Freshness analysis of functional sequences in launchers

Dorine Petit, Jean-Philippe Georges, Thierry Divoux, Bruno Regnier, Philippe Miramont

Freshness analysis of functional sequences in launchers ⋆

Dorine Petit [*,**] **Jean-Philippe Georges** [*,**]
Thierry Divoux [*,**] **Bruno Regnier** [***] **Philippe Miramont** [***]

[*] *Université de Lorraine, CRAN, UMR 7039, Campus Sciences, BP 70239, Vandœuvre-lès-Nancy Cedex, 54506, France (e-mail: firstname.name@univ-lorraine.fr)*
[**] *CNRS, CRAN, UMR 7039, France*
[***] *CNES, Direction des Lanceurs, 52 rue Jacques Hillairet, 75612, Paris, France (e-mail: firstname.name@cnes.fr)*

Abstract: Nowadays, many embedded systems use specific data buses to guarantee the exchange of data. The next generation of space applications is moving to components off-the-shelf (COTS) technology as switched Ethernet network in order to reduce the financial cost, being attentive to the mass. However, performance should be gained without sacrificing the reliability and the properties that enabled a tele-monitoring. This paper focuses on how to obtain a single traffic capture file that will be sent over ground-board communications and how to test later the freshness requirement satisfaction. A freshness formalization is proposed as well as a freshness verification algorithm based on the analysis of single trace captured using multicast communications. An experimental evaluation has been leaded in nominal and link failure cases.

Keywords: Observability; freshness; switched network; tele-monitoring; space vehicles.

1. INTRODUCTION

In space applications (aircrafts, satellites, launchers), conventional communication technologies are evolving from specific buses and federative control strategies to switched topologies that will support modular and distributed architecture Monchaux et al. (2012). Component off-the-shelf (COTS) technology is aimed at replacing the current MIL-STD-1553B (Department of Defense (1978)) for control traffic and (TDMA based) Pulse-Coded Modulation buses for telemetry traffic, buses embedded in the european (unmanned) launchers. Switched Ethernet network are hence expected to be embedded into the next-generation of the space launchers (Robert et al. (2013)). Candidates protocols may be AFDX (ARINC 664 P7 (2003)) or TTEthernet (SAE AS 6802 (2011)).

Traffic monitoring can be the cornerstone for understanding such communication networks. The monitoring activity aims at collecting from the various network devices a set of relevant data. This enables to characterize the network state and therefore to identify unusual network behavior. According to the application domain, the purposes of the monitoring can also be different like network management, network security, network performance analysis, etc. The monitoring mechanisms depend directly on the intended application and also on the nature of the observed system.

In launchers, the on-board controller manages the control state by sending specific data from/to the sensors/actuators. It corresponds to traffic command (noted TC) for mission control, on-board operations scheduling and automated procedures. Additionally, the network supports telemetry traffic (noted TM) for low rate mission control and housekeeping data Notebaert (2014). By analyzing the content of the packets, it should be possible to retrieve the values measured by the sensors and those sent by the controller to the actuators. Hence, the network acts as an observer of the control state. This is the applicable approach for launchers. Indeed, the whole traffic is monitored on-board and both TC and TM exchanges are sent over a ground-board link. It enables tele-diagnosis of launchers by post-analysis studies and replaying a network in order to improve the performances of a given architecture. The Ethernet protocols used in launcher applications may be different from those used for industrial factory automation, but the demands on end-to-end delays and other real-time requirements remain in the same order of magnitude. The main difference comes from the fact that the diagnosis of the devices and applications states relies here on the packet captured on the network. There is no additional supervision traffic, such performance metrics like freshness have to be recovered from the traces sent by air-ground data link.

The objective of this paper is to propose a method to validate end-to-end freshness inside a network as part of an observability analysis. The main novelty consists in achieving such validation for launchers just based on a single frame capture file that will be monitored through a ground-board communication. Section 2 summarizes the related work concerning network monitoring. Section 3 details the challenges and restrictions of such method

⋆ Co-founded study by CNES and CRAN in the frame of CNES Launchers' Research and Technology program.

in which the network state is observed from a specific machine. It shows ambiguous situations to be taken into account for both freshness and ordering properties with regards to a sequence of frames. Section 4 explains how the method might be implemented while section 5 illustrates it on an experimental platform. It should be helpful for engineers to discuss about the performance of the avionics during a past launch and provide capacity to detect, isolate and report faults.

