are rendered as square nodes whose solid fill is green and the attacks/faults as circular nodes whose solid fill is black. In both cases, the hollow-fill is white.

The nodes are each assigned a numeric reference in two-dimensional coordinates (to avoid the much longer numeric identifiers that result from nested numbering). These coordinates are assigned to each node as a function of its location in the complete attack tree model; nodes keep these coordinates when they are rendered in subtrees. Abstract nodes rendered in subtrees can appear as either a definition or a reference nodes. Reference nodes include their coordinate as a parenthetical '(X.Y)' at the tail of the node title (as are nodes 1.0, 1.1 and 1.2 in figure 2); whereas definitions of abstract nodes include their coordinate as a leading 'X.Y' in the title as do all concrete nodes (as does node 1.0 in figure 3 below). These reference nodes facilitate the re-use of entire subsections of the complete attack tree, yielding a model of multiple impacts of both mitigations and attacks/faults alike. The reference nodes are also the links between the separate renderings of subtrees. e.g. node 1.0 in figure 2 above is a reference to node 1.0 in figure 3 below.

Subtree 1.0 Not Attributable

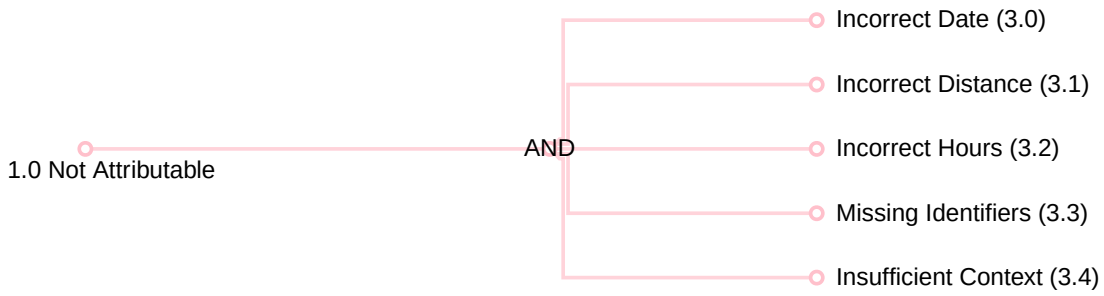


Figure 3: *Not Attributable Subtree*

Attributing a digital record to a real event requires at least a record indicating a unique source (3.3), like a serial number and a date (3.0), distance (3.1), or time record (3.2) to corroborate the event. Finally, the context of the event is important to ensure attribution (3.4). We are modeling insufficient context of the event as equivalent to an obfuscated context and a mitigation is noted. For details see the subtree of *3.4 Insufficient Context* in the appendix.

Some ECUs contain real-time clocks (RTC) that keep track of the calendar date and 24 hour time. These systems require a battery backup, usually in the form of a coin cell lithium battery. Time is tracked as the number of seconds from an epoch, rarely keeping accurate time; frequently subject to drift. It is critical to capture the clock drift at the time of the forensic investigation, by comparing the ECU time to a known reference clock, like network time protocol, or GPS. The HVEDR time stamp can be subtracted from the reference time stamp to create a delta at the time of the investigation. This time delta is then added back to the event trigger time to place the HVEDR record into the actual time. This adjusted time is often compared to first-responder records to establish attribution. To ensure the adjusted time is correct, both the ECU time and the external reference time must be genuine. In future applications of V2V all heavy vehicles will have GPS-based time sources which can be readily used to measure and correct drift.

Distance is the odometer reading; however, some ECUs keep track of distance using different memory elements. One could track the high resolution vehicle distance of 5 meters per bit (J1939 SPN 917) [6], whereas another record keeps track of distance in 0.125 km/bit increments (J1939 SPN 245).

The hours of ECU runtime is also used to tie recorded events to a crash. If the amount of time an ECU runs after an event is attributable through a forensic process, then the event can be attributed. For example, if a ECU runtime is a few seconds ahead of the event, then its is event was likely from the crash that just took place. Only one of these are attribution points are needed to tie the event to the crash; therefore, all of these (3.1-3.3) must be eliminated to be a successful attack.

Identifiers include engine serial number and vehicle identifier. These are important to tie the record to the vehicle. Often a photograph of the crash scene and a VIN inspection provides the necessary evidence. It is important to note that some ECUs allow the end user to customize the VIN, so the digital record may not match the chassis VIN. Engine replacements and glider kits also exist in heavy vehicles, which makes documenting the engine in the truck important.

ECU identifiers are typically not changeable by the end user, therefore, memory access to the HVEDR would be necessary to alter these data fields. Other attacks could parallel those where data is altered after being recorded – which is also the case for this and the other attribution data (this is modeled as including subtrees 2.1 and 2.3 in their respective attack trees). The most common issue is failing to correlate the engine serial number with the vehicle in question. For example, a DDEC Reports does not have a VIN in the report, instead, it reports the engine serial number. Therefore, the forensic investigator must attest to the connection between the engine and the chassis.

The context of the crash is important for proper attribution. For example, if there are multiple heavy vehicles involved in a crash, like an interstate pile-up, where some HVDERs are the same make and model with the same real-time stamp, correctly identifying the digital record to the correct vehicle is important. Making claims using only digital records is an attack on the context of the record.

Subtree 1.1 Not Representative

Figure 4: *Not Representative Subtree*

HVEDR data that is not representative of an event can be misleading and violate trust. The following examples demonstrate how these attacks on the digital data can manifest.

Incorrect recordings deal with the notion that an ECU can incorrectly translate sensor data into engineering units. Sometimes this can be from a configuration change, like gearing or tire size, that would skew the data based on an incorrect calibration. A more deliberate alteration would be to use a device like a GPS jamming device, or a vehicle speed sensor adjuster to change the data being recorded by the ECU. Most, if not all, fielded HVEDRs systems faithfully report the information it receives without question. Thus, sensor inputs are considered trustworthy by the EDR. Tampering with or compromising sensor inputs will affect data before the event, and may be apparent in other operational data. For example, GPS speeds[2] could deviate from vehicle

[2]GPS speed is used here an example reference speed; however, GPS speeds can be spoofed by attackers and therefore additional

speed if the VSS signal is compromised. Wear and wiring issues may also compromise sensor input. This will be examined in more detail in *Subtree 2.0 Data Recorded Incorrectly.*

Sometimes the data recorded in the ECU and the translation of the binary data to engineering units is not correct. The most well known instances of this attack is misrepresenting the HVEDR data sampling rates in some Caterpillar Snapshot Data [7] and some Cummins Sudden Deceleration Data [8]. In these scenarios, the mitigation is to perform externally referenced testing on similar make and model ECUs to properly characterizing these behaviors. Some people may refer to these as anomalies, however, an anomaly is often not repeatable, so a better term may be more appropriate, like a bug in the decoding software. Once public, the effects of these bugs can be modeled and corrected in records.

A hexadecimal translation tool, which is being used as forensic software, has a database to map the binary network data to engineering units. Altering this database is an attack on the data as not being representative of the actual event.

Changing the data during transmission from the ECU to the hex translation tool usually involves transmission of the data on a vehicle network to a vehicle diagnostic adapter (VDA) where the VDA will convert the message(s) into serialized data (USB or Ethernet). This PC friendly serial data is received by a device driver and served to a tool using the RP1210 API and an interprocess communication. This line of data transmission and the multiple network interfaces required to retrieve the data provide attack surfaces where the data can be altered.

In older J1708 networks, the data transmission is protected using an eight-bit checksum at the end of each message. This is insufficient to protect against inadvertent bit flipping caused by external noise induced in the signals. CAN, however, uses a 15-bit cyclic redundancy check to virtually guarantee the correct transmission of data over a network.

Once data is retrieved and stored, a file will likely exist on the forensic investigator's computer. If the ECU is put back into service and/or the data is overwritten, then the aforementioned file becomes the original source of evidence, which is subject to alteration. In some cases, an alteration is necessary. For example, a Detroit Diesel XTR file for DDEC Reports may have issues being interpreted if the date information is incorrect due to a low battery for the ECU's real time clock. Changing the bytes to affect the date in the file has a positive effect of enabling further interpretation of Hard Brake and Last Stop records by the DDEC Reports software. However, there is no mathematical method by which to verify the original authenticity of the file. This means other records could also be altered, like the recorded speed.

Many systems do not specify or store any original binary file and use PDF, HTML, or XML files to store these data. Some may include a checksum in the file to check tampering, but most do not. Therefore, it is strongly recommended that forensic investigators digitally sign original forensic files and obtain permission to perform any file manipulations.

Subtree 1.2 Not Recoverable

The idea of data not being available to recover means that the HVEDR should have created interesting data at one time, but the forensic recovery process was insufficient to obtain that data.

Physical destruction of an HVEDR can make the data unrecoverable. Fire is the most damaging and destructive phenomenon. The proximity of the HVEDR to fuel in some ECUs creates and environment where the hottest part of the fire will be near the HVEDR. In fact, some engine compartment ECUs use fuel to cool the electronics. Mounting the HVEDR in the cab can both reduce cost and provide better physical protections.

Other physical destruction of the HVEDR printed circuit board and connectors may render the HVEDR data irrecoverable through traditional means. However, some have show the ability to recover data from the memory contents of the chips themselves [9]. Further modeling of this procedure is beyond the scope of this paper.

sources of corroborating speed should be sought. For example, ABS modules use 4 to 6 wheel end sensors and carefully diagnose their signals.

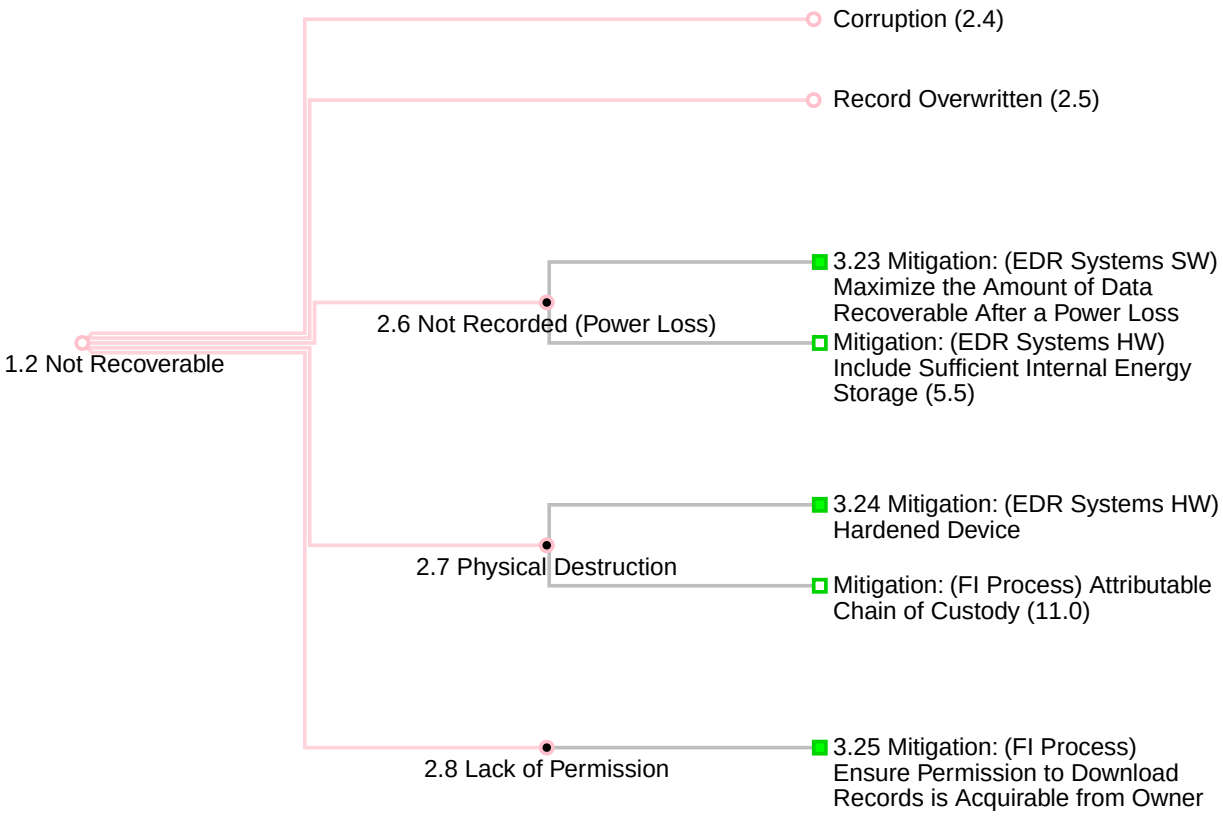


Figure 5: *Not Recoverable Subtree*

Physical destruction of the vehicle can also lead to system power loss. Experiments with Detroit Diesel and Mercedes HVEDRs show the loss of power after an event trigger prevented writing the HVEDR record to non-volatile memory [10]. Arguably, the data was never present in the HVDER, but a strategy of waiting for all post-trigger data to be available makes the system less robust when dealing with events that lead to compromised electrical systems. To confound the issue, EDR circular buffers cannot be assigned to flash memory because it has a limited number of writes. Building an energy reserve, like a large capacitor, into the HVEDR would enable longer duration recording resulting in more robust data captures. Furthermore, the tool needs to be able to extract and interpret partial records that failed to complete due to the power loss.

