Worst-Case Delay Analysis of Master-Slave Switched Ethernet Networks

M. Ashjaei[1], M. Liu[1], M. Behnam[1], A. Mifdaoui[2], L. Almeida[3], T. Nolte[1]
[1] Mälardalen Real-Time Research Centre (MRTC), Mälardalen University, Västerås, Sweden
[2] University of Toulouse, France
[3] IT/DEEC, University of Porto, Portugal

ABSTRACT

Switched Ethernet is increasingly used in real-time communication due to its intrinsic features such as micro segmentation and high throughput. However, COTS switches may impose long blocking times due to their FIFO queues and can also experience buffer overflow in outgoing queues due to uncontrolled packets arrival. The FTT-SE protocol uses a Master-Slave technique to overcome the COTS switch limitations in real-time applications. Recently, we extended the protocol for large scale networks and in this paper we present the worst-case delay analysis using the Network Calculus formalism for such a network. Moreover, we assess the end-to-end delay of traffic with simulation concluding that the obtained analytical results present a good match with the observed delays, providing uppers bounds that vary between 0% and 50% above the maximum measured values.

1. INTRODUCTION

Recently, there has been an increased interest on using Switched Ethernet technology in networked embedded systems as it provides high throughput, low cost, collision free domain, traffic isolation, wide availability and is generally a mature technology. However, using Commercial Off-The-Shelf (COTS) switches in time critical applications might not be straightforward due to the following reasons. First, the traditional use of FIFO queues in the switch ports might generate long blocking times to urgent real-time traffic. Second, the uncontrolled arrival of packets might overflow the switch buffers and lead to packet drops which is not acceptable for real-time applications. Third, most of COTS switches have a limited and low number of priorities to schedule traffic inside switches, providing very few options for real-time traffic scheduling.

Therefore, we propose using the FTT-SE (Flexible Time-Triggered Switched Ethernet) protocol which uses a Master-Slave traffic control technique [9]. A centralized master node coordinates the entire load submitted to the switch in order to avoiding the problem of buffer overflow inside the switch. In addition, the FTT-SE protocol handles all types of message streams including real-time periodic, real-time sporadic and non-real-time traffic, by defining and using specific reserved bandwidth for each type of message streams and thus providing temporal isolation between them. Moreover, by controlling the traffic submitted to the network from a central point, the protocol effectively allows implementing any desired traffic scheduling policy, overriding the queuing policies used by the switch.

However, the FTT-SE protocol was originally developed for small networks consisting of a single switch and a set of nodes [9], thus the scalability of the protocol has not yet been fully explored. Recently, two architectures were proposed to extend the FTT-SE protocol using multiple switches that are connected together to form a tree topology. The first approach uses a single master node, that is connected to the tree root, to control the traffic transmission [10]. The second approach uses multiple master nodes to coordinate transmission of the traffic, such that each master is attached to a single switch [2]. However, the latter architecture is not fully investigated and it lacks the traffic delay bound analysis.

In this paper, we address the second architecture and we present the worst-case delay analysis using Network Calculus formalism for the mentioned architecture. Moreover, we present a simulation of data transmission on a network example using a tool presented in [1]. In addition, we compare the results of the simulator with the results of applying the proposed analysis on the same case study to validate the correctness of the analysis and also to evaluate the level of pessimism in the analysis.

The rest of the paper is structured in the following way. The next section discusses related work. Section 3 illustrates the system model. Then, Section 4 describes the FTT-SE protocol for a multi-master architecture while Section 5 presents the worst-case delay analysis based on Network Calculus. Section 6 shows delay analysis of a network example using both worst-case delay analysis and simulation, and finally Section 7 concludes the paper and presents future work.

2. RELATED WORK

There is a large amount of work in the literature concerning support for real-time traffic over Ethernet. Here we will refer to some paradigmatic research approaches that are more related to our work, particularly concerning the applicability to multi-switch networks.

The EtheReal protocol [15] used Ethernet switches which

.

are modified to support specific traffic management and scheduling services. This protocol is connection-oriented in which the nodes should follow a connection setup protocol that reserves the needed bandwidth for sending messages to the real-time channel. The EDF Scheduled Switch [4] shortly followed the previous one and it was based on adding a specific real-time layer to both switches and end nodes, which was responsible for establishing real-time channels, carrying out admission control, time synchronization and message transmission control in the network. Despite the good timeliness of these protocols, the use of modified switches reduced their impact.

