Availability Constraints for Avionic Data Buses[1]

Alban Gabillon & Laurent Gallon
Laboratoire d'Informatique de l'Université de Pau et des Pays de l'Adour, IUT de Mont de Marsan, 371 rue du ruisseau, BP 201, 40000 Mont de Marsan – France
http://csysec.univ-pau.fr
{alban.gabillon;laurent.gallon}@univ-pau.fr

Abstract

The three traditional objectives of computer security are confidentiality, integrity and availability [8]. Availability can be defined as the prevention of denial of service. Confidentiality and integrity have been addressed in several theoretical works whereas the concept of availability has not been much investigated by the scientific community. This paper is an attempt to define through a case study the concept of availability. We first define a set of availability constraints that avionic data bus protocols should enforce. Then, we consider the ARINC 629 Basic Protocol (BP) [2] which was implemented on the Boeing 777, and we check whether that ARINC 629 BP respects our availability constraints or not.

1. Introduction

The three traditional objectives of computer security are confidentiality, integrity and availability [8]. Availability can be defined as the prevention of denial of service. A denial of service is an unauthorized withholding of information or resources. A denial of service can occur accidentally or it can take place as the consequence of a malicious action. Confidentiality and integrity have been addressed in several theoretical works whereas the concept of availability has not been much investigated by the scientific community. Let us, however, mention the models of Yu & Gligor [12], Millen [10] and Cuppens & Saurel [5].

As systems on Aircraft became progressively more digital in nature, it became apparent to avionic designers that a multiplexed bus system was required to enable all subsystems to be connected by only one set of wires. Since avionic systems place great emphasis on the reliable and timely transfer of information, specific data bus protocols for avionic systems have been designed. ARINC 629 is one of them. ARINC Specification 629 [2] defines a digital communication system, where Line Replaceable Units (LRUs : terminals connected to the network) may transmit and receive digital data using a standard protocol. A linear topology (bus) is used, and the protocol can be described as a Carrier Sense Multiple Access / Collision Avoidance (CSMA/CA). The bus access control is distributed among all the participating terminals. ARINC 629 was implemented on Boeing 777. There exist two implementations of ARINC 629:

- Combined Mode Protocol (CP)
- Basic Mode Protocol (BP)

In [7], we made a timing analysis of the ARINC 629 CP by using Stochastic Timed Petri Nets [6,9]. In [13], we did the same for the BP. In this paper we focus again on the ARINC 629 BP. However, our objective is not to evaluate the performances of this protocol. Our aim is to contribute to the definition of the concept of availability through a case study. We first define a set of availability constraints that Medium Access Control (MAC) protocols of avionic data buses should enforce. Then, we check whether ARINC 629 BP conforms to these constraints or not.

The remainder of this paper is organized as follows. Section 2 reviews previous works and informally presents our approach. Section 3 defines a language for specifying availability policies. This language is based on mathematical logic with deontic and temporal modalities. In section 4, we use our language to specify availability constraints that Medium Access Control (MAC) protocols should enforce. Section 5 gives an overview of ARINC 629 BP. Section 6 analyzes the

[1] This work has been funded by the French ministry of research under the label "ACI Sécurité Informatique 2004-2007". Project METROSEC.

availability of ARINC 629 BP. Section 7 concludes this paper

2. Approach and related works

Yu and Gligor [12] characterize availability as “how to provide a shared service with a specified *maximum waiting time*”. They use temporal logic to specify availability constraints. They also introduce the notion of *user agreements*. User agreements are constraints on the behavior of subjects. Yu and Gligor argue that lack of specifications for these agreements makes it impossible to demonstrate denial of service prevention.

Millen proposes a model [10] which resembles the Yu & Gligor model. In particular, the model includes the concept of *waiting time policy* defined by Yu and Gligor. The difference between the Yu and Gligor model and that of Millen’s is that Millen uses a set-theoretic approach and includes an explicit representation of time.

Cuppens and Saurel [5] propose another approach. They define a logic-based language to specify availability policies. Their language basically includes two major predicates:

- *disposal_right*(*s,r,d*) reads subject *s* is permitted to access resource *r* and must be provided with *r* within a maximum waiting time of *d*
- *use_right*(*s,r,d*) reads subject *s* is permitted to use resource *r* for a maximum duration of *d*.

