On Content Delivery Network protocols and applications

B. MOLINA, C. E. PALAU, M. ESTEVE, J. LLORET
Communication Department
Polytechnical University of Valencia
Camino de Vera s/n 46021 Valencia
SPAIN

Abstract: - Content delivery networks are overlay networks that reduce latency by placing a set of servers close to clients. It is specially effective for wide-area networks and the Internet, where network traffic may drive a user to wait for an unreasonable period of time. The approach for such a solution at application layers lacks for optimization techniques at network and data-link layer; from another point of view, it allows a rapid deployment of new applications and protocols, as well as enhancements of current tested ones. This article tries to describe and clarify current work and research in the mechanisms used at the upper-layers of the protocol stack of a content delivery network.

Key-Words: - Content Delivery Networks, application-layer protocols

1 Introduction
The web, often also called World Wide Wait, has grown up in the last years to carry a huge amount of traffic in the backbones. However, this has attracted more and more clients, who early get bored with static content and demand both dynamic and multimedia enabled content. Dynamic content depends on several factors, such as historical events, user’s profile and recent changes. Static content, on the other side, is easily cacheable on intermediate proxy caches, which can therefore improve content delivery. The idea behind content delivery networks (CDNs) consists of placing separate servers, called surrogates, near client location. If the user is redirected to a nearby surrogate, which acts as proxy, it can experience a significant reduction in the perceived response time.

Early content delivery networks were deployed at the end of the 90s addressing static content. As disk space was cheaper than network bandwidth usage, early CDNs were considered an overlay group of mirrors with intelligent redirection transparent to the user. After six years of historical background, CDNs are developing towards two basic new features: customization and streaming. These add-ons imply introduction of new devices, protocols and applications that have to be considered and analyzed. The rest of the paper is structured as follows. Section 2 introduces a series of considerations about the utilization of network protocols within a CDN. In section 3 we present upper-layer CDN protocols identifying different parts such as user interface protocols and applications, the CDN distribution network and the monitorization network. The paper finishes with the conclusions and future work.

2 CDNs and network protocols
Content networking makes networking decisions based on the content contained within the communication stream of help users and content providers to achieve their various expectations, such as performance, availability, security, flexibility, personalization, scalability and differentiation.

The Internet is the main arena where content networking takes place, though its scope is not limited to it and can occur at the so called enterprise CDNs. So let’s have a look at the former one. The Internet is a group of interconnected networks. Fifteen years ago, the number of networks were very reduced and its main aim addressed institutional and educational issues. Nowadays the Internet is the technological support of the Information Society, so a lot of players (carriers, ISPs and ITC enterprises) have appeared to obtain a strategic market position and share a part of the business cake. Despite the growth and changes within the networks, little has changed at low-level protocols, which are considered respect to internetworking, that is, network layer and transport layer.

The IP protocol makes possible communication between network entities within the Internet. The specification dates for 1981 and was very simple in its design conception; therefore, routers could operate at high-forwarding speeds and intelligence was pushed towards network edges. Some efforts are being made in the IPv6 specification to adapt the protocol to current technological demands, such as address namespace and scalability.

CDNs redirect clients depending on their geographical situation in the network according to the previous placement of surrogates. The performance
of the redirection mechanism is extremely important and can be done at various levels. However, since anycasting will not be added until IPv6, CDNs have to work at superior application layers.

Respecting transport layers, both TCP and UDP are utilized by CDNs; the former protocol allows a slow but reliable transmission of control and content information, whereas the latter one offers a non-reliable but fast delivery mechanism, both for resolution and streaming processes.

3 CDNs and upper-layer protocols

CDNs are overlay networks that mainly operate at application-layer protocols. When a client issues a content, it has to be redirected to a surrogate. But the network doesn’t have any intelligence to route according to a specific content, but to an IP address. So there must be a translation or mapping from content to IP addresses, which is served by the directory service, also called DNS within TCP/IP environments.

This is the first global redirection stage. Fig. 1 depicts a common DNS resolution phase. In last instance, the local nameserver queries an authoritative server, so here is the relationship between a client and the CDN service provider. The local nameserver represents a client; all clients connected to the same local DNS server are treated as if they were the same client. Therefore, performance analysis are dealt with clusters of clients. In the case of an ISP, a cluster corresponds to client connected to a certain Point of Presence (PoP). Note that, if the client is placed far away from its local nameserver, the geographic estimation cannot be accurate and is discouraged.

