A New Efficient Mechanism for Establishing IP Connectivity between Ambient Networks

Rui Campos and Manuel Ricardo
INESC Porto, Faculdade de Engenharia,
Universidade do Porto
Porto, Portugal
{rcampos,mricardo}@inescporto.pt

***Abstract*—The changes in the communication paradigm envisioned for future networks, with peer-to-peer/symmetric attachments gaining momentum and two IP (Internet Protocol) versions coexisting, will pose new challenges to mobile communication networks. Traditional IP auto-configuration mechanisms will not work properly, since they were designed mostly having in mind a client-server/asymmetric attachment model, they assume a single IP version paradigm, and they target the auto-configuration of devices only. The IST Ambient Networks project has introduced a new concept – the Ambient Network – that enables handling every communication entity, either a single device or an entire network, as an Ambient Network (AN). This paper describes a new efficient mechanism, named Basic Connectivity (BC) mechanism, for auto-configuring IP connectivity between attaching ANs. A proof-of-concept prototype, experimental results, and theoretical analysis show that BC suites the future networking paradigm and represents a solution more efficient than the current trial-and-error mechanism for auto-configuring IP connectivity.**

Keywords: Ambient Network, IP connectivity, auto-configuration

I. INTRODUCTION

In recent years, we have witnessed the deployment of many communication and networking technologies, from which the wireless field is a prominent example. Heterogeneity in communication and networking implies interworking problems. These problems are now being faced in IP networks. Two IP versions are expected to coexist for a long time and multiple IP auto-configuration mechanisms may be in place, bringing up problems either concerning the interconnection of simple devices or entire networks. On the other hand, new networking paradigms increasingly assume a symmetric, peer-to-peer relationship between communicating peers, and network components that start having dynamic roles. For instance, a device may act as a simple terminal at a given moment and also as an IP gateway at a subsequent moment, for providing Internet access within a Personal Area Network (PAN) [1][2]. Legacy attachment procedures are typically asymmetric and obey to a client-server model. Also, the roles of each party and the network services they offer are pre-defined. For example, a terminal (the client) attaches to an infrastructure network (the server) and runs a Dynamic Host Configuration Protocol (DHCP) [3] client to acquire IP configuration parameters from the DHCP server running in the infrastructure.

The communication paradigm assumed within the IST Ambient Networks project [4] considers both asymmetric and symmetric attachments between Ambient Networks (ANs), a new concept introduced by the project that enables handling every communicating entity as an AN. In symmetric attachment, peers have similar capabilities and both can request/offer network services; for example, within a PAN every device may be able to run a DHCP server for auto-configuring IP connectivity [1][2]. In the AN paradigm, attaching devices and/or networks cannot assume any type of network configuration service to be deployed by its peer, since there are multiple possibilities. Even when there is a clear definition of roles, such as in the case of a terminal attaching to an infrastructure network, the coexistence of two IP versions and multiple auto-configuration mechanisms may render IP connectivity auto-configuration difficult and inefficient. Using state of the art solutions, terminals try every possible local auto-configuration mechanism until a mechanism hopefully succeeds. This is not the most efficient solution, namely when the frequency of attachments a network device may perform is high, such as in scenarios where mobility and dynamics are present.

Concerning the attachment between networks, network layer heterogeneity brings up further problems. For instance, when networks with incompatible address spaces attach, some mechanism needs to be provided for internetworking. Moreover, plug and play attachment between the networks is expected to be supported, in a new paradigm where user configuration efforts should be avoided. Nowadays, there is not a generic solution solving these problems. A mechanism was defined by the IETF to enable internetworking between IPv4 and IPv6 networks, named Network Address Translation – Protocol Translation (NAT-PT) [5]. Nonetheless, it suffers from the same problems as the IPv4 NAT [6], namely it limits end-to-end connectivity and, in its simplest form, it only provides one-way connectivity between an IPv6 network and a peer IPv4 network. On the other hand, the usage of private IP address spaces may cause that two attaching ANs run on overlapping address spaces. Currently, there is no standard solution addressing this problem. In the IETF MANET AUTOCONF the problem has been raised regarding the merging of mobile ad-hoc networks (MANETs) [7] and draft solutions have been proposed [8][9]. They are based on defining MANET auto-configuration mechanisms that: 1) assign statistically unique addresses to each MANET device in order to avoid duplicate addresses when network merging occurs; 2) provide means for duplicate address detection on a

per-device basis, so that address conflicts are detected and solved when network merging occurs. Still, these proposals target MANET scenarios in particular. The assumption of assigning statistically unique addresses is not applicable in general.