HVEDRs in Caterpillar engine control modules (ECMs) take the approach of freezing data at the trigger and then appending post event data to the record [11], whereas Detroit Diesel ECMs continues to record for a specified time before the data is written to non-volatile memory [10]. This distinction becomes important to the integrity of the data when the events are significant enough to interrupt power to the ECU that is performing the HVEDR functions. For this discussion, power interruptions are considered cybersecurity attacks on HVEDRs, even if they are not intended.

Many HVEDRs will trigger an event from a Diagnostic Trouble Code (DTC). These DTC codes driven events may have time history data of vehicle operation before and after events. In some cases, these DTCs can be set as a result of the crash. For example, if the physical evidence shows the engine oil pan is compromised, then a low engine oil pressure DTC may be present along with it corresponding record.

The biggest threat to these DTC based HVEDR records is having the record overwritten with a new one. This threat is only increased by the additional pressure to record data on DTC events as will be required by forthcoming HD OBD standards. This is important enough to show a separate branch on the tree under SUbtree 2.5. Having records overwritten is particularly important when the electrical system on the vehicle is compromised and the device with the HVEDR functionality is removed. If that module is the engine control module, then it will expect to see many sensors and actuators connected. If those are absent when the HVEDR is turned back on, then new fault codes will be written and potentially overwrite the ones that were generated from the crash. A forensic investigator will want to know what, if any, DTCs were present at or before the crash event.

Mitigating overwriting fault codes can be done with logic that doesn't overwrite code freeze frame data with new data. If examining legacy systems, then transplanting the HVEDR onto a surrogate vehicle can create a fault free environment. Finally, a sensor simulation system can be used to bring ECUs back online while simulating all of the sensors and signals it is expecting, thus not setting any new fault codes.

Data that is not recoverable is data that should be present, but there is no easy way to obtain the information. A forensic investigator must be a good troubleshooter at times to be able to overcome power and connectivity challenges that would take an HVEDR offline after a crash. For this discussion, we will assume that electrical systems are in good repair.

Some data is not recoverable due to the lack of proper permission. In US courts, the process of collecting evidence is scrutinized under the fourth amendment of the constitution. As such, forensic investigators should gather consent from the proper authority or obtain a search warrant from the courts. This aspect of making data not recoverable does not have a technical solution.

Subtree 2.0 Data Recorded Incorrectly

Incorrectly recorded data happens when the stored binary data does not match the physical phenomena it is reporting. Determining if such an event has occurred requires testing of the physical phenomena with external reference instrumentation.

An incorrect recording can be a result of one of the following: 3.5 Corrupt Sensor Data, 3.6 Missing Network Data, or 3.7 Incorrect EDR Calibration as shown in the Appendix. There are two sources for data: the vehicle communications network (i.e. J1939) and sensors. For example, an engine control module with HVEDR

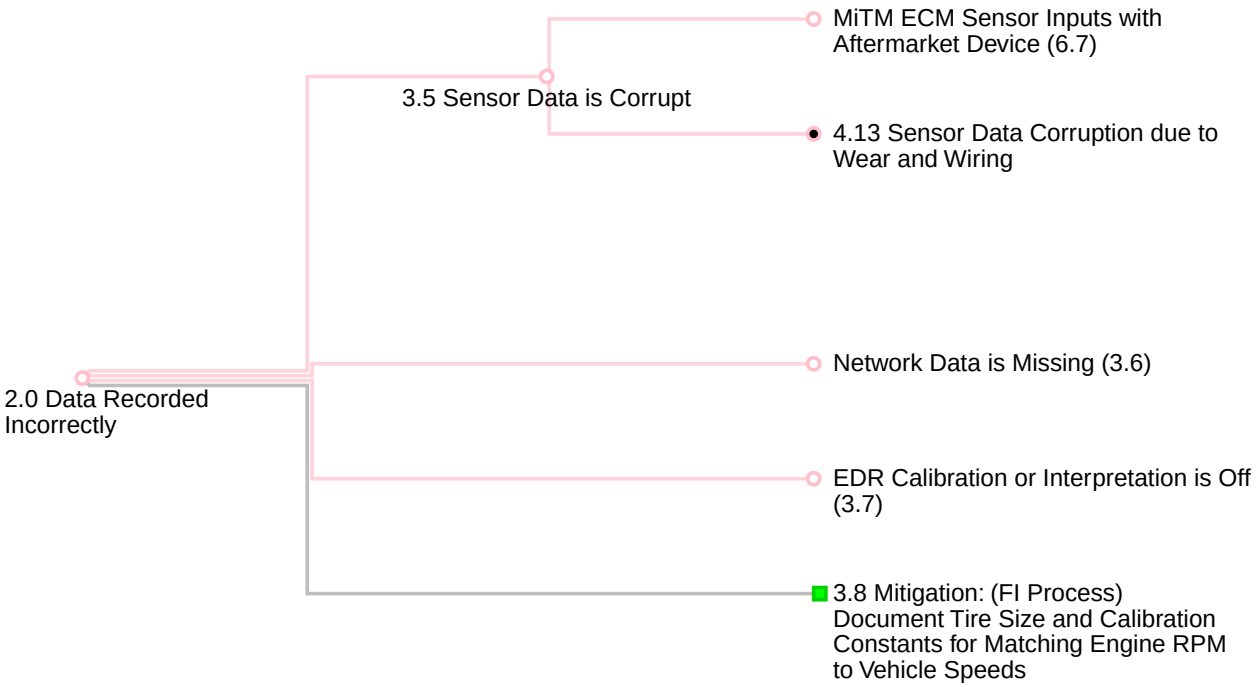


Figure 6: *Data Recorded Incorrectly Subtree*

functionality can be programmed to accept vehicle speed data from a vehicle speed sensor (VSS), which is uses a sensor monitoring a tone ring on the tailshaft to determine vehicle speed, or it can be programmed to determine speed from a J1939 source, like the transmission controller.

As vehicles age, wiring may become compromised and lead to erratic sensor inputs. Often these sensors are monitored by the ECU. If a sensor value becomes erratic, the ECU will often generate a diagnostic trouble code to reflect that status. This possible fault is shown in node 4.13 of the attack tree.

Attack node 6.7 says a MiTM ECM Sensor Inputs with Aftermarket Device. This is a Man in The Middle (MiTM) attack that potentially changes the sensor data before it reaches the ECU. This type of attack must be conducted before an incident and requires physical access to the vehicle. Examples include speed signal devices that affect the pulses coming from the tailshaft speed sensor. Mitigations to these include vehicle inspections and creating consistency checks from other sources. For example, the front axle speed from the electronic brake controller can be compared to the wheel-based vehicle speed from the tailshaft vehicle speed sensor and/or the rear axle speed (since ABS devices use wheel end VR or sometimes Hall effect inputs).

During the forensic investigation, the vehicle configuration and ECU calibration parameters should be documented. This process mitigation is shown in the 2.0 subtree as 3.8. The investigator can use this information to determine consistency within the HVEDR report. For example, engine speed may be correlated to the vehicle speed through the gearing and tire size.

Subtree 2.1 Data Changed During Transmission

An attack on the HVEDR data can take place during the extraction process when the data from the ECU is being download over the vehicle network. There are three distinct media carrying the data during a typical forensic process. First, the data is extracted from memory and placed into the vehicle communications network (J1939 or J1708) by the ECU. Second, the data passes through the diagnostics port to a vehicle diagnostics adapter (VDA). The VDA translates the vehicle network data into a PC friendly format like USB. The forensic application software reads the VDA data through a series of drivers and interprocess communication channels on the PC.

Subtrees 3.9 and 3.10 show a potential attack by affecting the vehicle communications network. Abusing a transmission protocol on the vehicle by means of a rogue node can be challenging for CAN, but much easier for J1708 due to the lack of error detection in J1708. A way to manipulate vehicle network traffic is to install an inspect and forward hardware device at the diagnostic port[3]. However, this additional hardware should be easily detected by a forensic investigator (unless the FI is the one who installs the device).

Then, considering the VDA; since the VDA itself is an inspect and forward device that translates network traffic, it is possible to exploit the firmware on the VDA to subvert the forensic data. This is also the case for devices which replay captured binary data, since the software parsing the replay would also be subject to exploitation by the replayed traffic. Conducting this attack will most likely target the VDA firmware update process and change the normal operation of the microprocessors on the VDA. This attack is shown as subtree 3.11. The practical difficulty of such an attack would depend on implementation details of the VDA, we broadly modeled it (for the purposes of capturing mitigations) as understanding the firmware enough to be able to make a change to it and then be able to write this firmware to the device and have the device 'accept' it. This does require expertise, specialized skills and access to the implementation details of the target.

After the data is translated by the VDA for the computer, it needs to be transfered to the hex translation tool though the VDA device drivers. This process provides an attack surface to alter the contents as they pass through the driver software, as shown in 3.12. This was introduced and discussed in the section above.

A mitigation (3.13) for all these potential attacks in 2.1 is to have verifiable integrity for the data that is traceable to the HVEDR. An example would be for the HVEDR to digitally sign the data record.

