HARP: A Split Brain Free Protocol for High Availability Implemented in FPGA

Rômerson Deiny Oliveira, Daniel Gomes Mesquita, Pedro Frosi Rosa
Federal University of Uberlândia
Faculty of Computing
Uberlândia, Brazil
romerson@mestrado.ufu.br, {mesquita, frosi}@facom.ufu.br

***Abstract*— High availability is a key requirement for today and future networks. In despite of large investments to achieve high availability, network providers cannot guarantee 100% of availability. The existing protocols have two harmful situations named 'No Brain' and 'Split Brain' conditions, which are algorithmic problems that attack the network availability. This paper aims to show the developing and implementation of a new high availability protocol and how it fixes its predecessors, regarding these conditions.**

Keywords-High Availability; Protocol Design; HARP; VRRP; Split brain.

I. INTRODUCTION

Internet has become one of the most important tools to personal and business transactions in the last years. According to International Telecommunications Union, the total of internet users has increased 389% between 2001 and 2011 [1]. In absolute numbers, this means a rising from 495 million to 2,421 billion of connected users.

Network downtime can bring large financial losses to companies. A published study says that the downtime costs North American businesses collectively $26.5 billion in revenue each year [2]. In order to keep the Internet available as long as possible, universities and manufacturers get started research about high availability.

The HA (High Availability) mechanisms are characterized by using solutions based on hardware redundancy, intelligent software and protocols to identify system failures [3]. This mechanisms work by physical elements seen by the local clients as a single one, named virtual element. The physical elements operate under the master/slave philosophy, such that they have always one master, neither more nor less than one and the others stay available as slaves [4]. The protocol must be able to advertise that there is a master within the group and elect a new one in case of failure on the current master.

The VRRP (Virtual Router Redundancy Protocol) [4] is the *de facto* standard protocol to HA equipments such as routers and switches, mainly in telecommunications networks. Despite all efforts to keep the network available, downtime intervals were noticed onto networks with VRRP in operation [5] and two harmful situations have been identified as responsible for these issues. The first situation is named 'split brain', while the second one is the 'no brain' condition (Subsection II.A).

Hashimoto et al. [5] presented a proposal extending the VRRP protocol to solve the addressed problems. The proposal was modeled in Petri net [6] and due its high level of abstraction, it was necessary refine it and define the five protocol elements [7], intending its implementation.

From this scenario, this paper aims to present a new HA protocol, named HARP (High Availability Router Protocol), free of such phenomena, by defining their elements and show how it fixes its predecessors. The protocol analysis is focused on their proof of concept. We bring data about hardware operation, but a deeper analysis about HARP performance will be presented in future works. The HARP elements, namely assumptions about environmental, services, vocabulary, formatting and procedure rules are presented in Section IV.

The HARP is part of a bigger project that researches Future Internet conducted by MEHAR research group and deals with the HA aspects of the EDOBRA project, that in turn, intends to expand the physical coverage of the experimental installation OFELIA in Brazil [8]. The HARP's first version is addressed here, developed to the current Internet architecture.

Future versions as IPv6 and clean slate compliant are not covered; however, there are researches in progress about these versions and they can be prototyped, once the reconfigurable platform chosen to prototyping. Furthermore, statefull protocols are out of the scope of this work, which is related with stateless protocols like IP protocol.

The remainder of this paper is organized as follows: Section II shows a summarized HA background and related work. In Section III, the method of development to HARP and the set of steps to its validation are presented. Section IV details the protocol analysis, and lastly, Section V shows the conclusion and potential future works.

II. RELATED WORK AND BACKGROUND

This section covers the basics of HA mechanisms and situations which affect the protocols. Following, there is a review of related work, showing the relation with this one.

A. High Availability Overview

HA refers to the network ability to remain available close to 100% of the time, preventing loss of service by reducing or managing failures and minimizing unplanned downtime for the system. HA is obtained by using a virtual address shared by two or more NEs (Network Elements). This address is defined to be the default network gateway for

 ISBN: 978-1-61208-279-0

internal hosts. The virtual router is an abstraction, which consists of one or more routers running a HA protocol. If a failure occurs in the main NE of the group (Master), another HA group's component takes its function, using a virtual IP address corresponding to a virtual MAC (Media Access Control). Because of this, the handover of network elements becomes transparent to local clients, since communication remains on [9].

