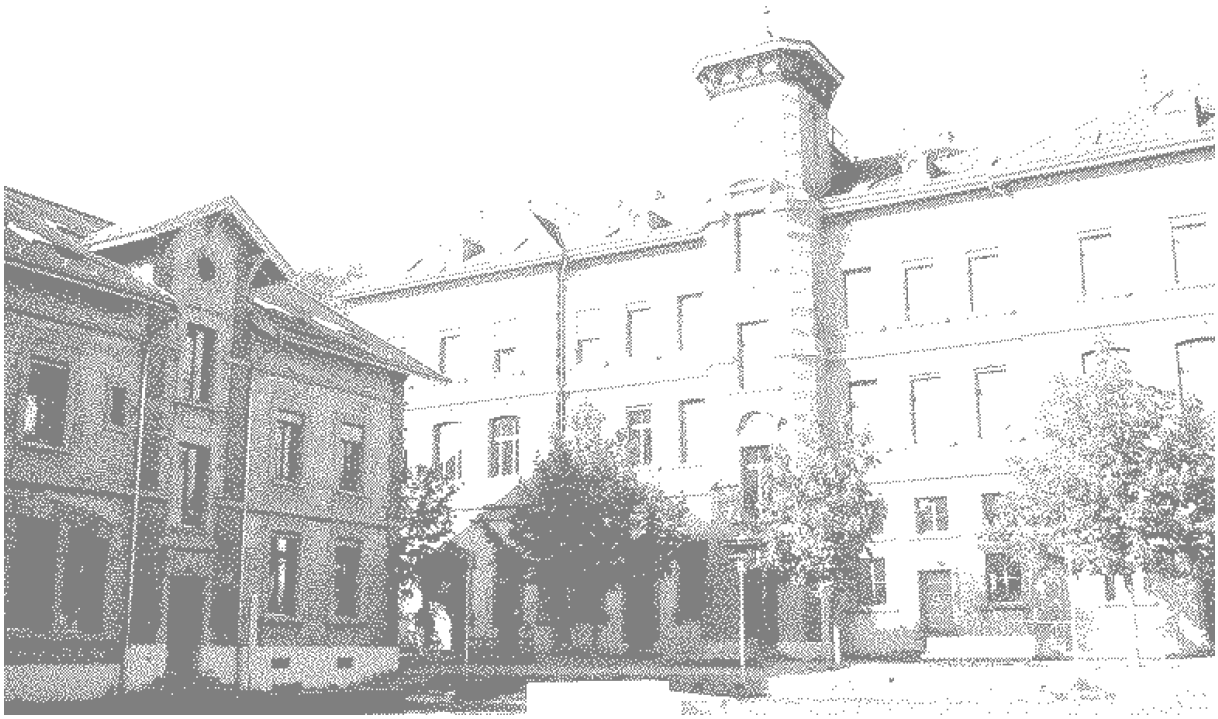


BeNeFri Summer School 2008 on Dependable Systems

Marc Brogle, Torsten Braun (eds.)

Technical Report IAM-08-003, November 18, 2008

Institut für Informatik und angewandte Mathematik, www.iam.unibe.ch

BeNeFri Summer School 2008 on Dependable Systems

Marc Brogle, Dragan Milic, Markus Anwander, Gerald Wagenknecht, Markus Wälchli, Torsten Braun, Raphael Kummer, Markus Wulff, Ronny Standtke, Heiko Sturzrehm, Etienne Riviere, Pascal Felber, Stephan Krenn, Christoph Ehret, Carolin Latze, Philipp Hurni, Thomas Staub

Technical Report IAM-08-003, November 18, 2008

CR Categories and Subject Descriptors:
C.2.1 [Computer-Communication Networks]: Network Architecture and Design; C.2.2 [Computer-Communication Networks]: Network Protocols; C.2.3 [Computer-Communication Networks]: Network Operations; C.2.4 [Computer-Communication Networks]: Distributed Systems

General Terms:
Design, Management, Measurement, Performance, Reliability, Security, Verification

Additional Key Words:
peer-to-peer, wireless mesh networks, wireless sensor networks, overlay multicast, network security, anonymity, transactions, automata theory

Institut für Informatik und angewandte Mathematik, Universität Bern

Abstract

The BeNeFri Summer School 2008 on Dependable Systems in Quarten from June 23-26 was organized by four research groups from the Universities of Bern, Fribourg and Neuchâtel. The University of Bern was represented by the research group “Computer Networks and Distributed System” of the Institute of Computer Science and Applied Mathematics at the University of Bern, headed by Prof. Torsten Braun. The research group “Telecommunications, Networks, Security” of the Department of Computer Science, headed by Prof. Ulrich Ultes-Nitsche, represented the University of Fribourg. The University of Neuchâtel was represented by two research groups from the “computer science department (IIUN)”, namely “distributed systems” headed by Prof. Peter Kropf and “dependable systems and networks” headed by Prof. Pascal Felber. The focus of this retreat was to present and discuss recent research results and currently ongoing research activities of the members of the four research groups. The research group members gave twenty-two presentations, form the areas of overlay networks, wireless mesh and sensor networks, network security, distributed systems, anonymity, transactions and automata theory. Extensive time (typically 45 minutes per talk) has been allocated to allow detailed presentations and discussions. This technical report summarizes the various talks and describes mostly unpublished work that is currently in progress.

Contents

1 Reviving IP Multicast Using QoS Enhanced Overlay Networks 1
- 1.1 Introduction . . . 1
- 1.2 Bridging IP Multicast & Overlay Multicast . . . 1
- 1.3 Quality of Service for Overlay Multicast . . . 1
- 1.4 MC-FTP: Multicast File-Transfer Protocol . . . 3
- 1.5 Conclusion and Outlook . . . 4

2 NetICE9 9
- 2.1 Introduction . . . 9
- 2.2 Fisheye Overlay . . . 9
- 2.3 NetICE9 . . . 11
- 2.4 Conclusion and Outlook . . . 12

3 Aspects of Energy-efficient Management of Heterogeneous Wireless Sensor Networks WSN 15
- 3.1 Introduction . . . 15
- 3.2 TCP Support for Sensor Nodes (TSS) . . . 15
- 3.3 Energy-efficient beacon-less 802.15.4 MAC protocol . . . 17
- 3.4 Reliable Multicast in Wireless Sensor Networks . . . 20
- 3.5 Conclusion and Outlook . . . 23

4 Fuzzy Classifier Design for Efficient and Robust Event Classification in Wireless Sensor Networks 25
- 4.1 Introduction . . . 25
- 4.2 Discussion . . . 26
- 4.3 Conclusion and Outlook . . . 29

5 Cognitive Networks 31
- 5.1 Introduction . . . 31
- 5.2 Cognitive Radio . . . 31
- 5.3 Cognitive Networks . . . 32
  - 5.3.1 Principles . . . 32
  - 5.3.2 Architecture . . . 33
  - 5.3.3 Applications . . . 34
- 5.4 Conclusion and Outlook . . . 34

6 Multicasting Solution for Mobile Ad-Hoc networks 37
- 6.1 Introduction . . . 37
- 6.2 The Tree Construction Algorithm . . . 38

6.2.1 The Connection Algorithm . . . 38
6.2.2 Finding More Potential Parents . . . 39
6.2.3 Finding Better Parents . . . 39
6.2.4 Removing Redundant Parents . . . 40
6.2.5 Multicast Message Gathering . . . 40
6.3 Conclusion . . . 41

7 Wireless Communication in Road Traffic Control 43
7.1 Introduction . . . 43
7.2 Road Traffic Control . . . 43
7.3 Car2Car Communication Consortium . . . 44
7.4 Other projects . . . 45
7.5 IEEE 802.11 for car-to-car communication . . . 46
7.6 Privacy and Security issues in VANETs . . . 46
7.7 Conclusion and Outlook . . . 46

8 Pretty Good Anonymity 49
8.1 Introduction . . . 49
8.2 Threat models . . . 49
8.2.1 Remote attacker . . . 49
8.2.2 Local attacker . . . 49
8.2.3 Global attacker . . . 49
8.2.4 Third party attacker . . . 51
8.3 Introduction to Project PGA . . . 52

9 Applicating Software Transactional Memory on Parallel Event Processing 55
9.1 Introduction . . . 55
9.2 Enhanced ESP Components . . . 56
9.3 Conclusion and Outlook . . . 58

10 Developing and Deploying Large-scale Distributed Applications using SPLAY 61
10.1 Introduction . . . 61
10.2 SPLAY architecture . . . 62
10.2.1 Controlling deployments . . . 64

11 Zero-Knowledge Proofs Of Knowledge 67
11.1 Introduction . . . 67
11.2 Theoretical background . . . 67
11.3 Conclusion and future work . . . 69

12 Achieving Fairness and Efficiency in RSS Publish/Subscribe Systems by Leveraging Physical and Semantical Proximities 73

13 Immune System Based Intrusion Detection 75
13.1 Introduction 75
13.2 Discussion 75
13.2.1 Intrusion Detection Systems 75
13.2.2 Immune System features 76
13.2.3 How the immune system based IDS should be . . . 78
13.3 Conclusion 79

14 EAP-TPM - Secure User Authentication in 802.11 Based Networks 81
14.1 Introduction 81
14.2 Discussion 82
14.2.1 The TPM 82
14.2.2 EAP-TPM - First Version 82
14.2.3 EAP-TPM - Second Version 83
14.3 Conclusion and Outlook 84

15 Traffic Adaptivity in Energy-Efficient MAC Protocols 87
15.1 Introduction 87
15.2 Discussion 87
15.2.1 WiseMAC 87
15.2.2 Improving the WiseMAC More Bit 88
15.2.3 Simulation Evaluation 89
15.2.4 Evaluation on Embedded Sensor Boards 90
15.3 Conclusions 91

16 Real-time Communication in Wireless Mesh Networks 93
16.1 Introduction 93
16.2 Adaptive Transport Over Multipaths (ATOM) 94
16.2.1 ATOM Architecture 94
16.2.2 Session Establishment and Release 97
16.2.3 Session Reconfiguration: ATOM Adaptation Process 98
16.3 Conclusion and Outlook 98

1 Reviving IP Multicast Using QoS Enhanced Overlay Networks

Marc Brogle, University of Bern
brogle@iam.unibe.ch

1.1 Introduction

Multicast is an efficient way to disseminate data from a sender to multiple receiver. IP Multicast [1, 2] is available since a while but unfortunately is still not widely deployed in the Internet today. It is very efficient way of dissemination because the routers replicate packets, therefore a sender only transmits the data once for multiple receivers. To support IP Multicast in the Internet, some router configuration and agreements between providers are required, which are some of the reasons why it is not available to the end user through the Internet today.

1.2 Bridging IP Multicast & Overlay Multicast

Application Level Multicast (ALM) [3] does multicasting in end systems. The multicast trees are directly built between end systems and not between routers as it is the case for IP Multicast. ALM uses an Peer-to-Peer (P2P) [4] Overlay Network on-top of a “real” network to distribute the data among the recipients. We developed the Multicast Middleware [5], which enables IP Multicast (with QoS support) on end systems across the Internet using only unicast communication. It uses a virtual network interface (TAP) [6] to capture IP Multicast traffic on end systems and transports the multicast payload over a P2P network (Scribe/Pastry) [7, 8]. This mechanism as shown in Figure 1.1 is completely transparent to the host system applications and requires no additional infrastructure support (routers, configuration, agreements, etc).

1.3 Quality of Service for Overlay Multicast

Quality of Service (QoS) enabled multicast trees need to be of a certain structure. A QoS metric (QoS class) could be for example bandwidth (3/4/5/6 Mbps). Nodes with higher QoS requirements have to be closer to the root node, which has the highest QoS in the Overlay. Each path from

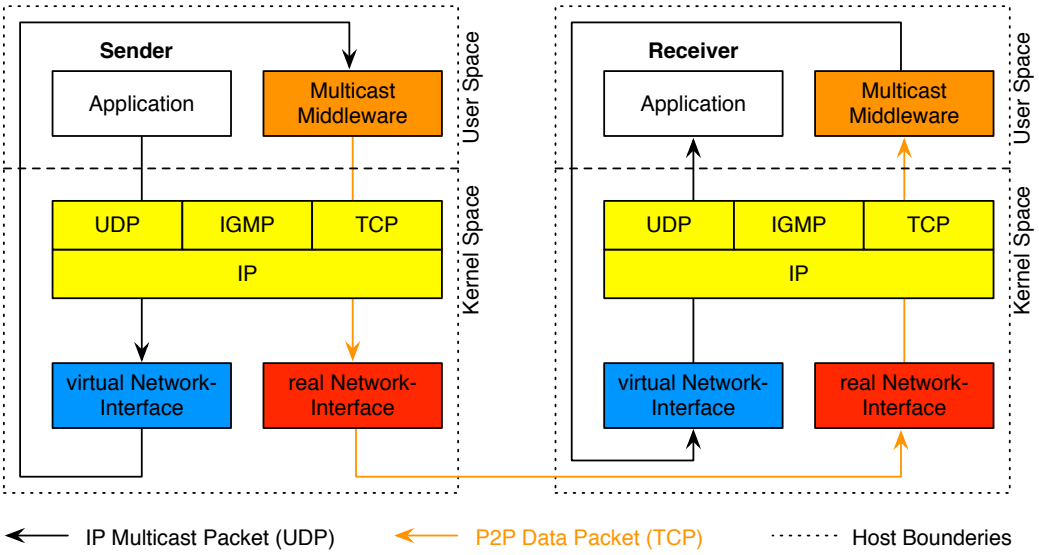


Figure 1.1: Multicast Middleware Packet Flow

the root to any leaf has monotonically decreasing QoS requirements as shown in Figure 1.2. Only QoS metrics with natural order allowed allowed. These can be for example jitter, bandwidth, CPU, etc. Also a combination of such parameters is possible, but the natural order has to be maintained.

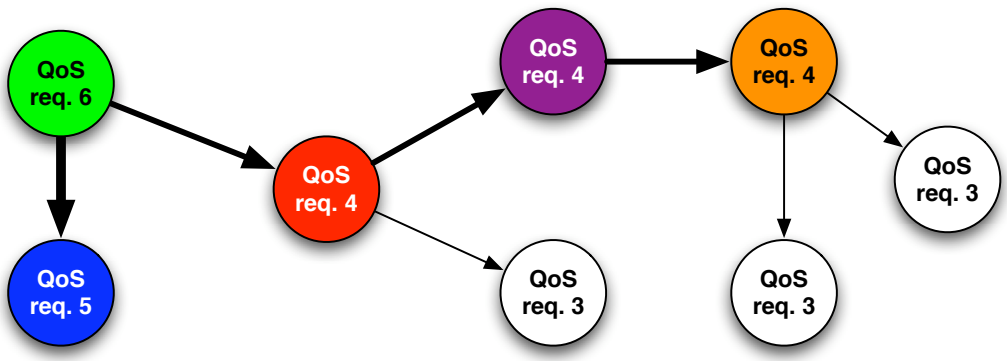


Figure 1.2: Structure of a QoS aware Multicast Tree

We have analyzed different P2P/ALM architectures and defined the OM-QoS framework with protocol dependent as well as protocol independent

mechanisms as described in [9] to make the tree building in those overlay networks QoS aware. For Scribe/Pastry, Bayeux/Tapestry [10, 11] and Chord [12] we introduced a QoS-aware ID assignment method. We also modified the NICE [13] Cluster leader determination mechanisms, using the QoS classes instead of delay characteristics. Figure 1.3 shows these QoS mechanisms for Scribe/Pastry and NICE.

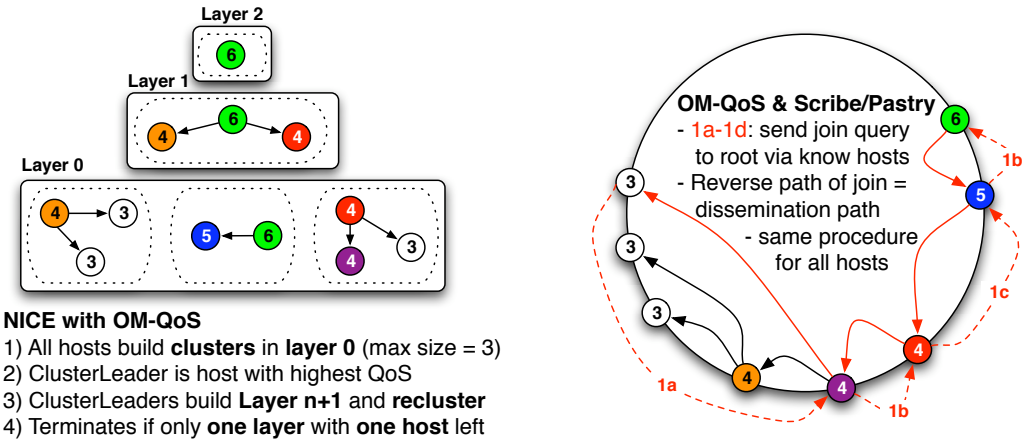


Figure 1.3: QoS Mechanisms for Pastry/Scribe and NICE

Other protocols, such as CAN [14] can not easily be adapted as the previous examples. Therefore we also introduced a protocol independent solution, which should work with almost any structured P2P overlay network. We introduce one dedicated P2P "layer" (a separate Overlay Network) per active QoS class. The sender forwards the data in its own Overlay Network and also to upper / lower layers. Receivers getting data from other layers forward this data in their own layer and also the next layer in the same direction. This behavior is depicted in Figure 1.4.

1.4 MC-FTP: Multicast File-Transfer Protocol

Multicast dissemination is more efficient than unicast, therefore file-sharing or file-exchange should be using multicast. We propose a protocol that aims to be comparable with Bittorrent regarding efficiency.
Basically the protocol works as follows. Files are divided into chunks, which are then sent on separate multicast groups. One dedicated multicast group is used for communication and information exchange, which is called the "File Management Group" (FMG). Our protocol either uses IP

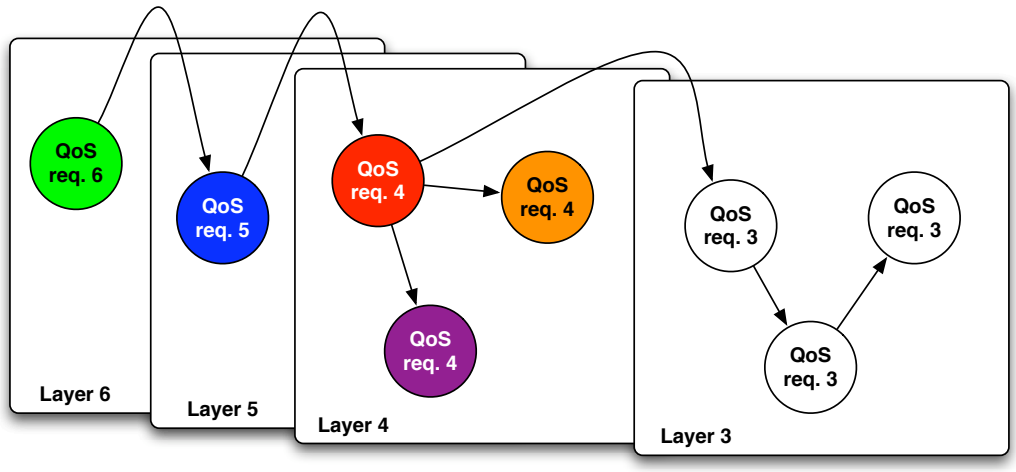


Figure 1.4: Protocol independent Approach using Layers

Multicast or Overlay Multicast for data dissemination.
The “File-Leader” based approach has a File-Leader (FL) that collects what hosts need / have and who tells hosts what to send at which rate to which multicast group. One multicast group, the FMG is used for management. Hosts send “Status Messages” to the FL for updates of their status. The FL send “Keep-Alive Messages” (KAM) to the hosts for instructions & information about who sends what where and how. The information flow using FL, FMG and KAM is shown in Figure 1.5. The FL has an omniscient view of all collaborating hosts.
The “File-Leader-Less” approach is completely decentralized, more robust to failures. It is based on announcements from hosts about what chunks they need and which chunks they send according to what they hear from other hosts. This corresponds to a marketplace strategy.
After the hosts know what will be sent (rate & group), the senders start sending to the corresponding groups and the receivers join the groups, in which they are interested in as shown in Figure 1.6.