The processing time of each task will not be taken into account in this study. It is not assumed for the traffic to be periodic and it should be robust to dynamic change of the topology (for instance a booster separation) or a different mission of the flight.

2. RELATED WORK AND BACKGROUND

This paper aims at providing a framework to analyse the traffic capture. It will be helpful for trade-off of quality-of-service and to detect, isolate and report faults. For such critical real-time constrained application, numerous works have already focused on the performance evaluation of the network end-to-end delays (network calculus, trajectory approach, real-time calculus). However, avionics requires to verify different properties that encompass not a single, but a sequence of frames mapping a functional command. An important property is to test if the end-to-end freshness is satisfied as defined in Lauer et al. (2011). Consider the periodic chain $T_1 \xrightarrow{d_1} \dots T_n \xrightarrow{d_n} T_{n+1}$ (task T_i regularly produces data d_i for T_{i+1}; each task is computed on a given device). A δ-freshness requirement may be defined such that a data d_n is "globally" produced from data d_i themselves produced not earlier than δ time units before. From a traffic capture point of view, a functional sequence would be observed according to the frames $F_1 \dots F_n$ (that respectively corresponds to the transportation of data $d_1 \dots d_n$). The verification of the freshness requires then to evaluate the difference between the timestamp of the frame related to the event at the end of a functional sequence and the timestamp of the frame that corresponds to the earliest dependent event at the beginning of the sequence. The paper will show the challenges to retrieve such analysis: how to obtain a single traffic capture file that will be sent over ground-board communications and how to test later the freshness requirement satisfaction.

The first issue deals with the acquisition of a unique traffic capture file. There exist several different techniques to capture network traffic. A point-to-point link can be split with a special device, named network Test Access Point (TAP) which enables to connect a monitor on this particular link in a passive way. A second method, called port mirroring, consists of using a special switches function (available on the most of commercial switches), which enables to copy all traffic coming from all or part of ports to a dedicated port.

Contrary to conventional topologies based on a bus, many monitors have to be implemented to obtain a real picture of the communications in switched networks whatever is the solution retained for traffic monitoring Robert et al. (2015). It means that all local capture files should be sent to a common device that will merge the files in order to generate a new file that will represent the load all along the network (file that has to be transmitted to the ground via the telemetry channel). To merge all the local traces, it needs a global reference time with synchronisation offsets have to be as small as possible since the clocks in each monitor are initially running asynchronously and may produce significant offsets. The underlying question is therefore the time synchronization method that may be addressed by using a synchronisation protocol as Network Time Protocol (NTP) or IEEE1588 - Precision Time Protocol (PTP). Some work (mainly, in an operating system tracing) suggest to rely on offline synchronisation by using a post-processing algorithm. These algorithms are mainly based on regression analysis (linear, least-squares, convex hull, etc.) or linear programming. Robert et al. (2015) discuss the challenges and the limits of those methods.

The verification of properties has been already studied. QoS performance of switched Ethernet networks, and in particular, end-to-end delays have been estimated by determinist methods as noticed by Bauer et al. (2010): network calculus, real-time calculus, trajectory approach. End-to-end properties have been also estimated, especially for Integrated Modular Avionics (IMA). Lauer et al. (2011) propose hence computation of upper-bounds of end-to-end properties computed as optimal solutions of Integer Linear Programming. Results integrate here the computation time required by each function. Badache et al. (2014) propose then criteria enabling to quantify hence the quality of valid temporal allocations. These works are strongly linked to the notion of virtual links valid for AFDX networks. Furthermore, these estimations consider the knowledge of the traffic.

3. OBSERVABILITY

Assume that each message is sent over the network in multicast. The destination address corresponds to a group that gathers the original destination and the address of a PC that will be called the observer o. This station has then the availability to capture each frame that will be sent over the network.

