Bus-Off Attack: Adversary Model

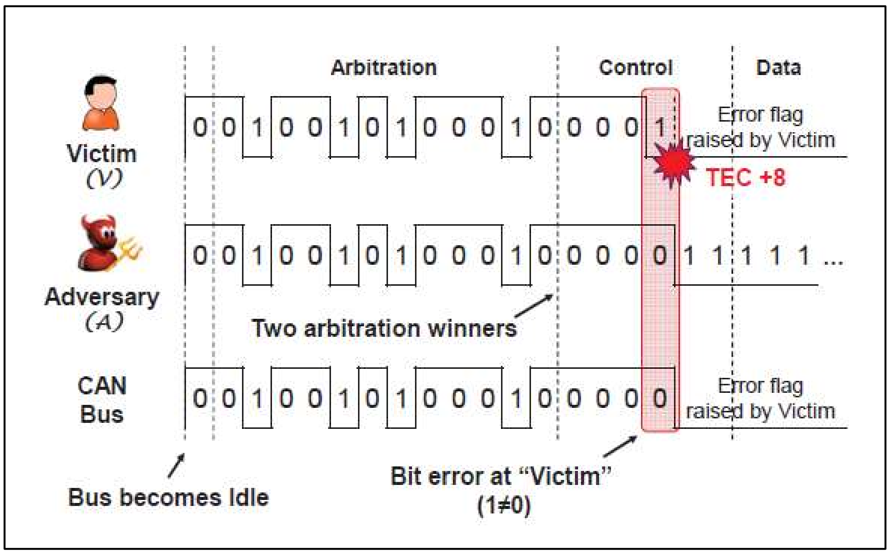
- The attacker can compromise an in-vehicle ECU either physically or remotely to gain its control.
- We do not require the adversary to reverse engineer messages or checksums to be able to deliver the attack. In fact, messages have different structures depending on the manufacturer, but this is not central in bus-off attacks.
- Once the ECU is compromised, the attacker can inject messages with any ID, DLC and data on the CAN bus.

2.2.1 The Bus-Off attack

Suppose that a victim V periodically sends a message M. The attacker can hence deliver a successful bus-off attack if all the following conditions are satisfied:

1. **C1: using the same ID**. In this way there is no winner and both ECUs have the same priority, causing an error on the bus (what an attacker wants to exploit).
2. **C2: transmitting at the same time as M** (the tricky part of the attack). If the attacker transmission is not aligned with victim one the attack fails.
3. **C3: having at least a bit that is dominant in the attack message while being recessive in M** (all preceding bits are equal).

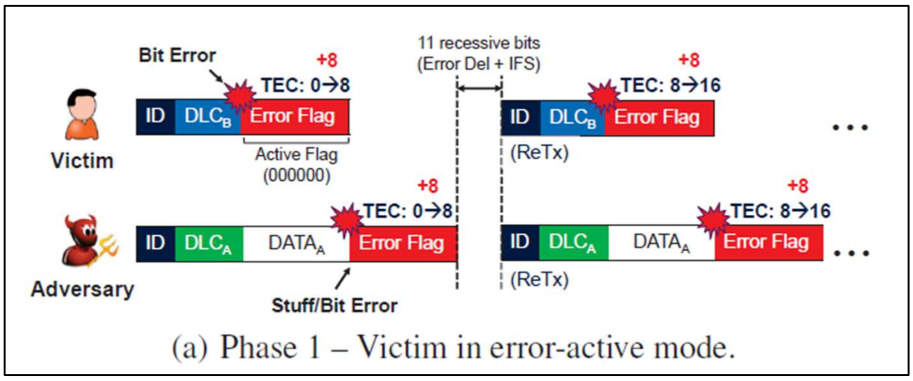
The first step for an attacker is to observe and listen the channel to collect IDs and observe the **messages periodicity**, then it is possible to plan the attack using the gathered information.