1.5 Conclusion and Outlook

Bridging IP Multicast and Overlay Multicast offers using IP Multicast through the Internet for end users. Using OM-QoS, we can introduce Quality of Service for Overlay Multicast, which is applicable to different P2P Overlay Multicast protocols. The protocol dependent approach works

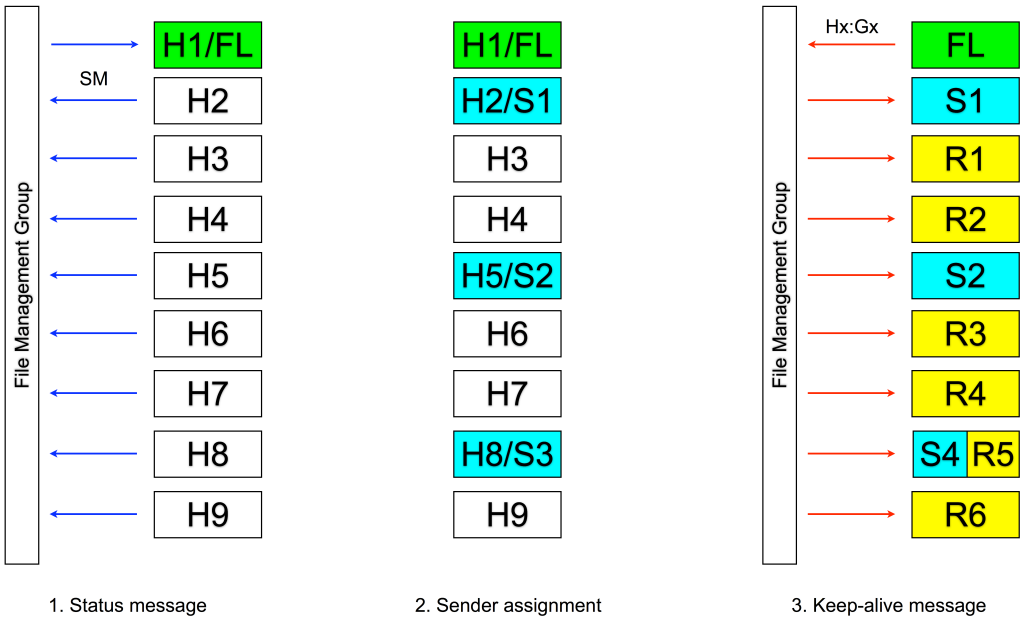


Figure 1.5: Information Flow between File-Leader and Hosts

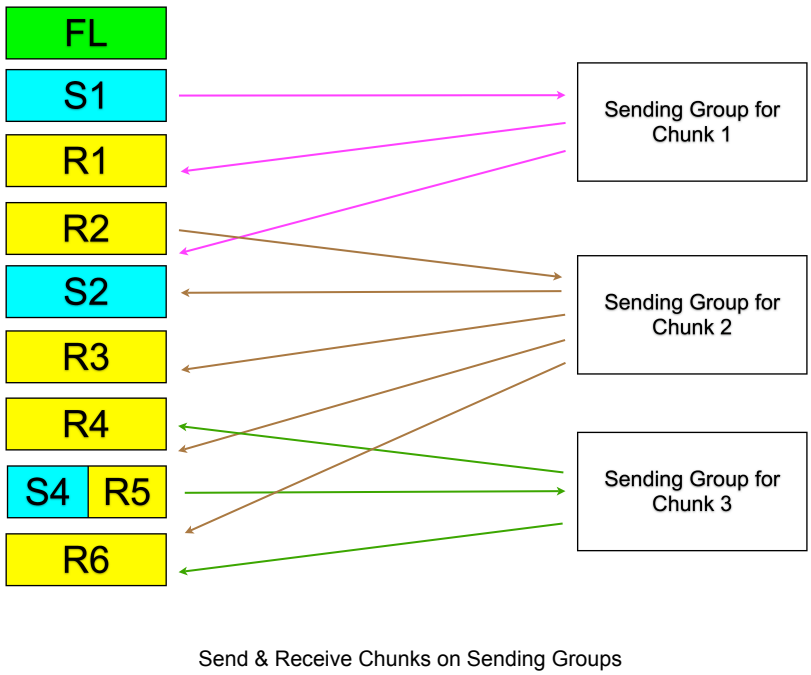


Figure 1.6: Data Exchange between Senders and Receivers

for specific P2P protocols while the protocol independent approach acts as general solution. Open issues are dynamic environments and delay-support, which will be further investigated in the future. MC-FTP (Multicast File-Transfer Protocol) is an efficient protocol for file-sharing and file-exchange using multicast. It works with IP Multicast and Overlay-Multicast. First simulations show that MC-FTP is potentially better than Bittorrent.

References

[1] S. Deering, "Host extensions for IP multicasting." RFC1112, August 1989.

[2] B. Cain, S. Deering, I. Kouvelas, B. Fenner, and A. Thyagarajan, "Internet Group Management Protocol, Version 3." RFC3376, October 2002.

[3] M. Hosseini, D. T. Ahmed, S. Shirmohammadi, and N. D. Georganas, "A survey of application-layer multicast protocols," *Communications Surveys & Tutorials, IEEE*, vol. 9, no. 3, pp. 58–74, 2007.

[4] K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim, "A survey and comparison of peer-to-peer overlay network schemes," *Communications Surveys & Tutorials, IEEE*, pp. 72–93, 2005.

[5] D. Milic, M. Brogle, and T. Braun, "Video broadcasting using overlay multicast," in *ISM '05: Proceedings of the Seventh IEEE International Symposium on Multimedia*, (Washington, DC, USA), pp. 515–522, IEEE Computer Society, 2005.

[6] "Universal TUN/TAP driver website." Available online: `http://vtun.sourceforge.net/tun/`, 2008.

[7] A. Rowstron and P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems," in *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, pp. 329–350, Nov. 2001.

[8] A. Rowstron, A.-M. Kermarrec, M. Castro, and P. Druschel, "Scribe: The design of a large-scale event notification infrastructure," in *Networked Group Communication, Third International COST264 Workshop (NGC'2001)*, vol. 2233 of *Lecture Notes in Computer Science*, pp. 30–43, Nov. 2001.

[9] M. Brogle, D. Milic, and T. Braun, "Quality of service for peer-to-peer based networked virtual environments," in *P2P-NVE 2008: Second International Workshop on Peer-to-Peer Network Virtual Environments*, (Melbourne, Victoria, Australia), December 8–10 2008.

[10] B. Y. Zhao, J. D. Kubiatowicz, and A. D. Joseph, "Tapestry: An infrastructure for fault-tolerant wide-area location and," tech. rep., 2001.

[11] S. Q. Zhuang, B. Y. Zhao, A. D. Joseph, R. H. Katz, and J. D. Kubiatowicz, "Bayeux: an architecture for scalable and fault-tolerant wide-area data dissemination," in *NOSSDAV '01: Proceedings of the 11th international workshop on Network and operating systems support for digital audio and video*, (New York, NY, USA), pp. 11–20, ACM Press, 2001.

[12] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, F. M. Kaashoek, F. Dabek, and H. Balakrishnan, "Chord: a scalable peer-to-peer lookup protocol for internet applications," *IEEE/ACM Trans. Netw.*, vol. 11, pp. 17–32, February 2003.

[13] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable application layer multicast," in *SIGCOMM '02: Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications*, vol. 32, (New York, NY, USA), pp. 205–217, ACM Press, October 2002.

[14] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker, "A scalable content-addressable network," in *SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, vol. 31, pp. 161–172, ACM Press, October 2001.

2 NetICE9

Dragan Milic, University of Bern
milic@iam.unibe.ch

2.1 Introduction

Round trip time (RTT) is one of the most import QoS parameters in the computer networks. This mainly due to its limiting role on the available bandwidth of TCP connections [1]. This is the main reason, why most of topology overlay network building approaches focus on optimizing RTTs [2, 3, 4]. In this work, we present a method for building an overlay network that uses RTT information to incrementally converge towards a better overlay network. We also present a method to use such an overlay network to embed hosts into a virtual space in a fully distributed manner with stable positions.

2.2 Fisheye Overlay

The two most important properties of the fisheye view are the decreasing detail density with increasing distance to the center and the geographical diversity. Divergent the other approaches based on some kind of partitioning of the space and limiting the number of neighbors per partition, we base our approach on minimizing the sum of gravity forces in a system. Our approach bases on the following observation. We starts with large number of planets with the same mass distributed in the universe. For each pair of those planets, we can calculate the gravity force acting between them. If we have a section of the universe, with larger density of planets, the total sum of gravity forces acting on each of the planets withing this section will be larger than for the rest. Now, if we remove the planet with the maximal sum of gravity forces from such a region, the sum of gravity forces on the neighboring planets will be reduced. In other words, by removing one planet we are increasing the entropy of the universe regarding the gravity force. We can repeat this method until we have a desired number of planets in the universe. With each iteration, we will obtain a new (smaller) universe with the property of geographic diversity, since we are always removing one planet from within the most “populated” region of the universe.
By using this approach, we would not obtain a fisheye view of the universe,

since the resulting planets would be evenly distributed trough the universe. But, if we have planets with masses, that increase with the distance to the center of the universe, our algorithm would result in a fisheye view.
Having this analogy, all what we have to do, is to apply it to an overlay network. The host that constructs its fisheye view is considered to be the center of its universe (virtual space). All other possible neighbors, are positioned according to the embedding in a virtual space regarding RTTs. We can apply the incremental removal of planets to this universe until we obtain a fisheye view. In the Fig. 2.1 we show an example result of such a reduction. We can observe an overlay network as a connected graph.

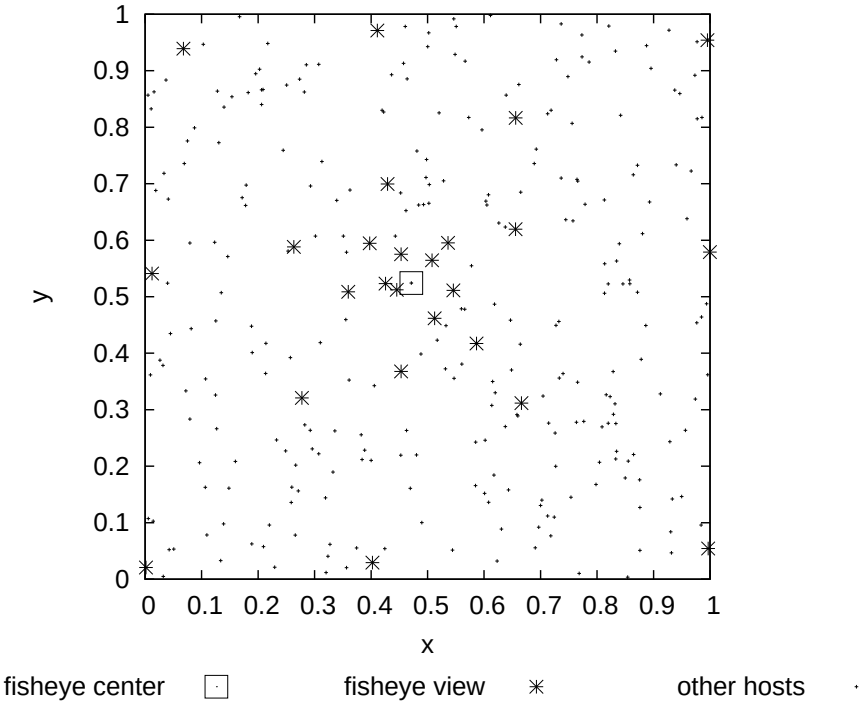


Figure 2.1: Example of fisheye view calculated using our gravity force minimization algorithm.

If each end system independently decides which other end systems are known to him, we obtain a directed graph. Such a graph has no guarantee that it will be connected. For example, if we have a portion of the virtual space with high end system density, the chances are growing that at least one of those end systems will not be “chosen” by some other end system to be its neighbor. As a consequence, if one of the end systems is never

chosen as a neighbor, there is no edge in the graph representing the overlay network, that leads to this end system. This means, that the graph is not necessarily connected.
To overcome this problem, we propose, that "is a neighbor of" relation of a fisheye view is bidirectional. This means n_1 is a neighbor of n_2 iff n_2 is a neighbor of n_1. If the "is a neighbor of" property holds, the graph is bidirectional and must be connected.

2.3 NetICE9

Almost all RTT prediction schemes base on embedding RTT information as distances into metric spaces. This is done so in such a way that the RTT information between two host is consistent with the measured RTT. If the exact embedding is not possible, due to violations of some metric space constraints such as the triangle inequality, all systems try to minimize the error of the embedding. Basically, there are two approaches to embedding RTTs into metric spaces. One approach initially proposed by GNP [5, 6] uses a set of hosts termed landmarks, to position all other hosts in the system relative to them. The other approach, proposed in VIVALDI [7] uses a distributed simulation of a spring system to position the hosts. Both of the approaches have their advantages and drawbacks. For example, VIVALDI does not need landmarks and is this fully distributed, but since the RTTs in the Internet violate the triangle inequality property of metric spaces, it never converges towards an stable state. Instead, the whole system rotates, translates and oscillates in the virtual space. As a result, the positions of hosts are not stable. GNP, on the other hand, results in stable coordinates for the host. But, on the downside, GNP needs landmarks, which tend to be a bottleneck in the system, since all RTTs for every non-landmark host is measured to the landmarks.
In our work, we develop an new approach, which combines advantage of both GNP and VIVALDI. Our goal is to have a system to assign stable coordinates to end systems without using landmarks. To achieve this we use the analogy of the crystallization process. Crystallization in the nature starts usually at one point, the nucleus. Starting at the nucleus, a crystal grows incrementally around the nucleus by incorporating atoms into a crystal structure until the whole crystal is built.
To transpose the process of crystallization we propose the following. We build the overlay network, as proposed in Section 2.2. After the overlay network is built, at an arbitrary point (at any end system) the crystallization starts. At the starting end system, the fisheye-view of that end system is

used to determine positions of all hosts from the fisheye view. This determining of the positions is done in the similar manner, as the landmark positions in GNP are determined (minimization of complete RTT information). After the initial end system has determined positions of its fisheye, it notifies the involved hosts regarding their positions. Every host, which received its position in the virtual space, re-iterates the process of crystallizing its fisheye. This is done in such a way, that only the neighbors, which do not have coordinates are crystallized. Since the overlay network built using the protocol described in Section 2.2 has a fisheye view property, the crystallization process is guaranteed to finish in $O(\log n)$ time relative to the number n of end systems in the overlay network.
Since the network topology may change, we propose periodical re-crystallization, which would reposition hosts in order to adapt to current RTTs in the overlay network.

2.4 Conclusion and Outlook

In last two sections we described a method for building an self improving topology overlay network and a method to use this overlay network. We also described a way to embed hosts into a virtual metric space, based on RTT information. Our goal was to obtain an embedding, which is fully distributed, but still has stable host positions. To determine if our approaches result in a better overlay network and a better embedding of the hosts, thorough simulations must be performed.

References

[1] S. Floyd and K. Fall, “Router mechanisms to support end-to-end congestion control,” tech. rep., Lawrence Berkeley National Laboratory, 1997.

[2] Y. hua Chu, S. G. Rao, and H. Zhang, “A case for end system multicast (keynote address),” *SIGMETRICS Perform. Eval. Rev.*, vol. 28, no. 1, pp. 1–12, 2000.

[3] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, “Scalable application layer multicast,” in *SIGCOMM ’02*, vol. 32, (New York, NY, USA), pp. 205–217, ACM Press, October 2002.

[4] A. Rowstron, A.-M. Kermarrec, M. Castro, and P. Druschel, "Scribe: The design of a large-scale event notification infrastructure," in *Networked Group Communication, Third International COST264 Workshop (NGC'2001)* (J. Crowcroft and M. Hofmann, eds.), vol. 2233 of *Lecture Notes in Computer Science*, pp. 30–43, Nov. 2001.

[5] T. S. E. Ng and H. Zhang, "Predicting internet network distance with coordiantes-based approaches," in *IEEE Infocom02*, (New York / USA), June 23-27 2002.

[6] T. S. E. Ng and H. Zhang, "A network positioning system for the internet," in *USENIX 2004*, (Boston MA, USA), pp. 141–154, June 2004.

[7] F. Dabek, R. Cox, F. Kaashoek, and R. Morris, "Vivaldi: a decentralized network coordinate system," in *SIGCOMM '04*, (New York, NY, USA), pp. 15–26, ACM Press, 2004.

3 Aspects of Energy-efficient Management of Heterogeneous Wireless Sensor Networks WSN

Markus Anwander and Gerald Wagenknecht, University of Bern
anwander@iam.unibe.ch - wagen@iam.unibe.ch

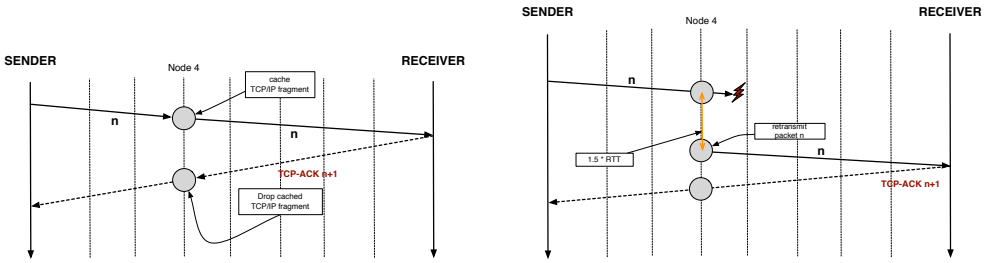
3.1 Introduction

Support and operation of Wireless Sensor Networsk (WSN) require reliable and energy-efficient communication. To interconnect such WSN with the Internet the WSN have to be TCP/IP-enabled. In this area we focus on three aspects: TCP Support for Sensor Nodes (TSS), energy-efficient beacon-less 802.15.4 MAC protocol and reliable IP-based Multicast.

3.2 TCP Support for Sensor Nodes (TSS)

To ensure a reliable transport in WSNs a new protocol between TCP and IP, the TCP Support for Sensor Networks (TSS) [1] protocol has been inserted. This protocol is responsible for the following tasks: Caching and local retransmission of TCP data packets, Improving the TCP acknowledgment mechanism, Flow and congestion control. TSS supports energy-efficient operation of sensor nodes by reducing the number of transmissions. The TSS protocol holds a buffer to store the incoming frames from the lower layers (radio transceiver→MAC) and the upper layer (TCP). The reliability mechanisms of TSS have an effect, when the network is overloaded (e.g. because of a high load of packets of other protocols or interferences). The MAC protocol must not be too reliable, because in case of unreliable transport protocols (such as UDP) there can be too high overload and delays.
In general, TCP is a reliable byte stream protocol designed for wired networks. Reliability is provided by positive acknowledgment with end-to-end retransmissions. In contrast to a wired connection, a wireless network possesses a high bit error rate. This leads to many end-to-end retransmissions. These extra packets reduce the throughput and increase the round-trip-time (RTT). The extra energy for retransmissions also reduces the lifetime of the individual sensor nodes. One of the basic ideas of TSS

is to cache a *TCP-DATA* packet in the WSN until the packet is acknowledged (shown in Figure 3.1(a)). The case of a packet loss is shown in Figure 3.1(b).

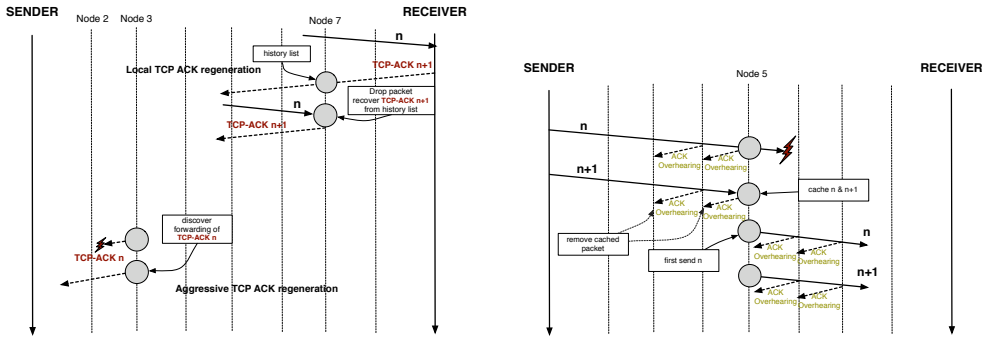


(a) Caching of TCP data packets.

(b) Retransmission of lost TCP data packets.

Figure 3.1: Basic ideas of TSS.

In TSS, the *TCP-ACK*s are very important to estimate the RTT, RTT timers, caching, etc. Experiments in [1] show that the loss of *TCP-ACK*s may have severe impact on the amount of *TCP-DATA* packet transmissions. Two mechanisms reduce the consequences of lost *TCP-ACK*s: local TCP acknowledgment regeneration and an aggressive TCP acknowledgment recovery (shown in Figure 3.2(a)). The local TCP acknowledgment regeneration is used to drop duplicated TCP data packets, which are already acknowledged by the receiver. Because of the history list, an intermediate node can discover an already acknowledged packet. The duplicated packet is dropped and a TCP acknowledgment with the highest acknowledgment number seen so far is regenerated and transmitted. A scenario is shown in 1) of Figure 3.2(a)). The aggressive TCP acknowledgment recovery becomes active when a sensor node cannot ensure by *MAC-ACK* that the *TCP-ACK* has been successfully transmitted to the next hop. Using *MAC-ACK*, the retransmission can be enforced directly. It is shown in 2) of Figure 3.2(a).
Because of the limited memory, only a few packets can be cached and an efficient caching algorithm is required. A node caches a packet until it knows that the successor node has successfully received the packet. This can be discovered by receiving an acknowledgment on the data link layer or through overhearing an implicit acknowledgment (shown in Figure 3.2(b)). The MAC protocol is responsible to establish a stable link to the neighbor nodes. The communication should be energy-efficient. Therefore, the physical layer has to provide additional information about the transmissions. The radio transceiver provides CCA (Clear Channel As-



(a) Two mechanisms to reduce the consequences of lost TCP acknowledgments.

(b) Cross layer support of the MAC protocol.

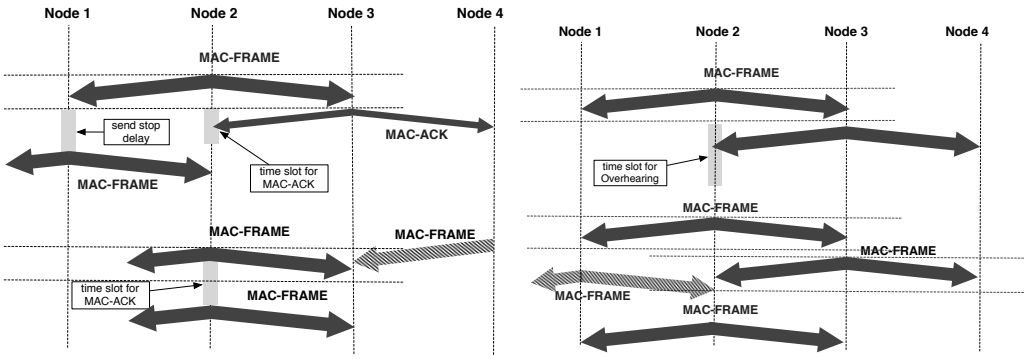
Figure 3.2: Basic ideas of TSS.

sessment), LQI (Link Quality Index), and RSSI (Receive Signal Strength Indicator). CCA provides information whether the channel is used or not. LQI indicates the quality of a link between two nodes and RSSI provides the signal strength. The MAC protocol decides with this information, if a frame can be transmitted to a neighbor node. The exchange of information between the MAC protocol and the TSS protocol is very important for the reliability mechanisms. The MAC protocol informs the TSS protocol about the transmission state of a frame. It can be transmitted successfully or the transmission can fail (e.g. if the channel was busy). Further, the MAC protocol gives information about retransmissions of segments and the traffic between the nodes in the neighborhood. This information is important for congestion control and avoidance in the TSS protocol. The TSS layer provides the information about the buffer size and the free space in the buffer. The MAC layer uses this information and drops a frame (with payload), if the buffer is full.

3.3 Energy-efficient beacon-less 802.15.4 MAC protocol

The MAC protocol at the data link layer depends strongly on the underlying radio transceiver and is specially tailored to it. We are using the CC2420 radio transceiver [2], which implements the physical layer of the IEEE 802.15.4 standard [3]. The proposed MAC protocol implements the MAC layer of beacon-less PANs defined in the IEEE 802.15.4 standard for peer-to-peer topologies [4, 5]. To our knowledge it is the first implemen-

tation in this way. It provides energy-efficient multihop communication. All nodes are full function devices (FFD). The format of the MAC frame is conform to the IEEE 802.15.4 standard.
There are two kinds of acknowledgment modes: MAC-ACK and MAC-OVERHEARING. In case of MAC-ACK the MAC protocol initiates the transmission of an acknowledgment. In case of MAC-OVERHEARING no acknowledgment frame is transmitted. Instead the radio transceiver listens whether the following node forwards the frame. The upper layers get informed about the state of acknowledgment (cross layer information). There are three states: **(1)** The frame has been successfully transmitted (confirmed via MAC-ACK or MAC-OVERHEARING). **(2)]** The frame has been transmitted, but there is no confirmation. **(3)** Frame transmission failed.



(a) MAC-ACK. (b) MAC-OVERHEARING.

Figure 3.3: Acknowledgment types of the MAC protocol.

The MAC-ACK mode shown in Figure 3.3(a) works as follows: *MAC-DATA* is sent after checking whether the channel is free and whether there is no *Send Stop* active. After successfully receiving the frame a *MAC-ACK* is transmitted immediately without checking the channel. A *MAC-DATA* is received which is not addressed for the receiver. It sets the *Send Stop* timer, thus corresponding *MAC-ACK*s from the receiver to the sender cannot be destroyed. After the timer has expired it can transmit regularly. The *MAC-ACK* is received within the *time slot for MAC-ACK*. If a collision occurs and the *MAC-DATA* frame is destroyed, the *MAC-DATA* frame is retransmitted after expiring the *time slot for MAC-ACK*. The MAC-OVERHEARING mode is shown in Figure 3.3(b). There exists no explicit acknowledgment. The sender listens when the receiver forwards the frame to the next hop. The mechanism works as follows: The sender transmits a *MAC-DATA* frame. The receiver forwards the frame and the sender overhears it (within the

time slot for overhearing). Also in the MAC-OVERHEARING mode collisions can occur. Because nodes 1 and nodes 3 cannot hear each other, they transmit frames at the same time. Thus frames collide and overhearing is impossible. On expiration of the *time slot for overhearing* there are no retransmissions. The TSS protocol gets the information that the frame is transmitted, but not acknowledged (not overheard). Retransmission is initiated by TSS.

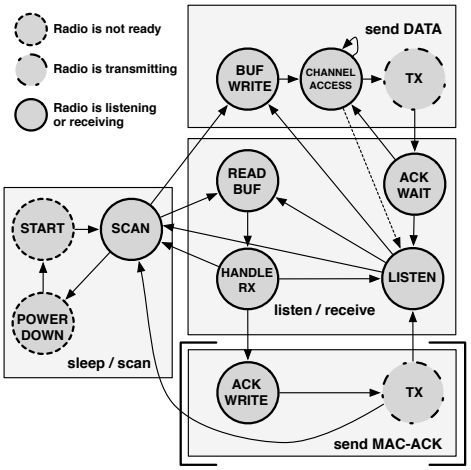


Figure 3.4: State machine of the MAC protocol.

The energy-efficient MAC protocol is realized as a state machine and is shown in Figure 3.4. It works as follows: According to the duty cycle the radio is started (START) and listens to the channel (SCAN). If there are no data to receive, the radio is switched off again (POWER DOWN). If data are received, they are copied to the MAC protocol buffer (BUF READ). The received packet is handled by the MAC protocol (HANDLE PACKET). The packet handling is described in more detail in [4]. An acknowledgment frame is created, written into transmission buffer, and transmitted conform to 802.15.4. The packet is dropped, if the node is not the receiver or if the checksum of the packet is not correct. The radio is turned into the scan/sleep mode. If a packet from the upper layers has to be transmitted to another node, then the packet is copied into the transmission buffer (BUF WRITE). Afterwards the channel is checked (CHANNEL ACCESS). If the channel is busy the packet is dropped, and the state switched to the LISTEN state without a transmission. If the channel is free, the packet is transmitted (TX). Afterwards the radio waits for the acknowledgment. If the neighbor does not respond, transmission of the data packet is retried during the duty cycle. If the energy-saving mode is enabled, the MAC

protocol switches immediately to the SCAN state. In contrast to the implicit acknowledgment mode, the MAC protocol waits for the forwarding of the transmitted packet. The OVERHEARING WAIT time depends on the duty cycle, which determines the length of the sleep/scan cycles. If the packet could not be overheard, the transmission is retried once. If the energy-saving mode is disabled, the waiting time for overhearing is longer. Because of higher traffic, the neighbor node might wait longer trying to forward the packet. Thus, it must be waited longer for overhearing.

3.4 Reliable Multicast in Wireless Sensor Networks

Due to the nature of WSNs, the IP Multicast implementation can not be simply ported from existing solutions for wired networks. In wired networks, routers are handling packet replication and forwarding, clients just send and receive simple IP UDP datagrams. On the other hand, WSNs would need to introduce the router functionality for IP Multicast management into each sensor node. Group management is normally concentrated on the routers that communicate with each other to handle multicast trees. The management for multiple groups and multicast trees requires memory and processing power, which is limited on sensor nodes. Also the default implementations of IP Multicast are designed to scale on large network groups with multiple receivers and senders. In practical WSNs typically the amount of nodes is rather low. Also the amount of active trees and general management communication should be kept to a minimum. Existing Overlay Multicast [6] solutions (such as Scribe/Pastry, CHORD, Bayeux) are normally not taking the wireless nature and limited capabilities of sensor nodes into account. Also reliability for a WSN multicast solution would also be desirable, because code updates and other critical tasks could then be solved efficiently using multicast.

Multicasting in WSNs can be designed in different ways. We will look at two approaches, reliable IP Multicast and Overlay Multicast. For both approaches we will look at source-driven and receiver-driven designs, both centrally managed as well as de-centrally organized. Generally we will distinguish between two node types. Branching nodes have to duplicate packets and store state information about receivers and/or about other branching nodes. Forwarding nodes have less or no information about the multicast state and just forward the multicast data to one neighbor. We will also limit our discussion to core-based trees, where only the dedicated root

node will disseminate the data, while other senders would need to transmit the data to the root node first for dissemination. An example topology with some branching nodes, forwarding nodes and three group members is shown in Figure 3.5.

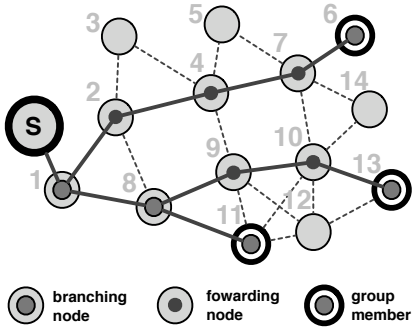


Figure 3.5: Example topology showing branching nodes, forwarding nodes and group members

For the source driven scenario we can use a de-centralized as well as a centralized approach. Generally we distinguish between active and inactive multicast trees. While data is transmitted to a multicast group, the tree is in the active state with all required TCP connections for the overlay links established. When no data needs to be transmitted, the tree is inactive and all required TCP connections to build the distribution overlay are closed. Nodes can only join to a multicast group while its tree is inactive. In the de-centralized and source-driven approach, the source sends the list of all receivers of the multicast data to its one-hop neighbors. If nodes from the list can be reached via different neighbors, the list is split accordingly and the partial lists. The node splitting the list becomes a branching node and opens a TCP connection for the overlay link to the sender of the original list, when the corresponding tree is activated. Therefore, TCP connections for the overlay network links are only established between the source, branching nodes and the receivers (see also Figure 3.5) when the corresponding multicast tree becomes active. New nodes are added to the multicast tree by sending a list message including the new nodes as described before. No connections are established directly, but potential new overlay links are just prepared and only established upon activation of the tree. If a branching node now receives a list message with new nodes, it changes the source address of the list message, splits the list if required, and forwards it/them further. If a forwarding node has to become a branching node, it prepares the overlay link to the source of the list message, splits the message and forwards the new list messages further as

described before. This new branching node tells the source of the original list message which receivers it handles in the future. Therefore, the previous branching node removes the overlay link that previously was using this new branching node as forwarder.
In the source-driven centralized approach, the source node determines all required branching nodes ahead. Therefore, the source also creates the complete distribution tree that is required for a multicast group. The branching nodes are then notified, process the information and further forward these notifications. If new receivers need to be added to a tree while it is inactive, the source calculates the new and/or modified branching nodes. Only these nodes are notified about the changes in the tree, branching nodes that do not need to be changed require no notification. For the centralized receiver-driven approach, the join messages from the receivers are forwarded to the source, which manages the tree as described for the source-driven centralized approach. In the receiver-driven de-centralized approach, receivers send the join message to their neighbor responsible for the default route. If this node is not a forwarder or branching node for that group it becomes a forwarding node (only knowing that it is on the path of an overlay link when the tree would become active) and forwards the join message further. Intermediate nodes, which are already branching nodes of the requested group drop the join message and prepare the overlay link to the new receiver. Forwarding nodes receiving join messages, become new branching nodes, prepare the new overlay link and send this information (about becoming a branching node) towards the source, dropping the original join message. A branching node receiving such a message modifies its overlay link in that direction. Receivers that want to leave a group send a leave message towards the source. Forwarders on the path update their status for that group and forward the leave message further. Branching nodes receiving a leave message update their status, remove the overlay link to the leaving node, and discard the leave message. Further, it sends a notification towards the source and all intermediate nodes update their states accordingly. They forwarding the message until it reaches a branching node, which then establishes the overlay link to the remaining receiver.
To support end-to-end reliability in overlay multicast, the receivers have to acknowledge the receipt of each multicast message or acknowledge the receipt accumulated after a series of messages. Branching nodes aggregate and forward the acknowledgments. In case of missing acknowledgments, they send negative acknowledgments further towards the source. Branching nodes also take care of retransmission of lost packets and therefore need to cache the multicast data up to a certain degree.

Contrary to Overlay Multicast (which uses TCP) we do not have a reliable end-to-end transport protocol. Instead we are using UDP. End-to-end reliability is realized using acknowledgment messages as described above. Branching nodes know only that their one-hop neighbors are forwarding the packets on their behalf. Acknowledgments be handled on one-neighbor basis (hop-to-hop), and not between branching nodes as for Overlay Multicast.
For the source-driven de-centralized approach the source sends the join list to its direct neighbors that should act as forwarders. The next forwarder is determined on a hop-to-hop basis. If a node has to become a branching node, it remembers from which neighbors it expects acknowledgments. Joins and leaves are handled by appropriate messages that could cause forwarders to become branching nodes (and vice versa) triggering a modification of the expected acknowledgments state for a node.
In the centralized source-driven approach the source sends the list of all branching nodes to the closest branching node, which processes and forwards them further to the nearest branching nodes on the path. Intermediate nodes become forwarders and store the status for the involved multicast group. Acknowledgments are handled directly between the branching nodes. Additional joining nodes trigger an update of the affected branching nodes all initiated directly by the source. Leaves are handled accordingly triggering updates of the branching and forwarding nodes involved.
In the receiver-driven centralized approach joins are sent to the source, which then acts as in the source-driven centralized approach. In the de-centralized receiver-driven approach, joins cause intermediate nodes to react as in the Overlay Multicast case. They either become forwarding nodes for that group if they are not handling that group yet or become branching nodes if applicable. Acknowledgments are handled the same way as described above in the de-centralized source-driven approach.

3.5 Conclusion and Outlook

We discussed three aspects of energy-efficient communication for management and operation of IP-based wireless sensor networks. We presented TCP Support for Sensor Nodes. In collaboration with the energy-efficient 802.15.4 MAC protocol, it allows an energy-efficient using of TCP in WSNs. Furthermore we discussed reliable Multicast in WSNs. We distinguished between Overlay Multicast and IP-based Multicast, between centralized and de-centralized, and between source-driven and receiver-driven.

References

[1] T. Braun, T. Voigt, and A. Dunkels, “TCP Support for Sensor Networks,” in *Proceedings of IEEE/IFIP Annual Conference on Wireless On demand Network Systems and Services (WONS’07)*, (Obergurgl, Austria), 2007.

[2] “CC2420: Datasheet for the Chipcon CC2420 2.4 GHz IEEE 802.15.4 compliant RF Transceiver,” 2007.

[3] “802.15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs),” 2006.

[4] M. Anwander, G. Wagenknecht, and T. Braun, “Management of Wireless Sensor Networks using TCP/IP,” in *Proceedings of the International Workshop on Sensor Network Engineering (IWSNE’08)*, (Santorini Island, Greece), pp. II.1–II.8, 2008.

[5] M. Anwander, G. Wagenknecht, and T. Braun, “Energy-Efficient Communication with a Beacon-Less 802.15.4 MAC Protocol for Wireless Sensor Networks,” in *Proceedings of the IEEE International Conference on Communications (ICC’09)*, (Dresden, Germany), p. submitted, 2009.

[6] M. Hosseini, D. T. Ahmed, S. Shirmohammadi, and N. D. Georganas, “A Survey of Application-Layer Multicast Protocols,” *Communications Surveys & Tutorials*, vol. 9, no. 3, pp. 58–74, 2007.

4 Fuzzy Classifier Design for Efficient and Robust Event Classification in Wireless Sensor Networks

Markus Wälchli, University of Bern
waelchli@iam.unibe.ch

4.1 Introduction

In this report preliminary work considering fault-tolerant event classification using wireless sensor networks is presented. The goal of the research is to develop a classifier, which is able to classify and filter distributed computed event reports. Thereby, the classifier shall be tuned by unsupervised learning, i.e., its parameters have to be learned from training data. Thus, the deployment of the the algorithm is simpler, i.e., needs no expert knowledge, and design flaws due to poor data abstractions can be prevented.
In previous work [1] a distributed event detection and tracking algorithm (DELTA) has been developed. DELTA establishes and maintains distributed tracking groups, which collect event observing data at a dedicated leader node. From this data event reports are generated. This reports can either be classified and filtered directly on the leader node in the operating phase, or they are sent to a base station in a training phase. The data collected at the base station in the training phase is used to learn the parameters of the classifier, which are then downloaded on the sensor nodes. Having estimated and downloaded the parameters of the classifier, each node is able to perform the classification.
Online classification and filtering have the advantage that costs can be saved in terms of communication, energy, time and money. If false alarms are filtered, the network is not charged with the routing of these, useless, reports. Moreover, no alarms are triggered and, accordingly, no actions by operators are required. On the other hand, the system must guarantee that correct alarms are not filtered. In order to classify and filter event reports, different event types need to be distinguishable and the relevance of specific event reports must be assessable.
To support false-alarm prevention, the usage of fuzzy logic concepts is proposed. By applying a fuzzy logic controller (FLC), each classification gets a membership degree assigned. This membership degree determines the belief degree with which a given (classified) report belongs to a certain

event class. Having computed this membership degree, event reports which do not satisfy a specified threshold can be filtered. The possible filtering of event reports requires a periodic reclassification of events. This is no restriction as most monitoring applications base on periodic sampling anyway. This is in particular true if the event is mobile and needs to be tracked. The DELTA framework, mentioned above, performs the recomputation of the event characteristics twice a second. This value could be changed depending on the application.
The filtering of event reports has an impact on reporting delays. If event reports are filtered, longer latencies could occur until an event report satisfies the filtering condition and is routed to the base station. Therefore, the trade-off between false alarm prevention and reporting latency will need to be evaluated for each specific application of the FLC classifier.

4.2 Discussion

As mentioned in the last section unsupervised classifier training is considered. The required training data is collected at a bases station in the form of event reports, which have been collected by DELTA in dedicated training phases. The tuning (learning) of the classifier is done in two steps. First, clusters representing the event patterns, which will have to be identifiable by the classifier, have to be extracted from the collected training data. The parameters of the classifier are then computed based on these clusters.
To learn, respectively extract, the event patterns, a fuzzy k-means algorithm [2] is used. The fuzzy k-means algorithm [2] is a generalization of the well-known k-means algorithm, expanding the binary decisions of k-means with membership degrees. The goal is to find a decomposition of the (training) set $\mathbf{Z} = \{\mathbf{z}_1, ..., \mathbf{z}_M\}$ into m clusters $\{C_1, ..., C_m\}$. Thereby, each $\mathbf{z} \in \mathbf{Z}$ is assigned to each cluster C_i with an estimated membership degree $\mu_i(\mathbf{z})$. The membership degree is computed according to [2]:

$$\mu_i(\mathbf{z}) = \begin{cases} 1, & \text{if } \mathbf{z} = \mathbf{m}_i, \\ \frac{1}{\sum_{k=1}^{K} \left(\frac{\|\mathbf{z}-\mathbf{m}_i\|}{\|\mathbf{z}-\mathbf{m}_k\|} \right)^{\frac{2}{\beta-1}}}, & \text{else.} \end{cases} \quad (1)$$

where the parameter β controls the membership gradient and must be larger than 1. $\|.\|$ is the Euclidean norm. Accordingly, $\mu_i(\mathbf{z})$ determines the degree to which $\mathbf{z}$ belongs to cluster C_i in dependence of the Euclidean distance between $\mathbf{z}$ and the cluster center $\mathbf{m}_i$. Having defined this membership degree function, the fuzzy k-means algorithm looks as follows:

Fuzzy k-means

input: Training set $\mathbf{Z}$; K = number of clusters;
ouput: m clusters $\{C_1, ..., C_m\}$;
$\mu_j(\mathbf{z}_i)$ for $1 \leq j \leq K$ and $1 \leq i \leq M$;
begin
choose K initial cluster centers $m_1, ..., m_K$;
repeat
compute $\mu_j(\mathbf{z}_i)$ for $1 \leq j \leq K$ and $1 \leq i \leq M$ according to (1);
update the cluster centers $\mathbf{m}_j$ for $1 \leq j \leq K$;
until termination criteria satisfied;
end

The algorithm depends on K, which is the number of clusters which are expected in $\mathbf{Z}$. K could be estimated from $\mathbf{Z}$, but is the only value provided by an operator into the system. This is reasonable, as the number of event patterns (K) should be known by the operator. The choice of the initial cluster centers has an impact on the results. The optimal choice of the initial clusters is, in general, not feasible. Accordingly, it cannot be guaranteed that the algorithm provides an optimal solution.
Having extracted the (fuzzy) clusters from $\mathbf{Z}$, the FLC classifier can be trained. First, some required definitions are listed. Defining $U = \{u_1, ..., u_n\}$ as the universal set, a fuzzy set $\tilde{A}$ on U is defined by a membership function:

$$\mu_{\tilde{A}} : U \rightarrow [0, 1] \tag{2}$$

where $\mu_{\tilde{A}}(u)$ expresses the membership degree to which u belongs to $\tilde{A}$. In our work, triangular and Gaussian membership functions will be considered. These functions are parameterized as follows:

$$\mu(x) = \begin{cases} \frac{x-a}{b-a}, & \text{if x} \in (a, b], \\ \frac{c-x}{c-b}, & \text{if x} \in (b, c], \\ 0, & \text{otherwise.} \end{cases} \quad and \quad \mu(x) = e^{-\frac{(x-a)^2}{2\sigma^2}}$$

Gaussian functions have the advantage that the whole feature space is covered. Considering the classification problem of assigning a sample $\mathbf{x} \in \Re^n$ to one of m possible event classes, a FLC classifier consists of m classification rules R_k of the form [3]:

$$\begin{aligned} R_k : IF \ & \mu_{\tilde{A}_{k,1}}(x_1) \wedge \mu_{\tilde{A}_{k,2}}(x_2) \wedge ... \wedge \mu_{\tilde{A}_{k,n}}(x_n) \\ & THEN \ g_k(\mathbf{x}) > g_i(\mathbf{x}), \ \forall i = 1, ..., m \end{aligned} \tag{3}$$

In order to apply these rules, the fuzzy sets $\tilde{A}_{k,i}$ of the premises as well as the functions $g_k(\mathbf{x})$ of the conclusions need to be known for each rule R_k. The fuzzy sets $\tilde{A}_{k,i}$ are extracted from the clusters C_k, whereas the conclusion is modeled according to a Takagi-Sugeno TSK2 classifier [3]:

TSK2 classifier

- $z_{k,i} \in \Re,\ \ k = 1, ..., m,\ \ i = 1, ..., m$;
- The conjunction (AND) is the product;
- The ith output of TSK2 is

$$g_i^{TSK2}(\mathbf{x}) = \frac{\sum_{k=1}^{M} z_{k,i} \prod_{j=1}^{n} \mu_{\tilde{A}_{k,j}}(\mathbf{x}_j)}{\sum_{k=1}^{M} \prod_{j=1}^{n} \mu_{\tilde{A}_{k,j}}(\mathbf{x}_j)} \tag{4}$$

The fuzzy sets $\tilde{A}_{k,i}$ of the premises are derived for each specific cluster by projecting the according cluster onto the coordinate axis of the n-dimensional space it is part of. Fig. 4.1 depicts the projection of a 2-dimensional cluster, i.e., of event patterns consisting of two kinds of signals.

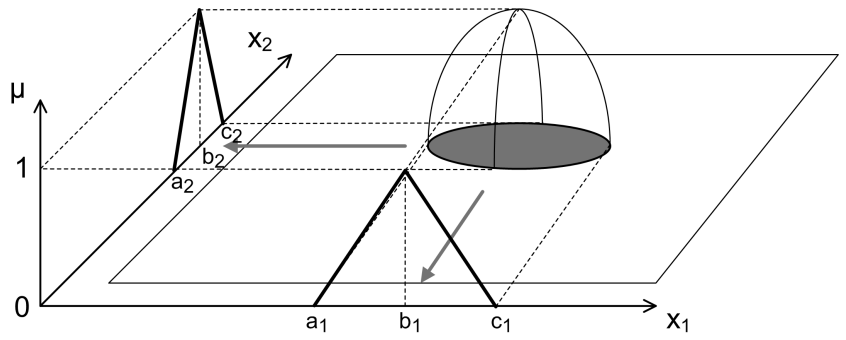


Figure 4.1: Extracting the fuzzy sets of the premises from a cluster in $\Re^2$.

In this example, triangular membership functions are derived for the fuzzy sets $\tilde{A}_{k,i}$. Alternatively, Gaussian membership functions could be modeled. In order to run the TSK2 classifier proposed above, the parameters $z_{k,i}$ in Equation (4) need to be computed. The $z_{k,i}$ are obtained according to [3]:

$$z_{k,i} = \frac{\beta_{k,i}}{\sum_{s=1}^{m} \beta_{k,s}} \tag{5}$$

The $z_{k,i}$ are the normalized values of the $\beta_{k,i}$, which are the firing strengths of each rule in respect to each event class C_i. It is important to keep in mind that each rule is designed to classify samples of one specific event class C_i. The $\beta_{k,i}$ are defined as follows [3]:

$$\beta_{k,i} = \sum_{\mathbf{z} \in \mathbf{Z}} Ind(\mathbf{z}, C_i)\tau_k(\mathbf{z}), \qquad \forall k = 1, ..., m; \tag{6}$$

where $Ind(\mathbf{z}, C_i)$ indicates the cluster C_i the element $\mathbf{z}$ belongs to, i.e., $Ind(\mathbf{z}, C_i)$ is 1 if sample $\mathbf{z}$ is assigned to C_i, and 0 else. The firing strength of a rule depends on the firing strenght of its premises. Accordingly, the firing strength $\tau_k(\mathbf{z})$ of the premises is defined as [3]:

$$\tau_k(\mathbf{z}) = \Lambda(\mu_{\tilde{A}_{k,1}}(\mathbf{z}_1), ..., \mu_{\tilde{A}_{k,n}}(\mathbf{z}_n)) \tag{7}$$

where Λ is a specific aggregation function, e.g., the product if TSK2 is used (see 4).

4.3 Conclusion and Outlook

In this report the design of a false-alarm-tolerant FLC classifier has been discussed. Well-known algorithms to tune the classifier parameters from training data have been introduced. The approach is completely self-learning and, having estimated the parameters, all subsequent classification can be performed locally within the network. In future we intend to evaluate the FLC algorithm in realistic environments.

References

[1] M. Wälchli, S. Bissig, M. Meer, and T. Braun, "Distributed event tracking and classification in wireless sensor networks," *Journal of Internet Engineering*, vol. 1, no. 2, pp. 117–126, 2008.

[2] M. Friedman and A. Kandel, *Introduction to pattern recognition: statistical, structural, neural, and fuzzy logic approaches*. World Scientific, 1999.

[3] L. I. Kuncheva, *Fuzzy Classifier Design*, vol. 49 of *Studies in Fuzziness and Soft Computing*. Physica-Verlag, A. Springer-Verlag Company, 2000.

5 Cognitive Networks

Torsten Braun, University of Bern
braun@iam.unibe.ch

5.1 Introduction

Cognitive networks are a new paradigm for future communication architectures that are able to adapt to the requirements of users and changing conditions in the environment. Cognitive networks evolved from cognitive radios and extend cross-layer designs.

5.2 Cognitive Radio

The wireless spectrum is rarely used continuously over space and time [1, 2]. Therefore, a new access paradigm, so-called cognitive radios, have been developed. In case of cognitive radios, secondary (unlicensed) systems are allowed to opportunistically utilize unused primary (licensed) bands. These unused bands are also called “white spaces”. In the US, the Federal Communications Commission (FCC) is interested in permitting access to white spaces in analogue TV bands, which are available due to more popular use of digital TV.
To access the wireless channel in cognitive radio networks, users must first determine the available portions of the spectrum (spectrum sensing) and then select the best available channel (spectrum decision). Channel access must be coordinated by centralized or distributed (non)cooperative spectrum sharing mechanisms to avoid simultaneous use and collisions. Finally, users need to vacate the channel, when a licensed user is detected, and continue operation in another vacant channel (spectrum mobility). Spectrum sensing, decision, sharing and mobility are spectrum management functions.
The key technologies for the implementation of cognitive radio networks are reconfigurability and the capability to identify white spaces. Reconfigurability, which is typically achieved using reprogramming, is required to allow systems to transmit and receive on different currently available frequencies. Different access technologies can be supported by software-defined radios, which requires to implement low-level functions on physical layer such as analog/digital conversion or baseband processing in software.

For white space identification, sophisticated congnitive capabilities are needed. Primary systems must be protected from effects of secondary users‘ interference. As a first option, primary systems have to provide secondary systems with spectrum usage information, e.g., using central data base or beacon broadcasting. As a second option, secondary systems identify whites spaces through spectrum sensing of licensed bands. Systems can detect activities based on noise or energy detection. The periodicity of sensing must be selected appropriately. Moreover, the detection sensitivity must be carefully selected to detect possible interference by secondary systems. Faded and shadowed primary channels must be distinguished from white spaces. Noise uncertainty (estimation of noise power) and aggregate interference uncertainty can be mitigated by cooperative spectrum sensing [1].

5.3 Cognitive Networks

5.3.1 Principles

While in the cognitive radio paradigm wireless network elements change their transmission or reception parameters to communicate efficiently and to avoid interference with (un)licensed frequencies based on monitoring radio frequency spectrum, user behaviour, or network state [3], cognitive networks operate also on higher protocol layers than only the physical layer.

In [4] a cognitive network is defined as a network with a cognitive process that can perceive current network conditions, and then plan, decide, and act on those conditions. The network can learn from these adaptations and use them to make future decisions, all while taking into account end-to-end goals. In that sense, cognitive networks are similar to cognitive radios, but cognitive networks consider end-to-end issues: Cognitive networks cover all network elements and links between end systems, not just link adaptation, and do not focus on physical layer only.

Cognitive networks have also some similarities with cross-layer design architectures [5]. Cross-layer designs violate the layered communication system architecture in order to optimize performance. Layers inside a communication system can exchange and access information that is only visible within a layer of a traditional communication architecture. In contrast to cross-layer designs, in cognitive networks, information about network state and parameters is shared among different systems. In contrast to cross-layer designs, network-wide optimizations can be performed.

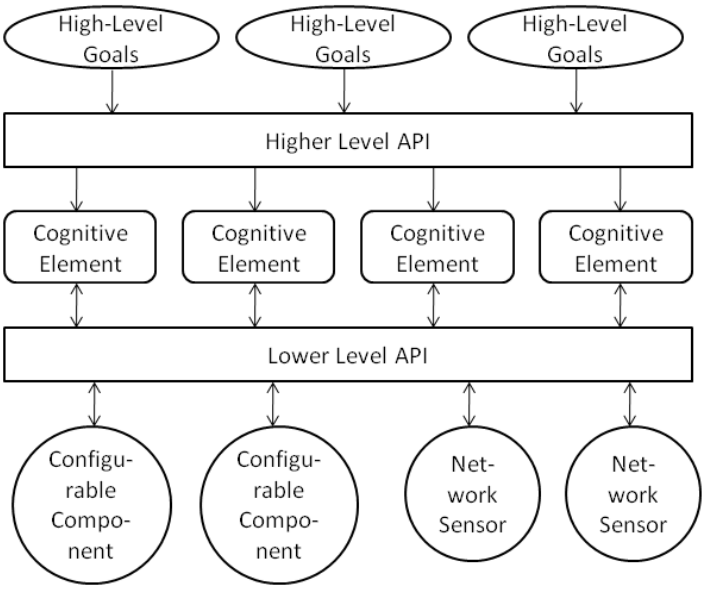


Figure 5.1: Architecture of a Cognitive Network Element

5.3.2 Architecture

Figure 5.1 shows the architecture of a cognitive network element, which includes cognitive elements that can perceive current network and environment conditions, and then plan, decide, and act on those conditions. The cognitive elements can learn from adaptations and use them for future decisions while considering higher-level end-to-end goals delivered via an higher-level API. The lower-level API is used for configuring components and for retrieving information from network sensors.

For the realisation of the cognitive elements, several cognition approaches have been proposed in the literature, such as neural networks, fuzzy logics, pattern recognition, expert systems, Kalman filters, learning automata, genetic algorithms, and other biology inspired algorithms.

A knowledge plane has been proposed to support decisions about appropriate actions and configurations of communication systems [6]. The knowledge plane builds and maintains high-level models of what the network is supposed to do, in order to provide services and advice to other elements of the network. It makes low-level decisions on its own. It creates, reconciles and maintains the many aspects of a high-level view, and then provides services and advice as needed to other elements of the network.

Critical issues in cognitive networks are the potentially computational complexity of the learning algorithms as well as the overhead to communicate and store information. The overhead must be traded against the achievable accuracy.

5.3.3 Applications

The applications of cognitive networks are manifold [7]. To support Quality-of-Service, cognitive networks can identify bottlenecks ion the network, estimate performance guarantees, and change priorities. In the area of security, cognitive networks can find attack patterns and security risks and then change security mechanisms such as rules, protocols etc. Using cognitive network concepts in heterogeneous wireless networks, we can learn from network status and adapt/optimize protocol parameters. In [8], a routing approach in wireless sensor networks is described, which optimized latency based on knowledge about the duty cycles of a node's 2-hop-neighborhood.

5.4 Conclusion and Outlook

Cognitive networks are a new paradigm for communication system architectures and autonomic network management. It can be sonsidered as an extension of cross-layer approaches as well as cognitive radios. A possible future generalization is to use input from the environment and sensors, not just information about network. New methods for adaptable and composable communication systems are needed and algorithms for different problems in wireless networks are required.

References

[1] A. Ghasemi and E. Sousa, "Spectrum sensing in cognitive radio networks: requirements, challenges and design trade-offs [cognitive radio communications]," *Communications Magazine, IEEE*, vol. 46, pp. 32–39, April 2008.

[2] I. Mitola, J. and J. Maguire, G.Q., "Cognitive radio: making software radios more personal," *Personal Communications, IEEE [see also IEEE Wireless Communications]*, vol. 6, pp. 13–18, Aug 1999.

[3] I. Akyildiz, W.-Y. Lee, M. Vuran, and S. Mohanty, "A survey on spectrum management in cognitive radio networks [cognitive radio communications and networks]," *Communications Magazine, IEEE*, vol. 46, pp. 40–48, April 2008.

[4] R. W. Thomas, D. H. Friend, L. A. Dasilva, and A. B. Mackenzie, "Cognitive networks: adaptation and learning to achieve end-to-end performance objectives," *Communications Magazine, IEEE*, vol. 44, pp. 51–57, Dec. 2006.

[5] V. Srivastava and M. Motani, "Cross-layer design: a survey and the road ahead," *Communications Magazine, IEEE*, vol. 43, pp. 112–119, Dec. 2005.

[6] D. D. Clark, C. Partridge, J. C. Ramming, and J. T. Wroclawski, "A knowledge plane for the internet," in *SIGCOMM '03: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, (New York, NY, USA), pp. 3–10, ACM, 2003.

[7] R. Thomas, L. DaSilva, and A. MacKenzie, "Cognitive networks," *New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005. 2005 First IEEE International Symposium on*, pp. 352–360, Nov. 2005.

[8] P. Hurni, T. Braun, B. Bhargava, and Y. Zhang, "Multi-hop cross-layer design in wireless sensor networks: A case study," 4th IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), October 2008.

6 Multicasting Solution for Mobile Ad-Hoc networks

Raphael Kummer, University of Neuchâtel
raphael.kummer@unine.ch

6.1 Introduction

Building multicast trees in mobile ad-hoc networks (MANETs) is an efficient method to distribute information to a subset of interested nodes. As communication between remote nodes require multiple hopes via relay nodes thus consuming energy and scarce CPU cycles, efficient solutions for MANETs have to minimize the number of nodes acting as relay outside of the tree (i.e., nodes that are not members of the tree).
Different approaches more or less structured have been proposed to build multicast trees in MANETs. MZR [1] relies upon the Zone Routing Protocol (ZRP [2]) to build multicast trees. Zones are defined around nodes and routes are maintained proactively within these zones. To reach nodes outside of these zones, ZRP uses a reactive routing method. This process is used to build multicast trees. Consequently, when a source has data to multicast, it floods its zone and then extends the tree to the nodes at the border of other zones. An interested node has to answer to the source and, when the message reaches a member, a branch is created. However the flooding is limited in a zone, the networks gets completely flooded zone by zone. Consequently, the energy and bandwidth consumption are significant and no efficient lookup mechanism is provided.
XScribe [3] and Georendezvous [4] use a distributed hash table (DHT) to support the multicast tree creation and to avoid flooding. XScribe exploits the DHT to route the multicast messages. The group membership is handled thanks to a bitmask system where all the nodes can be aware of all the members of available multicast groups by inspecting received routing packets. As the source has to send a message per member and all the nodes have to maintain the membership information for all groups and nodes, this approach doesn't scale well nor does it try to optimize resource consumption (by minimizing the number of relay nodes).
Georendezvous uses the underlying DHT, which groups physically close nodes in cell, to localize the cell responsible for a group. The nodes in the cell are responsible for the membership management and multicast messages distribution. As for XScribe, the centralized membership man-

agement and multicast messages distribution don't scale well nor consider resource consumption optimization.
As centralized algorithm offer but limited scalability and reliability, we propose a distributed multicast tree building algorithm. Our algorithm is supported by a DHT [5] specifically designed for MANETs which allows to efficiently localize the source of a multicast group. The membership management is distributed among all the participating nodes thus reducing the load of the source. Finally, our solution strives to minimize the number of relay node (i.e., Non-members node relaying multicast messages) implicated in the multicast messages distribution.

6.2 The Tree Construction Algorithm

Our algorithm strives to construct multicast trees that use member nodes as relays. Membership is handled in a decentralized way as a joining node might connect to the tree without the source knowing it. Consequently, the load on the source is reduced and we can avoid bottlenecks. One tree is constructed per active data source. We assume best effort delivery for multicast messages; additional mechanisms could be easily incorporated to implement reliable delivery.

6.2.1 The Connection Algorithm

The main principle of the algorithm consists in building a multicast tree using DHT lookups to locate the data source, and then applying various extensions to improve the tree. The DHT provides suitable facilities for efficient lookup without flooding the network and scales well to large number of nodes.
To join the multicast tree, a node routes a request to the identifier associated with the source of the multicast (group identifier). When receiving a join request on its way to the source, a node checks if it is member of the group. If it is the case, the current node replies to the joining node and proposes itself as parent in the multicast tree and forwards the request to the next node towards the source (according to the DHT lookup protocol). Otherwise, it simply forwards the request to the next node towards the source
To join the distribution tree as soon as possible, the requester connects to the first parent it receives. Thereafter, if it receives further responses from potential parents, it changes only if (1) the distance to the new parent is shorter than to the old parent and the new distance to the root is no more

than twice the old distance; or (2) the new parent is at the same distance as the old one but the distance to the root has shortened. At the end of the process, the node is connected to the multicast tree with the node that it considers as being the best parent.
With this straightforward algorithm, many nodes are connected to the source with direct paths and many non-member peers are located on the paths from the source to the members. We shall now present number of extensions that we have developed to improve the tree structure and to reduce the number of non-member nodes involved in multicasting. The first one is applied during lookup, while the second and third ones rely on information added by nodes to the multicast messages for reorganizing the tree. Finally, the last one listens to wireless communications for finding potential children. These extensions are always cumulated when applied (i.e., an extension also incorporates the former ones).

6.2.2 Finding More Potential Parents

The nodes that are not involved in a communication listen to the requests (radio communications are broadcasted) and cache some of the gathered information. In particular, a node listening to a join message will send a response to the requester if it is a member of the joined multicast group. Listening to communications is typically a cheap operation as it does not generate extra messages, yet it often allows to improve the structure of the tree. In particular, listening to messages avoids pathological situations where two multi-hop requests cross but do not traverse a common node.

6.2.3 Finding Better Parents

When a message is multicast, it usually traverses several non-member nodes between a parent and a child in the logical multicast tree. Two messages from one node to two different nodes may traverse a certain number of common relays. In fact, if two messages follow the same physical path, their destinations are likely to be in the same area of the network. Consequently, one can inform one of the nodes that the other one is possibly a better child or parent in the multicast tree. This optimization is interesting because, by reducing the distance between parents and children in the tree, we also reduce the network load.
We propose a solution where no additional messages are needed. When a node relays a multicast message (uniquely identified by a group and a message identifier) it memorizes the group, the message identifier, and the

destination address. If a node receives the same message (same group and message identifier) intended for another destination, the node adds the previously memorized address to the message before forwarding it. If more than one node have an address to add, only the last one is kept in the message (space overhead is negligible).
The receiver of a multicast checks if the address of a relay has been added to the message. If so, the receiver sends a message to the relay and proposes to become its parent or its child. A validity check is here necessary to avoid tree partition, see [6] for more details.

6.2.4 Removing Redundant Parents

As connections between a node and its children in the multicast tree are typically multi-hop, it may happen that a node is both a member of the logical tree and a relay along a multi-hop path between a node and one of its children. Obviously, the resulting structure is sub-optimal and this situation should be avoided, as the affected node receives the same message from more than one parent.
To deal with this situation, a node which detects that it receives the same message from multiple paths (both as a relay or as a member of the tree) keeps only one connection with the closer parent and discards the others. If the node is on a multi-hop path from a parent to its child, it may need to disconnect the child from its former parent and add it to its own children, or it may need to promote itself as new inner node of the tree along that path.
In either case, one physical path is discarded and the number of non-member relays is reduced. Moreover, the tree better maps to the underlying topology and multicast efficiency is improved.

6.2.5 Multicast Message Gathering

Finally, the last extension aims, as the first one, to take advantage of the nodes' operation by broadcast communication. Indeed, during their distribution, multicast messages may travel near a group member without being actually received or relayed by it. As communications are broadcast, members can however gather the information contained in those messages by simply listening to them.
The source decides to intermittently allow the members from a selected branch to use this grabbed messages by setting a flag in the multicast message sent to this branch. Each member from the selected branch

tries to find the worst case it can identify in the subset of valid messages (see [6] for more details) allowed to be used. The message considered as the worst case is the message having the largest distance with its sender. When the candidate message is found, the current member proposes to the addressee to become its parent.

6.3 Conclusion

Although much work has been done on the problem of multicast in ad-hoc networks, most of the solutions use some form of flooding or centralized solutions that are not scalable. We shortly presented an algorithm for the construction of efficient multicast trees using an underlying ad-hoc DHT overlay. Our algorithm strives to create trees that involve as few relay nodes as possible, requiring a limited amount of transmissions, with short inter-members paths and good scalability. Simulation results detailed in [6] indicate that our algorithm meets these objectives in the considered network settings and is able to maintain a good structure in a mobile environment as well.

References

[1] V. Devarapalli and D. Sidhu, “MZR: a multicast protocol for mobile ad hoc networks,” vol. 3, pp. 886 – 891, 2001.

[2] Z. Haas, “A new routing protocol for the reconfigurable wireless networks,” in *IEEE 6th International Conference on Universal Personal Communications Record*, vol. 2, (San Diego, CA, USA), pp. 562–566, 1997.

[3] A. Passarella, F. Delmastro, and M. Conti, “XScribe: a stateless, cross-layer approach to P2P multicast in multi-hop ad hoc networks,” in *MobiShare ’06*, (New York, NY, USA), pp. 6–11, ACM Press, 2006.

[4] N. Carvalho, F. Araujo, and L. Rodrigues, “Reducing latency in rendezvous-based publish-subscribe systems for wireless ad hoc networks,” in *ICDCSW ’06*, (Washington, DC, USA), p. 28, IEEE Computer Society, 2006.

[5] R. Kummer, P. Kropf, and P. Felber, “Distributed lookup in structured peer-to-peer ad-hoc networks.,” in *On the Move to Meaningful Internet*

Systems 2006: CoopIS, DOA, GADA, and ODBASE (R. Meersman and Z. Tari, eds.), vol. 4276 of *Lecture Notes in Computer Science*, pp. 1541–1554, Springer Berlin / Heidelberg, 2006.

[6] R. Kummer, P. Kropf, and P. Felber, "Building multicast trees in ad-hoc networks," in *MINEMA workshop, Autonomics 2008*, (Turin, Italy), September 2008.

7 Wireless Communication in Road Traffic Control

Markus Wulff, University of Bern
mwulff@iam.unibe.ch

7.1 Introduction

With the rapidly growing density of motorised transportation of goods and people all over the world, crashes become more frequent and, sadly, deaths and injuries are common.

> At the inquest into the worlds first road traffic death in 1896, the coroner was reported to have said this must never happen again. More than a century later, 1.2 million people are killed on roads every year and up to 50 million more are injured. [1]

Another important problem is the environmental pollution which accompanies the increasing use of cars, trucks, buses and other motorised vehicles.

> Every year, 12 billion liters of fuel are wasted in traffic jams - this results in an CO2 emission of 30 billion tons. [2]

One way to reduce the negative effects of this trend is the use of modern communication technology to optimise traffic flow and increase safety for drivers and passengers.
Since the 70s, when computers became affordable and could provide enough computing power they have been used also for optimising road traffic. Today, a lot of research is done on this area. Some keywords are “intelligent highways”, “wired cars”, “car-to-car communication”, and VANETs.
In the following, selected activities on the road traffic control sector are presented.

7.2 Road Traffic Control

The common goal of all research projects/activities is the improvement of driving safety and traffic efficiency on different areas of road traffic control [3]: traffic planning, traffic controlling, traffic flow optimisation, parking management and so on.

This can be achieved in many different ways. In this report, the main focus is on wireless communication techniques between cars and between cars and infrastructure facilities.
Technical problems here are are for instance the availability and real-time guarantees some applications require. Many information like those for prevention of accidents become useless if they are not delivered in real-time. Or the high mobility of the communicating nodes, the contact might last a few seconds only, is another challenge for the developers of car-to-car communication applications.

7.3 Car2Car Communication Consortium

The car-to-car communication consortium (C2C-CC) is a non-profit organisation initiated by European vehicle manufacturers, which is open for suppliers, research organisations and other partners. The consortium is dedicated to the objective of further increasing road traffic safety and efficiency by means of inter-vehicle communications [4].
The goals of the consortium are to create and establish industry standard for C2C communication and guaranty inter-vehicle operability. Therefore, specifications, prototypes, and demonstrations of C2C systems are developed. To ensure a transnational operability of the communication techniques, the C2C-CC promotes the allocation of exclusive frequency band and pushes the harmonization of C2C standards worldwide.
Car-to-car or car-to-infrastructure communication could for instance support the following applications:

- Cooperative forward collision warning
- Pre-crash sensing/warning
- Hazardous location notification
- Enhanced route guidance and navigation
- Green light optimal speed advisory
- Merging assistance
- Infotainment

To make this concept work, a minimum number of cars must support the car-to-car communication features. For the inter-vehicle danger warning application, at least 10 % of all cars must support it to work. In case of traffic information propagation the penetration should be at least 5 %.
Depending on the actual application, further requirements exist:

- Vehicles must trust information originated by other vehicles and/or roadside units.

- The accurate relative positioning of vehicles must be known for many applications.
- Vehicles must be able to share information over distances of up to several hundred meters
- Communication over several hops must be supported by the wireless communication infrastructure.
- Providers are needed to collect and maintain traffic information.

