This is the accepted version of the paper. The final version of this paper will be published at ACM ICN 2020

What's in a Name? Naming Big Science Data in Named Data Networking

Susmit Shannigrahi
Tennessee Tech
Cookeville, TN
sshannigrahi@tntech.edu

Chengyu Fan
Colorado State University
Fort Collins, CO
chengy.fan@gmail.com

Craig Partridge
Colorado State University
Fort Collins, CO
craig.partridge@colostate.edu

ABSTRACT

Naming data is the most important construct of Named Data Networking (NDN). The way a piece of content is named has a profound impact on content discovery, routing of user requests, forwarding, retrieval, and security. In addition, and perhaps more importantly, the naming of individual pieces of content seriously affects how *the network* behaves. While names are ubiquitous in NDN, the design of content names, and how different naming choices affect the network have largely been overlooked. NDN applications and protocols usually name content to fit their particular application scenarios, often derived from existing naming conventions. However, ad-hoc naming schemes often ignore the impact of these names on the network as well as the applications themselves. Our experience in applying NDN to multiple science use cases (e.g., Climate Science, Genomics, and High-energy particle physics) brought forward how different communities name their data, how these names can affect name-based networks and the commonality that exists across these communities. This work points out tradeoffs of different naming schemes, the effect of naming on the network and applications, and finally, provides a set of naming guidelines for future science applications.

1 INTRODUCTION

In today's Internet, data naming only affects application-level functionality; a misspelled file name or a wrong URL makes data access difficult but does not affect network functionalities (e.g., packet forwarding, routing). Naming in today's Internet is also application-specific; content names served by an HTTP web server do not affect the content names served by an FTP server, except when interoperability is needed.

NDN[27] is a future Internet architecture that relies on names for most operations. In NDN, the network not only uses content names for publishing and accessing content but also uses those names (or name prefixes) for in-network caching, routing, forwarding of packets, verifying trust, and performing in-network operations such as failovers and load balancing[1]. Consequently,the way applications name their data affects not only the applications themselves but also the network and other applications using the same network. In NDN, poorly conceived naming schemes has the potential to affect the network's scalability, performance, and usability. For example, if a climate data producer names their data as "`/ClimateData/YYYY/MM/DD`", the namespace is aggregatable and the producer only needs to announce "`/ClimateData`" into the network. The same data named as "`/YYYY/MM/DD/ClimateData`" would need the producer to announce n routes into the network where n is the number of years. Furthermore, these routes have the potential to waste in-network states such as the Forwarding Information Base (FIB), cache space, Pending Interest table, and network bandwidth.

Our contribution in this paper is as follows. First, we discuss current naming schemes in science communities and how these existing names translate into NDN names. Seconfd, we discuss how naming affects network operations, NDN data structures, and applications. Finally, we present a list of naming recommendations for the science communities as well as for the network operators.

2 BACKGROUND

2.1 Named Data Networking

Named Data Networking (NDN) [27] is an instance of Information Centric Networking (ICN). NDN uses application-defined, hierarchical, and human readable names for data publication, discovery, and retrieval. Consumer applications send "Interests" for "named data" that the routers forward based on their name. Upon reaching a data source (a producer, proxy, or an in-network cache) the Interest brings back a "Data packet" using the reverse path. The data is digitally signed by the data producer that helps establishing provenance. To forward Interests towards the Data producer, NDN utilizes [1] a Forwarding Information Base (FIB) that stores name prefixes and uses longest prefix match to find one or more outgoing interfaces. The Content Store (CS) caches returning Data packets and uses exact match to satisfy future Interests with the same name. Finally, a "strategy layer" allows NDN to utilze per namespace in-network states (delay measurements, loss etc.).

2.2 NDN Naming Studies

Several previous work have looked into NDN naming. Thompson et al. [23] created a name based API for that enables applications to work with hierarchical, named data collections using an abstract interface to NDN's request-response protocol. This work also provided a socket-like API for NDN application that can simplify applications that utilize name based operations. Afanasyev et al. [4] created NDNS (NDN DNS) protocol for looking up NDN names using operations similar to today's DNS. Shi et al. [22] provides a mechanism to filter packets based on their content names. Jahanian et al. presents an analysis frameworks, NSA [13] that models and analyzes NDN dataplanes where many of the verification are based on names. Cao et al [6] provides novel mechanism to announce content names to create a CDN-like mechanism. Fan et. al [10] and Olschanowsky et. al [18] provides naming schemes for science datasets. The NDN community has also published a tech report [2]

Conference'17, July 2017, Washington, DC, USA
2020. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

that provides general guidelines for NDN application developers to design their application namespaces. These works have looked at naming from specific applications' perspectives. Naming trade-offs across different application scenarios and how they affect NDN operations has largely been overlooked. This work aims to address this gap in literature.

3 EXISTING NAMING IN SCIENTIFIC COMMUNITIES

This section shows three existing science naming schemes that have been developed and deployed by the science communities in collaboration with the NDN community - Climate Science, High-energy particle physics, and Genomics.

The problem of organizing science data is hard due to the size, diversity, and number of datasets generated[18]. NDN naming paradigm fits nicely with the existing naming conventions of science data. NDN places few restrictions on naming, merely requiring that, (a) the name structure is hierarchical, (b) naming rules are globally agreed upon among content users, (c) name prefixes are allocated to publishers (similar to how the current DNS system assigns domain names), and finally, (d) names are human-readable providing some additional level of assurance [27].

We observe that there are many similarities in how these communities name their data. Names are almost always hierarchical and composed of several distinct components. The components (and therefore the names) are human-readable in most cases. Looking at the complete name, a domain expert can identify various information about the file and the data without actually looking into the file. However not all filenames are eleborate enough to present information about the content of the file and require additional metadata.

The fact that names are hierarchical (or can be converted into hierarchical names) fit naturally into the NDN paradigm. In some scenarios, converting the names to NDN names requires simply replacing the delimiters. In other cases, additional components need to be added, names need to be checked for errors, duplicate components need to be consolidated, and unnecessary elements removed. In some cases, these operations need consulting the metadata in the files, and in some cases even the data itself to mine for missing name components.

However, all names are not created equally, we have often encountered names where components were arbitrarily ordered in the naming schemas, components were missing or transposed, and components disagreed with the actual data in the files.

NDN naming, while flexible, also imposes the requirement for consistent naming. By naming data consistently, the communities can not only standardize data management logistics but also can gain multiple benefits such as request aggregation, caching, automatic failover and fast transfers from multiple sources.

The following sections provide more insight into current naming efforts and how we can convert existing names to consistent names that can benefit from a name-based network.

3.1 Climate Data Naming

The climate communities usually access individual files[10]. Each of these file names has several components that describe various attributes of the data, such as which project generated them, the temporal range of data in the file, the institute name where it was generated, and other.

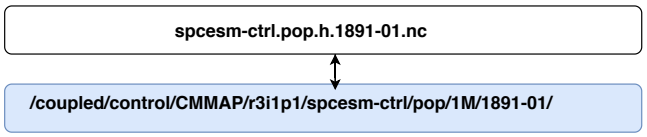

Figure 1: Naming a climate dataset into NDN. The name at the top is an existing name, the name at the bottom is the translated NDN name

Climate data names are hierarchical. The NDN names for climate applications can be divided into the routable prefix of the name that has a relatively fixed structure, and the model-specific portion of the name. Both these portions together form the complete name. The goal of the routable part of the name is to get an Interest routed to the set of machines that host the data. The following name components are standard across many different climate projects [18]: /Activity/[Sub-Activity or Product]/Organization/Model/ and can be used for data naming. For example, Figure 1 shows such a name.

It is relatively easy to see how to extend the names above to support operations such as subsetting. For example, the communities can add a portion (e.g., a suffix) to the NDN name to indicate which variables the subset should contain, for example using key-value pairs such as "`subset-variable=temperature and latitude = 30,60 and longitude = 90,120`".

3.2 High Energy Particle Physics data naming

High-energy physics (HEP) communities generally utilize "datasets", a collection of files. In this community, there is little importance of individual file names, and they can often be hexadecimal strings [21]. However, the directory structure is critical since it contains vital information such as parameters to expect in the data, when the data was generated, and which experiment generated the data. Not knowing the exact filenames does not impede data access; data consumers can ask for all files under a specific namespace.

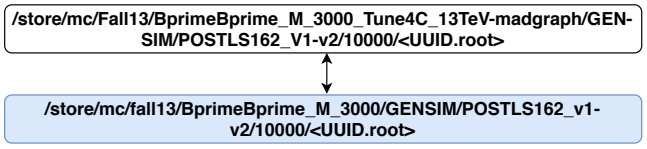

Figure 2: Naming a root dataset into NDN. The name at the top is an existing name, the name at the bottom is the translated NDN name.

High Energy Particle Physics (HEP) dataset names are also easily translated into NDN names. Figure 2 shows such a name. The HEP community has been able to agree on specific naming schemes and decide which name components are necessary[21]. Note that since the names are already agreed upon, the NDN names require little or no change. In this example, the original name is simply divided into logical segments separated with "/"s.

3.3 Genomics data naming

In Genomics, files are individually named just like the Climate and HEP communities. However, genomics communities use multiple

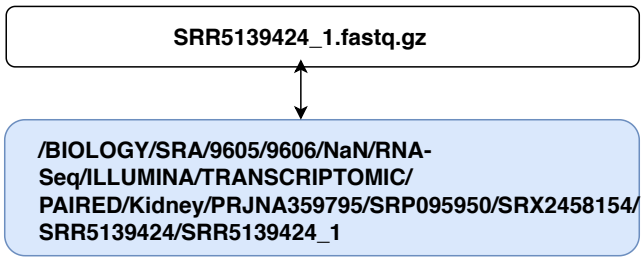

Figure 3: Naming a genomics dataset into NDN. The name at the top is an existing name, the name at the bottom is translated NDN name.

files with the same base name but different extensions. The common part of these names is the base portion of the name, and the extensions denote what types of data they contain. For example, one file might contain the base pair of a Genome, and another might content annotations. Since all files in the dataset have the same base name with different extensions, applications that use these names might be able to infer the existence of various files once they know the base name. Being able to use only the base name for inferring filenames is convenient but on the other hand, implicitly assuming the existence of files based on a name might be problematic for applications when files are missing.

Figure 3 shows a translated genomics data name. The actual filename provides very little information about what this file contains. The NDN name is derived from file directory information and metadata as well as the filename provides more information (that it contains RNA sequence of kidney, along with other domain specific information) to the scientists and is easier to index and use.

Modern genomic DNA data comes in the form of "static" reference genomes with coordinate-based annotation files, and "dynamic" measurements of genome output (e.g., RNAseq data files that contain RNA molecule snapshot strings in the tens of millions of sequence records) [8]. A common aspect of genomics datasets is that they can be named in an community agreed, evolution-based, hierarchical manner, which is easily mappable to an NDN framework. For example, an NDN name take the agreed upon form of "`/Genome/genus/species/infraspecificname/assembly-name`" as Figure 3 shows.

4 THE EFFECT OF NAMING ON NDN OPERATIONS

In NDN, in-network operations such as caching of content, forwarding of packets, routing, and measurements are name-based. The following sections discuss how naming choices affect these *in-network* operations. Before we discuss these choice, we first introduce the concept of a minimum usable data unit (MDU) in the context of scientific datasets.

4.1 Naming and Minimum Usable Data Units

In many science communities, such as high energy particle physics and genomics, individual files are not sufficient for experiments.

Table 1: An example MDU in the Genomics Community. It contains the genetic sequence of a fungal pathogen. All these files are necessary to run a computation, a single file is not important. We have shortend these names in the interest of brevity.

```
/Trichosporon/asahii/1.1/ht2
/Trichosporon/asahii/1.2/ht2
/Trichosporon/asahii/1.3/ht2
/Trichosporon/asahii/1.4/ht2
/Trichosporon/asahii/1.5/ht2
/Trichosporon/asahii/1.6/ht2
/Trichosporon/asahii/1.7/ht2
/Trichosporon/asahii/1.8/ht2
/Trichosporon/asahii/1/fa
/Trichosporon/asahii/1/gff3
/Trichosporon/asahii/1/gtf
/Trichosporon/asahii/1/meta.json
/Trichosporon/asahii/1/Splice_sites
```

Rather, a set of such files (often referred to as a datasets) are used together for experiments. We introduce the term "Minimal Usable Data Units" (MDU) to refer to these datasets in the context of NDN.

Specifically, an MDU is the unit of data that is used by domain applications. An MDU consists of a number of files often with various extensions and used together. Table 1 shows such an MDU in the genomics community that contains the genetic sequence of a fungal pathogen, "Trichosporon asahii". The rest of the files are as follows - ht2 files hold the actual data, meta.json is the metadata, .fa file contains nucleotide sequences, gff3 describes the genes and other features of the DNA, .gtf holds information about gene structure, and .Splice_sites identifies potential locations of mutations. All these files are **always** used together, and a single ht2 file is not useful without the other ht2 files or the accompanying metadata files. The concept of MDU occus in various other domains such as High Energy Particle Physics and Astronomy. However, the number of files in an MDU varies in different communities. In special cases, such as in some climate communities, each file is an MDU. The concept of MDU has profound impacts on NDN's in-network operations.

4.2 Name Length and Expressiveness

NDN only requires names to be hierarchical and encourages them to be human readable. The same content can have short names such as "`/BIOLOGY/SRR5139424_1`" or longer and more descriptive such as "`/BIOLOGY/SRA/9605/9606/NaN/RNA-Seq/ILLUMINA/TRANSCRIPTOMIC/PAIRED/Kidney/PRJNA359795/SRP095950/SRX2458154/SRR5139424/SRR5139424_1`".Both names work well in NDN though the longer name is more useful for most operations (albeit at a higher storage and processing costs and reduction of available space for payload in the returning data packets) as we discuss below. In the subsequent tables, we use the terms *expressivenames* and *shorternames*. *Expressivenames* are longer and contains more information while *shorternames* are brief and does not contain as much information.

4.3 Naming and In-network Caching

Content caching in NDN is based on names and by default, NDN utilizes exact name matching. These names need to be globally unique and content needs to be reusable for in-network caching to work. When an Interest for content arrives, NDN (by default) matches the full name of the Interest with the names of cached content. Appending timestamps (e.g., "`/Trichosporon/asahii_1/1/ht2/timestamp`") or identifiers (e.g., "`/Trichosporon/asahii_1/1/ht2/PublisherName`" to a content name makes it unique and eliminates content reusability in the network when default content store lookup policy is used; a cached content with the name "`/Trichosporon/asahii_1/1/ht2/timestamp`" does not satisfy a request for "`/Trichosporon/asahii_1/1/ht2`"). Appending a timestamp may be desirable for dynamic content (e.g., news) - however, most scientific datasets are static where they are generated once and reused often.

Table 2: Effect of Naming on Caching

	Expressive Names	Shorter Names
Popularity Based Caching	No effect	No effect
Probablistic Caching	Better context for decisions	Less information, worse decisions
Predictive Caching	More information for prediction, better decisions	Worse decisions
Context-aware Caching	Better context, better decisions	Less information, worse decisions

The goal of NDN, in the science data context, should be to cache and evict an MDU together. NDN does not provide a way to enumerate all files in an MDU. That is, we can not specify at the network layer that a request for $/Trichosporon/asahii$ should bring in all files under that namespace ($/Trichosporon/asahii/*$). This creates a problem on how requests for files in an MDU are expressed. Since the individual Intersts have no notion of an MDU, NDN can (potentially) forward them over multiple paths and create smaller caches that does not have the full MDU. When the next request come in, there is no way of directing the Interests towards a cache with the full MDU. Here is an example that illustrates this problem using Figure 4. Imagine a client is requesting two files - $/Trichosporon/asahii/1.1/ht2$ and $/Trichosporon/asahii/1.2/ht2$. The first file follows the path $D- > B- > A$ and the second file follows the path $E- > C- > A$. For subsequent requests, there is no way to direct the Interests for $/Trichosporon/asahii/1.1/ht2$ to go through B where it is cached. If this request goes through C, then the cache space in B is wasted and the client (A) receives no benefit from NDN's caching.

Similar problem happens with cache eviction too - not sending an Interest past a nearby cache potentially causes cache eviction faster even when content is present.

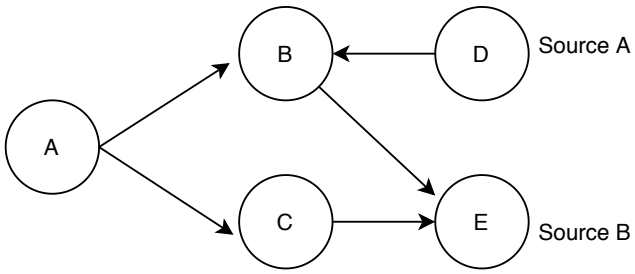

Figure 4: An NDN Topology

NDN caching schemes can be categorized into three broad categories - popularity based [15, 16], probabilistic[9, 19], and prediction based[3, 11]. In popularity-based caching, the cache eviction policies are based on how popular content is. In this case, naming has no bearing in cache eviction as long as content names are not made unique (e.g., by appending timestamps). If names are made unique, NDN replaces potentially resuable content with new content as we discussed above.

In the context of science data, looking at only the individual content popularity does not always yield the best result. Instead, context-aware caching is useful. For example, probabilistic and prediction based caching methods can use content names to cache content speculatively. In these cases, a longer and more expressive name might provide hints as to which content to cache. For example, if a user wants to study Trichosporon Asahii (see Table 1), any content that has either "Trichosporon" or "Asahii" in the name can be speculatively cached. A caching scheme may be able to use n name components or utilize a similarity function to cache the files of an MDU. Similar to caching, files in an MDU can be evicted together.

4.4 Naming and Interest Forwarding

Table 3: Effect of Naming on forwarding

	Expressive Names	Shorter Names
Default route	Finer grained forwarding	aggregated forwarding
Passive Measurement based	Finer grained forwarding, low overhead, fast	aggregated forwarding, low overhead, fast
Active Measurement based	Finer grained forwarding, high overhead, fast	aggregated forwarding, low overhead, fast
Probablistic	More accurate than shorter names, slow	Less accurate, fast

Forwarding of Interests in NDN is also dependent on naming[5][24]. In NDN, forwarding can be classified in three categories - default route (dictated by routing, such as the shortest path or lowest weight), default route complemented by measurements (e.g., lowest

delay, least congested), and probabilistic. Table 3 summarizes how naming affects forwarding. Longer, more granular names are useful in all three forwarding methods, default route, measurement-based, and probabilistic forwarding.

In the default routing based approaches, content producers (or their proxies) announce prefixes that populate the forwarding tables. For example, if a producer publishes all data beginning with "/Trichosporon/asahii", it can announce a prefix "/Trichosporon" into the network.

In measurement-based forwarding[14], the forwarding plane utilizes the default announcements but refines the forwarding decisions based on parameters such as throughput, delay, and packet loss of a link. Other forwarding decisions probabilistically forward packets with the hope that it will bring back data[7].

Naming plays a crucial role in efficient forwarding, especially in scientific communities. A huge amount of science data is already replicated and can benefit from NDN's measurement based forwarding. For example, a dataset is fully replicated and published (e.g.,"/Trichosporon") from multiple sources, the default forwarding scheme forwards Interests to one of the servers, which may not be the optimal source at that time. However, naming granularity is crucial since measurement to a generic namespace may not yield the best possible forwarding path. A content publisher that replicates content but only announce a very high-level prefix runs the risk of aggregating performance of all sub-namespaces under the larger namespace. For example, a smaller prefix of "/Trichosporon" may be the best route for "/Trichosporon/asahii" but not for "/Trichosporon/vadense". Rather, more specific prefixes such as "/Trichosporon/asahii" allow more granular handling of measurement and Interest forwarding.

Similar to the measurement-based forwarding, probabilistic forwarding can use arbitrary name component matching [7] as compared to the longest prefix match) for request forwarding. In these scenarios, more descriptive names that make sense to the users would be appropriate. If an application wants the network to forward packets based on a name component (such a rule might look for occurrence of "/asahii" in a name) occurring anywhere in the name, aggregated forwarding tables that only contain "/Trichosporon" will not be able to forward an Interest. However, if the forwarding table contains a prefix "/algie/amorphic/trichosporonaceae /asahii", the Interest can be probabilistically forwarded.

In science data, the concept of MDU also has a huge impact on how forwarding should be done. If some files in an MDU is large, once it is cached, there is a real incentive to forward future requests past the cache that holds the MDU. This is likely to reduce aggregate network load. First, there is currently no mechanism in NDN to accomplish this. Second, the "tussle" in this context is that this decision may not be in the best interest of the the requester if another faster data source is available or if the cache holding the MDU has a slower link. For optimal forwarding to an MDU, both proximity to the cached content and network characteristics to the cache should be considered. When content sizes are large, arbitrary forwarding (e.g., round robin) of Interests towards data sources should be avoided. Another observation is that if the network caches different parts of an MDU, the forwarding decision becomes more difficult. A manifest/catalog that enumerates the content names in an MDU should be useful to resolve this problem.

4.5 Naming and routing

Table 4: Effect of Naming on Routing

	Expressive Names	Shorter Names
Route Selection	More granular	Less announcements
Producer's control over forwarding	More	Less
Number of annoucements	High	Low

Routing influences how packets are forwarded in the network by populating the forwarding table (FIB)[12], therefore indirectly influences how packets are forwarded and which data sources are selected. An essential difference between forwarding and routing in NDN is that forwarding is based on FIBs in NDN routers. These routers are controlled by the network operators. On the contrary, routing announcements are controlled by the data publisher. The data publisher does not directly get to manipulate the FIB but can indirectly influence route selection. For example, when a publisher has two data servers and it announces "/Trichosporon/asahii" from both servers, the forwarding table may select either one of these possible routes [25] and the publisher has little control over which route it chooses (it may attempt to influcence routing by varying the weight but the local policy may ignore that). On the other hand, if the publisher announces "/Trichosporon/asahii" from the preferred server and "/Trichosporon" from the backup server, NDN's longest prefix match utilizes "/Trichosporon/asahii" over "/Trichosporon". Even when forwarding is probablistic, the data publisher can announce name prefixes that aids probablisitic forwarding (e.g., a name prefix "/fungi" maybe useful for forwarding "/genome/<organism name/fungi". Therefore, we observe that a more granular routing table ensures greater control over traffic patterns.

Content naming directly influences aggregatability of prefixes. If two species of "/Trichosporon" are named as "/Trichosporon /asahii" and "/genome/Trichosporon/vadense", the routing annoucements are not aggregatable. However, "/Trichosporon/asahii" and "/Trichosporon/vadense" can be aggegated under "/Trichosporon". Note that the aggregated prefix announcement is only possible for fully replicated namespaces. In case of partial replication, longer namespaces must be announced. For example, the Genus "Trichosporon" has seventeen subspecies. When the datasets are partially replicated, all the seventeen names must be individually announced into the network. If some organism have hundreds of subspecies, such annouuncements can cause FIB explosion. Use forwarding hints can retain shorter tables sizes in these scenarios. Publishers should utlize forwarding hints to a catalog that can then direct the request to a server that holds the actual data.

Table 4 summarizes these observations. While announcing a high-level prefix keeps the routing table smaller, it adversely affects other forwarding functions such as fine-grained load balancing, traffic engineering, and QoS, all of which can be based on names. On the other hand, the expressiveness comes at the expense of

more routing announcements and larger tables. Expressive names also allow for a more granular route section and allow the data producer to retain greater control over source selection and packet forwarding.

4.6 Effect of Naming on NDN data structures

Table 5: Effect of Naming on NDN Components

	Expressive Names		Shorter Names	
	Size	**Control**	**Size**	**Control**
Interest Packets	Larger	N/A	Smaller	N/A
Data Packets	Smaller payloads	N/A	Larger payloads	N/A
FIB table	Larger	Fine-grained	Smaller	Coarse-grained
RIB table	Larger	Fine-grained	Smaller	Coarse-grained
PIT Table	No effect	No effect	No effect	No effect
Measurement tables	Larger	Fine-grained	Smaller	Coarse-grained

In the previous sections, we discussed how naming affects network operations. NDN utilizes two packet types and three primary data structures for communication. Table 5 summarizes how naming affects these entities. Larger names in the Interest takes resources for forwarding and reduces the available space for returning payload. Further, large names or data appended to the name or put in the "Application Parameters" field may lead to high resource usage in the network. Generally, Interests should not be too large.

Cache sizes are independent of how content is named. However, in probabilistic and context-aware caching, naming affects how content is evicted and cached. The FIB and RIB tables are larger when names are more granular. The size of the Pending Interest Table (PIT) depends on the request pattern and is not affected by content naming. Finally, in the strategy layer, measurement tables are larger but are populated with more fine-grained measurement (see the discussion in the forwarding section above) when names are expressive, allowing more granular control over route selection and forwarding.

5 NAMING AND APPLICATION LAYER FUNCTIONALITY

Having discussed how naming affects network layer functionality, we now discuss how naming in NDN affects application layer functionality in big science. Table 6 provides a multi-dimensional view of how naming affects application functionality that, in turn, affects in-network functionality.

5.1 Data Publication

In science communities, data is often replicated - either fully or partially. When data is partially replicated, the routing prefixes from each content provider must reflect the partial data they serve. Take for example, a genomic dataset that has the following format - `"/genus/species/infraspecific-name/assembly-name"`. An example name following this format might be `"/Trichosporon/asahii/var_asahii/cbs_2479"`. If all datasets under `"/Trichosporon"` are fully replicated, all data publishers can announce the `"/Trichosporon"` prefix from all repositories. However, if one repository holds the `"/Trichosporon/asahii"` data and another holds the `"/Trichosporon/vadense"` data, they must announce both prefixes.

If NDN forwarding hints[4] are used, the publication granularity can be independent of names. In this case, the names are looked up in a catalog that tells the user which repository holds the actual content. For example, when a user looks for `"/Trichosporon/asahii"`, it sends an Interest to a predefined catalog, e.g., `"/Genome/Catalog"`. The catalog maintains a database that maps a content name to the actual repository holding the data. For example, upon looking up the repository for `"/Trichosporon/asahii"`, the catalog might redirect the user to `"/Genome/Repository/Instance1"`, which holds the actual data. In this case, the data name has no effect on data publication.

5.2 Data Discovery

Data discovery is an application-layer function. While it was (recently removed from NDN) possible to enumerate content names by (re-)expressing Interests into the network and excluding already discovered content (e.g., ask for `"/Trichosporon/asahii"`, retrieve some content, reexpress the same Interest excluding the name of the retrieved content), this approach is non-deterministic and might take a long time to complete. Science communities already use name catalogs (databases that hold content names) for data discovery. These catalogs use myriad techniques for finding names- keyword search, name trees, predictions, autocomplete. Table 8 enumerates these options. Descriptive names increase the efficiency of name discovery regardless of the techniques used.

Descriptive, human-readable names are important when publishing scientific datasets. For example, `"/BIOLOGY/SRA/cellular_organisms/Eukaryota/Opisthokonta"` provides enough context for the domain scientist to understand the names and the name catalog to index it. Each of these organisms also has a unique identifier associated with them (taxon IDs). So the same name can also be expressed as `"/BIOLOGY/SRA/cellular_organisms/2759/33154"`. However, the second name does not provide enough information to the user for an efficient search function even though these IDs are universally agreed upon.

5.3 Data Retrieval

In NDN, content retrieval is directly based on the names. Retrieval efficiency is directly related to routing, forwarding, and content caching efficiency. Expressive naming can enable efficient forwarding, in-network load balancing, failover, and intelligent strategies, as we discussed earlier.

Load balancing A special case of data retrieval is retrieving from multiple data sources[17]. NDN can support this by storing multiple paths to the same namespace. However, load-balancing is more efficient when content names are more granular, and endpoints are uniquely identified. Take for example the topology in Figure 4 where both D and E announces `"/NCBI"` prefix. Even if A utilizes routes through both B and C, the Interests might end up at node E, potentially eliminating any load-balancing benefits. If

Table 6: Effect of naming on application functionality

Application Functionality		Caching	Forwarding	Routing
Data Publication	Expressive Names	Useful for probablistic and predictive caching	Granular forwarding, Larger table	Better route selection, control over forwarding, Larger table
	Shorter, aggregatable Names	N/A	Lesser control over forwarding, Shorter table	Shorter routing table
Data Discovery	**Best accomplished using a catalog/manifest**	N/A	N/A	N/A
Data Retrieval	Expressive Names	Granular control over caching	Fine grained forwarding, more information for strategy layer, larger forwarding table, better multipath support	Larger routing table, better control over forwarding
	Shorter, aggregatable Names	Less information for predictive/probablistic caching	Coarse grained forwarding, less information for strategy layer, smaller forwarding table, multipath support for fully replicated content	smaller routing table, better control over forwarding
Load Balancing	Expressive Names	N/A	Fine grained, per namespace FIB entries, coarse grained load balancing	Fine grained RIB entries, more incluence over forwarding
	Shorter, aggregatable Names	N/A	FIB entries for larger namespace, fine grained load balancing	less influence over forwarding
Subsetting	**Application layer function, same as data retrieval**			
Resource Reservation	Expressive Names	Cache reservation for smaller namespace, smaller cache requirement	FIB/Bandwidth reservation for smaller namespaces, less unused resources	N/A
	Shorter, aggregatable Names	Cache can grow huge	More resources needed, potentially underutilized	N/A
Privacy	Expressive Names	N/A	FIB exposes more information	RIB exposes more information
	Shorter, aggregatable Names	N/A	FIB exposes high level prefix	RIB exposes high level prefix
Schematized trust	**Application layer function**			
QoS	Expressive Names	N/A	Fine grained, per namespace QoS	N/A
	Shorter, aggregatable Names	N/A	Aggregatated QoS for many applications	N/A

the endpoints are identified either as a name component or use forwarding hints, the network can utilize both sources, albeit at the expense of location transparency.

Subsetting Subsetting is another special case where computation (or subsetting request) is sent to the data[18], and a smaller portion of the data is retrieved. Table 9 summarizes the naming tradeoffs. The subsetting service can be accessed in two ways. First, it

Table 7: Effect of Naming on Data Publication

	Expressive Prefixes	Shorter prefixes
Replication	Needed for partial replication	May not support partial replication
Publication using forwarding hints	No effect	No effect

Table 8: Effect of Naming on Data Discovery

	Expressive Names	Shorter Names
Keyword-based search	More information but slower	Less information but faster
Autocomplete	Slower	faster
Predictive	Slower, less accurate	Faster

Table 9: Effect of Naming on Services

Special Namespace	Special Name component
Different paths for data and service	Same path for data and service
Robust	Service and name components must be unique

can be advertised as a service using the routing system (e.g., "`/NCBI/subsetting`"). An Interest, "`/NCBI/subsetting/<arguments>`", sent to this service will bring back a data subset. However, this requires a special namespace for subsetting that does not necessarily follow the same path as data; "`/NCBI/subsetting/Trichosporon/asahii/<arguments>`" might follow a different path than "`/Trichosporon/asahii/<arguments>`". It also requires multiple routing announcements.