Conversely, the FTT-SE protocol was devised to provide real-time communication services on top of COTS switches. The work in [12] proposed using Network Calculus to analyze FTT-SE in a multi-switch topology. For each node in the network, a specific bandwidth is reserved using traffic shapers to guarantee that their traffic would always fit inside one cycle at a time, independently of the traffic scheduling in the master. However, reserving bandwidth for each node in this way was not very efficient, as shown in [10]. This latter work presented a worst-case response-time analysis for FTT-SE networks with multiple switches connected in a tree topology in which the master node was connected to the root switch, coordinating all traffic transmissions in the network.

A worst-case response time analysis for multi-hop switched Ethernet was presented in [16], considering existing Ethernet hardware without any modifications. Upper bounds on delays were computed using Network Calculus, too. Moreover, the results showed that hard real-time communication requires coordination of all the streams at a global level, which led to introducing a dual-level traffic smoothing mechanism.

Moreover, Avionics Full Duplex Switched Ethernet, known as AFDX, is another kind of Ethernet technology tuned to real-time systems. The work in [3] presents a method to compute end-to-end bounds using the "pay burst only once" principle in Network Calculus along with a model that takes the shaping introduced by the medium into account. Furthermore, usage of Network Calculus for other Ethernet protocols such as Ethernet AVB for automotive networks is presented in [13]. In the latter work, a case study of distributed infotainment devices was investigated based on the proposed approach for worst-case delay analysis. The result shows that Network Calculus is suited for timing calculation in automotive Ethernet networks.

In addition, the work in [8] presents a worst-case validation of an in-vehicle Ethernet network based on a case study. The worst-case scenario was investigated using a Network Calculus model and the result showed that all considered functions met their requirements even in the worst-case scenario.

The work in [7] and [14] present a methodology based on Network Calculus to compute an end-to-end delay bound for a single flow in FIFO multiplexing sink-tree networks. In this type of network the topology is partitioned into a set of logically separated sink-trees having egress node at root and ingress nodes at leaves. The traffic is aggregated in nodes according to a FIFO policy named aggregated scheduling. In order to obtain a tight delay bound, an extended Network Calculus is presented in which a class of service curves is introduced to describe the service that is received in an aggregate scheduling network. Moreover, the mentioned methodology is utilized in [6] in order to investigate an admission control in sink-tree networks. Neither of these uses of Network Calculus can be directly applied to the multi-switch FTT-SE network due to the specific traffic control mechanisms of this protocol. This adaptation is the focus of this work.

3. SYSTEM MODEL

In this paper, we consider the FTT-SE multi-master architecture relying on a network of switches connected in a tree topology. We define a *sub-network* as the set of a switch, the master and slave nodes directly connected to it, e.g., M1, SW1, S1 and S2 in Figure 1. The switch in the top of the tree hierarchy is called the *root switch*. Each sub-network is a parent for the sub-networks in the lower level attached to it. Moreover, all sub-networks having the same parent sub-network define a *cluster*, e.g., cluster2 in Figure 1. The only exception is the root sub-network, which is included in its children cluster as it cannot be considered a separate cluster itself. For the remaining clusters, we define the *cluster master* as the master of the parent sub-network.

The switches are assumed to be Commercial Off-The-Shelf (COTS) possibly having several parallel FIFO queues with different priority levels for each output port, no queues in the input ports, and are store-and-forward.

In this paper, we consider a message m_i as an infinite recurring arrival of unicast network packets characterized by the following tuplet:

$$m_i = m_i(L_i, D_i, T_i, O_i, S_i, Ds_i, SP_i) \tag{1}$$

In this tuplet, L_i is the message length in bits, D_i and T_i are the relative deadline and period of the message respectively and $D_i \leq T_i$. Also, S_i is the source node and Ds_i is the destination node. Moreover, SP_i is the set of switches in the route of the message and O_i is the offset of message. The FTT-SE protocol supports both synchronous and asynchronous messages. Here we model both with the same tuplet above. However, for asynchronous messages, T_i is the minimum inter-arrival time of the message and $O_i = 0$. The traffic scheduling is based on fixed priorities and these can be implicitly set according to Rate-Monotonic, Deadline-Monotonic or any other criteria reflected in the indexes.

Furthermore, we classify messages, according to the connections of their source and destination nodes, into two categories; local and global messages. If the sender and receiver of a message belong to the same sub-network, the message is called *local*. Otherwise, the message is called *global* i.e., the sender and receiver of the message are connected to different switches.

