Basic Abstractions for an Autonomic Network Architecture

Christophe Jelger, Christian Tschudin
Computer Science Department
University of Basel, Switzerland
Firstname.Lastname@unibas.ch

Stefan Schmid
NEC Europe Ltd.
Heidelberg, Germany
Stefan.Schmid@netlab.nec.de

Guy Leduc
EECS Department
University of Liège, Belgium
Guy.Leduc@ulg.ac.be

Abstract

The ANA (Autonomic Network Architecture) project aims at providing a framework to flexibly host, interconnect, and federate multiple heterogeneous networks in an autonomic way, i.e. without requiring active human intervention. The guiding design principle is to strive for a maximum degree of flexibility at all levels of the architecture in order to inherently support heterogeneity and evolution. This paper describes the core abstractions and concepts of ANA (as defined during the first year of the project) and introduces their basic operation and interaction. While not autonomic themselves, the core architectural principles of ANA will enable autonomicity by not imposing a "one-size-fits-all" approach where protocols and paradigms are fixed by the architecture. We indeed argue that only the capacity of the network to be polyfunctional and fully adaptable will justify the label 'autonomic'. A prototype of ANA is currently being developed and will be released in 2008: the goal is to demonstrate the feasibility of autonomic networking within the 4 years of the project.

1 Introduction

The goal of the ANA project [1] is to explore novel ways of organising and using networks beyond legacy Internet technology. We are designing and developping a novel network architecture that can demonstrate the feasibility and properties of autonomic networking. The main guiding principle behind the architectural work in ANA is to strive for a maximum degree of flexibility and support *functional scaling* by design at all levels of the architecture. Functional scaling means that a network is able to extend both horizontally (more functionality) as well as vertically (different ways of integrating abundant functionality). New functions must be integrated, otherwise we do not have scaling of functionality, only function accumulation.

ANA is a meta-architecture in the sense that it is a framework to host, interconnect, and federate multiple heterogeneous networks. Unlike the Internet which relies on a unique and globally shared addressing scheme, ANA is not another "one-size-fits-all" network waist. As advocated in recent research in network architectures [8, 10, 14], ANA accepts network heterogeneity as a base fact and focuses on providing a set of basic networking abstractions which can support and "glue" together multiple networking styles and instances. The main challenge of the project is that ANA enables and demonstrates autonomic functionality, i.e. that the network(s) operates with as little human intervention as possible.

1.1 Self-Star in Networking

The term 'autonomic' refers to the ability of a system to perform its operation via (so-called) self-* properties i.e., without requiring active human intervention. In networking, autonomicity relates to the capability of nodes to *self-organize* into a network through local interactions with neighboring devices [15]. This typically requires *self-configuration* mechanisms so that nodes are able to automatically setup core networking items such as addresses and names. Heterogeneous network clouds (formed e.g., by different technologies or administrative boundaries) must be able to *self-federate* into larger networks in order to form a federation of networks with global reachability. Additionaly, and in order to prevent service disruptions and attacks, the network nodes must also implement *self-protection* techniques, and be able to recover from incidents through *self-healing* by restoring and potentially reconfiguring the operation of the network.

Current network protocols cover quite some part of the many self-* properties listed above. However, today these control loops are (rightly) confined to narrow tasks but are usually not interlinked. Adding more flexibility in the way a network solves its tasks, which is essential for letting the network architecture evolve, would currently fail because of the interaction complexity. This is where an autonomic approach with its self-* properties comes into the picture, so that despite increased complexity we can continue to change existing and add new functionality to our networks.

1.2 Organization of the paper

The rest of this paper is organized as follows. The next section discusses why the limited evolvability of the Internet core is triggering a renewed interest in network architecture design. Section 3 highlights some of the main principles and abstractions of ANA and provides examples to illustrate these concepts. Section 4 outlines some of the expected autonomic features of ANA with an emphasis on network re-configuration. The paper concludes with a section briefly summarizing future work in the project (in 2007).

