Network Fault Tolerance by Means of Diverse Physical Layers

Pablo Gutiérrez Peón*†, Wilfried Steiner*, and Elisabeth Uhlemann†
*TTTech Computertechnik AG, Vienna, Austria
†School of Innovation, Design and Engineering, Mälardalen University, Västerås, Sweden
Email: {pablo.gutierrez-peon, wilfried.steiner}@tttech.com, elisabeth.uhlemann@mdh.se

***Abstract*—Wired networks are deployed in scenarios requiring the highest levels of performance in terms of reliability and timeliness. Unfortunately, broken wires might permanently compromise the network operation unless fault tolerance mechanisms are in place. Fault tolerance is commonly achieved by replicating the wired network components, but this paper examines the use of a wireless backup network, since the wireless physical layer (PHY) is not expected to display permanent failures due to broken wires. Two mechanisms at medium access control (MAC) level are presented to take advantage of the wireless backup network: one allocating redundancy statically and one dynamically. Without loss of generality, redundancy is applied using the standard mechanisms from IEEE 802.3 (Ethernet) and IEEE 802.11 (Wi-Fi). The performance increase added by the backup network is studied both analytically and by simulation, showing considerable improvements in a very compromised mid-size wired network.**

I. Introduction

Many computing systems, especially embedded, are deployed in dynamic environments to which they need to react within some time boundaries. These systems frequently require to exchange data via networks that should also adhere to timing requirements. Having excessive delay and low reliability as the main factors compromising real-time behaviour, the usual choice is to deploy wired networks over wireless, as the latter are often more prone to fading, shadowing and interference. These problems are not absent in wired networks, but it is mostly assumed they are compromised by reliability problems to a significantly lower degree. With reliability as the main concern, diversity techniques can be applied in wireless networks to greatly increase this performance factor [1], making their adoption more common nowadays for real-time use cases. To a great extent, this is explained by the set of relevant features wireless networks come along with, including mobility, reduced wiring costs and flexibility of deployment.

In highly critical systems, fault tolerance is achieved by redundancy, as the only way to reach for the highest levels of reliability when single components can barely be improved further. A straightforward example would be a broken cable, destined to cause a permanent failure unless an alternative route is provided. Consider the case of industrial sabotage for example. Even though power backup systems are usually in place for critical infrastructure like power plants, etc., cutting wires used for communications could cause severe damage. Certainly, the addition of new components to overcome the loss of connection comes at a price. Being aware of this reliability problem, this paper comes aside the usual way of adding redundancy in wired networks based on replicating the wired network infrastructure. The paper explores the use of a wireless backup network to increase reliability, targeting potential savings in space, weight and money. This approach can be applied in situations in which wired links are especially compromised, like the example with industrial sabotage.

Two schemes at the medium access control (MAC) layer are proposed to take advantage of this wireless backup network: one statically allocating resources to the transmission of highly critical traffic, and the other based on reacting to broken wired links and trying to build an alternative path dynamically using the wireless counterparts. Without loss of generality, the schemes are proposed to work with the high throughput IEEE 802.3 (Ethernet) and IEEE 802.11 (Wi-Fi), the de facto local wired and wireless network standards respectively, defining the physical (PHY) and MAC layers. Yet, the backup mechanisms are in principle applicable using other wireless network technologies like 5G. Ethernet and Wi-Fi bring together the advantages of cheap and all-pervading technologies and the real-time and fault tolerance features that have recently come with the introduction of a new set of services in the IEEE 802.1 "Time-Sensitive Networking" (TSN) standard. TSN allows, among other features, to statically schedule transmissions avoiding unconstrained delays. The performance evaluation of the two schemes is done both through reliability analysis and in the network simulator OMNeT++ [2], showing their development for a growing sum of compromised wired links.

The remainder of the paper is structured as follows. Section II presents an overview of the related work. Section III addresses the threats to real-time networks service and the mechanisms enabling fault tolerance on Ethernet. Section IV presents the contribution based on combining wired and wireless PHY for fault tolerance using high throughput standardized solutions. Section V addresses the reliability analysis. Section VI covers the evaluation of the proposal analytically and through simulation. The paper conclusions are given in Section VII.

II. Related Work

The Ethernet standard does not cover any mechanisms for fault tolerance, but its popularity triggered the introduction of several solutions to address this gap at the link layer. The High-Availability Seamless Redundancy protocol (HSR; IEC

62439-3:2016) relies on two ring topologies to which nodes get attached using two ports, enabling spatial diversity by having two routes between each pair of nodes. Both paths are working in parallel, reducing to zero the time to recover when one of the rings fail. Similarly, the Parallel Redundancy Protocol (PRP; IEC 62439-3:2016) attaches the nodes to two independent networks, but in this case without restricting their topologies to rings. In the Media Redundancy Protocol (MRP; IEC 62439-2) devices are connected to a single ring using two ports. To avoid loops in the ring, one node is selected as *ring manager* and keeps one of its ports blocked. If a device in the ring detects a broken link, a message is sent to the ring manager so that it activates the blocked port. Both PRP and MRP provide a fast recovery when a failure in one of the networks is detected but require of specific hardware mostly conceived for industrial networks.