Phase 1: Victim in Error-Active

- Both adversary and victim start in error-active mode, and the attacker targets one of the victim's periodical messages.
- The attacker sends the malicious message and the victim's TEC increases by 8.
- The attacker's TEC also increases by 8, due to the bit errors triggered at the adversary node (active error flag transmitted by the victim that violates the bit stuffing rule).

- Both nodes **automatically retransmit the failed message**.
- Thanks to the automatic retransmission, the attacker brings the victim to error-passive by sending a single message.

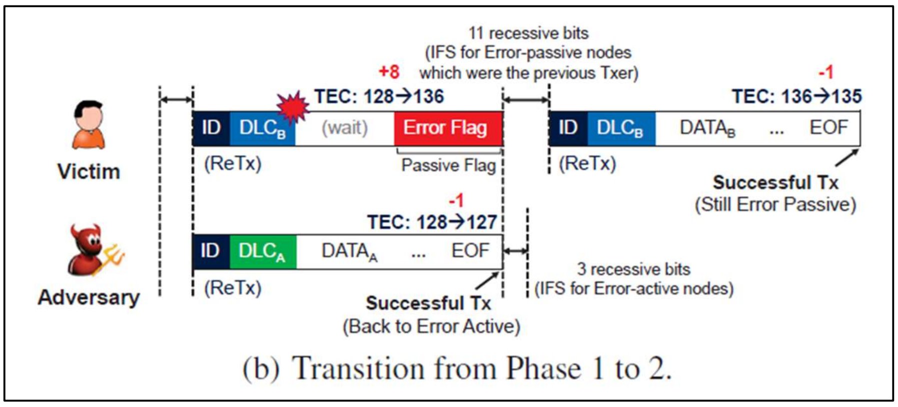


(a) Phase 1 – Victim in error-active mode.

Note: the victim's error counter increases due to failed transmission of packet while attacker has an error due to the victim's active error flag (6 dominant bits) that blocks the adversary transmission. So, the **victim and the attacker TECs increase at different times and for different reasons**. This is fundamental to understand for the success of the attack.

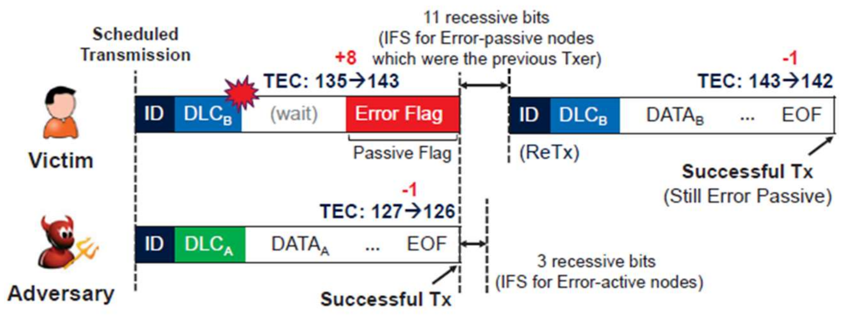
Phase 1 to 2

- After 16 retransmissions, both victim's and attacker's TEC = 128 → error-passive.
- Therefore, the victim attempts the delivery of a passive error flag with 6 recessive bits.
- **The passive error transmitted by the victim cannot override what is in the CAN bus because recessive bits cannot override dominant ones**. So, the attacker successfully transmits its message and return to error active (attacker's TEC = TEC - 1).
- Due to the successful transmission, the adversary does not go to error-passive.
- The victim can transmit later the error passive frame decreasing its TEC by 1 but remaining in error passive mode.
- Note: when the adversary node receives the victim's passive error frame, its TEC does not increment since **TEC increments whenever a node transmission is disrupted**. During the phase 1 adversary's TEC incremented because the error frame sent by the victim contained 6 dominant bits which override every frame on the bus, even the one the adversary had to retransmit.


