On Name-based Group Communication: Challenges, Concepts, and Transparent Deployment

Thomas C. Schmidt

HAW Hamburg, Dept. Informatik
Berliner Tor 7
D–20099 Hamburg, Germany
Email: t.schmidt@ieee.org

Matthias Wählisch

Freie Universität Berlin, Inst. für Informatik
Takustraße 9
D–14195 Berlin, Germany
Email: waehlisch@ieee.org

Dominik Charousset

HAW Hamburg, Dept. Informatik
Berliner Tor 7
D–20099 Hamburg, Germany
Email: dominik.charousset@haw-hamburg.de

Sebastian Meiling

HAW Hamburg, Dept. Informatik
Berliner Tor 7
D–20099 Hamburg, Germany
Email: sebastian.meiling@haw-hamburg.de

Abstract

Human-centric naming will largely facilitate access and deployment of network services in a future Internet. Information-centric networking (ICN) introduces such vision of a name-oriented, secure, globally available publish-subscribe infrastructure. Current approaches concentrate on unicast-like pull mechanisms and thereby fall short of naming and automatically updating content at groups of receivers. In this paper, we adopt the information-centric paradigm, but argue that an inclusion of multicast will grant additional benefits to the network layer. Our contribution bridges the gap between requesting content by name and applying requests to a scalable distribution infrastructure in a many-to-many communication model. We introduce a group-oriented naming concept that integrates the various available group schemes, simplifies rendezvous processes, and introduces new use cases. We present an open-source prototype of this name-oriented multicast access implemented in the H∀Mcast middleware.

Keywords: Real-time Multimedia Distribution, Content Centric Networks, Multicast, Naming, Addressing, Security, Routing

1. Introduction

The search for future directions in Internet development has drawn significant attention to name-based communication concepts that are built upon the publish/subscribe paradigm. Inspired by the Web use case and widely deployed content distribution networks, proposals for *Information Centric Networking* (ICN) abandon the current Internet model of connecting end nodes. Instead, consumers shall retrieve content by name directly from a network that provides storage, caching, content-based rendezvous, and searching at times.

Several proposals have been presented in recent years [1], among them TRIAD [2], DONA [3], NDN [4, 5], PSIRP/LIPSIN [6], NetInf [7], and CURLING [8]. The schemes differ in naming/addressing, routing/rendezvous, security/authentication, forwarding/caching, and minor design choices, but jointly consider multicast at most as a property of data paths. Only very recently, COPSS [9] has brought up the discussion of relevance for a group distribution model in ICN. COPSS targets at seamlessly extending NDN by a specific multicast service based on Protocol-independent Multicast (PIM)-type rendezvous points, thereby omitting a broader discussion about naming, addressing, routing, and security. The present paper attempts to fill this gap between requesting content by name and applying requests to a (heterogeneous) distribution infrastructure in a scalable and secure way.

Our core contributions consist of (i) a generic naming scheme that inherently supports the various forms of group communication and allows for stateless mapping to distribution systems, (ii) a concept of securing content *and* the distribution infrastructure based on self-certifying credentials, and (iii) an open-source prototype that implements the proposed concepts.

Multicast is the traditional approach to publish/subscribe on the Internet layer [10]. In contrast to ICN that relies on in-network state for each content item, multicast enforces highly efficient, scalable data forwarding for simultaneously operating senders and receivers. ICN also facilitates the consumption of a single (cached) data copy by multiple receivers, but requires content subscribers to act nearly synchronized in order to profit from tree distribution. Occasionally, traditional multicast is proposed to improve the ICN efficiency, e.g., in MultiCache [11]. In addition to ICN concepts, the multicast model features the following contributions.

- Data is pushed to receivers, supporting multiple transport streams in parallel and eliminating the need to ask for content changes.

- Data distribution supports immediate in-network forwarding and is suitable for efficient, scalable real-time streaming in particular. This mainly covers the use case of real-time data dissemination without storage or caching requirements.

- The multicast model contributes many-to-many communication, which is valuable whenever information is created at distributed origins. Multi-source communication using a single name faces strong conceptual difficulties in unicast-oriented ICNs.

- Multicast group communication enables rendezvous processes that interconnect content subscribers with the publishers. Even though publishers and subscribers remain decoupled and unknown to each other, the multicast routing establishes connectivity among group members.

In this paper, we resume arguments for the case of multicast communication in name-oriented pub/sub networks. Led by the observation that a majority of today's applications, such as Twitter, facebook, chat, gaming, conferencing, and IPTV, rely on group communication,[1] we start with a discussion of various multicast aspects that are missing in the current ICN models. As for our core contribution, we continue with a thorough conceptual examination of naming, addressing, grouping and security for multicast communication in information-centric networking. We present an integrative proposal for a common multicast access scheme that donates particular attention to the programming side, as well as an overview of our system-centric implementation H∀Mcast . Similar to the work of Tyson et al. [12], H∀Mcast is built on a middleware abstraction that allows for flexible instantiations of various network services.

Our naming concepts that include hierarchies and aggregation as well as named instantiations comply to the principles of "push enabled dissemination", "decoupling of publishers and subscribers", and "support hierarchies and context in naming content" as postulated in [9]. In addition, we are able to express further group concepts such as selective broadcast and selective data origins, and keep content from node names clearly separated in our syntax. The latter is of particular importance for building conceptually clear, type-safe programming models on top of our networking concept.

Multiparty content distribution requires security measures in two key aspects. At first, content must be protected from forgery and spoofing by authentication. Second, the distribution infrastructure must be enforced to resist resource exhaustion misuse in amplifying attacks. Based on self-certifying identifiers, our naming scheme devises an authentication concept that applies to both, network access and content receivers. It inherently provides protection against spoofing and infrastructure attacks.