Link-state routing protocols like the Spanning Tree Protocol (STP; IEEE 802.1Q) were originally designed to avoid loops on Ethernet networks, but serve also to enable alternative routes after topology changes. The protocol exchanges the views of individual routers about the network topology until the whole network converges. On STP, the protocol goes through several phases that can take up to 50 s to finalize. A faster convergence can be achieved with Rapid Spanning Tree Protocol (RSTP; IEEE 802.1Q), yet in the order of seconds, a time that is excessive for a large portion of real-time use cases. The spanning tree protocols have evolved and current state of the art in Ethernet is provided by Shortest Path Bridging (SPB; IEEE 802.1aq), described in more detail in Section III-B.

In wireless settings, WirelessHART (IEC 62591 and IEEE 802.15.4) stands out as industrial commercial solution. The protocol relies on mesh topologies over which a centralized network manager builds a couple of primary and backup paths between each pair of end nodes. The network manager reacts to topology changes and distributes the routes along with transmission schedules to the network participants, based on the topology and traffic requirements periodically reported by them. The use of a backup route provides seamless redundancy. Yet, a change in the topology might trigger the recalculation of primary and/or backup paths, a process that requires hundreds of seconds in a mid-size network [3].

The fault tolerant solution in [4] is conceived for low energy consumption and low throughput, and relies on the wired and wireless standards CAN and IEEE 802.15.4 respectively. Nodes are restricted to a clustered topology, with clusters gathering data periodically and sending them to other clusters over the wired links or, if broken, over the wireless ones until a base station is reached.

III. Threats to Real-Time Networks Service and Fault Tolerance Mechanisms on Ethernet

A. Threats to real-time networks service

The correct service of a real-time network is compromised whenever any of the following two circumstances incur: excessive delay and low reliability. Delay can go beyond the acceptable mainly due to end-systems transmitting at any time and by an unbounded amount. This behaviour is highly dependent on the MAC protocol, propagating the problem to the switches and causing the network to fail at processing traffic that tries to use the same shared resources, namely links and device memory. Reliability is compromised if the network components or the links used by the data messages are lost for various reasons, including overheating, electric damage, physical wear or interference. This paper focuses on the case when wired physical links break permanently, a scenario envisaged under extreme conditions like catastrophes or security breaches.

B. Fault tolerance on real-time Ethernet networks

The Ethernet standard describes a variety of wired PHY offering distinct levels of throughput and reliability. Together with rugged versions of the network components, it is possible to count with highly reliable Ethernet networks. Despite this advantage, transmissions still rely on physical connections over wires which can get broken in some cases. Further, a main obstacle remains for its adoption as a real-time technology. Ethernet's baseline MAC, carrier sense multiple access with collision detection (CSMA/CD), is built upon the concept of transmit as soon as no concurrent transmission is detected and does not provide any mechanism to control data messages (*data frames* in the Ethernet and IEEE 802.11 terminology) from overflowing the memory of the switches and getting lost or experiencing longer delays than acceptable.

This undesired situation is avoided by sharing links and switches memory in a deterministic manner. In the case of Ethernet, the mechanism described in the "Enhancements for Scheduled Traffic" amendment (IEEE 802.1Qbv) from TSN allows to control the moment when data frames are sent over the links. With this mechanism, data frames are classified into 8 different priorities (defined in IEEE 802.1Q), each of them having allocated memory, i.e. queues. This mechanism enables device memory partition between data frames, meant to serve applications of different criticality, while the individual control of each of the queues by the schedule allows a proper partition of the link access.

Yet, the problem of data frames sharing links and queues while requiring a timely delivery is challenging. This scheduling problem, not covered by the TSN standards, becomes harder when the simplicity given by bus or star topologies is replaced by multi-hop switched topologies. Luckily, there are already scheduling solutions available [5]. An additional requirement for a time-triggered mechanism like this to work is to have a shared time base for the different switches. Using synchronization protocols (e.g. IEEE 802.1AS), a common notion of time is established between switches, enabling schedules to be followed in a coordinated manner. Despite dynamic reconfiguration of the network, including the distribution of the schedules for the queues, is part of the TSN standardization, it is still an incipient field of study. The main cause are the user-specific scheduling tools that might struggle to create a new schedule as fast as to keep the network running when a topology change is detected (e.g. due to a broken link). For this

reason, in real-time networks both the schedule of the queues and the routing of data frames are typically kept static.

Commonly, resources are over-provisioned to cover for failure scenarios. Several routes can be allocated to allow a data frame to reach destination from the sender. Depending on how these routes are used, it is said to perform a *cold standby* if only one of the redundant paths is used and other paths get enabled only when a problem in the primary one is detected. On the contrary, *hot standby* operates all the routes concurrently. Having the obvious drawback of permanently using network resources even when there are no problems present in the network, hot standby features a reduced complexity. This happens because hot standby does not require a mechanism to detect that a problem happened to activate the backup and consequently does not incur in the time overhead this mechanism requires to recover the connection.

TSN includes mechanisms to implement redundancy logically when an underlying physical redundant topology exists. The "Path Control and Reservation" amendment (IEEE 802.1Qca) relies on the SPB routing protocol to calculate the shortest routes between each pair of switches. SPB runs on the information provided by the Intermediate System - Intermediate System (IS-IS) link state protocol that is encapsulated on Ethernet frames. IS-IS is used by each switch to discover the topology of the network and exchange information about the cost of the routes. At first, switches exchange IS-IS Hello messages to discover neighbour switches and detect topology changes. Hello messages give only a view of the adjacencies, while Sequence Number PDUs (SNPs) and Link State PDUs (LSPs) are used to sprawl the information. This is done until all switches converge in having the same view of the network topology, an information that is stored on each node in Link State Databases (LSDBs). Once convergence is reached, SPB sends the data messages on the route having the lowest cost. In case there are several routes having the same cost, a copy of the data frame is sent on each of them. With a proper tuning of this mechanism it is possible to have concurrent transmissions over paths having the same cost or enable backup routes after the primary one (that had the lowest cost) is not available.