11 recessive bits (IFS for Error-passive nodes which were the previous Txer)
+8
TEC: 128→136
ID
DLC_B
(wait)
Error Flag
Passive Flag
(ReTx)
-1
TEC: 136→135
ID
DLC_B
DATA_B
...
EOF
(ReTx)
Successful Tx (Still Error Passive)
Victim
-1
TEC: 128→127
ID
DLC_A
DATA_A
...
EOF
(ReTx)
Successful Tx (Back to Error Active)
3 recessive bits (IFS for Error-active nodes)
Adversary
(b) Transition from Phase 1 to 2.

Phase 2: Victim in Error-Passive

- Once the scheduled interval of the target message is elapsed, V retransmits M again.
- At the same time, the adversary reinjects malicious M.
- Since the victim is in error-passive, the attacker's TEC decreases by 1, whereas the victim's increases by 7 (+8 -1) thus maintaining the error-passive.
- The attacker iterates until the victim's TEC > 255, forcing the victim ECU to bus-off state.



(c) Phase 2 – Victim in error-passive mode.

Note: remember that CAN ID do not contain actual information on the transmitter. The ID contains information that define the **time interval** and **priority of messages**. The attacker should hit low-value IDs (dominant) to disconnect possibly safety-critical ECUs.

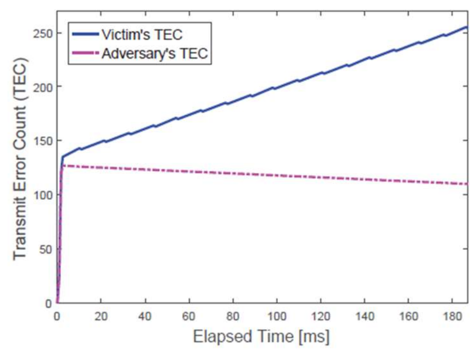


Figure 12: TECs during a bus-off attack.

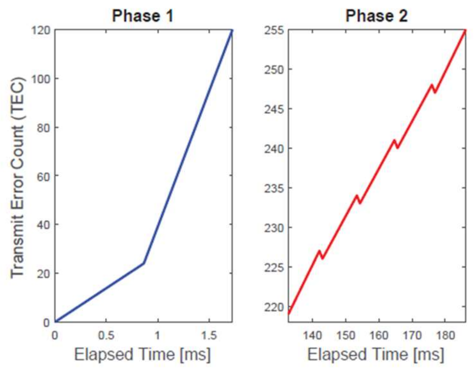


Figure 13: Victim's TEC during Phase 1 and 2.

Where to modify frames?

To trigger a bit error at the victim and increase its TEC, the adversary's attack message must satisfy C3: it must have at least one bit position where its signal is dominant (0) while the victim's is recessive (1), with all preceding bits being identical. Since the bus-off attack requires the attack and target messages to have the same ID (C1), this mismatch must occur in either the **control** or **data field**, as fields like CRC and ACK are managed by the CAN controller and not controllable by the attacker. A practical and effective way to achieve this is for the adversary to set its message's DLC or data values to all 0s. Given that the DLC for a specific CAN ID is typically constant over time, the attacker can observe and match this value to craft a message that satisfies C3, ensuring a dominant-recessive bit mismatch in the victim's transmission.

DIFFICULTIES OF THE BUS-OFF ATTACK

ECUs cannot acquire the IDs of all received messages. Only of those that pass through its message filter. We can distinguish hence between *received message* (the ECU can only see and read the content) and *accepted message* (messages that contains some information related of what the ECU has to do, some physical operations for instance).
The attacker can only read the ID from accepted messages, and he can create only messages with the same ID depends on the filter of the victim ECU → if there is no filter then done, but filters can be remotely manipulated.

