Home Networking
Internet-Draft
Intended status: Informational
Expires: January 4, 2016

K. Jin
Ecole Polytechnique / Cisco
P. Pfister
Cisco
J. Yi
LIX, Ecole Polytechnique
July 3, 2015

Experience and Evaluation of the Distributed Node Consensus Protocol (DNCP) Implementation

draft-jin-homenet-dncp-experience-00

Abstract

The Distributed Node Consensus Protocol (DNCP) is a protocol framework that offers dynamic network topology discovery and data synchronization within a network of participating nodes. This document reports experience with the main DNCP Open-Source implementation ('libdncp', part of 'hnetd') and provides a performance evaluation of this same implementation in a simulated environment.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

The Distributed Node Consensus Protocol (DNCP) is a protocol framework providing dynamic network topology discovery and data synchronization within a network of participating nodes. At the time of writing this document, DNCP is specified in an internet draft [I-D.ietf-homenet-dncp] and in standardization process by the Homenet working group.

DNCP leaves some parameters to be specified by DNCP profiles, which are actual implementable instances of DNCP. Nodes implementing the same DNCP profile, and operating within the same DNCP network, are able share TLV tuples (called Node Data), discover the network topology, and auto-detect arrival and departure of other nodes.

This document reports experience with the main Open-Source implementation of DNCP ('libdncp', part of 'hnetd') and provides a performance evaluation of this same implementation. For the purpose of this document, an early version of libdncp has been used. A newer version has since been published. The results presented in this document only reflect performances of the older version, but will be updated in next iterations of this document. A second DNCP implementation has also been published recently, but has not been evaluated.

DNCP was first specified for home networks, but is also applicable to other networks. The present document therefore presents DNCP performances on topologies that we might not expect in a home network. That is to evaluate the performances of DNCP in the most various situations.

Experiments and measures were made in a simulated environment using the Network Simulator version 3 (NS3). NS3 is a discrete event simulator widely used and recognized by the scientific community.

2. Implementations

At the time of starting this study, the main Open-Source implementation of DNCP was 'hnetd', available online on github (https://github.com/sbyx/hnetd). hnetd is an implementation of the Home Network Control Protocol, HNCP, which includes various elements, such as DNCP, Prefix Assignment, host configuration, and is detailed in [I-D.ietf-homenet-hncp]. At the time being, hnetd is the most complete implementation of HNCP.

A new implementation of DNCP has recently been made available online (http://www.pps.univ-paris-diderot.fr/~jch/private/

shncpd-20150701.tar.gz). We have not evaluated this implementation.

For the purpose of this work, 'hnetd' was modified in order to provide a statically linkable library containing DNCP implementation. This branch is still available on the main 'hnetd' repository in the 'libdncp' branch. Since then, 'hnetd' maintainers have published a new library, libdncp2, which is statically or dynamically linkable and which is based on the same code as the most recent versions of 'hnetd'. As a matter of timing, we could not evaluate libdncp2, but we plan to use it in further updates of this document.

'hnetd', DNCP included, is comprised of 15651 lines of code (18220 when including test files). The binary weights 576KB when compiled for debian X86_64 with no optimization and 727KB when compiled for OpenWrt MIPS. libdncp2 is roughly comprised of 2300 lines of code (2590 when including security option), it weights 193KB when compiled with no optimization for debian x86_64 and 192KB when compiled for OpenWrt MIPS.

3. Simulation Setup

3.1. Simulation Environment

The current dncp implementation relies largely on linux library (for opening sockets, sending and receiving packets..etc) and uses libubox for scheduling events. To integrate dncp into ns3, we have to redefine all the functions in the code that are related to these two parts so that packets can be sent and received in ns3 and events can be scheduled using ns3 scheduler.

We used CSMA model in ns3 to simulate layer one and layer two. CSMA model is designed in the spirit of Ethernet but different from the real-life Ethernet in the sense that the CSMA channel can provide instantaneous carrier sense and priority-based collision avoidance. The channel has three states: TRANSMITTING, PROPAGATING and IDLE, the states can be seen immediately by the devices attached to the channel so collision never happens. CSMA model consists of two parts: CSMA channel and CSMA device. CSMA channel is the model of the transmission medium, and CSMA device is like an Ethernet device, the CSMA devices are connected to the channel.