2 Why do we need a new architecture?

A recurrent topic in recent literature is whether the core principles of the Internet should be relaxed in order to allow unconstrained innovations to appear [6]. For example, Clark admits in [9] that "the end-to-end arguments are still valid and powerful, but need a more complex articulation in today's world". This does not question the incredible success of the Internet but its ability to integrate, as part of its core operation, new networking paradigms to support the services provided by e.g., telephone, radio, cable, and TV networks [13]. The *waist* of the Internet, i.e. the IP protocol, the end-to-end principle, and end-to-end connectivity, was indeed not meant to be negotiable and new functionalities like firewalls and NAT were added in a "patch" style that highlights the limited flexibility of the core architecture. This inability to evolve also relates to the limited success (in terms of deployment) of numerous projects like RSVP, IP multicast and mobility, and IPv6.

Recently there has been a strong renewed interest in the design of *clean-slate* network architectures (see e.g. [5, 11, 2]) and infrastructures [3, 4], which indicates that the research community is looking for an alternative to the Internet to carry out innovative and unconstrained research in networking. In this sense, "the time is right" to propose novel network architectures that do not necessarily need to be fully backwards compatible with the Internet. The research community is currently receptive to alternative network designs and in the next few years we will witness the birth of many new network architectures.

3 Defining a Network Architecture with an Autonomic Twist

A network architecture is a set of design principles describing the scope, the objectives, and the abstract operation of a communication system. It defines the atomic functions and entities that compose the network and specifies the interactions which occur between these various building blocks. While some network architectures solely define a set of high-level principles and goals, others also consider details such as packet formats and implementation directives. Moreover, defining a network architecture also implies outlining what is not part of the architecture, e.g. what is left open to implementers. For example the Internet architecture specifies that IP is the addressing protocol while the OSI networking model specifies that there is a network layer but does not mandate any specific addressing scheme.

In ANA we have started to isolate those networking abstractions that we consider key in order to let autonomic functionality "play" with the various ways networks can be implemented and be operated. Our view is that ANA has to provide a minimal yet generic set of networking concepts which permits different networking styles to be expressed. The result will be some sort of axiomatic definition of basic abstractions which an actual implementation will have to support. In this section we present a snapshot of the current abstractions and concepts that will form the backbone of the ANA.

3.1 Network Compartments

At the coarsest level, the ANA world is organized in *network compartments*, shown with dashed lines in the following figures. As a first approximation, a network compartment is similar to the concept of a *realm* as previously defined in the literature [7, 12]. However, the term realm is generally used for a network that autonomously manages its own private address space. This definition is restrictive in the sense that it is centered around the addressing scheme. In ANA, a compartment is a more elaborated set of properties: A compartment defines rules stating how and when nodes can join the compartment, how they can find other members of the compartment and how they resolve member information into actual communication channels. The complexity and details of the registration scheme is left to each compartment, e.g. ranging from complex trust-based mechanisms to "simple" registration schemes with a central database or a DHT-based system.

A key concept of compartments in ANA is the notion of *member*, where a compartment can be defined as the set of members which are able, willing and permitted to communicate among each other according to compartment wide policy and protocols. Conceptually, a compartment maintains a (hypothetical) database which contains its members, that is, each entry in the database defines a member. Before one can send a data packet to a compartment member, a resolution step is required which returns a means to "address" the member. The database is defined only through an access procedure and does not need to be realized as an actual database server. The resolution process may even be implicit, for example as it is currently the case with the Internet where an IP address "defines" a member that is reachable, in principle. In a sense, the notion of compartment can be seen as a generalisation of the concept of overlay network which, in ANA, is explicitly defined as a first-class network citizen with standardized abstractions.

Figure 1 illustrates a resolution request for a member "A" which returns a local identifier labeled 'a' that identifies a communication channel via which "A" can be reached. Note that we do not specify whether a member is a node, a set of servers or a software module: what matters is that a resolution request provides access to what we call an *"information channel"* (IC) that leads towards the member.

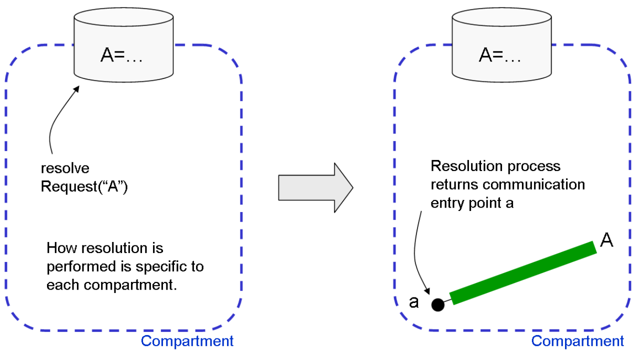


Figure 1. Resolution of a "member".