4. FTT-SE PROTOCOL

The FTT-SE protocol is an Ethernet real-time communication protocol that uses a particular node called master to coordinate all traffic in the network in a flexible and timely manner [9]. This protocol, which supports both synchronous and asynchronous traffic, was developed for small networks containing a single switch. One of the solutions for scaling the protocol to large networks is to utilize a master node for each switch while the switches are connected together in a hierarchical tree topology (Figure 1).

The protocol organizes the network traffic in fixed duration time slots named Elementary Cycles (EC). The EC is functionally partitioned into two phases, an initialization phase for protocol management purposes, and a following part for the transmission of the actual data messages which we call data transmission window. In the multi-master FTT-SE architecture, the data transmission window is divided into two sub-windows for handling periodic messages, called the Synchronous Window (SWin), and for handling aperiodic messages, called the Asynchronous Window (ASWin). These two windows are further divided among different types of traffic including local and global as depicted in Figure 2. Moreover, the window for global asynchronous traffic is partitioned among the clusters (in Figure 2 it is divided into two sub-windows, for cluster1 and cluster2).

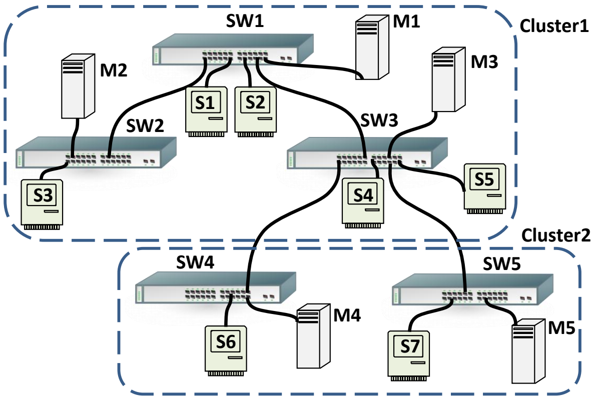

Figure 1: Multi-Master FTT-SE Network

Each master node is aware of the timing properties of the local traffic for its sub-network and all global synchronous traffic in the entire network. The global synchronous traffic is scheduled in all master nodes in parallel to ensure that it can cross all the switches in its path each EC. It is the master node that decides online which local and global messages should be transmitted in each EC. The traffic scheduling policy can be any, such as Fixed Priority Scheduling. The IDs of the scheduled messages are encoded in a particular message called the Trigger Message (TM) that the master broadcasts to the slave nodes at the beginning of the EC, e.g., TM3 and TM4 in Figure 2.

Unlike the synchronous traffic that is activated periodically by the master, activation of asynchronous traffic is unknown and can be at any time during the EC. In order to handle the asynchronous traffic, a signaling mechanism [11] allows the slave nodes to notify the master node of any pending requests using a particular message called Signaling Message (SIG) transmitted once per EC. However, in the multi-master architecture two different SIGs are used, one for local and the other for global asynchronous traffic. The SIG for local asynchronous messages is transmitted to the sub-network master approximately at the same time as the master sends the TM in the beginning of the EC, e.g., message A in Figure 2. Upon reception of a SIG message, the respective asynchronous transmission requests are placed in the traffic ready queue and then scheduled by the master. Conversely, a SIG for global asynchronous messages is generated by the source slave node and transmitted exactly after the local SIG to the master node of the cluster, e.g., Message B in Figure 2 is sent to M3. The master of each cluster is responsible for scheduling the global asynchronous messages of the cluster within the respective window. Then, a particular TM called *asynchTM* is used to inform the slave nodes about the transmission of the global asynchronous messages, which is sent after the TM, e.g., asynchTM3 in Figure 2.

All slave nodes, every EC, wait for the reception of the TM of their sub-network and then for the reception of the asynchTM of their cluster. After receiving the asynchTM, the slave nodes need a specific time to decode both TM and asynchTM that varies based on the processing time of the nodes, which is called *turn around time* (TRD). Afterward, the slave nodes send the traffic scheduled for that EC, which IDs were encoded in the trigger messages.