The remainder of this paper is structured as follows: In Section 2 we discuss the problem space and identify the unique contributions the multicast model can add to the ICN paradigm. Section 3 is dedicated to naming and its implications for group forming, routing, and security. An overview of our deployment-friendly

[1]Group communication today is - similar to content distribution - almost always implemented on the application layer.

implementation is presented in Section 4. Finally, Section 5 concludes on the achievements, and previews on the upcoming steps in this active domain.

2. Why do We Need Multicast in Name-oriented Publish/Subscribe?

ICN-style networks introduce the vision of a secure, efficient, globally available publish/subscribe system. In the current Internet, the publish/subscribe paradigm has been implemented by multicast, even though it is not globally deployed, yet, and only present at the intra-domain level. This paradigmatic coincidence raises two questions: What are the differences between multicast and information-centric networks in detail? Why should we care about multicast on the ICN layer?

2.1. Recap the Multicast Model

The multicast communication model implements the following functions:

1. Explicit data subscription by receivers.
2. Implicit, delocalized content publication.
3. Routing procedures that deliver rendezvous and replication per group.
4. Concepts for groups with open as well as closed access.

A multicast listener must join a multicast group to receive the corresponding data. Based on the subscription and an in-network rendezvous mechanism that finds the sources of data, distribution paths will be established dynamically. Thus, the network infrastructure delivers multicast data only to those nodes that requested the data explicitly. This is one face of a publish/subscribe system [13], and in contrast to current unicast communication, in which a sender may transmit data to parties on the network layer that never agreed on the communication.

Conceptually, the multicast group model splits up in one-to-many, i.e., Source Specific Multicast (SSM), and many-to-many multicast, i.e., Any Source Multicast (ASM). The latter is open per se, while the first allows for closed groups. Securing multicast group management has been discussed since a while [14], even for the mobile regime [15], where self-certifying group identifiers have been proposed to prevent a distribution of content by illegitimate sources [16].

Any source multicast inherently provides functions to address content by an identifier that is fully decoupled from the actual storing host [10]. On the application side, there is no direct binding between content ID and host address of the source. Content can be provided by one or multiple arbitrary sources, while receivers need not resolve the location of content.

In both perspectives—receiver-driven content access, and content naming instead of addressing—multicast and content-centric networking follow the same paradigm. However, there is an important difference in how content is duplicated within the network.

Content replication in multicast is initiated by receivers. Multicast on the data link, network, and transport layer is restricted to unbuffered push. Intermediate nodes do not store content in advance or cache the content for subsequent receivers. In contrast, in ICN the network itself takes decision on data replication and in-network storage. Early previous work in the context of web caching considers multicast conjointly with consumer-oriented content placement [17]. In the extreme case, a web server would continuously transmit content to a multicast group that is then joined by WWW clients. However, this is only efficient for continuous content transmission and may be generalized by push-caching schemes.

2.2. Content Replication

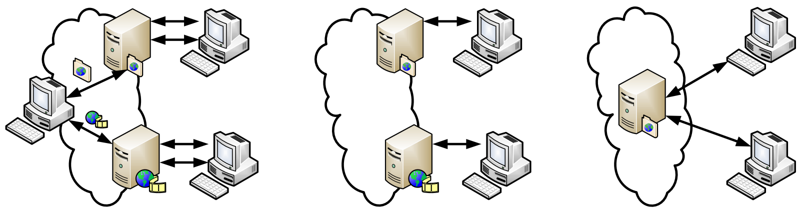
Content replication describes the process of moving multiple copies of the same data to different network locations. The content may be delivered directly to the consumers or stored at pre-located content repositories. Replication provides a consistent view on the distributed content. In this sense it differs from caching, which applies local replacement strategies and thus may lead to inconsistencies. Network layer multicast implements an efficient way to replicate content towards interested peers.

Network-based content replication is motivated by two objectives: the reduction of network cost (i.e., eliminate redundant data transmission) and the improvement of the end user experience (i.e., decrease delay). There are three generic application scenarios, in which this is helpful (see Figure 1). In the first case, a single node accesses the content multiple times. This may occur due to limited local buffers at lightweight mobile nodes, for example. To achieve both goals of content replication, the data should be stored as close as possible to the receiver and kept there for as long as possible to avoid unnecessary retransmissions. This is the traditional use case of ICN, and caching strategies have been in recent focus of the community [18, 19, 20].

The second case describes a host that downloads content one-time. The delivery should be as fast as possible. In contrast to the previous scenario, the content can be deleted immediately after the access to release memory resources, but must be available in advance to diminish network delay.

If multiple nodes request the same content in parallel (i.e., a case for multicast), the content needs replication at several branch points. From a network perspective of efficiency, such replication should not be simply performed close to content consumer, but focus on a group perspective. From the alternative storage point of view, content should be placed in the vicinity of the receivers.

In all three scenarios, an optimal replication strategy depends on the use case. To prevent repeated transmissions of the same content, a distributed storage is important for long-living data. The subset of participating peers determines the distances between content repository and the consumers. An advanced knowledge about the type of content (multicast vs. unicast) may thus help to improve the underlying distribution mechanism. As already noted in [17], a subscription-oriented model eases the identification of content that is of interest for receivers and so improves pre-fetching of the data. Information-centric networking may benefit from an explicit consideration of multicast, as this enables to adjust replication strategies, reduces complexity, and saves costs.



(a) Repeated access: On-demand replication close to the consumer, data caching for further access.

(b) One-time access: Storage in advance close to the consumer, but data can be removed after access.