A functional sequence is defined as a set of frames $F_1 \dots F_n$ sent by m devices ($m \leq n$). In launchers, three kind of devices can be distinguished: the sensors, the actuators and the controller. It leads that avionics requirements are usually linked to a set of exchanges between those three kind of devices. To simplify, the functional sequence $s \xrightarrow{X} c \xrightarrow{Y} a$ is now considered. It corresponds to the periodic transportation of a measure X produced by a sensor s and of an action Y computed by the (on-board) controller c for an actuator a. Fig. 1 details the exchanges, with the copy to the observer o.

We note $x(k)$ the date at which the sensor is sending the k-value of the measure and $y(l)$ the date at which the controller is sending the l-value of the command. An end-to-end freshness requirement means here that the value X, that has served to compute the value Y sent at $y(l)$, has to be received before but not δ time units before ($x(k) \geq y(l) - \delta$). Assume now that the delay of the frame sent at $x(k)$ is noted $\Delta(x(k))$. End-to-end freshness for a command $Y(l)$ will be satisfied if it can be found a measure $X(k)$ such that:

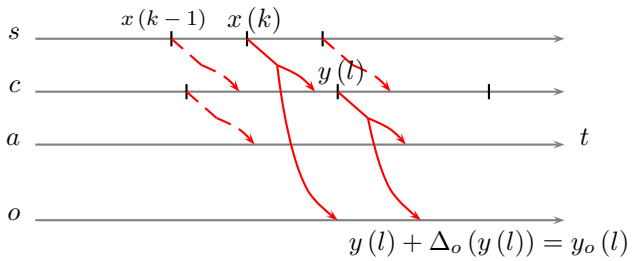


Fig. 1. Functional sequence and freshness

$$\forall l \in \mathbb{N}, \quad \exists k \in \mathbb{N} \mid y(l) - \delta \leq x(k) + \Delta(x(k)) \leq y(l) \quad (1)$$

The existence enables also to take into account potential frames losses and oversampling. The problem is that both $x(k)$ and $y(l)$ are local values, based on different clocks. From the point of view of the observer (that will receive each frame), available timestamps are slightly different. Dates are modified according the Quality of Service offered by the network, such that timestamps available are $x_o(k) = x(k) + \Delta_o(x(k))$ and $y_o(l) = y(l) + \Delta_o(y(l))$ where $\Delta_o(z)$ corresponds to the delay to send frame z to the observer. $y_o(l)$ represents the date at which the command $Y(l)$ has been captured by the observer (the timestamp in the trace). And since the path between the sensor and the controller is not the same as between the sensor and the observer, delays may be different ($\Delta(x(k)) \neq \Delta_o(x(k))$).

Since only the dates in the capture file are known, a new rule is proposed to check if the end-to-end freshness is satisfied for a functional sequence $s \xrightarrow{X} c \xrightarrow{Y} a$. It consists in verifying if the observer receives an instance k of a measure at a time compatible with the reception of a command l that will be based on this measure ($Y(l) = f(X(k))$). It is hence suggested the following proposition.

Proposition 1. The end-to-end freshness for a command $Y(l)$ ($l \in \mathbb{N}$) will be satisfied if it can be found a measure $X(k)$ such that:

$$\exists k \in \mathbb{N} \mid y_o(l) - \delta + \Delta_{max} \leq x_o(k) \leq y_o(l) - 2\Delta_{max} \quad (2)$$

where Δ_{max} corresponds to the maximum delay of any frame in the network. The proposition is valid for $\delta \geq 3\Delta_{max}$.

Proof. Starting from (1):

$$\exists k \in \mathbb{N} \mid y(l) - \delta \leq x(k) + \Delta(x(k)) \leq y(l)$$

First, by introducing $\Delta_o(y(l))$ and then $\Delta_o(x(k))$, it can be written that:

$$\begin{aligned} y(l) + \Delta_o(y(l)) &\geq x(k) + \Delta(x(k)) + \Delta_o(y(l)) \\ y_o(l) &\geq x(k) + \Delta(x(k)) + \Delta_o(y(l)) \\ y_o(l) + \Delta_o(x(k)) &\geq x_o(k) + \Delta(x(k)) + \Delta_o(y(l)) \end{aligned}$$

And hence:

$$y_o(l) + \Delta_o(x(k)) - \Delta(x(k)) - \Delta_o(y(l)) \geq x_o(k) \quad (3)$$