7.4 Other projects

The GeoNet project (geographic addressing and routing for vehicular communication) [5] brings the basic results from C2C-CC to the next step by further improving these specifications and creating a baseline software implementation interfacing with IPv6. The goal is to fill the existing implementation gap of geo-addressed networking, ongoing and future projects for Cooperative Systems can maintain their focus on architecture design, application development and field trials. GeoNet is a European project with members from 6 countries, France, Germany, Austria, UK, Hungary and Spain. Two research institutes (INRIA, IMDEA-Networks), one SME (BroadBit) and four industrial partners (EFKON, Hitachi, NEC and Lesswire) are in the GeoNet consortium.
CVIS (Cooperative Vehicle Infrastructure System) [6] develops a multichannel terminal capable of maintaining a continuous Internet connection over a wide range of carriers, including cellular, mobile Wi-Fi networks, infra-red or short-range microwave channels. An open architecture connecting in-vehicle and traffic management systems and telematics services at the roadside should be created. Further goals are the development of techniques for enhanced vehicle positioning and the creation of local dynamic maps and extended protocols for vehicle, road and environment monitoring.
The Safespot [7] integrated project aims to prevent road accidents developing a “Safety Margin Assistant” that detects in advance potentially dangerous situations and extends “in space and time” drivers awareness of the surrounding environment. Furthermore, the project aims to develop key technologies like ad-hoc dynamic networking, accurate relative localisation, dynamic local traffic maps, and new generation of infrastructure-based sensing techniques.

7.5 IEEE 802.11 for car-to-car communication

Especially for the application in vehicles a new wireless communication standard is being developed. The IEEE 802.11 standard adds wireless access in the vehicular environment (WAVE) to support Intelligent Transportation Systems (ITS) applications. [8, 9] It is operating on the frequency band between 5,85 and 5.925 GHz. 802.11p includes

- the radio transmission at the physical layer
- parts of the MAC layer

The higher layers are defined by IEEE 1609 working group. As general conditions

- a velocity of up to 200 km/h,
- a distance of max. 1 km and
- a data transfer rate between 3 Mbit/s and 27 Mbit/s

are assumed.

7.6 Privacy and Security issues in VANETs

As in many other (wireless) communication systems, privacy is a main issue. Sometimes it is necessary that individuals must be distinguished and some sort of authentication is introduced. The challenge is now to find a good balance between the privacy of the user and the functionality of the applications. The less personal information a user must provide the better will be the acceptance of the new technique. Probably nobody wants to be tracked if he/she is using the car.
Possible attacks are for instance the denial of service attack e. g. to prevent warning messages. Message could also be suppressed (drop selected packages) in order to harm other road users or to try to get an advantage over other motorists. For the same reasons false information could be broadcasted of alteration attacks (delay, replay) could be used. [10]

7.7 Conclusion and Outlook

Until today, a lot of research has been done on the area of techniques and applications for wireless car-to-car or car-to-infrastructure communication. And many projects are on their way or have just been started. Different partners from industry and academia working in these projects investigating different approaches.
However, there are still some problems to be solved before these new techniques and applications will become ready for the market. First of all,

common standards need to be defined to enable transnational compatibility and manufacturer spanning working solutions.
Finally, a bootstrapping problem might occur even once the technology is ready to be sold. Who will buy a feature which is still useless because nobody else is having it installed and how can the required minimum penetration be achieved? But the chances are that solutions will be found to these problems and the road traffic will become safer and more efficient than it is today.

References

[1] World Health Organization (WHO), "Road safety: a public health issue." http://www.who.int/features/2004/road_safety/en/, 2004.

[2] Bundesministerium für Bildung und Forschung (BMBF), "Mit Hightech für mehr Klimaschutz." http://www.bmbf.de/press/2004.php, 2007.

[3] Siemens, "Trafic solutions." http://www.industry.siemens.de/traffic/, 2008.

[4] Car2Car Communication Consortium, "Car 2 car communication consortium manifesto." http://www.car-to-car.org, 2007.

[5] "GeoNet: Geographic addressing and routing for vehicular communications." http://www.geonet-project.eu/geonet, 2008.

[6] "CVIS: Cooperative vehicle infrastructure system." http://www.cvisproject.org/, 2008.

[7] R. Brignolo, "The SAFESPOT Integrated Project," in *APSN NETWORK & APROSYS INTEGRATED PROJECT, 6th Annual Conference*, (Vienna, Austria), May 12 2006.

[8] L. Armstrong and W. Fisher, "Status of Project IEEE 802.11 Task Group p." http://grouper.ieee.org/groups/802/11/Reports/tgp_update.htm, 2008.

[9] A. Festag, "Wifi für Autos," *Funkschau*, no. 13, 2005.

[10] B. Parno and A. Perrig, "Challenges in securing vehicular networks," in *Fourth Workshop on Hot Topics in Networks (HotNets-IV)*, (College Parc, MD), November 14–15 2005.

8 Pretty Good Anonymity

Ronny Standtke, ronny.standtke@gmx.net
University of Fribourg

8.1 Introduction

There are several anonymity architectures for Internet communication in use today. We evaluate these architectures and - based on the evaluation - propose a new design.

8.2 Threat models

All threat models we are going to discuss in this section share a common underlying communications model. In that model, the user, who we want to protect, communicates with peers via an independent third party responsible for protecting the user's anonymity. For each of the methods, however, the strength of the attacker is different leading to different requirements and solutions for achieving anonymity.

8.2.1 Remote attacker

In this threat model the attacker is the communication partner. This threat model is very weak but probably the most common. To protect against an attacking peer one can use a simple proxy. The proxy hides the user's identity from his communication partner.

8.2.2 Local attacker

In this threat model the attacker can analyze the traffic sent by the user, e.g. the Internet Service Provider which monitors traffic. This threat model is very common today. Encrypting the connection between user and proxy makes uncovering communication paths by address evaluation impossible.

8.2.3 Global attacker

In this threat model all traffic received and sent by a third party can be analyzed by the attacker. Such a powerful attacker can apply various strate-

gies to link the user with the respective peer and vice versa. Most of these strategies have been mentioned in [1].

End-to-end content correlation. If the attacker can read message contents at both the user and the peer side, he will be able to confirm the correspondence very easily. This can be prevented by encrypting the messages exchanged between user and third party, as described in section 8.2.2.

End-to-end timing correlation. If the attacker can record and analyze the message timestamps at both the user and the peer side, he will be able to confirm the stream endpoints with high probability. This attack is impossible to circumvent by one user alone. Therefore anonymity groups [2] have to be established. An anonymity group is formed by a number of users. They must send and receive all messages from the third party with the same timing. This way an attacker can no longer correlate message times to a single user but only to a complete anonymity group. The third party must collect and re-order all messages before forwarding. Otherwise the order of incoming and outgoing messages at the third party would be identical and thus again allow their correlation.

End-to-end data volume correlation. An attacker can also confirm endpoints of a stream by measuring the data volume that the user sends and the peer receives (and vice versa). This attack can only be prevented (as in the previous case) by an anonymity group. All anonymity group members must send data at a fixed rate to the third party and must also receive data at a fixed rate from there. If users have no (meaningful) data to be sent, they must create arbitrary, meaningless traffic (dummy traffic). The same holds true also for links from the third party to the users.

Statistical disclosure. An attacker can analyze the times when users are initiating connections to the third party, and when connections to peers are established. This way repeated communications would eventually be revealed even if the anonymity service was otherwise perfect. This was first shown by Dogan Kesdogan et al. [3]. There is also a more general, statistical variant of these attacks [4] which was validated through simulations [5]. Mathewson et al. [5] and Agrawal et al. [6] estimated how long this attack would take to succeed. This attack can only be prevented if anonymity groups are assumed to be static. This requirement strongly contradicts typical Internet usage where users regularly log on and off and is therefore considered to be nonpractical by the authors.

$n-1$ *attack.* As described in [7] the attacker isolates one specific user by simulating all other users at the third party. As the attacker knows the addressing of his own messages, he can differentiate them from the victim's messages. This attack can not be prevented by technical measures: The attacker can always try to simulate many other users or convince others

to cooperate. Technically speaking, benign users cannot be distinguished from malicious ones.
Message tampering. If an attacker is able to modify messages to or from the third party he can produce garbage messages or disrupt data streams that use integrity mechanisms. This attack can only be countered partially by using message integrity mechanisms. This way the attacker can no longer produce garbage messages but can still disrupt data streams and thus facilitate passive end-to-end attacks.

8.2.4 Third party attacker

The most powerful attacker is the (trusted) third party itself. The third party must establish a link between users and their peers to enable communication in the end. If the third party can correlate all incoming and outgoing user and peer messages it can uncover any existing communication association.
To protect against the third party itself, the latter must be split up into a distributed system. For the protection mechanisms to be effective, the individual nodes of the distributed system must not cooperate.
Because the user data must pass through all involved nodes before it reaches the communication peer, a distributed third party is relatively slow and expensive. Additional mechanisms must be established on the user's side to ensure that every part of the distributed system has access only to the particular information it needs. A simple mechanism for that purpose is to use layered encryption. This idea was first introduced by David Chaum in his seminal paper [8] describing fundamental building blocks for anonymity.
One option of distributing the third party is to use ad hoc or peer-to-peer networks where any node can join and leave the distributed system anytime. This variant is mostly used by user-driven anonymity services where each user is also partially a trusted third party for other users. There are several disadvantages to this design:

- Latency and bandwith of the anonymization service are depending on individual user connections, which possibly can be very limited.
- Whenever one node leaves the system, it breaks all the communication links that were routed over this node.
- Volunteers joining the distributed system may be prosecuted for damages done by other (malicious) users of the system. This happened

already in Germany where the Public Prosecution Service confiscated anonymization servers of private individuals [9].

- It is not possible in an efficient way to enable all necessary security mechanisms (e.g. dummytraffic) between all distributed nodes, making the system weak against global attackers.

Another option of third party distribution is to split the third party by organizational means, e.g. by distributing the service to a fixed set of independent organizations. A substantial foundation of this method is the self-obligation of the organizations not to cooperate regarding the deanonymization of users. The unconditional keeping of this self-obligation cannot be controlled by the users and therefore can be doubted. The whole idea of organizational splitting is inconsistent, because all instances have to collaborate on a technical, organizational and even financial ground just to convince users that exactly these instances do not cooperate.

As stated above, the goal of distributed systems is to ensure the anonymity of the user even against the third party itself. But there are many remaining external attacks (see section 8.2.3, especially *Statistical disclosure*, $n-1$ *attacks* and *Message tampering*). Protection against the third party only makes sense in the end, if all remaining external attacks are significantly more difficult than internal attacks, which is not the case so far. The attacking third party can just execute an external instead of an internal attack.

In summary, one could say that anonymization over a distributed system is an expensive and sometimes inconsistent method that does not reach its goal.

8.3 Introduction to Project PGA

The above evaluation of threat models, countermeasures and their implementations led to a new project at the University of Fribourg Switzerland. The project is called *PGA* (Pretty Good Anonymity). In this project we are designing and implementing a new architecture for anonymous Internet communication by avoiding the weak points of other solutions.

The main idea is to use a single, trusted third party, similar to running a mix cascade with just one single mix. This architecture provides anonymity against a global attacker, drastically improves the performance and simplifies all involved protocols and mechanisms.

References

[1] S. J. Murdoch and G. Danezis, "Low-cost traffic analysis of Tor," in *Proceedings of the 2005 IEEE Symposium on Security and Privacy*, IEEE CS, May 2005.

[2] A. P. Reiner Sailer, Hannes Federrath, "Security functions in telecommunications – placement and achievable security," in *Proceedings of Multilateral Security in Communications* (G. Müller and K. Rannenberg, eds.), Addison-Wesley, 1999.

[3] D. Kesdogan, D. Agrawal, and S. Penz, "Limits of anonymity in open environments," in *Proceedings of Information Hiding Workshop (IH 2002)* (F. Petitcolas, ed.), Springer-Verlag, LNCS 2578, October 2002.

[4] G. Danezis, "Statistical disclosure attacks: Traffic confirmation in open environments," in *Proceedings of Security and Privacy in the Age of Uncertainty, (SEC2003)* (Gritzalis, Vimercati, Samarati, and Katsikas, eds.), (Athens), pp. 421–426, IFIP TC11, Kluwer, May 2003.

[5] N. Mathewson and R. Dingledine, "Practical traffic analysis: Extending and resisting statistical disclosure," in *Proceedings of Privacy Enhancing Technologies workshop (PET 2004)*, vol. 3424 of *LNCS*, pp. 17–34, May 2004.

[6] D. Agrawal, D. Kesdogan, and S. Penz, "Probabilistic Treatment of MIXes to Hamper Traffic Analysis," in *Proceedings of the 2003 IEEE Symposium on Security and Privacy*, pp. 16–27, May 2003.

[7] D. Kesdogan, J. Egner, and R. Büschkes, "Stop-and-go MIXes: Providing probabilistic anonymity in an open system," in *Proceedings of Information Hiding Workshop (IH 1998)*, Springer-Verlag, LNCS 1525, 1998.

[8] D. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Communications of the ACM*, vol. 4, February 1981.

[9] "Staatsanwaltschaft beschlagnahmt Anonymisierungsserver." Website, 1999. Available online at `http://www.golem.de/0609/47702.html`.

9 Applicating Software Transactional Memory on Parallel Event Processing

Heiko Sturzrehm, University of Neuchâtel
heiko.sturzrehm@unine.ch

9.1 Introduction

In event stream processing (ESP) applications, events flow through a network of components that perform various types of operations, e.g., filtering, aggregation, transformation. When the operation only depends on the input events, one can trivially parallelize its processing by replicating the associated components. This is not possible, however, with stateful components or when there exist dependencies between the events.
When processing is stateful, one cannot simply improve performance by replicating the components. First, multiple copies of the same component would need to maintain a consistent replicated state, which is in general non trivial and usually adds significant overhead. Second, events must most often be processed in a specific order, either because they have dependencies with one another or because the effect on the component's state depends on the processing order. It is important to note that, even when events arrive in the right order at stateful component, they typically *cannot* be processed in parallel.
In this paper, we consider ESP systems with components connected in a

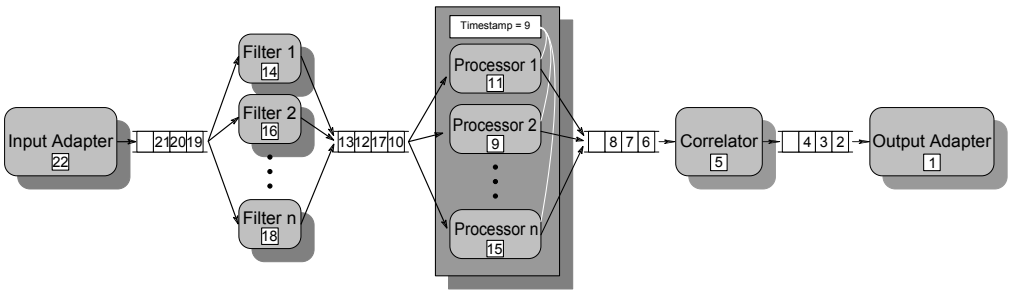


Figure 9.1: A simple typical ESP application network.

cascade (see Figure 9.1). We are interested in parallelizing the operations of the components by exploiting the processing capabilities of multi-core architectures. Some components are stateless and can be trivially parallelized, but in doing so they may reorder the events.
Other components are stateful and must typically (but not necessarily) process events in order. We assume that the order of events is determined when they enter the system, e.g., by associating monotonically increasing logical timestamps with each of them.
Our approach consists of using speculative execution to process events in parallel that should normally be processed sequentially, even when they are received out-of-order. In short, we use an underlying Software Transactional Memory (STM) [1] infrastructure to optimistically process the events in the context of transactions. STM provides the concept of transactions on the programming language level. It has become very popular recently as an approach to develop concurrent programs for many-core CPUs. So far, however, STMs do not provide features to pre-order transactions that execute in parallel. Hence, we have extended the design of the STM so that we can pre-assign commit timestamps to transactions, effectively imposing an order in which they need to complete. Transactions will execute in parallel but may have to delay their completion (i.e., their commit) when ordering is required.

9.2 Enhanced ESP Components

The key intuition behind our approach is that an STM will dynamically detect dependencies between events, if any, and will sequentialize the processing only when necessary (possibly delaying, or aborting and restarting some transactions). Without speculative execution, ESP components would not only have to execute events sequentially one at a time, but may have to *wait* idle when events are not received in the right order. As we shall see, the increased parallelism of our approach yields substantial performance benefits.
There is a thin line between being optimistic and being overly optimistic. Speculative execution of a transaction that has a high likelihood of having to ultimately abort (e.g., because it is “too much” out-of-order) can prove to be counterproductive.
A stateful component enhanced with support for speculative execution is similar to a regular component from the outside: it supports input and output queues. The main difference is that events in the input queues may be unsorted. In the output queue, they will be sorted according to their

timestamps.
An enhanced ESP component has several threads working in parallel. The number of threads typically depends on the processing capabilities (number of cores) of the system that hosts the component. Each thread can access the input queues and retrieve events to be processed. The manipulation of the event is performed in the context of a transaction, which means that modifications are invisible to other threads until the transaction commits.
The STM-enhanced components use an underlying time-based STM (TinySTM [2]) that utilizes a shared commit counter to maintain consistent snapshots of memory locations read by transactions without incurring the cost of incremental validation. Commit timestamps are essentially used to linearize transactions and detect whether the content of a memory location can be safely accessed (i.e., is consistent with the transaction execution order). We rely on the use of commit timestamps in our speculative parallelization approach.
Unlike the classical behavior of an STM, transactions cannot complete in any order and threads do not automatically commit their transactions. Instead, transactions have pre-assigned commit timestamps (determined according to the timestamp of the associated events). A thread checks if a transaction can commit by comparing its timestamp with the current commit counter of the component. If both are equal, the transaction commits and the event can be sent to the output queue. Otherwise, the whole transaction is suspended and inserted in a waiting list. Each time a new event is processed, the list is checked to see if a waiting transaction can now be committed. It may happen that a transaction in the waiting list is aborted due to a conflict with another transaction; in that case it is restated.
As for a regular ESP component, the developer of an enhanced ESP component must provide a function `execute()` that implements the actual event processing. In addition, s/he may provide two functions specific to transactional operation: `onCommit()` executes upon successful completion of a transaction, while `postCommit()` runs after commit. The main difference between both functions is that executions of the former are serialized (necessary, for instance, when inserting the processed event in the output queue) while multiple instances of the latter can execute in parallel (e.g., to gather statistics or perform cleanup operations). This difference has strong implications on performance, as we shall discuss later. Figure 9.2 illustrates the operations supported by the enhanced ESP component.

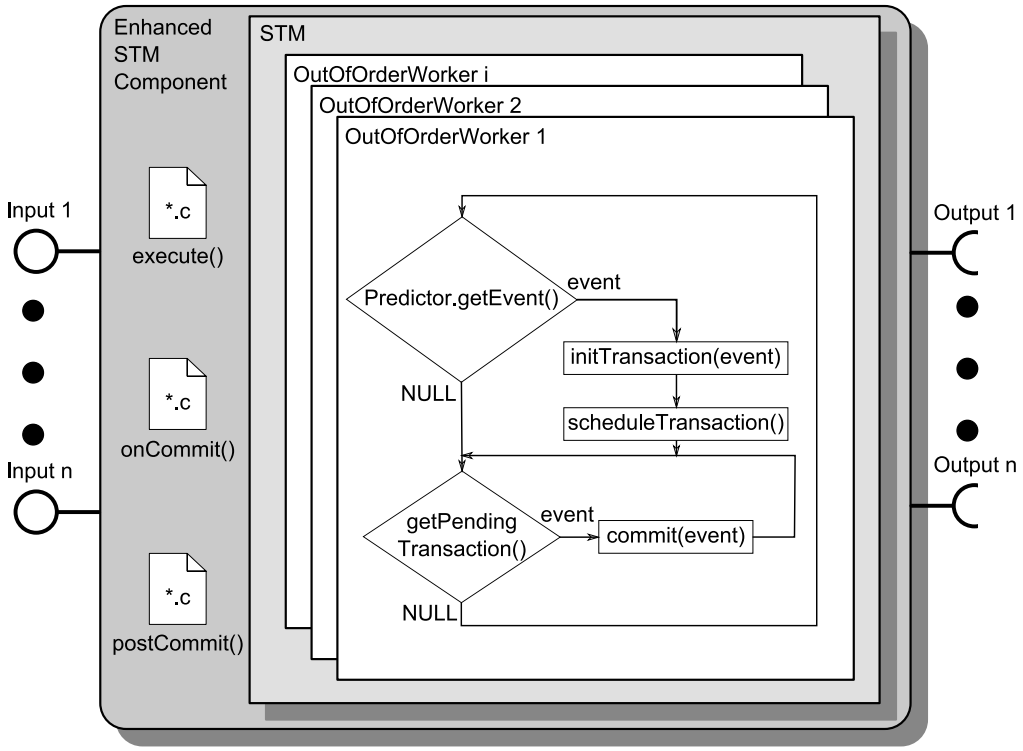


Figure 9.2: Enhanced ESP component internals.

9.3 Conclusion and Outlook

We have designed a speculative execution environment for Event Stream Processing (ESP) components. Events that are received out of order and/or conflict with one another are optimistically processed in parallel. To ensure that the system remains in a consistent state despite parallelization, we use an underlying Software Transactional Memory (STM) that was extended to account for the specificities of ESP. In particular, the STM can pre-assign timestamps to transactions to drive the commit order.

References

[1] M. Herlihy and J. E. B. Moss, “Transactional memory: Architectural support for lock-free data structures,” in *Proceedings of the Twentieth Annual International Symposium on Computer Architecture*, 1993.

[2] P. Felber, C. Fetzer, and T. Riegel, “Dynamic Performance Tuning of Word-Based Software Transactional Memory,” in *Proceedings of the 13th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP)*, 2008.

10 Developing and Deploying Large-scale Distributed Applications using SPLAY

Étienne Rivière and Pascal Felber, University of Neuchâtel
etienne.riviere@unine.ch - pascal.felber@unine.ch

PhD thesis project of Lorenzo L. Leonini, University of Neuchâtel
lorenzo.leonini@unine.ch

SPLAY *is further described in the research report [1]. A demonstration has been presented to the IEEE P2P 2008 conference [2] and the website* `http://splay-project.org` *presents an online demo, a live CD and* SPLAY *code under the GPL license.*

Abstract
SPLAY is an integrated system that facilitates the complete chain of distributed systems evaluation, from design and implementation to deployment and experiments control. Algorithms are expressed in a concise, yet very efficient, language based on Lua. Implementations in SPLAY are highly similar to the pseudo-code usually found in research papers. SPLAY eases the use of any kind of testbeds, e.g., PlanetLab, ModelNet clusters, or non-dedicated platforms such as networks of workstations. Using SPLAY and PlanetLab, this demonstration highlights a complete evaluation chain of an epidemic protocol and a churn-driven experiment using the Pastry DHT.