Registration and resolution are key functionalities that every compartment has to support, either by defining it implicitly (through the addressing and routing schemes), or by performing explicit actions (e.g. resolving an identifier to some local descriptor). The motivation is that ANA supports and federates multiple networking styles and instances that are *wrapped* with the generic compartment abstraction. For example, ANA could potentially host an Internet compartment where e.g., registration would refer to acquiring a (reachable) IP address and resolution would refer to performing a routing table lookup.

3.2 Startpoints instead of Endpoints

In classic network architectures, addresses are used to identify the endpoints of a communication. One problem of this approach is that changing the address format (e.g. IPv4 to IPv6) without changing all parts of the system is almost impossible. Another aspect is that in some contexts there are no addresses at all (content-based routing, for example). In ANA, we chose local labels: A label identifies a so called local *"information dispatch point"* (IDP) to which an information channel can be bound. Using this label, a source can send information to some member, via the information channel that is bound to the label. That is, the sender sees and selects [17] a startpoint, not the endpoint.

The information channel abstraction captures any communication venue, be it a unicast connection, a routing path, a multicast tree, an anycast or a concast tree: the in- and outlets of a channel are IDPs. An IDP is like an interface that is bound to some packet processing entity. For example, when you resolve a peer's name, you will get back a label of the startpoint to use. This binds the local IDP to some remote entity, more precisely to the channel leading to it, in an address-agnostic way. If some address is involved for sending data to a peer, it would be part of the local IDP's state and the application does not have to handle it itself: In this way it is possible to interface with Internet style systems which internally are based on global addresses. Note that one should not induce that IDPs require extra state "in" the network: IDPs call for a clear separation between network-wide identifiers and their node-local representations. Figure 2 illustrates IDPs (shown as black dots) and different types of ICs (shown as thick lines).

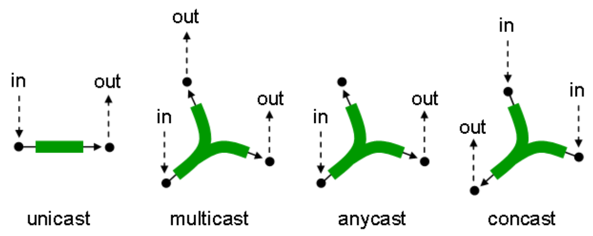


Figure 2. IDPs and ICs.

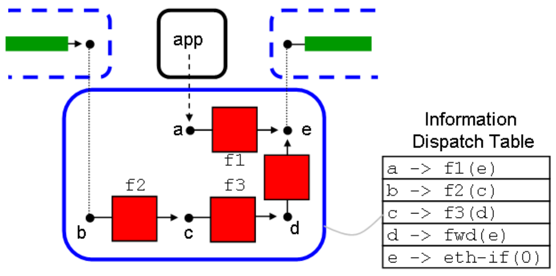
3.3 Functional Blocks

Beside binding to information channels, IDPs can also be bound to *Functional Blocks (FBs)*, shown as square boxes in figures 3, 4, and 6. Functional blocks are packet processing procedures that, if a network device supports it, can be inserted on demand into a data path. What in OSI terminology is a PE (protocol entity), would be represented in ANA by an FB or a set of (composed) FBs.

Unlike traditional layered network architectures, ANA does not restrict how FBs have to be spliced into the data path. In a compartment with layered functionality, FBs would typically be installed in a symmetric way across the network. However, transcoding proxies, local rate controllers and any other functionality that only matters locally, would be handled by ANA in the same way using the functional block abstraction.

3.4 Forward Information Base and Redirection

Information dispatch points permit to formalize the highly desired functionality of redirecting data traffic, at run time. The binding between an IDP and the channel, for example, can be undone and redefined without having to inform the data source, which continues to use the local label. Formally, the bindings (which is what an IDP is about) are collected in a *"forwarding information base"* (FIB) well known in today's routers. While previously a FIB would be used only in the forwarding process when mapping incoming packets to outgoing interfaces, we generalize the FIB to also map packets internal to a node towards the functional entity that will process the packet further. Forwarding inside ANA is illustrated by figure 3.



a -> f1(e)
b -> f2(c)
c -> f3(d)
d -> fwd(e)
e -> eth-if(0)

Figure 3. Forwarding via IDPs.