Secondly, on the left side of the inequality,

$$\begin{aligned} x(k) + \Delta(x(k)) + \Delta_o(y(l)) &\geq y(l) - \delta + \Delta_o(y(l)) \\ x(k) + \Delta(x(k)) + \Delta_o(y(l)) &\geq y_o(l) - \delta \\ x_o(k) + \Delta(x(k)) + \Delta_o(y(l)) &\geq y_o(l) - \delta + \Delta_o(x(k)) \end{aligned}$$

And hence:

$$x_o(k) \geq y_o(l) - \delta + \Delta_o(x(k)) - \Delta(x(k)) - \Delta_o(y(l)) \quad (4)$$

Based on (3) and (4), the end-to-end freshness property is satisfied if:

$$\begin{aligned} &y_o(l) + \Delta_o(x(k)) - \Delta(x(k)) - \Delta_o(y(l)) \geq x_o(k) \\ &\geq y_o(l) + \Delta_o(x(k)) - \Delta(x(k)) - \Delta_o(y(l)) - \delta \end{aligned} \quad (5)$$

Assuming that delays are bounded by Δ_{max}, then:

$$-2\Delta_{max} \leq \Delta_o(x(k)) - \Delta(x(k)) - \Delta_o(y(l)) \leq \Delta_{max} \quad (6)$$

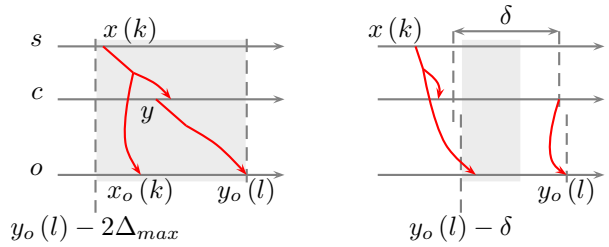
It enables to write that any $k' \in \mathbb{N}$ that verifies:

$$y_o(l) - 2\Delta_{max} \geq x_o(k') \geq y_o(l) - \delta + \Delta_{max}$$

would also verifies (5) (due to (6)). It implies also that it verifies (1).

$$(6) \implies (5) \iff y(l) \geq x(k') + \Delta(x(k')) \geq y(l) - \delta$$

It may be noticed that the proposition given in (2) is more restrictive since it rules out two time windows. It corresponds in fact to ambiguous time windows. Fig. 2 shows that it may contain measures not relevant for a given command $Y(l)$.



(a) Late measure (b) Early measure

Fig. 2. Ambiguous situations ($Y(l) \neq f(X(k))$)

Fig. 2a highlights a situation in which a frame $X(k)$ is captured before a frame $Y(l)$ whereas this command was not computed from this measure since it has been received after on the controller. In Fig. 2b, a frame $X(k)$ seems to be not earlier than δ time units before the frame $Y(k)$ but it was received before on the controller such that it was not considered to produce the command. That's the reason why time windows $[y_o(l) - 2\Delta_{max}, y_o(l)]$ and $[y_o(l) - \delta, y_o(l) - \delta + \Delta_{max}]$ are rejected (even if they may contain relevant situations). Moreover, other situations may happen outside those intervals. For instance, a measure observed just after a command might have been received before on the controller and then used to produce the command. Moreover, a measure observed before the freshness requirement might have been in fact received on the controller later, and might be hence valid. Fig. 3 summarizes the validity domains of the measures that will refresh the command.

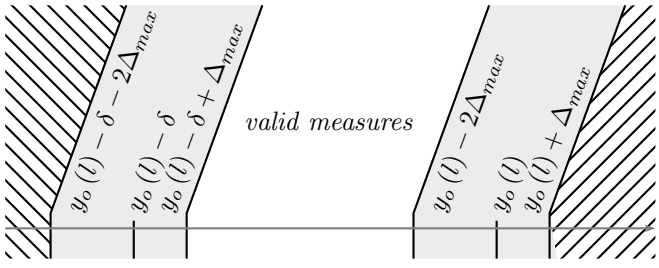


Fig. 3. Excluded time windows

Hatched areas in Fig. 3 correspond to measures out of the scope of a command $Y(l)$, while the white area contains valid (useful) measures. Gray areas correspond to domains that need to be excluded too since they are ambiguous (might contain both valid and not valid measures). Considering periodic measurements and commands, it means from there that measurement period has to be less than $\delta - 3\Delta_{max}$ (regardless clocks synchronisation).

