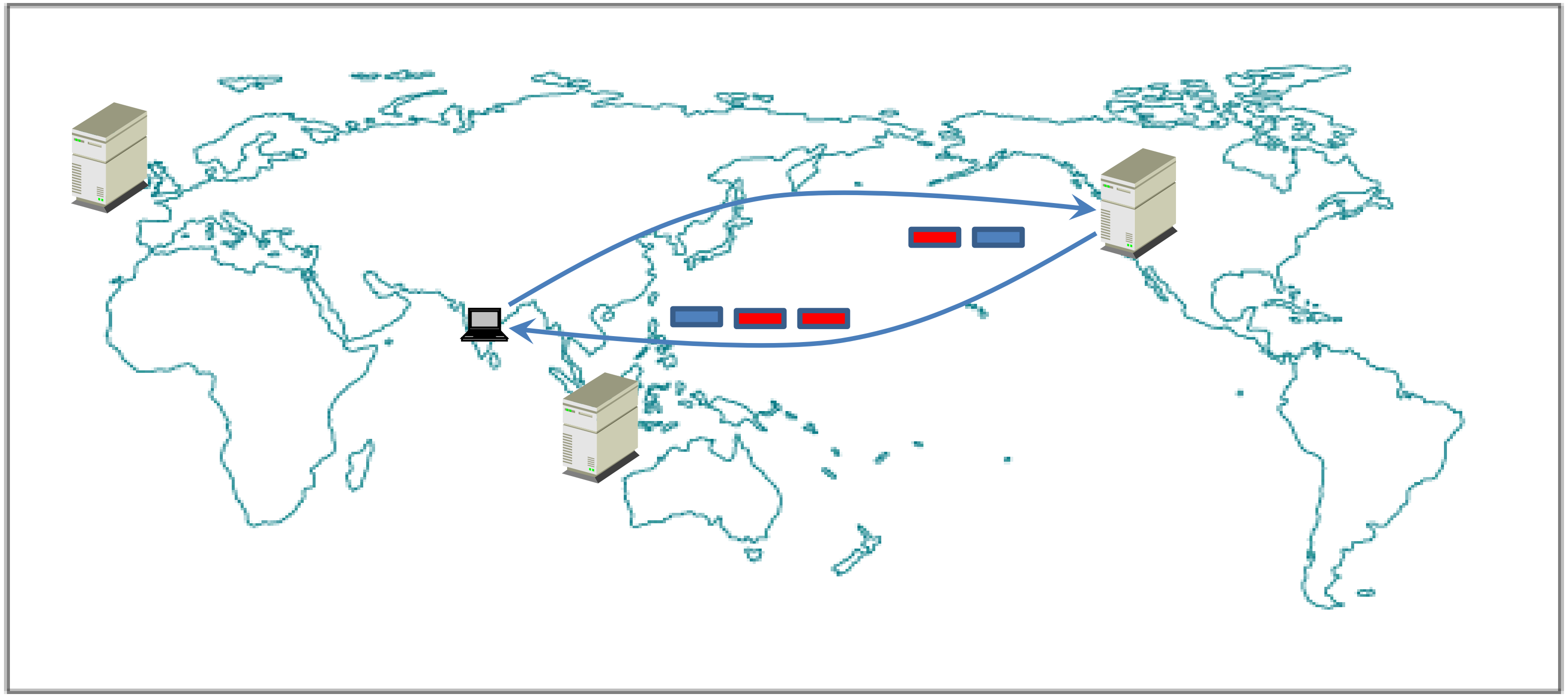
# EndRE: An End-System Redundancy Elimination Service for Enterprises

**Ram Ramjee**
**Microsoft Research India**

Bhavish Aggarwal^, Aditya Akella*, Ashok Anand*,
Athula Balachandran~, Pushkar Chitnis^, Chitra Muthukrishnan*,
and George Varghese#

