Multipath Transmission for Content-Centric Networking in Vehicular ad-hoc Networks

Bachelorarbeit
der Philosophisch-naturwissenschaftlichen Fakultät
der Universität Bern

vorgelegt von

Thomas Kolonko
2017

Leiter der Arbeit:
Professor Dr. Torsten Braun
Institut für Informatik und angewandte Mathematik

Contents

List of Figures

List of Tables

Acknowledgments

On this page I would like to thank everybody who supported me to write this bachelor thesis. First I would like to thank my coach Eirini Kalogeiton. She supported me trough the whole process of writing this thesis, spend many hours for pair programming sessions and gave valuable inputs when nothing seemed to work anymore. After that I would like to thank Prof. Dr. Torsten Braun who allowed me to write this thesis in his research group. I am also very grateful for the resources that were generously provided by the research group of Prof. Braun.

Abstract

Content-Centric Networking (CCN) is a new network approach based on the information-centric networking paradigm (ICN), which tries to provide a more secure, flexible and scalable network. CCN does not operate on a host-to-host basis but emphasizes the content itself by making it directly addressable and routable. Knowledge of the topology is therefore not required as the routing is based on the names of the content and happens only locally on the intermediate nodes. This allows much more efficient communication in highly mobile and dynamic environments, where no static topology is given, and devices enter and leave a given area very frequently.

In this thesis a new routing strategy for VANET's has been developed in ndnSIM version 2.0 within the ns-3 network simulator. The current implementation supports only routing based on incoming and outgoing faces. Since Wifi and other forms of wireless networking broadcast data by design, a new mechanism has been introduced for forwarding and discriminating by mac addresses. Two new fields have been added to the interest and data packages for original and target mac addresses. FIB tables were extended with the new mac address fields in order to forward the interests to the correct upstream nodes. PIT tables were also extended so that the data could follow the interests way back to the requesting consumer. A new strategy has been implemented for flooding the interest if the FIB entries have not yet been populated with known mac addresses and to forward specifically hop by hop if the FIB entries showed actual intermediate target nodes with lower costs. (TODO: add some info about the multiple netDevices when done)

The scenario was implemented and evaluated using the ns-3 network simulator. Simulation with 100 nodes were conducted for 21 times and ..... (TODO: finish this part as soon as you have some results)

(TODO: CHANGE WORDING from Amadeo14 BEGIN) -¿ Performance evaluation is carried by means of ndnSIM, the official NDN simulator, that is overhauled for use in realistic wireless ad-hoc environments. Results collected under variable traffic loads and topologies provide insights into the behaviors of both forwarding approaches and help to derive a set of recommendations that are crucial to the successful design of a forwarding strategy for named data ad-hoc wireless networking ¡- (TODO: CHANGE WORDING from Amadeo14 END)

Chapter 1

Introduction

In the past years content production and dissemination have both drastically increased. That has led among others to the wide use of Content Distribution Networks (CDN) that tackled the problem of distributing content more efficient and without slowing down the main servers. Not only the amount of data has changed but also the overall mobility has increased exponentially. A commuter expects to stream full HD movies on her cell phone while traveling in trains to work at high speeds. Mobile IP tried to solve this, but the main problem of an secure and always active host-to-host connection still remains.

1.1 Motivation

NDN is an implementation of ICN and tries to solve the above mentioned problems through a paradigm shift in networking. It is build upon the same innovative concepts as other ICN implementations. These are among others the named content and the routing by that name as in-network caching of data. The communication is also based on the Interest / Data model instead of maintaining an end-to-end connection at all time. This makes NDN with ndnSIM (it's simulator) very appealing for mobile ad-hoc environments. There are a few forwarding strategies already implemented within ndnSIM but as of ndnSIM version 2.0 they all work through faces and point to point wire connections. That makes it very difficult to use in a wireless fashion, since all net devices have the same face id's.

1.2 Study Subject

The goal of this theses is to implement a basic forwarding strategy that can be used in a mobile ad-hoc environment which then can be used as a baseline. The basic implementation of the strategy should then be reiterated into a improved strategy with better bandwidth efficiency. This goal can be divided into four subtasks. First a new forwarding mechanism should be implemented by not only using faces but also a further unique identifier, like a mac address. FIB and PIT entries need therefore to be extended in order to hold additional information about previous and next nodes of both interests and data. The second task of the thesis is around implementing a forwarding strategy that decides how to route the interests, once the flooding is done and the

FIB and PIT entries have been populated with the correct mac addresses. The third part is applying both mentioned mechanisms into a mobile ad-hoc scenario where nodes move around leave and enter the observed premises. The fourth part is to achieve a multipath transmission where specific nodes simultaneously send out interests and receive data.