It is important to note here that the width of the ambiguous domains is related to the maximum delay Δ_{max}. For high bandwidth networks, it means that the valid domain will be close to the interval given in (1). The accuracy may also be improved by considering separately the three delays mentioned in (6) rather than a common maximum value. By interconnecting the observer with a higher throughput link, it is also possible to decrease $\Delta_o(z)$.

This study was focussing on end-to-end freshness. It is not limited to periodic communications between a sensor, a controller and an actuator. A similar analysis may be however achieved regarding the end-to-end latency defined in Lauer et al. (2011): the time elapsed between an event at the beginning of a functional chain and the first event depending on it at the end of the chain. This paper shows furthermore that if someone wants to check the frame arrival order between two measures or commands, it is necessary as shown, in Fig. 2a, that the second value is received at the observer at least after $2\Delta_{max}$ time units (if the sending inter-arrival time is larger than Δ_{max}).

4. METHODOLOGY

4.1 Data preparation

At this point, it is assumed that a trace (captured by the on-board observer with a tool like tcpdump and sent to the ground) is available. The trace corresponds to a Wireshark pcap file as shown in Fig. 4. As explained previously, the multicast functionality is used to capture the traffic in order to be free of the synchronisation and multiple observability points. It is important to note however that it may introduced issues that need to be taken into account in the following data processing (several frames might be simultaneously forwarded on a switched network and the frame order perception might be different from one point compared to another).

In the trace, frames are timestamped (with their arrival date on the observer) and sorted chronologically. We assumed also that each frame is encoded according to a common application protocol and contains a data type (might be a measure, a command or even a telemetry information) and a sequence number fields. It means that is possible to associate for each frame i the type T_i, its sequence number K_i and its arrival date on the observer D_i. The capture file might be seen here as a matrix $\mathcal{F}$ where each row corresponds to the set $\mathcal{F}_i = \{T_i, D_i, K_i\}$ with $i \in \{1, \ldots, n\}$.

In addition, it can be retrieved from the specification of the mission of the flight a list of functional sequences for which the end-to-end freshness (and the ordering) property has to be satisfied. For a given sequence $\mathcal{S}_j$, it corresponds both to the ordered sets of the related type in the related chain

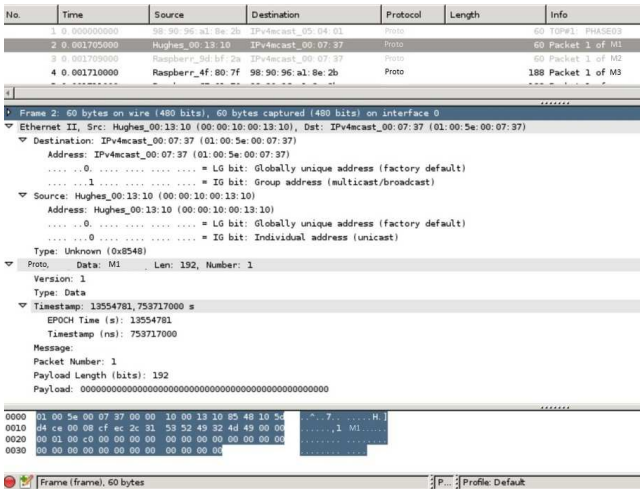


Fig. 4. Extract of a pcap Wireshark file

$\mathcal{C}_j = \{C_{j,1}, \ldots, C_{j,m}\}$ and of the freshness requirements $\mathcal{R}_j = \{\delta_{j,1}, \ldots, \delta_{j,m}\}$. $\delta_{j,m}$ means that it is required that a frame with the type $\mathcal{C}_{j,m-1}$ arrived not earlier than $\delta_{j,m}$ time-units before a frame with the type $\mathcal{C}_{j,m}$. It means that specifications might be gathered in a set $\mathcal{S}$ where each item corresponds to the set $\mathcal{S}_j = (\mathcal{C}_j|\mathcal{R}_j)$. We consider in this paper that a task only requires a single fresh data frame T_{j-1} to produce a data T_j.

