

NSF FIND

Breakout on “Networking at the Information Layer” (Nov 2007)

Editors: Kevin Fall (kfall@cs.berkeley.edu) Nadia Shalaby (nadia@bbn.com)

Host: Kevin Fall Notes: Joud Khoury (jkhoury@ece.unm.edu)

This is a brief report based on a discussion held among the listed participants. Over the course of two days, the group had a few hours to discuss what type of architectural implications would arise from considering information (as opposed to remote inter-process communication or IPC) as a primary item to be transferred, managed, and manipulated using a future clean-slate network architecture. Our task was therefore to consider networking “at the information layer.” Any errors, misunderstandings, or omissions in the following can be attributed to the authors. –KF & NS

Participants: Nadia Shalaby (BBN), Dmitri Krioukov (CAIDA/UCSD), Srinu Seshan (CMU), Deborah Estrin (UCLA), John Heidemann (USC/ISI), Vern Paxson (UCB/LBNL/ICSI), Arun Venkataramani (UMa/Amherst), Boon Loo (U Penn), Kevin Fall (Intel), Bob Kahn (CNRI), Joud Khoury (U New Mexico), Zhi-Li Zhang (U Minnesota), Dipankar Raychaudhuri (Rutgers/Winlab), Pablo Rodrigues (Telefónica Research), Ilia Baldine (Renaissance Computing Institute), Mike Afergan (Akamai Technologies)... **(apologies to anyone whose names was unintentionally omitted—please notify the authors of any names you know to be missing)**

Is there an ‘Information Layer’?

The group first discussed whether the assigned topic was essentially equal to content distribution. There was general agreement in the affirmative that it *includes* content distribution but there is likely more. The given term *information layer* was analyzed to determine whether we had been given a pre-determined edict that any relevant design should be layered. We decided to drop the term layer in favor of capability, so as to allow a greater degree of freedom in terms of implementation flexibility. Despite this decision, it is recognized that a detailed architecture and/or implementation might well adopt a layered approach.

Motivation and Approach

Today the Internet provides the most simple and basic service of moving bits from one place to another, and additional functionality is placed as a client of this particular basic service. This arrangement makes the Internet architecture less than ideal for supporting the goals of efficient distribution, access, and processing of information. We can manage to achieve these goals using engineering fixes and work-arounds, but many believe they can be achieved in a more efficient and powerful manner. Information-aware networks are, in our estimation, desirable for several reasons:

- Many applications are data and service-oriented, therefore native information support is desirable for greater efficiency and improvement management. For example, caching, policy routing, and rights management properties capabilities are desirable
- The current (Internet) architecture has a somewhat host-centric bias, given its origin as an IPC mechanism. Today, many applications do not care what particular host supplies or processes information, provided the results are useful and secure

While these points can be argued, it seemed less important to debate whether the overall idea of network information awareness is useful than to explore the design space. Therefore, we took as a stipulation the idea of the network being more “information aware” as desirable.

In terms of access to information, one of the constraining aspects of the current model is the necessity to couple time and space with communication in order to facilitate information access. In a less time-coupled network access abstraction (e.g., pub/sub, persistent database queries), it may be possible to more naturally express the desires among information *producers*, *manipulators*, and *consumers* and allow the network to more efficiently set up distribution and processing mechanisms among elements both within the network and beyond it (e.g., references to documents in libraries). These issues begin to touch naming, persistence and API concepts, and there was some discussion of the *handle* system (mentioned below).

We took a top-down approach to drive and focus the discussion:

1. Identify a set of applications representing a range of information processing behaviors
2. Identify coarse-grained application requirements placed upon the network by these applications
3. Derive implications for network capabilities to service these application requirements

Example Applications

We discussed several types of applications in hopes of covering a broad range of processing behaviors:

1. *Archiving* was discussed as one of the more temporally relaxed types. The issues surrounding very long term archiving are challenging. Discussion revolved around whether it was sufficient to establish open standards for archived material, or whether it is really necessary to not only capture archive documents but also all the supporting software and systems to render such documents. This issue is unlikely to be resolved firmly one direction or the other because open standards may take care of some significant portion of documents while proprietary formats in common use would tend to frustrate this approach.
2. *Searching* was singled out as an important application and routing can be considered as simply one form of searching. Based on the discussion above, the network should have a more sophisticated routing approach. Presumably, as a side benefit of such an approach, the ability to search for items other than paths to reach hosts would be supportable without additional

specific mechanisms. This relates to a lightning talk on embedding topology into hidden metric spaces (see lightning talk below), and a number of challenges remain.

As already suggested, it seemed apparent to us that a number of network information manipulators may provide various ‘value-added’ services from within the network. We briefly discussed what some of these services might include:

1. *Fusion*: processing data from multiple sources and re-publishing the product, with the benefit of attestation from the re-publisher
2. *Subsetting* : reducing data volume by various sampling/filtering techniques
3. *Anonymization*: obfuscation of data fields to confound various forms of correlation analysis. For example, re-writing IP addresses on traffic traces, transcoding, and statistics gathering.

While these applications presumably represent only a subset of the many types of applications a network and its architecture should support, we were able to utilize them to formulate a characterization useful for starting to define architectural requirements.

Information-Focused Applications and Actors

Information-focused applications range from content distribution to content processing/analysis, to on-the-fly request/response types of queries and responses. Based upon further thinking about the types of interactions these applications exhibit, we defined an axis of behavior capturing the “timeliness” of operation. An application may be very flexible in terms of its desire to send, process, or access information, or it may be highly time sensitive.

Processing or “enhancement” of data offered as a type of network service may be a primary driver of information-oriented networking. If so, we require terminology to identify the entities involved. For example: Who processes or enhances the data? When and where are these tasks accomplished? We coined the term *information actors*; in general, each has different timeliness requirements and semantics with respect to its information processing requirements. We decided upon the following three types of actors:

1. **Producers**: Defined as original authors of data. A producer’s timescale ranges from asynchronous to interactive.
2. **Consumers**: Defined as “final” users of the data. A consumer’s timescale ranges from persistent to request/response.
3. **Intermediaries**: Defined as actors who apply transformations, store and deliver data. This term is to be understood broadly, as an intermediary could be an automated media transformation service (e.g., transcoding), a delayed analysis service (e.g., human-in-the-loop diagnosis), a network-layer device (e.g., router), or archiving service.

Given these actors, it is then possible to characterize bilateral and multi-lateral network information access and processing in terms of the timeliness of interactions among various types of actors. For example, it is possible to have a real-time consumer receiving streaming video that was archived years ago by a delay-tolerant producer by way of an intermediary equipped with storage.

Coarse-Grained Application Requirements

Using the simple terminology of actors and timeliness, we attempted to formulate high-level requirements imposed upon the network. These requirements fall into four major areas:

- **Security in the broad:** authorship, attestation, attribution, data embargo/use control (e.g. via crypto). This involves several issues such as chain of custody (e.g. for debug and audit purposes), how to know who is being trusted, and the quality of the protection and security that the consumer or intermediary is receiving.
- **Privacy concerns:** addressed on two levels – who is talking and what are they saying, and how much say does the data “owner” have over these issues
- **Search and filtering:** involves several requirements such as (1) the degree of completeness [i.e., does a search guarantee a result of the searched item exists], (2) the existence of a rich query “language”, and (3) object description, which might include time and location.
- **Distribution quality (or quality of service):** this requirement would comprise (1) the identification of actors, which might include time and location, (2) ability to pay for better SLA quickly, and more fine grain, and (3) a framework for low QoS/QoD tolerance.

Implications for Network Capabilities

After identifying the coarse-grained set of requirements mentioned above, we outlined a more specific set of capabilities that the network (architecture) should provide in response to those requirements. We list these implied requirements in no particular order:

- **Storage in the network** in a manner that provides for temporal and location de-coupling. This also requires management of this stored information, usually within the intermediaries, but also (albeit to a lesser extent) within the producer and consumer nodes.
- **Intermediary processing or generalized ‘mashups’** introduce issues such as access mechanisms to data, as well as the need for appropriate standards and specifications. This may be partly underway already with the development of Web 2.0.
- **Auhorship/provenance/rights** to the information implies the need for some form of key infrastructure service, as well as the need for management control and a CRL-like capability assuming such capabilities are cryptographically based (which seems likely).
- **Naming of actors and objects** within the network necessitates some generalized mechanisms for this capability.
- **Search and rendezvous semantics** are necessary to provide a semantically richer information access that is perhaps agnostic to ownership and location within the network. A capability similar to current web searches but within the network itself may be appropriate.
- **QoS-on-demand:** This entails not static SLAs, but rather a diverse range of connection service and quality types, made available on demand (e.g. in seconds or less). A mechanism is desirable to express the QoS within the network by the application, as well as a mechanism to schedule resources as function of preferences.

- **Semantic API between application and network:** To enable the QoS on demand, and many of the other implications on this list, a network-application API is required that is rich enough to express all these capabilities, but flexible enough so as not to impose unnecessary constraints on the implementation. For example, selection of a storage intermediary may be left to the network to determine.
- **Distribution:** A network mechanism is needed to express fairly fine-grained distribution requirements, and the network needs to be capable to carry out these requirements.
- **Coordinated notion of time & location** is necessary for the sake of auditing, debugging and forensics.

Policy within the Network

It was agreed that we are pursuing a design philosophy in terms of the types of information manipulation and management capabilities a network should provide to its users. Some of the capabilities described above are assumed to be controlled by certain policies. To what extent should the network support policy? We understood *policy* to mean:

- *a combination of rights* (access, modification, publish, etc), as well as
- *distribution requirements* (when, where, how, and to whom is certain information available to/from certain actors)

An example is related to ISPs and how they participate in the “value proposition” of content. Today, they are largely paid based on traffic volume, yet they could be characterized as another important intermediary that could be specifically acknowledged in the architecture. This is in contrast to the current model (“dumb network”) where there is a relatively weak architectural abstraction of ‘middle boxes.’ Such a weak model makes utilizing such devices for value-added services and related accounting difficult. It was pointed out that there are some exceptions. For example, *Akamai* derives revenue from being a smart network overlay cognizant of both distribution performance and of distribution policy.

Policy on distribution and access invoked further discussion. An example of this relates to subscription services (e.g., NFL) where certain content is available in certain quality levels to certain consumers in certain places and/or times. Clearly, such policies could grow to be complex. It was felt that although the network should probably provide mechanisms to support such policies, it is useful to identify the minimum set of mechanisms required to achieve a reasonable set of policies. This topic requires further study (to say the least).

Security and Rights Management

There were differing points of view on whether much of the rights managements (authenticity, attestation, provenance, and data “use control”) should be considered under the mantra of “security.” Those resisting this classification note that security usually relates primarily to cryptographically securing information and the problem of rights management is considerably more complicated. Those supporting this classification argue that security can be interpreted “in the broad” so as to include these issues. In either case the term “*use control*” was suggested as a way to describe how data can be used once it has been transferred from its author. For example, types of use control might cover prohibitions

on collecting or using certain types of medical information, privacy rules regarding Internet traffic traces, data embargo rules common to scientific data collection, etc.

Regardless of the terminology within security in the broad, “information-centric” types of properties such as authenticity, attestation, and chain of custody/transformation were considered fundamental properties that should be provided by a network architecture, especially one contemplating intermediaries that may accomplish significant processing, storage, and distribution from within the network itself. In addition to these basic properties, it was deemed compelling to have a mechanism for users to determine what entities they are trusting (and for what purposes) when they use the network. As an example today, users often implicitly trust a collection of system and network operators that provide services. However, most users would not be able to identify which individuals are being trusted and for what behavior. Lastly, there was discussion regarding availability and high assurance. These issues can (and probably should) be separated from many of the more “rights-oriented” issues mentioned so far. It was mentioned that rights is an active area of work in its own right. There are existing standards (e.g., XRML) and other research in this area.

Lightning Talks

The group listened to five lightning talks.

1. *John Heidemann* discussed a world of many sensor networks being interconnected via the Internet, and underscored the importance of intermediary data processing capabilities, ranging from automated processing to human-in-the-loop image analysis.
2. *Nadia Shalaby* showed a framework for distributed systems, noting how today many middleware systems have to implement needed services and in some cases re-implement capabilities already present in the operating system (e.g., TCP) because it abstracts too much information—the middleware re-implements these types of capabilities in order to more efficiently manage its own data, distribution, timeliness and quality of service on behalf of the application.
3. *Dmitri Krioukov* discussed how a network topology could be embedded in a hidden metric space and a form of greedy shortest path selection can be utilized on the underlying space in order to find paths in the original topology. There was some discussion on how this would work when the topology changes, which is part of ongoing research.
4. *Boon Lu* discussed declarative networking where a variant of the Datalog language is used to express in-network processing.
5. *Ray Dipankar* presented insights from his “Postcard from the edge” wireless project, where he discussed the cache-and-carry architecture, a form of DTN routing architecture, where transport layer connections are not carried end-to-end, and where nodes may physically carry messages. This approach has a number of benefits: for example, when there is significant congestion over bottleneck links.

Related Work

We created a small, incomplete list of related work that may be useful as a basis for further contemplation on this style of networking. In terms of network architecture thinking, Huggle, DTN, DoT and DONA explore various aspects of data caching, naming, transport selection and opportunistic networking. Sensor networking research has provided some insights relating to interesting routing approaches (Directed Diffusion) and in-network processing (TAG). DOI, DONA, and HIP discuss the use of persistent identifiers for objects, and ROFL and name independent compact routing explore whether flat identifiers can ever really be considered as scalable. Some work is going on in terms of APIs, both at the network layer as well as in established products in the middleware arena (e.g. pub/sub in TIBCO, message queues, Mercury, Siena, Scribe). Some work focuses on cooperation (e.g. swarms) and creating an incentive structure for users to pool resources (Meraki is pursuing aspects of this issue). Programming network elements has been looked at in a general way (Active Networks) as well as within more limited contexts (declarative networking/P2).