These problems represent a motivation for defining a new mechanism coping with new communication paradigms. We propose the Basic Connectivity (BC) mechanism which enables plug and play IP connectivity between attaching peers, either devices or networks. Our proof-of-concept prototype demonstrates that the BC mechanism suites the envisioned networking scenarios. Additionally, experimental results and theoretical analysis show that the BC mechanism represents a solution more efficient than the current IP auto-configuration trial-and-error mechanism.

The remainder of the paper is organized as follows. Section II describes the BC mechanism, Section III presents the BC mechanism proof-of-concept prototype, Section IV provides the evaluation of the mechanism through theoretical analysis and experimental results and, finally, Section V draws the conclusions.

II. Basic Connectivity Mechanism

The Basic Connectivity Manager (BCM) is the central entity of the BC mechanism. It manages the establishment of IP connectivity between attaching peers (nodes or networks). BCM communicates with peer BCMs for negotiating the proper IP version, auto-configuration mechanism, and addressing scheme (when establishing connectivity between networks). Furthermore, it interacts with peer BCMs to coordinate the configuration of the local services and resources, network layer, and local auto-configuration mechanisms accordingly. The BC mechanism does not implement any auto-configuration mechanism by itself. Rather, it selects the proper local auto-configuration mechanism, such as DHCP [3] or IPv6 Stateless Address Auto-configuration [10], and relies on it for establishing IP connectivity. The flexibility of the BC mechanism allows easy integration of new auto-configuration mechanisms, e.g., MANET auto-configuration mechanism, and permits the selection of the proper auto-configuration according to the different contexts of a network node. For instance, if a specific node is connecting to a MANET it shall use a MANET auto-configuration mechanism; conversely, if it is connecting to a node deploying the Dynamic Configuration of IPv4 Link-local Addresses mechanism [11], it has to use this mechanism instead.

In the attachment between nodes, peer BCMs negotiate only the IP version and auto-configuration mechanism to be used for establishing IP connectivity, and select the agreed IP version and auto-configuration mechanism accordingly. Concerning the attachment of networks, the AN project assumes the existence of a Border Node (BN) [12] located at the border of the network, representing it to the outside world. Also, the project defines a new internetworking layer, named Node ID (NID) layer [12], sitting between the network layer (IP) and the transport layer, which is the lowest common communication layer within the AN framework. When dealing with the attachment of networks, besides establishing connectivity between the Border Nodes (BN) of the networks, BCMs negotiate how to interconnect the address spaces of the networks. The announcement of the internal address space depends on internal policies and leads to two scenarios. When internal policies disable revealing the internal addressing, BCM does not announce its internal address space to the peer network, and configures the local BN as a Node ID (NID) router, hiding the address space behind it. On the other hand, if internal policies enable revealing the address space, but the networks support incompatible address spaces, BCMs agree on the configuration of, at least, one NID router in one of the BNs. If networks have compatible address spaces, the BCMs configure one or two IP (IPv4 or IPv6) routers between the networks.

Fig. 1 shows the architecture of the BC mechanism. The Basic Connectivity Protocol (BCP) is used to exchange control information between BCMs. Within the AN framework, the BCP information is transported by the Ambient Network Attachment Protocol (ANAP) [13]. ANAP copes with the establishment of a security association between attaching peers and consists of a 4-way message exchange. When running between adjacent ANs, i.e., ANs sharing the same wired or wireless technology (e.g., WLAN, Ethernet), ANAP runs over Layer 2 protocols [14]. The information elements included in the BCP messages are piggybacked over these four ANAP messages. BCP considers three messages: NEGOTIATE, AGREEMENT, and DONE. Each message starts with a common header (1 byte), which includes the type of the message; the other information elements are specific to each message. The BCP messages are shown below using the ABNF syntax [15].