4.2 Data processing

The next step, following the data selection, is data processing. The Algorithm 1 is used to parse the trace file and hence verify if the freshness is respected for all types defined in all iterations of all sequences. In fact, it provides all frames collections that successfully satisfy the sequences specifications.

Algorithm 1: Parsing trace

Data: The frames captured $\mathcal{F} = (T|D|K)$ and the sequences specifications $\mathcal{S} = (\mathcal{C}|\mathcal{R})$
Result: $\mathcal{V}$ the successful frames collections

```
1  forall S_j ∈ S do
2      F' ⟵ {F_i ∈ F | T_i ∈ C_j};
3      l ⟵ |C_j|;
4      foreach F'_i | T_i = C_{j,l} do
5          P ⟵ {F'_i};
6          forall P_z ∈ P | P_z NEq C_j do
7              k ⟵ |P_z|;
8              p ⟵ x | F'_x = P_{z,k};
9              V ← FindInputs(F', p, C_{j,l-k}, R_{j,l-k+1});
10             P ⟵ P ∪ {P_z × V};
11             P ⟵ P\P_z;
12         if P ≠ ∅ then
13             V_j ⟵ V_j ∪ P;
```

Line 2 filters the trace (according to the sequence) in order to speed up the algorithm. If a sequence refers several times to the same type, line 4 might lead to consider a frame as the final end of a sequence while it was just in the middle. In line 6, all candidates chains that do not correspond to the sequence chain specification (function

`NEq` checks that all types regardful of the ordering are contained) are taken into consideration. The core of the algorithm consists then in retrieving a tree with all sets ($\mathcal{P}$) of frames that match the sequence specifications. The roots of the trees correspond to the different iterations of the ending type of the sequence. The function `FindInputs` detailed in Algorithm 2 is used to retrieve the list of frames that are relevant as input of the one considered in the sequence.

Algorithm 2: `FindInputs`$(\mathcal{F}, p, \tau, \delta)$

Input: The trace $\mathcal{F} = (T|D|K)$, the initial position p, the searched type τ and the δ-freshness property

Output: The list $\mathcal{V}$ of matching frames

```
1  V ⟵ ∅;
2  i ⟵ p − 1;
3  while i ≥ 0 and D_i > D_p − 2Δ_max do
4      i ⟵ i − 1;        /* skip ambiguous frames */
5  while i ≥ 0 and D_i ≥ D_p − δ + Δ_max do
6      if T_i = τ then
7          V ⟵ V ∪ {F_i};
8      i ⟵ i − 1;
9  return V;
```

The validity is analyzed in Algorithm 2 according to (2) (see lines 3 and 5). Δ_{max} is a pre-processing constant that will have been computed (for instance through Network Calculus). Several frames are possible when the inter-arrival time of the input is lower than the freshness property. These frames will generate the branches of the trees in Algorithm 1. If no frames are valid, the branch will be pruned at line 11. At the end, if the condition in line 12 is false, the sequence was not here satisfied. Otherwise, Algorithm 1 gives all frames collections that may be used to print the freshness (like in the following section) or the number of inputs that satisfied any specific type. It also possible to compute the whole latency of a sequence (the difference between the arrival date of the last and the first elements of the sequence). Finally, it may be noticed that a similar algorithm might be defined to check the ordering property of the frames of a given type. Such algorithm will check that the sequence number is increasing all along the trace and that the time difference between two consecutive numbers is at least $2\Delta_{max}$ (to avoid ambiguous situations).

5. CASE STUDIES

The results presented in this section were obtained by applying the previous algorithms on a real benchmark detailed in (Robert et al., 2015, Fig. 5).

5.1 The benchmark

Two scenarios are experimented in the upper stage of a launcher when the separation of lower stage and boosters already happened. The topology (Fig. 5) is a 100 Mb/s switched Ethernet network architecture. Experiments are achieved in a nominal and a link failure cases.