In the "Frame Replication and Elimination for Reliability" amendment (IEEE 802.1CB), the different copies of a data frame are tagged with the R-TAG placed at the beginning of the MAC payload which carries a sequence number. With the sequence number, duplicated data frames taking different paths are discarded at the switches in which routes merge.

IV. Combination of Diverse Physical Layers for Fault Tolerance

A. Network and traffic model

In this paper the wired network adopts the multi-hop switched topology in which *end-systems* are interconnected via *switches* that can at the same time be connected together. This topology allows several links to be used concurrently. Albeit a larger number of hops, throughput does not get as much compromised as in a bus topology when the number of end-systems increases. To enable the use of the wireless backup network, the end-systems are also equipped with wireless interfaces (called *stations* in IEEE 802.11). The wireless-capable end-systems that are in the range of each other will work in a fairly similar way to a wired bus, considering that wireless transmissions cannot take place concurrently in the same frequency. The topology adopted for the wireless network is *ad-hoc*, a setup that enables wireless interfaces to communicate directly. In contrast, the *infrastructure* wireless topology from IEEE 802.11 requires the use of intermediate nodes (*access points*), rather conceived to channel the access to a wired network. Switches could as well equip wireless interfaces, but a direct transmission between wireless end-systems that are in range seems more reasonable. This avoids the unnecessary overhead of adding more hops via switches for a message that should anyway make use of the wireless backup network. For cases where the range of a IEEE 802.11 link is not enough, a routing strategy via wireless-enabled switches can be used, but this is out of the current scope.

The topology of the network is described by the undirected graph $G(V, E)$, with V referring to network nodes (wired and wireless-capable end-systems and wired switches) and E representing the physical links between the nodes. To simplify the explanation of the coming algorithms and figures, the node $v_o \in V$ is used to refer to the wireless broadcast domain, meaning that any link with v_o as destination will reach any $v \in V$ having a wireless interface. Each physical link is bi-directional and defined by two directed dataflow links in L:

$$\forall [v_1, v_2] \in V : (v_1, v_2) \in E \Rightarrow [v_1, v_2], [v_2, v_1] \in L. \quad (1)$$

Data messages dm_k are transmitted periodically over the network, with k being the data message identifier and $dm_k.period$ representing the period. Each of the periodically generated instances $dm_{k,z}$ is identified by a sequence number z and the pair $\{k, z\}$ is used for detecting duplicates. Further, every dm_k has associated information about the set of wired paths $dm_k.paths \in dm.paths$ it is to be transmitted. For simplicity, this paper assumes that only one wired path exists per dm_k so $dm_k.path = dm_k.paths$. The actual path is stored in $dm_k.path.path$, that contains the ordered set of dataflow links it spans in the form $(dm_k.path.path_1, ..., dm_k.path.path_n) = ([v_a, v_b], [v_b, v_c], ..., [v_d, v_e])$. Further, the $dm_k.path$ can adopt two states in $dm_k.path.active$ that indicate if it is enabled.

B. Fault tolerance with a IEEE 802.11 backup network

IEEE 802.11's baseline MAC, carrier sense multiple access with collision avoidance (CSMA/CA), is also based on the listen before talk principle, but worsen by the half-duplex nature of wireless transmissions. The half-duplex operation imposes an important restriction over the use of the medium, since several wireless end-systems having information to transmit cannot get concurrent access and greatly restricting the use as compared to a wired switch topology. The main implication when using a wireless network to add redundancy to a wired network is that not every transmission can be multiplexed. Yet, the MAC protocol can be improved to support real-time requirements by guaranteeing that the access to the

medium is not done concurrently [6], e.g. by implementing mechanisms similar to those described by IEEE 802.1Qbv and IEEE 802.1AS. Taking this into consideration, this paper proposes two redundancy schemes differing in the way they allocate the wireless bandwidth: *statically* and *dynamically*.

1) Static redundancy: The static redundancy scheme, described by the pseudocode in Algorithm 1 and by the example given in Figure 1, is hot standby based with scheduled traffic on the wired network also scheduled to be transmitted on the wireless network (Algorithm 1: lines 6-9). Except for some cases in which the amount of traffic in the wired network is as little as for the wireless one to assimilate it, it is better to assume that not all traffic flows can be scheduled in the wireless network but a subset of them. To figure out how much traffic can be allocated in the wireless network, schedulability analysis can be used. With the wireless data frames able to carry the priority tag that identifies them with one of the 8 IEEE 802.1Q priorities, a subset of transmissions can be chosen starting from the highest priority.

The pseudocode shows the handling of a data message instance $dm_{k,z}$ at node v_i in the case it is an end-system (Algorithm 1: lines 1-13) or a switch (Algorithm 1: lines 14-20) when working under the static redundancy scheme and including a mechanism for discarding duplicates at the end-systems (Algorithm 1: lines 11-12). An implementation of the duplicate detection can be done with the sequence number contained in the R-TAG from IEEE 802.1CB, independently from having a wired or wireless PHY.