1.3 Outline

The remainder of this thesis is organized as follows. Chapter 2 explains shortly why a paradigm shift takes place from host-based networking to content-centric networking, explains what CCN is and why it could outperform the TCP/IP model. Chapter 3 explains ndnSIM and it's current implementation of the forwarding mechanism and the strategy as of version 2.0. Chapter 4 shows the steps taken to implement a new forwarding based on not only faces but also on macs and discusses the multipath approach taken. In Chapter 5 the results are evaluated and discussed. Finally a conclusion of the thesis is offered in Chapter 6. (TODO: take out the appendix or write here something).

Chapter 2

Related Work

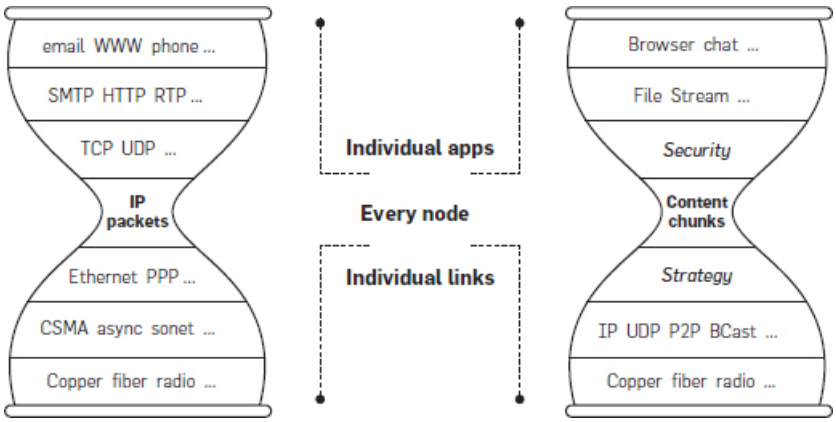
Todays use of the Internet is heavily based on content dissemination of all kinds of media like audio and video. TV-Shows, clips and tutorials on Youtube, podcasting for educational purposes like on Coursera need to be distributed around the globe. Popular TV-Shows are being distributed globally within the first few hours after being telecast to the audience through p2p networks. When a video clip goes viral millions of requests for the content are made from all over the world. In 2008 alone 500 exabytes were created and today's number is a multiple of that [TODO: reference]. This became possible, since computers and computing devices got so inexpensive that almost anybody can afford them. Limited storage on clouds is given free to all users by many providers. When the Internet was invented, there were only a few computers and few resources like tape storage devices, archives or computing power distributed geographically. A client requested some specific information or resources from a specific destination which needed to be known to the client. The client was connected through TCP/IP to the server, and after the connection was established and possibly secured, the transfer of the information needed by the client could start. Host-centric networking was very reasonable at that time, but the Internet evolved towards content-dissemination and with it's evolution the need for new and more efficient solutions.

TCP/IP is no longer best suited for todays use of the Internet. One of the main reasons is that a TCP/IP connection is established between two machines and requires it to be active at all times. For content dissemination a mechanism that supports multipoint to multipoint connections that do not have to be active at all times might be much better. Another reason is that a client often does not know where to get some specific information from (and doesn't really care) but knows exactly what it wants. Therefore a paradigm shift from host-centric networks started to evolve towards content-centric networks.

2.1 Named Data Network / Content Centric Networks

Information-centric networking (ICN) is a possible answer to today's problems with TCP/IP architecture. ICN originated as a possible new paradigm for the future internet to improve on scalability, reliability and efficient content distribution. One of the many ICN architectures is content-centric networking (CCN) originally introduced by Van Jacobson. It is being continuously researched around the globe and one initiative trying to implement CCN is Named Data Networking (NDN) project.

In CCN the content is made directly addressable and routable by name. An Interest (request by a client) is sent out and routed according to it's name until it reaches some endpoint that is able to respond with Content Objects to that Interest. Some of the main goals of CCN is better utilization of the bandwidth by increasing throughput and decreasing network traffic, better security, availability, flexibility and scalability of the networks. Better utilization of the bandwidth can be achieved by multicasting the same content to several endpoints and not re-unicasting the same content over and over again through the same channels near the content source. Also content caching in intermediate nodes reduces the strain on bandwidth and increases overall efficiency. Better security is achieved by hashing and signing the content itself instead the point-to-point connection between two hosts. (TODO: why is encrypting data not make it more private? No key -¿ no content) Encrypting the data leads to more privacy within the Internet and could substitute many current access policy patterns used by servers and webpages. (TODO: don't understand why not) Better availability and flexibility are intrinsically given by content caching. Better scalability is achieved by not needing pre-planned structures like CDN's and P2P networks. Data is not only kept at the content producer but everywhere along the route if necessary.