Non-Empty Message Filter

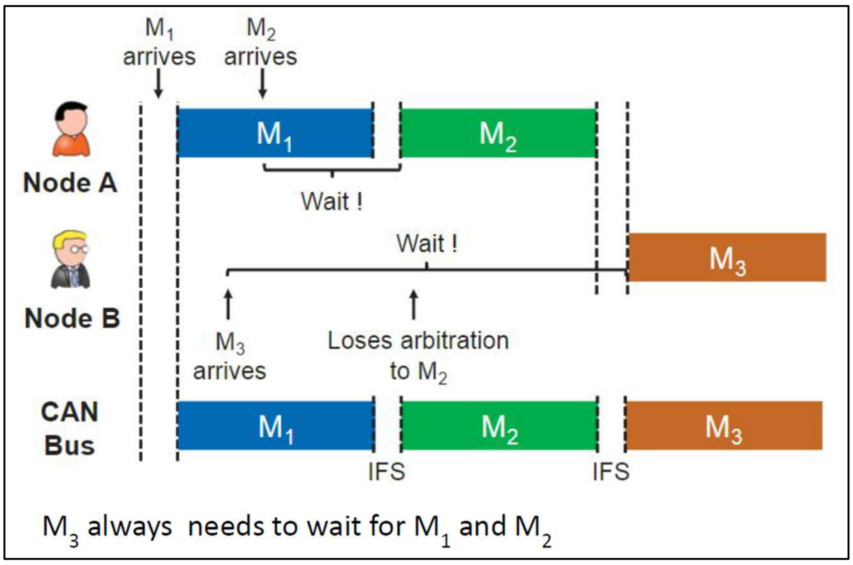
Instead of what Empty Message Filters do, here the compromised ECU only accepts messages with a certain (range of) ID. This however does not represent a restriction for the attacker in compromising it. The attacker can directly modify the message filter of the compromised ECU via software commands. For some controllers, the configuration mode can be triggered via user instruction through the Serial Peripheral Interface.

Another problem of the bus-off attack is that we need **precise synchronization**, so if an attacker does not transmit the malicious packet as soon as the victim is transmitting its packet then the attack is not going to work. It is possible to achieve the synchronization without taking care about the exact time of the victim's transmission: **all messages and priorities are periodic, and we can exploit this to infer when a certain message is going to be transmitted** (preceded message). Usually when a packet's transmission in the CAN bus ends, we wait for 3 bit-time before starting the transmission (IFS).

PRECEDED ID

Although CAN messages are periodic, **jitter** (time variation on sending messages) **may affect the effectiveness of the attack**. To overcome this issue the attacker can exploit the fact that nodes, which have either lost arbitration or had new messages buffered while the bus was busy, attempt to transmit their messages as soon as the bus becomes idle. Given a node with ID M, we define its **preceded ID as the ID of the message that was transmitted immediately before the one the attacker is targeting**.

Since the order of message transmission is determined by the priority of the IDs, and IDs are fixed, the preceded ID provides a reliable indicator for the attacker. By knowing the ID of the message that comes right before the victim's message, the attacker can better predict when the target message will start → Transmit time = 3 bits after the completion of the message with preceded ID. If the target message has no preceded ID, the attacker can create one and send both the preceded ID and the attack message.



Consider an example where node A transmits messages with ID=M1, M2, and node B transmits a message with ID=M3 which has the lowest priority among them. As shown in the above figure, if these messages are arriving and being queued at the depicted times, **M1 and M2 would be the preceded IDs of M2 and M3**, respectively, with only a 3-bit IFS separating the corresponding message pairs. In other words, **the transmissions of M2 and M3 are forced to be buffered until their preceded ID messages have been transmitted on the bus**. Since message priorities and periodicities do not change, such a feature implies that **one specific CAN message may always be followed by another specific message, i.e., there is a unique preceded ID for that specified one**.

As an example, if the periodicities of their transmissions are either same or integer multiples (e.g., 5ms for M1, M2, and 10ms for M3), then M2 would always be the preceded ID of M3, i.e., be the unique preceded ID. Hence, **regardless of jitter, the exact timing of message transmissions becomes rather predictable** and even determinative: **3 bit-time after the preceded ID's completion**.