HA is a subset of fault tolerance since the last one ranges from hardware redundancy up to communications management with a protocol. From the protocol aspects, there are two main causes that lead network to outage: (1) 'no brain' condition in which the master becomes out of service and the infrastructure has no other nodes (slaves) able to assume the master role, i.e., indeed, there is no routing of packets for a while, and (2) 'split brain', which is the situation in when two or more infrastructure nodes assume the master role, i.e., there are two or more routers sending the same packet forward by resulting in replicated requests, which will lead the transport protocol to unpredictable situation) and to the inaccessibility [5].

These conditions can be caused by interface failure (resulting losses or errors in messages) or attack by third parties. This work applies to the first issue.

B. Related Work

The literature provides several publications that can be related to this. The one hand, there are FPGA implementations applied to computer networks and fault tolerance at hardware level. On the other hand, there are investigations concerning protocols at higher levels of abstraction.

Jiang and Prasanna [10] explore the abundant parallelism of FPGA [11] to handle the new generation of packets classification and propose improvements in this regard, in order to make the packets classification and forwarding more scalable free from losses in transmission speeds. This feature is included in the HARP architecture and will be essential when different HARP versions are prototyped in the same core.

Casado *et al.* [12] show that use specialized hardware for packet forwarding is an efficient technique. Also in [12], it is inferred the idea of using more flexible network processors as a way to avoid redoing chips due to changes in protocols or add new features to this hardware, given that cost is a limiting factor.

Straka and Kotasek [13] present a methodology for building fault-tolerant systems based on FPGA. The architectures are based both on technique duplex system as in triple modular redundancy to improve fault detection. For this purpose, the use of testers on-line is shown. It is also shown how the parameters of availability (e.g., mean time between failure and recovery rate) may be affected by operating environment in which the fault tolerant system is implemented. The work is focused on hardware replication techniques and brings methods for building redundant hardware elements, whereas the work shown in this paper handles the communication between the elements.

Lopes Filho [14] examines the idea in that transport layer could be the main problem of high availability protocols, due to using connectionless protocols, resulting in a proposal of a transport layer based on the SCTP protocol. However, the author concluded that the problem lay not in the transport layer and the suspicion came to link layer, due to broadcast messages with false positive for the upper layers.

Hashimoto [15] pointed the errors not detected by the link layer control algorithms as cause for the split brain problem and concluded that would be necessary develop a new HA protocol, or even redesign an existing one. It was shown that the VRRP automaton is incomplete, once it does not take in account errors not detectable by the link layer. The author defends that the problem encountered in VRRP also applies to CARP and HSRP protocols. The author defends that the problem encountered in VRRP also applies to CARP and HSRP protocols. The work presents a Petri net specification, which foresees the loss of advertising messages depending of circumstances in the Link layer.

Pereira Junior [16] discusses the conditions for the concurrent no brain and split brain situations and defines a service specification that composes the HA protocol design. The thesis presents assumptions for an environment where a HA service should operate and how HA protocols could fix arising challenges from there.

This paper stands as a continuation of the research developed in [14][15][16]. These three, in turn, describe a sequence of hypotheses and conclusions about the problems that attack HA protocols and are summarized in [5]. The authors concluded that the problems lie in the no brain and split brain conditions, realized in the VRRP protocol.

III. METHOD

The construction process of the High Availability Router Protocol may be sorted in three stages. At first, we get started by analyzing a proposal of VRRP extension, modeled in high level of abstraction. Specification and development of the HARP elements came following. Lastly, we propose an evaluation system for HA protocols.

A. Characterising the problem

Unavailability situations had occurred onto a network with VRRP in operation [14]. In certain conditions, the finite state machine (FSM) generated by this protocol presents the split brain situation. In other conditions, it presents the brainless one [5].

Two protocols have been largely accepted regarding this matter: VRRP [4] and CARP (Common Address Redundancy Protocol) [17]. The first one has been developed as a proprietary protocol and the second as a protocol developed by the free software community. There also the HSRP (Hot Standby Router Protocol) [18] to provide HA. The finite state machine generated by these protocols presents the same problem in their FSM [5][15].

After finding out the downtime causes and point them out as algorithmic problems that attack the protocol, in [5] was presented an abstract specification, modeled on Petri nets and represented by an automaton to circumvent the cited problem. The automaton in [5] has not been implemented

 ISBN: 978-1-61208-279-0

and it is a description in the same Petri net specification philosophy: in view of the whole HA group's behavior (virtual element).

However, in [5], there is no individual behavior specification to each one physical element, which forms the virtual element. In this way, it would be needed refine the proposal in order to implement it and validate the VRRP extension and because of this, also specify the automaton and the four remaining elements to each protocol instance [7].