The architecture is divided in two networks (line and dashed) and is composed of 17 terminal machines and

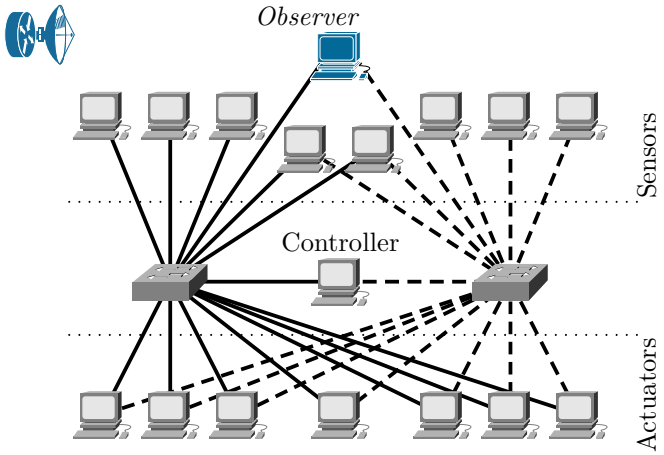


Fig. 5. Upper stage topology

2 switches. The observer is connected to both networks in order to get a unique trace gathering all traffics. The switching plane is implemented in a static way in order to not generate additional traffic on the network.

During the experiments, all traffic going through the network is scheduled and known. 66 flows are configured among the machines connected to a same network. The length of each flow is 72 bytes except for six of them that have a length of 200 bytes. The traffic load is around 2 Mb/s and the maximum delay, if all the flows are transmitted at the same time is 0.4416 ms. Traffic might be periodic, aperiodic or event-triggered.

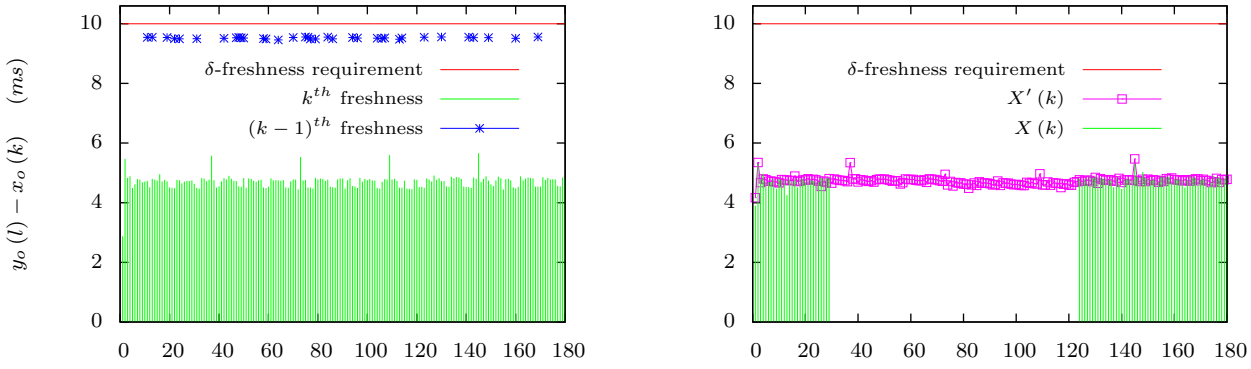
Several sequences are defined. Each frame requires the end-to-end freshness δ to be not greater than 10 ms. By consequence, the $\delta > 3 * \Delta_{max}$ hypothesis is verified and the Algorithm 1 is used to verify the freshness.

5.2 Results: nominal case

This section presents the results in the nominal case without any failures. The 10 ms-freshness property is verified for a functional sequence like in section 3, i.e. based on a command computed from a measure. The measure is periodic with a period of 5 ms, such that 1 or 2 measures ($X(k)$ and $X(k+1)$) might be used to compute a command $Y(l)$. The verification of the freshness is done by evaluating the time difference between a command $Y(l)$ and each valid measure $X(k)$. Fig. 6a shows each time difference that respects the freshness requirement.

Fig. 6a shows that at least one fresh measure is available to compute a command in the framework of (2). The variation of the freshness is due to the non synchronisation of the two flows and to the jitters. The clocks deviation between machines and observer remained however limited during the experiments, such that $X(k)$ freshness values remained similar and did not vary so much between 0 to 5 ms as it could be the case for another experiment. Sometimes, two measures are even available (when it is not the case, one of the two measure is rejected due to the non synchronisation issue or the ambiguous domains).