Having the ability to redirect packets permits much more flexible arrangements than is possible today. By design, the Internet provides a uniform forwarding service that (blindly) delivers packets from the source end-point to the destination end-point. However, many modern applications would benefit from a more flexible communication abstraction with indirection support [16] that decouples the sending hosts from the receiving hosts. For example, indirection is required for emerging applications as IP mobility, proxies (e.g. web caching), and multicast.

In ANA, data is always forwarded to IDPs and never directly to the network entities which are bound to them. This useful decoupling prevents network entities to be involved in and be aware of any (autonomic) re-binding procedure that can take place during active communications.

3.5 Representing Compartments

In summary, a compartment is the generalization of the "network cloud" or *realm* concept. ANA compartments offer access to *information channels*, not end systems, network nodes or software entities. We keep the notion of compartment member to refer to the (now broader) description of an intended communication configuration.

3.5.1 Abstractions vs. structure

Communication configurations like channel topology, set of receivers etc. are instantiated or accessed through a resolution step. Altogether, this definition of compartment refers to the *conceptual view* of an ANA compartment, i.e. it abstracts the "communication service" provided by a compartment. This view can be complemented by a *structural view* that details how a communication configuration is implemented with *functional blocks* which implement the functionalities (e.g., protocols and algorithms) of a compartment.

The two views, i.e. conceptual and structural, are essential. For example, an overlay compartment may only care about the conceptual view of an underlay compartment and uses the ICs without having to see how this is implemented in the underlay compartment. Hence a conceptual view is typically *exported* by a compartment and *imported* by other compartments that use the communication services (ICs) provided by the exporting compartment. Whether a conceptual view is seen as exported or imported depends on the viewpoint of the "observer".

Figure 4 illustrates the concept of views for an example compartment. The compartment box has an upper part, which represents the exported view, while the middle part depicts the structural view, and the lower part shows the imported view.

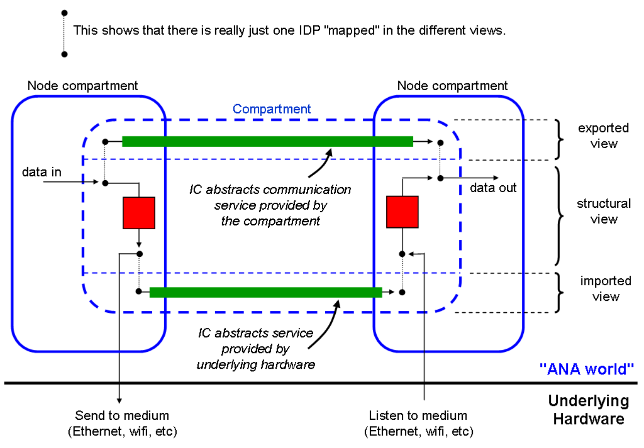

This shows that there is really just one IDP "mapped" in the different views.
Node compartment
Compartment
Node compartment
exported view
data in
IC abstracts communication service provided by the compartment
data out
structural view
imported view
IC abstracts service provided by underlying hardware
"ANA world"
Underlying Hardware
Send to medium (Ethernet, wifi, etc)
Listen to medium (Ethernet, wifi, etc)

Figure 4. Different views of a compartment.

3.5.2 Mapping IDPs into multiple compartments

This split is reminescent of OSI and in fact any layered (net- and software) architecture that offers the possibility to abstract away from the actual implementation of a service.

In Fig. 4 we see that access (data-in) to the information channel is via an information dispatch point (IDP) that exists in two views, but is identical: A client of this compartment will (conceptually) "talk" into the information channel while in reality (structurally) it talks to a functional block (FB) that implements the service, itself referring to some lower layer facilities outside the compartment itself. The lowest IDP on the sender side (labeled "send to medium") is provided by ANA as an access to the outer world implemented in form of drivers and hardware.

3.5.3 The node compartment

On Fig. 4 we also introduce the notion of "node compartments" which are the context in which FBs and IDPs exist on a physical device. In brief, the node compartment offers a view on the available networking resources, is organized like a general compartment, but is special insofar as it really hosts IDPs and FBs, while (network) compartments only refer to these building blocks.