[3]this would be an illicit inspect and forward device, as opposed to the forthcoming store and forward OEM modules

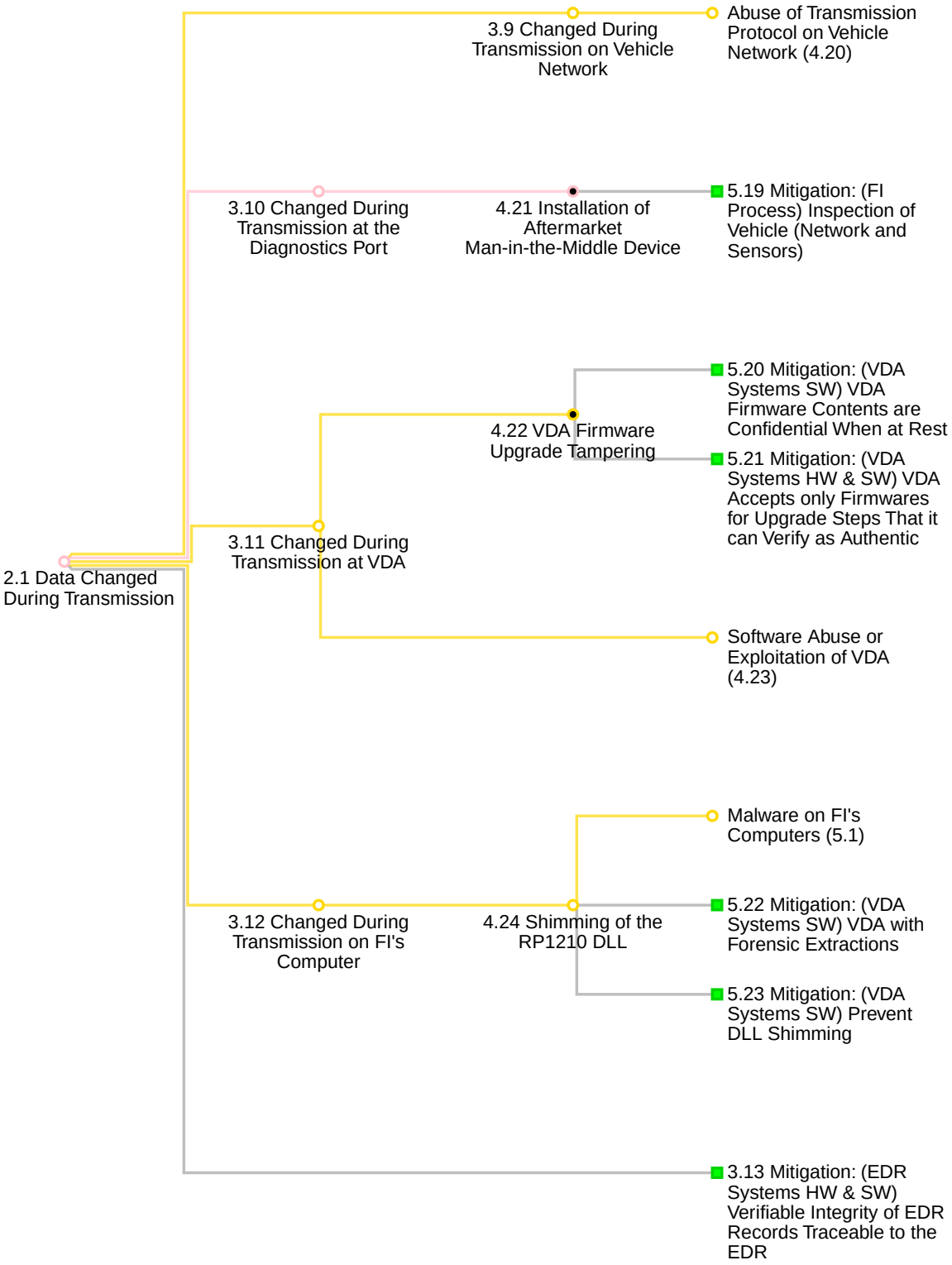


Figure 7: *Data Changed During Transmission Subtree*

Subtree 2.2 Hex Translation Incorrect

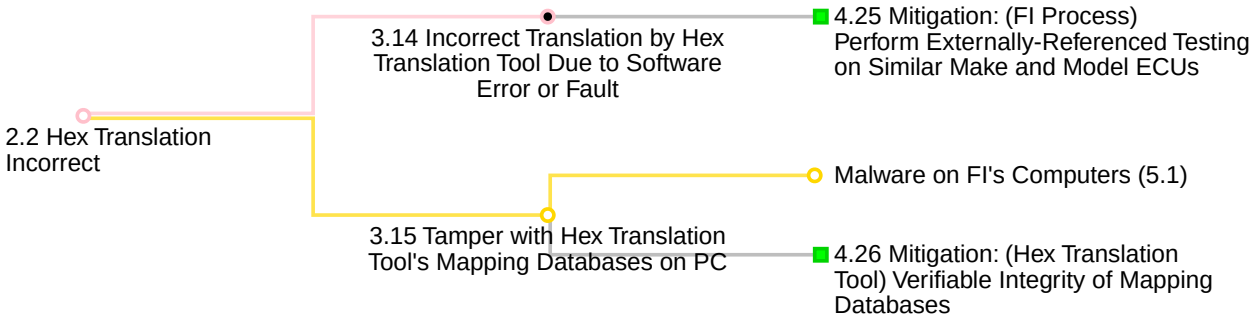


Figure 8: *Hex Translation Incorrect Subtree*

As mentioned, the raw data from the HVEDR may be interpreted incorrectly due to a software error or fault. Mitigating this issue requires externally referenced testing to give confidence the tool is performing the translations correctly.

Another attack on the translation is to affect the database that maps the raw binary values to engineering values. This can be done by compromising the hex translation tool via malware and mitigated by installing verification checks on the database that contains these maps.

Subtree 2.3 Data Storage Archive Altered

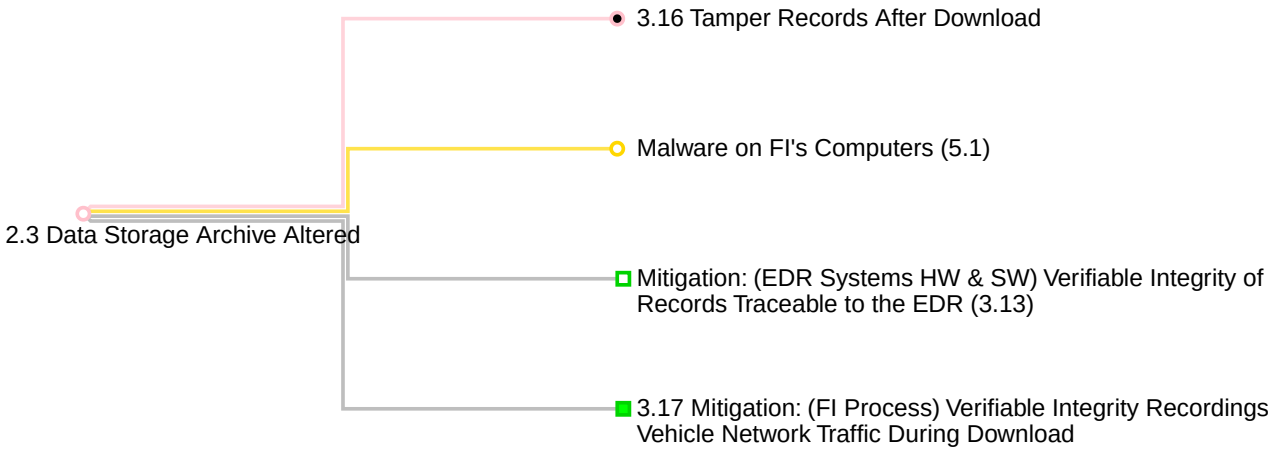


Figure 9: *Data Storage Archive Altered Subtree*

In this attack tree model we treat tampering of records differently than alteration. The former is considered to be malicious and, obviously, should be prevented. The latter (alteration) is acceptable when performed by

a forensic investigator for lawful purposes. For example analysis of records in the case of an EDR where the RTC battery has failed can require alteration of downloaded records to coerce the data into a format that is acceptable to the hex translation tool.

Altering already-downloaded data can be done by a proficient person in less than a day. There is some special knowledge required to understand how/what to alter but the task can be accomplished with standard equipment. Often the files used to store the data are easily manipulated with common tools of the trade.

The subject of deploying targeted malware to be deployed on an FI computer, as shown in subtree 5.1, is broad and complex. There are multiple options available to an attacker that seeks to deploy malware for the purposes of tampering with the EDR process. First, the attacker could develop malware specifically for this purpose, in which case this attack difficulty would look very much like *Discover, Develop Abuse Command or Exploit Payload to EDR (9.2)*; it is more likely; however, that the attacker would purchase toolkits for creating malware for PCs. In this later scenario less expertise and time is is required.

Note that either of the mitigations:

- *(EDR Systems HW & SW) Verifiable Integrity of EDR Records Traceable to the EDR (3.13)*
- *(FI Process) Verifiable Integrity Recordings of all Vehicle Network Traffic During Download (3.17)*

would be sufficient to reduce the likelihood of this event. But the first one will take a very long time to be implemented. In the meantime the situation can be improved with the second.

Subtree 2.4 Corruption

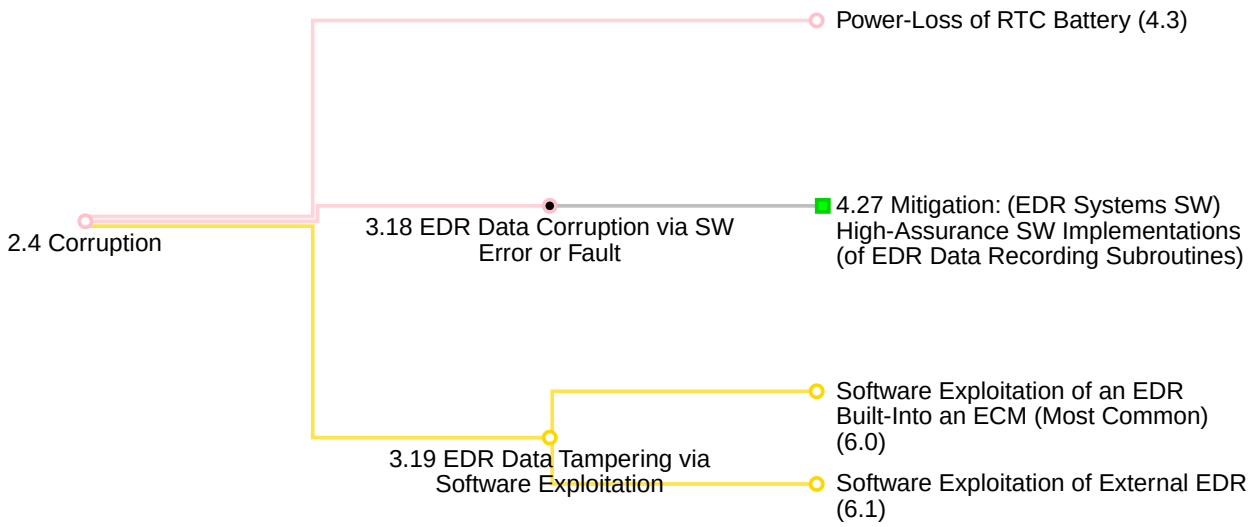


Figure 10: *Corruption Subtree*

Returning to the first-level event of Not Recoverable data, a possible set of contributing events could be corruption of the EDR records. It is possible that a power-loss, electrostatic discharge, or other external influence could cause corruption. Proper care and handling of electronics can reduce this risk. It is also possible

that corruption could occur as a result of wear-out of flash storage; proper handling will not resolve this issue but designing for operational lifetimes can mitigate the risk.

There is an interesting case of time-stamp corruption for older DDEC ECUs where the realtime clock has unrealistic values. The DDEC Reports downloading software has a difficult time accommodating these corrupt records. The corruption occurs from a lack of power on the realtime clock.

Both software faults or software abuse/exploitation regarding the EDR could contribute to corruption of data. However, the faults/error case is modeled as a non-malicious event, and a detailed as in the other cases covered in the tree.

The abuse/exploitation case is modeled both for EDRs built-into an ECM and for those that are external as subtrees 6.0 and 6.1, respectively. The practical details of software exploitation or abuse can vary significantly depending on implementation details, we generalized and made assumptions to be able to place classes of mitigations in the tree. These mitigations will be discussed in the Mitigation Section below.

Deeper Into The Attack Tree

We have covered the high level attacks on HVEDR data that can subvert trust in the data. There are additional attacks on the forensic data that are detailed in the graphical representation of the Attack tree in the Appendix. Including the details of each node would exceed the recommended length for these conference proceedings, so it is left to the reader to traverse the tree and develop an appropriate model for specific systems. Omitting details at this point in the tree is appropriate because accurate models of the attacks require implementation specific understanding of the systems to render meaningful discussions. This is particularly true of the lower branches of the tree where aspects such as software exploitation and abuse are introduced. Readers who are seeking to apply the findings of this analysis should consider only those subsections of the complete tree that apply to the specifics of their target.

Mitigations

Mitigations against the attacks shown in the attack tree can be broken down into the following broad categories: 1) EDR Systems Mitigations, 2) Forensic Process Mitigations, and 3) Connected Systems Mitigations . The EDR systems include both hardware (HW) and software (SW) based mitigations. In many cases the mitigations listed below are already found in some deployed EDR systems. Also, the mitigations are not listed in any particular order to suggest relative merit. The analysis of the complete attack tree model yielded a considerable number of mitigations; it is beyond the scope of this paper to suggest means for each mitigation identified. Their identification founds the trust model.

