Chamaeleon – Exploiting Multiple Channels to Mitigate Interference

Venkatraman Iyer*, Matthias Woehrle†, Koen Langendoen*
*Embedded Software group, Delft University of Technology, The Netherlands
†Computer Engineering and Networks Lab, ETH Zurich, Switzerland

***Abstract*—Due to their embedding into an unknown environment, wireless sensor networks are susceptible to adverse environmental conditions. A prime example is external interference, typically caused by 802.11, electric devices or co-located sensor networks, which can drastically influence communication. However, radios on sensor nodes provide multiple channels, which can help to alleviate interference problems on a particular channel. In this work, we introduce Chamaeleon, a novel lean, cross-layer protocol extension that helps to overcome external interference problems. Chamaeleon switches an affected set of nodes to a new channel based on key network parameters e.g., packet loss and channel backoffs. Initial simulation-based experiments show that Chamaeleon effectively handles the jamming of isolated channels.**

I. Introduction

Wireless sensor nodes are networked embedded systems integrated into an environment. Using sensors and actuators, nodes collectively stream information from and to a monitored phenomenon. These Wireless Sensor Networks (WSNs) are deployed for various application scenarios and, hence, operate in different locations. The integration into an unknown and ever-changing environment implies that a sensor network must adapt to the particular setting of a deployment site. In general, this may include protection from adverse environmental effects such as highly varying temperature, humidity, and even vandalism. This work focuses on handling the adverse effects on the wireless communication channel in the particular context of low-power data-gathering applications where sensed data needs to be collected from nodes and sent to a sink (gateway node) over multiple hops.

Sensor nodes communicate using low-power radios operating in the unlicensed ISM Bands. Most sensor nodes use standardized 801.15.4 radios such as the TI CC2420, used on the Crossbow Iris, TelosB and the MicaZ. 802.15.4 radios operate in the 2.4 GHz range. Other prominent communication technologies in this band are Bluetooth and especially WLAN (802.11). Hence, a sensor network deployed in an unknown environment must accommodate for *interference* from devices in its neighborhood. Furthermore, given the predicted ubiquitous nature of WSNs, the interference may also be generated by other sensor networks [1]. This *external interference*, i. e., interference created by devices outside the sensor network, is not necessarily a network-wide phenomenon - only a subset of the sensor nodes might be affected. Additionally, depending on the placement of nodes and the resulting network topology, *internal interference* between different sensor nodes in the same network can adversely affect the communication.

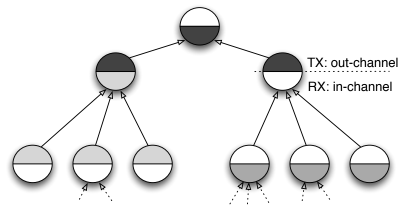

Fig. 1. Parent-children pairs each using a single frequency for communication

Interference effects can be mitigated by using the frequency diversity of 802.15.4 radios, which provide multiple channels. This paper presents Chamaeleon, a communication enhancement that can be integrated with commonly used protocol stacks. Chamaeleon exploits the frequency diversity offered by 802.15.4 radios in order to mitigate interference effects and allows for a stable operation of a WSN deployment. Chamaeleon extracts performance information of the MAC protocol and exploits local topology information to switch a parent and its children to a new frequency when the need arises.

The primary focus of our work is on increasing the robustness of the communication in sensor networks using multiple channels without incurring an increase in complexity. This paper presents the following contributions:

- A novel protocol extension for locally detecting and mitigating interference effects in data collection WSNs.
- An initial evaluation of *Chamaeleon* in simulation (TOSSIM) demonstrating its effectiveness.

In the following section, we describe the background of using multiple channels for WSNs, present the main idea behind our work, and compare it to related work. In Section III we describe the Chamaeleon protocol in general. Section IV provides specific implementation details of our TinyOS prototype and presents initial results from a set of TOSSIM simulations. Section V concludes the paper.

II. Background and General Idea

Before presenting the main idea behind Chamaeleon, we provide some background information on the radios and their typical use in sensor networks deployed for data-gathering applications. We also discuss some related work.

A. Background

Our work is based on the following observations concerning the 2.4 GHz ISM band: 1) External interference is unpredictable, and tends to interfere communication on several 802.15.4 channels with varying effects. Therefore, off-line approaches to channel switching, or use of a dedicated channel (e. g. as used in [2]) are not feasible. 2) Interference from external devices that operate in the free unlicensed band is localized, so only a small part of the network is affected by them. Network-wide switching of the communication channel is not desirable, as finding a single frequency where no node experiences interference entails large overheads and may well be impossible when multiple external "sources" are involved. Thus, we are looking for a distributed, adaptive and localized control algorithm. On the MAC layer, a fundamental issue of operating with multiple channels is the *deafness effect* caused by (long) delays in accessing a particular channel. Classical contention-based MAC protocols back off when finding the channel occupied and keep monitoring the channel for it to become free; while waiting the node becomes "deaf" to inbound traffic on other channels.

Switching channels costs time and energy. While the actual programming of the radio registers and the subsequent PLL synchronization only takes hundreds of microseconds, the setting from the microcontroller over the SPI bus takes over a millisecond on a Tmote Sky using a CC2420 radio. Therefore, the number of switches should be kept low, which is also important for the sake of stability as with every switch the chance of one node "missing the boat" increases.

Specifically, we focus on data collection applications, where long lifetime requirements have to be met. Hence, the focus of this work is on low-power, duty-cycling MAC protocols, in particular on the widely-used family of random-access MAC protocols, on modern packet-based 802.15.4 radios.

B. General Idea

Effectively operating with multiple channels requires a coordinated effort from the MAC *and* the routing layer. The MAC layer could autonomously switch channels, but when interference is localized a reshaping of the routing tree may very well be the most appropriate solution, i. e., affected siblings switch to a new channel/parent while the others continue with the current channel. As we feel an integrated (cross-layer) MAC/routing protocol would considerably compromise complexity and flexibility, we have developed Chamaeleon as a separate MAC-protocol enhancement. Chamaeleon controls the channel selection/switching, but relies on a traditional MAC for the sending and receiving of packets; for now the interaction with the routing layer is minimal, i. e., we expect it to set up a collection tree.

Chamaeleon combines information on the performance of the MAC layer, indicating the need for a channel switch, and the logical topology maintained by the routing layer, i. e., the local parent-child relations. To facilitate local channel selection, Chamaeleon distinguishes parent and child roles for each node. These different roles may operate on different frequency channels. In particular, the parent listens for incoming traffic on the so-called *in-channel*, and the child forwards outbound traffic on the so-called *out-channel*. Fig. 1 shows a sample channel assignment (indicated with different shades of grey) facilitated by the separation of in- and out-channels. When conditions deteriorate due to external interference parent and children coordinate to switch to a new, clean channel. However, as lines of communication may break completely, a channel switch may also be effected autonomously by a node (parent/child) counting on the other end (child/parent) to do the same. Below we outline the different actions for child and parent roles (nodes):

Child role: Nodes periodically report their MAC performance to the parent node by piggy-backing this information on regular data transmissions. If communication with the parent is not possible at all (i. e., no ACKs are received), a child performs an autonomous switch to find its parent on the subsequent (out-)channel. If this does not help, it falls back to a bootstrap phase, provided by the routing layer, to find a new parent.

Parent role: A parent monitors information of all its children. If the MAC performance of its children deteriorates, a parent decides to switch its in-channel *and* informs it children by means of piggy-backing on the ACKs associated with the regular incoming data packets. If communication with children is not possible at all (i. e., packets stop arriving) a parent performs an autonomous switch to find its children on the subsequent (in-)channel.

To facilitate the autonomous switching, Chamaeleon employs a pre-computed hopping sequence. As (external) interference may easily affect neighboring frequencies, this sequence "randomly" exercises the 802.15.4 channels possibly omitting overlapping WLAN frequencies. The length of the hopping sequence can be kept small as the local channel selection mechanism facilitates frequency reuse.

C. Related Work

There has been a substantial amount of work on using multiple channels in Wireless Sensor Networks, e. g. [3], [4], [5], [6]. We only discuss those that directly match our focus on robustness when faced with external interference.

Voigt et al. [6] target increasing robustness by using multiple channels. Like Chamaeleon, software acknowledgments are used to coordinate between parents and their children. However, different from Chamaeleon it is specifically targeted towards DMAC [7]. Chamaeleon works on top of any MAC protocol using acknowledgments and only uses additional channels when needed. Chamaeleon includes distinct measures to trigger channel updates (see Section III-B), while in [6] there is no clear strategy on how a parent decides which channel to use given observations about packet loss.

Closest in spirit is the work by Le et al. [5], which present a control-theoretic approach to channel allocation. Their multichannel MAC protocol (MC-MAC) is targeted towards high load and higher density rather than increasing robustness.

While MC-MAC's scope is data gathering, the actual design is based on data aggregation traffic, i. e., packets from children are joined with a parent's message. Hence, most of the metrics fail when considering data collection. Nevertheless, Chamaeleon and MC-MAC share common foundations and design strategies:

- Problems concerning interference (or load) are local effects, hence need to be detected and handled locally.
- A (logical) ladder of channel frequencies is beneficial for (autonomous) switching.
- In order to support local decisions about channel switching, neighborhood information needs to be distributed. MC-MAC uses energy-expensive and bandwidth-intensive periodic broadcasts. Chamaeleon, on the other hand, uses topology information available on each parent and piggy-backs decisions on acknowledgments for efficiency.

III. Protocol

The Chamaeleon protocol is divided into two distinct parts described in the following: A *switch mechanism* that devises the control logic of changing channels when it is necessary, and a *switch policy* that specifies the conditions on which to perform a switch.

A. Switch Mechanism

Chamaeleon assigns channels on a parent-children basis. Each node has two associated communication channels, an in- and an out-channel. The in- and out-channel assignments to a node are carried out independently.

Hence, Chamaeleon allows for switching to a new frequency in an affected region without the need of propagating the frequency change upstream towards the leaf nodes, nor downstream towards the sink. This allows for fast, localized channel changes. Additionally, in large deployments finding a single channel where all nodes may operate without any interference may either be impossible or take many iterations of channel switches.

Allocating a common out-channel to all children helps to minimize the deafness effect (cf. Sec. II-A). Additionally it significantly decreases protocol complexity: When using multiple frequencies for receiving, a parent would need to allocate bandwidth to each individual channel based on (i) the set of children on the channel and (ii) the forwarding load of each set of children.

B. Switch Policy

The switch mechanism controls the switching of a parent-children group given a certain policy. There are two distinct reasons for a channel switch: (i) A co-ordinated switch based on the parent continuously monitoring the MAC performance of its children. (ii) An autonomous switch by a node if conditions degrade such that normal communication is no longer possible.

The policies are implemented in two control loops: a fast *inner loop*, for coordinated switch and a lazy *outer loop*, for autonomous switching, which acts as a watchdog.

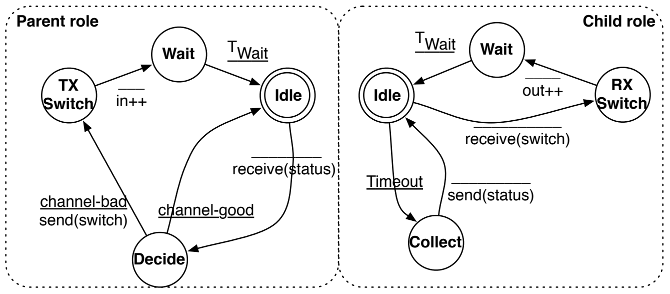

Fig. 2. State machine for parent and child role of the **inner loop**. Transitions are labeled with conditions including timers (on top) and action labels (on the bottom)

1) Inner loop: Fig. 2 presents an overview of the inner loop for a single node given our switch mechanism. It shows dedicated state machines for a node's role as a parent and as a child.

Child nodes monitor the *effort* required for transmitting data. This effort is computed as the number of MAC congestion backoffs $n_{backoff}$ normalized to the number of transmitted packets n_{TX}. Periodically, the MAC status is queried and the computed effort is piggy-backed on the next data packet to the parent. A child node reacts to a received switch-message from its parent (`receive(switch)`) by changing its out-channel (`out++`). After switching the channel, a node waits for a given time T_{wait} in order for the channel switch to take effect.

A parent collects and monitors the MAC information from its children. On each reception (`receive(status)`), it decides whether the currently used channel should be abandoned. If it finds the current channel to be bad, the parent initiates a co-ordinated switch by flagging a channel switch bit on every ACK messages, and finally switches its in-channel after notifying all of its children.

2) Outer loop: For the outer loop, a child node monitors the number of successful and failed transmissions over a much larger window. We consider a transmission successful if the child node receives an ACK message from its parent, and failed otherwise. If the number of successful transmissions is too low or if the number of failed transmissions is too high, the child decides to autonomously switch its out-channel. Likewise, if a parent receives no or only a minimal number of packets, it changes its in-channel. Note that the number of messages that a parent expects to receive depends on the application data rate T_{data} on every node. Since channel switch decisions based on the functioning of the outer loop do not depend on a co-ordination between parent and children, autonomous channel switches are instantaneous.

C. Wait After Switching

The wait state accounts for instabilities after switching: Since transmissions have previously failed, queues will be filled and contention increases immediately after the switch when flushing these. Hence, the wait state allows for ignoring transient effects after switching. This is implemented by a *windowed watchdog*. Over the longer wait period T_{wait}, a

Metric	No switch	Chamaeleon
Data yield (%)	61.65	98.11
Retransmissions (%)	90.36	42.99

TABLE I
PERFORMANCE ON A 70-NODE NETWORK

watchdog constantly monitors that at least a single packet is received (for the parent) or sent (for the child) over a window of a multiple of T_{data}. The watchdog allows for immediately triggering a successive channel switch if the new channel is completely jammed. While basic communication is possible, an evaluation of the channel is only performed after exiting the wait state after T_{wait}.

IV. FEASIBILITY STUDY

We implemented Chamaeleon and integrated it with an application that generates 20-byte data packets every T_{data}. The application runs on top of a routing layer that implements a static topology. We chose the Collection Tree Protocol (CTP) to build a fixed tree, as its implementation is available on TinyOS-2.x. X-MAC was chosen as the MAC protocol, as it features a strobed packet preamble that minimizes idle listening for radios. The specific timing details for the protocol stack can be found in [8]. Chamaeleon resides as a separate software component in-between the routing and MAC layers, and comprises a set of parameters that relate to control loop time settings, as well as threshold values that indicate a channel usage problem. A series of benchmark tests was performed as a means of parameter exploration. Parameters are determined so that nodes only switch channels when needed, and at minimal latency. Fixing the threshold values needs consideration, in order to strike a balance between switching latency and the occurrences of false positives that may lead to oscillations [8].

In order to determine the benefit of using Chameleon over a single channel solution in terms of packet yield and retransmissions, a set of testcases are simulated on TOSSIM. A testcase is simulated for 15 minutes and results are averaged over 10 runs. The jamming functionality is emulated by a node that continuously transmits packets with minimal pause between successive transmissions. The so-called jammer node bootstraps after one minute of simulation time, and continues to spam the channel until the end of the simulation.

We focus in this paper on a large-scale simulation of [8], which shows results for several testcases. The large-scale simulation is based on a simple network topology of a grid of 10-by-7 nodes. Two isolated regions of the network are jammed. Each region comprises a 3-hop boundary of 13 nodes each, that both receives and forwards traffic. Table I shows the results for the affected nodes. Chamaeleon clearly outperforms a single channel solution in both data yield and retransmissions. The results indicate a good scalability of the Chamaeleon protocol with regards to network size.

V. CONCLUSION

In this paper we have addressed the problem of external interference on the 802.15.4 radios hampering the long-running operation of WSNs. Using multiple frequencies is an attractive option to mitigate interference, but raises the complexity of the MAC and routing layers as neighboring nodes may no longer operate on the same channel.

We have proposed a protocol extension, called Chamaeleon, that interfaces between the routing and MAC layer. It provides an efficient mechanism to switch channels at the granularity of a parent and children pair in a typical collection tree supporting a distributed approach and frequency reuse. A parent determines whether or not to switch and flags the decision on the corresponding software ACK. Such a coordinated switch works well when conditions steadily deteriorate. In the case of a sudden, complete communication failure, both parent and children autonomously switch to the next channel on a predefined hopping sequence.

Experiments with TOSSIM have shown that Chamaeleon is quick to respond to various (localized) interference scenarios keeping data delivery yield fairly constant and reducing the number of retransmissions significantly in comparison to the standard single channel approach. These positive initial results motivate to further test Chamaeleon in a dynamic setting and on real hardware.

ACKNOWLEDGMENT

Venkatraman Iyer is supported by the ALwEN project, a Point-One project funded by SenterNovem. Matthias Woehrle was supported by the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a center supported by the Swiss National Science Foundation under grant number 5005-67322.

REFERENCES

[1] G. Zhou, J. Stankovic, and S. Son, "Crowded spectrum in wireless sensor networks," in *3rd Workshop on Embedded Networked Sensors (EmNets 2006)*, 2006.

[2] Y. Kim, H. Shin, and H. Cha, "Y-mac: An energy-efficient multi-channel mac protocol for dense wireless sensor networks," in *IPSN '08: 7th Int'l Conf. on Information processing in sensor networks*, 2008, pp. 53–63.

[3] C.-J. M. Liang *et al.*, "Racnet: Reliable acquisition network for high-fidelity data center sensing," Microsoft Research, Tech. Rep. MSR-TR-2008-144, 2008.

[4] R. Musaloiu and A. Terzis, "Minimising the effect of wifi interference in 802.15.4 wireless sensor networks," *Int. J. Sen. Netw.*, vol. 3, no. 1, pp. 43–54, 2007.

[5] H. K. Le, D. Henriksson, and T. Abdelzaher, "A practical multi-channel media access control protocol for wireless sensor networks," in *IPSN '08: 7th Int'l Conf. on Information processing in sensor networks*, 2008, pp. 70–81.

[6] A. Dunkels, F. Osterlind, and T. Voigt, "Improving sensor network robustness with multi-channel convergecast," in *2nd ERCIM Workshop on e-Mobility*, 2008.

[7] G. Lu, B. Krishnamachari, and C. S. Raghavendra, "An adaptive energy-efficient and low-latency mac for tree-based data gathering in sensor networks: Research articles," *Wirel. Commun. Mob. Comput.*, vol. 7, no. 7, pp. 863–875, 2007.

[8] V. Iyer, M. Woehrle, and K. Langendoen, "Chamaeleon - exploiting multiple channels to mitigate interference," Delft University of Technology, Tech. Rep. ES-2010-02, Feb. 2010.