The NEGOTIATE message is defined as follows:

```
NEGOTIATE      = message-type
                 node-type
                 network-layer-versions-supported
                 autoconf-mechanisms-supported
                 [address-space-type]
                 [netmask-length]
                 [subnet-address]
```

where *node-type* (1 byte) defines the type of device sending the message, a stand-alone node or a BN of an AN, and *network-layer-versions-supported* (1 byte) and *autoconf-mechanisms-supported* (1 byte) specifies the network layer versions (e.g., IPv4), and auto-configuration mechanisms supported by the current node, respectively. The a*ddress-space-type* (1 byte), e.g., IPv4 address space, the *netmask-length* (1 byte), and *subnet-address* (variable length), e.g., 192.168.10.0, are optional information elements that refer to attachments performed between networks (ANs). Thus, the size of the NEGOTIATE message depends on the type of node running

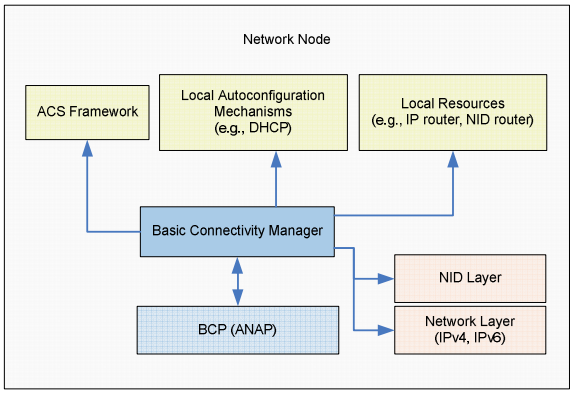

Fig. 1. BC mechanism architecture.

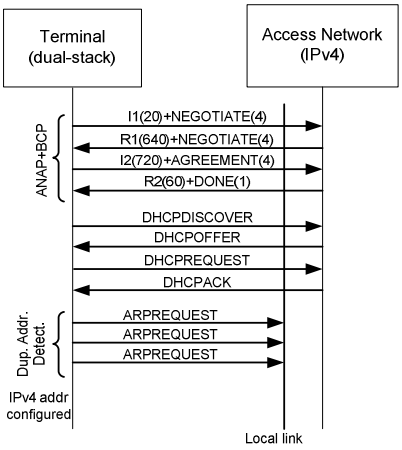

Fig. 2. Message sequence chart illustrating the execution of the BC mechanism when a terminal attaches to an access network.

the BC mechanism. If the mechanism is run by a stand-alone node, the size of the message is equal to 4 bytes. Conversely, if the mechanism is run by a BN, the optional information elements are in place and the size is greater. The actual size of the message depends on the type of address space. For instance, for an IPv4 address space the size of the message is 10 bytes, whereas for an IPv6 address space it is 22 bytes.

The AGREEMENT message includes the following information elements:

AGREEMENT = message-type
network-layer-version
autoconf-mechanism
local-remote-autoconf-server

where *network-layer-version* (1 byte) is the network layer version agreed to be used for establishing connectivity between the communicating peers, *autoconf-mechanism* (1 byte) specifies the auto-configuration mechanism to be used for address configuration, and *local-remote-autoconf-server* (1 byte) defines which peer deploys the server part of the auto-configuration mechanism, if applicable (e.g., DHCP). The size of the message is 4 bytes.

Finally, the DONE message includes the *message-type* field only, since no further information needs to be transferred between the peers. This message is used for terminating the process.

The execution of the BC mechanism is illustrated in Fig. 2 considering the scenario of a terminal attaching to an IPv4 access network; only the right IP auto-configuration mechanism is run over the local link. The size (in bytes) of the ANAP messages and the BCP information elements are shown between parentheses.

III. Basic Connectivity Mechanism Prototype

We have implemented the BC mechanism under FreeBSD Operating System (OS), using C++. Fig. 3 shows the class diagram of the BC mechanism prototype. Below, we provide a brief description of each class:

- **CBcm** is the core class. It implements the procedures assigned to the BCM in the overall BC mechanism and interacts with the other classes of Fig. 3.
- **CBcp** implements the BCP protocol. When *CBcm* needs to send signaling information to a peer BCM, it passes the corresponding information elements to *CBcp*, which is responsible for creating the proper BCP message and to send it towards the peer. On the other hand, *CBcp* is in charge of (1) handling incoming BCP messages to retrieve information elements transported on it, and (2) passing these elements to *CBcm*.

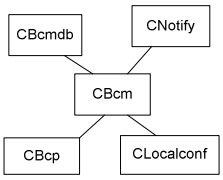

Fig. 3. BC mechanism prototype class diagram.

- **CLocalconf** deals with the interaction with local resources and auto-configuration mechanisms. After the negotiation phase, *CBcm* interacts with this class in order to perform the required configurations, such as "start DHCP server" and "configure IPv4 router".
- **CBcmdb** reads a local configuration file specifying the characteristics of the current node and the characteristics of the network to which it belongs, when the node is a BN. Information present in this configuration file includes: type of node (single node or BN), local IP versions and auto-configuration mechanisms supported and, if applicable, the address space being used within the network to which the BN belongs to.
- **CNotify** copes with notifications towards the Ambient Control Space [13]. Different notification mechanisms are used depending on the entity being notified; *CNotify* deals with this in order to render the process fully transparent from *CBcm*'s standpoint.

The BC prototype makes part of the overall AN prototype [16] and was demonstrated at the final audit of the project.

IV. Evaluation

This section is devoted to the evaluation of the BC mechanism. Our purpose is to evaluate the benefits of considering the BC mechanism as part of the attachment procedure defined by the ANAP protocol within the AN framework [13]. We name the trial-and-error mechanism used to establish IP connectivity, when multiple IP versions are present, as legacy mechanism. In order to compare the two mechanisms, the ANAP protocol is used in both mechanisms, for establishing a security association between attaching peers (ANs), as defined in [13]. The comparison is performed based on the overhead, auto-configuration delay, and energy consumption. The evaluation of BC from the functional point of view was carried out using the BC prototype.

A. Scenarios

The scenario shown in Fig. 4 (Scenario 1) was considered for evaluating the BC mechanism from a functional point of view. It consists of the attachment of two nodes connected to the same link, both nodes supporting a DHCP client and a DHCP server, as shown in Fig. 4. In this scenario, using the legacy mechanism, either IP connectivity was not established or it was established but two IP networks were created over

the link, which is clearly unnecessary. In the first case, the DHCP clients running in each node accepted the DHCPOFFER sent by the DHCP server running in the same node and no IP connectivity was configured at all. In the second case, each DHCP client accepted the DHCPOFFER of the server running in the peer node, and two IP networks were established over the same link; this problem would also be observed for DHCPv6 or if the attachment between IPv6 routers was considered. The BC mechanism solved the problem by defining, during the negotiation phase, the node running the DHCP client and the node running the DHCP server; it guaranteed that a single DHCP server was running after the attachment procedure completed, and that a single IP network was created. A scenario considering the attachment between networks was also tested using the BC prototype. For further details please refer to [12].

The two scenarios illustrated in Fig. 5 (Scenario 2 and 3) were used to compare the two mechanisms. Scenario 2 is purely symmetric and refers to the creation of a PAN. Scenario 3 is purely asymmetric and considers the attachment of multiple terminals to an access network. In both scenarios we assume the devices are dual-stack. IPv4 and IPv6 and their auto-configuration mechanisms were considered in the analysis. For IPv4, the DHCP protocol [3] and the Dynamic Configuration of IPv4 Link-Local Addresses [11] along with the Address Resolution Protocol (ARP) [17], used for duplicate address detection, were considered. For IPv6, the IPv6 Stateless Address Auto-configuration [10] and DHCPv6 [18], together with the Neighbour Discovery Protocol (NDP) [19] and the Multicast Listener Discovery (MLD) protocol [20] were considered.

B. Results

This section presents the results obtained for Scenario 2 and 3. We characterize the auto-configuration delay, the overhead (total number of bytes of signaling exchanged), and the wasted energy. Overheads do not consider Layer 2 headers, so the results are made independent of the medium used. The analysis was performed using messages sizes according to the specifications [3] [10] [11] [17] [18] [19] [20] [21] [22], or according to the typical sizes found in practice when standards leave values as open. The results provided herein are meant to give insights on the differences between the use of the legacy and BC mechanisms.

The plots of Fig. 6 show the total overhead incurred by the attachment process when the legacy and the BC mechanisms are used in Scenario 2 and 3 and IPv6 is used to establish IP connectivity; similar plots for IPv4 can be found in [12]. The plot on the left-hand side refers to Scenario 2 and the plot on the right-hand side refers to Scenario 3. For Scenario 3, the total overhead represented by the curves considers one attachment per device performed while forming the PAN. There is a significant saving provided by the BC mechanism. For instance, for a PAN of 30 devices the BC mechanism saves about 60 kB of signaling, whereas for 50 devices it can save about 100 kB. The overhead reduction – the difference between overheads normalized to the legacy mechanism overhead – is shown in Table I. The reduction is slightly higher when IPv4 is used to establish IP connectivity within the PAN. For Scenario 3, it is worth noting that the BC mechanism saves about 0,5 MB and 1 MB of signaling when 500 and 1000 terminals, respectively, perform a single attachment to the access

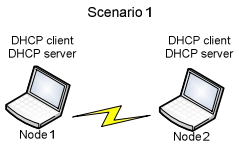

Fig.4. Attachment between two nodes supporting both DHCP client and server.

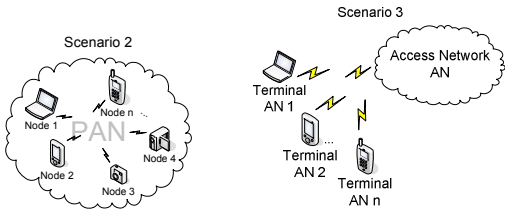

Fig. 5. Two scenarios used to compare the overhead incurred by legacy mechanism and the BC mechanism.

network. The saving is even more relevant if we consider multiple attachments per terminal. For instance, if 10 attachments per terminal are considered, the BC mechanism saves about 5 MB for 500 attaching terminals and about 10 MB for 1000 attaching terminals. The overhead reduction provided by the BC mechanism with respect to the legacy mechanism is also shown in Table I. As in Scenario 2, the reduction is slightly higher when IPv4 is actually used.

Table I. Overhead reduction when using the BC mechanism for Scenario 2 and Scenario 3.

Scenario	IP version actually used	Overhead Reduction (%)
2	IPv4	30
	IPv6	25
3	IPv4	60
	IPv6	55

In order to assess the energy wasted by the legacy mechanism, due to the use of a trial-and-error approach, we have considered Scenario 2 and an 802.11b IPv4 access network. This energy is saved by the BC mechanism and can be used to send user data instead. The additional energy required to transmit the BCP information elements piggybacked over the attachment protocol is negligible, since the fixed energy cost of transmitting/receiving packets is more significant [23]. The plots of Fig. 7 show the amount of energy wasted when using the legacy mechanism in this scenario, and the amount of data that could be transmitted/received by an 802.11b Network Interface Card (NIC) using the wasted energy. The plots were obtained by using the linear equations provided in [23] for calculating the energy consumed by an 802.11b NIC when sending/receiving unicast/broadcast packets. Different equations are provided for each case: 1) unicast packet transmission; 2) unicast packet reception; 3) broadcast packet transmission; 4) broadcast packet reception. In our calculations we considered that terminals attach sequentially; as such, the number of nodes receiving the broadcast messages sent out by an attaching terminal increases linearly as new devices attach. The leftmost plot in Fig. 7 was obtained by considering all the unnecessary messages broadcasted by the legacy mechanism when *n* terminals attach to the access network. These messages are both unnecessarily

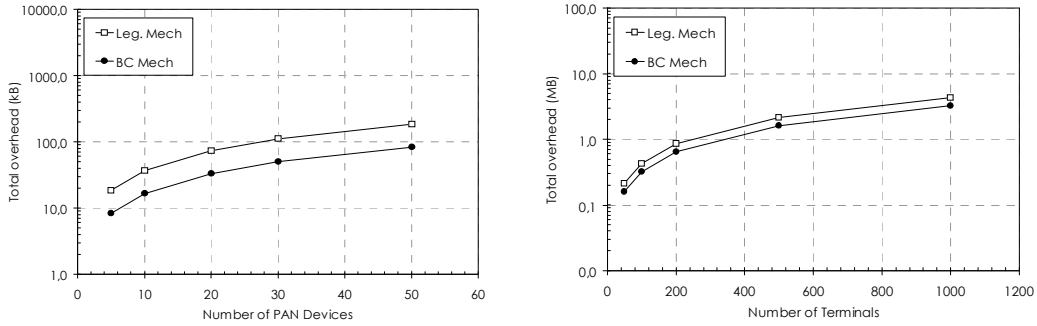

Fig. 6. Total overhead incurred by each mechanism for Scenario 2 and 3 when IPv6 is used for establishing IP connectivity.

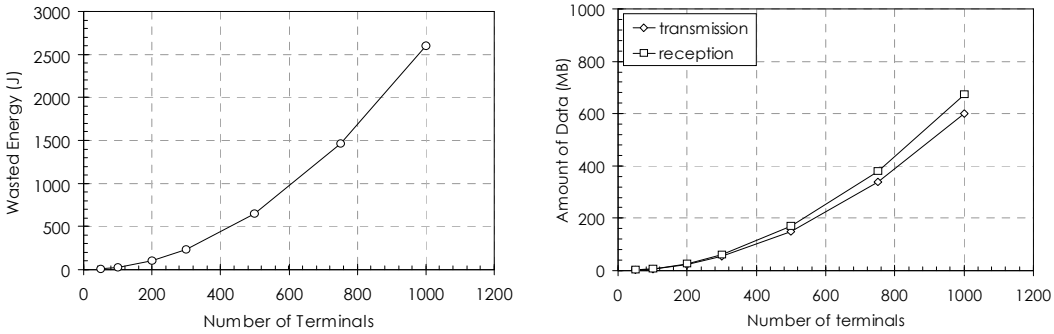

Fig. 7. Total energy wasted in Scenario 3 by the legacy mechanism and the amount of data that could be transmitted/received by an 802.11b NIC using the wasted energy.

transmitted by a node and received by the terminals already attached to the access network; both transmission and reception of messages contribute to the total wasted energy. The total wasted energy increases exponentially with *n*. For *n*=1000, about 2600 J of energy is wasted when using the legacy mechanism, which corresponds to the energy required by an 802.11b NIC to transmit/receive more than 600 MB of data (cf. rightmost plot in Fig. 7). Different results would be obtained if other NIC was considered; here we aim at reasoning about the order of magnitude of the wasted energy.

Now, we present the auto-configuration delay introduced by each mechanism in Scenario 2. The delay per PAN device during the auto-configuration of an IPv4 address was measured experimentally using Ethereal [24]; the logs are available at [25]. Ten samples were taken into account. Table II presents the measured mean delay and the corresponding 95% confidence interval, as well as the delay reduction – the difference between delays normalized to the legacy mechanism delay. It is important to realize that these values are implementation dependent; for instance, the number of retransmissions of the DHCPDISCOVER message before the device gives up (in our case 3 retransmissions), will have impact on the auto-configuration delay.

Table II. Auto-configuration delays introduced by the BC and legacy mechanisms in Scenario 2.

Parameter	BC mechanism (s)	Legacy mechanism (s)	Delay Reduction (%)
Mean	3.0	61.8	95.2
Confidence Interval (95%)	[2.2, 3.7]	[61.6, 62.0]	[94.0, 96.4]

The delay introduced by the legacy mechanism is greater than 1 minute, while the BC mechanism takes 3.0 seconds, on average, to complete. This has to do with the trial-and-error approach assumed by the legacy mechanism. When using the legacy mechanism, a PAN device first tries to auto-configure an IPv4 address using DHCP by sending out multiple DHCPDISCOVER messages until it realizes there is not any DHCP server running. Subsequently, it tries the alternative auto-configuration mechanism and configures an IPv4 link-local address. The BC mechanism saves 58.8 seconds, on average, by negotiating in advance the proper auto-configuration mechanism to be used. During the negotiation phase, BCMs can conclude that IP address auto-configuration can only be performed using the Dynamic Configuration of IPv4 Link-Local Addresses mechanism. Consequently, the delay introduced by the attempt to auto-configure IP connectivity through DHCP is eliminated from the BC mechanism.

C. Discussion

The heterogeneity currently found in IP networks and the advent of a new communication paradigm brings up new problems. State-of-the-art solutions either do not deal with these problems at all or do not deal with them in the most efficient way. The BC mechanism addresses the new problems and represents a solution more efficient than the legacy mechanism used to auto-configure IP connectivity. By testing Scenario 1 we have confirmed that the BC mechanism can deal with the new envisioned communication paradigm, where symmetric attachments can take place, devices may have dynamic roles, and two IP versions and corresponding auto-configuration mechanisms are present. We have demonstrated that the BC mechanism can establish IP connectivity in scenarios where the legacy mechanism cannot. On the other hand, through experimental and theoretical analysis, we verified that the BC mechanism is significantly more efficient

than the legacy mechanism. The BC mechanism avoids both the transmission and reception/processing of unnecessary messages by each network device connected to the link over which IP connectivity is being configured. This allows energy and CPU time saving, an important issue namely for devices with limited resources and running on battery power, such as handheld and sensor devices. In addition, the BC mechanism avoids the auto-configuration delay that may be introduced by the legacy mechanism, which renders IP auto-configuration a slow process; for example, the bootstrapping of a PAN may take more than one minute. We shall observe that the actual overhead and delay reduction provided by the BC mechanism depends on multiple factors, such as the scenario being considered, the network layer, and the corresponding auto-configuration mechanism used for IP connectivity configuration. Also, the actual energy saving depends on the hardware and the specific technology(ies) considered.

In the evaluation presented we considered IPv4 and IPv6 for establishing IP connectivity and their corresponding auto-configuration mechanisms. Moreover, we evaluated the BC mechanism for two representative scenarios. For these scenarios the BC mechanism provides significant reductions in overhead and auto-configuration delay, and it may save a considerable amount of energy. The benefits of BC would be even more significant if: 1) further possible network layers and/or auto-configuration mechanisms were considered; 2) the number of retransmissions of the DHCPDISCOVER message, while attempting to auto-configuring IP connectivity using DHCP, was higher. Our analysis considered 2 retransmissions, according to [3]. In practice, higher number of retransmissions may take place. For instance, the DHCP implementation under Linux OS considers 5 retransmissions. In that case, the overhead and delay reductions and the wasted energy would be higher.

Although the BC mechanism was designed having ANs in mind, and with the purpose of being integrated into the Ambient Network framework [13], it can be used in other setups. BC can be integrated in any existing attachment procedure. For instance, it could be integrated within the 802.11 attachment procedure usually carried out using the Extensible Authentication Protocol (EAP) [26], with the BCP information elements piggybacked over the EAP messages.

V. Conclusion

The paradigm in mobile communication networks is changing, with symmetric attachment between peers gaining momentum and two IP versions coexisting. In this paper we presented the BC mechanism, a mechanism used to establish IP connectivity between communicating peers (ANs). By using a proof-of-concept prototype, experimental results, and theoretical analysis, we showed the usefulness of the mechanism, both from the efficiency and functional points of view. The analysis performed focused on overhead, auto-configuration delay, and energy consumption, and it gives the first insights on the real benefits provided by the BC mechanism. The evaluation of the mechanism with respect to other metrics is left for future work.

Acknowledgments

This paper describes work undertaken in the context of the Ambient Networks project, which is part of the EU's IST program. Over 30 organizations from Europe, Canada and Australia are involved in this Integrated Project, which runs in 2006-2007 in its second phase. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the Ambient Networks Project.

References

[1] R. Campos et al., *Dynamic and Automatic Interworking between Personal Area Networks using Composition*, in Proc. of 16th Annual IEEE Int. Symp. on Pers. Indoor and Mob. Radio Comm., Berlin, Germany, Sep. 2005.
[2] R. Campos and M. Ricardo, *Dynamic and Automatic Connection of Personal Area Networks to the Global Internet*, in Proc. of ACM IWCMC'06, Jul. 2006.
[3] R. Droms, *Dynamic Host Configuration Protocol*, IETF RFC 2131, Mar. 1997.
[4] IST FP6 Ambient Networks. <http://www.ambient-networks.org>.
[5] G. Tsirtsis et al., *Network Address Translation – Protocol Translation (NAT-PT)*, RFC 2766, Feb. 2000.
[6] P. Srisuresh and M. Holdrege, *IP Network Address Translator (NAT) Terminology and Considerations*, IETF RFC 2663, Aug. 1999.
[7] S. Singh et al., *Address autoconfiguration for MANETs: definition and problem statement*, Internet Draft, draft-singh-autoconf-adp-03, Mar. 2006.
[8] J. Jeong et al., *Ad Hoc IP Address Autoconfiguration*, Internet Draft, draft-jeong-adhoc-ip-addr-autoconf-06, Jan. 2006.
[9] F. Ros et al., *Extensible MANET Auto-configuration Protocol (EMAP)*, Internet Draft, draft-ros-autoconf-emap-02, Mar. 2006.
[10] S. Thomson et al., *IPv6 Stateless Address Autoconfiguration*, IETF RFC 4862, Sep. 2007.
[11] S. Cheshire et al., *Dynamic Configuration of IPv4 Link-Local Addresses*, IETF RFC 3927, May 2005.
[12] B. Ahlgren et al., *Connectivity and Dynamic Internetworking Prototype and Evaluation*, FP6-CALL4-027662-AN P2/D23-E.2, V1.0, Ambient Networks project public deliverable, Dec. 2007.
[13] M. Johnsson et al., *Ambient Network System Description*, FP6-CALL4-027662-AN P2/ D07-A2, V7.0, Dec. 2007.
[14] T. Rinta-aho, R. Campos, A. Mehes, U. Meyer, J. Sachs, G. Selander, *Ambient Network Attachment*, in Proc. of IST Summit, Budapest, Jul. 2007.
[15] D. Crocker et al., *Augmented BNF for Syntax Specifications: ABNF*, RFC 4234, Oct. 2005.
[16] R. Rembarz et al., D17-*H.4 Ambient Control Space Prototype Design*, FP6-CALL4-027662-AN P2/D17-H.4, V1.0, Jun. 2007.
[17] D. Plummer, *An Ethernet Address Resolution Protocol*, IETF RFC 826, Nov. 1982.
[18] R. Droms et al., *Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*, IETF RFC 3315, Jul. 2003.
[19] T. Narten et al., *Neighbor Discovery for IP version 6 (IPv6)*, IETF RFC 4861, Sep. 2007.
[20] R. Vida and L. Costa, *Multicast Listener Discovery Version 2 (MLDv2) for IPv6*, IETF RFC 3810, Jun. 2004.
[21] S. Alexander and R. Droms, *DHCP Options and BOOTP Vendor Extensions*, IETF RFC 2132, Mar. 1997.
[22] R. Droms, *DNS Configuration options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*, IETF RFC 3646, Dec. 2003.
[23] L. Feeney and M. Nilsson, *Investigating the energy consumption of a wireless network interface in an ad hoc networking environment*, in Proc. of INFOCOM 2001, Anchorage, 2001.
[24] Network Protocol Analyzer, *Ethereal*. < http://www.ethereal.com>.
[25] Log files. <http://telecom.inescporto.pt/~rcampos/logsIPautoconf.zip>.
[26] B. Aboba et al., *Extensible Authentication Protocol (EAP)*, IETF RFC 3748, Jun. 2004.