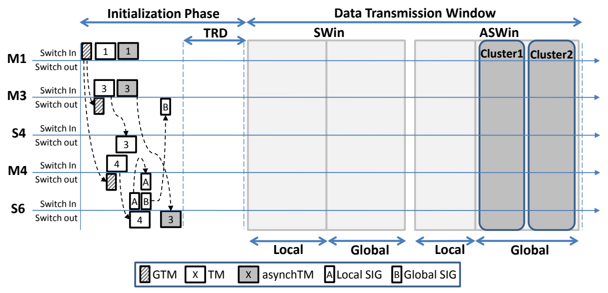

Figure 2: Multi-Master FTT-SE Elementary Cycle

The main aim of the master node is to schedule traffic without causing overrun in the EC, i.e. the scheduled messages must be received before the end of the EC. The scheduler in each master node inserts the ready messages into four different ready queues depending on the message types, i.e. local/global and synchronous/asynchronous. The master node picks the messages from the heads of the queues and checks whether they fit in the dedicated window. The messages that fit in the current EC are encoded in the TM for transmission, while remaining messages are kept in the ready queues for the upcoming ECs.

In order to check the messages, the scheduler in each master node keeps track of message transmission times in each link using bins with limited capacity representing the dedicated window for that type of message. Each switch port is represented by two bins: one for each direction, i.e. output and input ports as shown in Figure 3.

Returning to the network architecture illustrated in Figure 1, assume that a synchronous message m_1 is ready to be transmitted from S7 to S5. Therefore, m_1 is categorized as a global message since it is transmitted beyond its sub-network. As a global message, all schedulers in all master nodes add m_1 in In5, Out5 (= In3) and Out3 bins shown in Figure 3. If the message fits in all bins associated to the links, the ID of m_1 is encoded into TM5 and broadcast within its sub-network. The slave S7 will then receive the TM5 and send m_1. Note that, when the scheduler in all master nodes add m_1 to the bins associated to output ports of switches (Out5 and Out3), they take into account the delays imposed by the switch and other traffic. These delays are key in the end-to-end analysis of traffic.

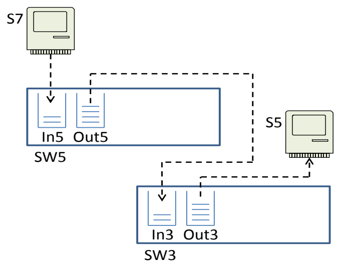

Figure 3: Master Scheduling using Bins

Finally, for the scheduling of global traffic to be consistent, all master nodes must be synchronized. This can be achieved in many ways. In this paper we use a particular message named Global Trigger Message (GTM) that is transmitted from the root master to the entire network at the beginning of each EC. The master nodes wait to receive the GTM in order to trigger the ECs in their sub-networks as shown in Figure 2.

5. WORST-CASE DELAY ANALYSIS

In this section, we present the end-to-end delay analysis for the traffic in the multi-master architecture using the Network Calculus formalism.

5.1 Network Calculus Principal

Network Calculus [5] is a framework for analyzing deterministic queuing systems based on *min-plus* algebra. The traffic transmitted to a network is affected by constraints imposed by the network components such as link bandwidth and traffic shapers. The constraints associated to the traffic flow are expressed by an *arrival curve* $\alpha(t)$, whereas the availability of resources such as crossed nodes is described by a *service curve* $\beta(t)$. The delay bound which represents upper bound to the worst-case response time and the backlog that represents an upper bound to the maximum queue length are computed using the arrival and service curves.

Basically, the delay bound Db is the maximum horizontal length between $\alpha(t)$ and $\beta(t)$, while the maximum vertical distance between them is the backlog bound B. In a simplified case the arrival curve can be upper bound by $\alpha(t) = b + rt$ such that b is the maximum burst and r is the rate of traffic. Moreover, the service curve for a node is $\beta(t) = max(0, R(t - \Delta))$, where R is the rate of service and Δ is the latency imposed by that node.

In addition, using min-plus convolution $\otimes$ the service curve for concatenation of two nodes is computed as shown in (2).

$$\beta(t) = (\beta_1 \otimes \beta_2)(t) = inf_{0 \le s \le t}(\beta_1(s) + \beta_2(t-s)) \tag{2}$$

Moreover, in case of a non-preemptive priority based network mode the service curve for medium priority traffic $\beta_M(t)$ is affected by all higher priority arrival curves and blocked by lower priority traffic [5] which is shown in (3), where $\alpha_H(t)$ is the arrival curve for the higher-priority traffic and L^L_{max} is the maximum length of low priority traffic.

$$\beta_M(t) = \beta(t) - \alpha_H(t) - L^L_{max} \tag{3}$$

In our multi-master FTT-SE protocol, a message incurs in several delays from the time it becomes ready until it is received by the destination node. In the remainder of this section we analyze all such delays to compute the maximum end-to-end delay bound using the Network Calculus framework.

