nCDN: CDN Enhanced with NDN

Xiaoke Jiang Jun Bi
Institute for Network Sciences and Cyberspace, Tsinghua University
Department of Computer Science and Technology, Tsinghua University
Tsinghua National Laboratory for Information Science and Technology
jiangxk10@mails.tsinghua.edu.cn junbi@tsinghua.edu.cn

***Abstract*—Content Delivery Network (CDN) improves large scale data delivery with widely distributed data replicas; But the fundamental goal of IP is to connect two hosts. As a consequence, request routing, which selects the best server to serve the requested data, is introduced to meet the mismatch between CDN and IP. In contrast to IP, Named Data Networking (NDN) makes content the first-class citizen of the network. Its specialties, such as multicast, content multihoming, cache and content-oriented security, are designed for large scale data delivery. Due to the essential consistency between CDN and NDN, we propose Named Content Delivery Network, or nCDN, which embeds NDN into existing CDN framework to simplify the implementation and improve the efficiency. nCDN supports existing running CDN infrastructure, by setting up NDN over UDP/TCP. NDN takes charge of request routing and content delivery only; While other components of CDN, such as billing, accounting, data analysis, data management etc, remain changeless.**

The advantages of nCDN include: 1) As NDN's routing plane holds content distribution information, requests are routed to the best data copies straightforward. 2) NDN's stateful forwarding plane detects the network state in real time, and responses to congestion, link or node failure quickly. 3) NDN naturally supports multicast and content multihoming. Multicast eliminates identical requests, while content multihoming fully utilizes redundant resources, such as bandwidth and storage.

***Index Terms*—NDN, CDN, Content Delivery Network**

I. Introduction

CDN makes use of geographical distributed data replicas servers, which are called surrogates or edge servers, to cover the large amount of concurrent request from all over the world, and it focuses on delivering requested data to users, no matter where the data comes from; but the fundamental goal of IP is to connect two hosts. The essential mismatching leads to complexity and inefficiency. Data transmission in CDN is split into two steps: request routing to select the optimal server, and content delivery on top of UDP/TCP's end-to-end channel. The drawbacks include:

- Content delivery begins only after request routing finishes successfully, As a consequence, delay is introduced.
- Since TCP/UDP just provides basic end-to-end service, almost all the components, including content to container mapping, real-time detecting, monitoring, traffic engineering, are implemented on application layer. Therefore, the CDN system becomes complex.

In contrary to IP, NDN focuses on data itself and ignores the data container. NDN offers a lot of important elements that CDN desires, for example, named contents, cacheable nodes, and multicast. What is more, NDN's routing plane holds information of content distribution and stateful forwarding plane detects and adapts to dynamic of the Internet. The intrinsic consistency makes NDN a perfect underlying component for CDN:

- Name-based routing directs requests to the optimal server straightforward; Content multihoming, denoting that content is provided on multiple nodes, is naturally supported, redundancy resources are fully utilized directly.
- Stateful forwarding plane detects the network state in real time, network congestion, link or node failure could be covered by the network itself.
- Multicast eliminates repeated requests, bandwidth and storage are saved; And the CDN is able to serve more requests.

We propose to enhance current CDN framework with NDN, dubbed Named Content Delivery Network (nCDN). Considering that IP is the de facto dominator of the Internet, nCDN runs on UDP/TCP. NDN takes charge of request routing and content delivery. And CDN focuses on business components, such as accounting, data analysis, etc, which are inherited from existing framework.

NDN could replace existing request routing and content delivery subsystems, while a conservative deployment solution for existing running CDN is to treat nCDN as a alternative parallel subsystem of existing infrastructure, and let nCDN serve the latest popular contents only.

The simulation demonstrates that nCDN is better than traditional CDN on almost all the aspects, including the scalability, reliability, and QoS. Furthermore, nCDN also helps to improve security and solve CDN Interconnecting.