With the first predicate, Cuppens & Saurel define the *waiting time policy*. With the second predicate, they define the *use time policy*. Holding a *disposal right* on a resource implies holding a *use right* on the same resource. Cuppens and Saurel express the meaning of both predicates in terms of deontic modalities. With their language, they also state some availability properties that the system should enforce. Basically, these properties express the fact that there is a security violation whenever the policy is not respected.

As we can see, Cuppens and Saurel declare explicit permissions in their policy. Consequently, there is a difference between the *waiting time policy* of Cuppens & Saurel and the *waiting time policy* of Millen and Yu & Gligor. For Millen and Yu & Gligor, the waiting time policy consists of a security constraint which says that a subject which asks for a resource has to be provided with it with a maximum waiting time. The policy of Millen and Yu & Gligor does not refer to explicit permissions.

Our approach resembles the Cuppens & Saurel model. In particular, the language we propose in section 3 is inspired from their language. Our approach can be summarized as follows: we do not define availability policies for avionic data buses. We rather state a set of availability constraints that MAC protocols of avionic data buses should enforce. Informally, we state that the access control policy of an avionic data bus should respect the following principles:

- Only one terminal at a time should hold the right to use the bus.
- There should be a fair distribution of rights among the terminals. Each terminal should be granted the right to use the bus once per bus cycle.
- If a terminal has been granted the permission to use the bus and actually uses it, then it should be permitted to use the bus as long as it needs, provided it does not exceed a specified duration.

We shall say that the bus is fully available if and only if these principles always apply. In the last part of this paper we study the conformance of the ARINC 629 BP access control protocol with our availability constraints and we discuss the concept of denial of service.

3. Language

In this section, we define the language which we use to express the availability constraints. This language can be seen as a simplified version of the language proposed by Cuppens & Saurel [5].

The language is based on first order logic with equality. It is extended with temporal logic and deontic concepts.

3.1. Temporal notions

The temporal dimension of the language comes from Sripada [11]. The language uses a discrete representation of time:

- $\mathrm{date(t)}$ reads *t* is a date

The language introduces the following two temporal modalities:

If *p* is a formula and *t* is a date then

- $\mathrm{p_t}$ reads *p* is true at time *t*
- $\mathrm{p_{t_1,t_2}}$ reads *p* is true during the temporal interval [t_1,t_2]

Axiomatic of these modalities is the following:

- $\mathrm{(p \wedge q)_t \leftrightarrow p_t \wedge q_t}$
- $\mathrm{(\neg p)_t \leftrightarrow \neg p_t}$
- $\mathrm{p_{t_1,t_2} \leftrightarrow \forall t, t_1 \leq t \leq t_2 \rightarrow p_t}$

- if p is a theorem then p_t is also a theorem for every date t.

We shall use the following abbreviation:

- $\mathrm{always}(p) \leftrightarrow \forall t, \mathrm{date}(t) \rightarrow p_t$

Our language includes the following constant D_{max} which represents an arbitrary number of time units.

3.2. Deontic modalities

The language uses the following classical deontic modalities: **O**, **F** and **P** [4].

- **O**p reads p is mandatory
- **F**p reads p is forbidden
- **P**p reads p is permitted

We have the following equivalences:

- $\mathbf{F}p \leftrightarrow \mathbf{O}\neg p$
- $\mathbf{P}p \leftrightarrow \neg\mathbf{F}p$

Axiomatic of **O** is the following:

- $\mathbf{O}(p \wedge q) \leftrightarrow \mathbf{O}p \wedge \mathbf{O}q$
- $\neg(\mathbf{O}p \wedge \mathbf{O}\neg p)$
- if p is a theorem then **O**p is also a theorem.