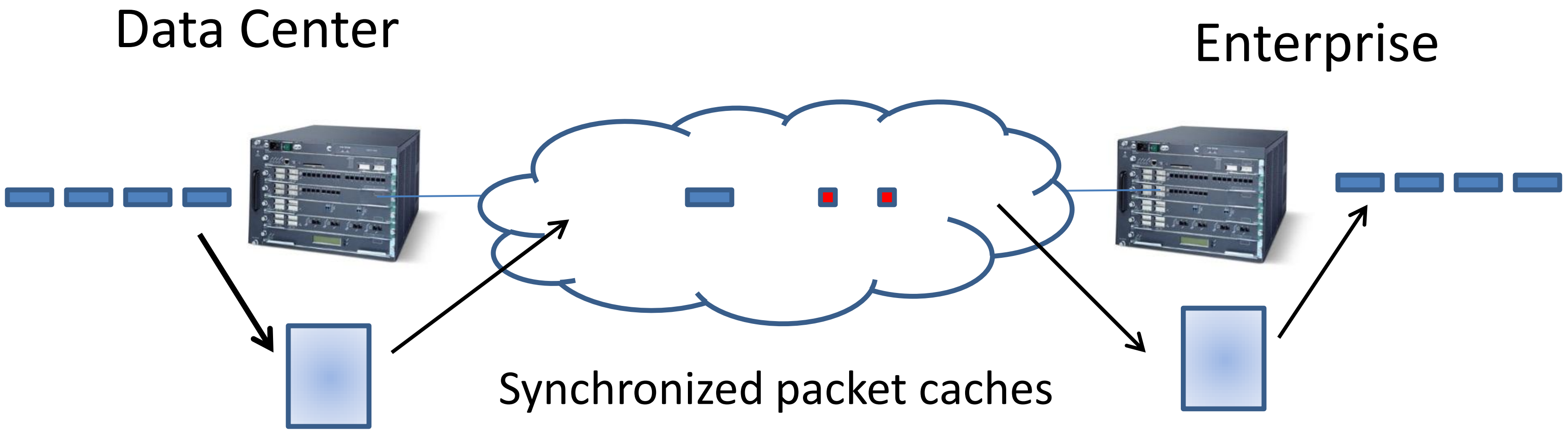
^: Microsoft Research India *: University of Wisconsin-Madison ~: CMU #: University of California, San Diego

# Enterprise Dilemma



- Large enterprises have a global footprint
- Data centers consolidated to save management cost
- Diminished performance due to Wide Area Network (WAN) bandwidth and latency constraints

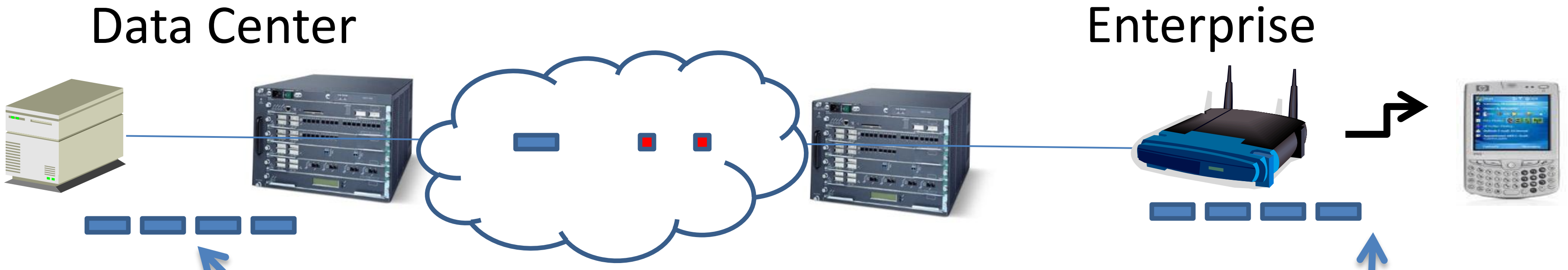
# Middlebox-based WAN Optimizers



- Protocol independent redundancy elimination using synchronized in-memory caches at two ends [Spring & Wetherall, Sigcomm 2000]
- Disk-based caches for large static objects
- Current leaders: RiverBed, Juniper, Cisco,…
- Annual revenue > $1Billion

