Decentralized and Autonomous Bootstrapping for IPv6-based Peer-to-Peer Networks

Roland Bless, Oliver P. Waldhorst, Christoph Mayer and Hans Wippel
Institute of Telematics, Universität Karlsruhe (TH), Zirkel 2, D–76128 Karlsruhe, Germany
Email: {bless, waldhorst, mayer, wippel}@tm.uka.de

***Abstract*—Peer-to-peer (P2P) overlay networks are the foundation of a significant number of today's most popular Internet applications, such as distributed file sharing, streaming videos, or Internet telephony. The current IPv4 protocol, however, has significant drawbacks within the P2P context: Firstly, extensive use of Network Address Translation (NAT) mechanisms hinders direct connectivity between arbitrary peers. Secondly, this impedes also discovery of P2P members in order to initially join such a P2P network, commonly referred to as the bootstrapping problem. The first drawback is overcome by IPv6's sufficiently large address space and the lacking need for using IPv6-NATs. An efficient solution for the second problem based on the innovative features of IPv6 is discussed in this paper.**

I. Introduction

Using IPv6 for peer-to-peer (P2P) overlay networks is promising since end-to-end reachability in both directions can be achieved by using IPv6 global unicast addresses. Nowadays IPv4-based P2P networks often struggle with connectivity problems due to the wide use of NAT and the lack of globally reachable public IPv4 addresses. This inherent problem in IPv4 is also reflected in the 'bootstrapping' process: finding P2P members of a particular P2P application in order to join the P2P overlay network. Bootstrapping constitutes often the only centralized task in otherwise decentralized P2P systems. It typically relies on a set of dedicated bootstrap servers that are reachable under well-known public addresses. This aspect turns out to be crucial for robustness of the overall P2P network, e. g., if bootstrap servers become unreachable. A prominent example was the Skype outage due to failure of the login nodes [1]. A fully decentralized solution should try to find P2P members quickly without help of existing infrastructure, thereby increasing the overall robustness of the P2P network.

Current work [2] proposes a two step-approach for decentralized bootstrapping. It shows that a list of observed active peers is useful for reconnecting to a P2P network after short times of disconnection. However, for infrequent/first-time users, and small P2P networks that change quickly it proposes to discover new peers by *random address probing*: a peer tries to find other peers for a particular P2P network by sending probe packets to randomly chosen IP addresses. This approach can work quite well for IPv4 addresses, in particular if the addresses are chosen from a list of dial-up networks [3], [2]. With IPv6 this approach can be extended to a more efficient solution.

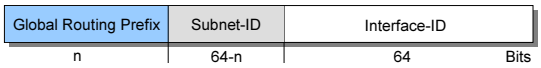

Figure 1. IPv6 Global Unicast Address Format

Figure 1 shows the structure of a global unicast IPv6 address. As depicted, IPv6 addresses contain a 64 bit 'host' part, which is referred to as *Interface Identifier* (IID). There are no restrictions to the structure of such IIDs, except for some reserved special IID values [4]. We realize a more efficient random address probing by utilizing this property of IPv6 IIDs. Specific IIDs for particular P2P networks are derived using a hash function. Globally reachable IPv6 addresses for P2P members are composed of such generated IIDs and their related /64 network prefixes. Consequently, only one such IPv6 address per subnet has to be probed by a bootstrapping P2P node in the proposed scheme.

We assume that a probing P2P client can initially be equipped with a (static) list of active network prefixes by using data from a BGP routing table dump. This is used as reference for the networks to be queried. If the IPv6 routing table had the same size as the current IPv4 table, this list would be between 150 000 and 300 000 global prefixes (currently 1 600 entries only in IPv6 [5]). A standard prefix is usually /48, so the subnet part is 16 bit wide only. This subnet part must be also chosen when specifying an IPv6 address to query. Here we exploit the fact that subnets are usually numbered subsequently starting from 1.

IPv6 provides mechanisms that can be employed for more effective random address probing. These mechanisms include (1) the ability to configure multiple IPv6 addresses per interface, (2) built-in duplicate address detection, and (3) anycast support. We will present an approach that is based on these strengths of IPv6 in the following section.

II. Proposed Bootstrap Mechanism

To allow for decentralized and autonomous bootstrapping with IPv6, we propose to use a host-based IPv6 anycast solution. In the following we present two implementation variants of which one variant does not require any modifications of protocols and routers (basic approach), whereas the other variant requires small extensions (advanced approach).