3.3. Subjects

We use the following predicate to represent the subjects:

- LRU(s) reads s is an LRU (terminal)

Following constants represent the different terminals which are connected to the bus:

- $L_1, L_2 \dots L_n$

We state the following axiom:

- $\mathrm{always}\begin{pmatrix} \forall s, \mathrm{LRU}(s) \\ \leftrightarrow s = L_1 \vee s = L_2 \vee \ldots \vee s = L_n \end{pmatrix}$

Following predicates respectively represent the fact that a subject uses the bus and the fact that a subject needs the bus:

- use(s) reads s uses the bus
- need(s) reads s needs the bus

We assume the following integrity constraints hold:

- $\mathrm{always}(\forall s, \mathrm{use}(s) \rightarrow \mathrm{LRU}(s))$
- $\mathrm{always}(\forall s, \mathrm{use}(s) \rightarrow \mathrm{need}(s))$

First constraint says that if s is using the bus then s is an LRU. Second constraint says that if s uses the bus then s needs the bus. In other words this constraint assumes that a subject would not use the bus if it does not need it. Such an integrity constraint is referred to as a *user agreement* by Yu and Gligor [12].

4. Availability constraints for avionic buses

In this section, we state some availability constraints that MAC protocols of avionic data buses should enforce. Our proposal comes from our comprehension of what the "*ideal*" MAC layer of an avionic bus should be. Principles like fair distribution of rights and optimization of bus utilization have guided us for stating those constraints.

We consider that the ideal MAC protocol for avionic data buses should respect the following constraints:

The first rule says that only one terminal at a time can be granted the permission to access the bus.

1. $\mathrm{always}(\forall s \forall s', \mathbf{P}\mathrm{use}(s) \wedge \mathbf{P}\mathrm{use}(s') \rightarrow s = s')$

The second rule says that, at any time, there is at least one terminal which is granted the permission to access the bus

2. $\mathrm{always}(\exists s, \mathbf{P}\mathrm{use}(s))$

From rule 1 and rule 2, we can easily deduce that there is always one and only one terminal which is granted the permission to access the bus.

The third rule says that a terminal which was granted the right to use the bus cannot be granted the right to use the bus again before all the other terminals have been granted the right to use the bus. In other words, distribution of rights has to be fair.

3. $$\begin{aligned} &\forall s \forall t_1 \forall t_2 \forall t_3, [\mathbf{P}\mathrm{use}(s)]_{t_1} \wedge [\mathbf{F}\mathrm{use}(s)]_{t_2} \wedge [\mathbf{P}\mathrm{use}(s)]_{t_3} \\ &\wedge (t_1 < t_2 < t_3) \\ &\rightarrow \begin{pmatrix} \forall s', \mathrm{LRU}(s') \wedge s' \neq s \\ \rightarrow \exists t_4, [\mathbf{P}\mathrm{use}(s')]_{t_4} \wedge (t_1 < t_4 < t_3) \end{pmatrix} \end{aligned}$$

The fourth rule says that a terminal which is granted the right to access the bus holds the right as long as it needs the bus.

4. $\forall s \forall t, [\mathbf{P}\mathrm{use}(s)]_t \wedge [\mathrm{need}(s)]_{t+1} \rightarrow [\mathbf{P}\mathrm{use}(s)]_{t+1}$

However, the fifth rule says that the amount of consecutive time a terminal can hold the right to use the bus is bounded.

5. $\forall s \forall t_1 t_2, [\mathbf{P}\mathrm{use}(s)]_{t_1,t_2} \rightarrow t_2 - t_1 < D_{max}$

The last rule says that if a terminal is permitted to use the bus and if it needs the bus then it is mandatory that the terminal uses the bus.

6. $\mathrm{always}(\forall s, \mathbf{P}\mathrm{use}(s) \wedge \mathrm{need}(s) \rightarrow \mathbf{O}\mathrm{use}(s))$

We could easily show that these constraints are consistent, except in one case: if a terminal has been holding the right to use the bus for a duration which has become greater than D_{max} then rule 4 conflicts with rule 5. We solve this conflict by considering that rule 5 is of a higher priority than rule 4. Rule 4 and rule 5 define the *use time policy*.