This paper brings the behavior rules for a HARP instance in a physical element, from its FSM viewpoint, which was lacking in [5]: items such as parameters and conditions for the master election process, input and output conditions considered in transitions between states for an instance, service specification, and foresee losses of primitives were still necessary for an implementation. This refinement process yielded the HARP.

B. Bulding the High Availability Router Protocol

In this phase, the HARP specification and the creation of its elements were done, as well as its hardware implementation. The five protocol elements are presented with details in Section IV, while the implementation aspects are treated here.

Hardware implementation allows detect some details and correct failures which can remain unnoticed when we are designing protocols at high levels of abstraction. Nevertheless, the production costs to have a specific hardware is too large. In this context, use a reconfigurable hardware could improve the results and reduce costs, so that FPGA is the most indicated for this purpose.

By using FPGA, it was possible to make corrections after each HARP prototyping. This process allows find hardware errors, come back to review the specification and fix them. This iterative sequence makes possible to specify the HARP and prototype a specific hardware till we get a final version. The FPGA usage to implement network algorithms has been widely used in enterprises and universities due to its adaptability and flexibility, reducing production costs as well as time-to-market when compared to an ASIC.

A platform was built in order have a proof of concept of the HARP. For that, a scheme was made with three prototyping boards simulating network elements, connected each other over a star topology. Each NE has an FPGA Cyclone 2 [19] connected as shown in Figure 1.

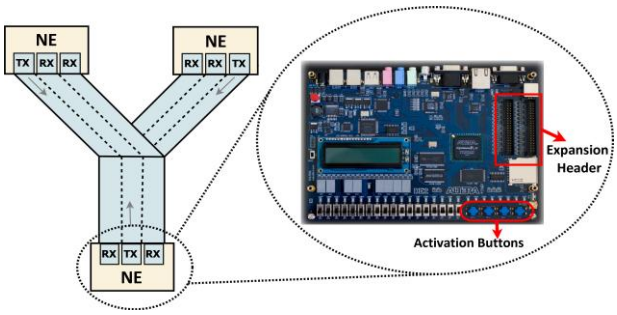

Figure 1. Proof of concept plataform

Each chip relays data to any other one through dedicated connections on a flat cable of 40 paths connected to the expansion header into the board (36 paths for data and 4 paths for supply). Service requests are done by pressing the activation buttons on the board. The channel was divided in three paths with 12 wires each one.

The activation buttons trigger the services shown in Table 1. Each service was tested and after a set of hardware reconfigurations, they worked correctly. Since HARP is based on timeouts (Subsection IV.D), control delays into the states and attempt to all possibility of losses of primitives were the main benefit due to FPGA reconfigurability.

HARP was tested foreseeing the loss of each recognized primitive in its vocabulary. Thus, it possible to guarantee that HARP can deal with all situations of loss of primitives. HARP was tested foreseeing the loss of each recognized primitives in its vocabulary.

Thus, it possible to guarantee that HARP can deal with all situations of lost primitives. This analysis was done concurrently to the implementation. Some examples of loss of messages are given in Subsection IV.D.

As mentioned previously, Figure 1 represents the proof of concept to the protocol elements and it was done successfully, but it did not address the system time to recovery neither bring data about how long each primitive takes to being processed into the HARP core. To treat this, we propose a validation system in the next section, capable to test any HA protocol.

C. Validation System

Figure 2 shows the proposed validation system to test HA protocols. It has a virtual element composed by three (no limited to) network elements (FPGA) [19] connected each other through a shared channel. This channel is connected to PCs responsible for generate information flow.

The system checks HA protocols from the viewpoint of interface and channel failures, since this is the main prompter of no brain and split brain conditions. Another possible cause is the network attack by third parties, but it is out of this work scope.

Each one of NEs in Figure 2 is configured as a System on Chip (SoC) centered on an embedded processor. The HA protocol may be one SoC component, when implementet directly in hardware, or may be a portion of the application running over the SoC.