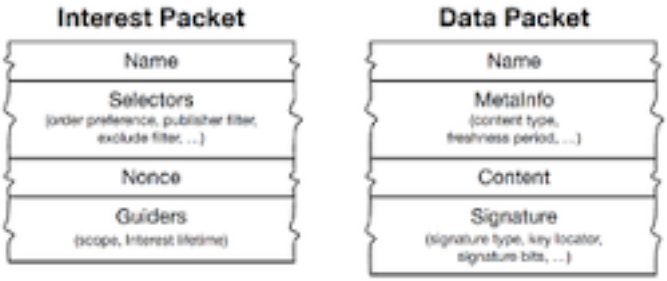
In Figure 1 the IP and CCN protocol stacks are compared to each other. Both protocol stacks are built in a modular fashion that makes the architecture very flexible and scalable. The thin waste of the TCP/IP protocol stack consists of IP packages that have a source and a destination address. This Network Layer is kept very simple and it makes only very weak demands on layer 2. This thin waist in TCP/IP protocol stack (*where*) is replaced in CCN with

a content layer (*what*) that describes what the package is and has even fewer demands on the lower layer, keeping much of the advantages from IP. Lower layers of the CCN protocol stack are responsible for the routing, encoding and decoding of the information while the higher layers consist of security and interpretation of the information. Because of the modularity CCN can be implemented on top of IP.

Two big differences of TCP/IP and CNN are the strategy layer and the security layer. The strategy layer is responsible for all dynamic routing decisions based on the name and the strategy. The strategy can be a different one for different namespaces. E.g. an emergency message could be always broadcasted according to it's name. The Security layer differs from the TCP/IP protocol stack, since the content chunks are signed and encrypted themselves, contrary to TCP/IP, where the connection is secured.

2.1.1 CCN/NDN Node Model

In CCN there are no clients and servers anymore but **consumers** and **producers**. Consumers request some information by sending out an **interest**. This interest packet consists of a content name, some selectors and a nonce. The interest is being forwarded according to the node's strategy until it reaches a node that can satisfy it. If a node can satisfy the received interest, it will respond with a **data** package consisting of the same content name as the interest, a signature, signed info and the data. The data will be sent back towards the consumer. The node having the requested information is called producer (it generates the data). Interests and data are received and sent out through interfaces which can be network or application interfaces. (TODO: is an intermediate node that does NOT generate it but supports it also a producer???).



The most important data structures for routing the interest to the producer and the data back to the consumer are called the pending interest table (PIT), the forwarding information base (FIB), and the content store (CS). A PIT entry is also needed to detect loops among other things. If an interest arrives at the node having the same content name as a previous interest, it's nonce will be checked against the PIT entry's nonce. If both nonces are the same the interest has been looped otherwise another consumer requested the same data.

PIT

The Pending Interest Table (PIT) keeps track of all the interests that have been forwarded towards potential content sources. It also keeps track of all incoming and outgoing faces of the specific interests (multiple in-faces and out-faces reflect the multipoint to multipoint characteristics of CCN). If a second interest with the same content name but different nonce arrives at the node, the incoming face will be added to the already existing PIT entry of the previously forwarded interest. The interest will not be re-forwarded and shortly after dropped. It won't be deleted right away since a looped interest is identified by it's nonce. When an interest reaches a content source or a producer data is send back. It follows the the breadcrumbs (faces) left in the PIT entries to find it's way back to the consumer. The PIT entries are deleted shortly after the requested data has been sent downstream.

CS

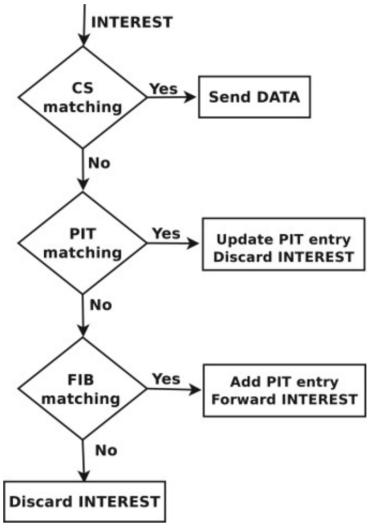
The Content Store (CS) is located within the intermediate nodes. It is a cache of data packages that have passed this node and have been saved for later use. That is a critical advantage over TCP/IP where data packages are meant only for point to point delivery. It depends on the implementation of the CS to decide which packages (solicited and unsolicited) should be saved to the cache and how they should be replaced if the cache is full. Current implementation focus mainly on Least Recently Used (LRU) and Least Frequently Used (LFU) replacement strategies. The CS needs to be searched before the interest is handed off to the forwarding strategy.

FIB

The Forwarding Information Base (FIB) is used by the strategy to forward interests upstream towards potential producers or intermediate nodes, that have cached the requested data. Every interest that needs to be forwarded will be matched against the FIB entries. If an entry is found (longest prefix match) then the interest will be send upstream to the outgoing faces. If there is no match the interest can be broadcast or dropped. The FIB is always checked last if there is no PIT entry (it has not yet been forwarded or it has been already satisfied) and there is no CS data that can satisfy the interest directly.