We do not assign priorities to terminals. We claim that priorities are application dependent. Priorities, if any, have to be defined by the availability policy of an upper layer. Consequently, our security policy does not say in which order the terminals receive the right to use the bus. Our constraints do not express an explicit *waiting time policy*. However, from rules 3 and 5, we can derive that the maximum amount of time a terminal has to wait before being granted with the permission to use the bus is equal to $(n-1)\times D_{\max}$ units of time. This maximum waiting time can theoretically be reached in the following situation:

- The terminal has just released the bus and is waiting for its next turn
- Each of the *n*-1 other terminals uses the bus for the $D_{\max}$ duration.

The MAC layer of an avionic bus is secure if and only if the security policy is not violated. This means that the following two *security constraints* should be enforced:

The first security constraint says that a terminal which uses the bus should be permitted to use the bus. Violation of this constraint is an *access control violation*

A. $\text{always}(\forall \text{s}, \text{use}(\text{s}) \rightarrow \mathbf{P}\text{use}(\text{s}))$

The second property says that if it is mandatory that a terminal uses the bus then the terminal actually uses the bus. Violation of this constraint is a *denial of service*.

B. $\text{always}(\forall \text{s}, \mathbf{O}\text{use}(\text{s}) \rightarrow \text{use}(\text{s}))$

5. Overview of the ARINC 629 BP

The ARINC 629 Basic Protocol is one implementation of the ARINC specification 629. The purpose of this section is to briefly describe the medium access control policy of this protocol. For more detailed information the reader can refer to [2] or [3]. An ARINC 629 message has variable length up to 31 wordstrings. Each wordstring is made up of one label and several data words. Bus time is decomposed into bus cycles. Each terminal sends one message per bus cycle. The ARINC 629 BP defines a Carrier-Sense Multiple Access/Collision Avoidance (CSMA/CA) scheme. Bus access control is distributed amongst all participating terminals, each of which autonomously determines its transmission sequence. This is achieved by the use of bus access timers as follows:

- Transmit Interval (TI): same for all terminals; 0.5 to 64.0ms; the longest timer; starts every time the terminal starts transmitting; is equal to the minimum bus cycle time
- Synchronization Gap (SG): same for all terminals; values are 17.7µs, 33.7µs, 65.7µs, 127.7µs; ensures that all terminals are given access; chosen to be greater than the maximum TG; starts every time *Bus Quiet* (BQ) is sensed and may be reset before it has expired if bus activity sensed; restarted next time the terminal starts to transmit.
- Terminal Gap (TG): used to differentiate between terminals; unique for each terminal; values are 3.7µs to 127,7µs; starts only after the SG has expired and the bus is quiet; reset if bus activity sensed.

The initialization phase of bus operation may be relatively uncontrolled depending on the sequence in which local clocks are initialized across multiple terminals. However, as a result of synchronization mechanisms bus operation will normally settle down to some steady state after a very short time.

Figure 1 shows basic protocol periodic mode timing for a very simple bus with just three terminals. The order in which terminals achieve bus access is the same for all bus cycles and is determined by the initialization sequence. It does not necessarily equate to the relative durations of the terminals' TG timers. If the sum of all the TGs, transmission times and SG is less than TI then the bus cycle time remains fixed and equal to TI.

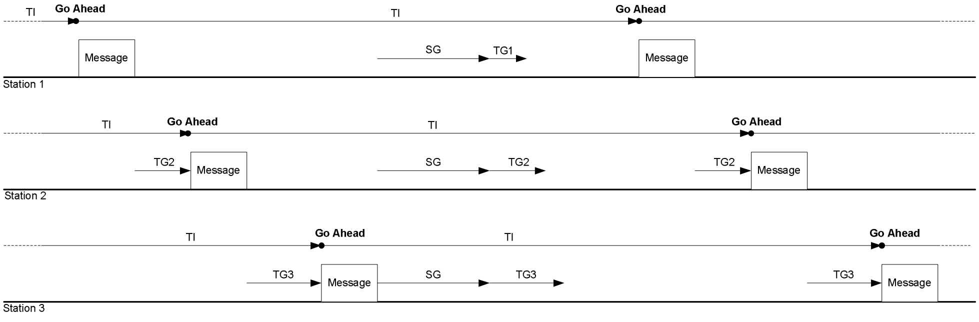

Fig 1 : periodic mode

If the sum of all the TGs, transmission times and SG is greater than TI then the system operates in aperiodic mode as shown in Figure 2. For every bus cycle the terminals transmit in the order of their TG durations. Here, TG1<TG2<TG3. Each cycle consists of a sequence of transmissions separated by the various TG delays; these are followed by an SG delay that provides synchronization. The two modes are not exclusive and the aperiodic mode will revert to periodic mode after transient overload.