(c) Synchronous multi-party access: Storing of content in the vicinity of a group of consumers.

Figure 1: Different content replication scenarios

2.3. Synchronous Content Access

Audio/video conferencing, IPTV and multiplayer online games are prevalent application examples of the third use case of the previous section. They jointly follow an application semantic of *synchronously* accessing content streams in groups, while disregarding any historic data. Multicast naturally serves this need by transmitting the same (sub-)piece of information to all group members at the same time using a single identifier for subscription. The pub/sub paradigm of ICN grants sufficient flexibility to allow for the identification of any single piece of continuous content, but requires polling of each data packet [21] to assemble a desired stream. Moreover, there is no obvious, simple solution to name and pull data fragments in a synchronized fashion.

Zhu et al. [22] have discussed a path to solving this issue for the example of conferencing in NDN. A new party unaware of the available streams and current stream pointers (i.e., chunk addresses) issues interests to learn about the different states of the conference. Once acquired, corresponding stream pointers are then pipelined into an interest request chain to proactively pull whatever may be sent to the conference. Iterated update requests are used to adapt to group changes and de-synchronisation. This complex approach to synchronous communication not only introduces significant overhead without assuring continued synchrony, but leaves important operations undefined. First, ICN naming cannot inherently express synchronous streams. Unless for the weak concept of naming conventions, the ICN layer cannot identify the need for adjusting chunk pointers, nor for building a request chain which pushes such network intrinsics to the application. Second, the replicative routing system has no notion of synchronous data flows, but is eventually burdened with data-driven event trading and buffering at large scale [23]. Third, multisource group communication requires a mapping to individual stream request. In contrast, multicast describes a distribution mechanism that is closely bound to the specific content and group type and will forward synchronous flows without data-driven control overhead or caching.

2.4. Application Programming

The programming of multicast applications follows a different perspective than unicast-type requests. In multicast, a programmer opens a network interface, subscribes to content, and awaits data for the time of interest. An application program thus decouples the time of data reception from the instant of its readiness, but attempts to synchronize to transmission. In the case of content that remains unavailable during the subscription period, the application programmer does not experience an error, neither sees the distribution system at fail, nor the application layer. In contrast, when a chunk of file is accessed in unicast mode, the requesting end point experiences an error, whenever the content is unavailable. The networking layer triggers an unreachable message for a connection that cannot be established, or the application layer typically responds with a failure message (e.g., HTTP 404).

Content-centric networks may or may not operate similarly to multicast and omit or provide an explicit error signaling on the distribution layer. Only this corresponding function can then be reflected at the content programming interface. Consequently, an application programmer needs to handle either one of the cases on his own (e.g., by timeouts). However, this imposes complexity to the implementation side. The support of both mechanisms, pull-oriented unicast and push-awaiting multicast, is indeed desirable.

3. Naming and Addressing in Multicast

Traditionally, naming and addressing in multicast-type group communication follow technology. Group addresses in IP that serve the ASM model have never successfully entered DNS, while no naming standard supports source-specific (S, G)-channels[2]. Similarly, overlay and application-layer schemes have been built on diverse hashing or naming conventions that remain bound to a specific user access to routing or application processing. At present, *programmers* need to decide on deployment technologies by selecting types for names or addresses.

Multicast today does not involve a clear concept of addressing. On the Internet layer, 'multicast addresses' are de-localized parameters for routing states, unless unicast route information is included for sources [24] or rendezvous [25]. A similar picture is visible for application layer multicast, where rendezvous [26], instantiation [27] or reflector services are the only use of location semantics in multicast group names. Following such observations, John Day coined "Multicast addresses is a set of distributed application names" [28, p. 329]. Up until now, traditional networking has failed to deliver a uniformly applicable naming and addressing concept for groups, as they do not represent routing endpoints. The newly aligned perspective of a general publish-subscribe paradigm in networking may encourage to undertake this approach anew.

3.1. Design Aspects for Multicast Names

Starting from the plain objective of naming groups, one could be tempted to opt for simple string identifiers. However, the design of a naming scheme for multicast groups bears a number of specific aspects that go beyond flat strings or unicast-type content subscription. We discuss core aspects in the following.

[2]In source-specific multicast, a single group G decomposes into per-source channels—one (S, G) channel per source S

3.1.1. Coverage of Group Semantics

A uniform naming scheme for multicast should cover the large variety of prevalent group semantics. In particular, an application programmer should not be forced into selecting an interface or data type that predefines the kind of group management or distribution technique. Multicast names, instead, need to provide the expressiveness of all multicast flavours within a single (meta-) data type. Naming, for example, should equally cover traditional IP-layer identifiers like `[224.1.2.3]` and application-specific group names like `#peanuts` without formal distinction. Application code including user interfaces may thus remain transparent with respect to the distribution system, while its corresponding intelligence can be established at the end system or network level.

3.1.2. Namespace Support

The heterogeneity of multicast semantic, but also diverse aspects of routing and transport request for a variable interpretation and processing of group names. For example, the routing system may need to contact a mapping service in some instances, in others multicast data may be accessible via distributed content replication servers (e.g., a CDN) that are expressed as a set of instantiations. A name may further hint to an address mapping in a default technology, or indicate a cryptographic algorithm.

In all these cases, the network, the end system, or the application are required to identify and process the encodings and operate accordingly. To allow for an unambiguous interpretation and a type-safe implementation, indicators of namespaces are of versatile use. Examples of namespaces could be application-specific (e.g., `sip`), technology-centric (e.g., `mcast-ip`), or algorithmically driven (e.g., `sha-2`).