3.6 Examples

Figure 5 shows the data-plane modeling of a bridge between two Ethernet segments, using the primitives introduced above. Two communication channels are spliced together in a bridge node via an IDP: Packets received from one channel are directly forwarded to the other channel based on the FIB's content (not shown). The middle compartment is a "node compartment" that represents the physical device: Its FIB hosts the common IDP that is mapped into the two compartments representing the two Ethernet segments. That is: the two IDPs in the compartments with the dashed line coincide with the IDP in the node compartment, which actually hosts the IDP resource.

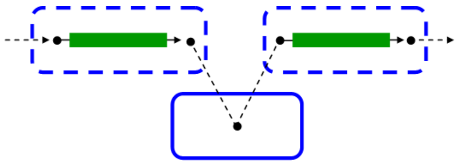


Figure 5. Bridging with IDPs.

A second example illustrated by Figure 6 is the federation of two compartments via a gateway functional block and an overlay that hides the fact that two different compartments are involved. The compartments with the solid lines *(A,B,C)* are ANA nodes, while *c1* and *c2* are compartments representing for example a LAN segment and a point-to-point link. The compartment *c3* is the overlay that defines a new address space independently of the underlying compartments[1]. An application residing on node *A* can ask compartment *c3* to resolve some peer node and will obtain the top left IDP. In this example, the IDP in *c3* coincides with the IDP also mapped inside the LAN segment *c1*. That is, although resolution was carried out in the overlay, the returned IDP is one that is provided by the node compartment *A* and which is bound, inside compartment *c1*, to a channel leading to the intermediate node *B*. In this intermediate node, packets can be transformed in order to be able to ship them over the compartment *c2*. The end point is again an IDP that is mapped in all three compartments *C, c2 and c3*, such that receiving a packet does not incur any overhead.

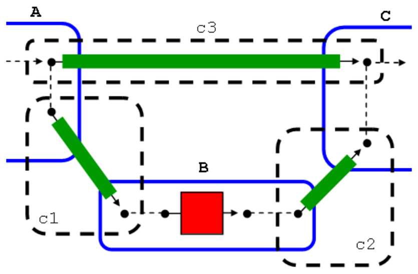


Figure 6. Overlays in ANA.

The primitives described above and their semantics are currently being evaluated and refined. We are in the process of defining the API for each of these first level citizens. In parallel, first decisions have been made on the organization of an "ANA node" such that a minimal version can be implemented in very short time.

4 An Autonomic View on Network Operation

One of the key objectives of ANA is that the architecture must allow – in reaction to changing networking conditions – for the dynamic adaptation and re-organization of the network elements. Only the capacity of the network to be polyfunctional and fully adaptable will justify the label 'autonomic'. In order to fulfill the objective of dynamic (and autonomic) network re-configuration, ANA must envisage multiple alternatives to perform a given network operation. That is by design, the architecture supports the selection and switching of network protocols at run time. It will have to feature autonomic decision routines that control which protocols should be run when and how. This contrasts with the current situation where the network protocols to be used are typically chosen off-line by human

[1]Note that to simplify the figure we only show the (exported) conceptual views of the compartments *c1*,*c2*, and *c3*.

operators based on their knowledge of the topology and the expected performance and behavior of the network.

To fulfill these objectives, ANA will natively include two frameworks which are key to achieve autonomicity: *functional composition* and *monitoring*. Functional composition refers to the ability of dynamically setting up communication "stacks" according to some requirements. In practice and as illustrated by Figure 3, this framework permits that networking modules are assembled on-demand in a flexible and non-disruptive way (thanks to the generic use of IDPs). This feature is essential in order to allow dynamic adaptation of network protocols and integration of future functionalities. The information that will be used to analyse the behavior and operation of network elements, and that may trigger re-configuration of network components is fetched in ANA by a generic monitoring framework that all elements must support. Monitoring capabilities will be built-in in all elements of ANA as information gathering and distribution are key components to enable autonomicity and adaptation.

5 Conclusion and Future Work

In ANA, testbeds and prototypes will be implemented at an early stage, following the Internet tradition that networking software matures through implementation. The feedback obtained from preliminary experimental results will be used to steer and refine the architectural design. Inside the four year program we plan for two such prototyping cycles. The goal of the first prototype is to demonstrate complete self-organization of individual nodes into a network, while the second testbed will focus on the self-federation of networks into a global network.

The ANA project consortium has released in January 2007 the core documents describing the architecture (i.e., the ANA "Blueprint"). The Blueprint not only defines the high-level principles and abstractions of ANA, but also specifies the elements of the core architecture and the various APIs for the compartment membership registration framework, the communications with network compartment entities, and the interfaces of the core entities of ANA. In parallel, a set of documents describes the initial routing and service discovery architectures, the monitoring and resilience frameworks, as well as the testbed architecture.

Based on this set of rough specification documents, our focus has shifted in 2007 to the development of the first ANA prototype. For example, we now focus on the development of the self-association mechanisms allowing nodes to automatically join compartments and on the resolution process used to discover peers and to route data through and across a compartment. Our objective is to be able to demonstrate a first set of autonomic properties by the end of 2007.

6 Acknowledgments

The 4-year (2006-2009) ANA Project (FP6-IST-27489) [1] is funded by the European Commission under the proactive initiative on "Situated and Autonomic Communications" (SAC). Some of the ideas presented in this paper result from work and discussions with other members of the ANA project. The authors of this paper hence do not claim exclusive credit with respect to the content.

References

[1] Autonomic Network Architecture - `http://www.ana-project.org`.

[2] FIND – Future Internet Design (FIND). At `http://www.nsf.gov/pubs/2006/nsf06516/nsf06516.htm`.

[3] Future Internet Research and Experimentation (FIRE) initiative - `http://cordis.europa.eu/ist/fet/comms-fire.htm`.

[4] GENI – Global Environment for Networking Investigations. `http://www.geni.net/`.

[5] T. Anderson, L. Peterson, S. Shenker, and J. Turner. Report of NSF Workshop on Overcoming Barriers to Disruptive Innovation in Networking, January 2005. Available at `http://www.geni.net/documents.php`.

[6] M. Blumenthal and D. Clark. Rethinking the design of the Internet: The end to end arguments vs. the brave new world. *ACM Transactions on Internet Technology*, 1(1):70–109, August 2001.

[7] D. Cheriton and M. Gritter. TRIAD: A New Next-Generation Internet Architecture, July 2000. Stanford Computer Science Technical Report.

[8] D. Clark, R. Braden, A. Falk, and V. Pingali. FARA: Reorganizing the Addressing Architecture. In *Proceedings of ACM SIGCOMM Workshop on Future Directions in Network Architecture*, August 2003. Karlsruhe, Germany.

[9] D. Clark, K. Sollins, J. Wroclawski, and R. Braden. Tussle in Cyberspace: Defining Tomorrow's Internet. In *Proceedings of ACM SIGCOMM 2002*, August 2002. Pittsburgh, USA.

[10] J. Crowcroft, S. Hand, R. Mortier, T. Roscoe, and A. Warfield. Plutarch: an Argument for Network Pluralism. In *Proceedings of ACM SIGCOMM Workshop on Future Directions in Network Architecture*, August 2003. Karlsruhe, Germany.

[11] J. Crowcroft and P. Key. Report from the Clean Slate Network Research post-SIGCOMM 2006 Workshop. *ACM Computer Communication Review*, 37(1):75–78, January 2007.

[12] P. Francis and R. Gummadi. IPNL: A NAT-Extended Internet Architecture. In *Proceedings of ACM SIGCOMM'01*, pages 69–80, August 2001. San Diego, CA, USA.

[13] P. Molinero-Fernandez, N. McKeown, and H. Zhang. Is IP going to take over the world (of communications)? In *Proceedings of First ACM Workshop on Hot Topics in Networks (HotNets-I)*, October 2002. Princeton, NJ, USA.

[14] S. Schmid, L. Eggert, M. Brunner, and J. Quittek. TurfNet: An Architecture for Dynamically Composable Networks. In *Proceedings of 1st IFIP TC6 WG6.6 Workshop on Autonomic Communication (WAC 2004)*, October 2004. Berlin, Germany.

[15] S. Schmid, M. Sifalakis, and D. Hutchison. Towards Autonomic Networks. In *Proceedings of the 3rd Workshop on Autonomic Communication (WAC 2006)*, September 2006. Paris, France.

[16] I. Stoica, D. Atkins, S. Zhuang, S. Shenker, and S. Surana. Internet Indirection Infrastructure. In *Proceedings of ACM SIGCOMM 2002*, April 2002. Pittsburg, USA.

[17] C. Tschudin and R. Gold. Network Pointers. In *Proceedings of First ACM Workshop on Hot Topics in Networks (HotNets-I)*, October 2002. Princeton, NJ, USA.