Enumerated EDR Systems Mitigations

Event Data Recorder (EDR) systems comprise the union of all of the vehicle components contributing data to the record. In many cases a combination of multiple software components running in the engine control module (ECM). The mitigations in table 1 for these systems would improve the soundness of the forensic data records.

SW	HW	Verifiable Integrity of EDR Records Traceable to the EDR	(3.13)
SW		Maximize the Amount of Data Recoverable After a Power Loss	(3.23)
	HW	Hardened Device	(3.24)
SW		High-Assurance SW Implementations (of EDR Data Recording Subroutines)	(4.27)
SW		Log all Trigger Occurrence Timestamps in Tamper-Evident Storage	(4.29)
SW		Include Requirements to Prevent Distance Tampering via Abuse or Exploitation of HU[4] SW	(5.10)
SW		Include Requirements to Prevent Hours Tampering via Abuse or Exploitation of ECM SW	(5.11)
SW		Include Requirements to Prevent Identifiers Tampering via Abuse or Exploitation of EDR SW	(5.12)
SW		Include Requirements to Prevent Network Data Denial via Abuse or Exploitation of EDR SW	(5.14)
SW		Access Controls on Calibrations Modifications	(5.16)
SW		Include Requirements to Prevent Calibration Modifications via Abuse or Exploitation of EDR SW	(5.17)
SW		Include Requirements to Prevent Tampering of EDR Data During Download	(5.18)
SW		Use UTC for all Timestamps	(5.4)
	HW	Include Sufficient Internal Energy Storage	(5.5)
SW	HW	Use Other Available Time Reference Standards	(5.6)
SW		Include Security Requirements to Prevent EDR Clock Subversion via Abuse or Exploitation of EDR SW	(6.2)
SW		Make Vehicle Network Protocols More Resilient to Tampering	(7.3)
SW		Make HU Software More Resilient	(8.0)
SW		Make ECM Software More Resilient	(10.0)
SW		Make EDR Software More Resilient	(10.4)

Table 1: EDR Systems Mitigations

Discussion on EDR Systems Mitigations

Mitigation *Hardened Device (3.24)* refers to the mechanical robustness of the device to survive crashes. While building an HVEDR system with similar attributes to a flight data recorder in aviation is possible, the expense associated with such a mitigation is not practical. However, moving the HVEDR from the engine compartment can go a long way to protecting the data in a crash. Locating the physical recording device near the center of the vehicle minimizes the exposure of the HVEDR to crushing forces.

The mitigation *Make Vehicle Network Protocols More Resilient to Tampering (7.3)* involves implementing mitigations to make injecting, corrupting or otherwise tampering with normal vehicle network traffic more difficult. For example, adding Message Authentication Code (MAC) checks to all messages on the vehicle network. This is, of course, a mitigation which is of limited feasibility, particularly when it comes to legacy vehicles. The CAN standard does not provide enough frame bytes to allow for MACs on all frames. Deploying MAC on the vehicle networks would require, at least, a move to a network standard with larger frames, such as CAN-FD and this is ignoring the potentially untenable impacts of transmission and processing latency incurred by a MAC scheme. This is obviously a non-starter for existing vehicles and poses many integration issues for designers of new vehicles.

Furthermore, please note that in this particular case a necessary second-order mitigation [5] would be to prevent

[5]Second-order mitigations are those mitigations which are deemed necessary based on an analysis of the attacks to defeat a given (first-order) mitigation. The complete attack tree model of this analysis *did not* model second or higher order mitigations; in this case the second-order mitigation is known to be necessary from the experience of the authors. Similar justification applies

abuse of the ability to send MAC validated messages from modules via software exploitation. This second-order mitigation would be a natural realization of *(Connected Vehicle Systems SW) Includes Security Requirements to Prevent Sending Vehicle Network Traffic via Abuse or Exploitation of Connected Module SW (11.4)*. Ultimately, even if the second-order mitigations are realized, the addition of MAC to all messages would not prevent hardware tampering such as interposing ECM sensor inputs to falsify the pulse rate it captures; the ECM would, naturally, trust that input and send a MAC-validated message encoding the false value. Note that the shortcoming of this individual mitigation (and the shortcomings of other single mitigations) are modeled in the complete attack tree, where the disjunctive relationship of the parents of the mitigations indicates that even if the mitigation were perfect, attacks/faults are still possible to contribute to the goal.

Mitigations to make the head unit (HU), engine control module (ECM), and event data recorder (EDR) software more resilient are modeled as nodes 8.0, 10.0, 10.4, respectively. The development of resilient software systems is a necessary condition for secure systems. For this mitigation we are grouping a large number of varied software development strategies to yield resilient software. The subject of what to secure and how to secure it in application security is wide and complex. The authors recommend consulting experts in the selected platform technologies for the connected systems applications, following best practices during development and also of designing for security before development as well as testing the security after development. These recommendations can be found in industry standards such as the SAE J3061 guidebook [12] where cybersecurity best practices can be found mapped to the phases of the v-model for product development.

Mitigations 5.10, 5.11, 5.12, 5.14, 5.17, 5.18, and 6.2 are all to prevent privilege escalation via abuse or exploitation of software. Protections against privilege escalation should be included independent of implementations of resilient software – a so-called 'layered defense'. The attack tree has highlighted several useful privileges that should be protected against escalation by attackers in the design of EDR systems, listed below. The utility of modeling these privileges separately when they may, in fact, be the same privilege for some implementations of ECMs is that the individual impact of segmenting, sandboxing or otherwise restricting the use of the privilege can be estimated from the attack tree model.

- Distance Tampering
- Hours Tampering
- Identifiers Tampering
- Vehicle Network Data Denial
- Calibration Modifications
- Tampering of EDR Data During Download
- EDR Clock Subversion

Enabling the EDR to produce records along with an authenticity certificate or token which would allow any interested party to verify the integrity of those records mitigates against both corruption of data during download and altering the record after download. This mitigation is modeled as *Verifiable Integrity of EDR Records Traceable to the EDR (3.13)*. An example implementation that would satisfy this requirement would be to provide digital signatures (in the pubic-key cryptography sense) of the records where the private key used would be protected within EDR. It would also be required in this case to provide some infrastructure available to all interested parties with records of the public keys of each of the EDR types and to also ensure that this registry of public keys could not be easily tampered. Also, a necessary second-order mitigation in this case would be to prevent the use of this private key *after* software exploitation of the EDR.

Mitigation 3.13 provides a necessary technical solution since forensic investigators would need to continue to be able to alter the records after downloads for lawful purposes of forensic analysis but is not sufficient by itself. In which case, a security requirement that the records, after download, be of verifiable integrity traceable to the FI.

Note that this second-order mitigation is very similar to *(FI Process) Verifiable Integrity Recordings of all Vehicle Network Traffic During Download (3.17)*.

to the second-order mitigations in the discussion that follows.

Enumerated Forensic Investigation Process Mitigations

The process followed by forensic investigators has an impact on the soundness of EDR records. The attack tree not only considers events arising from malicious action but also events that can occur accidentally. Many of the process mitigations exist to reduce the likelihood of any of these events negatively impacting the soundness of the records extracted. Furthermore, many of the process mitigations may already part of the standard forensic investigator processes. These mitigations are listed in table 2 below.

Verifiable Integrity Recordings of all Vehicle Network Traffic During Download	(3.17)
Timeliness of Extraction (e.g. No Driving)	(3.21)
Use a Surrogate Truck or a Fault-Free Simulator to Prevent New Triggers	(3.22)
Ensure Permission to Download Records is Acquirable from Owner	(3.25)
Document Tire Size and Calibration Constants for Matching Engine RPM to Vehicle Speeds	(3.8)
Photograph the physical HVEDR in-situ With Identifiers	(4.11)
Perform Externally-Referenced Testing on Similar Make and Model ECUs	(4.25)
Resolve Odometer With Network Saves	(4.6)
Compare the Various Different Tags That Represent Engine Time	(4.9)
Use Other Forensic Evidence to Properly Correlate EDR Records to Correct Vehicle and Event	(5.13)
Look for Signs or Repair and Document Tire Size	(5.15)
Inspection of Vehicle (Network and Sensors)	(5.19)
Use Additional Available Time Reference Standards (e.g. GPS)	(5.2)
Ongoing Training / Accreditation of FI for Assurance of Process Quality	(5.3)
Manually Adjust Timestamps	(5.7)
Compensate for Clock Drift in Timestamps	(5.8)
Safe Operating Procedures with Analysis Computers (e.g. up-to-date OS etc.)	(6.3)
FI System & Software with Verifiable Integrity of Programs, incl. DLLs	(6.4)
Attributable Chain of Custody	(11.0)
Look for Devices Attached to the Vehicle Network	(11.1)
Remove or Disconnect the Associated Wireless EDR Interfaces to the Vehicle	(11.5)

Table 2: FI Process Mitigations

Discussion on Forensic Investigator Process Mitigations

The analysis of the attack tree has highlighted *Verifiable Integrity Recordings of all Vehicle Network Traffic During Download (3.17)* as a mitigation against both the corruption of data during download and of altering the record after download. If an investigator recorded the vehicle network traffic as it was being produced during a forensic investigation, then there is a primary source record available for archival purposes. With modest development of tools 3.17 could be implemented in the HVEDR extraction process today and in so doing increase the soundness of the records recovered in the field with much more rapidity than the alternatives.

The nature of vulnerabilities in the PC space and the rapidity with which malware is developed and deployed to take advantage of new vulnerabilities necessitates keeping operating systems up-to-date. This suggests the

FI should follow *Safe Operating Procedures with Analysis Computers (6.3).* This step, and other best practices for securing personal computers is recommended. Detailing best practices for maintaining a PC is beyond the scope of this paper. The procedures will often be dictated by information security professionals working with the FI.

Whereas *Safe Operating Procedures with Analysis Computers (e.g. up-to-date OS etc.) (6.3)* above mitigates unintentional attacks on the EDR records (such as ransomware); the *FI System & Software with Verifiable Integrity of Programs, including DLLs (6.4)* mitigates targeted attacks against the extraction software. Attacks of this nature would require differ capabilities, since the target for subversion will likely be a Windows executable.

The fleet management utility of EDR hardstop events is clear, where fleet safety directors are using these events as part of their driver assessment and counseling programs; for this reason, it could be unfeasible to implement any design that truly satisfies the requiement captured by *remove of Disconnect the Associated Wireless EDR Interfaces to the Vehcile.* These wireless interfaces will be needed for, at least, the driver's sake. Furthermore, quarantining the interfaces within the vehicle network may not be possible either. In the case of wireless interfaces integrated into the EDR, this would be equivalent to quarantining the EDR itself – which is only possible in the case of an external EDR, where it could be behind a read-only protected conenction to the vehicle bus. In the other possible case of an external wireless interface, it will require bi-directional access to the EDR itself – precluing any simple segmentation.

Enumerated Data Extraction Tool Mitigations

The data extraction tools and associated software components for converting the data encoded for the vehicle systems into human readable forms, referred to as *Hex translation tools* or the *VDA* in the attack tree. Because these systems must be trusted to perform the correct translations to realize soundness of records, there were several mitigations highlighted in the analysis of the attack trees to keep these systems sacrosanct, as highlighted in table 3 below.

SW		(Hex Translation Tool) Verifiable Integrity of Mapping Databases	(4.26)
SW		VDA Firmware Contents are Confidential When at Rest	(5.20)
SW	HW	VDA Accepts only Firmwares for Upgrade Steps That it can Verify as Authentic	(5.21)
SW		VDA with Forensic Extractions	(5.22)
SW		Prevent DLL Shimming	(5.23)
SW		Make VDA Firmware More Resilient	(7.4)

Table 3: VDA Systems Mitigations

Discussion on Data Extraction Tool Mitigations

Examples of implementations which would satisfy the *Verifiable Integrity of Mapping Databases (4.26)* include incorporating software protection mechanisms into the hex translation tool to increase the difficulty of tampering with the databases. Another possibility would be to include the hex translation tool functionality in an embedded device where the embedded systems software could provide integrity verification for the databases.