3.1.3. Instantiation

The model of source-specific multicast restricts subscriptions to single-source (S, G) states for the sake of a simplified routing. It resolves the rendezvous problem of ASM in the special case of a single publisher. SSM-type service instantiation should be supported by a uniform naming scheme, but in addition may be extended to cover more general cases.

A group communication system can be instantiated not only by a single source, but by a set of originators that either act (a) as content replicators following an anycast semantic, or (b) as a source-group representing an explicitly named or implicit many-source semantic, or (c) as a remote distribution system (e.g., an overlay) that discloses multicast content on some membership contact.

A corresponding syntax, e.g., of the form `<group>@ <instantiation>`, must comply with this rich set of semantics. The clause `<instantiation>` can yield this expressiveness (i) as an indirection by pointing to some mapping service or rendezvous node, or (ii) by referring to a bootstrap point for contacting a remote overlay, or (iii) by explicitly enumerating a set (e.g., $\{inst_1, ..., inst_n\}$), or (iv) by implicitly naming a set in the form of a statistical filter (e.g., a Bloom filter [29]).

While an instantiation is part of the logical identifier of a multicast group (e.g., `news@cnn.com` and `news@bbc.co.uk` name two different groups), the proposed syntax clearly distinguishes namespaces and semantics. The `<group>` clause names content without reference to a network endpoint, whereas `<instantiation>` refers to a (group of) publishing node(s). Much like in the IP/SSM model, naming (sets of) instantiations can guide the routing layer, an end system, or an application to steer pub/sub contact messages. We note that false positives induced by our use of Bloom filters [30] can lead to unwanted contact paths, which some distribution technologies may mitigate by negotiating pub/sub with the instantiation nodes in a two way handshake.

3.1.4. Hierarchy and Aggregation

Hierarchical naming introduces aggregation, which bears an inherent concept of grouping. The corresponding expressiveness of names gives rise to a number of group applications.

A selective broadcast may for instance become accessible by 'wildcarding': Suppose there are news channels `politics@cnn.com`, `economics@cnn.com`, etc.

Simultaneous publishing to all (possibly unknown) channels may be enabled via `*@cnn.com`.

For a subscriber-centric example, consider a layered video stream `blockbuster` available at different qualities $\texttt{Q}_i$, each of which consist of the `base` layer plus the sum of $\texttt{EL}_j, j \leq i$ enhancement layers. Each individual layer may then be accessible by a name $\texttt{EL}_j\texttt{.Q}_i\texttt{.blockbuster,}\ j \leq i$, while a specific quality aggregates the corresponding layers to $\texttt{Q}_i\texttt{.blockbuster}$, and the full-size movie may be just called `blockbuster`.

It may be useful to aggregate instances of publications, as well. Multiple news channels, for example, may be available from `news@cnn.com`, `news@bbc.co.uk,...`. A subscriber may wish to select multiple channels jointly expressed in a set of sources, or request for all news channels using the group aggregator `news`. While the latter step resembles the transition from SSM to ASM, it is worth noting that a growing number of sources (aggregated in a set or in a Bloom filter) steadily reduces the specificity of the instantiation and thereby implies a smooth transition from the SSM to the ASM multicast model.

Jointly operating on the identifiers for groups and instantiations, this name-aggregation concept provides rich expressiveness with heterogeneous deployment options. In particular, it enhances scalability when dealing with group names, as corresponding states allow for aggregation. We should emphasize that it transparently abstracts from a particular service deployment as a multi-source or single-source multicast communication.

3.1.5. Canonical Support for Stateless Mapping

Many different distribution technologies for multicast are deployed today, and heterogeneity may be expected to persist in future content centric networks. Some technologies use group identifiers of a specific syntax such as IP multicast addresses, overlay hash values, or SIP group conference URIs. A creator of a group may want to express the desire of using a dedicated group ID in a deployment (s)he prefers. The multicast naming scheme should provide a corresponding expressiveness that enables a stateless canonical mapping of group names to technological identifiers.

For example, `mcast-ip://224.1.2.3:5000` indicates the correspondence to `224.1.2.3` in the default IPv4 multicast address space at port `5000`. This default mapping is bound to a technology and may not always be applicable, e.g., in the case of address collisions.

3.2. Security in Multicast Naming

Multicast content distribution requires measures to ensure the four common components of network security: integrity, confidentiality, provenance, and availability [31]. Unlike in unicast, though, group communication imposes the following additional constraints.

1. Multicast publishers and subscribers are decoupled and commonly allow only for unidirectional signaling. This conflicts with cryptographic handshaking as frequently used to create confidentiality or authenticity.
2. Multiple sources may contribute to the content of a single group, making it difficult to prove original and legitimate publishers (provenance).
3. Streaming is a common multicast application. Rather than plain content hashes, the use of stream ciphers is required to ensure integrity.
4. The network infrastructure assists multicast distribution (even) more strongly than unicast-based content centric networks. Data replication services at a possibly large scale raise well-known threats to the infrastructure and the end systems. Admission control and infrastructure protection are required to ensure availability.

Multicast names are commonly created by sources (publishers) that somehow have gained admission to inject content streams into the network. It is thus reasonable to use the source identification as a trust anchor [16] when generating self-certifying names [32] of multicast content.

In detail, the creator (controller) of a group that has generated its cryptographic ID from a public-private key pair $(\mathcal{K}_{pub}, \mathcal{K}_{sec})$, will use $\mathcal{K}_{pub}$ to configure a group name G equally as a cryptographic identity. Conflicts within the node

ID space can be avoided by adding a counter. In signing content using $\mathcal{K}_{sec}$ and attaching $\mathcal{K}_{pub}$, the group controller will provide cryptographically strong proof of ownership to any receiver of the packet. Each intermediate router (or receiver) can verify the source-group-content relationship after extracting $\mathcal{K}_{pub}$ and validating the signature. Assuming that access permission has been granted at the network edge, multicast replication services can be consequently bound to an autonomous packet authentication without feedback loop.