Listed below are several attributes of the CSMA device that we can configure:

o MTU:The mac level maximum transmission unit, set to 1500

- o Encapsulation Mode: Type of link layer encapsulation to use. In our simulation we use the default mode "Dix" which is commonly used in Ethernet.

- o TxQueue: Type of the transmit queue used by the device. In ns3, we have the possibility to choose from Codel queue, drop tail queue and RED (random early detection) queue. Here we use the drop tail queue and set the buffer of the queue to 100 packets. (bytes can also be used as the maximum queue size metrics)

- o Interframe gap: The pause between two frames, in the simulation we just use 0.

And the attributes of the CSMA channel that we can configure:

- o Data rate: The transmission data rate to be provided to the devices connected to the channel. That is the rate of the device pushing data into the channel. This attribute applies to all the devices on the same channel. In the simulation we set it to 1000Mbps thus providing an infinite throughput to eliminate the impact of throughput on the performance of dncp, in order to calculate the actual throughput consumed.

- o Delay: The speed-of-light propagation delay over the medium. Imagine there is a symmetrical hub that is of equal cable length to all the devices of the channel. When one device sends a packet to another device, the packet fist reaches the hub and is forwarded to the destination device, so the propagation delay is always the same for a given channel. In our simulation, this delay is set to 1 micro second.

3.2. Performance metric

- o Convergence time: The time that dncp takes for the network to converge. We use a concept of converging percentage to represent the converging state, basically the converging percentage is the proportion of the biggest cluster of nodes that share the same network hash. Apparently when this percentage is 100%, the network has coverged.

- o Traffic consumption: The amount of traffic that dncp uses to converge. To evaluate the traffic consumption we count the overall amount of bytes sent during the converging process as well as the throughput per second.

3.3. Chosen toplogies

This section describes the different topologies that have been used for our performances analysis. We picked topoligies which were:

- Deterministic.
- Easily described and generated as a function of the number of nodes (called N).
- Representing different situation ultimatly testing different scalability properties.

3.3.1. Single link topology

The single link topology puts all the nodes on the same link. Each node therefore has a single DNCP End-Point with N-1 neighbors. Such topology is well suited to evaluate DNCP scalability in terms of number of neighbors on a given link.

```
n1          n2       n3       n4
|        |        |        |
----------------------------
```

The single link topology for N=4.

Figure 1

3.3.2. String topology

The string topology chains all nodes one after the other. Each node has two DNCP End-Points with one neighbor on each side (except for the two extremities). Such topology is well suited to evaluate the converge time depending on the diameter of a network, as well as the scalability in terms of number of nodes.

```
        ---------       ---------
        |       |       |       |
n1      n2      n3      n4      n5      n6
|       |       |       |       |       |
---------       ---------       ---------
```

The string topolofy for N=6

Figure 2

3.3.3. Mesh topology

The mesh topology connects all nodes with distinct links. Each node has N-1 DNCP End-Points with one neighor on each side. Such topology is well suited to evaluate DNCP scalability in terms of the amount of nodes and end-points.

3.3.4. Tree topology

The tree topology forms a typical binary tree. More formaly, node i is connected with node 2*i + 1 and 2*i + 2, unless those numbers are greater or equal to N. In such topology, all nodes except the root one have three DNCP End-Points with one neighbor on each. This topology offer a more realistic tradeoff between the diameter and the number of nodes.

3.3.5. Double Tree topology

The double tree topology is identical to the binary tree case, but each node is doubled with a redundancy node. In such topology, all nodes except the two root node have 6 DNCP End-Point with one neighbor on each. This topology also offers a more realistic tradeoff between the network diameter and the number of nodes, but also adds redundancy and loops.

