# How NewNode Works

**And Who Needs a Content Distribution Network Anyway?**

## Efficient and Inefficient Networks

If all networks were fast, efficient, and never suffered from congestion there wouldn't be any need for a Content Distribution Network. But that's not how modern networks function: it's easy to create a system where demand overwhelms supply and a requested connection is slow or impossible to create. Good network design can minimize these problems so the fundamental goal is to create the most efficient and resilient network possible. But what would this look like? What are the features of such a network?

As a way to understand how efficient and inefficient networks function let's consider a simple example arising from an informal high school election. All the students recently voted for High School President and the cast ballots sit in a large bucket on the principal's desk. Who won? What's the best way to inform the students of the results?

One extremely bad way to answer this question, used by no school anywhere at any time, is the following approach: 1) all the students are put in a line leading up to the principal's office, 2) each student enters the principal's office one-by-one, 3) each student individually counts the ballets in the bucket, and finally, 4) after a student figures out who won the election they're ushered out of the room by another door and the next student's let in. Eventually the election result (the important bit of data) is known to everyone in the student body (making up the network).

In a school with 200 students this method would require the ballets to be counted 200 times and the overall process is long, arduous, and subject to error. It's also extremely inefficient. Isn't there a better way? Of course.

A faster method involves letting three or four students into the room at the same time. After they count the ballots and determine the victor they sign a scrap of paper showing that they all agree on who won the election, snap a picture of the paper on their smartphones, and are released into the school at large. The students excitedly inform their friends of the election victor (and their friends inform their friends) and the news rapidly spreads across the school. Within minutes everyone knows the agreed-upon result.

In this scenario the ballots were counted three or four times at most and everyone present during the counting certified the validity of the winner. This is the normal way to transmit important information to a network made up of humans and it works very efficiently.

### Waiting in Line

Unfortunately the first, inefficient procedure is the way traditional internet clients interact with internet servers. For example, if twenty games installed on twenty devices require an update, all twenty devices contact the game server at roughly the same time and essentially wait in line as the update is downloaded twenty times (one for each of them). This puts inordinate pressure on the game server as every game attempts to update from the same source at roughly the same time. Like the students waiting in line outside the principal's office each game device takes a turn receiving the data and has no contact with any game device that's already been updated.

As was shown in our election example, a better procedure involves a few game devices updating themselves from the game server and thereafter transmitting that update *directly to other game devices* (let's remember that it only takes a few talkative students to quickly spread the news of the election results to the entire school). This type of network is called peer-to-peer as it does not depend upon a centralized server to control data distribution; each peer (member of the network) is able to transmit validated and encrypted information to any other peer it's connected to. A peer-to-peer network would allow valid game updates to be shared *between the game devices* instead of forcing each game device to download the update from a single congested source, the game server.

Information in peer-to-peer networks has the ability to flow very quickly as each peer contacts multiple other peers and transmits data. Even if each peer is only able to communicate with two other peers the amount of data flowing through the network ramps up quickly: one peer updates two, those two peers update four others, those four peers update eight others, etc.

# Traditional Solution: Content Delivery Networks (CDNs)

In order to get around the data bottleneck involved with a single server attempting to deliver data to thousands or millions of devices, traditional Content Delivery Networks (CDNs) developed. CDNs consist of hundreds or thousands of privately-owned servers which take data from an original source and spread out the distribution load. A game company's CDN might consist of twenty game servers located in each continent; when game updates are required each game device is updated from the CDN game server closest to its physical location. Instead of all game devices updating from a single server multiple servers are now available. This helps reduce the initial bottleneck but CDNs are expensive to create and maintain and there are still a limited number of servers transmitting game updates. It's no longer a single server doing all the work but it's still a relatively small number of servers that have to handle a global load (if the game is popular and has millions of installations). CDNs improve network performance but represent an expensive and limited technical evolution.

# NewNode's Solution: Decentralized Content Delivery Network (dCDN)

NewNode is a Decentralized Content Delivery Network (dCDN), which means that data transfer occurs peer-to-peer if a direct connection to a web server is slow or compromised.

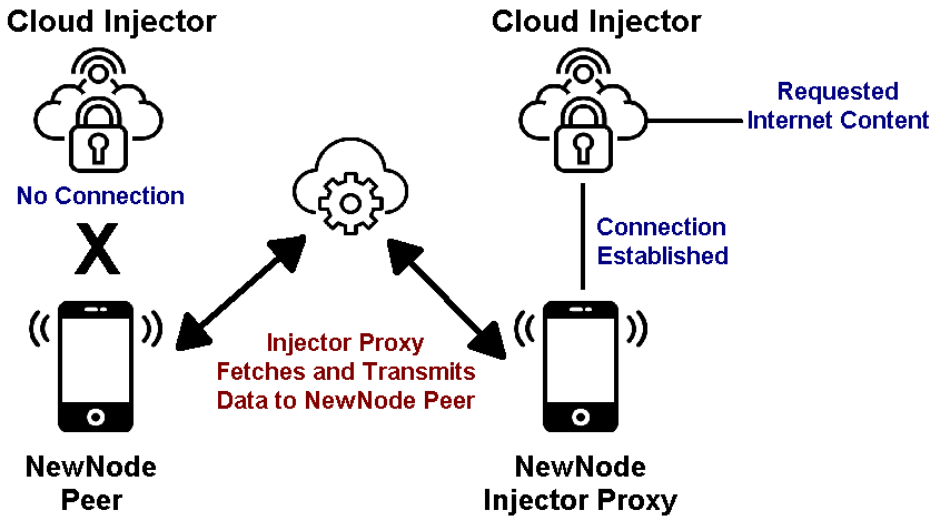
To continue our example: if a game company used NewNode to distribute updates, game devices would be capable of communicating with other game devices, forming a peer-to-peer network. Updates would flow, for the most part, not from a central server or traditional CDN but device to device as each game device functions as a server (peer). If there are a million game devices in use around the world there are (potentially) a million ways for them to communicate to each other and distribute updates. Significantly, this method takes advantage of a large and well-distributed network already in existence: the already-connected and relatively powerful game devices themselves. Billions don't have to be spent creating a custom and unique CDN just to service a single game (or game platform).