5.2 Transmission Delay

In the FTT-SE protocol, the traffic is transmitted periodically for synchronous messages and with minimum interarrival time for asynchronous traffic. Thus, the arrival curve for m_i is shown in (4), where $\frac{L_i}{T_i}$ is the rate of the arrival curve considering the period T_i of the message.

$$\alpha_i(t) = L_i + \frac{L_i}{T_i}t \tag{4}$$

Moreover, according to the protocol, each type of traffic has a dedicated window. Therefore, the service curve for each switch (SW_k) is shown in (5), where the rate of the service is a factor of the network capacity C. The rate of the service curve is expressed as $\frac{BW-I}{EC}$, where BW is the duration time of the dedicated window for the message, e.g., Local Synchronous Window duration time and I is the idle time that is used to prevent the overrun problem in the mentioned window, which is the size of longest packet among all messages in similar type as m_i. The latency Δ in each service represents the switch fabric latency ϵ and store-and-forward delay SFD (in case of using store-and-forward switch type). In this analysis, we consider the maximum message length L_{max} that generates the maximum SFD delay in the switch. Therefore, $\Delta = \epsilon + \frac{L_{max}}{C}$.

$$\beta_{SW_k}(t) = max(0, \frac{BW - I}{EC}C(t - \Delta)) \tag{5}$$

Based on Network Calculus, having two sequential network components with two service curves $\beta_1(t)$ and $\beta_2(t)$, we can group them as a single global component with a service curve $(\beta_1 \otimes \beta_2)(t)$. This new service curve for the global component gives a better bound for end-to-end computation due to considering a burst in one component instead of both at the same time, i.e., the so called pay burst only once phenomena [5]. Moreover, in the FTT-SE protocol the scheduler in the master node considers all switch queuing delays when checking whether the message fit in the dedicated window within EC. Consequently, there is no buffered traffic at the end of each EC since the master only schedules the traffic that can be completely transmitted by then. In the multi-master architecture the source node for m_i, under analysis, and all switches that the message crosses, are grouped as one global node with a service curve computed using min-plus convolution as shown in (2). The service curve for a global node that is offered for m_i is presented in (6), that has a rate equal to the minimum among the source node and all switches in SP_i and the global node latency is the summation of all latencies.

$$\beta_{global_i}(t) = \frac{BW - I}{EC}C \times (t - \sum \Delta_k), \quad \forall k \in \{SP_i\} \tag{6}$$

In the multi-master FTT-SE architecture, a message may suffer from three different types of interference known as source node delay, direct and indirect interference.

The source node delay is caused by all higher-priority messages in the same source node with the message under analysis. The set of messages that generate this delay, $src(m_i)$ is shown in (7), where $ip_{i,1}$ is the input port of the switch which node S_i is directly connected and $hp(m_i)$ is the set of messages with priority higher than that of m_i. Moreover, all interfering messages should be of the same type as m_i which is shown with $Typ(m_i)$, i.e., synchronous or asynchronous, global or local.

$$src(m_i) = \{\forall_{m_j} : m_j \in ip_{i,1} \wedge m_j \in hp(m_i) \wedge m_j \in Typ(m_i)\} \tag{7}$$

The direct interference appears due to the messages that share an output port with m_i and have priority higher than that of m_i. All messages that were considered in source node interference should be excluded from direct interference set. This set of messages $direct(m_i, k)$ is derived in (8) for switch k, where $op_{j,k}$ is the output port of switch k, $hp(m_i)$ is the higher-priority set and $Typ(m_i)$ denotes that the interfering messages should be of the same type as m_i, i.e. local/global synchronous or asynchronous. Note that, the messages that were considered in previous switches along the path should be excluded from the messages that have interfered at SW_k.

$$\begin{aligned} direct(m_i, k) = \{\forall_{m_j} : m_j \in op_{j,k} \ \wedge \ m_j \in hp(m_i) \ \wedge \\ m_j \in Typ(m_i) \ \wedge \ m_j \notin src(m_i) \ \wedge \\ m_j \notin direct(m_i, l) | SW_l \in SP_i\} \end{aligned} \tag{8}$$

Finally the effect studied in [10] and known as *indirect effect* further adds to the interference. To show this effect, consider the following example, referring to the topology in Figure 1.