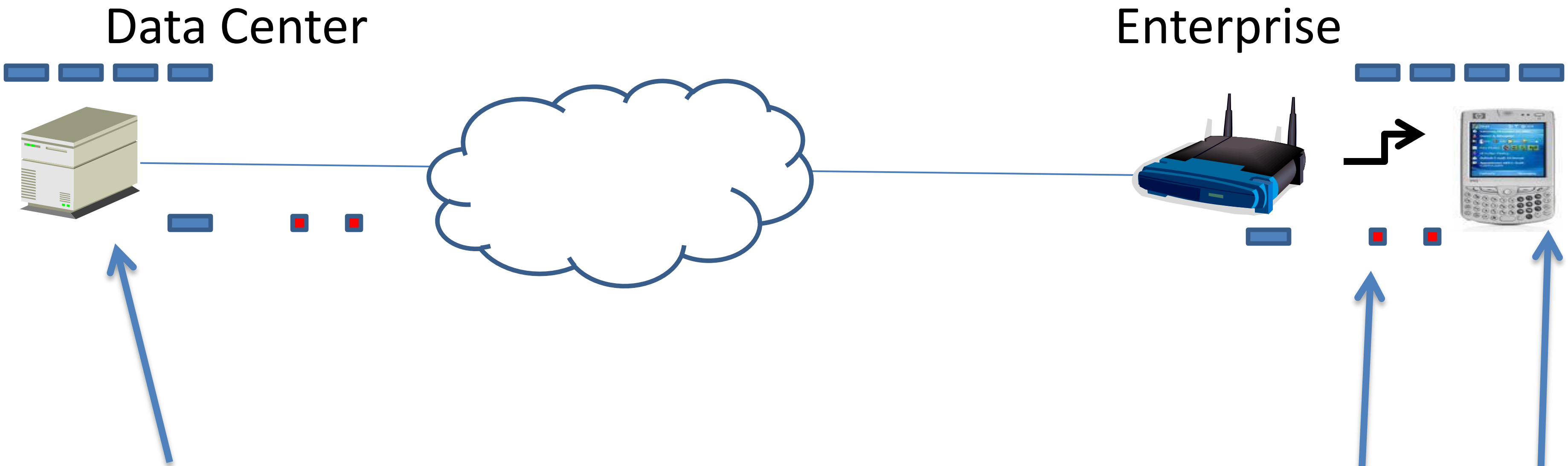
➢ Are middleboxes the right approach for enterprises?

# Issues with Middleboxes



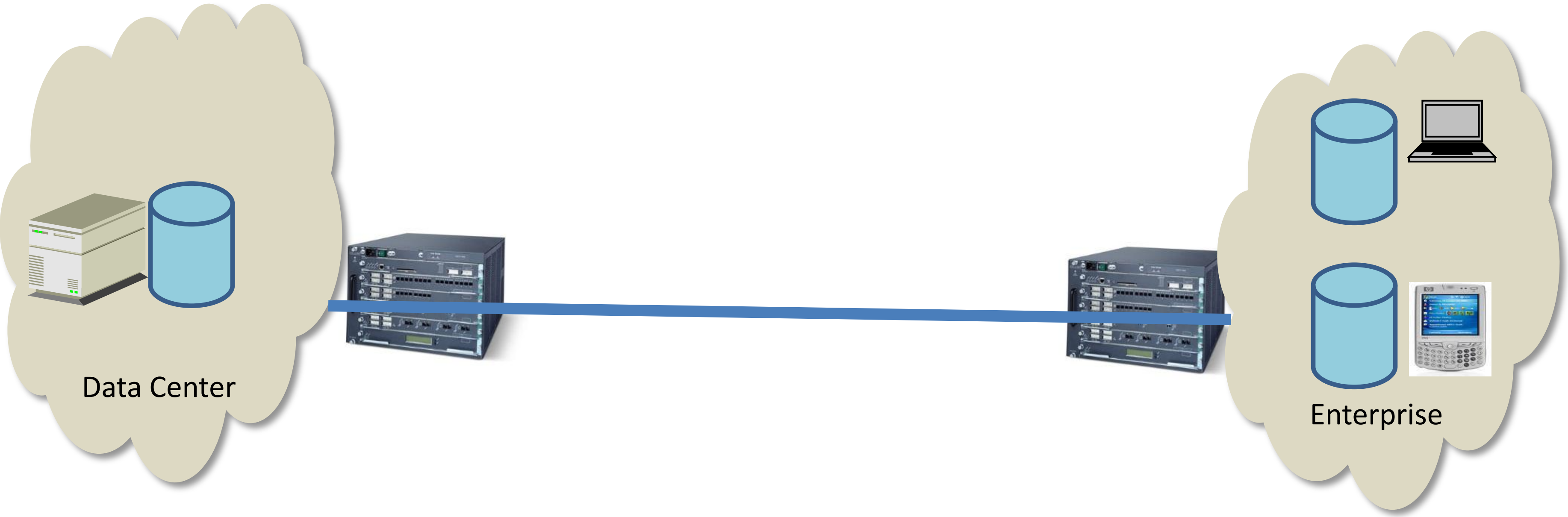
1. End-to-end security and encryption
   - Either no RE or require key sharing
2. Resource-constrained mobile smartphones
   - No RE on the bandwidth limited 2.5/3G wireless link
3. Cost