Algorithm 1 Static redundancy scheme

```
1: procedure STATIC_REDUNDANCY_ES_i
2:     dm_recv ← {}
3:     while true do
4:         wait_for_msg()
5:         if dm_{k,z} ←recv_from_app() then
6:             for j ← 1 to |V| do
7:                 if [v_i, v_j] ∈ dm_k.path.path then
8:                     send_scheduled(dm_{k,z}, [v_i, v_j])
9:             send_scheduled(dm_{k,z}, [v_i, v_0])
10:        else if {dm_{k,z}, [v_x, v_i]} ←recv_from_link() then
11:            if dm_{k,z} ∋ dm_recv then
12:                dm_recv ← dm_recv ∪ {dm_{k,z}}
13:                send_to_app(dm_{k,z})

14: procedure STATIC_REDUNDANCY_SW_i
15:     while true do
16:         wait_for_msg()
17:         {dm_{k,z}, [v_x, v_i]} ←receive_from_link()
18:         for j ← 1 to |V| do
19:             if [v_i, v_j] ∈ dm_k.path.path then
20:                 send_scheduled(dm_{k,z}, [v_i, v_j])
```

2) Dynamic redundancy: The dynamic redundancy scheme, described by the pseudocode in Algorithm 2 and by the example given in Figure 2, is based on cold standby and does not pre-allocate any wireless bandwidth. Instead, it reacts to broken paths and builds an alternative route between the end-systems. The dynamic redundancy scheme works in three steps. First it should detect the loss of a wired device or a link

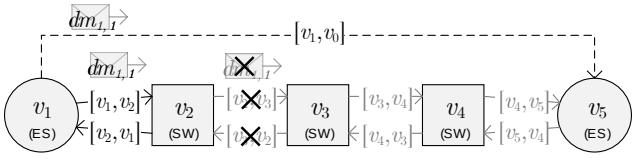

Fig. 1. Static redundancy scheme. In this example, two paths are concurrently used to transmit a data message instance $dm_{1,1}$ from v_1 to v_5 via end-systems (ESs) and switches (SWs). The wired path (continuous line arrow) breaks, leaving the wireless one (doted line arrow) as the only functional.

(Algorithm 2: lines 10 and 27; Figure 2: step 1). A common approach in link-state routing protocols is to periodically send heartbeat messages from every switch to its neighbour switches. After some time without receiving a heartbeat, the link can be declared as broken. For simplicity, this paper assumes that the PHY on the switches is able to detect broken links (e.g. by checking the electrical connection).

Once the sender realizes that a link $[v_i, v_j]$ is broken, the switches having routes traversing the broken link are informed. Scheduled paths, reverse to the original data message paths from $dm_k.paths$ could be employed for such task. However, for simplicity and considering that cold standby targets a better bandwidth allocation, any time left after scheduled traffic will be used for these exceptional transmissions. A link state message lsm carrying an identifier for the broken link $lsm.broken\_link \leftarrow [v_i, v_j]$ is sent on all the routes that include the broken link $[v_i, v_j] \in dm_k.path.path$ (Algorithm 2: lines 28-29; Figure 2: step 2). The lsm might hop through several switches, with each switch deleting the broken path from its own list of routes and relaying it over the affected paths (Algorithm 2: line 37). An implementation over Ethernet can use the LSP messages from IS-IS to perform the updates in the routing tables. Next time a switch has to send a data message, it will use the other redundant paths, if any, instead of the just broken one (Algorithm 2: line 13; Figure 2: step 3). Considering that the wireless backup network will be accessed only after random situations like the one described, wireless transmission slots are not pre-allocated. Instead, the standard CSMA/CA protocol is used despite it might be affected by other backup transmissions.

The pseudocode shows the handling of a data message instance $dm_{k,z}$ and link state messages lsm at node v_i in the case it is an end-system (Algorithm 2: lines 1-23) or a switch (Algorithm 2: lines 24-37) when working under the dynamic redundancy scheme and including the same mechanism for discarding duplicates at the end-systems from the static redundancy case (Algorithm 2: lines 17-18).

For both static and dynamic redundancy cases, no transient failures are considered on the wired links, meaning that once they break they cannot be recovered as it would happen when a wire is physically damaged.

V. Reliability Analysis

The analysis here presented uses reliability block diagrams (RBD) to calculate the reliability of the whole system at instant

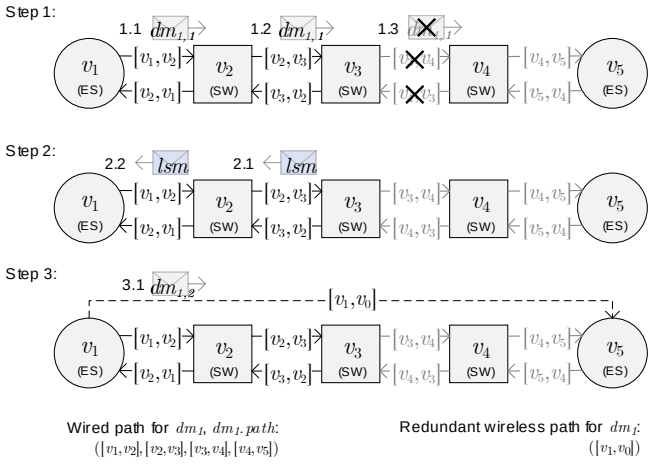

Fig. 2. Dynamic redundancy scheme. In this example, the original wired path (continuous line arrow) $dm_1.path$ for a data message instance $dm_{1,1}$ between v_1 to v_5 via end-systems (ESs) and switches (SWs) breaks (step 1). The sender gets informed (step 2) with a link state message lsm so that from that time on (step 3) it sends the data message instances $dm_{1,x}$ over the wireless interface (doted line arrow).

t: $R_S(t)$. A data message dm_i has an associated reliability $R_i(t)$ and its influence in $R_S(t)$ is weighted by w_i depending on the amount of data it carries yielding

$$R_S(t) = \sum_{i=1}^{|dm|} w_i R_i(t) \tag{2}$$

and the weight w_i being proportional to $dm_i.period$ as in

$$w_i = \frac{dm_i.period}{\sum_{j=1}^{|dm|} dm_j.period}. \tag{3}$$

Each dm_i goes over the set of wired dataflow links specified in $dm_i.path.path$ that fail at a constant rate λ_{L_wd}.