2.1.2 Forwarding of an Interest

Interests are forwarded based on the content name and the implemented strategy on all intermediate nodes. The above discussed tables are used for deciding if and how to further process the interest. The strategy is only responsible for forwarding the interests towards content sources or producers. The Data coming back follows the path of the interest back to the consumer.



When an interest arrives at some intermediate node the first thing to do is to check if there is already some data in the content store (CS). If the interest can be satisfied by some cached data the data is send downstream towards the consumer and the interest is dropped (not further processed). If the CS has no data with the same content name as in the interest, the PIT entries are checked. If a PIT entry already exists for the interest the node has already requested the data and is awaiting it. The incoming face(s) are added to the existing PIT entry and the interest is dropped. If a PIT entry does not exist yet and no cached data can satisfy the interest, the node needs to forward the interest upstream towards a potential content source. The strategy checks the FIB entries for a longest prefix match. If an entry is found a new PIT entry has to be created with the content name of the interest, it's incoming face and outgoing face (from the FIB). The interest is then forwarded according to the FIB and the strategy.

Sending Data downstream to the original requester is straightforward since no special routing is required. The Data follows the path of the interest left within the PIT entries (in-face(s) of the interest at that node).

2.1.3 Transport and Routing

As mentioned above the "content chunks" layer in the CCN protocol stack makes even weaker demands on the lower layers then the IP layer makes on layer 2. It operates on unreliable, best-effort packet delivery services in potentially highly dynamic and mobile environments. Interests and Data packages are expected to get lost and/or corrupted. In CNN the strategy layer of the intermediate nodes is responsible to resend the interest if within the timeout no data has been received. The strategy layer knows which outgoing faces were used and what the timeout was, therefore able to adjust the parameters for a retransmission. The same is true for the strategy layer of the consumer. If it does not receive any data back within a given timeframe, it too, will resend (possibly rebroadcast) the interest. Flow control can be managed by the consumer in terms of how many interests can be sent out before having received the first data packages back.

It is also managed on a hop-by-hop basis. Each intermediate node decides when to retransmit an interest due to loss or corrupted data coming back. The buffer for the interests is the PIT whereas the buffer for the data passing downstream to the consumers is the CS. There is no need for special congestion control techniques. (TODO: find the reference for that)

2.1.4 Sequencing

One of the big advantages in CCN over the host-to-host based TCP/IP approach is that the data in transition can be used many times by many different consumers requesting the same data possibly far away from the original producer. That leads to the problem of uniquely identifying the data in a self explanatory way. Consumers must be able to deterministically construct a name for the data without having previously seen it. Hierarchical names that reflect the content and the organizational structure of their origin are very well suited to solve that problem. Since the naming is absolutely irrelevant for the network (TODO: find the reference for that), applications can choose a naming that fits their need best. For example, to get the segment 34 of a video version 2 by group A of the University of Bern the name could be: **/unibe.ch/group/A/videos/introduction.mpg/_v2/_s32**
This name can be aggregated with more components if needed by the application or network. It only needs to adhere to previously specified rules.