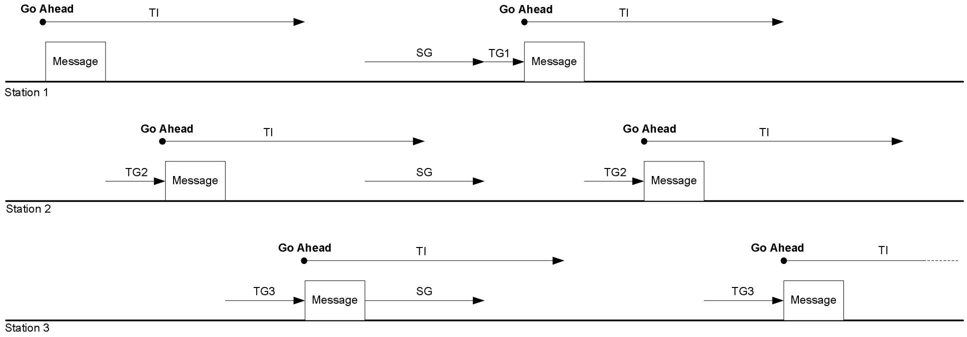

Fig 2 : aperiodic mode

6. Analysis of ARINC 629 BP

In this section we first check whether ARINC 629 BP conforms to the constraints we defined in section 4. The purpose of this section is not to make a formal analysis of the ARINC 629 BP. Our aim is to consider some particular configurations in order to understand the concept of availability and denial of service. For our analysis, we choose the following four configurations:

- We consider the periodic mode after the initialization phase that is, after the bus has settled down to some steady state.
- We consider the aperiodic mode
- We consider the initialization phase
- We consider the transition between the periodic mode and the aperiodic mode

Then, in the last sub-section, we show that there is denial of service each time the constraints are not respected.

6.1. Periodic mode

Periodic mode is described by figure1. A terminal is permitted to transmit (to use the bus) when it reaches the go-ahead condition. The terminal reaches that condition immediately after its three timers have expired. In figure 1, we can see that each terminal which reaches the go-ahead condition can actually start transmitting. Therefore, constraint 6 is satisfied. After it has started transmitting, a terminal is permitted to transmit as long as it needs and cannot be pre-empted. Therefore, constraint 4 is respected. A terminal cannot send a message which is more than 31 wordstrings (see section 5). Since the length of a wordstring ranges from 20 to 4540 bits and since a 2Mbps serial data transmission is specified for twisted pair conductor, the maximum transmission time of a message is 70.37ms. Therefore, constraint 5 applies and constant D_{max} is equal to 70.37ms. No terminal has its three timers expired while another terminal is still transmitting. Therefore, constraint 1 is satisfied. Within a single bus cycle, terminal 1 transmits then terminal 2, then terminal 3. The same procedure restarts at the next bus cycle. Therefore, constraint 3 is satisfied. In figure 1, we can see that after terminal 3 has released the bus none of the terminals have their timers expired yet. Therefore, none of the terminals have the permission to use the bus. Consequently **rule 2 is violated.**

6.2. Aperiodic mode

Aperiodic mode is described by figure 2. If we proceed in the same way as we did for the periodic mode then we can easily show that only **rule 2 is violated**. Indeed, there are some time intervals where all the terminals are waiting for some timers to expire. Other rules are never violated.