10.1 Introduction

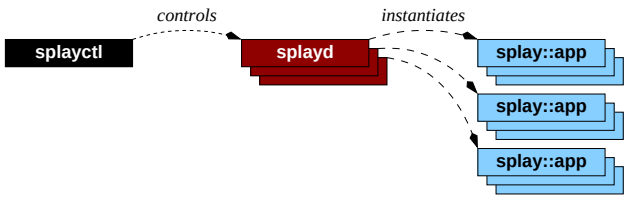
Evaluating large-scale distributed applications is a higly complex, time-consuming and error-prone task. One of the main difficulties stems from the lack of appropriate tools for quickly prototyping, deploying and evaluating algorithms in real settings. Several dedicated testbeds are available that can be leveraged for better evaluations of these systems: PlanetLab [3], Everlab [4], or network emulators such as ModelNet [5] or Emulab [6]. Meanwhile, non dedicated testbeds such as networks of idle workstations usually found in research labs or schools, are difficult to use for distributed systems experiments, as one usually require access rights

that are not easily granted by the admistrators.
All these testbeds are appealing as they allow real or realistic experiments to be conducted, but they are not used as systematically as they should. Indeed, strong technical skills are typically necessary to develop, deploy, execute and monitor applications for such testbeds. The learning curve is also usually slow. Technical difficulties are even higher if one wants to deploy an experiment on several testbeds at the same time, for instance a population of peers on adversial testbeds such as PlanetLab and another population of peers on a local ModelNet cluster.
A side effect of these difficulties is that the performance of evaluated systems is greatly impacted by the technical quality of their implementation, overshadowing the underlying algorithm's intrinsic qualities. This may in turn make comparisons unsound or irrelevant.
All these observations call for novel development-deployment systems that would straightforwardly exploit these testbeds and bridge the gap between algorithmic specifications and live systems. Researchers could use such a system for real evaluations instead of simulations, and teachers to focus their lab work on the core of distributed programming, i.e., algorithms and protocols, letting students experience distributed systems implementation in real settings.

Related work There were a number of proposals in the litterature for evaluation or deployment frameworks. The former includes systems such as Mace [7] (C++ extension) or P2 [8] (dedicated declarative language) that allow developers to express algorithms in a high-level language, hiding most of the complexity. These languages, however, do not provide any support for deploying the applications or controling the behavior of an execution. The latter include deployment tools such as Plush [9] or Weevil [10]. Both allows the creation of deployment scripts for testbeds such as PlanetLab. Using the user's description of the experiment, they instantiate the applications on the testbed or create scripts for this task. Nonetheless, these systems do not allow deployment on non-dedicated testbeds. They do not allow either for complex deployment, for instance involving multiple testbeds or complex network scenarios, that SPLAY allows natively and without efforts.

10.2 SPLAY architecture

A SPLAY infrastructure is composed of a controller, daemons and sandboxed application processes:



The controller is a trusted entity that manages the deployment and execution of SPLAY application. Lightweight daemons processes (`splayd`) are responsible for instantiating sandboxed applications processes, as instructed by the controller. A `splayd` can run multiple sandboxed application instances. Sandboxing is a primary feature of SPLAY; it allows the administrator who deploy the daemons to restrict the usage of local resources (memory, disk, network). Applications instances have absolutely no direct access to the hosting system. We elaborate more on sandboxing while describing the language and the libraries.

Language and libraries SPLAY is based on Lua [11], a highly efficient scripting language. A dedicated language is needed for several reasons. First, as we need to support non dedicated testbeds, sandboxing is a sound basis. Lua support for scoping and first-order functions allows us to redefine all standard library functions to impose boundaries on resource usage. This allows us to ensure that buggy or ill-behaved code will not harm the system hosting the application. Second, it is necessary to support a large number of application processes on a single host. Our tests have shown that Lua for SPLAY is able to run more than 1,250 instances of Pastry [12] on a single dual-core machine with 2GB of memory. The others reasons include the possibility to run applications on any hardware or host OS, and the performance of the libraries.
We provide an extensive set of libraries for developping distributed applications with SPLAY, including: networking libraries with sandboxed RPCs and message passing with UDP or TCP (including automatic serialization); a sandboxed virtual filesystem; threading based on coroutines and event-based programming; a logging library to seamlessly report statistics about running applications.
An important goal of SPLAY is to allow application developers to write concise, readable code that highly resemble pseudo-code found in research papers. During this demonstration, we will code a self-contained epidemic diffusion protocol. We did implement a set of distributed systems using SPLAY. Figure 10.2 presents the implementation length (in term of lines of code) of a subset of them. This metric gives a rough yet clear idea of the readability and maintanability of the systems developped with SPLAY.

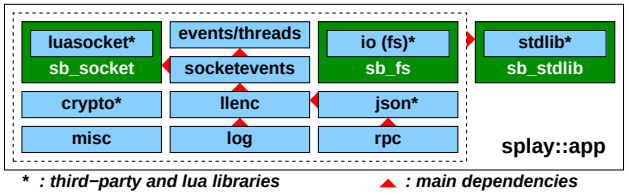


Figure 10.1: SPLAY libraries.

All these numbers take lines delimiters into account, and we did not try to reduce the line count in a way that would impair readability. The Chord [13] and Pastry [12] DHTs are respectively 101 and 265 lines (including fault tolerance and initialization). Middleware using Pastry such as Scribe [14] and Splitstream [15] are 79 and 58 lines, respectively. BitTorrent [16] uses 420 lines, most of which is needed for deciphering the binary format used by other BitTorrent clients (our client participates in existing swarms and is fully compatible with existing clients). Gossip-based protocols such as membership management using Cyclon [17] or epidemic broadcast [18] are particularly concise. Finally, “trees” stands for a multiple, balanced trees based diffusion structure as presented in [19]. Notably for the latter, the performance of the SPLAY-based implementation is much similar to the one of a hand-tuned C implementation made by the authors. Details can be found in [1].

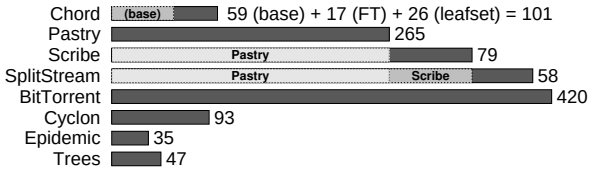


Figure 10.2: Implementation length (Lines Of Code - LOC) for various distributed systems built within SPLAY.

10.2.1 Controlling deployments

SPLAY provides either a command line or Web-based interface. It allows to select the daemons that will host an experiment based on geographical location, performance, load, resource limitations, etc. An interesting

feature is the churn management module. To allow fair comparison of systems under the same conditions, and since the natural churn in PlanetLab is not always sufficient to derive a protocol's behavior, SPLAY can reproduce the dynamics of a system, either from a synthetic description (for instance, creating massive churn, steady increase, etc.) or from a real trace (e.g., [20]).

References

[1] L. L. Leonini, E. Rivière, and P. Felber, "Splay: Distributed systems evaluation made simple (or how to turn ideas into live systems in a breeze)," Tech. Rep. RR-I-08-05.1, Université de Neuchâtel, Neuchâtel, Suisse, jun 2008.

[2] L. L. Leonini, E. Rivière, and P. Felber, "P2p experimentations with splay: from idea to deployment results in 30 min.," in *Proceedings of the Eigth International Conference on Peer-to-Peer Computing (P2P'08), demo sessions.*, (Aachen, Germany), IEEE, sep 2008.

[3] A. Bavier, M. Bowman, B. Chun, D. Culler, S. Karlin, S. Muir, L. Peterson, T. Roscoe, T. Spalink, and M. Wawrzoniak, "Operating system support for planetary-scale network services," in *NSDI'04*.

[4] E. Jaffe, D. Bickson, and S. Kirkpatrick, "Everlab: a production platform for research in network experimentation and computation," in *LISA'07*, pp. 1–11.

[5] A. Vahdat, K. Yocum, K. Walsh, P. Mahadevan, D. Kostic, J. Chase, and D. Becker, "Scalability and accuracy in a large-scale network emulator," in *Proc. of OSDI*, pp. 271–284, 2002.

[6] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar, "An integrated experimental environment for distributed systems and networks," in *Proc. of OSDI*, (Boston, MA), pp. 255–270, Dec. 2002.

[7] C. E. Killian, J. W. Anderson, R. Braud, R. Jhala, and A. M. Vahdat, "Mace: language support for building distributed systems," in *Proc. of PLDI '07*, pp. 179–188.

[8] B. T. Loo, T. Condie, J. Hellerstein, P. Maniatis, T. Roscoe, and I. Stoica, "Implementing declarative overlays," in *SOSP'05*.

[9] J. Albrecht, R. Braud, D. Dao, N. Topilski, C. Tuttle, A. C. Snoeren, and A. Vahdat, "Remote Control: Distributed Application Configuration, Management, and Visualization with Plush," in *LISA'07*.

[10] Y. Wang, M. J. Rutherford, A. Carzaniga, and A. L. Wolf, "Automating experimentation on distributed testbeds," in *Proc. 20th ASE*, (Long Beach, California), Nov. 2005.

[11] R. Ierusalimschy, L. de Figueiredo, and W. Celes, "The implementation of lua 5.0," *Journal of Universal Computer Science*, vol. 11, no. 7, pp. 1159–1176, 2005.

[12] A. Rowstron and P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems," in *Middleware'01*.

[13] I. Stoica, R. Morris, D. Liben-Nowell, D. Karger, M. Kaashoek, F. Dabek, and H. Balakrishnan, "Chord: a scalable peer-to-peer lookup protocol for internet applications," *IEEE/ACM Trans. Netw.*, vol. 11, no. 1, pp. 17–32, 2003.

[14] M. Castro, P. Druschel, A.-M. Kermarrec, and A. Rowstron, "SCRIBE: A large-scale and decentralized publish-subscribe infrastructure," *IEEE J. Sel. Areas Commun.*, 2002.

[15] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "SplitStream: High-bandwidth multicast in a cooperative environment," in *Proc. of 19th SOSP*, Oct. 2003.

[16] B. Cohen, "Incentives to build robustness in BitTorrent," tech. rep., `http://www.bittorrent.org/`, May 2003.

[17] S. Voulgaris, D. Gavidia, and M. van Steen, "Cyclon: Inexpensive membership management for unstructured p2p overlays," *J. Network Syst. Manage.*, vol. 13, no. 2, 2005.

[18] K. P. Birman, M. Hayden, O. Ozkasap, Z. Xiao, M. Budiu, and Y. Minsky, "Bimodal multicast," *ACM TOCS*, vol. 17, no. 2, pp. 41–88, 1999.

[19] E. Biersack, P. Rodriguez, and P. Felber, "Performance analysis of peer-to-peer networks for file distribution," in *Proc. of 5th QofIS*, pp. 1–10, Sept. 2004.

[20] R. Bhagwan, S. Savage, and G. M. Voelker, "Understanding availability," in *IPTPS'03*.

11 Zero-Knowledge Proofs Of Knowledge

Stephan Krenn, stephan.krenn@bfh.ch
University of Fribourg and Bern University of Applied Sciences

11.1 Introduction

An honest-verifier zero-knowledge proof of knowledge (ZK-PoK) is a two party protocol, where a prover P wants to convince a verifier V that he knows some secret, in a way that satisfies three properties: first, if both of them behave honestly, the verifier will always accept. Second, if the prover does not know the secret, he won't be able to make the verifier accept with non-negligible probability. And finally, even if the verifier behaves maliciously, he won't gain any useful information on the secret. For a formal definition we refer to [1].
Although there are results that such proof systems exist for all relations in NP, [2], this is only of theoretical interest, as most of those protocols are too inefficient for being used in real world applications. Further, even if efficient protocols are known, their adaption to concrete problems is a time consuming, and error-prone task. Hence, an automatic translation of high-level descriptions into efficient executable code would be desirable.

11.2 Theoretical background

Essentially, all efficient ZK-PoK protocols used in practice today are based on so called Σ-protocols. These are three-move protocols consisting of the first message from the prover, called *commitment*, a *challenge* uniformly chosen at random by the verifier and the corresponding *response* from the prover again. What is typically being proven is the knowledge of a preimage under a homomorphism; for an illustration see Figure 11.2, where c^+ is smaller than the smallest prime dividing the order of $\mathcal{H}$. The notation $k \in_R \mathcal{G}$ denotes that k is randomly chosen in $\mathcal{G}$ following the uniform distribution.
In such protocols the prover can cheat with a probability of $1/(c^+ + 1)$ in each run of the protocol. By repeating the protocol, this can be made arbitrarily small. If the order of $\mathcal{H}$ is known, e.g. if $\mathcal{H}$ is an elliptic curve group, c^+ can be chosen accordingly large, yielding a small cheating-probability.

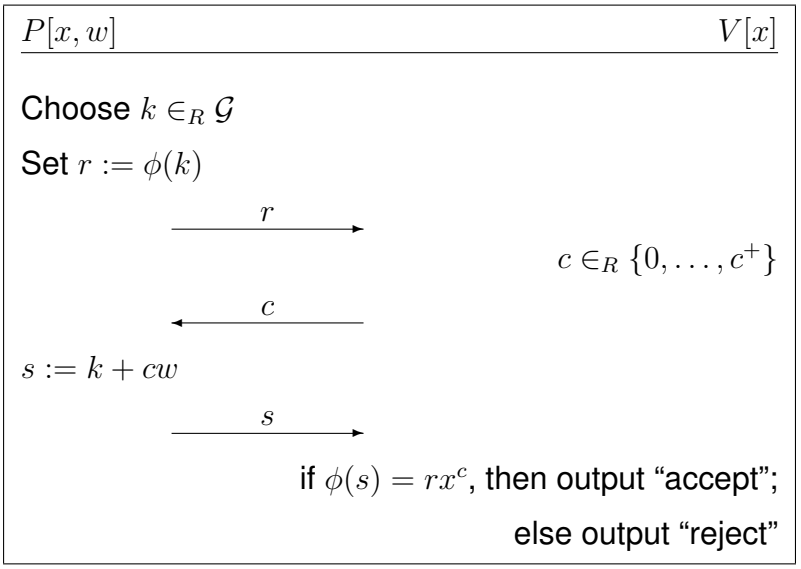


Figure 11.1: Σ-protocol for homomorphisms with a finite domain

In this case, often one single run of the protocol is enough. But if the order of $\mathcal{H}$ is not known, one has to set $c^+ := 1$, resulting in inefficient protocols; hence, in this situation other techniques have to be applied.
Such protocols play an important role in applied cryptology, where they are used as main building blocks. Examples are efficient secure multi-party computation [3], secure watermark detection [4] and identification schemes [5]. In some of those applications, it's necessary to prove knowledge of multiple preimages (AND-proofs), knowledge of one out of a set of preimages (OR-proofs), or a combination of both.
To obtain efficient and sound proof protocols, a close collaboration of a cryptographer, who designs the protocol, and a programmer, who does the implementation, is needed. Most often, the design happens on a very abstract level, based on a notation due to [6]. Such an expression looks for instance like this:

$$ZPK\left[(\alpha, \beta, \gamma) : x_1 = \phi_1(\alpha) \quad \wedge \quad x_2 = \phi_2(\beta, \gamma) \quad \wedge \quad \gamma = \alpha \cdot \beta\right]$$

This notation means, that knowledge of α, β, γ satisfying the terms on the right hand side has to proven, where x_1, x_2 as well as ϕ_1, ϕ_2 are publicly known.
To ease this collaboration, it would be desirable to automate the translation of a high-level description into efficient JAVA- or C-code, depending on the environment the protocol is supposed to run in.

A first approach into this direction has been done in [7], and was extended in [8]. Yet, both of them only consider the following situation as basic building blocks: given a group-homomorphism $\phi : \mathcal{G} \rightarrow \mathcal{H}$ with finite domain $\mathcal{G}$ and known-order co-domain $\mathcal{H}$, and a public value $x \in \mathcal{H}$, the prover wants to convince the verifier that he knows an $w \in \mathcal{G}$, such that $x = \phi(w)$. Further, expressions containing ANDs and ORs as well as linear constraints among secret preimages can be proven.
Although, by combining these techniques also multiplicative and polynomial relations among the secrets can be proven, it turns out that this is not sufficient for practice, as unknown order groups play an important role, among others, in identity escrow schemes or anonymous credential systems, see [9, 10].
Especially exponentiation homomorphisms, i.e. $\phi(w_1, \ldots, w_n) = h_1^{w_1} \cdots h_n^{w_n}$ with $h_1, \ldots, h_n \in \mathcal{H}$ for an unknown order group $\mathcal{H}$ play an important role. Most often, $\mathcal{H}$ will be a class group or an RSA-group. For these homomorphisms, several solutions can be found in literature, e.g. [8, 11, 12], although some of them do not yield proofs of knowledge any more, but, loosely speaking, something weaker.
We want to note that usually one only considers Σ-protocols which are honest-verifier zero-knowledge, as there are standard solutions to transform them into proof systems, which are also secure for malicious verifiers. Further, each such protocol can easily be turned into a digital signature scheme [13].

11.3 Conclusion and future work

To make the application of zero-knowledge proofs easier and more usable in practice, an automation of the translation from a high-level description into source code is necessary. Further, this will raise the reliability of those protocols, as misunderstandings among cryptographer and programmer can be avoided. That this is possible has been shown in [7, 8], but as discussed, this can only be seen as a successful proof of concept.
One main issue for future work is to consolidate the theory for known and unknown order groups, in order to get a unified approach to handle proofs of knowledge of preimages for arbitrary homomorphisms. Further, to raise efficiency of the output protocols, we have to automate the choice of the most appropriate solution for specific applications, taking into account costs of computations, data transfer or storage, depending on where the protocol is going to be run. This also includes finding short addition chains in real time for efficiently evaluating exponentiation homomorphisms, or

finding *smooth* secret sharing schemes [14] for given access structures.
The current version of the compiler demands that the input is formed correctly, especially that the verifier's challenge set is chosen appropriately. In its current state, i.e. for known order groups, this can be checked manually. Yet, this tasks becomes much more challenging when the compiler becomes more comprehensive. Hence, an automated verification of the security of the output is desirable, i.e. missformed inputs should be spotlighted automatically.
Other future work on the compiler will allow the usage of pre-implemented and user-written macros in order to increase the flexibility of the tool. Such macros could include commonly used homomorphisms such as power homomorphisms or exponentiation homomorphisms, as well as widely spread groups such as RSA-groups or groups over elliptic curves.

References

[1] M. Bellare and O. Goldreich, "On defining proofs of knowledge," in *Advances in Cryptology – CRYPTO 92*, vol. 740 of *Lecture Notes in Computer Science*, pp. 390–420, Springer, 1993.

[2] O. Goldreich, S. Micali, and A. Wigderson, "Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems," *Journal of the ACM*, vol. 38, no. 1, pp. 691–729, 1991. Preliminary version in 27th FOCS, 1986.

[3] Y. Lindell, B. Pinkas, and N. Smart, "Implementing two-party computation efficiently with security against malicious adversaries," in *Security and Cryptography for Networks – SCN 2008*, vol. 5229 of *Lecture Notes in Computer Science*, pp. 2–20, Springer, 2008.

[4] A. Adelsbach, M. Rohe, and A.-R. Sadeghi, "Complementing zero-knowledge watermark detection: Proving properties of embedded information without revealing it," *Multimedia Systems*, vol. 11, no. 2, pp. 143–158, 2005.

[5] C. Schnorr, "Efficient signature generation by smart cards," *Journal Of Cryptology*, vol. 4, no. 3, pp. 161–174, 1991.

[6] J. Camenisch and M. Stadler, "Efficient group signature schemes for large groups (extended abstract)," in *Advances in Cryptology*

– *CRYPTO 97*, vol. 1233 of *Lecture Notes in Computer Science*, pp. 410–424, Springer, 1997.

[7] T. Briner, "Compiler for zero-knowledge proof-of-knowledge protocols," Master's thesis, ETH Zurich, 2004.

[8] E. Bangerter, J. Camenisch, S. Krenn, A.-R. Sadeghi, and T. Schneider, "Automatic generation of sound zero-knowledge protocols." 2008.

[9] J. Camenisch and A. Lysyanskaya, "An efficient system for non-transferable anonymous credentials with optional anonymity revocation," in *Advances in Cryptology – EUROCRYPT 01*, vol. 2045 of *Lecture Notes in Computer Science*, pp. 93–118, Springer, 2001.

[10] J. Camenisch and A. Lysyanskaya, "Dynamic accumulators and application to efficient revocation of anonymous credentials," in *Advances in Cryptology – CRYPTO 02*, vol. 2442 of *Lecture Notes in Computer Science*, (London, UK), pp. 61–76, Springer, 2002.

[11] E. Bangerter, J. Camenisch, and U. Maurer, "Efficient proofs of knowledge of discrete logarithms and representations in groups with hidden order," in *8th International Workshop on Practice and Theory in Public-Key Cryptography – PKC 2005*, vol. 3386 of *Lecture Notes in Computer Science*, pp. 154–171, Springer, 2005.

[12] I. Damgård and E. Fujisaki, "A statistically-hiding integer commitment scheme based on groups with hidden order," in *Advances in Cryptology – ASIACRYPT 02*, vol. 2501 of *Lecture Notes in Computer Science*, pp. 77–85, Springer, 2002.

[13] A. Fiat and A. Shamir, "How to prove yourself: practical solutions to identification and signature problems," in *Advances in Cryptology – CRYPTO 86*, vol. 263 of *Lecture Notes in Computer Science*, pp. 186–194, Springer, 1987.

[14] R. Cramer, I. Damgård, and B. Schoenmakers, "Proofs of partial knowledge and simplified design of witness hiding protocols," in *Advances in Cryptology – CRYPTO 94*, vol. 839 of *Lecture Notes in Computer Science*, pp. 174–187, Springer, 1994.

12 Achieving Fairness and Efficiency in RSS Publish/Subscribe Systems by Leveraging Physical and Semantical Proximities

Étienne Rivière, University of Neuchâtel
etienne.riviere@unine.ch
Joint work with:

- Indranil Gupta, University of Illinois at Urbana-Champaign, IL, USA.
- Anne-Marie Kermarrec, INRIA Rennes-Bretagne Atlantique, France.
- Jay A. Patel, University of Illinois at Urbana-Champaign, IL, USA.

Disclaimer. This report only contains the abstract of the presented talk. For confidentiality reasons and due to UIUC and INRIA publishing policies, only a small abstract can be published in this report. A publication on the topic is under reviewing for the Computer Networks Jounal (Elsevier Publishing). Interested readers can claim the preprint from the author.