In the basic case with no redundancy, the reliability of a data message is calculated as the reliability of a set of blocks, i.e. wired links, connected in series:

$$R^i_{no\_red}(t) = R^i_{wd}(t) = \prod_{k=1}^{|dm_i.path.path|} R_k = \prod_{k=1}^{|dm_i.path.path|} e^{-\lambda_{L\_wd}t}. \tag{4}$$

When static redundancy is applied, a selected number of data messages are additionally transmitted over a wireless dataflow link. Note that for all the other data messages that are not working with static redundancy, Equation 4 still applies. Under static redundancy, dm_i is transmitted successfully if either or both the wired or the wireless paths work. For simplicity, a wireless transmission performed between two devices in range fails only when it collides with other transmissions. Since the static allocation avoids concurrent transmissions, it yields a reliability value of 1:

Algorithm 2 Dynamic redundancy scheme

```
1: procedure DYNAMIC_REDUNDANCY_ES_i
2:     dm_recv ← {}
3:     for all dm_k.path ∈ dm.paths do
4:         dm_k.path.active ← 1
5:     while true do
6:         wait_for_msg()
7:         if df_{k,z} ←recv_from_app() then
8:             for j ← 1 to |V| do
9:                 if [v_i, v_j] ∈ dm_k.path.path then
10:                    if check_phy([v_i, v_j]) is broken then
11:                        dm_k.path.active ← 0
12:                    if dm_k.path.active = 0 then
13:                        send_asap(dm_{k,z}, [v_i, v_0])
14:                    else
15:                        send_scheduled(dm_{k,z}, [v_i, v_j])
16:        else if {dm_{k,z}, [v_x, v_i]} ←recv_from_link() then
17:            if dm_{k,z} ∋ dm_recv then
18:                dm_recv ← dm_recv ∪ {dm_{k,z}}
19:                send_to_app(dm_{k,z})
20:        else if {lsm, [v_x, v_i]} ←recv_from_link() then
21:            for j ← 1 to |dm| do
22:                if lsm.broken_link ∈ dm_j.path.path then
23:                    dm_j.path.active ← 0

24: procedure DYNAMIC_REDUNDANCY_SW_i
25:     while true do in parallel
26:         for j ← 1 to |V| do
27:             if check_phy([v_i, v_j]) is broken then
28:                 lsm.broken_link ← [v_i, v_j]
29:                 send_lsm(lsm)
30:     while true do in parallel
31:         wait_for_msg()
32:         if {dm_{k,z}, [v_x, v_i]} ←recv_from_link() then
33:             for j ← 1 to |V| do
34:                 if [v_i, v_j] ∈ dm_k.path.path then
35:                     send_scheduled(dm_{k,z}, [v_i, v_j])
36:         else if {lsm, [v_x, v_i]} ←recv_from_link() then
37:             send_lsm(lsm)

38: procedure SEND_LSM(lsm)
39:     for all dm_k.path ∈ dm.paths do
40:         for j ← 1 to |dm_k.path.path| do
41:             if lsm.broken_link = dm_k.path.path_j then
42:                 send_asap(lsm, dm_k.path.path_{j-1})
```

$$R^i_{sta}(t) = 1 - (1 - R^i_{wd}(t))(1 - R_{wl\_sta}(t)) = 1 - \left(1 - \prod_{k=1}^{|dm_i.path.path|} e^{-\lambda_{L\_wd}t}\right)(1-1) = 1. \tag{5}$$

As for dynamic redundancy, dm_i will be transmitted over the wireless path whenever the wired path is broken ($dm_i.path.active = 0$):

$$R^i_{dyn}(t) = 1 - (1 - R^i_{wd}(t))(1 - R^i_{wl\_dyn}(t)) = 1 - \left(1 - \prod_{k=1}^{|dm_i.path.path|} e^{-\lambda_{L\_wd}t}\right)(1 - R^i_{wl\_dyn}(t)). \tag{6}$$

The reliability of dm_i transmitted over the wireless path in the dynamic redundancy case depends on not colliding with other wireless paths:

$$R^i_{wl\_dyn}(t) = \prod_{j=1, j\neq i}^{|dm_i.path.path|} R^{i,j}_{not\_col}(t). \quad (7)$$

The probability of dm_i to not collide with another data message dm_j over the wireless channel depends on the probability of dm_j to have a wired path working ($dm_j.path.active = 1$) or, if not, that it does not collide on the wireless channel:

$$R^{i,j}_{not\_col}(t) = R^j_{wd}(t) + (1 - R^j_{wd}(t)) \cdot R_{not\_col\_wl}(t). \quad (8)$$