6.3. Initialization phase (worst case)

We could show that during a safe initialization phase only rule 2 is violated. We consider the worst case of an initialization phase which leads to a bus clash. Figure 3 describes this configuration. Terminal 2 starts before terminal 1. Both terminals are running their SG and TG (rule 2 is violated). After their SG and TG have expired, both terminals are permitted to transmit at the same time. Therefore, **rule 1 is violated**. Both terminals try to transmit at the same time. A Bus Clash (BC) is detected. Both terminals enter into a recovery phase. **Security constraint B is violated** since none of the terminals could use the bus whereas it was mandatory that both terminals could use the bus. After their TI has expired, they have to run their TG again. Since TG1 has expired first, terminal 1 is the first to transmit data.

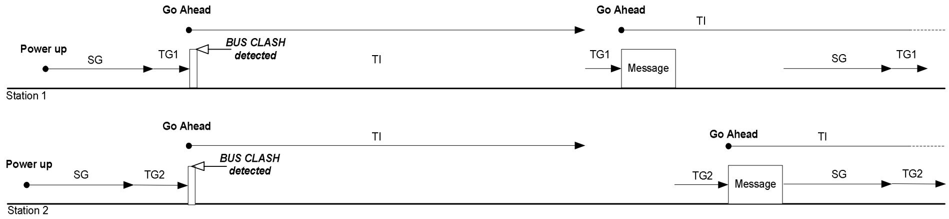

Fig 3 : Bus Clash

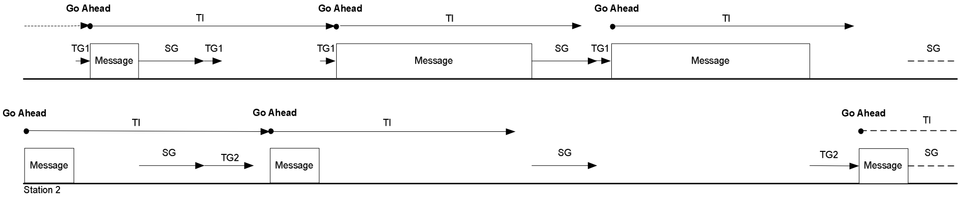

Fig 4: Transition between periodic mode and aperiodic mode

6.4. Transition between modes

Figure 4 shows a particular case of transition from periodic to aperiodic mode. At the end of the initialization phase, terminal 2 is the first to transmit data. During the first cycle the bus is operating in the periodic mode. The order in which the terminals transmit depends on bus initialization. At the second bus cycle the order in which the terminals send their message remains the same (first terminal 2 then terminal 1). However, since terminal 1 has sent a large message, TI timers have expired before the SG timers. During this second cycle, the bus is operating in a mode which can be seen as a transient mode between the periodic and the aperiodic mode. At the third cycle, the bus is operating in the aperiodic mode. The order in which the terminals send their messages has changed. The terminals now transmit in the order of shortest to longest TG. Terminal 1 is the first to transmit data then terminal 2. To sum up, the order in which the messages are sent is the following (m_it_j denotes the i^{th} message of terminal *j*):

- m_1t_2, m_1t_1, m_2t_2, $\mathbf{m_2t_1}$, $\mathbf{m_3t_1}$, m_3t_2

Between the second bus cycle and the third bus cycle, **rule 4 is violated** since terminal 1 is granted two times consecutively the permission to transmit. Distribution of rights becomes temporarily unfair.

6.5. Availability of the ARINC 629 BP

The BP protocol enforces an access control policy which does not respect the availability constraints we define in section 4 in the following cases:

- Rule 2 is regularly violated. Indeed, within each bus cycle, there are time intervals when nobody has the right to transmit. During these intervals all the terminals are waiting for some timers to expire.
- Rule 1 is violated in one rare collision case which might happen during the initialization phase. Violation of this rule leads to a bus clash and a violation of security rule B.
- If the bus is operating in the periodic mode and the terminals are not transmitting in the order of shortest to longest TG then rule 4 is temporarily violated when the bus switches to the aperiodic mode.