For the cases where the VDA system is based on the standard RP1210 API, there will be a DLL with that API. The *Prevent DLL Shimming (5.23)* would make it harder to tamper with data during transmission. Examples of implementations that would satisfy this requirement include software protection mechanisms to enforce only known DLLs in the process space of the VDA software.

Connected Systems Mitigations

The introduction of connected components onto the vehicle network has changed the threat landscape. In this analysis, we have highlighted several places where mitigations are recommended to increase the difficulty of remote attacks on HVEDR systems and vehicle systems in general. These mitigations are listed in table 4 below.

SW		Include Security Requirements to Prevent Sending Vehicle Network Traffic via Abuse or Exploitation of Connected Module Software	(11.4)
SW		Make Connected Systems Software More Resilient	(12.0)
SW	HW	Restrict Access to Connected Module Remote Networks	(12.1)

Table 4: Connected Vehicle Systems Mitigations

Discussion on Connected Systems Mitigations

In addition to making the application software of connected systems harder to exploit and harder to find exploits for (see below *12.0*), considerations must also be made to make it harder to achieve the goal of sending arbitrary traffic on the vehicle network *after* a connected system is compromised by software exploitation.

The practical implementation of the requirement for *11.4* will vary, depending on the role of the contributor to the development of connected systems. For example, for OEMs of connected systems, they could specify security requirements to their suppliers. Suppliers could consider architectural attacks on their systems to ensure designs have sufficient layered defenses. System integrators could employ sandboxing, integrity verification, virtualization or other systems-level security technologies to meet the designs and security requirements thereof.

In the case of multiprocess operating systems which may be hosting the connected systems software, the ability to send vehicle network traffic should be implemented as a privilege and the systems design should separate that privilege from other at-risk software and assign it minimally to make it more difficult for attackers to escalate to that privilege in the event of a compromise by software exploitation.

In the case of vehicle owners incorporating connected modules over which they have little control; the installation of vehicle network isolation elements could be a useful strategy. This is especially true for mandated electronic logging devices (ELDs).

Mitigation *11.4* was captured in the analysis separately from the *resiliency* mitigations to highlight the fact that mitigations against arbitrary control of vehicle network traffic should be considered also at a systems level.

Conclusion

Using digital data in a forensic context is predicated on trust. The courts need to trust the data being presented in court. To date, this trust is often only established by the forensic investigator and expert witness. However, advancements in adversarial cyber-activities will necessitate additional methods to establish trust for digital data from heavy vehicles. Requiring the forensic investigator to acquire and maintain the skills necessary to definitively prove the absence of cyber-attacks on the forensic data may be impractical, which means the forensic process must be resilient against attackers.

The threats facing digital forensic data were presented as an attack tree to create a comprehensive list of mitigations for the attacker goal introduced in section *Attacker's Goal: Compromised HVEDR Data* where HVEDR data is presented that does not accurately represent the physical phenomena under question. The attack tree model covers the acquisition, transfer, storage, and verification of the data. It also addresses scaling and offset issues when converting digital data in to scaled engineering units, but verifying trust the data values

requires testing and comparison to external reference measurements. This attack tree model represents the *dual* of the trust model introduced in the earliest sections of this paper.

The analysis revealed, for example, that increased attack surface through vehicle connectivity solutions can increase the burdens on the forensic investigator to address potential alteration of data in different phases of the forensic process. Current processes of using local connections with diagnostic software may lack high assurance methods to maintain data integrity. The attack trees show 28 different potential attack vectors or error conditions, some of them impractical, but many that are quite realistic.

Once the threats were enumerated, mitigation strategies were proposed. The model places the mitigations in the context of which attacks or error conditions that they will mitigate. For example, by simply signing forensic data at the time of acquisition with cryptographically sound techniques, like the Pretty Good Privacy (PGP) framework, will mitigate the risk of altering data records after download and will do so on vehicle networks deployed today. See mention of *Verifiable Integrity Recordings of all Vehicle Network Traffic During Download (3.17)* above for more context and details.

These mitigations (which can be mapped equivalently to security requirements after some effort to rework the tone and tact of the mitigation), if deployed correctly, could alleviate the burden on the investigator to maintain high assurance of the digital forensic data. This paper expanded on what security requirements are needed to ensure the continued high assurance of the forensic data extraction process. Some technologies, mechanisms, designs or techniques would be sufficient for these security requirements; but no specific solutions are proposed as implementations of the requirements and additional solutions are not only possible, but encouraged.

Digital forensic data from heavy vehicles is critical in determining crash causation and the trust in this data is paramount. Improving the cybersecurity controls on this forensic data is necessary for society to adopt more data-driven technologies such as autonomous operations.

Acknowledgments

The authors are grateful for the support of the University of Tulsa and Cloakware for Connected Transport by Irdeto. Also for the thoughtful suggestions for improvement by Eric Thomas Swenson of Fort Wayne, Indiana.

References

[1] B. Schneier, "Attack Trees - Modeling security threats," *Dr. Dobb's Journal*, December 1999.

[2] A. Shostack, *Threat modeling: Designing for security.* John Wiley & Sons, 2014.

[3] A. Ruddle, D. Ward, B. Weyl, S. Idrees, Y. Roudier, M. Friedewald, T. Leimbach, A. Fuchs, S. Gürgens, O. Henniger, *et al.*, "Security requirements for automotive on-board networks based on dark-side scenarios," *EVITA Deliverable D*, vol. 2, p. 3, 2009.

[4] R. McKemmish, "When is digital evidence forensically sound?," in *IFIP International Conference on Digital Forensics*, pp. 3–15, Springer, 2008.

[5] J. Johnson, J. Daily, and A. Kongs, "On the digital forensics of heavy truck electronic control modules," *SAE International Journal of Commercial Vehicles*, vol. 7, pp. 72–88, apr 2014.

[6] SAE International, "Serial Control and Communications Heavy Duty Vehicle Network - Vehicle Application Layer," 2016.

[7] T. P. Austin and M. J. Farrell, "An examination of snapshot data in caterpillar electronic control modules," *SAE International Journal of Passenger Cars - Mechanical Systems*, vol. 4, pp. 611–635, apr 2011.

[8] F. P. Bayan, A. D. Cornetto, A. L. Dunn, C. B. Tanner, E. S. Sauer, B. Boggess, D. R. Morr, R. L.

Stansifer, S. A. Noll, G. J. Heydinger, and D. A. Guenther, "Comparison of heavy truck engine control unit hard stop data with higher-resolution on-vehicle data," *SAE International Journal of Commercial Vehicles*, vol. 2, pp. 29–38, apr 2009.

[9] J. Daily, A. Kongs, J. Johnson, and J. Corcega, "Extracting event data from memory chips within a detroit diesel DDEC v," in *SAE Technical Paper Series*, SAE International, apr 2015.

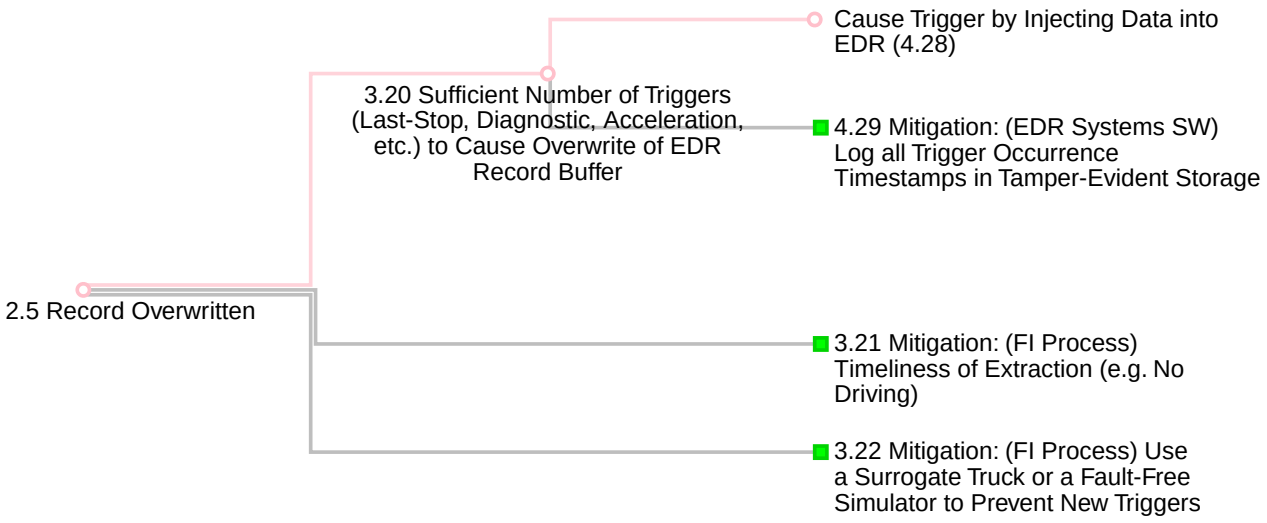
[10] W. Messerschmidt, T. Austin, B. Smith, T. Cheek, D. Plant, and C. Voeglie, "Simulating the effect of collision-related power loss on the event data recorders of heavy trucks," in *SAE Technical Paper Series*, SAE International, apr 2010.

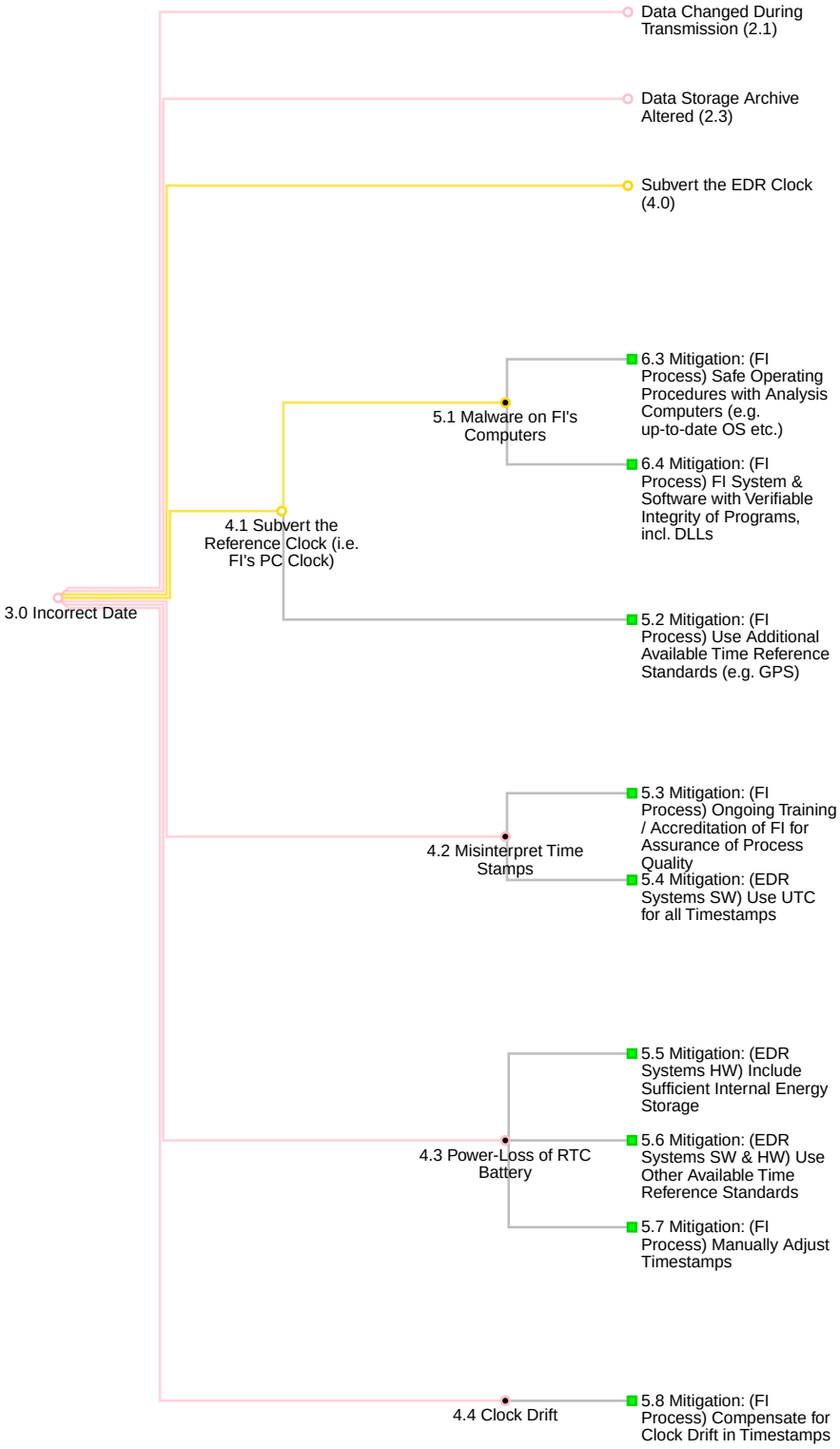
[11] J. Daily, J. Johnson, and A. Perera, "Recovery of partial caterpillar snapshot event data resulting from power loss," in *SAE Technical Paper Series*, SAE International, apr 2016.