Multiple multicast senders contributing to the same multicast group require admission by the group controller. This admission authority has created the cryptographic group name. Before an additional multicast source S injects data, it requests a certificate. The group controller authenticates the sender and—according to an application policy—issues the certificate, which includes S, the peer membership of G and an optional lifetime. The certificate is signed with the private key corresponding to the creation of G. A multicast source that wants to transmit data attaches this certificate and signs packets with its own private key. An intermediate router verifies whether the group certificate is valid and the group address G has been generated from the public group key. Additionally, the router authenticates the cryptographic identity of the source according to the certificate and the peer identifier as described in the single-source case.

Multicast content streams require stream cyphers to be linked with the cryptographic identity, the details of which are beyond the scope of this article, see [14] for a general overview.

3.3. A Naming Scheme for Multicast

Following our previous design discussion, we now summarize an URI-based naming scheme (cf., [33]).

```
scheme "://" group "@" instantiation ":"
             transport-ID "/" sec-credentials
```

The *scheme* refers to the specification of the assigned identifier (e.g., mcast-ip, sip, ...), *group* denotes the group ID, *instantiation* identifies the entity that generates the instance of the group, *transport-ID* holds optional transport-specific interface to the application, and *sec–credentials* add optional cryptographic identities, see [34] for a corresponding approach. Valid group IDs will be mcast-ip://224.0.1.1:4000 and sip://hypnotic-talks@psychic.org, for example.

The proposed syntax of the group name provides a consistent term for ASM as well as SSM-type groups on the application layer. For example, `mcast-ip://224.10.20.30@1.2.3.4/groupkey` describes an SSM group name, where `mcast-ip://224.10.20.30/groupkey` denotes a source–independent multicast group.

The syntax also allows for flexible namespace support. The group name is defined by the multicast source. A multicast source is enabled to indicate a preferred forwarding scheme using a namespace that corresponds to a distribution technology.

Along this line, a multicast application developer opens a URI–aware multicast socket without predefining the distribution technology. In the case of a receiver subscription, for example:

```
ms=createMSocket()
ms.join(URI(
  "mcast-ip://224.10.20.30@1.2.3.4/groupkey"
)).
```

The socket can be used for another multicast group, as well:

```
ms.join(URI(
  "sip://hypnotic-talks@psychic.org"
)).
```

4. Implementing Name-based Multicast

In this section, we report on our system middleware for name-based multicast and the evaluation of its performance. This is an ongoing open-source development, but provides a ready-to-use proof-of-concept at `http://hamcast.realmv6.org/developers`. It allows for rapid prototyping and real-world multicast applications.

4.1. Design Requirements

An implementation of name-based multicast should comply with the following requirements to allow for deployment in the current and a future information-centric Internet.

Interoperability ICN is an ongoing research field and multiple ICN solutions are around. In the current state, it is neither clear whether multiple ICN solutions will coexist nor which solution(s) will succeed. A group communication solution, thus, should allow for pluggable technologies underneath to distribute and receive multicast data. Application programming should be decoupled from the current and future technology deployment.

Efficiency ICN-enabled devices may range from data center servers up to mobiles. Multicast support at end devices should reflect this. An implementation should be lightweight and suitable for heterogeneous platforms in terms of available hardware resources (i.e., CPU and memory). High performance with small footprint is required.

Universality Implementing group applications is not restricted to specific programming languages. For a fast deployment, the multicast API should be available for common languages and the underlying architecture should be easily extensible. This requires abstraction of the core functionalities from specific, language-dependent libraries.

Overall we argue that these requirements can be achieved with a modular group communication stack at the system level.

4.2. The H∀Mcast *Prototype*

To allow for generic name-based group communication support, we designed and implemented the H∀Mcast stack. H∀Mcast focuses on the integration of

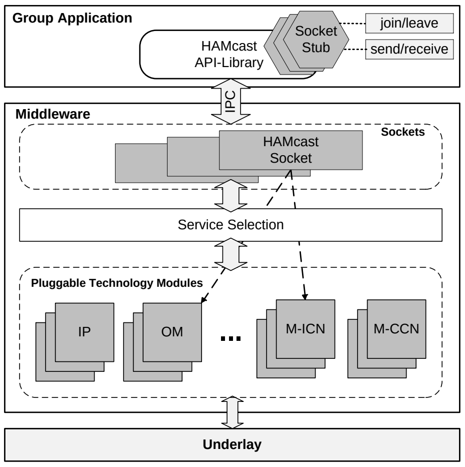
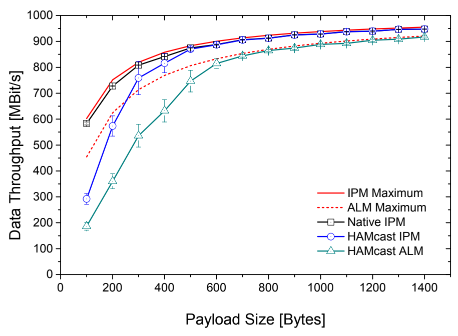


Figure 2: The H∀Mcast group communication stack

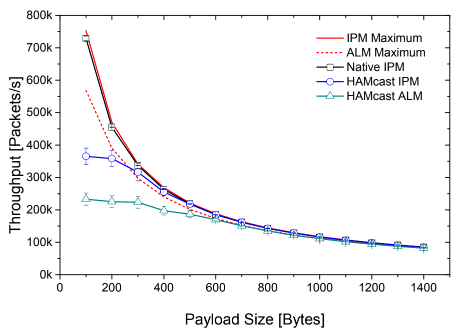
pluralism in network service deployment. It dynamically selects distribution technologies provided at the current environment and hides technology-specific treatment at the socket level. The H∀Mcast system architecture aims for ease of deployment, availability, and efficiency. Our approach is evolutionary and rather extends the network architecture than making changes to existing protocols.