First, we introduce a set of definitions that we use to detail on the algorithms: s denotes an application unique string that identifies the corresponding P2P network. A hash function

$h(s) \Rightarrow x_i,\ i \in 0 \ldots 2^{64} - 1$ maps s to a 64 bit identifier. If s is a public key as in [6], nodes can even ensure that they are bootstrapping to the correct P2P network. We denote an IPv6 address using a, global routing prefix using pr, subnet id using sid, and IID as x. A set of global routing prefixes is defined as Ω and a single entry $\beta \in \Omega$. Finally, we define a constant c that defines the number of subnets to probe inside a global routing prefix β (c acts as termination condition in the probing scheme).

The bootstrapping procedure consists of the following steps:

- *Address Probing*
  First, the P2P node tries to find a member of the P2P network s (see lower left part in Figure 2). For this, he derives the IID x for the particular P2P network by using $h(s)$.
  1) The node chooses a prefix $\beta_1 \in \Omega$ and composes an address to probe by also choosing a subnet part sid_j and adding the IID x, so $a_t = \beta_1|sid_j|x$.
  2) The node sends a probing packet to a_t, usually a UDP packet to an application specific port.
  3) This process is repeated with different prefixes β_l and up to c different subnet ids sid per prefix, until one or several probing responses are received.
  4) The subsequent initial handshake for joining the P2P network follows according to the application specific protocol.
- *Registration*
  After successful integration into the P2P network the node will register as bootstrap node.
  1) For each of its subnet prefixes $pr_i|sid_i$ each P2P node constructs an address $a_i = pr_i|sid_i|x$.
  2) For each address of the previous step the node either tries to add the address a_i to its corresponding interfaces (basic approach, see upper right part in Figure 2) or register itself as a node belonging to the corresponding anycast group (advanced approach, see lower right part in Figure 2). Adding an address in the basic approach may fail if another node in the subnet already configured as bootstrap node.
  3) In case of successful registration the P2P application will start listening on a specific UDP port for bootstrap probing packets.

In the following steps the basic and advanced algorithms are detailed on separately.

A. Basic Approach

The P2P node tries to configure the additional IPv6 addresses for the IID x on his interface(s). Existing and already configured prefixes $pr_i|sid_i$ are used for composition of the address $a_i = pr_i|sid_i|x$. Due to Duplicate Address Detection mechanisms in Neighbor Discovery for IPv6 [7] nodes fail to configure an address if a node in the subnet already possesses the same address. Therefore, the one resulting node per subnet is the bootstrapping representative for this subnet and serves thus as rendezvous point for the particular P2P network.

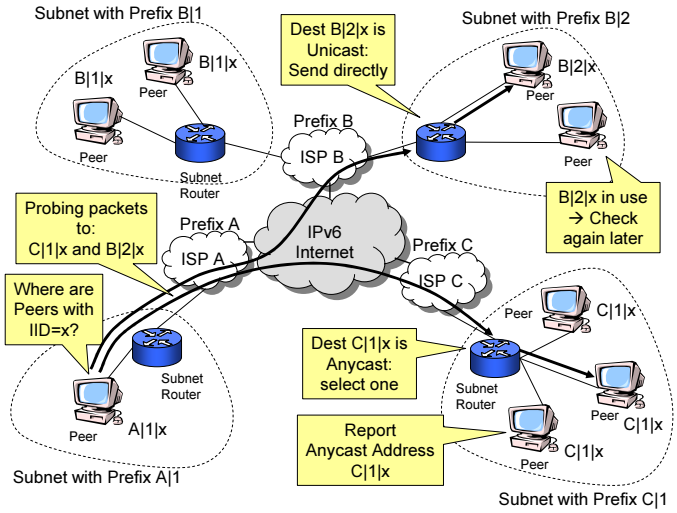

Figure 2. P2P Bootstrap Scenario

B. Advanced Approach

In the advanced approach each P2P node will join a local anycast group at its access router(s) using a protocol similar to Multicast Listener Discovery (MLD) [8] for registering membership to multicast groups. This anycast group with IPv6 address a is representing the specific P2P application. The main difference is that the router manages the anycast group membership instead of multicast group membership. A packet destined to such an anycast address, i. e., one with IID x, is not replicated for each group member, but forwarded to a randomly chosen member of the anycast group. A corresponding MLD extension proposal was already described in [9]. Using this mechanism the local subnet router is aware of the nodes that belong to the specific anycast group in the specific on-link subnet. P2P nodes themselves do not need to configure additional IPv6 addresses on their interfaces.

C. Bootstrapping of a P2P Node

Bootstrapping P2P nodes into the overlay network is identical from the node's perspective regardless of whether the basic approach or advanced approach is used (actually, the presented approaches are compatible). The difference appears only in the router behavior.

Router behavior after reception of a probe packet differs—but is still compatible—in the basic and advanced approach. Therefore we will detail on the behavior separately.