Abstract
In this talk, we present the design, implementation and evaluation of RAPPEL, a peer-to-peer feed-based publish-subscribe service. By using a combination of probabilistic and gossip-like techniques and mechanisms, RAPPEL provides noiselessness, i.e., updates from any feed are received and relayed only by nodes that are subscribers of that feed. This leads to a fair system: the overhead at each subscriber node scales with the number and nature of its subscriptions. Moreover, RAPPEL incurs small publisher and client overhead, and its clients receive updates quickly and with low IP stretch. To achieve these goals, RAPPEL exploits "interest locality" characteristics observed amongst real multi-user multi-feed populations. This is combined with systems design decisions that enable nodes to find other subscribers, and maintain efficient network locality-aware dissemination trees. We evaluate RAPPEL via both trace-driven simulations and a PlanetLab deployment. The experimental results from the PlanetLab deployment show that RAPPEL subscribers receive updates within hundreds of milliseconds after posting. Further, results from the trace-driven simulator match our PlanetLab deployment, thus allowing us to extrapolate RAPPEL's performance at larger scales.

13 Immune System Based Intrusion Detection

Christoph Ehret, University of Fribourg
christoph.ehret@unifr.ch

13.1 Introduction

Intrusion detection systems (IDS) are nowadays a very important component of the global IT security architecture of every company which is concerned with data and sensitive systems. Even if a lot of research was already done on this topic, the perfect IDS has still not been found and it stays a hot and challenging area in computer security. An emerging idea in IDS design is to use the human immune system (HIS) as a model: the threats and intrusions in IT systems can basically be compared to human diseases with the difference that the human body has an effective way to deal with them, what still need to be designed for IT systems. The immune system is intuitively a good candidate with all its features and similarities with the intrusion detection systems properties. An interesting challenge is to see how we can implement the different features of the HIS to improve the IDS.
We will first present the common design of the intrusion detection systems followed by a brief overview of the immune system. Finally we discuss similarities between IDS and the immune system and enumerate interesting features our immune based IDS model should have.

More on this subject can be found in [1].

13.2 Discussion

13.2.1 Intrusion Detection Systems

An intrusion detection system can be compared with a house burglar alarm: if somebody tries to enter illegally in the house, one of the sensors will detect it what will trigger the alarm bell and alert the house owner and the police. Similarly, if somebody tries to compromise the confidentiality, the integrity or the availability of a computer system or network, or tries to break the security protections, an intrusion detection system will alert the system owner and the security team [2]. Intrusion detection is the process

of monitoring and analysing events of a computer system or network and tries to find intrusions. Events like trying to break into a system from the Internet using software exploits or trying to gain higher privileges on a system are representative events that will be recognized as an intrusion.

As IDS were already presented in details last year in the *RVS Retreat 2007 at Quarten* [3], we will shortly resume the important features of the IDS and focus our discussion on the comparison between host based IDS (HIDS) and network based IDS (NIDS) within table 1.
Intrusion detection systems have two different approaches to detect intrusions : misuse and anomaly detection. The former uses signatures to detect intrusions what has the advantages of a low false-positives rate and the possibility to name attacks, but yet unseen attacks will not be detected as it does not exist any signatures to identify them; the latter tries to detect intrusions by looking at anomalies like high CPU or network traffic load, listening processes on uncommon ports, strange user behavior or strange processes, what has the advantage to possibly detect new kind of attacks but has the drawback of an often quite high false-positives rate.
Intrusion detection sensors can be placed on a host itself, what is called host based IDS, or on a network node, what is called network based IDS. In table 1 we will compare both kind of IDS regarding important functional features of an intrusion detection system.
Table 1 will help us to understand our ideas about an immune system based intrusion detection system which will be discussed in section 13.2.3.

13.2.2 Immune System features

The human immune system (HIS) is quite complex and elaborate. The defence of the HIS is organised in different layers, mainly the exterior defences, which are biochemical and physical barriers like for example skin or bronchi, the physiological barrier, where pH and temperature provide inappropriate living conditions for pathogens, the innate system and finally the adaptive system. Every layer has different defence mechanisms and stops different types of pathogens. We will not go into more immunological details as this was already covered in last year technical report [3]. We will briefly enumerate the different features of the immune system that we would like to have in an IDS in table 2; these features will be discussed more in details in section 13.2.3.
We do not have to forget that when we use a model, in our case an immunology model based on self and non-self recognition, it could be incom-

Features	**HIDS**	**NIDS**
Management	Harder to manage due to the heterogeneity of the environment and its high number in large networks with many hosts	Simple to manage due to its homogeneity and a few NIDS are sufficient to monitor a large network with many hosts
Analyse encrypted network traffic	YES	NO
IDS evasion techniques	Harder to perform than on NIDS	Evasion techniques like fragmentation will easily work with NIDS when they have no possibility to reconstruct locally the fragmented network packets
Knows if an attack was successful or not on a host	YES	NO
Protection against targeted attacks	Can be disabled during the attack of a host or by specific denial-of-service attacks	Easier than HIDS to protect against targeted attacks and can run in stealth mode
Detects large network attacks	NO	YES
Uses computing resources of the monitored host	YES	NO

Table 1: Advantages and disadvantages of HIDS and NIDS

Immune system features	
Multilayered (defense in depth)	Diversity
Distributed and autonomous	Integration with other systems
Fault tolerant	Adaptive
Pattern recognition	Anomaly detection
Interaction between immune cells	

Table 2: Interesting features of the immune system for the design of IDS

plete or wrong and we would again find these errors in our IDS model. In immunology [4] the self and non-self theory is not the only one that tries to explain the immune system; the new *danger theory* [5] is also another model that uses other assumptions.

13.2.3 How the immune system based IDS should be

In this section we will discuss what we can use from the immune system to design a more efficient intrusion detection system model.
If we need to choose between a HIDS or NIDS placement strategy and compare this to the IS, we clearly see that the HIDS is the best choice: the immune system itself is running inside the body it protects and not somewhere in the air; nevertheless, our IDS should be able to communicate with other hosts on a network and with other security systems, thinking at the ”integration with other systems” feature of table 2, like firewall or log analyzer using for example the Intrusion Detection Message Exchange Format (IDMEF) protocol . As the IS has both anomaly and misuse detection, we should design a hybrid IDS that also has both capabilities; the anomaly based detectors could be trained using the Artificial Immune System paradigm [6] with the self-nonself model [7]. When the IS finds the good antibody to fight a pathogen, this antibody will turn into a memory cell; our IDS should also be able to have some kind of automatic generation of signatures engine. To reduce the false-positives rate we could use the co-stimulation mechanism of the IS : a pathogen can be destroyed only if two different immune cells have recognized it as pathogen.
When we are ill with a headache and fever, we will of course stay in bed and try to recover as quick as possible. When an intrusion happens, our IDS should be able to live with it and continue to work as usually and give the *”sick”* computer enough resources to recover from the intrusion, meaning that all the other processes work much slower preventing also a further

outbreak.
Defense in depth can be realized by using different ”cells” or agents monitoring different parts of the system like the network, the filesystem or the resources.
ADENOIDS [8] is an interesting framework that implements a few of the above enumerated points.

13.3 Conclusion

We discussed in this article an immune-system-inspired approach to intrusion detection. The similarities between the tasks of the human immune system and intrusion detection systems suggest that IDS can be improved by converting concepts from the biological to the digital world. Clearly, we must abstract from the concrete biological principals to benefit from them in intrusion detection. Interaction between misuse and anomaly detection, distributivity, avoiding single points of failure, and locality, possibly affecting only single processes, are what we have extracted as main features of immune-system-inspired IDS.

References

[1] C. Ehret and U. Ultes-Nitsche, “Immune system based intrusion detection system,” in *Proceedings of the ISSA 2008 Innovative Minds Conference*, July 2008.

[2] Rebecca Bace and Peter Mell, “Intrusion detection systems,” *NIST Special Publication 800-31*, November 2001.

[3] C. Ehret, “Immune based intrusion detection system,” tech. rep., RVS Retreat 2007 at Quarten, 2007.

[4] D. M. Ivan Roitt, Jonathan Brostoff, *Immunology*. Mosby, sixth edition ed., 2001.

[5] Polly Matzinger, “The danger model: a renewed sense of self,” *Science 296, pp. 301-305*, 12Apr. 2002.

[6] Leandro N. de Castro and Jonathan Timmis, *Artificial Immune Systems: A New Computational Intelligence Approach*. Springer, 4July 2002.

[7] Stephanie Forrest, Alan Perelson, Lawrence Allen, and Rajesh Cherukuri, "Self-Nonself Discrimination in a Computer," *Proceedings of the 1994 IEEE Symposium on Research in Security and Privacy*, 1994.

[8] L. d. G. P. de Paula, F.S.; de Castro, "An intrusion detection system using ideas from the immune system," *Evolutionary Computation, 2004. CEC2004. Congress on*, vol. 1, pp. 1059–1066 Vol.1, June 2004.

14 EAP-TPM - Secure User Authentication in 802.11 Based Networks

Carolin Latze, University of Fribourg
carolin.latze@unifr.ch

14.1 Introduction

With the emergence of 802.11 enabled mobile phones, users wanted to switch from GSM calls to Voice over IP (VOIP) calls, but everybody who ever tried to connect his 802.11 enabled mobile phone to a wireless access point - no matter whether this access point was public or private - knows that 802.11 is far away from being as comfortable as GSM. From an user's point of view, GSM works very easy: Just buy a GSM phone and a SIM (Subscriber Identity Module) card, switch it on and use it. Things are different for wireless networks. Contrary to GSM, 802.11 did not include an authentication infrastructure right from the beginning, which was in 1997 [1]. The Wired Equivalent Privacy (WEP) standard only came in 1999 [1]. But WEP as well as WPA (WiFi Protected Access), which came afterwards are not useful for user authentication, they only provide a means for filtering devices that are allowed to connect to the network. User authentication came with the 802.11i standard in 2004 [2]. This standard proposed to use the Extensible Authentication Protocol (EAP) [3] for user authentication. There are several possibilities inside EAP and one of them is Transport Layer Security (TLS) with certificate based mutual authentication [4]. EAP-TLS with mutual authentication relies on the fact that both sides - client and authentication server - have valid X.509 certificates, which makes it a very secure authentication protocol. But obtaining a X.509 certificate is a hard task for a naïve user. Therefore, in 2007, the authors of [5] proposed a new version of EAP-TLS called EAP-TPM. This protocol does not use X.509 certificates for authentication but certificates that come with the Trusted Platform Modules (TPMs) build into many new PCs. In the first version, the user has to trigger the certificate retrieval before connecting to an EAP-TPM secured access point [5], whereas in the second version, published in 2008 [6], the authors extended that protocol with a zero configuration scheme.

14.2 Discussion

EAP-TPM as proposed in [5] and refined in [7] relies on a so called Trusted Platform Module (TPM) on the client side.

14.2.1 The TPM

The TPM as specified in [8] is a hardware module build in into most of the new desktop PCs and notebooks. It provides several cryptographic functions like asymmetric decryption and key and platform binding. The latter allows to bind data to certain platform states, which are stored as hashes in the TPM. Furthermore, the TPM may store asymmetric keys - at least their private parts and it comes with a so called identity certificate infrastructure. Identity certificates are certificates that attest a TPM that it is a real, well working TPM. In order to request an identity certificate, a request has to be sent to a so called Privacy Certificate Authority (PCA) including some certificates issued by the TPM manufacturer, which can be done automatically without user interaction. This identity certificate is a valid X.509 certificate with a RSA key that is restricted to SHA-1 hashing.

14.2.2 EAP-TPM - First Version

The first version of EAP-TPM as published in 2007 [5] relies on the fact, that an user requests an identity certificate before connecting to an EAP-TPM secured network. In order to do so, he triggers the request in another (maybe wired) network and stores the identity certificate locally. As this identity key is restricted to SHA-1 signing, it is not possible to use this certificate directly in a TLS based authentication. Therefore, a new key will be generated locally, that will never leave the TPM. This key will be certified as non-migratable, TPM generated key by the identity key. The TPM provides a function call Tspi_Key_CertifyKey in order to do that. This function returns some kind of certificate which is not compatible to the X.509 standard as shown in [7], which means although this new key provides the signing features needed in TLS, the lack of a valid X.509 certificates forced the authors to think about another solution. There are two solutions for this problem:

SKAE In 2005, the Trusted Computing Group (TCG) developed a new X.509 extension called Subject Key Attestation Evidence (SKAE) [9]. As

explained above, the user has to request an identity certificate and generate and certify a new key using Tspi_Key_CertifyKey. Afterwards, a new X.509 certificate will be generated including the new SKAE extension that carries the structure returned by Tspi_Key_CertifyKey. This request will be sent to a SKAE aware Certificate Authority (CA) that issues a new valid X.509 certificate with that SKAE extension. This certificate may now be used as every X.509 certificates. However as mentioned in [6], this procedure requires modifications on the server side, that has to accept SKAE extended certificates as well as at the CA (which is usually different from the Privacy CA mentioned above). The authors of [6] think that relying on a modified CA is a hard constraint for applications, since it is unlikely that there will be many such CAs in the near future.

A new Certificate Called TPM Certificate In 2008, the authors of [7] proposed to implement a new type of certificates called TPM certificates. Such a certificate may carry the structure returned by Tspi_Key_CertifyKey and a reference to its X.509 parent, which is the identity certificate. As the TPM is a trusted device and the identity key is already certified by a third party - the Privacy CA - it is sufficient to generate and certify that certificate locally without a third party. Such a certificate only needs modifications on the server side that has to accept it.
The authors of EAP-TPM decided to implement the new certificate type since this requires less modification in the whole certification and authentication process than the SKAE approach. This decision is also the reason, why this new authentication protocol is not called EAP-TLS anymore: TLS requires X.509 certificates [4]! That means, if other certificates are used, this is not TLS anymore.

14.2.3 EAP-TPM - Second Version

In 2008, the authors of [5] extended their protocol with zero configuration scheme that allows an user to connect to an EAP-TPM secured network without any prerequisites (as long as he has an activated and owned TPM in his device) [6]. This scheme is realized by modifying the SSL handshake: If the user tries to connect to an EAP-TPM enabled access point, the access points sends him a list of acceptable Privacy CAs. The client application checks whether there is an identity certificate signed by one of those Privacy CAs. If there is one, it is very likely that there is already a valid child certificate, that may be used directly for authentication. If there is only the identity certificate and no child, the child certificate may be gen-

erated on the fly and used for authentication. If there is no such identity certificate, the application sends an alert to the access point indicating that it has no such certificate and wants to request one from a Privacy CA, it specifies in this alert. The access point will then allow the client to connect to that Privacy CA without authentication. After having received the identity certificate, the client generates the child certificate and goes on with the authentication process. If there was a timeout on the authentication server during this (maybe long) certificate generation, the client has to start a new authentication process after having created the certificate. This second process will succeed since he now already owns a certificate accepted by the access point.
Such a zero configuration scheme allows even naïve users to connect to an EAP-TPM secured 802.11 network as comfortable as to GSM networks.

14.3 Conclusion and Outlook

According to the authors, the extension of the work proposed in [5] with the zero configuration scheme proposed in [6] has really the potential of making 802.11 authentication as comfortable as GSM authentication. All the other features of GSM like QoS, roaming, accounting and so on already exist in 802.11. The only thing that was really missing was a comfortable authentication scheme. Captive portals as mostly used today are neither comfortable nor automatable. EAP-TPM provides both properties.
The implementation of EAP-TPM is still ongoing, but a very first prototype has already shown very promising results [7].

References

[1] “IEEE 802.11 - The Working Group Setting the Standards for Wireless LANs.” Last Accessed: May 2008 [Online] http://grouper.ieee.org/groups/802/11.

[2] “IEEE 802.11i.” Last Accessed: May 2008 [Online] http://grouper.ieee.org/groups/802/11/Report/tgi_update.htm.

[3] L. Blunk and J. Vollbrecht, “PPP Extensible Authentication Protocol (EAP).” RFC 2284, 1998.

[4] D. Simon and B. Aboba, “PPP EAP TLS Authentication Protocol.” RFC 2716, 1999.

[5] C. Latze, U. Ultes-Nitsche, and F. Baumgartner, “Strong Mutual Authentication in a User-Friendly Way in EAP-TLS,” in *Proceedings of the 15th International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2007)*, (Split - Dubrovnik, Croatia), 2007.

[6] C. Latze, U. Ultes-Nitsche, F. Baumgartner, “Towards a Zero Configuration Authentication Scheme for 802.11 Based Networks,” in *Proceedings of the IEEE Conference on Local Computer Networks (LCN 2008)*, (Montreal, Canada), 2008.

[7] C. Latze and U. Ultes-Nitsche, “A Proof-of-Concept Implementation of EAP-TLS with TPM support,” in *Proceedings of the Innovative Minds conference (ISSA 2008)*, (Johannesburg, South Africa), 2008.

[8] The Trusted Computing Group, 2002.

[9] The Trusted Computing Group, “Subject Key Attestation Evidence Extension - Specification Version 1.0,” 2005. https://www.trustedcomputinggroup.com/specs/IWG.

15 Traffic Adaptivity in Energy-Efficient MAC Protocols

Philipp Hurni, University of Bern
hurni@iam.unibe.ch

15.1 Introduction

The WiseMAC [1] protocol a very energy-efficient medium access control protocol for wireless sensor networks (WSNs). WiseMAC is based on low duty cycles, periodic wake-ups and preamble sampling. In many typical WSN scenarios however, the maximum achievable throughput of WiseMAC is very limited when high traffic occurs and has to pass certain bottleneck nodes. The more bit mechanism of WiseMAC to date only allows additional traffic along one link, e.g. for large messages or frame bursts in point-to-point scenarios. This section portrays and extension of the more bit mechanism to improve reception of increased traffic from multiple senders.

15.2 Discussion

15.2.1 WiseMAC

WiseMAC is based on short, unsynchronized duty cycles and preamble sampling. A preamble is used to alert the receiving node to stay awake for any upcoming transmission. When the receivers wake-up pattern is unknown, the duration of the preamble equals the full basic wake interval T (cf. Figure 15.1). Schedule offsets are piggybacked to frames and acknowledgements and stored in a table. Based on this table, nodes can

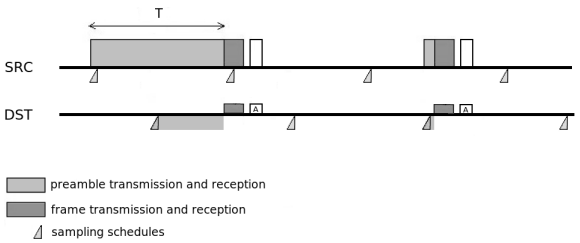


Figure 15.1: WiseMAC

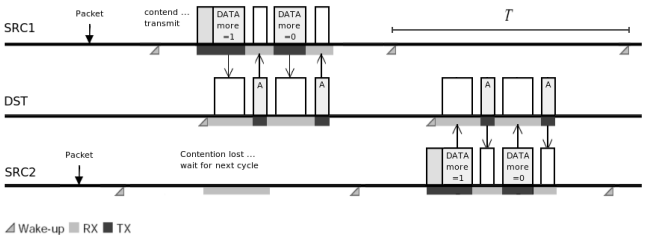


Figure 15.2: More Bit

determine their neighbors' wake-ups and minimize the preamble length for upcoming transmissions (cf. Figure 15.1).
To increase the maximum achievable throughput in case of packet bursts and higher traffic load, WiseMAC suggests an optional fragmentation scheme called more bit mode. WiseMAC sets a flag (more bit) in a unicast MAC frame whenever a node has more packets to send. The bit in the frame header signals to the receiving node that it shall not turn off the transceiver after receiving the frame, but switch to the receive mode again after frame acknowledgement in order to receive the next packet, cf. Figure 15.2. The scheme proved to be very effective in scenarios with varying traffic, especially with packet bursts generated by single nodes. The more bit scheme only serves to improve traffic adaptivity between one sender and one destination. In a wireless sensor network scenario, there are often nodes that have to forward data from large sub-trees. Such bottleneck nodes will have to forward messages generated by many other nodes. The scheme does not help at all if several nodes aim to simultaneously transmit a packet to the same bottleneck node. One node after the other will have to wait for a wake-up of the bottleneck node in order to forward a frame.

15.2.2 Improving the WiseMAC More Bit

We worked on an improvement of WiseMAC that would yield a better traffic adaptivity in cases with multiple senders concurrently sending packets to one destination node. We extended the semantics of the more bit to a so-called stay awake promise bit. This is also called *extended more bit* hereafter. Figures 15.2 and 15.3 illustrate the basic idea and the difference to the existing More Bit scheme. Two sources SRC1 and SRC2 simultaneously aim to transmit some packets to the same node DST. If both senders try to reach DST in the same wake-up, the medium reservation preamble will decide who is first. SRC1 wins the contention and sends its first two frames with the (extended) more bit set. In Figure 15.2, the existing

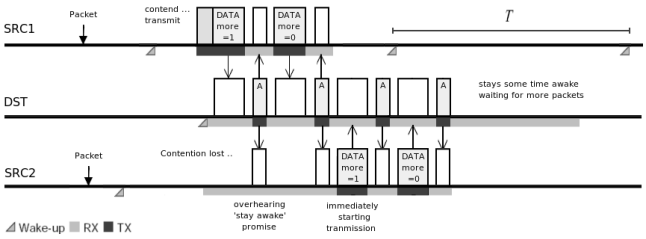


Figure 15.3: Extended More Bit

WiseMAC More Bit mechanism is applied. The destination node acknowledges the reception of the frame and SRC1 empties its buffer until it has nothing more to send. Then, SRC1 and SRC2 go to sleep until the next wake-up of DST, where SRC2 finally starts transmitting its packets.
In Figure 15.3, the *extended more bit* based on the *stay-awake promise* is illustrated. The destination node acknowledges the more bit in the ACK packet and stays awake for some predefined time interval (e.g. one basic wake interval T). SRC2 having lost the contention will wait and overhear the transmission to DST. Hearing the stay awake promise in the ACK, SRC2 knows that it can start sending its frames right after SRC1 has finished. The advantage of this scheme is that no time is wasted for waiting, as the transmission of SRC1 can start immediately after the transmission of node SRC1. The mechanism is only activated when there is a node buffering more than one frame that requests its destination to stay awake for one next packet, which is a signal of increased load.

15.2.3 Simulation Evaluation

We chose a scenario with 90 nodes uniformly distributed in an area of 300 m x 300 m in the OMNeT++ network simulator [2] and the mobility framework [3]. Traffic using a Poisson model is generated for 1 hour at each node and sent towards a single sink. We use static shortest path routing. Each node uses a basic interval T = 250 between two wake-ups and a duty cycle of 5 %. The energy consumption model is based on the amount of energy that is used by the transceiver unit. We used an energy consumption and state transition model with three operation modes sleep, receive and transmit, and applied the respective energy consumption values and state transition delays of the transceiver manufacturer [4] of our sensor hardware testbed, the ESB nodes [5].
Figures 6 illustrates the increase of maximum throughput with both the more bit and the stay awake promise bit. The stay awake promise ap-

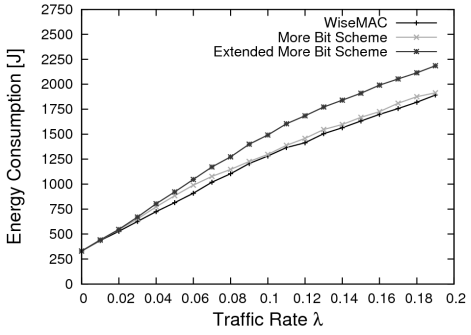


Figure 15.4: Energy consumption

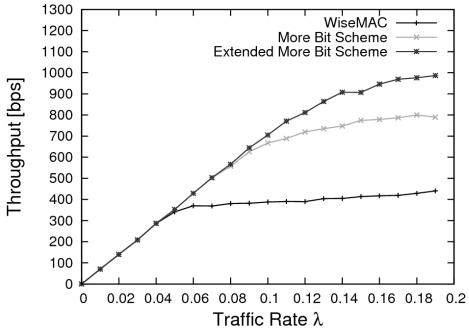


Figure 15.5: Throughput

proach is superior in terms of all considered performance metrics such as throughput, packet loss, and one-way delay. However, the improved performance comes with higher energy costs. But when we consider the ratio of throughput and energy, the stay awake promise scheme is even better than the more bit scheme for high traffic.

15.2.4 Evaluation on Embedded Sensor Boards

We implemented the original WiseMAC mechanism and the (extended) more bit on Embedded Sensor Boards [5] to examine the real-world behavior of the MAC mechanisms. ESBs run the sensor node operating system ScatterWeb and are equipped with a micro-controller MSP430, various sensors and communication interfaces such as an 868.35 MHz wireless transceiver. We evaluated the throughput of the two schemes more bit and extended more bit when generating traffic of equal rate from two senders to one receiver. When both senders aim to concurrently forward packets to the receiver, the receiver becomes a bottleneck. With the extended more bit scheme, the receiver node promises to stay awake for at least T = 500 ms by a single bit in the acknowledgement frame. Figure 15.6 shows the measured throughput at the receiver in the given scenarios. The WiseMAC protocol without the more bit scheme can only deliver one packet per wake-up, and therefore, throughput is limited to two packets per second (T = 500 ms). When increasing the rate, packets are subsequently queued in the buffer and dropped when the buffer is full. When two stations apply the (extended) more bit scheme, they can alternately empty their transmit buffers by packet bursts for increasing traffic. The throughput reaches nearly 8 packets per second for the more bit scheme

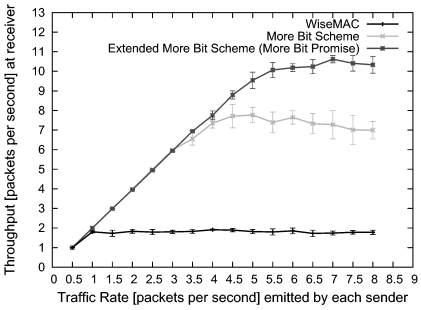


Figure 15.6: Throughput Evaluation

and exceeds 10 packets for the extended more bit scheme using the stay-awake promise bit. The throughput increase for the extended more bit scheme compared to the original more bit scheme exceeds 20%.

15.3 Conclusions

The measurement results confirm that the extended more bit basing on the so-called stay-awake-promise performs better than the original WiseMAC more bit scheme. The superior performance of roughly 20% has been found similar in both simulation and real-world experiments.

References

[1] El-Hoiydi, A., Decotignie, JD., “Wisemac: An ultra low power mac protocol for multihop wireless sensor networks,” ALGOSENSORS 2004.

[2] Varga, A., “The omnet++ discrete event simulation system,” European Simulation Multiconference 2001. http://www.omnetpp.org.

[3] W. Drytkiewicz, S. Sroka, V. Handziski, A. Köpke, H. Karl, “A mobility framework for omnet++,” 3rd Intl. OMNeT++ Workshop, 2003. http://mobility-fw.sourceforge.net.

[4] RF Monolithics, “Datasheet for tr1001 868.35 mhz hybrid transceiver,” http://www.rfm.com/products/data/TR1001.pdf.

[5] Schiller, J. Liers, A. Ritter, H. Winter, R. Voigt, T., “Scatterweb - low power sensor nodes and energy aware routing,” 38th Annual Hawaii International Conference on System Sciences, 2005.

16 Real-time Communication in Wireless Mesh Networks

Thomas Staub, University of Bern
staub@iam.unibe.ch

16.1 Introduction

Wireless Mesh Networks (WMNs) have caught on as a technology for increased network coverage [1]. There exist serveral research networks (e.g. MIT Roofnet [2], Berlin Roofnet [3]) as well as deployments in cities as a public service (e.g. [4], [5]). Connections inside a WMN are automatically established over wireless links, which eases the deployment. The mesh nodes build up an ad-hoc network over wireless links, which is adapted by the network's self-organisation capabilities. As all communication runs over wireless links, no wired backbone is required as in conventional access networks. These facts make a WMN easy to deploy and maintain.

Although WMNs provide increased network resilience due to self-organisation and redundancy in the network, the performance of real-time application suffers from quality variations of the wireless links. The noisy and erroneous wireless communication channel degenerates the quality of the real-time communication by bit errors or link failures. The bit errors in the data transmission require either additional redundancy or retransmissions and therefore reduce available bandwidth. Interference or other environmental changes may further cause link failures. High delays or packet loss are the consequence. The user received transmission quality becomes unacceptable due to high rate of artefacts, stumbles, or even interruptions. Hence the deployment of real-time applications in WMNs is challenging.

We propose to use path diversity and multi-stream coding to cope with the link quality variations in the network. The combination of both techniques may drastically reduce the effects of the wireless medium to the transmission. Multiple paths are usually not affected by the same errors, delays, jitter and loss rates at the same time (see Fig. 16.1 for an illustration). Their error characteristics are mostly uncorrelated. Therefore, the transmission over multiple paths (path diversity) adds redundancy and therewith lowers the influence of the unreliable medium. The degree of redundancy can be optimised by using advanced multi-stream coding schemes such as lay-

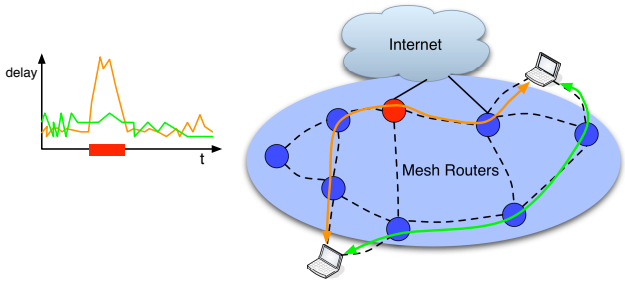


Figure 16.1: Delay Characteristics of Multiple Paths in the Network

ered (LC) or multi-description coding (MDC) [6, 7, 8] instead of sending the same stream multiple times. LC requires the correct reception of the base layer to reconstruct the input stream, whereas MDC can use any combination of received streams for reconstruction. Depending on the network conditions, the most appropriate parameter set of encoding and number of paths varies [9]. Therefore, the system has to dynamically adapt the parameter set, that consists of the encoding selection, the number of paths used, and the mapping of the individual streams to the paths.

16.2 Adaptive Transport Over Multipaths (ATOM)

Adaptive transport over multipaths (ATOM) [10] provides support for real-time communications in WMNs. Path diversity and multi-stream coding are combined with dynamic adaptation to the current network conditions. ATOM uses the most appropriate selection of the paths provided by the multi-path routing and the multi-stream encoding options. It further maps the streams to the paths. The decision-making process is supported by network measurements, monitoring, and statistical evaluation. It is further continuously repeated and therefore adapts to changes in the network conditions.

16.2.1 ATOM Architecture

The general architecture of ATOM is shown in Fig. 16.2. It consists of an ATOM aware application and the ATOM core components. The ATOM aware application implements the ATOM application-programming interface (API) in order to communicate with the ATOM core. It exchanges the available codecs and coding options with the ATOM core and receives the

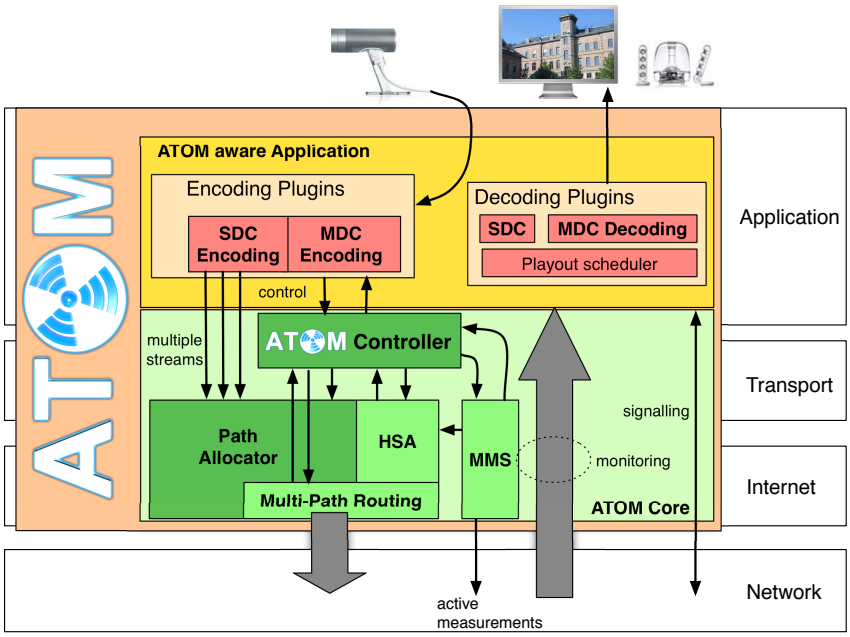


Figure 16.2: ATOM Architecture

configuration set (encoding, number of streams). In addition, it may provide feedback to the ATOM core during a communication session. The core then revises the decision and releases a new parameter set. The main functionality of ATOM is implemented in the ATOM core, which is split into the following components: ATOM controller, history and statistical analyser (HSA), monitoring and measurement system (MMS), multi-path routing, and path allocator. As the central component, the ATOM controller gathers control data and decides about the communication configuration used. It receives data about available encodings from the application(s), available paths from the multi-path routing, the robustness of the path or individual links from HSA, and the current network conditions from MMS. Based on this data, it selects an optimised set of configuration parameters. The application is then requested to provide n streams with the selected encoding. The path allocator receives a map describing which stream has to be transmitted on which path. If necessary, the ATOM controller triggers a route discovery of new paths by the multi-path routing. According to the feedback of the MMS and the application, it further revises its decision and adapts the transmission parameters.

Besides the multi-path routing, which has to be installed in all mesh nodes, the ATOM core components can be placed at different locations inside

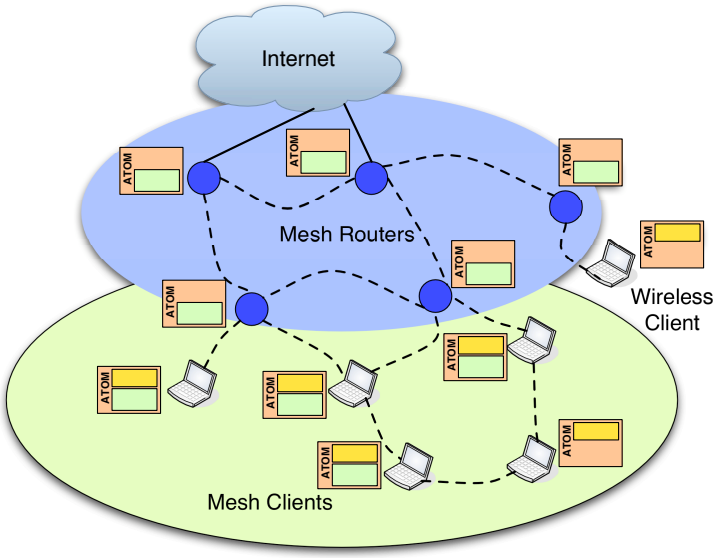
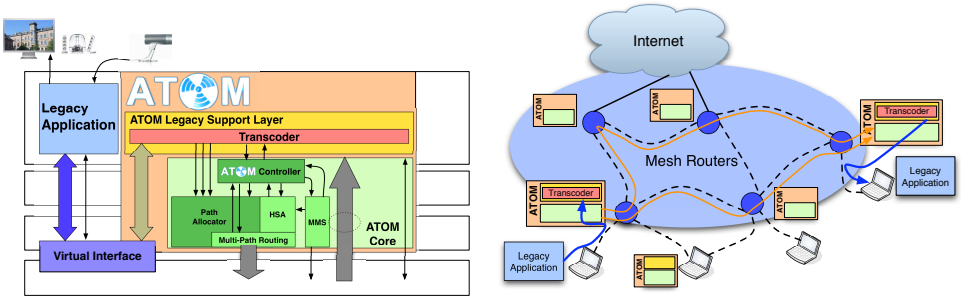


Figure 16.3: Placement of ATOM Components in the Mesh Network

WMNs (see Fig. 16.3). Either all components are included in the end system or the core components are only situated in the mesh routers. The placement in the end system requires modifications in each client. ATOM core components have to be implemented for the different operating systems of the different client devices. As second possibility, the ATOM core components can only be installed in the mesh nodes. This requires no further software installation at the client besides the ATOM aware application. Therefore, there is no need for porting the ATOM core to different operating systems. Only the mesh nodes have to be supported. The control traffic between an ATOM-aware application and the ATOM controller then runs over TCP/IP. One drawback of the mesh placement is that the quality of the transmission strongly depends on the link quality between client and first mesh node. Furthermore, a combination of both approaches is possible.

Moreover, the benefits of ATOM are offered to legacy applications. The two methods of ATOM legacy support are illustrated in Fig. 16.4(a) and 16.4(b). The first method provides the legacy support at the end system. The traffic of a legacy application is sent to a virtual interface, which forwards it to the ATOM legacy support layer. This layer establishes the ATOM session and transcodes the input stream to multiple streams. At the destination, the ATOM legacy support layer then merges the streams and forwards the output stream to the destination legacy layer. The second

method puts the complete ATOM and transcoding functionality to the in- and outbound mesh routers.



(a) at the End Systems (b) at In-/Outbound Mesh Routers

Figure 16.4: ATOM Support for Legacy Applications by Traffic Interception

16.2.2 Session Establishment and Release

ATOM has to establish a session before the data communication can start and after communication has terminated it releases the session (see Fig. 16.5). The lifetime of an ATOM session is divided into three phases:

- Session establishment: The ATOM-aware application at the source invites its communication peer and receives the available coding options. Then it requests an ATOM session at the ATOM controller. The request includes the mutually available coding options of both communication peers. The ATOM controller makes a decision based on the provided data including network state. It either accepts or denies the request. If the session is accepted, monitoring at the remote ATOM controller is started. The ACCEPT message includes the necessary parameters for the ATOM-aware application. The application then informs its remote peer about the acceptance of the requested session.
- Data communication: The ATOM-aware application transmits the streams to the destination.
- Session release: The ATOM-aware application closes the session by transmitting a CLOSE message. This message is acknowledged by a CLOSED message. After that, the application triggers the release of the session at the ATOM controller, which then stops the monitoring at the remote ATOM controller.

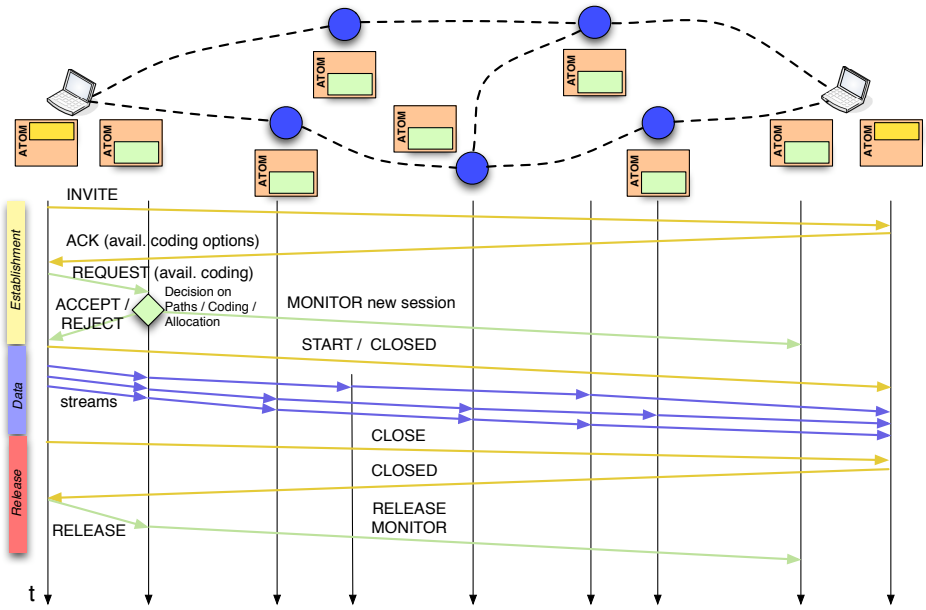


Figure 16.5: Session Establishment and Release

16.2.3 Session Reconfiguration: ATOM Adaptation Process

Changes in the network conditions provoke an ATOM adaptation process during data communication (see Fig. 16.6). If the remote ATOM controller or the (remote) application recognise a route break or high delays on one link, they inform the source ATOM controller through a NOTIFY message. The ATOM controller then revises its decision based on the new data. The new parameter set is sent to the source application and to the monitoring ATOM controller. Afterwards, the application informs its communication peer by a MODIFY message and then starts to transmit the streams according to the new parameter set.

16.3 Conclusion and Outlook

WMNs provide a reliable and robust communication infrastructure. But the erroneous nature of the wireless medium limits their usage as a network for real-time communication. ATOM solves this issue by combining multi-path routing, multi-stream coding, and dynamic adaptation. ATOM is work in progress. We are specifying the decision making process and the statistical analysis of the HSA. The ATOM components are currently

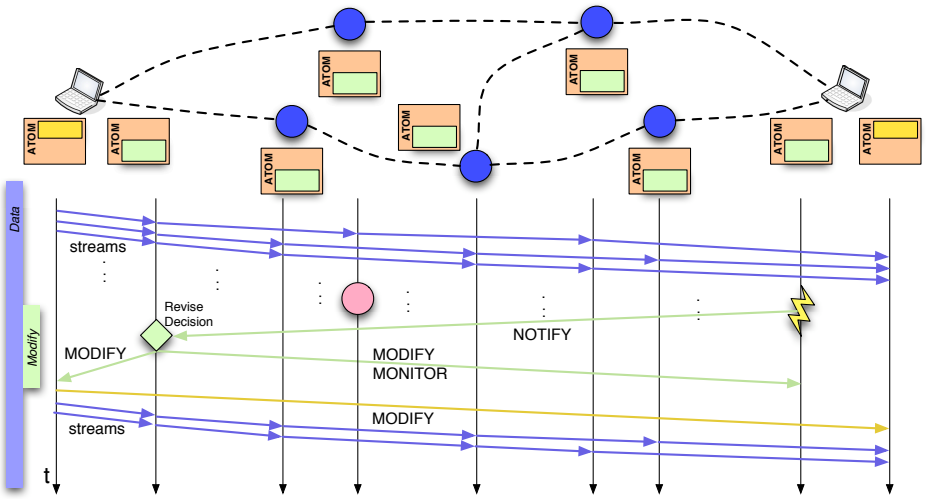


Figure 16.6: Session Reconfiguration

being implemented in the network simulator Omnet++. Moreover, we are preparing the implementation of ATOM on our Linux based wireless mesh network testbed.

The work on ATOM was supported by the Swiss National Science Foundation under grant number 200020-1136777/1 and parts are published in [10]. This report is submitted for publication in WIRELESS4D'08.

References

[1] I. F. Akyildiz and X. Wang, "A survey on wireless mesh networks," *Communications Magazine, IEEE*, vol. 43, no. 9, pp. S23–S30, 2005.

[2] J. C. Bicket, D. Aguayo, S. Biswas, and R. Morris, "Architecture and evaluation of an unplanned 802.11b mesh network.," in *11th Annual International Conference on Mobile Computing and Networking (MOBICOM 2005)*, (Cologne, Germany), pp. 31–42, August 28 - September 2 2005.

[3] R. Sombrutzki, A. Zubow, M. Kurth, and J.-P. Redlich, "Self-organization in community mesh networks the berlin roofnet," *Operator-Assisted (Wireless Mesh) Community Networks, 2006 1st Workshop on*, pp. 1–11, Sept. 2006.

[4] R. Karrer, A. Sabharwal, and E. Knightly, "Enabling large-scale wire-

less broadband: The case for taps.," in *2nd Workshop on Hot Topics in Networks (Hot-Nets II)*, (Cambridge, MA, USA), November 2003.

[5] "Meraki 'free the net' project in San Francisco." available online: `http://meraki.com/about/freethenet/`, 2007.

[6] J. G. Apostolopoulos and M. D. Trott, "Path diversity for enhanced media streaming," *IEEE Communications Magazine, special issue on "Proxy Support for Streaming on the Internet"*, vol. 42, pp. 80–87, August 2004.

[7] Y. J. Liang, E. G. Steinbach, and B. Girod, "Real-time voice communication over the internet using packet path diversity," in *MULTIMEDIA '01: Proceedings of the ninth ACM international conference on Multimedia*, (Ottawa, Canada), pp. 431–440, ACM Press, 2001.

[8] N. Gogate, D.-M. Chung, S. Panwar, and Y. Wang, "Supporting image and video applications in a multi-hop radio environment using path diversity and multiple description coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, pp. 777–792, September 2002.

[9] J. Chakareski, S. Han, and B. Girod, "Layered coding vs. multiple descriptions for video streaming over multiple paths," *Multimedia Systems*, vol. 10, pp. 275–285, April 2005.

[10] T. Staub and T. Braun, "ATOM: Adaptive Transport over Multipaths in Wireless Mesh Networks," in *2nd ERCIM Workshop on eMobility*, (Tampere, Finland), May 30 2008.