The building blocks of the H∀Mcast stack are visualized in Figure 2. A middleware layer abstracts between applications and various transport technologies, offering a globally available multicast service via the common multicast API [33]. The application programmer and the end user operate on transparent, content-centric identifiers (cf., Section 3), which will be mapped to technology specific addresses or names. The middleware has a plug-in architecture for multicast technologies. In our prototype of name-based multicast routing, we support IPv4/v6 and overlay multicast (OM) as underlying transport. It is worth noting that currently no multicast implementation for ICN (M-ICN, M-CCN) is available. However, extending the stack by upcoming solutions is easy (and transparent for the group application) as only a new technology module needs implementation.

The middleware is implemented in C++ and runs as a system process. Sup-



(a) Data Throughput at Sender



(b) Packet Throughput at Receiver

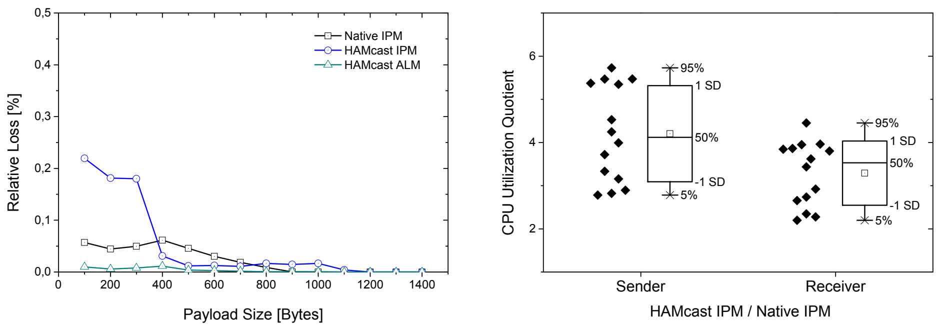
Figure 3: Communication performance of the H∀Mcast middleware versus native IP at 1 Gbit/s link

port for different programming languages on the application site is realized by dedicated libraries that communicate via IPC with the middleware. This is the only part that has to be implemented for different programming languages, which reduces complexity for extensions. We currently provide libraries for Java and C++.

4.3. Evaluation

We evaluate our software prototype from a system-centric perspective to demonstrate its feasibility for real-world communication requirements. We compare the performance of our stack in both use cases - H∀Mcast IP and application layer multicast (ALM) - with the native IP stack of the hosting node. The system performance is analyzed with respect to the metrics (i) data throughput, (ii) packet loss, (iii) scalability with the number of simultaneous groups, and (iv) CPU load. A more general measurement of hybrid multicast performance in real-world deployments goes beyond the scope of this article. We refer the reader to [35] for a thorough evaluation of the distribution system that includes one-way delay measurements on a global scale.

The set-up for our experiments consists of a local network connecting nodes at homogeneous 1 Gbit/s links, with test machines chosen from commodity Intel hardware with QuadCore-CPUs at 2.93 GHz, 8 GB RAM and 1 GE interface. A



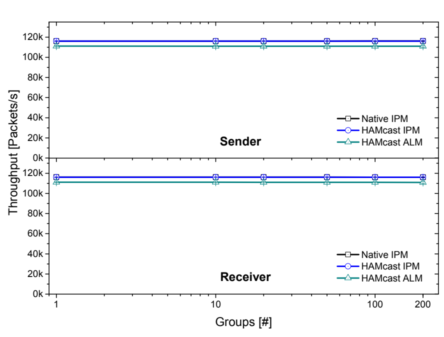
(a) Packet Loss at Receiver (b) CPU Efficiency

Figure 4: H∀Mcast stack performance: Packet loss and CPU efficiency at 1 Gbit/s link

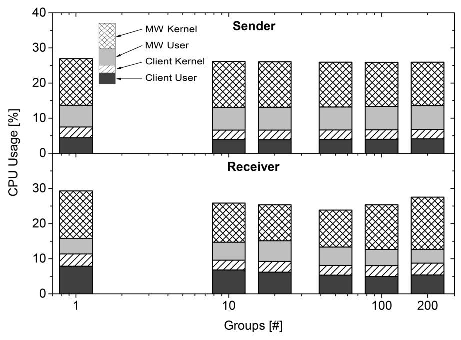
sender submits data at maximal capacity and receivers strive to process whatever arrives, passing it to the upper layer. We vary packet sizes, as the frequency in packet processing characterizes complexity, as well as the number of group names used within a communication experiment. With a random mixture of different groups in multicast traffic, we test for possible overheads from sorting packets to group channels that could lead to scalability issues.

First we present the results on transmission capacities. Figure 3(a) compares the data throughput at the sender, while Figure 3(b) displays packet reception of listeners. In both cases, the H∀Mcast IP processing approximates the native IP stack performance with minor flaws only for small packet sizes (< 400 Bytes). Application layer multicast (ALM) distribution adds overhead in packet headers and increases overlay routing complexity in the middleware. Correspondingly, the ALM performance is lower, but approaches the optimal throughput (dotted lines) from about 500 Byte packets upwards.