This paper is organized as follows. First, backgrounds on CDN and NDN are listed in Section II. nCDN architecture, including naming, request routing, content delivery, deployment, is introduced in Section III. Performance is evaluated in Section IV. Finally, we conclude the paper in Section V.

II. Background

A. Content Delivery Network

To fully introduce CDN, here we give a general framework of CDN and a insight into a specific implementation, YouTube CDN, respectively.

1) CDN Design: In the context of CDN, content consists of two main parts: the encoded media and metadata. The encoded media includes static, dynamic and continuous media data, such as audio, video, documents, images and web pages. While metadata is the content identifier that is used to content identification, discovery and management.

There are some key techniques in the context of CDN, including content distribution and management, request-routing, performance measurement [1].

Content distribution and management include surrogate placement, cached content selection and delivery, content outsourcing and cache organization. request-routing includes request-routing algorithm and request-routing mechanism. Performance measurement is in charge of measuring internal and external network state.

Request-routing algorithm encompasses adaptive and non-adaptive algorithms. Adaptive algorithms takes the real-time system conditions into consideration while non-adaptive ones make use of heuristics. Non-adaptive algorithms are simple but not very accurate, especially in the face of events like flash crowds [2]; while adaptive algorithms are complex and robustness.

Request-routing mechanism includes Global Server Load Balancing (GSLB) [3], DNS-based request routing [4], HTTP redirection, URL rewriting etc. GSLB and DNS-based request routing resolve domain name into numerical IP address of best surrogate. HTTP redirection leads to lack of user transparency problem and overhead. URL rewriting leads to delay for URL parsing, the possible bottleneck and non-cachebility of the contents. In reality, HTTP redirection, DNS-based request-routing and GSLB are the most widely used request-routing mechanisms.

Mapping system is the traditional CDN's global traffic director by providing information to request routing: it uses histroic and real-time data about the health of both the CDN network and the Internet at large in order to create maps that are used to direct traffic on the CDN network in a reliable, efficient, and high performance manner [5].

2) Youtube CDN Reverse Engineering: Since the detailed implementation of CDNs are kept secret by CDN providers. Researchers do reverse engineering to explore specific CDN design and solution.

Given the traffic volume, geographical span and scale of operations, the design of YouTube's content delivery infrastructure is perhaps one of the most challenging engineering tasks. YouTube video delivery could represents an example of the "best practices" in the design of such planet-scale systems. According to [6], the YouTube video delivery system consists of three major components: a "flat" video id space, multiple DNS namespaces reflecting a multi-layered logical organization of video servers, and a 3-tier physical cache hierarchy, i.e., primary, secondary and tertiary. By mapping the video id space to the logical servers via a fixed hashing and cleverly leveraging DNS and HTTP redirection mechanisms.

YouTube uses HTTP based redirection to achieve dynamic load-balancing. For instance, if the primary server responsible for a video cannot serve the the requested video, it sends a HTTP 302 response back to the client. The response tells the client to go to a secondary server to download the video from. The secondary server also response with the same logic.

The whole request routing process follows a very specific namespace hierarchy. A primary video cache server may redirect a video request to a corresponding secondary cache server, or directly to a corresponding tertiary server. Secondary server ony redirect a request to a tertiary server. And if all the cache servers cannot serve the request video, then it simply fails with a HTTP 503 error code.

B. Named Data Networking

NDN[7] or Content-Centric Networking[8] is one of the most promising future Internet architecture. NDN replaces *where* with *what*. Packets carry the *name* of the data instead of source and destination addresses.

There are two kinds of datagrams in NDN architecture: Interest and Data. End users send Interest which contains the name of the Data they want, NDN routers forward Interests by their names, which is called name-based routing, and create Pending Interest Table (PIT) entries to record the incoming face and outcoming face. If the Interest meets its corresponding Data, the Data follows the path marked by PIT entries back to the originator of an Interest. In addition, NDN contains Content Store (CS) to cache data. NDN decouples requests and responses in both time and space: data requester and producers need not to known each other's location, nor need they be online at the same time [9].

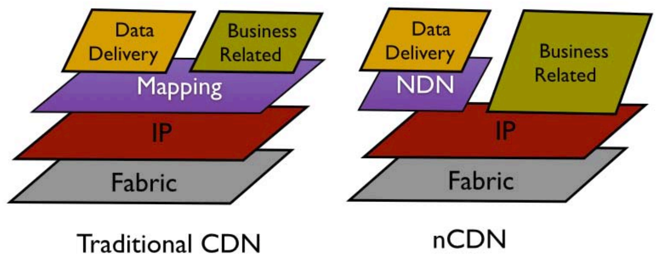
Cheng Yi et al. gives in-depth analysis on forwarding of NDN [10], which is a highlight (also challenge) of NDN. After forwarding Interest packet, NDN routers maintain the state of every pending Interests. Thus, routers are aware of the network state by observing the two-way traffic. In this case, router could score different paths at the granularity of name prefix according to historical data. Furthermore, Immediate routers can explore multiple alternative paths to save congestion, node or link failure, under the guidance of forwarding strategy without leaving the task to the end consumer.

III. nCDN Architecture

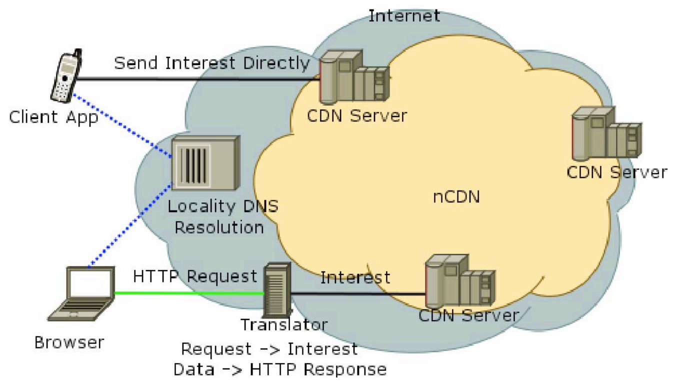
Since NDN takes charge of request routing and content delivery only, we focus on naming, request routing, content delivery and deployment only in this section. Other components can be inherited from existing framework.

As shown in Figure 1a, the most significant change is that nCDN does not rely on mapping to translate what to where; Instead, data delivery is built on NDN. In addition, business related subsystems are build directly on IP, since those subsystems contains a lot of features and functionalities.

In nCDN, servers are connected via NDN. NLSR[11], Named-data Link State routing protocol, is used to create routing table. Thus, the CDN build a pure NDN overlay network. CDN servers announce name prefix of the content which they serve into the routing system, NLSR naturally supports content multihoming. The Name-based routing systems keeps



(a) Architecture: nCDN does not rely on mapping



(b) Infrastructure Deployment

Fig. 1: Architecture Design. nCDN does not rely on mapping system as traditional CDN does. Surrogates make up the pure NDN network. All the requests are encapsulated in Interest and transmitted in the network.

the content distribution information and request routing is overlap with routing in nCDN.

At the end hosts, almost all the Content Service Providers (CSPs), such as YouTube, Facebook, Google, publish their client applications on different platforms, including mobile devices and PC. As shown in Figure 1b, the client applications connect to the nearest server, named "entry point", and send Interest directly into nCDN. For general situation, browser send HTTP requests to the translator, which translate HTTP request to Interest, then the translator send Interest into nCDN. Our previous work on the translation between HTTP request and Interest has successfully deployed in a similar scenario[12].

Locality aware DNS resolution[6], which is widely adopted mechanism, could direct users to the best/nearest translator.

A. *Naming*

Metadata, in the context of CDN, plays key roles in identification, discovery, and management. These functions are exactly what *Name* does in the context of NDN. Therefore, naming, the fundamental requirement of NDN is inherently satisfied by existing architecture. For the sake of simplicity, we use metadata to construct the name of contents. Furthermore, content multihoming is built-in, different servers could announce identical name prefix. And naming may have different schemes, for example, by geographical position, by date, or by subject. For example, youtube.com/us/tx/id134.flv/_s0 refers to the first segment of a video file, whose unique id (inherited here metadata) is 134. This file belongs to youtube.com and it seems very popular in Texas. Note that routing announcement is at the granularity of name prefix, instead of single content.

B. *Request Routing v.s. Content Routing*

In traditional CDN, request routing aims to find the best surrogate. There are two manners to use name-based routing:

1) Interest is used to request the optimal server's IP address. After the client get the Data which contains the answer, it sends the normal HTTP request to the server. In this manner, name-based routing becomes a new mechanism to take over request routing.
2) Interest is used to request the data directly. In this manner, request routing is unnecessary, since content name is routable.

In nCDN, the second manner is selected as the basic services for the following reasons:

- NDN overlay network could provides better QoS, such as latency, jitter, and loss rate, by utilizing NDN's features.
- The identical Interests are aggregated, which get rid of waste of resource.

However, the first manner is alternative as a way to relieve traffic pressure if it is necessary, especially when the the nCDN network suffers from heavy traffic.

In traditional CDN, goals such as load balancing, traffic engineering or lowering cost, etc, are integrated into routing-request algorithm, which attracts a lot of research attention. In nCDN, we can achieve those goals with NDN's routing and forwarding strategy. By customizing routing priority and forwarding strategy, requests are routed to the "optimal" server according to expected goals.

C. *High-Performance Content Delivery*

In this section, we demonstrate nCDN's high performance content delivery with a toy case.

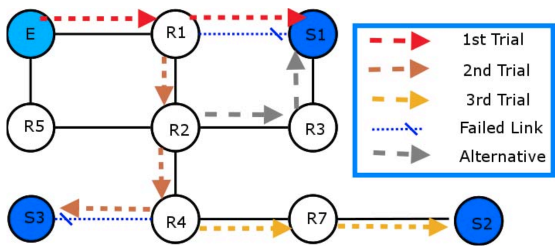


Fig. 2: Toy Case: NDN saves packet loss by immediate routers

In Figure 2, the circles represents nCDN surrogates. The node E is the entry point of a specified Interest. Nodes S1, S2, and S3 are surrogates which can originally satisfy the Interest. We assume that the routing metric and delay of each link is the same. In this case, Link R1-S1and R4-S3 fails to get Data back, possibly due to congestion, link or node failure. By default, NDN nodes try different alternative surrogates one by one. In addition, we also assume Interest is not satisfied by cached data and the corresponding PIT entries does not timeout during the case.

The following steps show the process of content routing and delivery:

Step 1. E chooses R1 as its first tiral, because the minimized path cost of choosing R1 is 2 (E-R1-S1) while that of R5 is 4.

Step 2. From view of R1, S1 is its next best hop. So Interest packet is forwarded to S1.

Step 3. Due to link R1-S1 failure, R1 tries another path and forwards the Interest to R2. This is the second trial.

Step 4. There are two best choices for R2: R2-R4-S3 and R2-R3-S1.In this case, we assume R2 tries path R2-R4-S3 first by random, thus, Interest is sent to R4. However, in fact, R2 forwards Interest to R3 is the best choice in our case, since link R4-S3 fails.

Step 5. Since link R4-S3 fails, Interest is forwarded to R7, then to S2. S2 provides Data, which follows the reverse path S2-R7-R4-R2-R1-E.

The path E-R1-R2-R4-R7-S2 is not the optimal one. Forwarding strategy shown in the case is quite simple but inadvisable. It could be customized to improve the efficiency. For example, R2 tries R3 first instead of R2 according to the historical performance; Or R2 triggers another trial to R3 after waiting response from R4 for round-trip delay between R2 and S3; Or R2 tries R4 and R3 simultaneously, and select the path which returns Data faster, i.e. R3-S1. Despite of the hop-by-hop control, NDN's forwarding plane could remembers the previous choice and score different paths, which means only the first Interest after failure (the one shown in our case) has to explore different paths, while the following similar Interests directly follow the right path without spending time and resources on trials. Since all these are done on data plane, nCDN responses to the congestion, link or node failure very quickly and efficient.

Comparing to IP's end-to-end control, the hop-by-hop control is more flexible and faster.

Another advantages is that, the same Interest can be aggregated on every router. For example, if R1 also receives identical Interest from other entry point, this Interest will be aggregated with Interest coming from E, R1 will not forward twice.

NDN does well in large scale data delivery due to its delicate design, almost all of its features, such as multimust, content multihoming, hop-by-hop congestion control, show their place on content delivery.

D. Deployment

At current stage, IP is the de facto dominator of the Internet. NDN can run on top of any layer-2 technology or above, including IP, UDP and TCP. The existing current prototype, CCNx/NDNx can run on UDP or TCP. CDN surrogates are connected via NDN over UDP/TCP. What's more, Implementing the strategy customization with software is very flexible and quick.

In addition, nCDN can replace the whole existing request routing and content delivery subsystems. Another conservative deployment manner for existing running CDN is to treat nCDN as a alternative parallel system of existing infrastructure, and let it serve requests for latest popular contents only.

Since CDN is expensive service, the amount of content in the CDN is very limited comparing to the whole Internet. With special care on naming and placing contents, amount of routing entries can be limited in a reasonable range.

In traditional CDN, surrogates and contents placement are limited by the mapping system and manual configuration. However, in nCDN, NLSR could generate the routing table no matter how the surrogates and contents are placed. Therefore, surrogates and contents placement are more flexible, and management is more user-friendly.

E. Discussion on Security and CDNi in nCDN

Content is stored in several nodes in CDN, if security is bound with data channel or end-point, security turns out complicated. For example, data secured by HTTPs or IPSec cannot be pre-encrypted, since it is stored in multiple nodes and requested by multiple end users. Content-oriented security separates the channel or end-point from security, thus, data is encrypted once and can be stored/cached on any server and transmitted to anywhere.

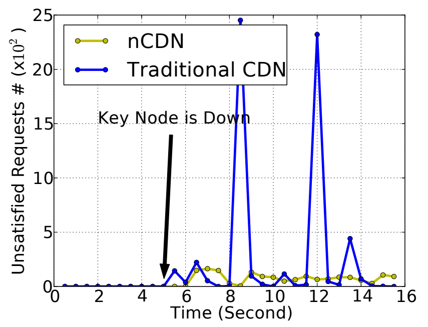
Access control is another important topics. Traditional mechanism does not work here, since the producer does not know the consumer's IP. There are two strategies in nCDN: 1) Name prefix announcement is limited in the legally accessible area. This solution works for those contents with geographical restriction. 2) Access control is implemented by key exchange, and only authorized clients can fetch the key.

CDN interconnecting (CDNi) is a hot topic. It aims to interconnect separately administered CDNs in support of the end-to-end delivery of content from CSPs through multiple CDNs and ultimately to end users. In the context of nCDN, CDNi is translated to inter-domain routing. Core IP routing protocols, BGP, can be used as-is to deploy in NDN [7] to solve CDNi.