Finally [7] provides an estimation of the probability that one wireless transmission does not collide under DCF considering the total number of wireless end-systems that might try to transmit, which for simplification in this paper is assumed to be the same as the number of data messages $|dm|$ (each data message is transmitted by a different wireless end-system, adding some pessimism to the analysis) yielding

$$R_{not\_col\_wl}(t) = (1 - \tau)^{(|dm|-1)} \quad (9)$$

with τ being the channel access probability of a single end-system during a t_{slot} which is the time required by a wireless end-system to detect any ongoing transmissions from other end-systems, a value that is dependant on the PHY used and is given by the IEEE 802.11 standard. Given the simplification that each dm_i is transmitted by a different end-system, the channel access probability of an end-system v_i is proportional to its period $dm_i.period$, so τ is calculated as an average value for the channel access probability of all end-systems:

$$\tau = \frac{\sum_{i=1}^{|dm|} \frac{t_{slot}}{dm_i.period}}{|dm|}. \quad (10)$$

VI. Performance Evaluation

A. Evaluation setup

The performance of the static and dynamic redundancy schemes compared to the non-redundancy case has been measured by simulation and analysis. Values are provided to evaluate the *reliability* $R_S(t)$. In the simulation, this value is obtained by calculating the proportion of data messages that were received by the destination end-systems as compared to the ones that were sent. Additional results are provided by means of simulation. These include the overhead introduced by static redundancy by showing the *percentage of duplicated data messages* received at the destination end-systems compared to the data messages sent. Results are also offered for the performance of wireless transmissions under the dynamic redundancy mechanism by showing the *percentage of colliding data messages* as compared to the total number of performed ones, given that the CSMA/CA channel access might cause collisions between end-systems. Another performance measure for the dynamic redundancy mechanism addresses its *responsiveness* by measuring the time it passes between a wired link breaks and the wireless end-systems with flows going through that link get notified.

A total of 12 data messages have been generated based on the parameters from Table I. The transmission times of the data messages over the data message paths, depicted in Figure 3, are scheduled to avoid collisions and transmitted over a mid-size wired multi-hop star topology network. The end-systems (edge nodes from Figure 3) are wireless-enabled and can perform direct ad-hoc transmissions. It is assumed that all end-systems are in range and collide whenever concurrent transmissions take place.

TABLE I
GENERATED DATA MESSAGE PARAMETERS

Parameter	Values
Data message (dm) payload size	512 B
Link state message (lsm) size (set to an average value)	720 B
Data message (dm) R-TAG size	6 B
Wired data rate	100 Mbps
Wireless data payload rate (IEEE 802.11g)	54 Mbps
t_{slot} (IEEE 802.11g)	9 µs
Data message period ($dm.period$; number of messages sharing the same period)	1 ms (6x) 2 ms (2x) 4 ms (4x)

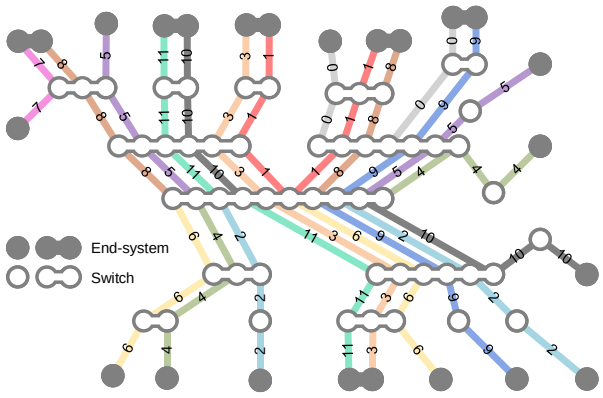

Fig. 3. The 12 generated data message paths, identified by a different number, on the multi-hop network.

A summary of the scenarios considered in the evaluation is given in Table II. The value of λ_{L_wd} is set to resemble the behaviour of a network having the number of broken wired links going from 0 to the maximum number and is not specified to represent any real failure rate of specific components. In the simulation, the two dataflows that are part of the same wire break at the same time but on average the λ_{L_wd} is still kept.

The simulation consists of a model of the redundancy schemes created for OMNeT++, a discrete event simulation framework. The model relies on the INET library [8], that implements the Ethernet and IEEE 802.11g PHY and MAC. Both MAC layers have been modified to work in a time-triggered manner as in IEEE 802.1Qbv, with transmissions allocated using the scheduler from [9]. The numerical analysis has been realized using Matlab. Results on $R_S(t)$ are retrieved for the different evaluated scenarios. Both simulation and

TABLE II
EVALUATION SCENARIOS

Parameter	Values
Redundancy scheme	None Static Dynamic
Data message period ($dm.period$) of redundant flows in static redundancy scheme (number of messages sharing the same period)	1 ms (1x) 4 ms (1x)
λ_{L_wd} (failures/s)	0.00005

analysis have been run until $R_S(t)$ becomes stable, something that happens a short time after all wired links break. The choice of λ_{L_wd} for the wired links impacts directly the amount of time the simulation and analysis should be run to observe some trends in the results. In this paper, the lower limit for λ_{L_wd} is given by the simulation tool, not in terms of time but due to the amount of logged data which easily reaches tens of gigabytes.