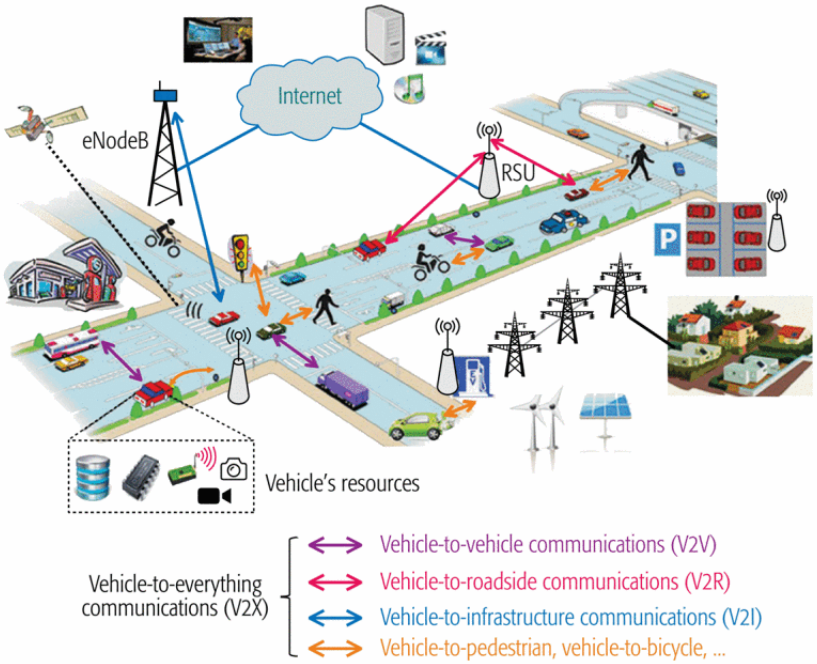
2.1.5 Network Security

CCN digitally signes and encrypts the content itself and not the connection over which it travels. TCP/IP needs to secure the connections which in turn must link the content to the server infrastructure. To trust a content the user must fetch it from it's original source making it very difficult to cache popular content and make it available to other users. For a rich and robust content-based security model the consumer must be able to assess the integrity (content is not corrupted), pertinence (what question does it answer) and provenance (who claims this is an answer). TCP/IP can only provide weak integrity through a checksum and only implicit pertinence and provenance through securing the host to host channel and therefore trusting the source and destination addresses. CNN on the other hand transparently provides then content name and therefore the meaning of the content which satisfies pertinence. Through public key signatures the consumer and any node in between can check the content's authenticity, therefore verify integrity and satisfy provenance. Content Protection and Access Control could be solved solely by encrypting the content with different keys. No trusted servers or directories would need to enforce complicated access policies on the filesystem. Expensive authentication services like SWITCH AAI could be saved. If for example certain documents within a database should be only readable by a certain group of people these documents could be simply encrypted while still accessible to everyone. Only the people with the correct key could encrypt it and use it. Network security is improved against many classes of network attacks. Every node can possibly (if the resources allow it) check the integrity of the data and cache it for further use. There is no single host that provides the data, therefore hiding content from consumers is very difficult. DDoS attacks with data packages are not really a problem since every node can ignore unsolicited data coming in. If resources allow it, the node can simply store it in CS and mark it as unsolicited. If the space

runs low on the CS these data packages will be removed first. No propagation of unsolicited data takes place since no PIT entries exist for the data. DDoS attacks therefore would have to be done through interests. If the prefix stays the same the PIT entry get's updated for every new interest, but no forwarding takes place. If the prefix changes constantly the strategy has many means to take action like limiting the rate of the interests with certain prefix patterns or lower prioritization of interests that result in data coming back.

2.2 VANETs

Vehicular ad hoc network's (VANETs) operate in very dynamic and mobile environments under possibly poor and intermittent connectivity. To the few static roadside unites (RSU) there are many devices ranging from trackers and phones on pedestrians other vehicles like cars, motorcycles or drones to airplanes and satellites as shown in Figure (TODO: X).



In such dynamic and mobile environments the TCP/IP based approach that focuses on the source and destination and their secured connection quickly becomes a burden. Dynamic name-to-IP resolutions are difficult to do and infer a high management overhead. Keeping the connection up in urban areas where signal propagation is obstructed frequently can quickly become a challenge. Mobile IP is a workaround for this problem but does not solve the issue really. The CCN approach on the other hand seems to be very well suited for such ecosystems where the focus lies on the information itself (trusted road safety information for a specific area) and not on the identity of the host (other vehicles, RSU, Internet) the data might have originated from. With in-network caching the CCN approach also tackles the problem of poor signal strength and intermitted connectivity within a very heterogenous network system. A vehicle can even store information and propagate it to a otherwise disconnected area through a *store-carry-and-forward* mechanism. The multipoint to multipoint characteristics already mentioned before allows to aggregate the same interests (maps, safety warnings, road condition, congestion warnings....) and multicast the arriving data through different faces and different channels back to the consumers simultaneously.

There are two main routing schemes for VANET's. In the *proactiv* scheme the content providers advertise periodically in order to keep the FIB entries updated with fresh routing information. In the *reactive* scheme no advertisement by the content providers is done and the FIB's are populated based on interest flooding. Flooding-based discovery seems to be better suited for VANET's since periodical FIB updates on all intermediate nodes incur a high and mostly unnecessary cost on the network. An interest is able to find it's way quickly to some node having a copy of the data or the producer itself. Collision avoidance and packet suppression is done on lower layers, although packet suppression will be done in the forwarder module in ndnSIM (see chapter 3). Caching policies in VANET's differ slightly from regular CCN use since the vehicles are moving fast into and out of regions relevant to the data. Data about possible congestion gets outdated pretty quickly so do often warnings of any kind. Further the question arises if unsolicited data should be cached into CS and if yes, under which conditions and for how long? As mentioned before vehicles could link otherwise disconnected areas, but in that case the data naming should be clear or that specific intent.

While the vehicles and devices are getting smarter and are being equipped with ever more sensors, it is expected that they will produce a huge amount of data. Most of it will be aggregated and logged, some of it will be available for distribution some of it should remain private. There are many open questions how to best support the information flow given the networks capacity. Although not originally intended by ICN the intermediate nodes could be included into in-network processing of the data like filtering useless data or aggregating redundant information.

Chapter 3

ndnSIM

NDN is a newly proposed Internet communication paradigm that tries to keep most of the well accepted and tested TCP/IP Internet architecture while evolving the thin waist introducing many new features. These features differ in fundamental ways from a point-to-point communication architecture and need to be simulated extensively under ever changing parameters and implementations.

ndnSIM is an open source simulator for NDN networks within the existing NS-3 simulator framework. It's main goals are to facilitate experimentation inside the research community and make all basic NDN protocol operations accessible and therefore changeable like routing, data caching, packet forwarding and congestion management. Packet-level interoperability with CCNx implementation is given in order to support traffic measurements, traffic traces and analysis tools between CCNx and ndnSIM. Large-scale simulations should be supported and made easy to set up through helper classes. Helper classes automate the repetitive creation of single entities like nodes and set them up in a standardized way. Therefore the simulator has been implemented in a very modular fashion making it very easy to modify or replace or re-implement specific components like the FIB , PIT or forwarding strategy. Replacing components have no or minimal impact on other components as long as they adhere to the modules API's and other components they interact with.

3.1 Design overview

NS-3 and ndnSIM both follow a philosophy of maximum abstraction for all modeled components making experimentation on one hand very fine-grained and on the other hand very isolated and decoupled from the rest. The NDN core protocol stack can be installed on every simulated network node in a consistent manner through helpers that take care of all the parts that need to play together like the inter-node communication with installed applications through their respective application faces.

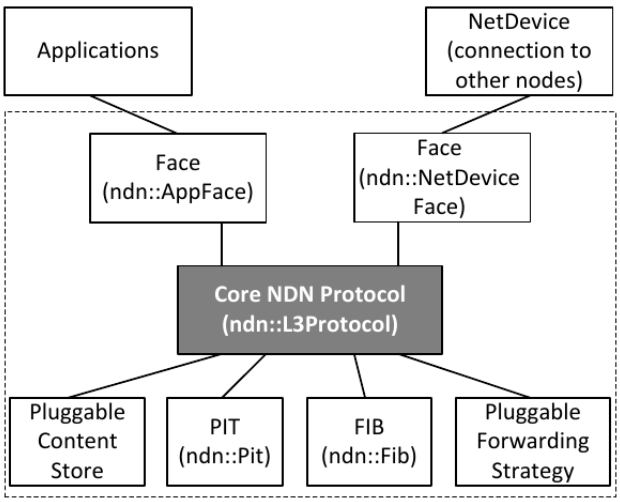


Figure 3.1: Basic component abstractions in ndnSIM

The basic component abstractions are seen in figure 3.1. The core NDN Protocol is the ndn::L3Protocol that receives interest and data packages from upper and lower layers through the corresponding faces. As of ndnSIM version 2.0 there are two distinct faces. The application face (ndn::AppFace) is responsible for inter-node communication between the application and the node itself while the net device face (ndn::NetDeviceFace) is responsible for inter-network communication with different nodes. Other faces for different purposes are expected to be added by the core developer or the community as needed. The ndn::ContentStore is an abstraction for in-network caching of data and can easily be omitted or replaced by another implementation of a different storing policy. The ndn::Pit abstracts the data structure that is responsible to log all received interests with their nonce and incoming faces while the FIB abstracts the data structure to guide the strategy in interest forwarding. The ndn::ForwadingStrategy is responsible to implement how interests and data are forwarded. That includes lookups in the content store for cached data, in PIT for already forwarded interests and in FIB if both previous searches didn't yield any matches. Each step is represented as virtual function calls in the forwarder header class of ndnSIM and can be overwritten.

3.1.1 Face abstraction

The face abstraction plays an important role in the overall modularity of the NDN simulator by acting as an interface therefore making ndnSIM design independent from any underlying transport layers. All communication between application and the NDN core protocol and other nodes with the L3 protocol happens through faces that take care of any needed conversion.

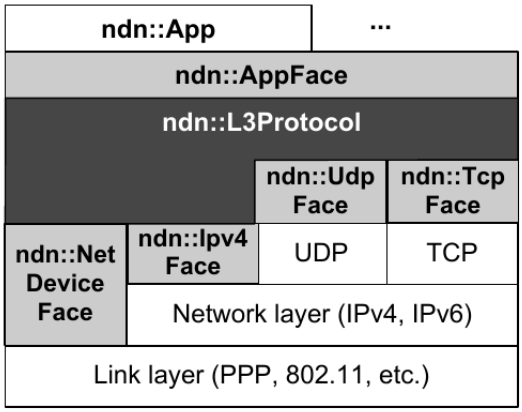


Figure 3.2: Face abstractions around ndn::l3Protocol

As shown in 3.2 the application can communicate with ndn::L3Protocol through the AppFace whereas for inter-node transmission it depends on the transmission protocols which faces need to be used. TCP and UDP have their respective ndn::UdpFace and ndn::TcpFace. IPv4 and IPv6 also have their own ndn::Ipv4Face making it particularly easy to implement the NDN architecture on top of IP or even TCP/IP. If NDN needs to be implemented without TCP/IP protocol it can communicate directly with the link layer through the ndn::NetDeviceFace.

3.1.2 Content Store abstraction

The CS is crucial for the NDN Internet architecture as it caches data for later use in potentially all the intermediate nodes. It can do rudimentary error recovery and multicast the data asynchronously downstream to the requesters. The replacement policy does determin what data is saved into cache and how it gets replaced or deleted after a certain time. Currently implemented versions of the CS support Leas Recently Used (ndn::cs::Lru), First In First Out (ndn::cs::Fifo) and a Random Replacement Policy (ndn::cs::Random). Each implementation is based on a dynamic trie-based data structure with hash-based indexing as are the PIT and FIB implementations.

3.1.3 Pending Interest Table (PIT) abstraction

The PIT data structure keeps information about each forwarded interest. Each PIT entry is uniquely identified by the content name of the interest. It holds a list of all incoming faces on which the interest was received and a list of all outgoing faces the interest has been forwarded to. The arrival and expiration time are also kept in order to retransmit a lost interest. The nonce of an interest is a randomly generated number that is attached to the interest and identifies the interest for loop detection. Different consumers issuing the same interest will very likely have different nonces. In that case no loop is detected and the incoming face is added to the already existing PIT entry.

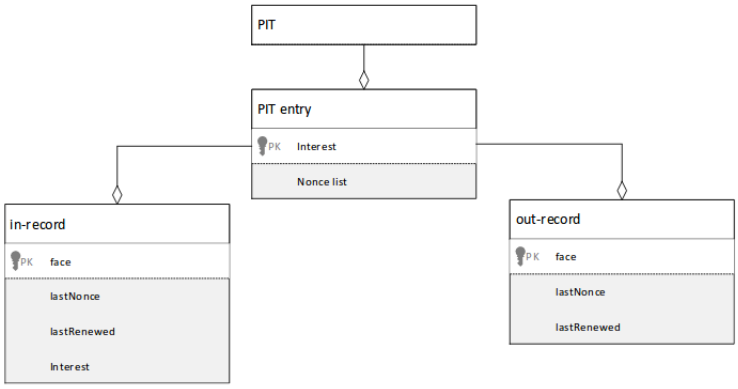


Figure 3.3: PIT data structure

Figure 3.3 shows the different PIT classes and how they relate to each other. There are also two additional timers on every PIT entry that are not explicitly shown here. The *unsatisfy timer* is the lifetime of an interest and counts down. If it reaches 0 the PIT entry has expired and needs to be forwarded again. The *straggler timer* starts counting down as soon a interest got rejected or satisfied. The straggler timer gives the node the time to detect further loops by the satisfied or rejected interest still floating within the network. Also measurements might require to still finish just after that event. After the straggler timer runs out the PIT entry is deleted. The PIT abstraction provides basic functions to insert, lookup, delete and get measurements.

3.1.4 Forwarding Information Base (FIB) abstraction

The FIB guides the forwarding strategy in making decisions about Interest forwarding. It is similar to an IP's FIB but contains name prefixes instead of IP prefixes and holds several interfaces (out-faces) therefore enabling multicast forwarding. The faces are ordered according to their cost (routing metric) putting the cheapest connections at the beginning of the list. The lookup is performed on the content name prefixes. The longest prefix match yields the requested FIB entry with it's outgoing face(es).

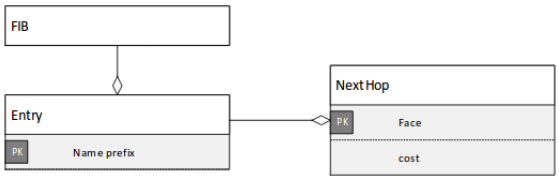


Figure 3.4: FIB data structure

As shown in Figure 3.4 each FIB entry that has been identified by longest-prefix match has an aggregated collection of NextHops. The NextHop collection must be non-empty and for every NextHop there is one outgoing face toward a possible content source. There may only be

one NextHop with a specific face id. The FIB abstraction provides basic insertions, deletions and exact match operations.

3.1.5 Forwarding Strategy abstraction

ndnSIM's modular architecture allows to experiment with different types of forwarding strategies without having to adapt any core components. The forwarder class interacts with the strategy and has the following important functions:

- *onIncomingInterest*: checks for localhost violations and if the interests has looped. It then inserts or updates the pit, resets it's timers and looks if the interest can be satisfied by cached data.
- *onContenStoreMiss*: if there is no data to satisfy the interest, the strategy is called.
- *onOutgoingInterest*: this function is called by the strategy and forwards the interest to the outgoing faces.
- *onIncomingData*: checks for localhost violations and if there are any PIT entries for that data. If there are no PIT entries the data is dropped, otherwise the data will be handed on to the onOutgoingData function.
- *onOutgoingData*: forwards the data downstream towards the content requester.

The strategy is called on three occasions. The first time from the forwarder::onContentStoreMiss() when the decision has to be made how to forward the interest upstream. A second time from the forwarder::onContentStoreHit() to decide how to proceed with an satisfied interest. And a last time from the forwarder::onIncomingData()

The strategy has one important function:

- *afterReceiveInterest*: it receives the interest and the corresponding FIB and PIT entries from the forwarder and decides how to forward them further and through which faces.