1) Basic Approach – Router Behavior: In the basic approach the router is not aware of an anycast group for the specific IPv6 address a and forwards the probe packet as a packet destined to a unicast address. If a node with configured address a exists in the subnet, it will reply to the bootstrap request.

2) Advanced Approach – Router Behavior: When the subnet router receives a probe packet it detects that this is actually an anycast address during look up in its destination cache. It then chooses a random anycast group member as the next hop

on-link neighbor and forwards the packet. The random node that receives the packet responds to the probe packet. If no node is registered for the anycast group, no response will be sent to the bootstrapping node from the current subnet.

D. Comparison

First of all, both the basic approach and advanced approach are transparent from the bootstrapping node's point of view as detailed in II-C. Furthermore, both approaches can be deployed in parallel as (1) the probe packet delivery inside a network can differ from network to network, and (2) anycast addresses are not syntactically distinguishable from unicast addresses.

Both approaches have pros and cons that we will now shortly explain. This is a strength of our proposal as (1) the proposed mechanism can be deployed iteratively, (2) each network can decide which approach fits best, and (3) global interoperability of the approaches is possible.

a) Basic Approach: An advantage of this approach is that it does not require any modified components or protocols. Disadvantages of the approach are that the application must have means to configure additional IP addresses (i. e. super-user privileges) and that probing load in a subnet is concentrated at a single node, probably making it an attractive DDoS target. Furthermore, in case of failure or leaving the P2P network, another node must register at the same address. Most implementations will, however, not explicitly retry to configure a duplicate address, therefore this must be done periodically by the application itself. There are also Denial-of-Service (DoS) attacks possible if a malicious node configures the address but refuses to properly answer a probing request for a subnet. However, since requests are also sent to other networks in parallel, the process is merely slowed down.

b) Advanced Approach: The advanced approach has the advantage of probing load being evenly distributed among all P2P nodes of the particular application in a subnet and therefore is more robust against failure and attacks. The disadvantage is that it requires slight modifications of the MLD protocol and anycast functionality inside the last hop access router. We believe though, that the protocol and implementation modifications are quite moderate. Furthermore, we note that MLD should be modified or extended in a way so that MLD snooping switches should not interpret and track the anycast related group membership since it is not necessary for them.

III. Conclusion

Decentralized bootstrapping of P2P nodes is an important problem that has only been insufficiently solved with IPv4 so far. We presented a promising solution that builds on the features and protocols of the IPv6 suite. The solution provides (1) decentralized and autonomous bootstrapping, (2) can be deployed globally and iteratively using two compatible approaches, (3) makes P2P networks more robust, and (4) can be easily implemented using IPv6 due to larger addresses and the given end-to-end reachability. We believe that our scheme can push deployment of IPv6 as (1) P2P applications have gathered large importance in the last years, and (2) it provides concrete benefits over the IPv4 solution.

References

[1] V. Arak, "What happened on August 16," (WWW-Publication), 2007, http://heartbeat.skype.com/2007/08/what_happened_on_august_16.html.

[2] J. Dinger and O. Waldhorst, "Decentralized Bootstrapping of P2P Systems: A Practical View," in *Proc. 8th IFIP TC6 Int. Conf. on Networking*, May 2009, pp. 703–715.

[3] M. Conrad and H.-J. Hof, "A Generic, Self-Organizing, and Distributed Bootstrap Service for Peer-to-Peer Networks," in *Proc. 2nd Int. Workshop on Self-Organizing Systems (IWSOS 2007)*, Sep. 2007, pp. 59–72.

[4] S. Krishnan, "Reserved IPv6 Interface Identifiers," RFC 5453 (Proposed Standard), Feb. 2009. [Online]. Available: http://www.ietf.org/rfc/rfc5453.txt

[5] G. Huston, "BGP Reports," http://bgp.potaroo.net, Apr. 2009.

[6] T. Aura, "Cryptographically Generated Addresses (CGA)," RFC 3972 (Proposed Standard), Mar. 2005, updated by RFCs 4581, 4982. [Online]. Available: http://www.ietf.org/rfc/rfc3972.txt

[7] T. Narten, E. Nordmark, and W. Simpson, "Neighbor Discovery for IP Version 6 (IPv6)," RFC 2461 (Draft Standard), Dec. 1998, obsoleted by RFC 4861, updated by RFC 4311. [Online]. Available: http://www.ietf.org/rfc/rfc2461.txt

[8] R. Vida and L. Costa, "Multicast Listener Discovery Version 2 (MLDv2) for IPv6," RFC 3810 (Proposed Standard), Jun. 2004, updated by RFC 4604. [Online]. Available: http://www.ietf.org/rfc/rfc3810.txt

[9] B. Haberman and D. Thaler, "Host-based Anycast using MLD," draft-haberman-ipngwg-host-anycast-01.txt, May 2002.