4. Performance Evaluation

4.1. Scenario 1: Link topology of different size

Convergence time

The average value is calculated over 10 experiments

10 nodes	20 nodes	30 nodes	40 nodes	50 nodes	60 nodes	70 nodes	80 nodes
1.84s	3.09s	*4.43s	5.14s	6.53s	*8.61s	11.57s	14.05s

*: the average value is calculated over the results of 9 experiments because the other one diverges too much

Table 1: the average convergence time of link topology

Note that we observed two accidents during the simulation. One happens in one experiment among the 10 that we ran for 30-node network, the network first converges at 4.016s, which is very close to the average convergence time, but at 25.949s this converging state is broken and the network finally reconverges at 26.12s. The other one happens in the case of 60-node network where it fist converges at 7.081s then gets disturbed at 25.822s and comes back to converging at 26.303. As for the the reason of this happening, we will dig deeper into the logs and hope to find an explanation soon.

Verbosity

The first row shows the overall bytes sent in the converging process, the second shows the bytes sent per node, calculated over 10 experiments

10 nodes	20 nodes	30 nodes	40 nodes	50 nodes	60 nodes	70 nodes	80 nodes
85.3KB	604.7KB	2.3MB	5.4MB	11.9MB	23.7MB	51.7MB	88.1MB
----	----	----	----	----	----	----	----
8.5KB	30.2KB	79.6KB	140.7KB	245.KB	404.8KB	757.2KB	1.1MB

Table 2: the traffic of dncp in link topology

With the number of nodes increasing, the traffic grows dramatically. Actually, when we run the simulation of large link network (more than 60 nodes) with limited data rate (12Mbps) and larger delay (6us), the network does not converge at all. Part of the reason may be that in the current implementation, dncp sends a packet for every node state request and every node state reply instead of wrapping all the requests or replies together in one packet. So when there are many nodes in the network, at certain moment, a node may send tons of requests or replies, thus congesting the device buffer and causing a lot of packets loss.

4.2. Scenario 2: String topology of different size

Convergence time

The average value is calculated over 10 experiments

10 nodes	20 nodes	30 nodes	40 nodes	50 nodes	60 nodes	70 nodes	80 nodes
1.84s	3.65s	5.24s	7.09s	8.79s	11.11s	12.87s	15.03s

Table 3: the average convergence time of string topology

If we plot the average converging time against the number of nodes, it is discernible that the graph is leaner. This result is exactly the same as we expected. Because the convergence time should be proportional to the diameter of the network.

Verbosity

The first row shows the overall bytes sent in the converging process, the second shows the bytes sent per node, calculated over 10 experiments

10 nodes	20 nodes	30 nodes	40 nodes	50 nodes	60 nodes	70 nodes	80 nodes
51.5KB	243.4KB	605KB	1.2MB	2MB	3MB	4.1MB	5.6MB
----	----	----	----	----	----	----	----
5.1KB	12.2KB	20.1KB	30.9KB	40.5KB	50.4KB	59.2KB	70.1KB

Table 4: the traffic of dncp in string topology

The average traffic sent per nodes is almost linear against the number of nodes, since for string topology, the convergence time is also linear against the number of nodes, we can deduce that the average traffic sent per nodes per second is almost a constant value irrelevant to the network size.

4.3. Scenario 3: Mesh topology of different size

Convergence time

The average value is calculated over 10 experiments

10 nodes	20 nodes	30 nodes	40 nodes	50 nodes	60 nodes	70 nodes	80 nodes
1.71 s	3.2s	4.83s	*6.19s	10.64s	13.02s	15.33s	17.93s

*: the average value is calculated over the results of 9 experiments because the other one diverges too much

Table 5: the average convergence time of mesh topology

The converging time grows faster after 50 nodes, the possible reason is DNCP fragmentation limit

Verbosity

The first row shows the overall bytes sent in the converging process, the second shows the bytes sent per node, calculated over 10 experiments

10 nodes	20 nodes	30 nodes	40 nodes	50 nodes	60 nodes	70 nodes	80 nodes
202.7 KB	1.6MB	6.6MB	18.1MB	49.1MB	95.8MB	167.4MB	271.9MB
----	----	----	----	----	----	----	----
20.3KB	83.5 KB	222.1KB	453.8KB	983.1KB	1.6MB	2.4MB	3.4MB

Table 6: the traffic of dncp in mesh topology

4.4. Scenario 4: Tree topology of different size

Convergence time

The average value is calculated over 10 experiments

10 nodes	20 nodes	30 nodes	40 nodes	50 nodes	60 nodes	70 nodes	80 nodes
1.16s	1.57s	1.86s	2s	2.33s	2.42s	2.56s	2.6s

Table 7: the average convergence time of tree topology

With the number of nodes increasing, the time used to converge grows more and more slowly. The difference between 60-node tree and 70-node tree, 70-node tree and 80-node tree is only about 0.1 seconds, the network converges quite fast.