The SoC should have at least a processor, clock generator and memory core. To test the protocol prototyped in

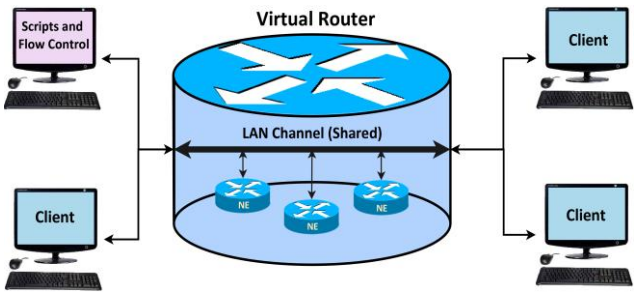

Figure 2. Validation System model proposed

 ISBN: 978-1-61208-279-0

hardware, it has to be included as a SoC module and a hardware abstraction layer must be provided to enable the communication between the protocol and the system interconnection bus. To test the protocol in software, it must be added as a portion of the application source code. The application has to be formed by Ethernet, ARP and IP cores. PCs generate flow information by using ICMP messages.

To test HARP, it was included as a hardware module in a SoC. In normal operation, the master Ethernet channel was intentionally crashed. After that, the Wireshark software [21] shown the message exchanging in the PC Ethernet ports, as well as the embedded application showed the packets in the SoC Ethernet port. Hence, we can control the whole election process and get the advertisement messages from the new master.

Prototyped in an Altera Cyclone II EP2C35F672C6 FPGA, the HARP had used a small portion of available resources. The occupied area reached the mark of 9.2% of the 33216 logical elements (LE). For testing purposes, it was synthesized to an Altera Stratix IV EP1S80F1508I7 FPGA. In this second case, the total used area reached only 1% of the 182400 ALUTs available in the FPGA [20].

The actual HARP performance evaluation is based on the circuitry [20]. The maximum frequency reached by the HARP hardware module is 92.68 MHz. From this point, it was established a ratio between the HARP frequency and the time required to transmit a bit in different channel speeds. This ratio is described in [20] and, summarily, it takes in account the size (bits) of a HARP message and the minimal time required by the channel to transmit this amount of bits.

Actual traffic has been considered in a production scenario to verify the error rate in a local high availability infrastructure. Regarding HARP, only a proof of concept has been constructed, as we don't have how to reproduce real telecommunications network traffic. An ongoing research is preparing an environment to couple HARP with other TCP hardware modules and submit it to a real traffic in a partner telecommunication company.

IV. HARP ELEMENTS

The five protocol elements are presented in this section: assumptions about environmental, services, vocabulary, formatting and procedure rules. To save space, services and vocabulary will be shown together.

A. Assumptions about Environmental

HARP, as a HA protocol, must to operate in each single element into a group of redundant ones. This group forms the virtual element, which is seen by the local clients as a single point of packets forwarding. Particularly, HARP is projected to operate into network layer of the Internet architecture, coupled with Internet Protocol.

B. Services and Vocabulary

In this section, there is a presentation of all services provided by HARP. The primitives are also introduced according to the Request, Confirm, Indication and Response taxonomy. Table 1 summarizes the services set and specifies what messages are exchanging during its execution. Subsection IV.D resumes this topic and provides more details about services execution and the usage of messages.

There are also two messages not cited in the Table 1, given that they are not part of specific services, they are:

- Active Slave Request (ACTS_REQ): sent by the master, in broadcast, intending update its active address table.
- Active Slave Response (ACTS_RESP): sent by a slave, to the master, confirming its activity

C. Message Format

In order to attend the services requirements and also by operate on network layer of Internet architecture, HARP messages have ten fields defined in its first version, as shown in Figure 3.

TABLE 1. HARP SERVICES AND VOCABULARY

Service	Messages	Description
Keep Alive (KA)	*KA Request (KA_REQ)*	Unconfirmed service. It acts as heartbeat and is used by the master to advertise its availability. By monitoring these heartbeats, the slaves determine when a master instance has stopped.
Given Master (GM)	*GM Request (GM_REQ)*	Confirmed service. Used by the master to indicate to the others nodes that it is willing to transfer its role. The target slave address must be included in the active address table.
	GM Response (GM_RESP)	
	GM Failed Request (GMFAIL_REQ)	
	GM Ready Request (GMRDY_REQ)	
Inform Node (INF)	*INF Request (INF_REQ)*	Confirmed service. A service used to indicate that a new node is becoming a member of the HA group.
	INF Response (INF_RESP)	
Remove Node (REM)	*REM Request (REM_REQ)*	Confirmed service. A service used by a slave node, to indicate to all other nodes of the group, that it is leaving the HA group.
	REM Response (REM_RESP)	
Check Brain (CB)	*CB Request (CB_REQ)*	Confirmed service. A service used by a slave node to certify itself that there is no master node in the group HA and avoid the split brain situation during the master election process.
	CB Response Positive (CB_RESP(+))	
	CB Response Negative (CB_RESP(-))	

The messages have 128 bits of header in this version compliant with IPv4. HARP messages are mainly control ones and then have the header bigger than the data field. By the way, DATA field is not used yet, but can be used in the next versions. Each field is explained following.