B. Evaluation results

Figure 4 shows the value of $R_S(t)$ when $\lambda_{L\_wd} = 0.00005$ failures/s for the case of no redundancy compared to when static and dynamic redundancy schemes are applied. The results are given for the simulation and numerical analysis. The overall pessimistic approximation of the numerical analysis becomes clear when compared to the results from the simulation. Yet, the lines for analysis and simulation are in the same order of magnitude and follow a similar pattern. As it can be expected, when no redundancy mechanisms are in place $R_S(t)$ drops to 0 with the time. These results reflect the case when the links in a wired network get broken and there is no other possible way of transmitting data between the end-systems. Static redundancy already achieves a sustained value of $R_S = 0.2$ when all the wired links break. The value of $R_S(t)$ is subject to the weight (w) of data messages that are given the chance to transmit over the wireless backup network, in this case they transmit about 20% of the traffic. The dynamic redundancy mechanism keeps a stable value over $R_S = 0.5$ about 30% improvement with respect to static redundancy in the analysed case. The limits on the $R_S(t)$ improvement are given by the amount of traffic the wireless channel is able to absorb, so intuitively the $R_S(t)$ of the static and dynamic redundancy mechanisms should be pretty similar. However, the collision-free guarantee of the static redundancy scheme limits the addition of redundant paths, even if these suffer seldom collisions.

Figure 5 shows the *percentage of duplicated data messages* at the receiver end-systems obtained by simulation. Simulated $R_S(t)$ has been added as a reference. The duplicated data messages are the main drawback from static redundancy scheme while no duplicates are present in case of no redundancy and dynamic redundancy. Duplicates indicate that both wired and wireless transmissions succeed for a data message, despite only one of them would have been needed. The figure shows that the amount of duplicates is the highest at the beginning of

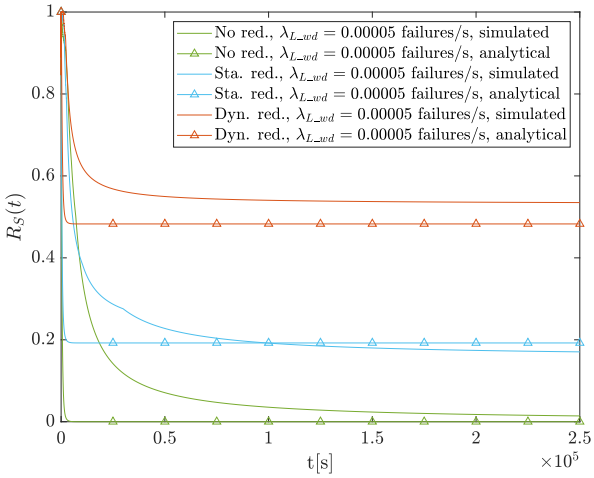

Fig. 4. Simulated and analytical reliability for $\lambda_{L\_wd} = 0.00005$ failures/s in the case of no redundancy, static redundancy and dynamic redundancy.

the simulation, matching the proportion of data messages that are able to transmit both over the wired and wireless media. These high numbers rapidly decrease due to the effect of wired links breaking, also considering that one broken wired link might affect several wired paths. The number of duplicates tends to 0 when approaching the moment when all wired links are broken. This indicator can be interpreted as the amount of redundancy present in the system and how it evolves until it is completely lost, which is the moment when the systems losses its ability to tolerate further faults.

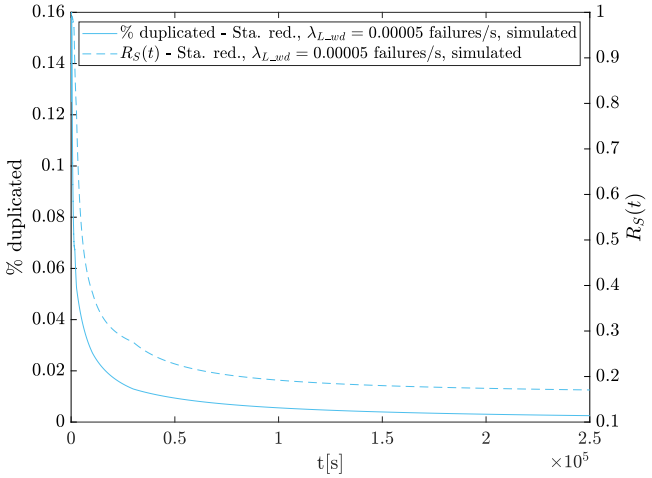

Fig. 5. Simulated percentage of duplicate data messages compared to simulated reliability for $\lambda_{L\_wd} = 0.00005$ failures/s. Duplicates only appear when applying static redundancy.

The results from Figure 6 show the *percentage of colliding data messages* when transmitted over the wireless medium. Simulated $R_S(t)$ has been added as a reference. This performance measure is interesting for the dynamic redundancy scheme in which the wireless backup network is a shared resource between every end-system that detected the wired path as broken. With the amount of traffic considered in the simulation, about 40% of the data messages sent on the

wireless network collide and do not reach destination. This number has its peak at the beginning of the simulation once wired links start to break and stabilizes with the number of wireless transmissions. Despite the proportion of lost data messages is quite high and the consequent impact on $R_S(t)$, the dynamic redundancy mechanism still outperforms the static scheme in the performed test. Yet, several techniques can be used to tackle the problem of colliding transmissions and improving $R_S(t)$ by providing a better allocation of the wireless bandwidth than the one given by CSMA/CA [10].