Another option is to append the service name at the end (or at an arbitrary location) of the name "`/Trichosporon/asahii/<arguments>/subsetting`". This way, both the service and data follows the same path. However, this process can be fragile - a name component coinciding with a service name can cause unpredictable behavior.

5.4 Security and Privacy

Table 10: Effect of Naming on Security and Privacy

	Expressive Names	Shorter Names
Schematized trust	Better support	May not support well
Name Based Access Control	Granular	Higher level
Privacy	Less	More

In NDN, security aspects such as authentication, trust verifications, and access control are name-based[28]. Additionally, naming conventions can be utilized to automate trust checking and verification (schematized trust)[26]. Naming does not directly affect trust at the network layer since NDN routers typically do not verify trust. At the receiver side, a longer and more expressive name can support a set of rich and flexible trust schemes. One example might be the following - to perform a gene annotation, the "`gene-annotation`" component has to occur anywhere in a data name and must be signed by "`/human-genome/gene-annotation`" key (or a sub-key signed by this key).

Well-formed, hierarchical names are essential for schematized trust - access to content with a name in the form "`/NCBI/Trichosporon/asahii`" or "`/Trichosporon/asahii/../NCBI`" can be restricted only to the scientists from NCBI by requiring a user to present a key (e.g., "`/NCBI/Scientist/Key`") that is signed by NCBI's key (e.g., "`/NCBI/Key`"). Unstructured names without enough context make these security associations difficult to make. Table 10 summarizes the implications of naming on security. Expressive names provide more information for trust verification, name-based access control, and security associations. However, depending on the trust association and where the name components are placed (at the beginning, at the end of the name, arbitrary positions), the security verification may take longer for due to a potentially larger parsing cost.

Resource Reservation

Table 11: Effect of Naming on Reservation

	Expressive Names	Shorter Names
Per-namespace Resource reservation	Not necessary	Sufficient
Per-application Resource Reservation	Required	Difficult to utilize

Resource reservation[20] is widely used in data-intensive science for ensuring timely completion and data transfer performance. In big-data science, two types of resource reservations are possible, per-namespace and per-application. Similar to forwarding, aggregating a larger namespace makes it difficult to reserve resources at a finer granularity. For example, if two genomics applications/services "`/NCBI/gene-comparison`" and "`/NCBI/gene-annotation`" are combined under "`/NCBI`", any resources reserved for "`/NCBI/data-transfer`" will also be usable by "`/NCBI/gene-annotation`", which may not be desirable. For resource researvation, finer granularity names should be preferred to avoid unwanted resource uses by other applications.

Quality of Service

Similar to resource reservation, QoS in NDN is still a mostly unexplored topic. QoS mechanisms typically utilize Interest names for packet marking and shaping on a per-namespace basis. QoS over a shorter prefix potentially applies the same policy over a number of sub-namespaces where such namespaces may belong to different applications. When longer and more expressive names are used by applications, applying per-prefix QoS for individual applications is easier.

Table 12: Effect of Naming on Cache Eviction

	Expressive Names	Shorter Names
Fine grained QoS	Yes	Difficult
Fine Grained Resource Reservation	Yes	Difficult

6 GENERAL NDN DISCUSSION AND NAMING RECOMMENDATIONS

This section summarizes the lessons we have learned from naming science data and generalizes them into a set of recommendations for both science applications and the network operators. The recommendations are specific to science communities - other application domains might warrant a sligtly different set of recommendations.

6.1 Considerations for Science Applications

(1) Scientific data names should be built as a set of well-defined *components*. Data names should be hierarchical and human readable. We observe that existing file names are often very short and does not provide enough information for the users. These names should longer and incorporate some metadata into the names. Having more information in the names help an NDN network to make better decisions in terms of caching, forwarding, and measurement. They also help the users to better understand the content of the files. Longer names also makes it easier to index names for searching. For example, the name "`/BIOLOGY/SRA/cellular_organisms/Eukaryota/Opisthokonta`" provides more information than "`/BIOLOGY/SRA/cellular_organisms/2759/33154`" and should be generally preferred.

(2) While longer names are generally beneficial, they occupy space on the return Data packets. The default Data packet size in NDN is 8800 bytes. Care should be taken not to make the names too long since it reduces the amount of available space for payload.

(3) When including components in the Interest names, it is possible to include some data (e.g., application parameters) directly in the name or by utilizing the "ApplicationParameters" field. However, putting very large data into these fields should be avoided since these can lead to high resource usage in the network. Further, sending such larger Interests into the network may create an unintended denial of service attack.

(4) Content caching in NDN is based on names and by default, NDN utilizes exact name matching. These names need to be globally unique and content needs to be reusable for in-network caching to work. When an Interest for content arrives, NDN (by default) matches the full name of the Interest with the names of cached content. Appending timestamps (e.g., "`/cmip5/MIROC/timestamp`") or identifiers (e.g., "`/cmip5/MIROC/PublisherName`" to a content name makes it unique and eliminates content reusability in the network.

(5) While science applications should prefer longer and more desriptive names, longer names are at odds with what the network should prefer. Space and computation needed to utilize these longer names in an NDN network is higher. Fortunately, NDN allows multiple names to refer to the same object, and also provides the concept of a "link object" (a redirection from a shorter name to a longer name, or vice-versa), perhaps easing the decision process. For example, a shorter name with essential name components might help the scientists to understand the data, and at the same time, it might point to a larger name with additional components that can be used for cataloging and indexing.

(6) Science application should name data in such a way that individual datasets are identifiable as part of a larger collection (MDU). The network and the applications using these names should be able to utilize these datasets together. For example, MDUs should be cached and evicted together. If one file of an MDU is selected for eviction from the cache based on some probability, the same probability can be applied to all other files in the same MDU.

(7) More general (more common) name components should appear earlier in the hierarchy. Consider two applications at running at CSU. Placing the application name immediately after general name component speeds up name parsing. For example, a name like $/cmip5/csu/filteringapplication$ requires parsing only three name component before handing the data over to the appropriate applications. On the contrary, placing application names at the end requires parsing the full name before deciding which application should receive the request. Consequently, naming designs must weigh the trade-offs of placing name components earlier vs. later.

