Named data networking: towards a data distribution network

Nathan Djojomoenawie (s2191229)

3/11/2022

With the internet being commonly used for data distribution, it becomes increasingly clear that the current endpoint-to-endpoint network architecture is not ideal. To better suit this modern distribution use, it has been argued that the current host-centric IP architecture should evolve into data-centric architecture. One way of achieving this is with a network architecture called *named data networking* (NDN). This architecture is described in the 2014 paper of the same name, written by Zhang et al. [1] The following is a review of this work.

Architecture

With NDN, users would be able to request data from the network itself, rather than the network simply acting as infrastructure across which to send datagrams to a certain endpoint. Hence, NDN could be referred to as an anycast architecture for data rather than hosts.

All Data packets are identified by names, so users can specify what data they desire by supplying its name to the network. They can do this by sending an Interest packet onto the network. Interest packets contain the name of the desired data, which the network will use to return the Data packet of with same name.

Each Interest packet can result in at most a single Data packet being returned. This is called *flow balance*. Nodes can leverage this to do flow control and retransmission, but it does likely mean that there inherently will be quite some overhead and delay. This might be troublesome for applications that typically rely on UDP, such as voice calling. For TCP applications it is presumably less of an issue; the flow balance property is comparable current TCP acknowledgement (as the paper also claims), except that NDN pulls the data while TCP pushes it.

Forwarding

Routers forward Interest packet across the network until it reaches a node that has the requested Data packet or if that node is awaiting that packet. The node could be the data producer, but it is ideally a router close to the data consumer. Routers might have the data because they cache Data packets that they have previously forwarded. Once an Interest packet reaches a node that has the Data packet with the same name, the Data packet is forwarded over the reverse path of the Interest packets. This is effectively multicasting if multiple Interest packets requested the data.

The innovation of routers storing the data within the network (in-network storage) is arguably the greatest advantage of NDN. Hypothetically, it means that data can be delivered quicker due to it being closer to the consumer. It may also reduce network load, because Interest and Data packets may only have to be requested once on upstream links to serve multiple consumers on downstream links.

However, not all services have data that is requested like this. In regular NDN, routers only have Data packets that they have seen before. In this case, the benefits of in-network storage would only exist when multiple nearby consumers request the same data. They might also have to request the data roughly at the same time, depending on how long it remains in cache. Consequently, these services are unlikely to benefit from NDN, even though it would presumably still work.

Unfortunately, the paper does not meaningfully consider this issue and provides no quantitative data on the improvement NDN would provide in practice. This data would likely be important to convince network administrators to adopt NDN, so it probably will have to be obtained sooner rather than later.

Security

In NDN, data consumers are not necessarily in contact with data producers and therefore cannot rely on verifying the identity of other hosts. Instead, consumers should verify that the received data has been created by the right producer. This is done using signatures. Data packets contain a signature that binds the data name to the packet's content. Consumers should verify this signature using the public key of the producer that should have originated the data. Such keys, being data, can also simply be shared using NDN. The same applies to keys used for decrypting data.

While this architecture seems realistic, it does raise the obvious question of how consumers can verify that the received key is originated by the right producer; essentially the same problem as earlier. The paper does acknowledge this problem and its importance, but unfortunately does not provide a definitive solution.

Names

But what do the names of named data look like? A lot of freedom is given to application developers here, as the architecture leaves it mostly up to them design a naming scheme that is most fulfilling of their requirements. This is possible because names do not have any meaning to NDN routers. However, for scalability it is assumed that names are hierarchical, which the routers do recognize.

There is the additional benefit that developers are not limited to a certain length by the architecture, which means that the namespace in principle cannot be exhausted and NAT is unnecessary. Of course, in practice, names will likely not be able to be of arbitrary length, because then packets might become infeasible large. How name size should be limited is not discussed in the paper.

Naturally, it is also required that globally available data has a globally unique name, otherwise ambiguity would occur in the network when different Data packets share the same name. Avoiding this is done via namespace management, which is not considered part of the architecture and therefore the paper does not attempt to solve this. However, it does state that this is being studied.

Naming schemes' responsibility for version indication is not explicitly stated by the paper, but it can be inferred from it. It might also be necessary for names to indicate when a Data packet has the latest version of the data, as the architecture does not have a mechanism to do so. It is not explained what this would look like.

The network could be made responsible for this, which would mean that data consumers simply have to indicate that they want the latest version, or not indicate version at all. However, the network would likely have to regularly retrieve the latest data from the producer. Routers would also have to synchronize doing so, otherwise the same name might refer to different data. This would be against the requirement that globally available data has a globally unique name. The downside is that synchronization might be troublesome in practice.

Alternatively, the responsibility could lie with data consumers, which would have to indicate a specific version rather than simply requesting "latest". Determining what version is the most current naturally cannot be done by contacting the data producer every time. That would also negate the benefit of in-network storage. For dynamically generated data, the paper states that consumers should be able to deterministically obtain the name of the desired data either based on shared information or iterative retrieval. Presumably, the same would apply to determining the version number.

Development and deployment

Several NDN applications have been build as part of the development of the architecture. The paper lists the following four as primary examples: video streaming, real-time (video) conferencing, automation and management systems for buildings, and vehicular networking.

Unfortunately, the paper does not convince that NDN is an improvement over the conventional IP network architecture, because no evidence is presented to support that. However, several other works are cited that present more details on these application. More persuasive arguments might be found in there.

Thus far, NDN does not appear to have seen any commercial deployment. Nonetheless, the paper does give several compelling arguments why doing so is feasible. First of all, NDN does not require a replacement of IP infrastructure, as it can run over IP instead. This is a clear advantage for the adoption as NDN, as replacing IP is not eagerly done by network administrators. If this would happen at all, it would take extremely long. Moving from IPv4 to IPv6 serves as an example of this.

Secondly, NDN does not need to be deployed on a large scale. NDN can be deployed locally, which may provide local benefits. Meanwhile, there could still be a larger NDN network by interconnecting these local “islands” via tunnels. Therefore, only few parties have to be convinced of using NDN before adoption can begin. It also means that NDN can be used only by those who consider it beneficial, while those who do not want it keep using the current IP architecture.

While perhaps not (yet) practically relevant, this paper does provide an intriguing possible innovation of computer networking. To the interested reader, I would highly recommend going through the paper. If necessary only the first two sections that lay out the architecture. And it might well be that we one day watch our favorite TV shows through an NDN network, but most likely not before the operational benefits have been clearly demonstrated.

References

[1] L. Zhang, A. Afanasyev, J. Burke, V. Jacobsen, kc claffy, P. Crowley, C. Papadopoulos, L. Wang and Z. Beichuan, "Named Data Networking," *ACM SIGCOMM Computer Communication Review (CCR),* 2014.