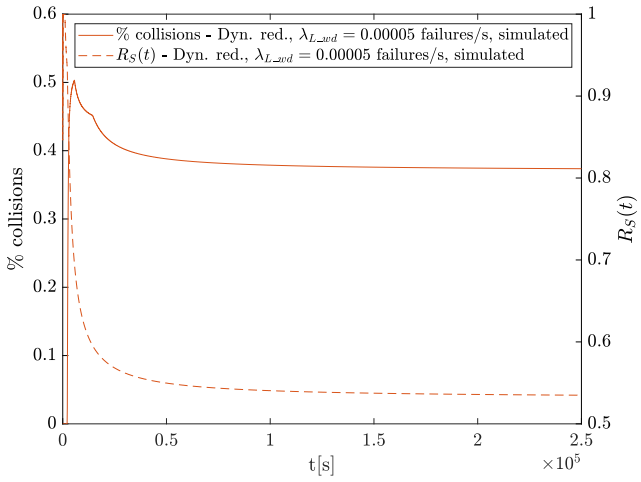

Fig. 6. Simulated percentage of wireless collisions compared to simulated reliability for $\lambda_{L\_wd} = 0.00005$ failures/s. Collisions only appear when applying dynamic redundancy.

One of the advantages of the static redundancy mechanism is that it does not take time to recover when a failure happens, since the redundant paths are working in parallel. Luckily, the *responsiveness* of the dynamic redundancy scheme is also good. The delay to notify the sender is in the interval [58.3 µs, 291.4 µs], with an average of $\overline{x}$ =131.2 µs and a standard deviation σ =103.2 µs.

Despite the $R_S(t)$ numbers of the static redundancy mechanism in the run experiments are worse than those given by the dynamic mechanism, the former is recommended whenever there are data messages with higher criticality than others. In static redundancy, these data messages are guaranteed to be collision-free with respect to other data messages in the network. While the dynamic redundancy mechanism might provide a better average $R_S(t)$, this guarantee is not given. Therefore, the choice of the redundancy mechanism depends on what is to be prioritized from the user perspective: a better average performance or a given guarantee to the most critical data messages.

VII. Conclusion

In this paper, two redundancy mechanisms have been proposed that make use of a wireless backup network to increase the reliability of a wired network that suffers from broken links. One redundancy mechanism allocates communication slots statically and works in parallel with the wired network. The other redundancy mechanism reacts dynamically to broken wired links and establishes an alternative wireless connection. The evaluation of reliability is done analytically and by simulation, proving both methods by having a similar behaviour in both cases. A constant failure rate over the wired links causes a gradual degradation of the wired network performance. When all the wired links end up breaking, the static redundancy mechanism is able to offer 20% reliability, reaching 100% for selected scheduled data messages while the overhead expressed as the percentage of duplicated data messages tends to 0 as a function of time. The dynamic redundancy mechanism gets over 50% of the data messages delivered, with almost an instant recovery of the path, even though its dynamic allocation of the wireless medium causes a considerable amount of collisions. The results show that using a wireless backup network as a fault tolerance mechanism can effectively increase the reliability of a wired network, avoiding the problem of permanent failures like in the aforementioned industrial sabotage. The choice of one of the proposed redundancy mechanisms is subject to a trade-off between guaranteeing the most critical data messages in the static case or a better average performance in the dynamic case.

Acknowledgements

This work has been supported and funded by the Austrian Research Promotion Agency (FFG) via the "Austrian Competence Center for Digital Production" (CDP) under the contract number 854187. E. Uhlemann is partly funded by the Serendipity project through the Swedish Foundation for Strategic Research.

References

[1] P. Gutiérrez Peón, E. Uhlemann, W. Steiner, and M. Bjorkman, "Applying Time Diversity for Improved Reliability in a Real-Time Heterogeneous MAC Protocol," in *Proc. IEEE VTC-Spring*, Sydney, Australia, 2017.

[2] A. Varga, "The OMNeT++ discrete event simulation system," in *Proc. ESM*, Prague, Czech Republic, 2001.

[3] J. Shi, M. Sha, and Z. Yang, "Distributed Graph Routing and Scheduling for Industrial Wireless Sensor-Actuator Networks," *IEEE/ACM Trans. Networking*, vol. 27, no. 4, pp. 1669–1682, 2019.

[4] C. Alippi and L. Sportiello, "Robust hybrid wired-wireless sensor networks," in *Proc. IEEE PERCOM*, 2010, pp. 462–467.

[5] S. S. Craciunas, R. Serna Oliver, M. Chmelík, and W. Steiner, "Scheduling Real-Time Communication in IEEE 802.1Qbv Time Sensitive Networks," in *Proc. ACM RTNS*, Brest, France, 2016, pp. 183–192.

[6] P. Gutiérrez Peón, E. Uhlemann, W. Steiner, and M. Björkman, "Medium Access Control for Wireless Networks with Diverse Time and Safety Real-Time Requirements," in *Proc. IEEE IECON*, Florence, Italy, 2016, pp. 4665–4670.

[7] G. Bianchi, "Performance Analysis of the IEEE 802.11 Distributed Coordination Function," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 3, pp. 535–547, 2000.

[8] "The INET Framework. An open-source OMNeT++ model suite for wired, wireless and mobile networks," http://inet.omnetpp.org/, accessed: 2020-03-20.

[9] F. Pozo, W. Steiner, G. Rodriguez-Navas, and H. Hansson, "A Decomposition Approach for SMT-based Schedule Synthesis for Time-Triggered Networks," in *Proc. IEEE ETFA*, Luxembourg City, Luxembourg, 2015.

[10] P. Gutiérrez Peón, P. M. Rodríguez, Z. Fernández, F. Pozo, E. Uhlemann, I. Val, and W. Steiner, "Cognitive Radio for Improved Reliability in a Real-Time Wireless MAC Protocol based on TDMA," in *Proc. IEEE ETFA*, Limassol, Cyprus, 2017.