# End-to-End RE: Benefits



1. RE before encrypt ⇒ End-to-end security
2. RE on mobiles ⇒ Bandwidth savings over wireless
3. Bandwidth savings + simple decode ⇒ Energy gains
4. Operate above TCP ⇒ Latency gains

# Our Contributions



1. EndRE Design
   - New SAMPLEBYTE fingerprinting for fast processing: 10X speedup
   - Optimized data structures for reducing memory overhead by 33-75%
2. Evaluation of benefits
   - Analysis using 6TB of packet traces from 11 sites over 44 days
   - Small-scale deployment

# Outline

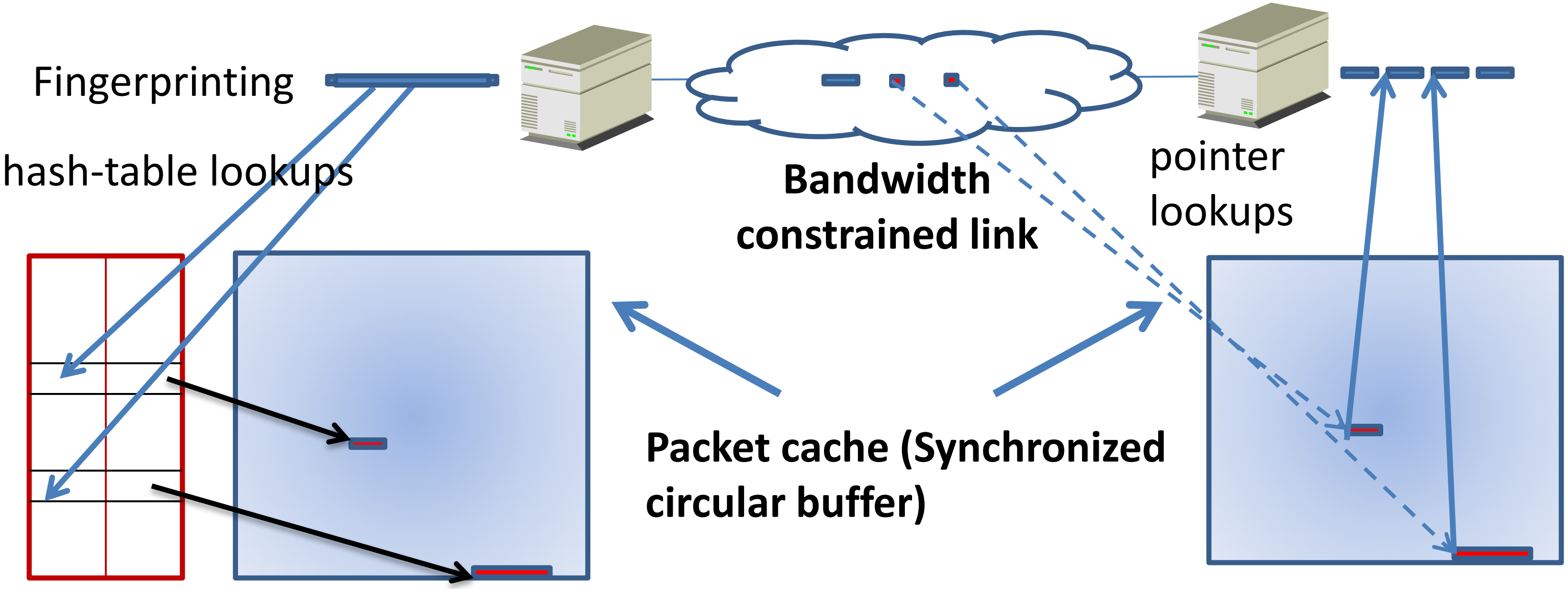
- Overview
- **Design of EndRE**
- EndRE costs and benefits
- Summary

# EndRE: Design Goals

➢Opportunistic use of limited end host resources

1. Fast and adaptive RE processing
   – Lightweight and tunable depending on server load
2. Parsimonious memory usage
   – Data structure and design optimizations to reduce memory overhead
3. Asymmetric
   – Simple client decoding

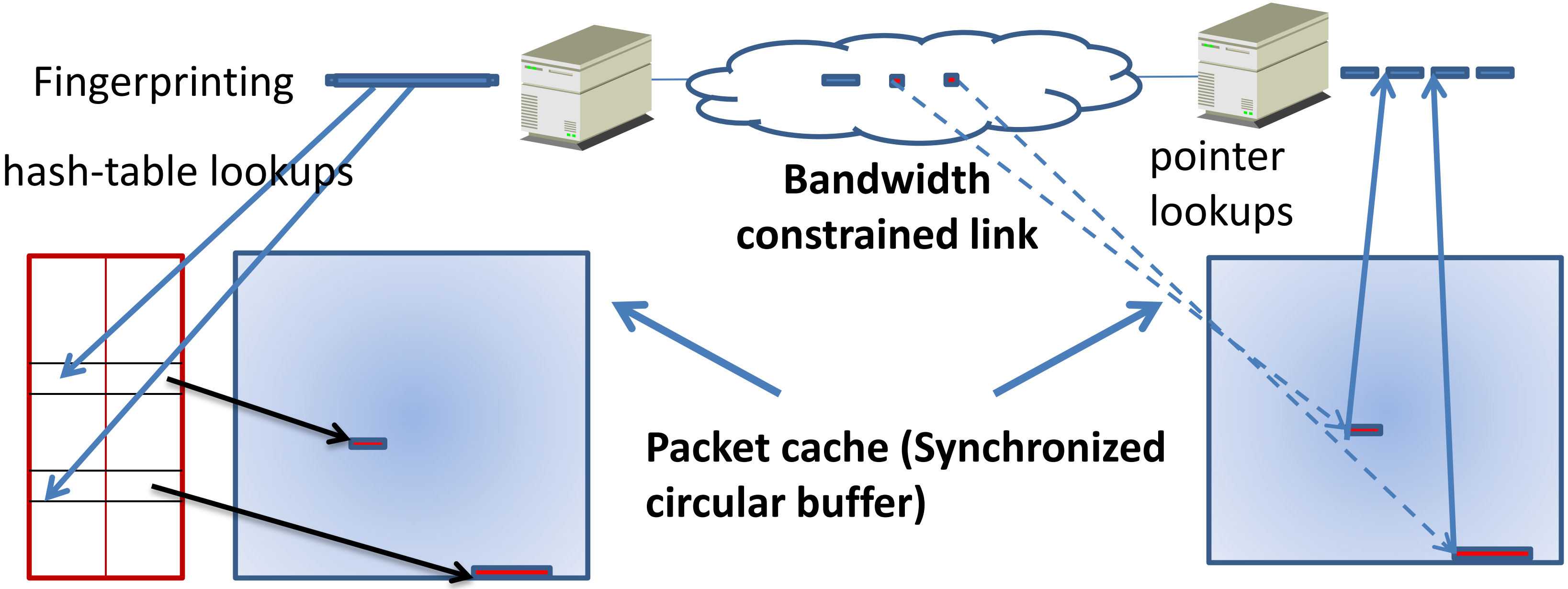
# Redundancy Elimination: Overview


Fingerprinting
hash-table lookups
Bandwidth constrained link
pointer lookups
Packet cache (Synchronized circular buffer)

Need to quickly identify repeated content (≈32 bytes)

- Identifying all matches (optimal) impractical
- Sampling-based approach necessary but comes at the cost of missed redundancy identification

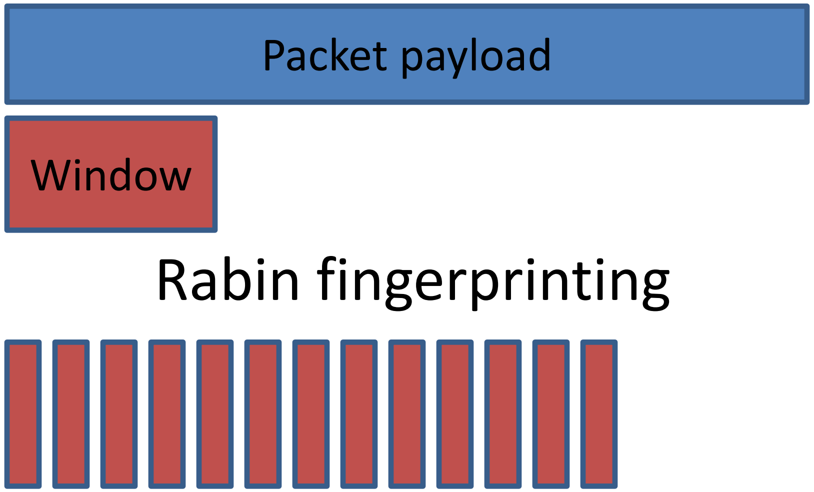
# Redundancy Elimination: Overview


Fingerprinting
hash-table lookups
Bandwidth constrained link
pointer lookups
Packet cache (Synchronized circular buffer)

1. **Fingerprinting**
   - Generate representative fingerprints of packet
   - New SAMPLEBYTE fingerprinting algorithm
2. **Matching & Encoding**
   - Lookup fingerprints in a hash-table of cache fingerprints
   - *Max-Match*: Byte-by-byte comparison between cache & packet
   - *Chunk-Match:* Full chunk matches (see paper)
   - Encode matched region with (position, length) tuples

# 1. Fingerprinting: MODP

- Compute fingerprints based on *content* [Spring & Wetherall]



Packet payload

Window

Rabin fingerprinting

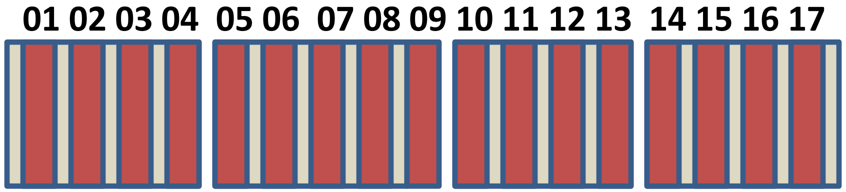
Value sampling: **sample those fingerprints whose value is 0 mod p**

+ Robust to small changes in content ⇒ better bandwidth savings

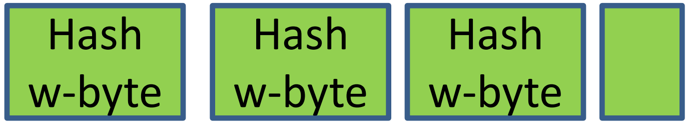
– Rabin hashes expensive and not adaptive ⇒ lower speed

# 1. Fingerprinting: FIXED

- Fingerprints chosen at *fixed intervals by position* in the packet



*Choose marker every* ***p*** *bytes*

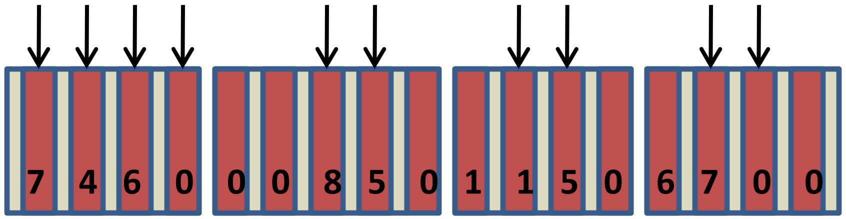


*Fingerprints*

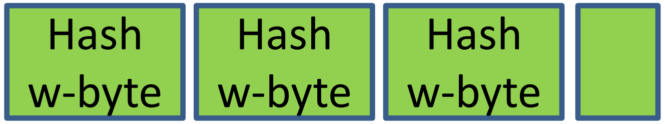
+ Simple selection criteria and tunable ⇒ fast and adaptive
– A small insertion/deletion in content will result in failure in detecting redundancy ⇒ lower bandwidth savings

# 1. Fingerprinting: SAMPLEBYTE

- Can we get the speed/adaptability of FIXED and the robustness of MODP?



*Choose marker if F(singlebyte) = 1; e.g., F(0) = 1, F(5) =1*
*Once chosen, **skip p/2** bytes*

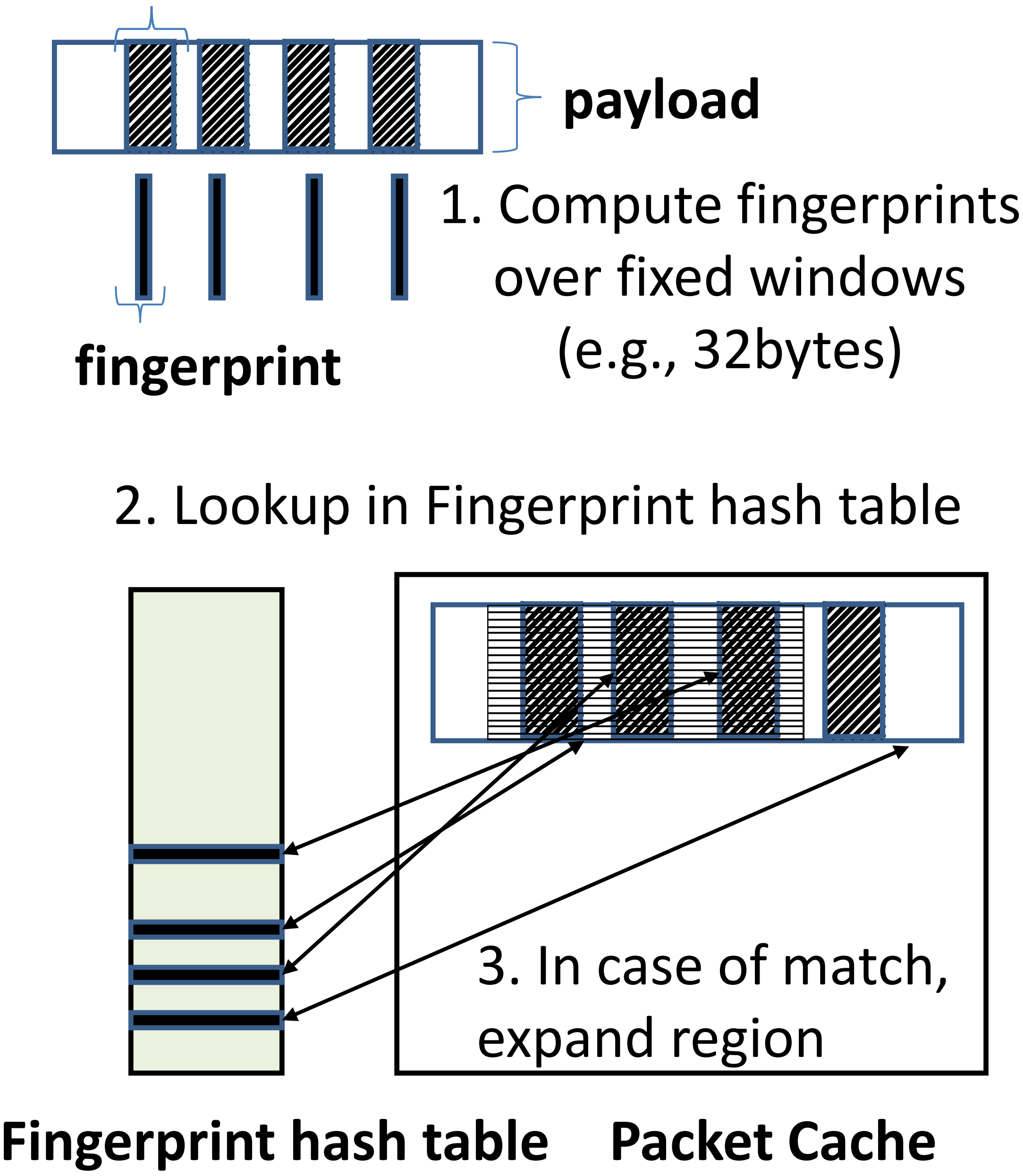


*Fingerprints*

- F(singlebyte) derived from training data using a greedy strategy

+ Content-based ⇒ bandwidth savings close to MODP?
+ Simple selection & tunable skipping ⇒ speed/adaptability of FIXED?