On the other part of the world, the CDN service provider owns the authoritative server in charge for analyzing the incoming request (from the local nameserver) and responding with at least one IP for a surrogate to contact. Under normal conditions, the target surrogate is one placed near client location in order to reduce response time. Besides, load balancing can be reached at this level. An intelligent entity on the authoritative server can have a global knowledge of surrogate load as well as possible congestion paths. This way, a client is redirected to a nearby, low congested surrogate.

However, caching techniques in the previous steps (both in the local DNS server and even in the client browser) makes it only possible to achieve a coarse-grained load balancing redirection strategy.

A more precise, fine-grained redirection takes place as the client retrieves an IP address of the surrogate to contact. This surrogate can behave in two modes:

- if it is low-loaded, it can simply answer to the client with the desired content. This is the ideal scenario as the user perceives a reduced response time
- if it is working under severe conditions, it can either redirect the request to another surrogate or to a central intelligent entity that estimates a better surrogate. This situation can be caused as a caching drawback in the DNS resolution phase.

A second redirection is normally unpleasant for a user. Therefore, in order to avoid this situation local scalability can be used at this stage in form of a server farm: the client contacts a virtual IP address hosted by a layer 4-7 switch, which then (on-the-fly) redirects the request to a local low-congested server. Once the main functions of a CDN have been explained, it is now time to center in the protocols that allow the communication within the CDN architecture.

3.1 User interface protocols and applications

Some users think Internet and the World Wide Web is the same. This is because of its common use; even some applications such as e-mail and ftp offer a web interface. The web is supported under HTTP and the standard programming language is HTML. HTTP has evolved from a simple 0.9 version released in 1991 to a powerful 1.1 version in 1998. HTTP/1.1 improves response time by using persistent connections that allow a client to share multiple requests over a single TCP/IP connection by means of the ‘Keep-Alive’ field in the connection header. Besides, the protocol also incorporates a pipeline facility, i.e., the capacity of performing multiple requests before receiving a response.

Relating to content delivery and CDNs, HTTP/1.1 includes cache control mechanisms (cache-control and pragma header) in order to inform clients and proxies that certain data should not be cached. Besides, the protocol supports status messages or codes in order to tell the result of a web transaction. Among them, the 3xx codes are specially suitable for redirecting a client from a surrogate to another one at the application layer.
Once the communication between client and server via HTTP has been established, it can happen that it has to be secure, for instance, in case of a business transaction. HTTP supports two different ways of authentication—basic and message digest—but none of them are secure enough, so another protocol has to be used: HTTPS, which normally corresponds to the encryption of HTTP by an SSL session. In the case of a CDN, the use of HTTPS is generally accompanied by a redirection to a secure server from a surrogate, that is, the number of clients to buy is much smaller than those who only want to see available products; therefore, they can be handled by a unique server. However, encryption is based on complex algorithms which suppose a performance reduction unless SSL acceleration mechanisms are used via cryptographic cards [2] or proxy-SSL accelerators [3].

Besides HTTP, which serves images and text to users, the demand of a multimedia interface implies using other applications and protocols for presenting audio and/or video to the user. The old method of downloading the audio/video file and playing it locally has changed to a streaming delivery mechanism. Common famous applications are Windows Media Player, RealPlayer and QuickTime. All of them include a plugin for the web browser in order to embed the streaming inside HTML. Multimedia streaming formats vary differently, as well as the encapsulation methods. Normally, as latency is severely perceived, packets are sent via UDP instead of TCP. However, due to proxy firewalling of UDP packets, sometimes HTTP and TCP is introduced to bypass such circumstance. Timestamps for synchronization are introduced via RTP, and add-on features such as video-on-demand is provided by RTSP, though the latter is hardly deployed currently.

3.2 The Distribution network
Though often mentioned as interchangeable terms, the communication process inside a CDN can be divided into two separate ones: (i) the distribution network, between origin site and surrogates, and (i) the delivery network, between surrogates and clients. The collection of surrogates that compound the CDN replicate content of the origin server. As content changes, freshness has to be managed. The two most common approaches are propagation and invalidation. In the former one, whenever a piece of content is updated in the origin server, it is also delivered to all replicas. In the invalidation approach only an invalidation message is sent to all replicas which will have to fetch the content when it is requested by a user. Propagation is the technique often used in CDNs, though there are also hybrid approaches that tend to be more efficient [4]. FTP over TCP are normal protocols to guarantee a reliable data transmission to all replicas. In other advanced and decentralized architectures, a surrogate can fetch content from multiple sources via MFTP, similar as it happens with popular peer-to-peer filesharing clients. More concretely, there are specific protocols at higher layers that deal with cache management, such as the Internet Cache Protocol (ICP), largely implemented in a variety of software: Squid, Microsoft Proxy, Volera, Cisco Cache Engine, etc. Other alternative protocols and algorithms are HyperText Caching Protocol (HTCP), Cache Array Routing Protocol (CARP) and Cache Digests.