We measure the reliability of the middleware by packet loss at the receiver side. Figure 4(a) visualizes the relative losses as functions of the packet size. Both the H∀Mcast and the native IP multicast stack loose a few packets at small sizes due to the very high processing requirements of several hundred thousand packets per second. Figure 4(b) shows the efforts of the middleware in terms of CPU load. The box plots summarize the statistical distribution of relative



(a) Data Throughput



(b) CPU Consumption

Figure 5: Scalability for parallel groups: Throughput and CPU consumption for 1 KB packet size

packet processing overheads in H∀Mcast over IP multicast. On average, the H∀Mcast stack adds about three times the IP effort on top of the underlying IP stack.

Our final evaluation addresses the scalability of the stack as a function of groups operating in parallel. In this setting, our sender/receiver pair exchanges 1 KB packets with randomly varying group addresses at line speed. Figure 5 illustrates the dependencies of data throughput and CPU load on varying group numbers. Strikingly, the performance of the H∀Mcast middleware remains completely unaffected by multicast group handling.

Regarding the long history of squeezing IP implementations, the performance of our H∀Mcast prototype can be considered reasonable. In particular, results indicate negligible impact on packet processing imposed by the middleware. We conclude that this work may successfully serve as a proof-of-concept for establishing name-based multicast communication.

5. Conclusions and Outlook

In this paper, we have discussed the role of multicast communication in future name-oriented networks and identified the deficiencies of a pure unicast-based model. We thoroughly explored the aspects of group-oriented naming and demonstrated its potentials in a human-centric access to group forming, routing, rendezvous and security.

We presented H∀Mcast [3], our implementation of a name-oriented abstraction of multicast. This concept introduces a deployment-friendly approach to universal multicast that supports publish/subscribe of content by abstract names. Within the H∀Mcast middleware, we are able to access groups on a level of abstraction that allows for a simultaneous treatment of traditional IP-layer along with information-centric communication, while providing an automated mapping to available distribution technologies.

In future work, we will further narrow the gap between users (programmers) and the underlying multimedia networking technologies. By including name-based group access into high-level programming paradigms such as Actors [36, 37], the distribution of multimedia data will appear natural to the user, while remaining native to the network.

References

[1] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, B. Ohlman, A Survey of Information-Centric Networking, IEEE Communications Magazine 50 (7) (2012) 26–36.

[2] M. Gritter, D. R. Cheriton, An Architecture for Content Routing Support in the Internet, in: Proc. USITS'01, USENIX Association, Berkeley, CA, USA, 2001, pp. 4–4.

[3] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, I. Stoica, A Data-Oriented (and beyond) Network Architecture, SIGCOMM Computer Communications Review 37 (4) (2007) 181–192.

[4] L. Zhang, D. Estrin, J. Burke, V. Jacobson, J. D. Thornton, Named Data Networking (NDN) Project, Tech.report ndn-0001, NDN (2010).

[3] See http://hamcast.realmv6.org for a current release.

[5] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, Networking Named Content, in: Proc. of the 5th Int. Conf. on emerging Networking EXperiments and Technologies (ACM CoNEXT'09), ACM, New York, NY, USA, 2009, pp. 1–12.

[6] P. Jokela, A. Zahemszky, C. E. Rothenberg, S. Arianfar, P. Nikander, LIPSIN: Line Speed Publish/Subscribe Inter-networking, in: Proc. of the ACM SIGCOMM 2009, ACM, New York, NY, USA, 2009, pp. 195–206.

[7] B. Ahlgren, et al., Second NetInf Architecture Description, Tech.report D-6.2 v2.0, 4Ward EU FP7 Project (2010).

[8] W. K. Chai, N. Wang, I. Psaras, G. Pavlou, C. Wang, G. de Blas, F. Ramon-Salguero, L. Liang, S. Spirou, A. Beben, E. Hadjioannou, CURLING: Content-ubiquitous Resolution and Delivery Infrastructure for Next-generation Services, Communications Magazine, IEEE 49 (3) (2011) 112–120.

[9] J. Chen, M. Arumaithurai, L. Jiao, X. Fu, K. Ramakrishnan, COPSS: An Efficient Content Oriented Publish/Subscribe System, in: ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS'11), IEEE Computer Society, Los Alamitos, CA, USA, 2011, pp. 99–110.

[10] S. E. Deering, D. R. Cheriton, Multicast Routing in Datagram Internetworks and Extended LANs, ACM Trans. Comput. Syst. 8 (2) (1990) 85–110.

[11] K. Katsaros, G. Xylomenos, G. C. Polyzos, MultiCache: An Overlay Architecture for Information-centric Networking, Comput. Netw. 55 (4) (2011) 936–947.

[12] G. Tyson, A. Mauthe, S. Kaune, P. Grace, T. Plagemann, Juno: An adaptive delivery-centric middleware, in: CCNC, IEEE, 2012, pp. 587–591.

[13] P. T. Eugster, P. A. Felber, R. Guerraoui, A.-M. Kermarrec, The Many Faces of Publish/Subscribe, ACM Comput. Surv. 35 (2) (2003) 114–131.

[14] Y. Challal, H. Bettahar, A. Bouabdallah, A Taxonomy of Multicast Data Origin Authentication: Issues and Solutions, IEEE Communications Surveys & Tutorials 6 (3) (2004) 34–57.