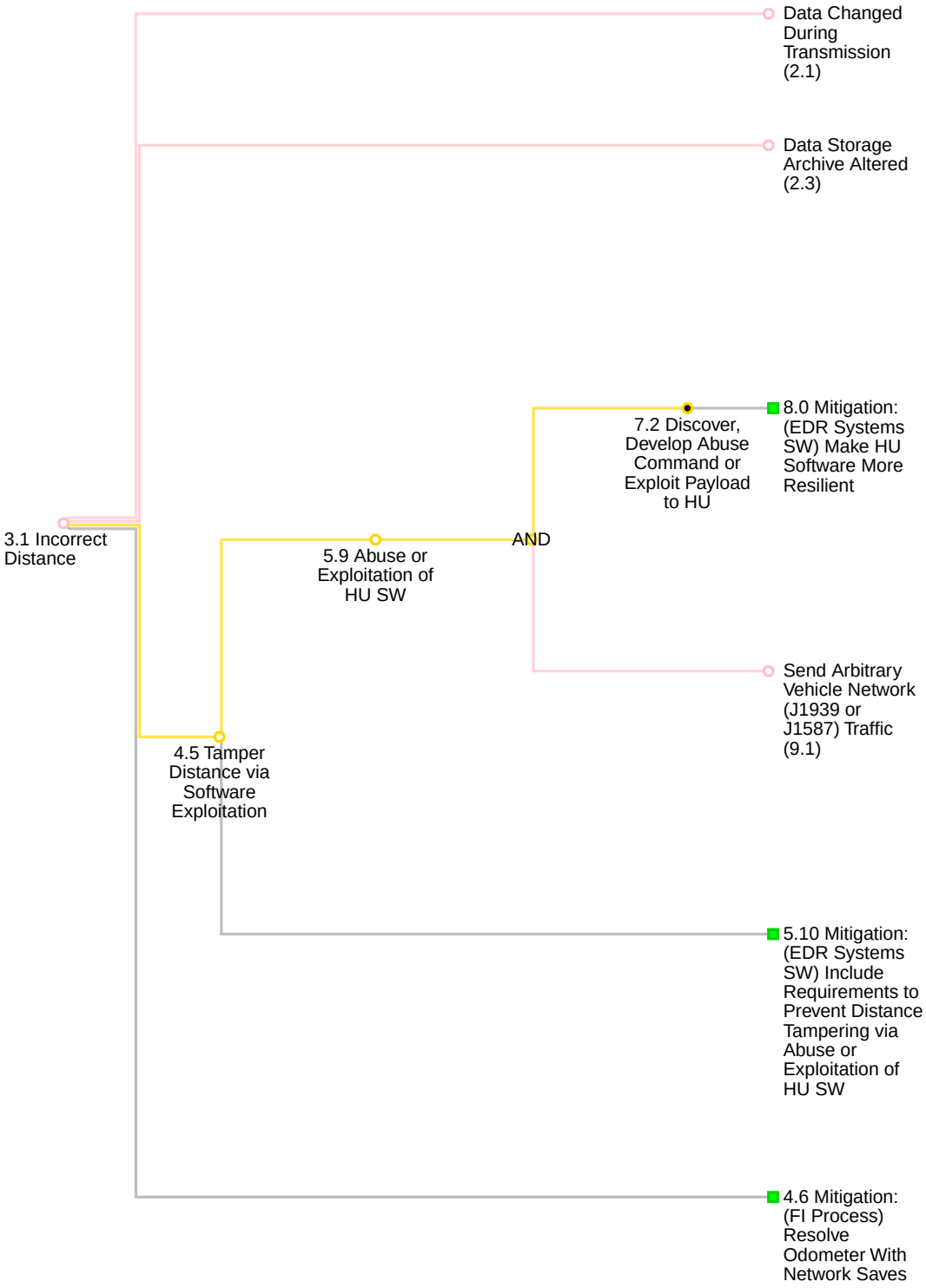
[12] V. E. S. S. Committee *et al.*, "Sae j3061 cybersecurity guidebook for cyber-physical automotive systems," *Society of Automotive Engineers*, 2016.

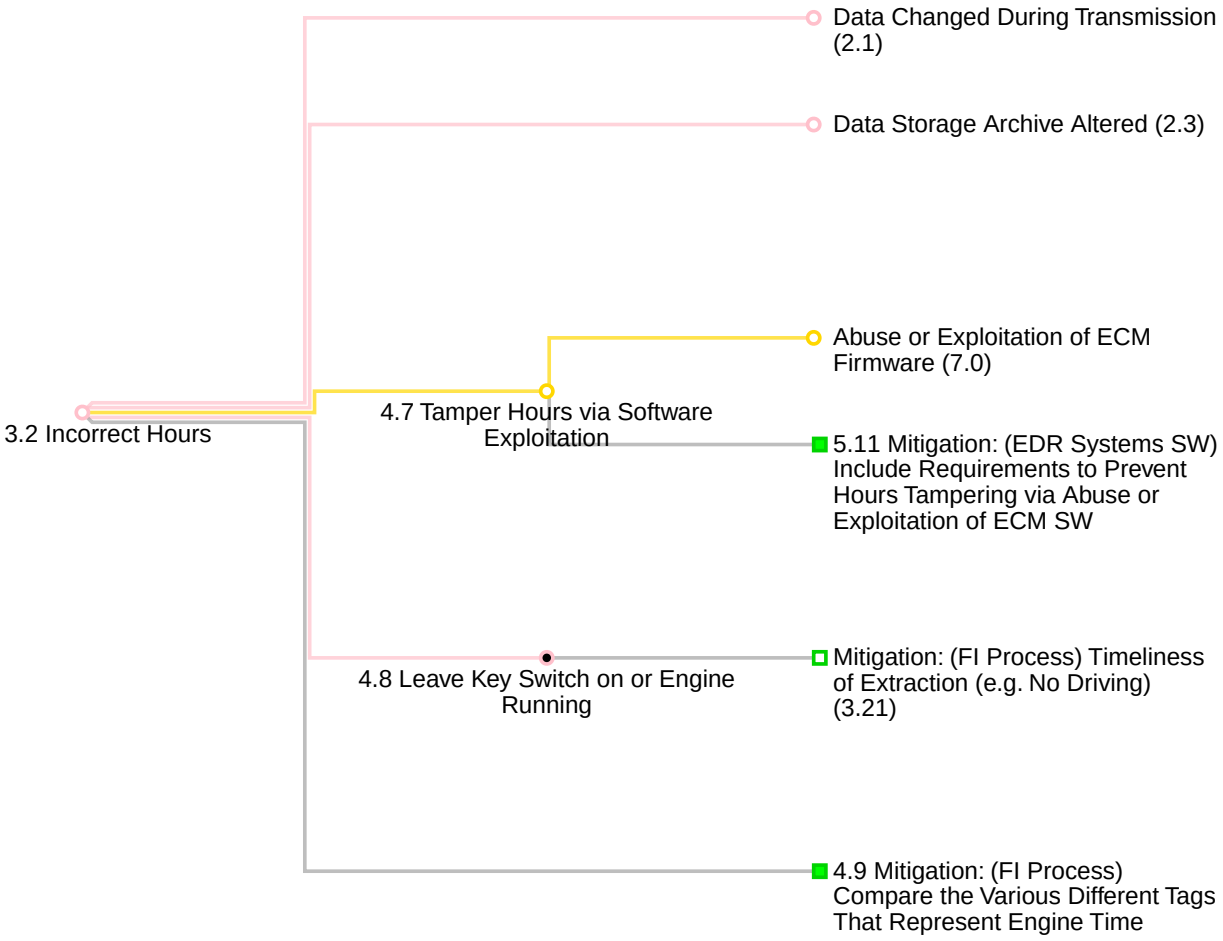
Appendix: Full Attack Tree

We include here the remaining subtrees which complete the rendering of the remainder of the attack tree, for reference. We direct interested readers also to a companion website which includes an interactive complete attack tree and generated TARA report for the model presented in this paper: https://heavy-vehicle-networking-at-u-tulsa.github.io/CybersecurityForHVDERs/ .