### NewNode's Peer-to-Peer Network

For NewNode, the nodes (peers) making up its decentralized network are mobile devices linked to each other using dynamic addresses hosted by the large and already-established (250 million+

peer) BitTorrent network. Because the BitTorrent network is globally distributed and extremely resilient, NewNode peers are able to reliably broadcast their availability and securely identify and communicate with each other. In addition, NewNode uses trusted Cloud Injectors to encrypt and transmit data from web origins and "inject" this data into the peer-to-peer network.

If a NewNode-enabled device is able to connect to a Cloud Injector quickly and reliably it will receive encrypted web content directly from the originating web source. If a NewNode-enabled device is *unable* to reach a Cloud Injector (or the connection is slow) the device instead connects to a NewNode peer which serves as an Injector Proxy; if the Injector Proxy don't already have the requested data they, in turn, download the data from a faster and more reliable Cloud Injector and transmit it directly to the requesting node.




A single node connected to a Cloud Injector is able to quickly populate the entire peer-to-peer network with requested data even if Cloud Injectors are unavailable to other nodes due to network congestion, local bottlenecks, or regional blocking. Once data is entered into the NewNode network it quickly spreads throughout the network.

For example, if there are ten NewNode-enhanced mobile apps deployed across the globe, with a total of 10 million app users, there will be an ever-fluctuating base of 10 million nodes capable of receiving and transmitting data depending on the number of connected devices. Ultimately, NewNode's distributed network will be orders of magnitude larger than typical CDNs, which use at most hundreds of thousands of servers. The large and ever-increasing number of mobile devices in the world, combined with high connectivity availability, means that a mature NewNode network may eventually include 500 million nodes, constituting a distributed network with a size, speed, and flexibility impossible for traditional CDNs match or deploy.

## Who Can Make Use of NewNode?

Any mobile device can make use of NewNode to increase data transfer speeds and ensure that data delivery to peers precisely matches the data requested from the originating web server. NewNode also offer a powerful workaround for local or regional network slowdowns; peer-to-peer connections are often much faster and more reliable. NewNode can be immediately deployed to handle a wide variety of network demands required by game developers, cryptocurrency clients, multimedia companies, and bulk data transmission services. In addition NewNode functions as an additional layer of network security, encrypting all data on the peer-to-peer network and the original site data coming from the Cloud Injectors, making governmental or corporate censorship and/or surveillance impossible. NewNode also functions as a defense against many types of cyberattacks, such as DDoS: because the network is fully distributed and constantly changing, adversaries are unable to pinpoint and overwhelm a specific web address or range of addresses.

## Technical Components

NewNode relies on three key technical building blocks:

1. Low Extra Delay Background Transport (LEDBAT)

   LEDBAT is a congestion protocol that ensures that NewNode network transmissions don't impact other app network functionality or otherwise slow down/congest the network. LEDBAT was invented by one of NewNode's developers, and is used by Apple for all Software Updates, by Microsoft for Windows 10, as well as countless other companies distributing bulk data in the background. LEDBAT is estimated to carry 15-20% of all Internet traffic.

2. BitTorrent Distributed Hash Table (DHT)

   BitTorrent is the largest currently deployed, decentralized, and serverless piece of infrastructure on the Internet, encompassing a quarter billion nodes. NewNode takes advantage of the deployed BitTorrent base by using the DHT table to power a robust discovery mechanism, allowing NewNode-enabled devices to dynamically locate and connect to each other.

3. [Future Functionality] Device-to-Device Connectivity

   The NewNode developers are pioneers of device-to-device connectivity, and recently

developed and deployed the FireChat app based upon this technology. Device-to-device connectivity allows NewNode to function without Internet connectivity (via Bluetooth or other types of local connections), and enables connectivity with laptops which are often crucial for content distribution.

## Easy to Get Up and Running

NewNode is implemented by an app developer as a easy-to-deploy Software Development Kit (SDK) library. NewNode doesn't require a high level of technical expertise or network sophistication to get up and running: an app can be NewNode-enhanced simply by including a software library and adding two lines of code.

Once an app is modified in this way, the new version is pushed to users as an app update. After updating, the end user may even be unaware that the app has been NewNode-enhanced; no special actions or configuration adjustments are needed for them to continue to use the app as normal.

*The NewNode end user experience is fundamentally transparent*, with the only noticeable effects being positive: higher app download speeds, more reliable and durable data connections, and a greater ability to receive app data regardless of attempted censorship, site blocking, or cyberattacks.

---

NewNode overcomes the shortcomings of existing CDNs, *allowing app developer content to reach users despite network congestion and attempts to block, censor, or control the data*.

NewNode also *defends against DDoS attacks* and *features strong encryption defeating Deep Packet Inspection and other passive attack mechanisms*.

---