The process of delivering this data can take place late at night or at low traffic periods. Not all surrogates must retrieve all the content available at the origin server. It depends on an (arbitrary) administration policy that considers disk space on each surrogate, client population demands and the amount of total available content. All of this is easy for static content, but a CDN also supports both dynamic and live content. Dynamic content, such as an ASP or JSP page, is simply not cacheable, as it is generated on the fly; however, some modules or components that do not change or are common for a considerable amount of requesting clients can be reused and thus cached across multiple user sessions. Market examples are Xcache [5] and ESI [6].

Live content, audio or video, must be delivered as fast as possible; the best distribution platform to avoid the Internet is via satellite, but in a normal scenario we cannot do that. In this case, content must not be cached at the surrogates, which act as merely data relayers. Here it is important the efficiency in the distribution path: the shorter the path, the slower the delay and jitter. As traditional IP multicast is not supported, an application layer multicast tree has to be created. In reality, for a small number of surrogates the origin site sends data directly to all of them via unicast streams; if the number of surrogates grows up to a certain number, the origin site can hardly keep track information of all the surrogates which leads to a severe reduction of its performance; thus it is necessary to build a multicast hierarchy (tree) at application layer. In general, Application Layer Multicast (ALM) protocols can be classified as tree-first or mesh-first approach. The former ones, such as HMTP, TBC and NICE [7], are more commonly used, above all in P2P networks. Note however that implementation may vary from CDNs to P2P networks: within a CDN the availability of a surrogate is guaranteed, which is not the case of a peer in a P2P network.
3.3 The monitorization network

The intelligence of the CDN resides in one or several processes that take decisions based on input information. This input information is predominantly taken as feedback through a monitorization system. In a normal distributed system the availability of servers and their processes is the target issue. Network connectivity can be tested with a simple ping from an agent, whereas upper-layer monitorization – track for working processes – can be done via applications such as Big Brother and Nagios [8]. Those applications offer more sophisticated services, such as network services via SMTP, POP3, HTTP, NNTP, etc., and host resources (server load, disk usage, memory utilization, running processes and logs).

Surely, the most common monitorization protocol is SMTP. Though its specification does not offer a lot of potential resources to take track of, there are many plugins available, some of them even open.

In the case of a CDN, the system has to monitor several aspects:
- server availability, in order to redirect a user to a different surrogate if one fails
- server general load, normally in terms of CPU and memory usage and number of open connections.
- client location, in order to redirect it to a nearby low-congested surrogate at the arrival of a request
- system overall performance

Many people try to avoid server agents, as some analysis have proven to be it less efficient than the traditional LeastConns metric. However, not all servers are standalone, and the correlation between CPU load and connections is sometimes not clear.

Moreover, in the case of a streaming connection, the incurred load is different depending on the content features (size, quality and encoding type, frames/s). Anyway, LeastConns is mostly implemented via hardware in front-end switches and therefore, application layer control is avoided.

Client location is difficult to calculate in the redirection process if the number of surrogates is considerable. A simple strategy consists of pinging or tracerouting the client from the surrogates to estimate proximity.

Overall performance is critical for a commercial CDN, as it has to show their customers how well their site is playing in order to charge them accordingly. The information customers want to know varies from the number of requests per second up to the top pages most requested by their clients. This can be done by log-analyzers or via private communication protocols.

4 Conclusions

Content Delivery Networks are application-layer networks on top of the Internet which consist of a distributed system of surrogates that serve content to users issuing efficiency, scalability, availability, security and response time. The necessary intelligence is created by means of process entities which communicate through the network in form of many protocols. Being the Internet the infrastructure, common standardized protocols have to be used to accomplish all the necessary tasks, such as HTTP, RTP and TCP. Other protocols, public or secret, can be used on top of the former ones in order to achieve special tasks. This article has focussed on both protocols, trying to describe the operation of a CDN through the series of protocols involved in the overall communication process.

There are different approaches for understanding what a CDN is and study how does it perform, such as the analitical model [9] and an implementation description [10]. Both are being studied within our research department to put clarity and offer an open and standard CDN model and implementation.

References: