Homenet Working Group
Internet-Draft
Intended status: Informational
Expires: January 7, 2016

A. Augustin
Cisco Systems

July 6, 2015

DNCP Use Case in a Distributed Cache System
draft-augustin-homenet-dncp-use-case-00

Abstract

In a distributed cache system, at some point the nodes need to share and synchronise some information. This document explains how this cache system works and shows how DNCP is useful in this situation.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 7, 2016.

Table of Contents

1. Introduction

The cache system described in this document works under the assumption that each chunk of data that may be cached is uniquely associated to an IPv6 address. When a client desires to retrieve this chunk of data, it establishes a TCP connection towards the IPv6 address of the chunk. The system only relies only on the use of IPv6 and TCP, it is therefore independant of the application used on top of it (usually HTTP). This connection is routed through a proxy, which acts as an entry point in the cache system. The proxy inserts a Segment Routing Header in each packet which destination is the IPv6 address of a chunk, in order to have these packets sequentially routed through a certain number of cache servers. Cache servers intercept the packets to reply directly to clients if they have the requested chunk available locally (Segment Routing interception).

2. Terminology and Abbreviations

The terminology and abbreviations used in this document are described in this section.

- DNCP: The Distributed Node Consensus Protocol, as defined in [I-D.ietf-homenet-dncp]

- SRH: IPv6 Segment Routing Header, defined in [I-D.previdi-6man-segment-routing-header]

- SR List: The list of addresses included in a SRH specifying the different destinations a packet will go through before reaching its final destination.

- Chunk: A blob of binary data, which is associated to a unique IPv6 address, and which a client may request.

- Client: A host that requests a chunk by establishing a connection towards the associated IPv6 address.

- Proxy: A node that insert SRHs in the packets corresponding to client requests in order to have them routed to cache servers.

- Cache server: A node that may have a certain number of chunks available. Upon receiving packets with a segment routing header, a cache server can either intercept the packets and reply directly if it has the requested chunk, or forward it to the next hop in the SRH.

- Source server: A node that has a given list of chunks available.

3. Context

The distributed cache system is composed of one or more proxies, cache servers, and source servers.

When a client needs a chunk of data, it establishes a TCP connection towards the IPv6 address identifying the chunk. This TCP connection is routed through a proxy which allows to keep compatibility with existing clients. Depending on the content being requested, which is known by looking at the destination address of the packet, the proxy chooses a SR List. The SR List contains, in this order, the addresses of several cache servers, the address of a source server that has this content, and the address of the chunk. This SR List is converted to a SRH and inserted in the packet. No information is retained per connection, this operation is applied to each packet individually.

The cache servers maintain a list of the chunks that they have cached and that are available locally. When receiving a packet, they look at the last address of the SRH. If the corresponding chunk is available, they intercept the packet and deliver it locally. If the chunk is not available, they forward it according to the SRH. This method ensures that the TCP connection is established with the first server in the SR List that has the requested chunk.

Each cache server monitors the requests that it receives by counting the TCP SYN packets and looking at the last address in the SRH. It caches the most requested chunks by using a caching algorithm, such as Least Frequently Used or Least Recently Used. This implies that, in order to be efficient, a given proxy should always use the same SR List for a given chunk.

From an other point of view, for a given SR List, the first server on the list caches as many chunks as possible among the most requested ones, the second server caches the most asked chunks that have not been cached by the first server, and so on. In order to minimize the distance that most requests cover in the network, proxies should

generate SR Lists that are dependant on the underlying topology: close servers should be put first in the SR List, then servers further away, and eventually a source server. Proxies should also balance the load between adjacent servers (clustering), stop using servers that are unavailable, and automatically start using new servers that become available.

As the servers are passive, the efficiency of the system will be dependant on how smartly the proxies generate SR Lists. The exact logic of the proxy is out of scope of this document, but the proxies don't need a lot of information to generate good SR Lists. They can deduce many things from:

- The list of proxies, cache servers and source servers that are available.

- The distances between all servers in the system, which can be measured from several metrics (for instance the RTT).

4. Interest of using DNCP

In this context, DNCP allows to extremely simply share the data needed (the list of nodes and the distances between them) across all nodes in the system, without reimplementing a full-fledged protocol. DNCP ensures that the data is synchronised between nodes and that data from unreachable nodes is removed.

The distance is measured at startup and is not republished unless it changes significantly. Consequently, the data only changes when a server becomes available or unavailable, or when something changes in the underlying network. This should not happen too often, and so the DNCP Network State Hash should not change too often, which will limit DNCP excitation.

Without DNCP, we would have needed to design a whole protocol with hello messages, keepalives, a flooding mechanism, and so on. DNCP manages all this automatically, all we had to do was to define which data had to be shared and adjust the profile for our needs (faster dead neighbor detection, unicast UDP, etc.).

5. DNCP Profile

This section describes the particular DNCP Profile used by the nodes in this system.

5.1. DNCP Constants and parameters

- The unicast transport protocol is UDP, no multicast transport protocol is used.
- The transport protocol is not secured.
- The UDP port used is 16255.
- Upon node identifier collision, both nodes will choose a new random node identifier.
- Imin = 0.1s, Imax=20s, K=1
- The non cryptographic hash function used is MD5.
- DNCP_NODE_IDENTIFIER_LENGTH = 4
- DNCP_GRACE_INTERVAL = 30s
- Keepalives are enabled, DNCP_KEEPALIVE_INTERVAL = 5s, DNCP_KEEPALIVE_MULTIPLIER = 2.1

5.2. DCNP TLVs used

5.2.1. Server Announce TLV

This TLV is published when a server is ready to join the system. It includes the node type: cache, source server or proxy; and the node's IPv6 addresses.

5.2.2. Server Distance TLV

Whenever a node receives a new Server Announce TLV, it measures the distance to the new server (for instance by pinging it), and then publish a Server Distance TLV. This TLV includes the distance and the public IPv6 of the pinged server.

5.3. DNCP Setup

Because this system is to be inserted in a ISP network, we do not assume that a multicast routing protocol is running. Because of this, we use only unicast communications.

The system includes one or more nodes called "DNCP Relays" that have fixed addresses. Other nodes are configured with the addresses of these relays and contact them in unicast on startup. When they are ready, the nodes publish their Server Announce TLV, and start sending Server Distance TLVs for each Server Announce TLV they receive.

By extracting the data from DNCP, the proxies can get an idea of where each server is, and optimize the SR Lists they generate accordingly. Moreover, as they know that other proxies are using exactly the same data, they can know what their decisions will be and collaborate to a certain extent without exchanging additionnal data.

6. IANA Considerations

This draft includes no requests to IANA.

7. Security Considerations

No security considerations.

8. References

8.1. Normative References

[I-D.ietf-homenet-dncp]
Stenberg, M. and S. Barth, "Distributed Node Consensus Protocol", draft-ietf-homenet-dncp-05 (work in progress), June 2015.

8.2. Informative references

[I-D.previdi-6man-segment-routing-header]
Previdi, S., Filsfils, C., Field, B., and I. Leung, "IPv6 Segment Routing Header (SRH)", draft-previdi-6man-segment-routing-header-06 (work in progress), May 2015.

Author's Address

Aloys Augustin
Cisco Systems

Email: aloys.augustin@polytechnique.org