[15] T. C. Schmidt, M. Wählisch, G. Fairhurst, Multicast Mobility in Mobile IP Version 6 (MIPv6): Problem Statement and Brief Survey, RFC 5757, IRTF (February 2010).
URL `http://www.rfc-editor.org/rfc/rfc5757.txt`

[16] T. C. Schmidt, M. Wählisch, O. Christ, G. Hege, AuthoCast — a mobility-compliant protocol framework for multicast sender authentication, Security and Communication Networks 1 (6) (2008) 495 – 509, special issue on Secure Multimedia Communications.
URL `http://dx.doi.org/10.1002/sec.86`

[17] P. Rodriguez, K. W. Ross, E. W. Biersack, Improving the WWW: caching or multicast?, Computer Networks and ISDN Systems 30 (22–23) (1998) 2223–2243.

[18] D. Rossi, G. Rossini, On Sizing CCN Content Stores by Exploiting Topological Information, in: INFOCOM Workshops: Nomen Workshop, IEEE, 2012, pp. 280–285.

[19] I. Psaras, W. K. Chai, G. Pavlou, Probabilistic In-network Caching for Information-centric Networks, in: Proc. of the second ICN workshop on Information-centric networking, ACM, New York, NY, USA, 2012, pp. 55–60.

[20] S. Eum, K. Nakauchi, M. Murata, Y. Shoji, N. Nishinaga, CATT: Potential Based Routing with Content Caching for ICN, in: Proc. of the second ICN workshop on Information-centric networking, ACM, New York, NY, USA, 2012, pp. 49–54.

[21] V. Jacobson, D. Smetters, N. Briggs, M. Plass, P. Stewart, VoCCN: Voice over Content-Centric Networks, in: Proc. of the Int Conf. on emerging Networking EXperiments and Technologies (CoNext'09). ReArch Workshop'09 (ReArch'09), ACM, New York, NY, USA, 2009, pp. 1–6.

[22] Z. Zhu, S. Wang, X. Yang, V. Jacobson, L. Zhang, ACT: Audio Conference Tool over Named Data Networking, in: Proceedings of the ACM SIGCOMM workshop on Information-centric networking, ICN '11, ACM, New York, NY, USA, 2011, pp. 68–73.

[23] M. Wählisch, T. C. Schmidt, M. Vahlenkamp, Backscatter from the Data Plane – Threats to Stability and Security in Information-Centric Network Infrastructure, Computer NetworksAccepted for publication. doi:http://dx.doi.org/10.1016/j.comnet.2013.07.009.

[24] H. W. Holbrook, D. R. Cheriton, IP Multicast Channels: EXPRESS Support for Large-scale Single-source Applications, in: Proceedings of SIGCOMM '99, ACM, New York, NY, USA, 1999, pp. 65–78.

[25] P. Savola, B. Haberman, Embedding the Rendezvous Point (RP) Address in an IPv6 Multicast Address, RFC 3956, IETF (November 2004).

[26] A. Rowstron, A.-M. Kermarrec, M. Castro, P. Druschel, Scribe: The Design of a Large-Scale and Event Notification Infrastructure, in: J. Crowcroft, M. Hofmann (Eds.), Networked Group Communication. Third International COST264 Workshop, NGC 2001. Proceedings, Vol. 2233 of LNCS, Springer–Verlag, Berlin Heidelberg, 2001, pp. 30–43.

[27] S. Ratnasamy, M. Handley, R. M. Karp, S. Shenker, Application-Level Multicast Using Content-Addressable Networks, in: J. Crowcroft, M. Hofmann (Eds.), Proc. of 3rd Intern. Workshop on Network Group Communication (NGC'01), Vol. 2233 of LNCS, Springer–Verlag, London, UK, 2001, pp. 14–29.

[28] J. Day, Patterns in Network Architecture, Prentice Hall, Upper Saddle River, NJ, USA, 2008.

[29] B. H. Bloom, Space/Time Trade-offs in Hash Coding with Allowable Errors, Commun. ACM 13 (7) (1970) 422–426.

[30] A. Broder, M. Mitzenmacher, Network Applications of Bloom filters: A Survey, Internet Mathematics 1 (4) (2004) 485–509.

[31] A. Ghodsi, T. Koponen, J. Rajahalme, P. Sarolahti, S. Shenker, Naming in Content-oriented Architectures, in: Proceedings of the ACM SIGCOMM workshop on Information-centric networking, ICN '11, ACM, New York, NY, USA, 2011, pp. 1–6.

[32] D. Mazières, M. Kaminsky, M. F. Kaashoek, E. Witchel, Separating Key Management from File System Security, in: Proc. of the 17th ACM Symp. on Operating Systems Principles, SOSP '99, ACM, New York, NY, USA, 1999, pp. 124–139.

[33] M. Wählisch, T. C. Schmidt, S. Venaas, A Common API for Transparent Hybrid Multicast, IRTF Internet Draft – work in progress 08, IRTF (April 2013).
URL `http://tools.ietf.org/html/draft-irtf-samrg-common-api`

[34] S. Farrell, D. Kutscher, C. Dannewitz, B. Ohlman, A. Keranen, P. Hallam-Baker, Naming Things with Hashes, RFC 6920, IETF (April 2013).

[35] S. Meiling, T. C. Schmidt, M. Wählisch, Large-Scale Measurement and Analysis of One-Way Delay in Hybrid Multicast Networks, in: Proc. of the 37th Annual IEEE Conference on Local Computer Networks (LCN'12), IEEE Press, Piscataway, NJ, USA, 2012.

[36] D. Charousset, S. Meiling, T. C. Schmidt, M. Wählisch, A Middleware for Transparent Group Communication of Globally Distributed Actors, in: Middleware Posters 2011, ACM, DL, New York, USA, 2011.

[37] D. Charousset, T. C. Schmidt, libcppa - Designing an Actor Semantic for C++11, in: Proc. of C++Now, 2013.