- DEST_ADDR: target IP address. It may be unicast or broadcast;
- SRC_ADDR: source IP address. It says what NE sent that message;

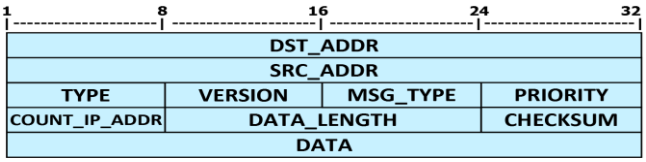

Figure 3. Message Formatting

- TYPE: protocol type, i.e., reserved number to the HARP identification;
- VERSION: HARP version, in this case it is one;
- MSG_TYPE: this field indicates what type of HARP message is being transmitted;
- PRIORITY: NE priority, set up by the network manager;
- COUNT_IP_ADD: in a KA_REQ, indicates the total of slaves in the HA group, can also be used to tell how many addresses are into the data field;
- DATA_LENGTH: it means how many bits are carried in the DATA field;
- CHECKSUM: verification code to detect errors;
- DATA: field designated to carry message parameters when necessary.

The 0x0000 and 0xFFFF addresses are reserved. The first one means that nothing is in the address field, whereas the last one mains broadcast. Next HARP versions can have not identical messages to these ones, in order to serve different network architectures.

D. Procedure Rules

Procedure rules explain the messages exchange and the protocol behavior. Intending to express the HARP behavior unambiguously, represents the HARP FSM. Each service provided by HARP has a couple of automata representing sender and receiver. The FSM in the is the union of each single automaton (according to services) considering sender and receiver instances. The final automaton has eight states. This section will clarify the transitions between states, linking them to each service.

Some states have timeout intervals as input in their transitions. For a while, they are just introduced here, its usage is cited over the text. There are five kinds of timeouts based on a time interval *t*, whose values are: $TO_1 = t$; $TO_2 = t$; $TO_3 = (2+\text{Priority})*t$; $TO_4 = t$ and $TO_5 = t$ (sender) or $2t$ (receiver).

In the case of loss of messages, the HA group can be driven to no brain or split brain conditions. A set of simulations are conducted along this section in order to show how HARP fixes this harmful situations, especially the split brain, which can be triggered by distinct actions.

1) Keep Alive (KA): The node starts at *Idle* state. Based on priority (zero for master and higher to slaves), it will from *Idle* state to *Master* state or to *Slave* state. If a node assumes *Master* state, it sends periodically the KA_REQ by informing its state. Master uses the COUNT_IP_ADDR field to inform how many slaves are in the group.

2) Inform Node (INF): When a NE wants to come into the network, it sends an INF_REQ and waits for a INF_CONF from the master. The master will include its address into the active address table. A master only needs to send INF_REQ when it becomes a slave, but does not need wait for confirmation, since the connection is already established.

3) Remove Node (REM): when a NE sends the REM_REQ it means that it should be removed of the active address table. The requesting node will leave the network only after receiving the service confirmation REM_CONF sent by the master node.

4) Given Master (GM): The master NE sends GM_REQ and transits to *Wait For Given Master Confirm* state (WF_GM CONFIRM). If it receives a GM_CONF, the NE goes to slave state and sends a GMRDY_REQ informing the process conclusion. However, if TO_1 happens, it sends a GMFAIL_REQ informing the receiving error and the transition is to *Master* state again.

At the receiver side, when the slave NE receives a GM_IND, it sends a GM_RESP and goes to *Given Master Accepting* state (GM_ACCEPTING), waiting an interval to ensure there is not NE sending Keep Alive messages. If a GMRDY_IND arrives, the transition is to *Master* state, but if there is a TO_2, a KA_IND or a GMFAIL_IND, the receiver returns to *Slave* state, avoiding the split brain condition.

Some situations are illustrated within a GM context in order to show the HARP ability to recover itself:

- If GM_REQ is lost, the receiver does not receive GM_IND and neither replies with GM_RESP, then the master comes back to Master state at TO_1 and sends a GMFAIL_REQ. Assuming that TO_3 is the interval to realize the master failure and start an election process, TO_1 must be lower than TO_3, ensuring that the will not occur no brain situation;
- In the case when GM_RESP is lost, GMRDY_REQ will not be sent by the sender, so the receiver will not go to *Master* state. The receiver, currently at GM_ACCEPTING state returns to Slave state when TO_2 occur;
- If GMFAIL_REQ is lost, there is no worry, since TO_2 or an incoming KA_IND cover this situation;
- When GMRDY_REQ is lost, the sender now is already a slave and the receiver will also return to slave. This is the worst case within a GM context, but the Check Brain process will be consequently started and fixes this situation.

5) Check Brain (CB): If the slave detects a master absence, a process to elect a new master is started. This is a crucial point of the protocol. The CB_Flag is used to inhibit that the CB process being started by more than one slave at the same time.

If there is a TO_3 without KA_IND, the slave NE sends a CB_REQ to all others slaves by ensuring there is no master and goes to *Wait For Check Brain Confirm* state (WF_CB_CONFIRM). By arriving a CB_CONF(+) or KA_IND or happening a TO_4, it returns to *Slave* state, knowing there is a master onto HA group. But, by arriving a CB_CONF(-), it transits to *Master Election* state to wait for more negatives confirmations, ensure the master failure and changes to *Master* state.

On the receiver's side, when a CB_IND is received, the slave NE goes to *Search Master* state, which verifies the KA_IND receiving. If this reception has been successfully, a CB_RESP(+) is sent and a transition returning to *Slave* state happens, otherwise an CB_RESP(-) is sent and the this NE

 ISBN: 978-1-61208-279-0

also going to *Master Election* state (which forces it to wait for $TO_5 - t$ for sender and $2t$ for receiver, while the sender concludes the CB process).

TO_3 interval is based on node priority in order to be different in each one of them, so minimizing the probability of having more than one slave beginning the election process at the same time. Even so, if two slaves start the process concurrently, the other nodes will respond the first message that arrives to them. O interval $2t$ with no KA_IND is already enough for a slave reply a CB_REQ with CB_RESP(-), to ensure the slave with lowest priority can also respond.

If a master becomes unavailable, a new master is elected and then the previous one returns to master role, the split brain may occur. To fix it, if a master receives a KA_IND, it goes to *Slave* state, if the two masters go to *Slave* state, the Check Brain process is started.

The nodes in a HA group belong to a multicast group. Master sends KA_REQ messages only in this domain. Hence, switches can control the message flow to do not reach end systems. The same configuration happens about the CB_REQ message. It can be transmitted only in a multicast group. The rest of the HARP messages are exchanged in unicast mode and, because of this, there no discussion about to go out on all end systems.

To make HARP compatible with IPv6, a new message format must be done. The vocabulary will be hold as the actual, but the fields have to be revised. By keeping the vocabulary, there is no need to increase the message size, but merely reorganize the fields. Philosophically, it will not be performed any change in the protocol, regarding IPv6 compatibility. On the other hand, the prototyped architecture needs to be reconfigured, intending to keep the exploration of spatial parallelism and the processing frequency.

In this work, we were concerned in demonstrate that there was a misconception in current high availability protocols like VRRP and CARP. HARP is a free split brain protocol, but security issues must be opportunely addressed. For example, we could introduce some confidentiality and authentication to avoid the issue mentioned above. A foreseen assumption is respect introducing the tamper-resistant and tamper-evident environment to the whole HA core circuit environment [22].

E. Final Automaton

Thus, by combining the service automatons, it is possible generating a global automaton which expresses whole protocol instance's behavior. Figure 4 shows all events and actions arising from this. Description of all states in the FSM was seen at Subsection D, as well as all recognized events considered to transitions between states. This FSM brings the HARP behavior to any provided service.

The states S5, S6 and S7 (Figure 4) are the main states to be visited intending keep the HARP free of the Split Brain condition. In case of receiving HARP messages with errors, the election process are invoked and the conversation between the HARP instances must confirm that more than one equipment do not see the master. This avoids that a slave becomes master when it owns the punctual failure.