We define four global messages in the example network where the message m_{14} is sent from node S1 to node S4, the message m_{64} is transmitted from node S6 to node S4, the message m_{25} is transmitted from node S2 to node S5, and the message from node S7 to Node S5 is denoted as m_{75}. We assume that m_{14} has the lowest priority among all messages. Also, m_{64} needs one EC to be transmitted and $m_{25} + m_{75}$ require one EC to be transmitted as well.

Let us consider that m_{14} and m_{64} are activated simultaneously, while m_{25} and m_{75} are activated one EC later. In the first EC m_{64} is scheduled and m_{14} pends for the next EC due to having lower priority. Afterwards, in the second EC two other messages are activated which both are scheduled to be transmitted in the second EC. Finally, m_{14} is scheduled in the third EC. However, if m_{75} was not activated then it would be possible to schedule m_{14} together with m_{25} as the length of m_{14} is assumed to be equal or less than m_{75}.

From the scenario outlined above, we can conclude that two messages may interfere with each other even if they do not share links directly. In the above example the response time of m_{14} is affected by m_{75}, although they do not share links with each other.

To add this effect into the response time, not only all messages that share links with the message under analysis should be taken into account, but also all the messages that share links with the message that delay the message under analysis should be taken into account. Thus, the set of messages that generate the indirect effect on the switch k, $indir(m_i, k)$, is derived in (9). Note that, all messages that were considered in direct interference should be excluded from the indirect interference message set.

$$\begin{aligned} indir(m_i, k) = \{\forall m_k \in direct(m_j, k) : m_j \in direct(m_i, k) \\ \wedge \ m_k \notin direct(m_i, k) \ \wedge \ m_k \notin src(m_i)\} \end{aligned} \tag{9}$$

Consequently, all the arrival curves of interfering messages affect on the global service curve. Therefore, considering the arrival curves of the mentioned interference messages as $\alpha_H(t)$ in (3) and given explicit arrival curves, the service curve available for m_i is shown in (10). Note that, in order to apply the indirect interference, we consider a pessimistic assumption, which simplified the computation of worst-case delay and allows to include indirect effect in the same way as the direct interference into the global service curve. Also, in the FTT-SE protocol the higher-priority message is not blocked by lower-priority due to the scheduler that takes into account the EC finishing time during scheduling of the messages.

$$\beta_i(t) = Rate_i \times t - V_i \tag{10}$$

$$\begin{aligned} Rate_i = \frac{(BW - I)C}{EC} - \sum_{m_j \in src(m_i)} \frac{L_j}{T_j} - \\ \sum_{m_p \in direct(m_i)} \frac{L_p}{T_p} - \sum_{m_q \in indir(m_i)} \frac{L_q}{T_q} \end{aligned} \tag{11}$$

$$\begin{aligned} V_i = \frac{(BW - I)C}{EC} \times \sum_{k \in \{SP_i\}} \Delta_k + \sum_{m_j \in src(m_i)} L_j + \\ \sum_{m_p \in direct(m_i)} L_p + \sum_{m_q \in indir(m_i)} L_q \end{aligned} \tag{12}$$

Finally, the maximum end-to-end delay for m_i is the maximum horizontal distance between $\alpha_i(t)$ and $\beta_i(t)$ which is computed in (13). Moreover, store-and-forward delay caused by switches that m_i crosses, is affected the bandwidth availabel. In order to consider this effect, we inflate the length of the message by the number of SFD delay for the message. Therefore, $L_i' = L_i + n(SFD + \epsilon)$, where n is the number of switches in the route of m_i.

$$Db_i = \frac{L_i' + V_i}{Rate_i} \tag{13}$$

6. EVALUATION

In this section we validate the proposed analysis, and assess the level of pessimism embodied in our analysis compared with the simulation results of the proposed online scheduling algorithm in one particular example. We consider a network consisting of five switches as shown in Figure 4.

The network parameters are $EC = 1.5ms$, $TM = 24\mu s$, $SIG = 6\mu s$, $SLD = 17\mu s$ and the transmission speed of the Ethernet network is considered as $100 Mbps$. Moreover, the data transmission window during each EC is divided as follows. The synchronous and asynchronous local scheduling windows are selected to have both times equal to $300\mu s$. Also, synchronous and asynchronous global scheduling windows are selected $400\mu s$ for each. In the example, the network is composed of two clusters and the window of the asynchronous global scheduling window is further divided equally among them, i.e. $200\mu s$.