Verbosity

The first row shows the overall bytes sent in the converging process, the second shows the bytes sent per node, calculated over 10 experiments

10 nodes	20 nodes	30 nodes	40 nodes	50 nodes	60 nodes	70 nodes	80 nodes
40.7KB	166.7KB	374KB	644.5KB	1MB	1.3MB	1.9MB	2.4MB
----	----	----	----	----	----	----	----
4.1KB	8.3KB	12.4KB	16.1KB	20.2KB	22.8KB	26.7KB	29.9KB

Table 8: the traffic of dncp in tree topology

4.5. Scenario 5: Double tree topology of different size

Convergence time

The average value is calculated over 10 experiments

10 nodes	20 nodes	30 nodes	40 nodes	50 nodes	60 nodes	70 nodes	80 nodes
1.04s	1.44s	1.5s	1.7s	1.96s	1.98s	2.06s	2.09s

Table 9: the average convergence time of double tree topology

The situation is similar to that of tree topology, and it converges even faster.

Verbosity

The first row shows the overall bytes sent in the converging process, the second shows the bytes sent per node, calculated over 10 experiments

10 nodes	20 nodes	30 nodes	40 nodes	50 nodes	60 nodes	70 nodes	80 nodes
66.9KB	265KB	605.1KB	1MB	1.5MB	2MB	2.8MB	3.5MB
----	----	----	----	----	----	----	----
6.7KB	13.2KB	20.2KB	25.3KB	30.8KB	33.2KB	39.7KB	44.7KB

Table 10: the traffic of dncp in double tree topology

5. Conclusion

The convergence time is proportional to the network diameter.A good example is string network where the convergence time is linear to the network size. It is also interesting to look at the tree topology, a tree-network of 80 nodes only takes 2.6 seconds to converge because its diameter is only 6.

We believe another factor affecting convergence time is the average number of neighbors of a node in the network. The diameters of mesh and link network are also very small but they converges really slowly, because for a node in such network, the number of neighbors is n-1 (assuming the network is of size n). As mentioned above, a tree network of 80 nodes converges at 2.6s, while a string network of

7 nodes converges at about 1.6s (They are of the same diameter).That is probably because in string network a node only has 2 neighbors but in tree network it has 3.

It is obvious from the results that dncp is well suited for tree-like topologies. It is logic because this kind of topologies has small diameters. And the bigger the network is, the tree takes more nodes to gain one depth, which explains why the convergence time of tree and double tree grow more and more slowly as number of nodes increases.

The amount of traffic depends on the average number of neighbors as well as the size of the network.

Another interesting point to note is that from about 50 nodes, link and mesh topology change the pace of growing in convergence time. We suppose it is due to dncp fragmentation.

In conclusion, dncp can provide very good convergence time at a low traffic price in a proper topology.

6. Informative References

[I-D.ietf-homenet-dncp]
Stenberg, M. and S. Barth, "Distributed Node Consensus Protocol", draft-ietf-homenet-dncp-06 (work in progress), June 2015.

[I-D.ietf-homenet-hncp]
Stenberg, M., Barth, S., and P. Pfister, "Home Networking Control Protocol", draft-ietf-homenet-hncp-06 (work in progress), June 2015.

Authors' Addresses

Kaiwen Jin
Ecole Polytechnique / Cisco
France

Phone:
Email:
URI:

Pierre Pfister
Cisco
France

Phone:
Email:
URI:

Jiazi Yi
LIX, Ecole Polytechnique
France

Phone:
Email:
URI: