The Power of Prediction: Cloud Bandwidth and Cost Reduction

Eyal Zohar  Israel Cidon  Osnat (Ossi) Mokryn

Technion  Tel-Aviv College
Traffic Redundancy Elimination (TRE)

Traffic redundancy stems from downloading same or similar information items. We found around 70% redundancy in end-clients traffic, compared with past traffic and local files.
TRE Importance

Moving to the cloud => higher e2e traffic.
Cloud users pay for traffic used in practice => incentive to use TRE.
How TRE Works

Server parses the outgoing stream to content-based chunks and signs with SHA-1

Insertion example
Problems in Existing Solutions

In the cloud environment:

1. High processing costs in the cloud.
2. Scalability – remember each client.
3. Elasticity - unaware of data from other sources.
4. Do not handle long-term repeats (days/weeks).
Our Solution: PACK (Predictive ACK)

Redundancy detection by the client.
Repeats appear in chains.
Tries to match incoming chunks with a previously received chain or local file.
Sends to the server predictions of the future data.
PACK: The Client Prediction

Each prediction:
1. TCP seq. – no server parsing
2. Hint – spare unnecessary SHA-1
3. SHA-1 signature
PACK: Server Operation

The server compares the hint with the last-byte to sign. Upon a hint match it performs the expensive SHA-1. PACK saves cloud’s computational effort in the absence of redundancy.

First receiver-based TRE: the server does not parse. It signs with >99% confidence.
PACK Benefits

Minimizes processing costs induced by TRE.
- Signs with SHA-1 in the presence of redundancy.

Receiver-based end-to-end TRE => suitable for cloud server elasticity and client mobility.
- Does not require the server to continuously maintain clients’ status.
Server Effort Experiment

Several data-sets in 3 modes: baseline **no-TRE**, **PACK** and a **sender-based** TRE.

25%-30% redundancy: common to many data-sets

SIGCOMM 2011
YouTube Redundancy

Traces of 40k clients, captured at an ISP. Found 30% end-to-end (personal) redundancy.
Long-Term TRE

Social network: eliminated 30% with one hour cache and 75% with a long-term cache.
Cloud Email Redundancy

Gmail account with 1,000 Inbox messages. Found 32% static redundancy (higher when messages are read multiple times).
Implementation

Linux with Netfilter Queue, 25k lines of C and Java, available for download.

Receiver-sender protocol is embedded in the TCP Options field.

Transparent use at both sides.
Processing Effort in the Client

Laptop experiment: PACK-related CPU consumption is ~4% when playing HD video (9 Mbps with 30% redundancy).

Smartphone experiment: PACK consumes ~3% of the battery power when processing 1 GB video (avg. monthly data plan).

Virtual traffic saves the client the need to chunk or sign.
New Chunking Algorithm

Most existing solutions use Rabin fingerprint.

Proc. 8 PACK chunking algorithm

1. \( mask \leftarrow 0x00008A3110583080 \) \{48 bytes window; 8 KB chunks\}
2. \( longval \leftarrow 0 \) \{has to be 64 bits\}
3. for all \( \text{byte} \in \text{stream} \) do
4. \quad \text{shift left} \( longval \) by 1 bit \{lsb \leftarrow 0; \text{drop msb}\}
5. \quad longval \leftarrow longval \text{ bitwise-xor byte}
6. \quad \text{if processed at least 48 bytes and} \ (longval \text{ bitwise-and mask}) = mask \ \text{then}
7. \quad \text{found an anchor}
8. \quad \text{end if}
9. \quad \text{end for}
Summary

Current TRE solutions may not reduce cloud cost. PACK is the first receiver-based TRE – leverages the power of prediction. Minimizes processing costs induced by TRE. Suitable for cloud server migration and client mobility. Implementation is available for download.