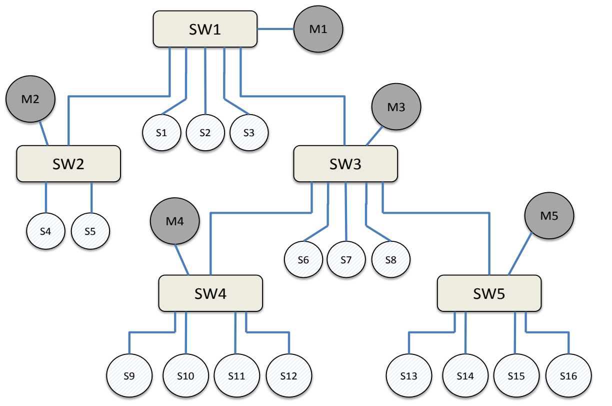

Figure 4: A Network Example

In this example, 30 messages including all four types of traffic are defined such that one particular message per each type is selected to have the worst-case scenario among all other messages. The worst-case scenario for both global synchronous and asynchronous messages are defined to have the lowest priority among all other messages with the longest rout, e.g. from SW5 to SW2. Moreover, while crossing each link several messages are defined with higher priority to interfere with the mentioned messages. In order to determine the worst-case scenario for local synchronous and asynchronous messages, the lowest priority is assigned and several messages are defined to interfere in the source and destination links. This does not guarantee a worst-case situation, which is rather complex to determine with accuracy, but it is certainly a very unfavorable situation with strong interference. The Fixed Priority Scheduling Policy is assumed in this study and the priority of messages is selected according to the Rate Monotonic priority assignment.

In addition, the minimum inter-arrival of all asynchronous messages are taken into account in simulation to have the worst-case activation of asynchronous messages. Furthermore, during the simulation we tagged those particular messages defined as *worst-case* and we measured their maximum end-to-end delays. We then compared the measured end-to-end delays obtained from simulation with those computed with the proposed delay analysis.

The tagged messages properties are presented in Table 1. For each message, T is the period of the synchronous messages and minimum inter-arrival time of the asynchronous messages presented in number of ECs. Also, DPL is the data payload of the message in bytes.

Message id	Message Type	T(EC)	DPL
m1	Local Synchronous	20	958
m2	Global Synchronous	19	583
m3	Local Asynchronous	20	833
m4	Global Asynchronous	18	583

Table 1: Tagged Messages Properties

Figure 5 presents the maximum response time of the tagged messages measured from simulation (for 500 ECs simulation time) and the response time computed using the proposed analysis. Note that in the figure, the x-axis represents a message id and the y-axis shows the response-time in number of ECs.

Comparing the results of the simulation and the proposed analysis, the results from the worst-case delay analysis using Network Calculus is always equal or greater than the results of the simulation. In our experiments, the result computed for local synchronous message equals to the result observed in simulation. However, for other types of messages we obtained different levels of pessimism, varying from 20% and 50%. We believe, as referred before, that the larger difference is still due to a less than worst-case simulated situation.

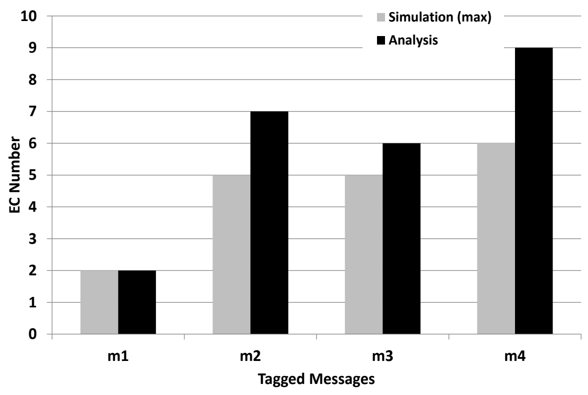

Figure 5: Tagged Messages Response Time

7. CONCLUSION AND FUTURE WORK

In this paper, we analyzed one architecture that has been proposed to extend the FTT-SE protocol to large networks. The architecture consists of multiple switches along with multiple master nodes. The switches are connected together in a hierarchical tree topology. In particular, this paper has presented a worst-case delay analysis using Network Calculus for different types of traffic including global and local synchronous and asynchronous messages. Finally, we compared the results of simulation and the analysis. The analysis was validated with simulation and showed the ability to find the actual worst-case response time for some local sycnhronous messages. On the other hand, it revealed a potential level of pessimism of up to 50% for global asynchronous messages, which are those that involve the highest overhead and protocol operations but also for which the actual worst-case scenario is harder to obtain in simulation. Future work aims at tightening the end-to-end analysis by optimizing the source of pessimism, specially related to the direct and indirect interference, and to better identify traffic patterns that generate worst-case situations. Moreover, another direction of future work is investigating solutions for time synchronization between the master nodes in the network such that it generates less effect on the protocol performance.