We observe also the ordering in a second experiment. Here the tested sequence consists of 22 aperiodic frames (of three types). Table 1 gives additional information that might be derived from the proposed algorithms. In nominal



(a) Nominal case

(b) Link failure case

Fig. 6. Freshness of each measure valid for a command. Abscissa gives the sequence number l of the command

case, no packet is lost and the inter-arrival time at the observer is very close in average to the theoretical value. Indeed, the minimum inter-arrival time is always higher than the $2\Delta_{max}$ criteria, such that frames are received according to the predefined order.

Table 1. Order results

Criteria		Nominal case	Link failure case
Packet lost ratio		0 %	29 %
Inter-arrival time	avg	0.705 s	0.340 s
	min	0.009 s	0.009 s
	max	4.010 s	1.990 s

5.3 Results: link failure case

This case is experimented by unplugging the controller of the network in solid in Fig. 5 during about 10 s (the whole experiment lasts 18 s). Table 1 shows that, in this case, losts occur and the inter-arrival time decreases.

Regarding the freshness analysis (see Fig. 6b), no measure $X(k)$ is received and hence enables to satisfy the freshness requirement during the link failure (for $l \in [30, 123]$). In the launcher case, this issue is however taken into consideration by using a redundancy in the nodes strategy. As shown by Fig. 5, a second sensor and a second network (the dashed lines) will sent the same type of data, such that the controller might use it to compute a new command. The proposed algorithms enable to verify that this redundancy will also be able to satisfy freshness requirements since a measure $X'(k)$ will be at least received in time.

6. CONCLUSION

This paper addresses an important topic for embedded systems like launchers (or even distributed network satellites): the tele-monitoring. It focus especially on how advanced properties may be verified simply based on a trace of all exchanges through the network during a flight. The analysis of the single capture is complex since the timestamps in the trace do not correspond to the exact sending or receiving time of the frame from the source to the destination. This paper proposes methods to verify the end-to-end freshness and arrival ordering requirements based on a single trace.

Future works aim at adding pre-processing of the trace into the observer (embedded in the launcher) through a Software Defined Network architecture. Here the frames capture might be substituted by OpenFlow communications.

REFERENCES

ARINC 664 P7 (2003). Part 7 – avionics full duplex switched ethernet (afdx) network. Aeronautical Radio Incorporated Specification 664, Aircraft Data Network.

Badache, N., Jaffrès-Runser, K., Scharbarg, J.L., and Fraboul, C. (2014). Managing temporal allocation in integrated modular avionics. In *Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA)*. Barcelona.

Bauer, H., Scharbarg, J.L., and Fraboul, C. (2010). Improving the worst-case delay analysis of an afdx network using an optimized trajectory approach. *IEEE Transactions on Industrial Informatics*, 6(4), 521–533. doi: 10.1109/TII.2010.2055877.

Department of Defense (1978). Military standard aircraft internal time division command/response multiplex data bus. MIL-STD 1553B.

Lauer, M., Ermont, J., Boniol, F., and Pagetti, C. (2011). Latency and freshness analysis on ima systems. In *Emerging Technologies & Factory Automation (ETFA), 2011 IEEE 16th Conference on*. Toulouse, France.

Monchaux, D., Gast, P., and Sangare, J. (2012). Avionic-x: A demonstrator for the next generation launcher avionics. *Embedded Real-Time Software and Systems (ERTS 2012)*.

Notebaert, O. (2014). Future on-board networks in space systems. In *10th IEEE International Workshop on Factory Communication Systems*. Toulouse, France.

Robert, J., Georges, J.P., and Divoux, T. (2015). On the observability in switched ethernet networks in the next generation of space launchers: Problem, challenges and recommendations. In *The Seventh International Conference on Advances in Satellite and Space Communications (SPACOMM)*.

Robert, J., Georges, J.P., Divoux, T., et al. (2013). Toward self-reconfiguration of switched ethernet architectures in the next generation of space launchers. In *The Fifth International Conference on Advances in Satellite and Space Communications (SPACOMM)*.

SAE AS 6802 (2011). Time-triggered ethernet. AS-2D2 Deterministic Ethernet and Unified Networking.