(8) Naming has implications on how replication work in NDN. In a fully replicated scenarios, the names can be anything (/cmip5/...). In case of partial replication, the names should be named in such a way that they can be classified under a few higher level prefixes. When data is expected to be fully replicated, care should be taken not to bind names to locations when data is expected to be replicated. The top-level prefix should describe the data, not an organization. For example, a name "`/CSU/CMIP5`" binds the data to a location (CSU), but a name like "`/CMIP5/CSU`" does not.

(9) Granular naming and route annoucement ensures better control over data movement. For example, if a producer announces two namespaces "`/CMIP5/MIROC`" and "`/CMIP5`" from two servers, NDN prefers the longer prefix (by default) that can allow the data producer control which server is more utilized for content delivery.

(10) More expressive names provide less provacy since they expose more information. This problem can be solved by encrypting the names. However, encrpyting the names might remove any caching benefits along with forwarding benefits. Therefore, sensitive name components should be separated into public and private portions. The routing portion of the names at the beginning (first n components) should not contain private information and should not be encrypted. The non-routing part can be encrypted without any loss in functionality.

(11) There might be cases where names do not aggregate well, e.g., in the genomics use case above. Name developers should be mindful that routing/forwarding table sizes might be at odds with intelligent in-network functionality.

(12) When datasets are partially replicated and the namespace is large, the in-network states (e.g., FIBs and RIBs) might explode in size. Publishers should utlize forwarding hints to a catalog that can then direct the request to a server that holds the actual data. This approach prevents state explosion while maintaining most of NDN benefits. However, note that some NDN features such as automatic failover does not work with this approach.

6.2 Naming Considerations for Network Operators

(1) From the netwroks point of view, shorter names save space and speeds up forwarding. NDN's forwarding speed is tied to the name length, utilizing longer names require more processing and therefore is generally slower. Furthermore, longer names require more space in in-network data structures, such as the FIB and the PIT. In networks where devices are resource constrained, shorter names should be preferred. Similarly, when very high end-to-end performance is needed, shorter names are better. However, expressive names are preferred for one-to-many scenarios, since they help with caching and forwarding.
(2) On the other hand, utilizing longer, more granular names that provide more information has the ability to make the network more flexible. Longer names are useful in all three forwarding methods, default route, measurement-based, and probabilistic forwarding. Similarly, these names can be better utilized for caching, routing, and other in-network operations such as load-balancing and failover.
(3) Science organizations spanning multiple organizations have two options to announce name prefixes - location dependent and location independent. For example, for the /CMIP5 namespace, two organizations CSU and UCLA can announce $/cmip5/csu$ and $/cmip5/ucla$, respectively. The alternate approach might be to have /cmip5 as a sub-namespace of participating organizations - such as $/CSU/cmip5$ and $/UCLA/cmip5$. However, the second approach creates a fragmented namespaces. However, more critically, the second approach also binds data to locations and should be generally avoided since creating in-network mechanisms (e.g. transparent failover) becomes tricky.
(4) In the context of science data, the network's goal should be to cache an MDU together. If some files in an MDU is large, once it is cached, there is a real incentive to forward future requests past the cache the holds the MDU. This is likely to reduce aggregate network load. First, there is currently no mechanism in NDN to accomplish this. Second, this decision may not be in the best interest of the the requester if another faster data source is available or if the cache holding the MDU has a slower link. For optimal forwarding to an MDU, both proximity to the cached content and network characteristics to the cache should be considered. When content sizes are large, arbitrary forwarding (e.g., round robin) of Interests towards data sources should be avoided. Another observation is that if the network caches different parts an MDU, the forwarding decision becomes more difficult. A manifest/catalog that enumerates the content names in an MDU should be useful to resolve this problem.
(5) Location transparency is often at odds with efficient content forwarding and resource utilization. For example, if load balancing between multiple sources is desired, the endpoints serving the content should be uniquely identified. Without knowing either the exact endpoint or the whole network topology, it is impossible in an NDN network to specify an exact data source. Figure 4 provides such an example.

This section shows how to name data to be compatible with an NDN network, how to translate existing data names into NDN, and presents general naming guidelines for NDN based applications. The next section will demonstrate how to find names once they are published. We show that naming data and finding names are the most critical components of the whole name-based ecosystem. Once data is named, and infrastructure for finding names is in place, all other functionality in the network becomes easy to implement. The effect is profound - scientific applications are not only simplified, but the intelligent functions in the network are shareable among various communities, reducing cost and efforts of implementing similar but domain-specific solutions.

7 CONCLUSION

Naming is a critical aspect of NDN since it dictates how the network and the applications perform. This work presents an comprehensive study of naming in science communities, discusses the various trade-offs of naming choices, and how naming affects scince applications.

In this work, we find that the concept of MDU has large impact on NDN's operations. The concept of MDU opens up several new research directions in NDN - joint caching and forwarding decisions, caching decision optimization, and others. Another observation is that expressive names that encode application specific information are useful to both applications and the network. However, utilizing such names in an NDN network requires more storage in the network, are expensive to process, and reduce the available space for the payload in the Data packets. In this paper, we point out those trade-offs and present a number of naming recommendations for both the scientific community as well as the NDN network operators.

REFERENCES

[1] 2018. NFD Developer's Guide. *Technical Report, NDN-0021* Revision 9 (2018).
[2] 2020. NDN Technical Memo: Naming Conventions - Named Data Networking (NDN). (May 2020). https://named-data.net/publications/techreports/ndn-tr-22-2-ndn-memo-naming-conventions [Online; accessed 19. May 2020].
[3] Noor Abani, Torsten Braun, and Mario Gerla. 2017. Proactive caching with mobility prediction under uncertainty in information-centric networks. In *Proceedings of the 4th ACM Conference on Information-Centric Networking*. 88–97.
[4] Alexander Afanasyev, Xiaoke Jiang, Yingdi Yu, Jiewen Tan, Yumin Xia, Allison Mankin, and Lixia Zhang. 2017. NDNS: A DNS-like name service for NDN. In *2017 26th International Conference on Computer Communication and Networks (ICCCN)*. IEEE, 1–9.
[5] Alexander Afanasyev, Cheng Yi, Lan Wang, Beichuan Zhang, and Lixia Zhang. 2015. SNAMP: Secure namespace mapping to scale NDN forwarding. In *2015 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 281–286.
[6] Jianxun Cao, Dan Pei, Xiaoping Zhang, Beichuan Zhang, and Youjian Zhao. 2016. Fetching popular data from the nearest replica in NDN. In *2016 25th International Conference on Computer Communication and Networks (ICCCN)*. IEEE, 1–9.
[7] Kevin Chan, Bongjun Ko, Spyridon Mastorakis, Alexander Afanasyev, and Lixia Zhang. 2017. Fuzzy interest forwarding. In *Proceedings of the Asian Internet Engineering Conference*. 31–37.
[8] Deanna M Church, Valerie A Schneider, Tina Graves, Katherine Auger, Fiona Cunningham, Nathan Bouk, Hsiu-Chuan Chen, Richa Agarwala, William M McLaren,

Graham RS Ritchie, et al. 2011. Modernizing reference genome assemblies. *PLoS biology* 9, 7 (2011).