8. ACKNOWLEDGMENTS

This work is supported by the Swedish Foundation for Strategic Research, via Mälardalen Real-time Research Center (M-RTC) at Mälardalen University. Also, it is partially supported by FEDER through the COMPETE program, and by the Portuguese Government through FCT grants SENODS CMU-PT/SAI/0045/2009 and Serv-CPS "PTDC/EEA-AU T/122362/2010".

9. REFERENCES

[1] M. Ashjaei, M. Behnam, and T. Nolte. The design and implementation of a simulator for switched ethernet networks. In *3rd International Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems (WATERS'12)*, July 2012.

[2] M. Ashjaei, M. Behnam, T. Nolte, L. Almeida, and R. Marau. A compact approach to clustered master-slave ethernet networks. *9th IEEE International Workshop on Factory Communication Systems (WFCS'12)*, May 2012.

[3] M. Boyer and C. Fraboul. Tightening end to end delay upper bound for afdx network calculus with rate latency fifo servers using network calculus. In *7th IEEE International Workshop on Factory Communication Systems (WFCS'08)*, May 2008.

[4] H. Hoang and M. Jonsson. Switched real-time ethernet in industrial applications - deadline partitioning. In *9th Asia-Pacific Conference on Communications (APCC'03)*, September 2003.

[5] J. Leboudec and P. Thiran. *Network Calculus.* Berlin, Germany: Spriger-Verlag, 2001.

[6] L. Lenzini, L. Martorini, E. Mingozzi, and G. Stea. A novel approach to scalable cac for real-time traffic in sink-tree networks with aggregate scheduling. In *the 1st international conference on Performance evaluation methodolgies and tools.* ACM, October 2006.

[7] L. Lenzini, L. Martorini, E. Mingozzi, and G. Stea. Tight end-to-end per-flow delay bounds in fifo multiplexing sink-tree networks. *Elsevier Performance Evaluation*, 63, October 2006.

[8] M. Manderscheid and F. Langer. Network calculus for the validation of automotive ethernet in-vehicle network configurations. In *International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC'11)*, October 2011.

[9] R. Marau, L. Almeida, and P. Pedreiras. Enhancing real-time communication over cots ethernet switches. In *6th IEEE International Workshop on Factory Communication Systems (WFCS'06)*, June 2006.

[10] R. Marau, M. Behnam, Z. Iqbal, P. Silva, L. Almeida, and P. Portugal. Controlling multi-switch networks for prompt reconfiguration. In *9th International Workshop on Factory Communication Systems (WFCS'12)*, May 2012.

[11] R. Marau, P. Pedreiras, and L. Almeida. Asynchronous traffic signaling over master-slave switched ethernet protocols. In *6th International Workshop on Real Time Networks (RTN'07)*, July 2007.

[12] A. Mifdaoui, F. Frances, and C. Fraboul. Performance analysis of a master/slave switched ethernet for military embedded applications. *IEEE Transactions on Industrial Informatics*, November 2010.

[13] R. Queck. Analysis of ethernet avb for automotive networks using network calculus. In *IEEE International Conference on Vehicular Electronics and Safety (ICVES'12)*, July 2012.

[14] J. Schmitt, F. Zdarsky, and M. Fidler. Delay bounds under arbitrary multiplexing: When network calculus leaves you in the lurch... In *The 27th IEEE Conference on Computer Communications*, April 2008.

[15] S. Varadarajan and T. Chiueh. Ethereal: a host-transparent real-time fast ethernet switch. In *6th International Conference on Network Protocols*, October 1998.

[16] M. Zhang, J. Shi, T. Zhang, and Y. Hu. Hard real-time communication over multi-hop switched ethernet. In *IEEE International Conference on Networking, Architecture, and Storage (NAS'08)*, June 2008.