The Fountain of Knowledge.

Infinitely scalable storage in the data centre.

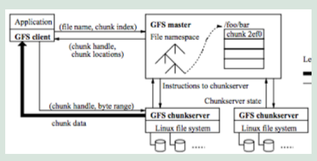
Toby Moncaster, George Parisis, Anil Madhavapeddy, Jon Crowcroft *(first.last@cl.cam.ac.uk)*

Existing approaches to data centre storage

Centralised disc-based storage (e.g. SAN)

- remote array of discs presented as local storage to servers
- Connected using storage network
- Use TCP as transport - leads to incast

Centralised metadata (e.g. Colossus)



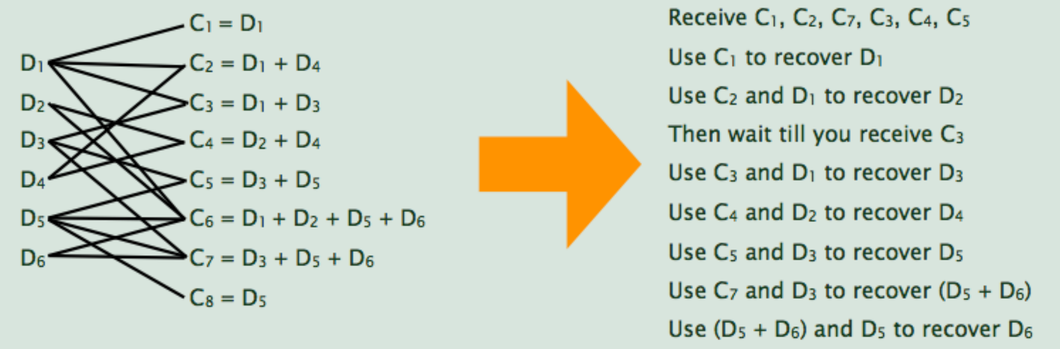
- chunks stored in RAM
- central metadata server determines where to store new chunks
- offer “raw” storage (block or chunk)
- issue with size of metadata - dictates the size of chunks (e.g. Mbytes)

Distributed metadata (e.g. FDS)

- Storage is split into blobs. Data is split into tracts.
- Each blob has tract server which allocates space to tracts.
- Central server just lists location of tract servers.
- Simple hash determines which blob(s) contain which tract.
- Requires full bisection bandwidth network.

Fountain coding

- Data is encoded using sparse erasure codes (Luby Transforms, Tornado codes, etc).
- Truly rateless coding technique - receiver needs to get N + ∂ codewords to recover N data blocks, but can get *any* N + ∂ codewords.
- Data to be coded is split into blocks.



- Combinations of blocks are then XORed together
- To decode you need to start with a codeword with 1 block. Then XOR it with all blocks containing it.

Using fountain codes for storage

Fountain Codes offer several neat advantages

- Rateless - so no need for feedback, timeouts, etc. If a codeword is lost you just have to wait for another.
- Efficient - encoding penalty is a constant 3-10% (depending on approach). **ANY** N + ∂ codewords allow you to recover the original data.
- Data can be multicast - better than simple replication (this allows the data to be read in parallel from many sources at the same time)
- Offers a chance to load balance and hence make better use of limited storage and network resources.

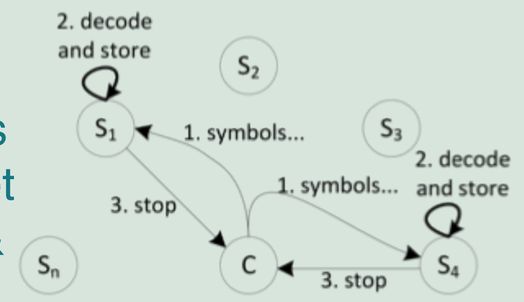
Two drawbacks:

- XOR is relatively computationally expensive. But it is very easy to do in hardware (c.f. NetFPGA as a possible solution)
- Storage has to be semi-immutable (e.g. write to erase). Could use a checkpointed git like file system (e.g. Irminsule)

Writing data

Central controller, C, decides where blocks are to be sent.

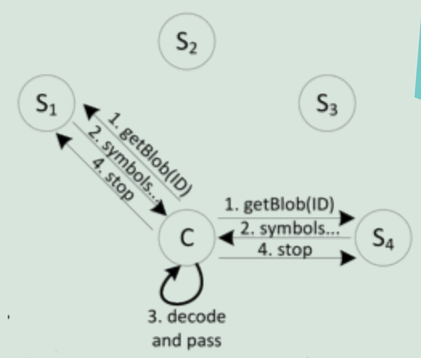
1. Data is converted to symbols
2. symbols are distributed to set of storage nodes, decoded & stored
3. Once all symbols received storage nodes send stop.



Reading data

Send request to C.

1. C sends getBlob request. Sn recovers correct data and creates a set of symbols.
2. These symbols are sent to C.
3. Once enough symbols are received storage C sends stop.



Data Centre Basics

- Massive warehouses full of commodity servers: the “home” of the Internet and cloud services
- Can consume upwards of 100MW each - rapidly exceeding airline industry as a source of CO2
- Biggest contain 250,000+ computers connected by very fast network (10-40GbE)
- Raise interesting research challenges:
  - Latency measured in nanoseconds - large packets may be in source and destination at same time
  - Often quicker to move application to data, rather than getting data from disc
  - Suffer from specific problems including TCP incast and broadcast storms

UNIVERSITY OF CAMBRIDGE

Research Group Systems srg@cambridge:~$

Computer Laboratory
William Gates Building
15, JJ Thomson Avenue
Cambridge, CB3 0FD

EPSRC
Pioneering research and skills

INTERNET
INTelligent Energy awaRe NETworks