As we can see, only violation of rule 1 leads to a denial of service since only violation of rule 1 leads to a violation of security rule B. Violation of rules 2 and 4 leads to security violations only if we assume that the availability constraints defined in section 4 are *mandatory* and *should* apply to the security policy. In other words if the bus protocol has the *obligation* to enforce an access control policy respecting the availability constraints we define in section 4 then violation of rule 2 or rule 4 leads also to security violations. Violation of rule 2 leads to a denial of service. Let us consider a date which belongs to a time interval where all the terminals are waiting for some timers to expire. Since rule 2 says that there should always be a terminal with the permission to use the bus, we can say that there is a denial of service for one of the terminals at this date. Violation of rule 4 leads both to a denial of service and an access control violation. Indeed, example of section 6.4 shows that terminal 1 uses the bus two times consecutively. There are two security violations when terminal 1 uses the bus for the second time:

- Terminal 2 is victim of a denial of service since, according to rule 4, it should be its turn to use the bus
- Terminal 1 illegitimately uses the bus since, according to rule 4, it should not be its turn to use the bus.

In [1], Ausley and Grigg make a timing analysis of the ARINC 629. Their aim is to evaluate the ability of the ARINC 629 to support real-time applications. They propose an approach to predict delays associated with worst case behavior. It is interesting to note that regarding the BP, some of the worst case delays they identify occur when our availability constraints are violated.

Finally, let us mention that the ARINC 629 CP is an improvement of the ARINC 629 BP. Some of the security violations we have pointed out in this paper never occur with the CP.

7. Conclusion

In this paper, we revisited the ARINC 629 BP from availability constraint standpoint. Our work is a contribution toward the definition of a general availability model. It could be extended in several directions:

- We could extend our approach to the global problem of resource allocation.
- We could use the same approach to analyze other protocols like the ARINC 629 CP or Ethernet (which obeys a CSMA/CD philosophy).
- We could investigate the problem of merging several availability policies. For example we could choose an avionic application and express its availability requirements. Then, we could merge the availability policy of the MAC layer with the availability policy of the application layer and see if conflicts appear. We feel that such approach would allow us to make the same analysis as Grigg and Ausley [1] made but in a more formal and complete way.

References

[1] Neil C. Audsley, Alan Grigg. *Timing analysis of the ARINC 629 data bus for real-time applications.* Microprocesssors and Microsystems 1997.

[2] Airlines Electronic Engineering Comittee. *Multi-Transmitter Data Bus. Part 1 : technical description*, Aeronautical Radio Inc. edition, march 1999

[3] B. Beizer, *The architecture and Engineering of digital computer complexes*, vol. 1, Plenum Press, 1971.

[4] B. Chellas. *Modal Logic: an introduction.* Cambridge University Press.

[5] F. Cuppens and C. Saurel, *Towards a formalization of availability and denial of service.* Information Systems Technology Panel Symposium on Protecting Nato Information Systems in the 21st century. Washington, 1999

[6] G. Juanole and L. Gallon, *Critical Time Distributed Systems: Qualitative and Quantitative Analysis Based on Stochastic Timed Petri Nets*, Formal Description Techniques VIII, Proceedings of the IFIP TC6 Eighth International Conference on Formal Description Techniques, Montreal, Canada, October 1995

[7] L. Gallon, G. Juanole and I. Blum, *Modelling and Analysis of the ARINC specification 629 CP Mac Layer protocol*, IEEE workshop on factory communication systems. Barcelona, 1997

[8] Information Technology Security Evaluation Criteria. June 1991.

[9] G. Juanole and L. Gallon, *Concept of Quantified Abstract Quotient Automaton and its Advantage.* IFIP TC6 WG6.1 Joint International Conference on Formal Description Techniques for Distributed Systems and Communication Protocols (FORTE X), 18-21 November, 1997, Osaka, Japan

[10] J.K. Millen. *A resource allocation model for denial of service.* IEEE Transactions on Security and Privacy, Oakland, USA 1992.

[11] S.M. Sripada. *A metalogic programming approach to reasoning about time in knowledge bases.* Proceedings of the 13th International Joint Conference on Artificial Intelligence.

[12] C. Yu and V. Gligor. *A specification and verification method for prevention of denial of service.* IEEE Transactions on Software Engineering 16(6): 581-592, June 1990.

[13] A. Gabillon, L. Gallon *An Availability Model for Avionic Databuses.* Proceedings of the Workshop on Issues in Security and Petri Nets (WISP). Eindhoven, The Netherlands, 23 June 2003