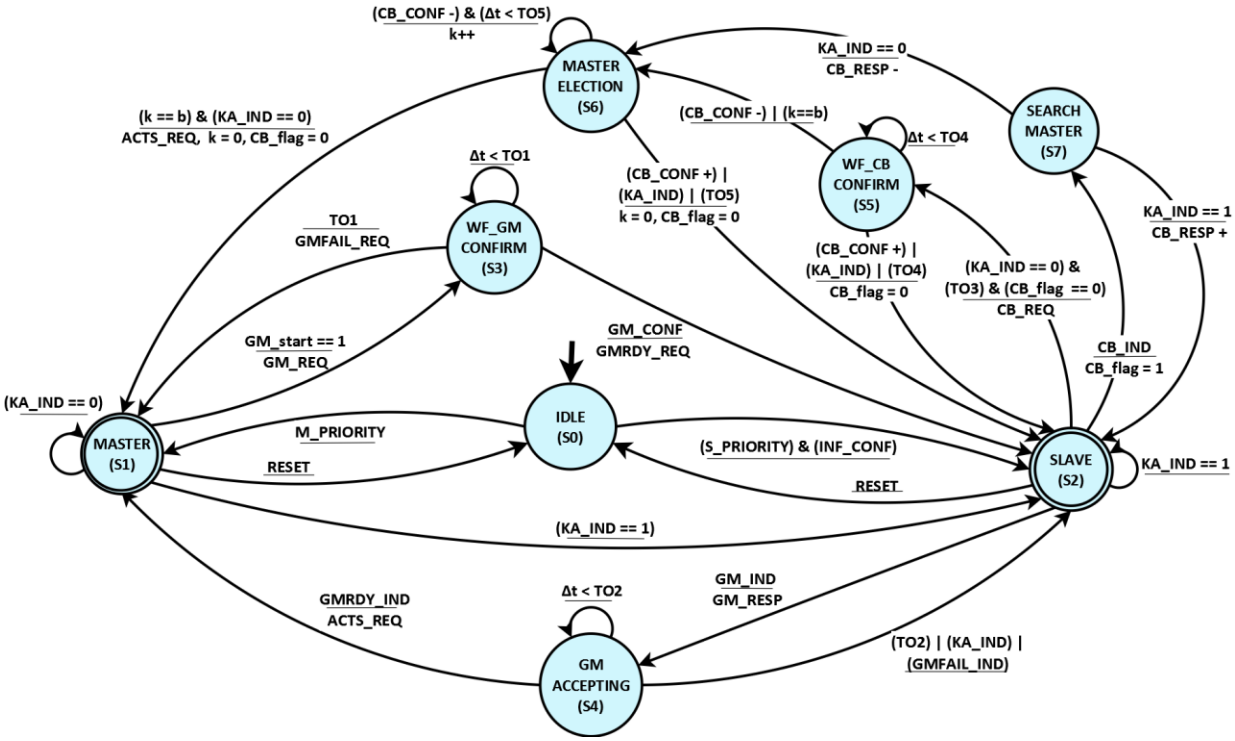

Figure 4. HARP Finite State Machine

The automaton is well formed and keeps the good properties: it is k-limited, guaranteeing the FMS being finite; It does not have states that do not present state thereafter, i.e., free from deadlock; is free of live-lock, i.e., it ensures that there is not a sequence of states that has no successor state; and there is always a sequence of transitions/states that lead to the initial state. So, the modeling was done successfully, guaranteed by complete description of its elements and demonstration of good properties.

V. CONCLUSION AND FUTURE WORKS

The main contribution of this paper is presenting the High Availability Router Protocol and how it fixes the algorithmic problems that attack the existing high availability protocols. It is important to remark that VRRP, a leading protocol of the telecommunications industry, has been the benchmark.

To get HARP, we started from a proposal at high level of abstraction, modeled with Petri net. Thus, we made a complete specification at a lowest level of abstraction to the HARP, without escaping crucial issues to implementation.

The text shows, especially by analyzing the HARP automaton, that HARP maintains the FSM good properties: it is k-limited, free from livelock and deadlock and it is re-initializable from any state.

The two main problems that attack high availability are the no brain and split brain conditions. These problems are fixed by the HARP, considering that the new FSM has enough states and transitions to cover these possibilities. Furthermore, QoS parameters could be considered in order to elect the actual router to support a specific traffic. Apart HARP has a complete FSM, it introduces a service that allows transferring the master role spontaneously.

FPGA implementation allowed executing updates without costs, due to reconfigurability. Thus, it was possible change the hardware even after each prototyping. It should be remembered that performance and timing parameters were not measured till the current version. This phase aimed demonstrate HARP correct functioning. Nevertheless, HARP can operate in Gigabit network with its current 92.68 MHz frequency, reached with a low cost FPGA [20], keeping the protocol convergence time pretty lower than one second.

The time to realize a master failure is actually 3 seconds in the existing HA protocols. HARP brings an adaptable time, depending on the time to transmit a message in the system. HARP default interval to realize a failure is 100 ms. It can be higher if in the system, one message takes more than 100 ms to be transmitted between HARP instances.

One issue can be addressed as disadvantages to use HARP. It is still missing the considerations about state transfer to warranty the failover. In addition, there is no specification about load balancing.

This paper presents results about a new network protocol. Such results were built after an iterative process of tests and prototyping in reconfigurable hardware. From now on, we can foresee the development of a specific hardware based on HARP to deal with high availability network. As potential future works, we suggest (1) implement the validation system, considering TCP/IP protocols in the network and link layers and (2) prototype HARP architecture to be compliant with IPv6 and clean slate approaches.

REFERENCES

[1] International Telecommunications Union Internet users. <http://www.itu.int/en/ITU-D/Statistics/Pages/facts/default.aspx> 05.06.2013

[2] C. A. Technologies. <http://www.ca.com/us/news/Press-Releases/na/2010/North-American-Businesses-Lose-26-5-Billion-Annually.aspx> 05.06.2013

[3] R. Cameron, B. Woodberg, M. K. Madwachar, M. Swarm, N. R. Wyler, M. Alber, and R. Bonnell, Configuring Juniper Networks NetScreen & SSG Firewalls. Rockland: Syngress Publishing, 2007, pp. 587-589

[4] R. Hinden, "Virtual Router Redundancy Protocol". RFC 3768, 2004.

[5] G. T. Hashimoto, E. L. Filho, J. E. Pereira Junior, and P. F. Rosa, "High avaliability: A long-term feature in network elements". Fifth International Conference on Systems and Networks Communications (ICSNC), IEEE, 2010, pp. 201 –206.

[6] C.A. Petri, Kommunikation mit Automaten, Ph.D. Thesis. University of Bonn, Germany, 1962.

[7] G. J. Holzmann, Design and validation of computer protocols, New Jersey: Prentice-Hall, 1991, p. 21.

[8] MEHAR (2012). Extending and deploying ofelia in brazil. <http://www.mehar.facom.ufu.br/projects/ofelia-edobra.dot> 05.06.2013

[9] J. Sonderegger, O. Blomberg, K. Milne S. and Palislamovic, Junos High Avalability: Best Practices for High Network Uptime. USA: O'Reilly Media, 1st edition, 2009

[10] W. Jiang and V. Prasanna, "Scalable packet classification on fpga". IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol 20, no 9, Sept. 2012, pp. 1668 –1680.

[11] S. Brown. and J. Rose, "Fpga and cpld architectures: a tutorial", Design Test of Computers, IEEE, vol.13, no.2, 1996, pp.42,57.

[12] M. Casado, T. Koponen, D. Moon, and S. Shenker, "Rethinking packet forwarding hardware", In ACM Workshops on Hot Topics in Networks, volume VII, 2009, pp. 1–6

[13] M. Straka and Z. Kotasek, "High availability fault tolerant architectures implemented into fpgas". In Digital System Design, Architectures, Methods and Tools, DSD '09. 12th Euromicro Conference, 2009, pp. 108 –115.

[14] E. Lopes Filho, "Arquitetura de alta disponibilidade para firewall e ips baseada em sctp". Master Thesis, Federal University of Uberlândia, Brazil, 2008.

[15] G.T. Hashimoto, "Uma proposta de extensão para um protocolo para arquiteturas de alta disponibilidade" Master Thesis, Federal University of Uberlândia, Brazil, 2009.

[16] J. E. Pereira Júnior, "Especificaçãoo de serviço e suposições sobre o ambiente para um protocolo de alta disponibilidade". Master Thesis, Federal University of Uberlândia, Brazil, 2010

[17] OpenBSD. Pf: Firewall redundancy with carp and pfsync. <http://www.openbsd.org/faq/pf/carp.html> 05.06.2013

[18] T. Li, B. Cole, P. Morton and D. Li,. Cisco Hot Standby Router Protocol. RFC 2281, 1998.

[19] Altera. Development Education Board DE2 <http://www.altera.com/education/univ/materials/boards/de2/unv-de2-board.html> 05.06.2012.

[20] D. G. Mesquita, R. D. Oliveira, and P. F. Rosa, "The High Availability Router Protocol in FPGA," unpublished.

[21] Wireshark. <www.wireshark.org> 05.06.2013

[22] G. E. Suh, D. Clarke, B. Gassend, M. V. Dijk, and Srinivas Devadas. "AEGIS: architecture for tamper-evident and tamper-resistant processing", In Proceedings of the 17th annual international conference on Supercomputing (ICS '03), ACM, 2003, pp. 160-171.

 ISBN: 978-1-61208-279-0