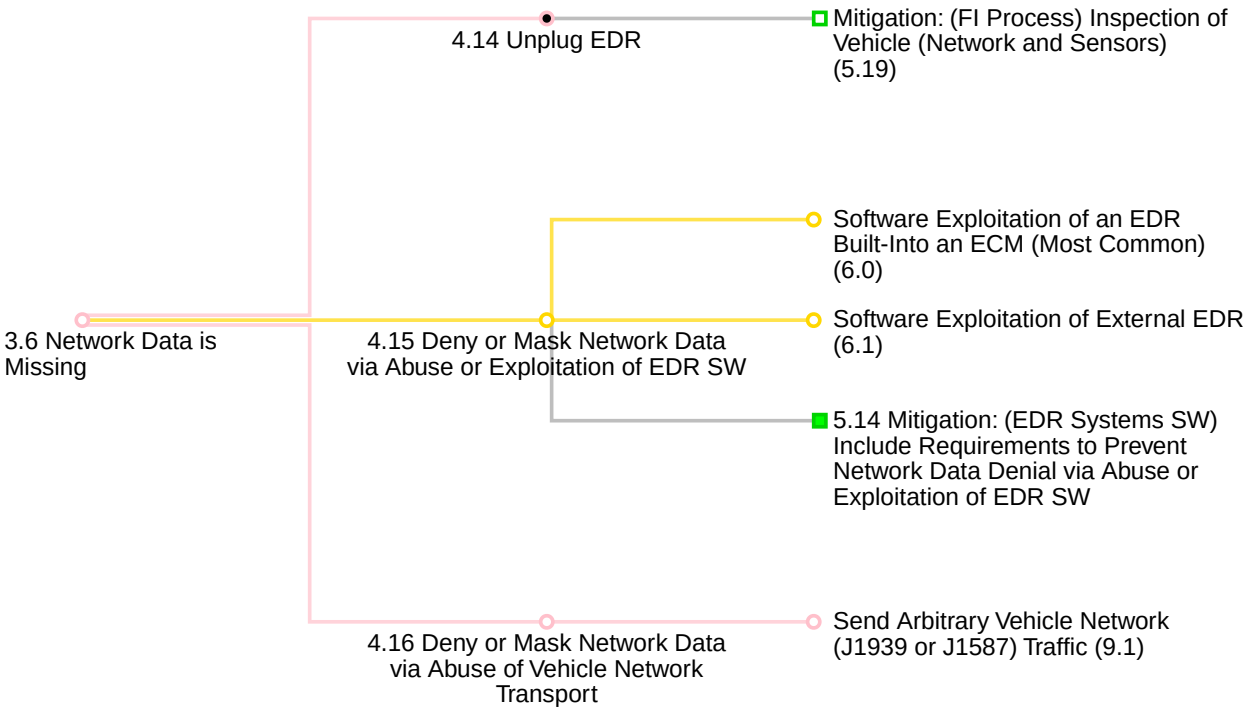


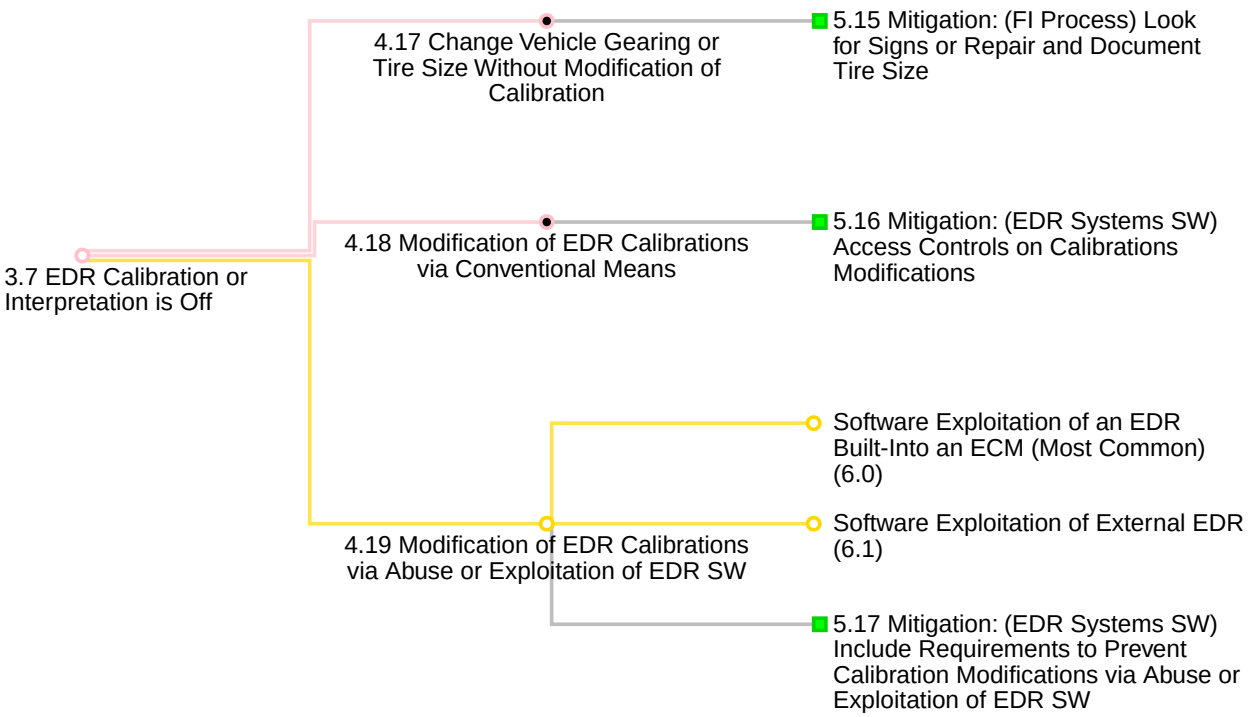

Data Changed During Transmission (2.1)
Data Storage Archive Altered (2.3)
Subvert the EDR Clock (4.0)
6.3 Mitigation: (FI Process) Safe Operating Procedures with Analysis Computers (e.g. up-to-date OS etc.)
5.1 Malware on FI's Computers
6.4 Mitigation: (FI Process) FI System & Software with Verifiable Integrity of Programs, incl. DLLs
4.1 Subvert the Reference Clock (i.e. FI's PC Clock)
3.0 Incorrect Date
5.2 Mitigation: (FI Process) Use Additional Available Time Reference Standards (e.g. GPS)
5.3 Mitigation: (FI Process) Ongoing Training / Accreditation of FI for Assurance of Process Quality
4.2 Misinterpret Time Stamps
5.4 Mitigation: (EDR Systems SW) Use UTC for all Timestamps
5.5 Mitigation: (EDR Systems HW) Include Sufficient Internal Energy Storage
4.3 Power-Loss of RTC Battery
5.6 Mitigation: (EDR Systems SW & HW) Use Other Available Time Reference Standards
5.7 Mitigation: (FI Process) Manually Adjust Timestamps
4.4 Clock Drift
5.8 Mitigation: (FI Process) Compensate for Clock Drift in Timestamps

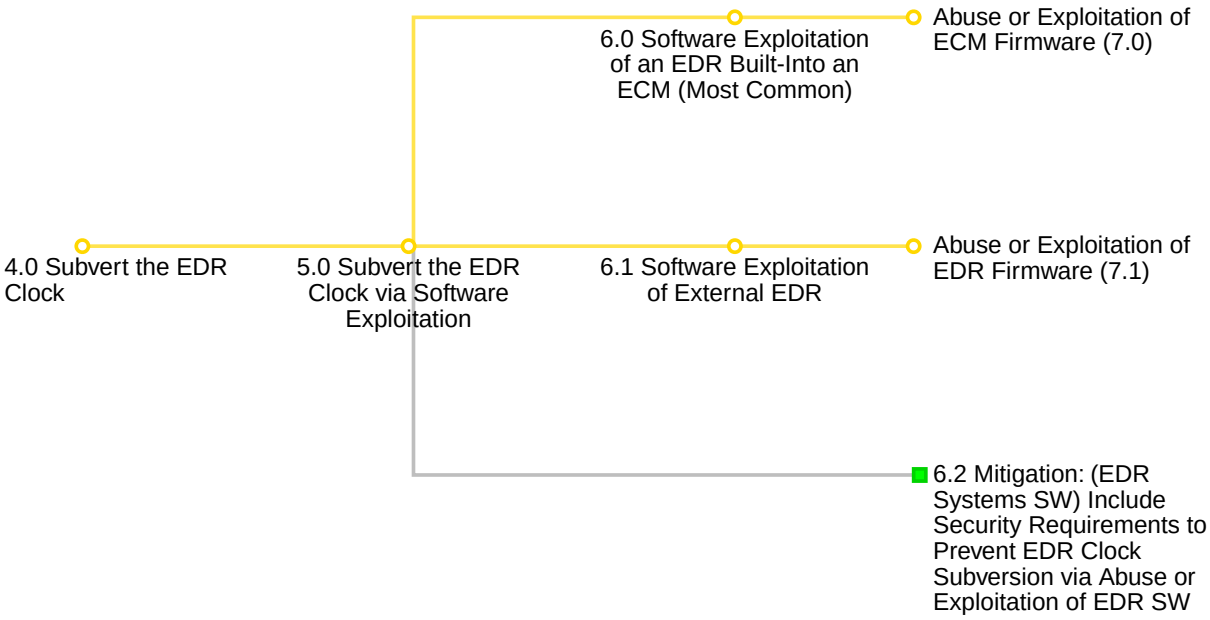

Data Changed During Transmission (2.1)
Data Storage Archive Altered (2.3)
8.0 Mitigation: (EDR Systems SW) Make HU Software More Resilient
7.2 Discover, Develop Abuse Command or Exploit Payload to HU
AND
3.1 Incorrect Distance
5.9 Abuse or Exploitation of HU SW
Send Arbitrary Vehicle Network (J1939 or J1587) Traffic (9.1)
4.5 Tamper Distance via Software Exploitation
5.10 Mitigation: (EDR Systems SW) Include Requirements to Prevent Distance Tampering via Abuse or Exploitation of HU SW
4.6 Mitigation: (FI Process) Resolve Odometer With Network Saves

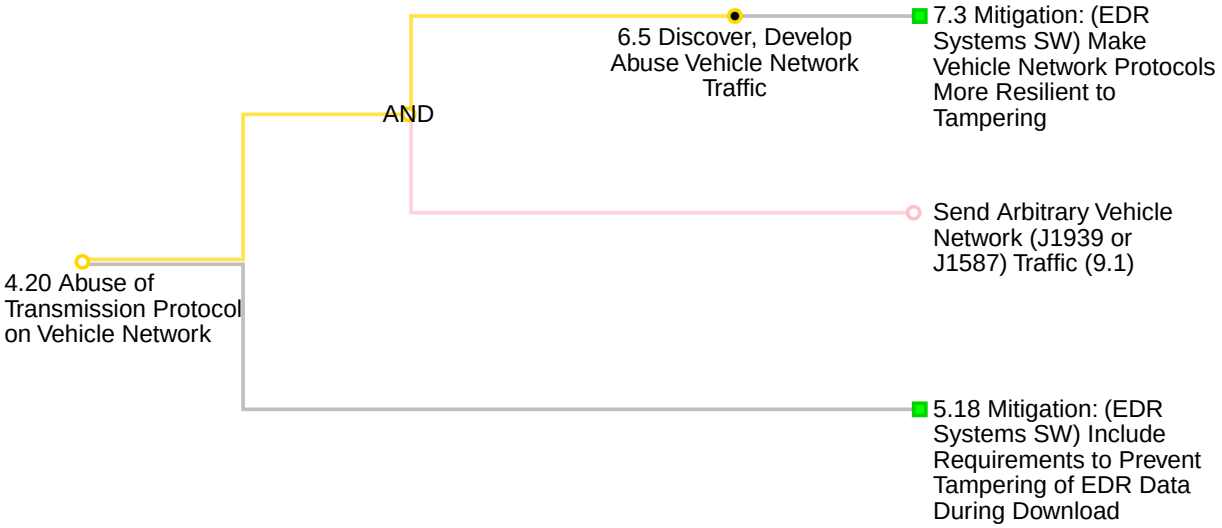

Data Changed During Transmission (2.1)
Data Storage Archive Altered (2.3)
Abuse or Exploitation of ECM Firmware (7.0)
4.7 Tamper Hours via Software Exploitation
3.2 Incorrect Hours
5.11 Mitigation: (EDR Systems SW) Include Requirements to Prevent Hours Tampering via Abuse or Exploitation of ECM SW
Mitigation: (FI Process) Timeliness of Extraction (e.g. No Driving) (3.21)
4.8 Leave Key Switch on or Engine Running
4.9 Mitigation: (FI Process) Compare the Various Different Tags That Represent Engine Time

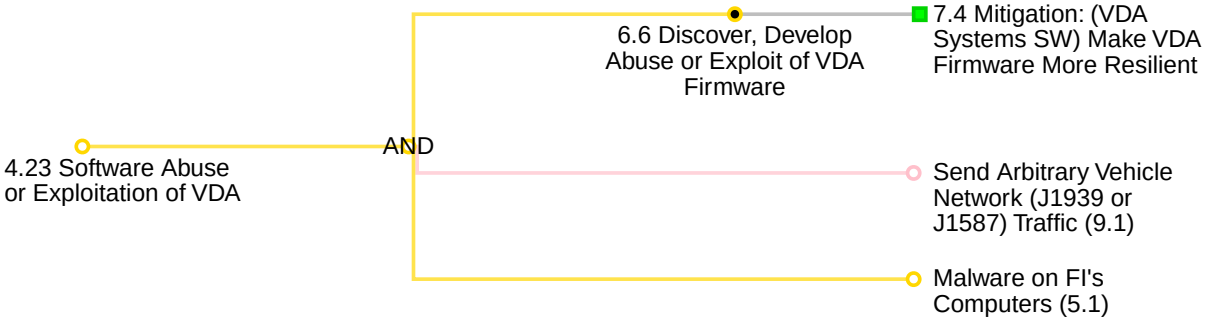

Data Changed During Transmission (2.1)
Data Storage Archive Altered (2.3)
3.3 Missing Identifiers
Software Exploitation of an EDR Built-Into an ECM (Most Common) (6.0)
Software Exploitation of External EDR (6.1)
4.10 Tamper Identifiers via Software Exploitation
5.12 Mitigation: (EDR Systems SW) Include Requirements to Prevent Identifiers Tampering via Abuse or Exploitation of EDR SW
4.11 Mitigation: (FI Process) Photograph the physical HVEDR in-situ With Identifiers
3.4 Insufficient Context
4.12 Obfuscated Context
5.13 Mitigation: (FI Process) Use Other Forensic Evidence to Properly Correlate EDR Records to Correct Vehicle and Event