[9] Gang Deng, Liwei Wang, Fengchao Li, and Rere Li. 2016. Distributed probabilistic caching strategy in VANETs through named data networking. In *2016 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 314–319.

[10] Chengyu Fan, Susmit Shannigrahi, Steve DiBenedetto, Catherine Olschanowsky, Christos Papadopoulos, and Harvey Newman. 2015. Managing scientific data with named data networking. In *Proceedings of the Fifth International Workshop on Network-Aware Data Management*. ACM, 1.

[11] Hesham Farahat and Hossam Hassanein. 2016. Optimal caching for producer mobility support in named data networks. In *2016 IEEE International Conference on Communications (ICC)*. IEEE, 1–6.

[12] AKM Mahmudul Hoque, Syed Obaid Amin, Adam Alyyan, Beichuan Zhang, Lixia Zhang, and Lan Wang. 2013. NLSR: named-data link state routing protocol. In *Proceedings of the 3rd ACM SIGCOMM workshop on Information-centric networking*. 15–20.

[13] Mohammad Jahanian and KK Ramakrishnan. 2019. Name Space Analysis: Verification of Named Data Network Data Planes. In *Proceedings of the 6th ACM Conference on Information-Centric Networking*. 44–54.

[14] Vince Lehman, Ashlesh Gawande, Beichuan Zhang, Lixia Zhang, Rodrigo Aldecoa, Dmitri Krioukov, and Lan Wang. 2016. An experimental investigation of hyperbolic routing with a smart forwarding plane in NDN. In *2016 IEEE/ACM 24th International Symposium on Quality of Service (IWQoS)*. IEEE, 1–10.

[15] Jun Li, Hao Wu, Bin Liu, Jianyuan Lu, Yi Wang, Xin Wang, YanYong Zhang, and Lijun Dong. 2012. Popularity-driven coordinated caching in named data networking. In *2012 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS)*. IEEE, 15–26.

[16] Muhammad Ali Naeem, Muhammad Atif Ur Rehman, Rehmat Ullah, and Byung-Seo Kim. 2020. A Comparative Performance Analysis of Popularity-Based Caching Strategies in Named Data Networking. *IEEE Access* 8 (2020), 50057–50077.

[17] Ashok Narayanan and David Oran. 2011. NDN and IP routing: Can it scale?. In *Proposed Information-Centric Networking Research Group (ICNRG), Side meeting at IETF-82*.

[18] Catherine Olschanowsky, Susmit Shannigrahi, and Christos Papadopoulos. 2014. Supporting climate research using named data networking. In *2014 IEEE 20th International Workshop on Local & Metropolitan Area Networks (LANMAN)*. IEEE, 1–6.

[19] Yang Qin, Weihong Yang, and Wu Liu. 2018. A probability-based caching strategy with consistent hash in named data networking. In *2018 1st IEEE International Conference on Hot Information-Centric Networking (HotICN)*. IEEE, 67–72.

[20] Susmit Shannigrahi, Chengyu Fan, and Christos Papadopoulos. 2018. Named Data Networking Strategies for Improving Large Scientific Data Transfers. In *2018 IEEE International Conference on Communications Workshops (ICC Workshops): Information Centric Networking Solutions for Real World Applications (ICN-SRA) (ICC 2018 Workshop - ICN-SRA)*. IEEE.

[21] Susmit Shannigrahi, Christos Papadopoulos, Edmund Yeh, Harvey Newman, Artur Jerzy Barczyk, Ran Liu, Alex Sim, Azher Mughal, Inder Monga, Jean-Roch Vlimant, et al. 2015. Named data networking in climate research and hep applications. In *Journal of Physics: Conference Series*, Vol. 664. IOP Publishing, 052033.

[22] Junxiao Shi, Teng Liang, Hao Wu, Bin Liu, and Beichuan Zhang. 2016. Ndn-nic: Name-based filtering on network interface card. In *Proceedings of the 3rd ACM Conference on Information-Centric Networking*. 40–49.

[23] Jeff Thompson, Peter Gusev, and Jeff Burke. 2019. Ndn-cnl: A hierarchical namespace api for named data networking. In *Proceedings of the 6th ACM Conference on Information-Centric Networking*. 30–36.

[24] Yi Wang, Keqiang He, Huichen Dai, Wei Meng, Junchen Jiang, Bin Liu, and Yan Chen. 2012. Scalable name lookup in NDN using effective name component encoding. In *2012 IEEE 32nd International Conference on Distributed Computing Systems*. IEEE, 688–697.

[25] Cheng Yi, Alexander Afanasyev, Ilya Moiseenko, Lan Wang, Beichuan Zhang, and Lixia Zhang. 2013. A case for stateful forwarding plane. *Computer Communications* 36, 7 (2013), 779–791.

[26] Yingdi Yu, Alexander Afanasyev, David Clark, Van Jacobson, Lixia Zhang, et al. 2015. Schematizing trust in named data networking. In *Proceedings of the 2nd ACM Conference on Information-Centric Networking*. ACM, 177–186.

[27] Lixia Zhang, Alexander Afanasyev, Jeffrey Burke, Van Jacobson, Patrick Crowley, Christos Papadopoulos, Lan Wang, Beichuan Zhang, et al. 2014. Named data networking. *ACM SIGCOMM Computer Communication Review* 44, 3 (2014), 66–73.

[28] Zhiyi Zhang, Yingdi Yu, Haitao Zhang, Eric Newberry, Spyridon Mastorakis, Yanbiao Li, Alexander Afanasyev, and Lixia Zhang. 2018. An overview of security support in named data networking. *IEEE Communications Magazine* 56, 11 (2018), 62–68.