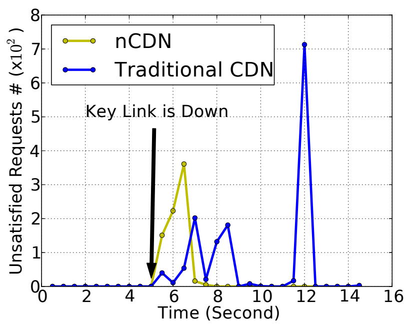
IV. EVALUATION

CDN is designed for scalability, reliability and performance [5]. We evaluate these features with ndnSIM[13]. A commercial CDN is a very large distributed system which may consist of thousands of globally deployed servers that run sophisticated algorithms to enable the delivery of highly scalable distributed applications. We have tried our best to make our simulation closest to the reality. But due to lack of public CDN topology, we use Rocketfuel's AT&T Point of Presence topology [14], which contains 625 nodes, including 221 backbone routers, 108 gateway routers and 296 leaf routers, and 2,101 links with bandwidth, OSPF metric, transmission delay and queue capacity.

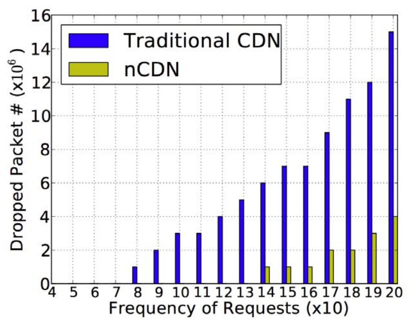
The dataset of CDN comes from a latest sigcomm paper[15]. The dataset contains more than 100,000 objects, whose request frequency distribution follows the zipf law,



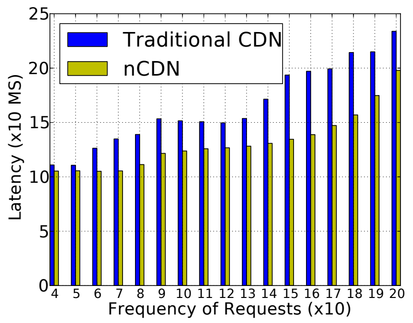
(a) Node Failure: NDN's stateful forwarding plane saves the failure without leading to unsatisfied request



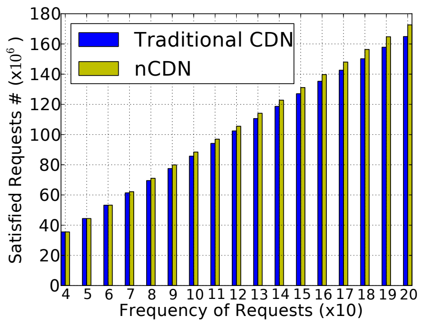
(b) Link Failure



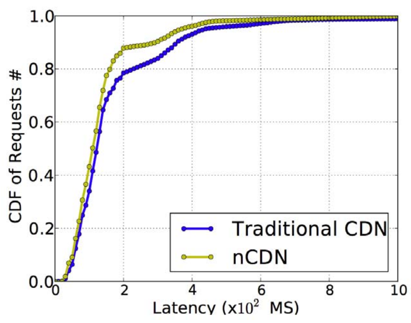
(c) Packet Loss: Packet Loss is saved by forwarding plane in nCDN



(d) Latency: nCDN adapts to the network state and chooses the best path dynamically



(e) Goodput: Due to identical limitation of bandwidth during the simulation, the gap is not significant



(f) Jitter: latency of 90% of requests in nCDN is less than 200ms, 80% in traditional CDN

Fig. 3: QoS and Reliability

with parameters 0.99 in US, 0.92 in Europe and 1.04 in Asia, respectively. We choose 100,000 objects and 0.99 as our zipf parameter.

We randomly pick some gateway routers to serve as surrogates that can originally provide the data in our simulation. All leaf routers serve as entry points.

nCDN is built with a coloring-based forwarding strategy[10] and NDN routers try available ranked faces one by one. Traditional CDN is built with a GSLB, which selects nearest surrogate which can serve the request. If a request from an entry point fails to get data back, GSLB will select an alternative surrogate for the entry point in traditional CDN scenario.

A. Reliability

In this scenario, we measure reliability by amount of unsatisfied requests after node or link failure. Suppose that node or link fail events are found and solved by CDN within 10 seconds. So we monitor the performance during the 10 seconds after failure. In the experiment, after 5 seconds of warming up, we let a key node or link go down. Figure3a and Figure 3b show the unsatisfied request number after failure. As we can find from Figure 3a, traditional CDN line shows instability and some extraordinary peaks (at 8.5th and 12th second). nCDN line just waves at a low quantity. The link failure shows almost the same trend. This result agrees to our conclusion in Section III-C, nCDN tries different paths hop-by-hop and saves the failure with its stateful data plane. However, traditional CDN makes decisions by entry points and selects a new surrogate after a round-trip-delay. All the requests sent during the round-trip-delay get no contents back. Thus, the unsatisfied requests will be gathered and form peak points. During these times, all the requests distributed to the node or link will be influenced.

B. QoS

We explore QoS by checking goodput, packet loss, latency and jitter.

As shown in Figure 3c. nCDN does not lose packet until Interest sending frequency reaches 140 per second. Furthermore, the number of lost packets is significantly less than that of traditional CDN. Average latency of nCDN is lower than that of traditional CDN, especially when the network traffic is heavy as shown in Figure 3d. This is because NDN adaptively forwards requests to the most suitable links and surrogates dynamically. In Figure 3e, we measure the goodput by number of satisfied requests. The goodput of nCDN is always slightly higher than that of traditional CDN. However, due to the identical limitation of link during the simulation, the gap is not very significant.

The above three figures share a same feature: when the network traffic is light, performances between nCDN and traditional CDN are very close. However, when the network traffic is heavy, nCDN performs better. This result shows that nCDN can carry heavier load pressure. nCDN and traditional CDN have the best forwarding choice when network traffic is light, since there is no congestion, nodes just pick the shortest path. However, when congestion happens, traditional CDN relies on entry points to adjust routing path, while nCDN can detect the congestion in time and adjust forwarding path accurately and dynamically.

Jitter is also an important factor of QoS, as we can find in Figure 3f (here frequency of requests is 150 per second and retransmission is enabled). 90% of the requests in our nCDN scenario had latency less than 200ms, while the ratio decreases to 80% in traditional CDN scenario. When we want to meet 96% of requests, in the nCDN scenario the latency to achieve this is less than 400ms, while the latency is increased to 500 ms in traditional CDN scenario.

C. Scalability

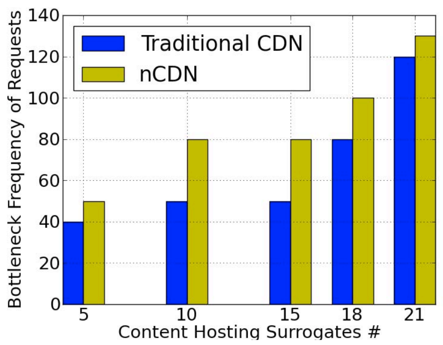


Fig. 4: Scalability: Bottleneck goodput of nCDN is greater than traditional CDN under all different surrogates set

We measure scalability by bottleneck goodput, i.e., frequency of consumer sending requests when the packet loss rate reach a threshold (10 per second). For a given set of surrogates, We increase the frequency of request by 10 every time until the number of unsatisfied requests reaches the bottleneck. We measure the bottleneck goodput under different sets of surrogates.The result is shown in Figure 4.

For every specified set of surrogates, the bottleneck of nCDN is higher than that of traditional CDN. The bottleneck goodput is lower than linear growth with the number of surrogates, this is due to the limitation of other resources, such as link and queue capacity.

V. CONCLUSIONS

In this paper, we propose nCDN which embeds NDN to deliver data in existing CDN. NDN's delicate features greatly improve CDN on large scale data delivery:

- Data delivery is no longer rely on mapping or request routing. Name-based routing directs the requests to the content source straightforward;
- Stateful forwarding plane adapts to the network state in real time, and saves packet loss caused by congestion, node or link failure quickly;
- Multicast eliminates identical requests, while content multihoming fully utilizes redundant resources.
- Content-oriented security and Inter-domain routing also helps to improve security and solve CDN Interconnecting.

According to our simulation, nCDN provides better reliability, QoS, and scalability, compared to tradition CDN, especially in dynamic network state. Furthermore, it also can handle heavier load pressure than traditional CDN. In summary, NDN's advantage on large scale delivery is fully manifested in CDN scenario.

VI. ACKNOWLEDGEMENT

This work is supported by the National High-tech R&D Program ("863" Program) of China(No.2013AA010605), the National Science Foundation of China (No.61161140454), and National Science & Technology Pillar Program of China (No.2012BAH01B01). Jun Bi is the corresponding author.

REFERENCES

[1] Al-Mukaddim Khan Pathan and Rajkumar Buyya. A taxonomy and survey of content delivery networks. *Grid Computing and Distributed Systems Laboratory, University of Melbourne, Technical Report*, 2007.

[2] Limin Wang, Vivek Pai, and Larry Peterson. The effectiveness of request redirection on cdn robustness. *ACM SIGOPS Operating Systems Review*, 36(SI):345–360, 2002.

[3] Markus Hofmann and Leland R Beaumont. *Content networking: architecture, protocols, and practice*. Morgan Kaufmann, 2005.

[4] Balachander Krishnamurthy, Craig Wills, and Yin Zhang. On the use and performance of content distribution networks. In *Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*, pages 169–182. ACM, 2001.

[5] Erik Nygren, Ramesh K Sitaraman, and Jennifer Sun. The akamai network: a platform for high-performance internet applications. *ACM SIGOPS Operating Systems Review*, 44(3):2–19, 2010.

[6] Vijay Kumar Adhikari, Sourabh Jain, Yingying Chen, and Zhi-Li Zhang. Reverse engineering the youtube video delivery cloud. In *Proc. of IEEE Hot Topics in Media Delivery Workshop*, 2011.

[7] L. Zhang, D. Estrin, J. Burke, V. Jacobson, J.D. Thornton, D.K. Smetters, B. Zhang, G. Tsudik, D. Massey, C. Papadopoulos, et al. Named data networking (ndn) project. Technical report, Tech. report ndn-0001, PARC, 2010.

[8] V. Jacobson, D.K. Smetters, J.D. Thornton, M.F. Plass, N.H. Briggs, and R.L. Braynard. Networking named content. In *Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies*. ACM, 2009.

[9] A. Ghodsi, S. Shenker, T. Koponen, A. Singla, B. Raghavan, and J. Wilcox. Information-centric networking: seeing the forest for the trees. In *Proceedings of the 10th ACM Workshop on Hot Topics in Networks*, page 1. ACM, 2011.

[10] Cheng Yi, Alexander Afanasyev, Lan Wang, Beichuan Zhang, and Lixia Zhang. Adaptive forwarding in named data networking. *ACM SIGCOMM Computer Communication Review*, 42(3):62–67, 2012.

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

[12] Sen Wang, Jun Bi, Jianping Wu, Xu Yang, and Lingyuan Fan. On adapting http protocol to content centric networking. In *Proceedings of the 7th International Conference on Future Internet Technologies*, pages 1–6. ACM, 2012.

[13] Alexander Afanasyev, Ilya Moiseenko, and Lixia Zhang. ndnsim: Ndn simulator for ns-3. http://irl. cs. ucla. edu/ndnSIM. html.

[14] Neil Spring, Ratul Mahajan, and David Wetherall. Measuring isp topologies with rocketfuel. *ACM SIGCOMM Computer Communication Review*, 32(4):133–145, 2002.

[15] Seyed Kaveh Fayazbakhsh, Yin Lin, Amin Tootoonchian, Ali Ghodsi, Teemu Koponen, Bruce M Maggs, KC Ng, Vyas Sekar, and Scott Shenker. Less pain, most of the gain: Incrementally deployable icn. 2013.