

RTnet, a new approach to in-home real-time multimedia communication

Hans Scholten, Pierre G. Jansen, Ferdy Hanssen, Tjalling Hattink
University of Twente, Dept. of Computer Science, Enschede, the Netherlands
E-mail: {scholten, jansen, hanssen}@cs.utwente.nl


This abstract describes a new approach to token management for in-home digital networks, which we call RTnet. The proposed network offers a guaranteed quality-of-service, based on a distributed token mechanism to schedule communication streams. RTnet is based on the CSMA/CD Ethernet protocol [1]. In effect this means that RTnet can be built on any existing Ethernet specification and regular Ethernet hardware can be used without modification. CSMA/CD uses an exponential back-off mechanism to resolve collisions, which makes the network non-deterministic, and thus non-real-time. To make it deterministic, a token-based protocol is deployed to avoid collisions, just like Rether [2], developed at SUNY Stony Brook. Rether distributes its token among the nodes in a simple, static Round-Robin manner. RTnet, however, uses a more sophisticated algorithm, where the token is allocated to nodes according to their bandwidth demands. A preemptive earliest deadline first (EDF) scheduler is used to determine the route a token follows in the network. This type of scheduler has the advantage over other schedulers that it can achieve a 100% utilization. However, in case of overload its performance will degrade dramatically. Whenever a node has the token (the active node) it may use the network for some time: the *token holding time* (THT). The THT is determined by the scheduler and is likely to be different for each node. Typically the node will send one frame of a periodic multimedia stream. Note that our network works with constant-bit-rate streams, but variable-bit-rate streams can be mapped using several techniques [3]. The EDF scheduler is distributed over the nodes and the token is the place where the schedule is kept; nodes will keep backups of the schedule though. If a node has the token and it wants to add or remove a stream, it calculates a new schedule and acts upon it. Before a new stream may be added the node does an EDF feasibility test to determine if the newly added real-time stream will meet its deadlines without making other streams miss theirs. The EDF feasibility test is simple [4]: the total of bandwidth utilizations by all streams may not exceed 100%. However, only 80% of network bandwidth is dedicated to real-time (multimedia) communications, the rest is used for non-real-time purposes. Actually, the maximum bandwidth is slightly less because of the token transmission and Ethernet packet overhead.

When the scheduler at the active node decides that another node must become active it stores the global schedule information in the token and sends the whole package to the new node. This is a critical action and it should never occur that the token becomes lost, or worse, duplicated. When a token is lost the schedule is lost too and the network stalls. Duplicated tokens mean that more than one node will make schedules and collisions will occur, causing deadline misses and non-deterministic behaviour. To prevent this the concept of a monitor is introduced. When the active node relinquishes the token, this node becomes the monitor for the new active node. Monitoring is a three step process: (1) the monitor sends the token to the new active node. If the token is garbled the active node will request a new token. This happens only once. If a second token is garbled too, the active node stays inactive, while the monitor times out and recovers the token. (2) The active node is now in the 'transmission state' for the duration of the THT. While the active node transmits, the monitor waits for a reply from the active node. It sets a timer for the duration of the THT. (3) At the end of the THT the active node must send a reply to the monitor signalling that it is still alive and sends the token to a new active node. The old monitor now ends its activity, the old active node becomes monitor, and the new active node begins transmitting. Then the process starts all over. Many things can go wrong. Detectable token loss situations are: token does not arrive at new active node, reply from active node is lost, active node dies before sending a reply, and the monitor dies. Undetectable token loss occurs when: the active node dies after sending a reply, and the monitor and active node die simultaneously. Token duplication is difficult to detect, but there are some hints the network will give: a token arrives at a node that already holds a token, and streams miss their deadlines because nodes cannot resolve bandwidth requirements. If a node detects this, it will delete the token. The network will probably end up without a token and initiate a reset.


To validate correctness, robustness and usability of RTnet a simulation of the network protocols is made, and a prototype is realized for demonstration and calibration of the simulation. The prototype is based on the Linux operating system and Ethernet hardware. From the simulation we found the following general characteristics: (1) the total bandwidth utilization is very close to the set 80%, (2) the smaller the stream periods, the higher the overhead, and (3) the higher the offered load, the higher the overhead. The simulation was carried out using a network of 5 nodes, where we tested with combinations of 1, 3, 6, 10, and 25 streams, comprising offered total loads of 10%, 20%, 50%, 70%, and 80%, thus creating a total of 125 tests. Some figures calculated from the simulation are: average overhead per stream related to total bandwidth: 0.54%; average worst-case overhead per stream related to total bandwidth: 0.78%; average overhead per stream related to effective stream utilization: 1.33%; and average worst-case overhead per stream related to effective stream utilization: 1.95%. The feasibility test always computes the worst-case overhead and uses that to check if the total utilization does not exceed the maximum real-time utilization. This is checked by the simulation, where worst-case overhead is higher than real overhead in every case.

References


- [1] Institute of Electrical and Electronics Engineers, *IEEE Standard for Information Technology – Local and Metropolitan Area Networks – Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) access method and physical layer specifications*, ISO/IEC 8802-3:2000 ed., 2000. IEEE Std. 802.3.
- [2] "Rether web site." <http://www.ecsl.cs.sunysb.edu/rether/>.
- [3] S. V. Anastasiadis, K. C. Sevcik, and M. Stumm, "Server-based smoothing of variable bit-rate streams," in *Proceedings 9th ACM Multimedia Conference*, (Ottawa, Canada), pp. 147–158, October 2001.
- [4] C. L. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in a hard real-time environment," *Journal of the ACM*, vol. 20, no. 1, pp. 40–61, 1973.


 University of Twente
 department of
 computer science


RTnet, a new approach to in-home real-time multimedia communication



Hans Scholten, Pierre G. Jansen,
 Ferdy Hanssen, Tjalling Hattink
{scholten,jansen,hanssen}@cs.utwente.nl



1



 University of Twente
 department of
 computer science

Existing networks

- Ethernet
- Token bus/ring networks
- ATM
- Firewire (IEEE1394, iLink)
- Wireless (IEEE802.11b, Bluetooth)
- Rether
- ...

These solutions do not provide enough QoS or are too expensive


4


 University of Twente
 department of
 computer science

Summary

- Introduction
- Existing networks
- RTnet
- Simulation
 - Token
 - Real-time streams
 - Scheduling
 - Feasibility
 - Network management
- Conclusions

2


 University of Twente
 department of
 computer science

Ethernet


Ethernet is a CSMA/CD protocol:

- All hosts may transmit packets to any other host over Ethernet
- When two hosts transmit at the same time a *collision* occurs
- Collisions are resolved by the *back-off* algorithm:
 - Both hosts wait for random period, then they retransmit the packet
 - When the packets collide again, repeat the back-off algorithm

Ethernet collisions result in *non-deterministic* behaviour!

- To provide QoS, deterministic behaviour is required
- Avoid collisions by allowing only one transmitting host at a time
- To achieve this a *token* is used

5



 University of Twente
 department of
 computer science

Introduction

At Home Anywhere (@HA)

- Integration of home appliances and devices into one coherent architecture:
 - one common integrated network for entertainment, command & control, and information, supporting both real-time and non-real-time traffic with a high degree of robustness
 - integration of inexpensive, small (resource-lean) appliances
- Network requirements
 - Provide enough *Quality of Service* (QoS) for handling high bandwidth multimedia streams
 - Provide best effort services to handle normal traffic
 - Based on cheap hardware

3


 University of Twente
 department of
 computer science

A new approach

- Ethernet as base protocol
- Build new RTnet protocol on top providing QoS

Network layer	IP		IP
Data link layer	RTnet		RTnet
Physical layer	Coax/UTP		

6

University of Twente
department of computer science

Token

- The network holds only one token
- Only the host that holds the token may transmit
- The token contains global network state information
- Only the host holding the token can change global state
- Thus, all hosts should receive the token on a regular basis

7

University of Twente
department of computer science

Earliest Deadline First

Pre-emptive EDF currently used as scheduler

- EDF is a simple, fast algorithm
- EDF can achieve 100% utilization
- The feasibility test is simple
- Not good in overload situations

How does EDF work?

- The *ready* RT stream with *lowest absolute deadline* obtains the token
- Host may hold the token until:
 - The RT stream consumed all its transmission time (for 1 message)
 - Another stream with a shorter absolute deadline *pre-empts*

10

University of Twente
department of computer science

Real-time streams

If a host wants to transmit real-time traffic to another host it creates a *real-time stream*

- Relevant RT stream parameters:
 - Maximum "network execution" time C_i
 - Period T_i

During one period the RT stream delivers a *message* to other host before the period is expired

A message is a set of network packets. It usually contains one frame of video and/or audio

8

University of Twente
department of computer science

Feasibility test

Feasibility test verifies a set of streams. It checks if the set is schedulable

Network utilization per stream: $\frac{C_i}{T_i}$

For EDF a set of streams is feasible when:

$$U = \sum_{i=1}^n \frac{C_i}{T_i} \leq 1$$

where C_i is the network execution time of stream i and T_i is the period of stream i and n is the number of streams

11

University of Twente
department of computer science

Scheduling

- Only one host can transmit a stream at a time
- To select the active stream we need a *scheduler*

The scheduler determines the *token holder* and the *token holding time*

RTnet can support standard schedulers:

- Round Robin
- Rate Monotonic
- Deadline Monotonic
- Earliest Deadline First (EDF)

9

University of Twente
department of computer science

Non-real-time traffic

- When no RT stream is ready the network is idle
- This idle time is used to send non real-time traffic or perform network management

A *Round-Robin* scheduler is used

- All hosts are visited in a fixed order
- Every host holds the token for the same period
- The host holding the token may transmit non RT traffic
- When a RT stream becomes ready the RT scheduler is used
- Maximum RT utilization is limited to give non RT traffic room

12

Network management

Network management is *decentralized*

- There is no 'master' node used for management
- The node holding the token may perform management

When a node holds the token it may:

- Add new nodes by broadcasting an announcement
- Remove itself
- Adding new RT streams
- Removing RT streams

The network is initialized by the first node that generates a token

13

Simulation (1)

RTnet is implemented using the *OMNeT++* simulator

- Object oriented
- Event driven
- Provides detailed visual feedback
- Perfect for simulating network protocols

Three network layers are built in simulator:

- Ethernet layer
- RTnet layer
- Application layer

16

Monitoring

Token loss:

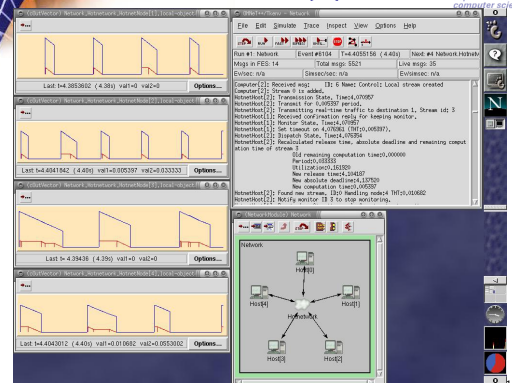
- Packet with token is lost in network
- The host owning the token dies (disconnection)

To prevent token loss a *monitor* is used

- Besides a node holding the token another node is monitoring
- The monitor keeps an eye on the token holder
- Token holder should reply before monitor times out
- When monitor times out it polls token holder if it is still alive
- If poll fails the monitor assumes that token holder is dead

14

Simulation (2)



17

Token loss

Detectable token loss situations

- Token does not arrive at new node
- Reply from token holder is lost
- Token holder dies before sending reply
- Monitor dies

Undetectable token loss situations

- Token holder dies after sending reply
- Both nodes (token holder and monitor) die simultaneously

Token duplication situation

- Both reply and poll are lost but token holder still lives

15

Conclusions

- RTnet is first network that uses pre-emptive EDF scheduling
- It does not require new expensive hardware
- Decentralized management and monitoring seems practical
- Simulation shows that protocol works

Is RTnet hard real-time?

- YES:** It uses hard real-time scheduling and feasibility test

Is RTnet fail-safe?

- NO:** Token loss in network may cause non-deterministic delays

18