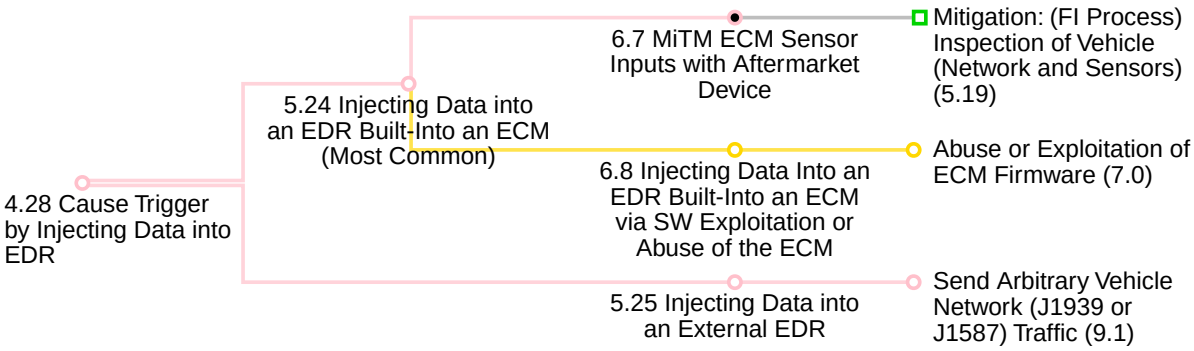

3.6 Network Data is Missing
4.14 Unplug EDR
Mitigation: (FI Process) Inspection of Vehicle (Network and Sensors) (5.19)
4.15 Deny or Mask Network Data via Abuse or Exploitation of EDR SW
Software Exploitation of an EDR Built-Into an ECM (Most Common) (6.0)
Software Exploitation of External EDR (6.1)
5.14 Mitigation: (EDR Systems SW) Include Requirements to Prevent Network Data Denial via Abuse or Exploitation of EDR SW
4.16 Deny or Mask Network Data via Abuse of Vehicle Network Transport
Send Arbitrary Vehicle Network (J1939 or J1587) Traffic (9.1)

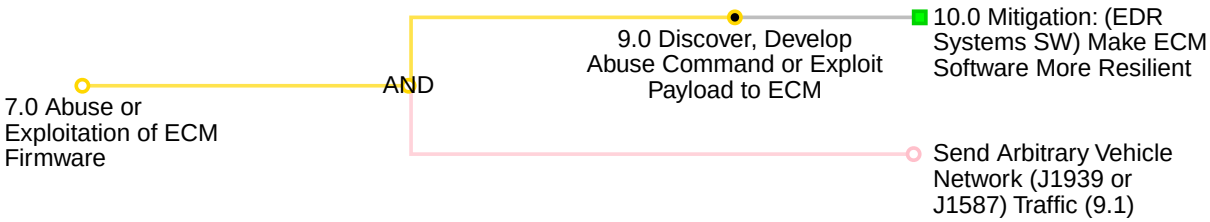

4.17 Change Vehicle Gearing or Tire Size Without Modification of Calibration
5.15 Mitigation: (FI Process) Look for Signs or Repair and Document Tire Size
4.18 Modification of EDR Calibrations via Conventional Means
5.16 Mitigation: (EDR Systems SW) Access Controls on Calibrations Modifications
3.7 EDR Calibration or Interpretation is Off
Software Exploitation of an EDR Built-Into an ECM (Most Common) (6.0)
Software Exploitation of External EDR (6.1)
4.19 Modification of EDR Calibrations via Abuse or Exploitation of EDR SW
5.17 Mitigation: (EDR Systems SW) Include Requirements to Prevent Calibration Modifications via Abuse or Exploitation of EDR SW

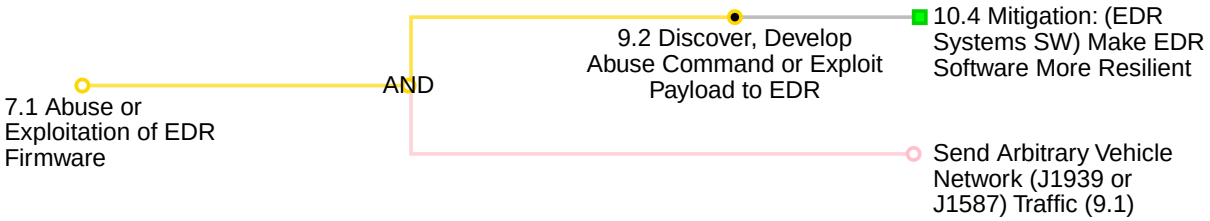

6.0 Software Exploitation of an EDR Built-Into an ECM (Most Common)
Abuse or Exploitation of ECM Firmware (7.0)
4.0 Subvert the EDR Clock
5.0 Subvert the EDR Clock via Software Exploitation
6.1 Software Exploitation of External EDR
Abuse or Exploitation of EDR Firmware (7.1)
6.2 Mitigation: (EDR Systems SW) Include Security Requirements to Prevent EDR Clock Subversion via Abuse or Exploitation of EDR SW

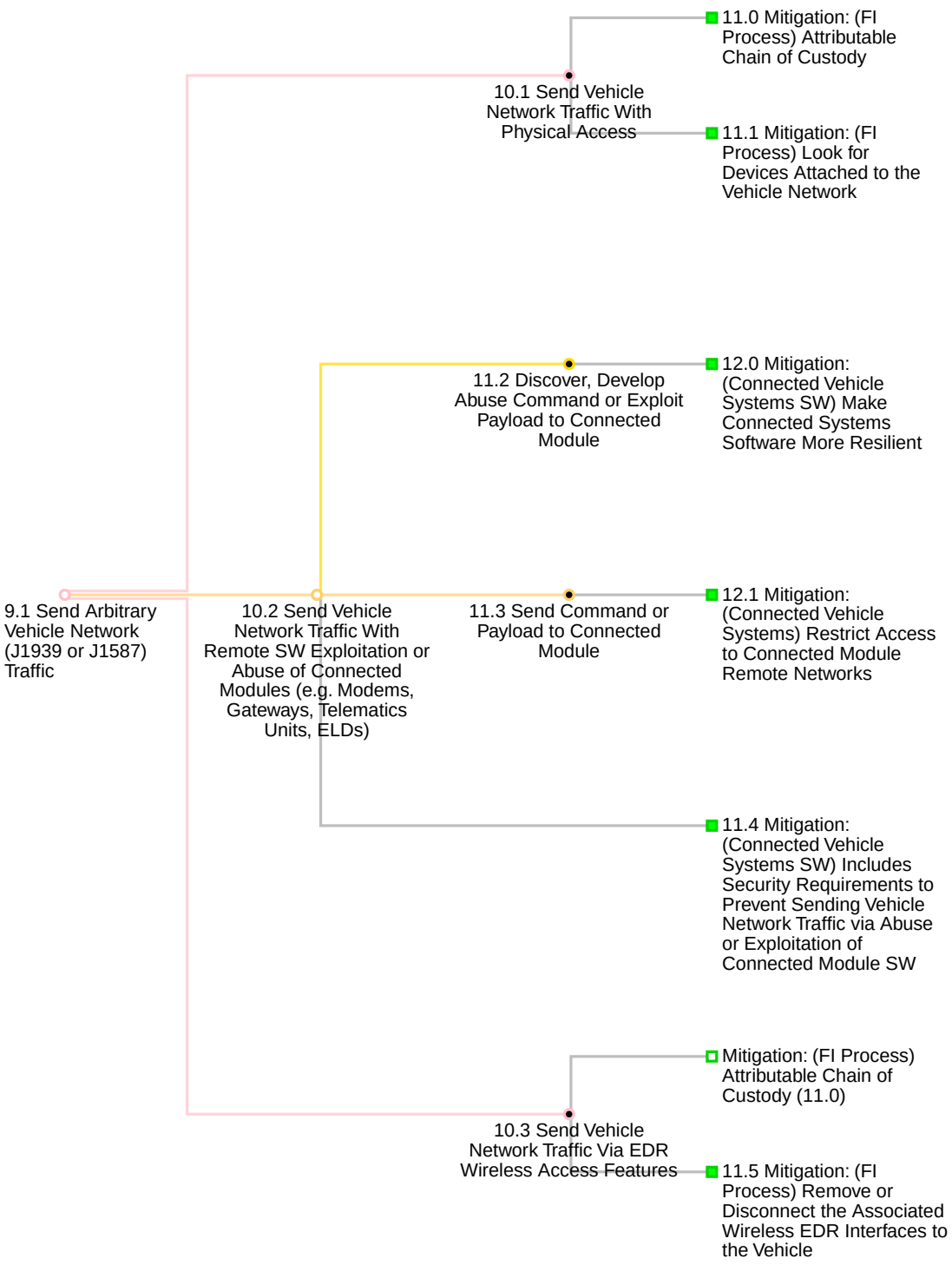

4.20 Abuse of Transmission Protocol on Vehicle Network
AND
6.5 Discover, Develop Abuse Vehicle Network Traffic
7.3 Mitigation: (EDR Systems SW) Make Vehicle Network Protocols More Resilient to Tampering
Send Arbitrary Vehicle Network (J1939 or J1587) Traffic (9.1)
5.18 Mitigation: (EDR Systems SW) Include Requirements to Prevent Tampering of EDR Data During Download


4.23 Software Abuse or Exploitation of VDA
AND
6.6 Discover, Develop Abuse or Exploit of VDA Firmware
7.4 Mitigation: (VDA Systems SW) Make VDA Firmware More Resilient
Send Arbitrary Vehicle Network (J1939 or J1587) Traffic (9.1)
Malware on FI's Computers (5.1)


4.28 Cause Trigger by Injecting Data into EDR
5.24 Injecting Data into an EDR Built-Into an ECM (Most Common)
6.7 MiTM ECM Sensor Inputs with Aftermarket Device
Mitigation: (FI Process) Inspection of Vehicle (Network and Sensors) (5.19)
6.8 Injecting Data Into an EDR Built-Into an ECM via SW Exploitation or Abuse of the ECM
Abuse or Exploitation of ECM Firmware (7.0)
5.25 Injecting Data into an External EDR
Send Arbitrary Vehicle Network (J1939 or J1587) Traffic (9.1)


7.0 Abuse or Exploitation of ECM Firmware
AND
9.0 Discover, Develop Abuse Command or Exploit Payload to ECM
10.0 Mitigation: (EDR Systems SW) Make ECM Software More Resilient
Send Arbitrary Vehicle Network (J1939 or J1587) Traffic (9.1)


7.1 Abuse or Exploitation of EDR Firmware
AND
9.2 Discover, Develop Abuse Command or Exploit Payload to EDR
10.4 Mitigation: (EDR Systems SW) Make EDR Software More Resilient
Send Arbitrary Vehicle Network (J1939 or J1587) Traffic (9.1)


9.1 Send Arbitrary Vehicle Network (J1939 or J1587) Traffic
10.1 Send Vehicle Network Traffic With Physical Access
10.2 Send Vehicle Network Traffic With Remote SW Exploitation or Abuse of Connected Modules (e.g. Modems, Gateways, Telematics Units, ELDs)
10.3 Send Vehicle Network Traffic Via EDR Wireless Access Features
11.0 Mitigation: (FI Process) Attributable Chain of Custody
11.1 Mitigation: (FI Process) Look for Devices Attached to the Vehicle Network
11.2 Discover, Develop Abuse Command or Exploit Payload to Connected Module
11.3 Send Command or Payload to Connected Module
11.4 Mitigation: (Connected Vehicle Systems SW) Includes Security Requirements to Prevent Sending Vehicle Network Traffic via Abuse or Exploitation of Connected Module SW
11.5 Mitigation: (FI Process) Remove or Disconnect the Associated Wireless EDR Interfaces to the Vehicle
12.0 Mitigation: (Connected Vehicle Systems SW) Make Connected Systems Software More Resilient
12.1 Mitigation: (Connected Vehicle Systems) Restrict Access to Connected Module Remote Networks
Mitigation: (FI Process) Attributable Chain of Custody (11.0)