3.2 Simulation Environment

The NS-3 simulator supports several simulation environments. In this theses NS-3 PyViz was used as a live simulator. PyViz also has an interactive console that vizualizes the connections between the nodes of the simulation. It also can be used to degub the state of the running object, show PIT and FIB entries in live and where packages are being dropped.

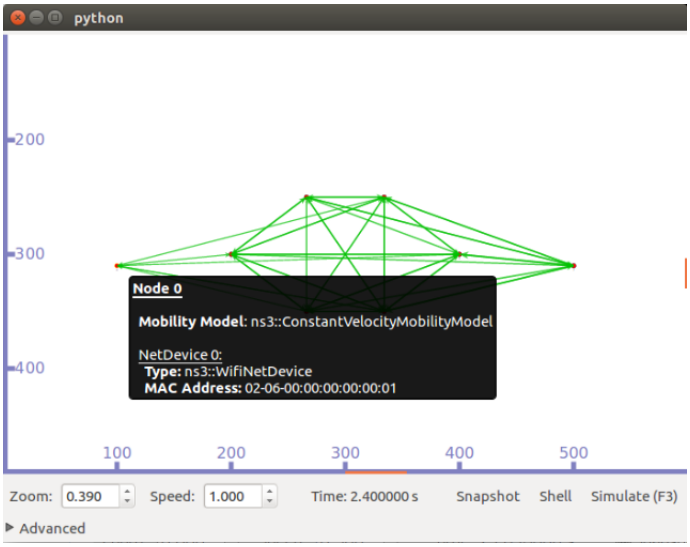


Figure 3.5: Simulation of a possible scenario

Figure 3.5 shows a current scenario in development. Zoom and speed can interactively be set, the simulation started and paused at any time. Node specific information can be shown while hovering over the node.

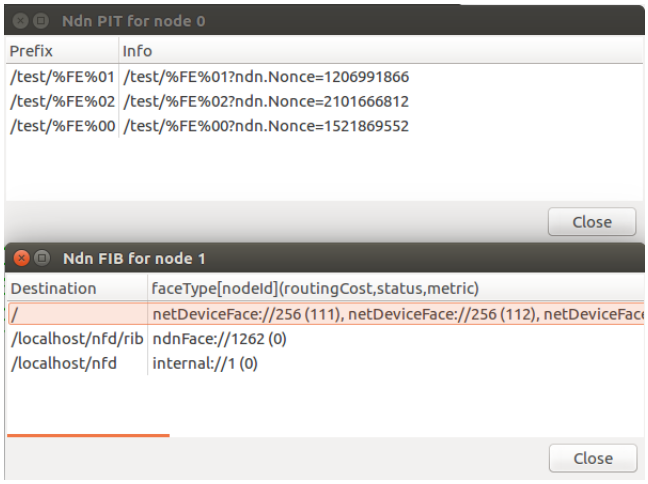


Figure 3.6: PIT and FIB entries in real time

Figure 3.6 shows all the PIT and FIB entries in real time. Face id's with the corresponding face type are also shown for easier debugging purposes.

Chapter 4

Design and Implementation

4.1 Problem Description

4.2 Multipath Approach

Chapter 5

Evaluation

Chapter 6

Conclusion

6.1 Summary and Conclusion

6.2 Future Work

Chapter 7

Appendix

Bibliography

[1] Van Jacobson, Diana K. Smetters, James D. Thornton, Michael Plass, Nick Briggs, Rebecca Baynard, *Networking Named Content*, Parc, Paolo Alto, 2009

[2] Dnyanada P. Arjunwadkar *Introduction of NDN with Comparison to Current Interet Architecture based on TCP/IP*, In International Journal of Computer Applications, 2014

[3] Marica Amadeo, Claudia Campolo, Antonella Molinaro, *Forwarding Strategies in Named Data Wireless Ad hoc Networks: Design and Evaluation*, In Journal of Network and Computer Applications, 2014

[4] Giuseppe Rossini, Dario Rossi *Evaluating CCN multi-path interest forwarding strategies*, Technical Report, Telecom ParisTech, 2012

[5] "NFD Developer's Guide", October 04, 2016. [Online]. Available: https://named-data.net/wp-content/uploads/2016/10/ndn-0021-7-nfd-developer-guide.pdf

[6] Spyridon Mastorakis and Alexander Afanasyev and Ilya Moiseenko and Lixia Zhang *"ndnSIM 2: An updateded NDN simulator for NS-3"* Technical Report, NDN, November, 2016

[7] Alexander Afanasyev and Ilya Moiseenko and Lixia Zhang *"ndnSIM: NDN simulator for NS-3"* Technical Report, NDN, October, 2012

[8] "PyViz," https://www.nsnam.org/wiki/PyViz, 2017. [Online]. Available: https://www.nsnam.org/wiki/PyViz