# 2. Matching & Encoding: Max-Match



payload

fingerprint

1. Compute fingerprints over fixed windows (e.g., 32bytes)

2. Lookup in Fingerprint hash table

3. In case of match, expand region

**Fingerprint hash table**

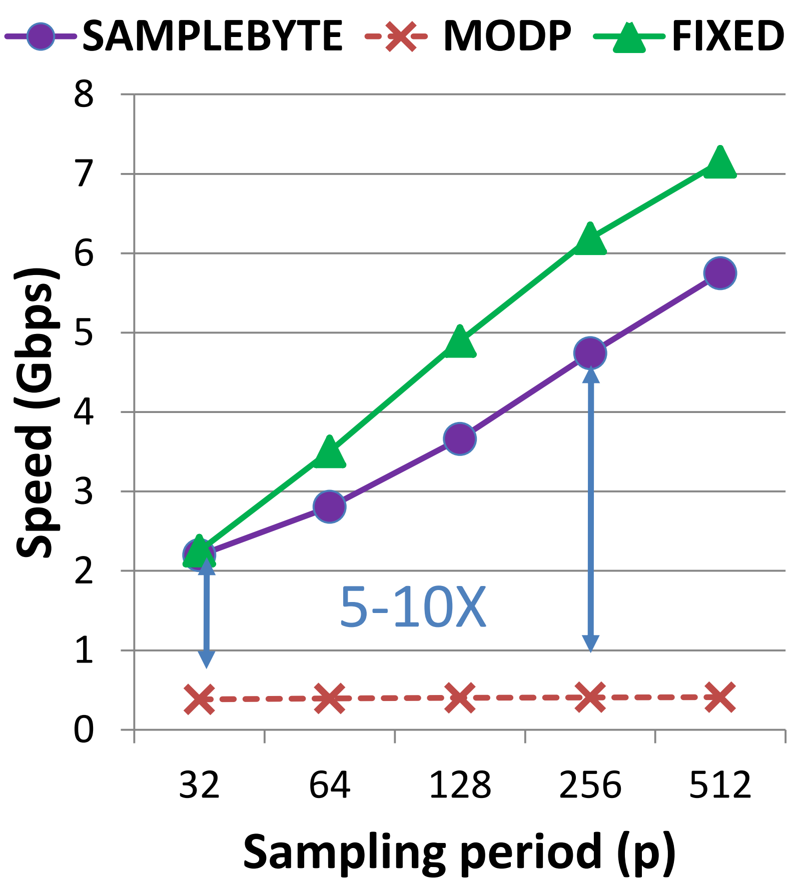
**Packet Cache**

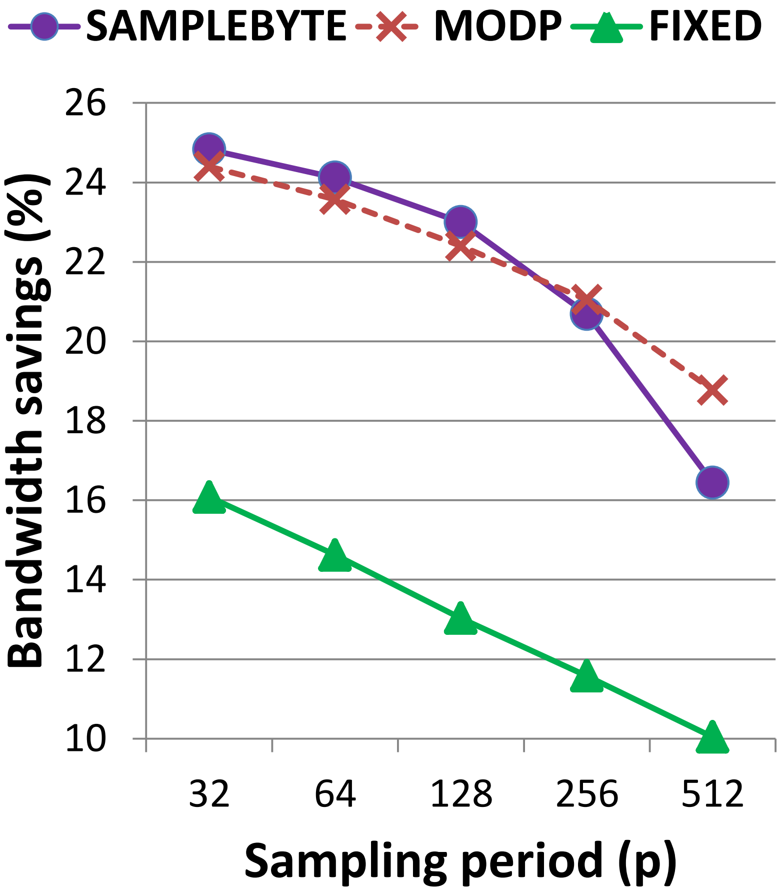
- Approach used in Spring & Wetherall
  - Meta data overhead is 67% of cache size
- Collisions are not costly
  - Simple hash function
  - Overwrite hash table
  - No deletion
- Don’t store fingerprints!
  - Use the table index to implicitly represent part/all of fingerprint
- Meta data overhead is 6-12% of cache size

# Outline

- Overview
- Design of EndRE
- **EndRE costs and benefits**
- Summary

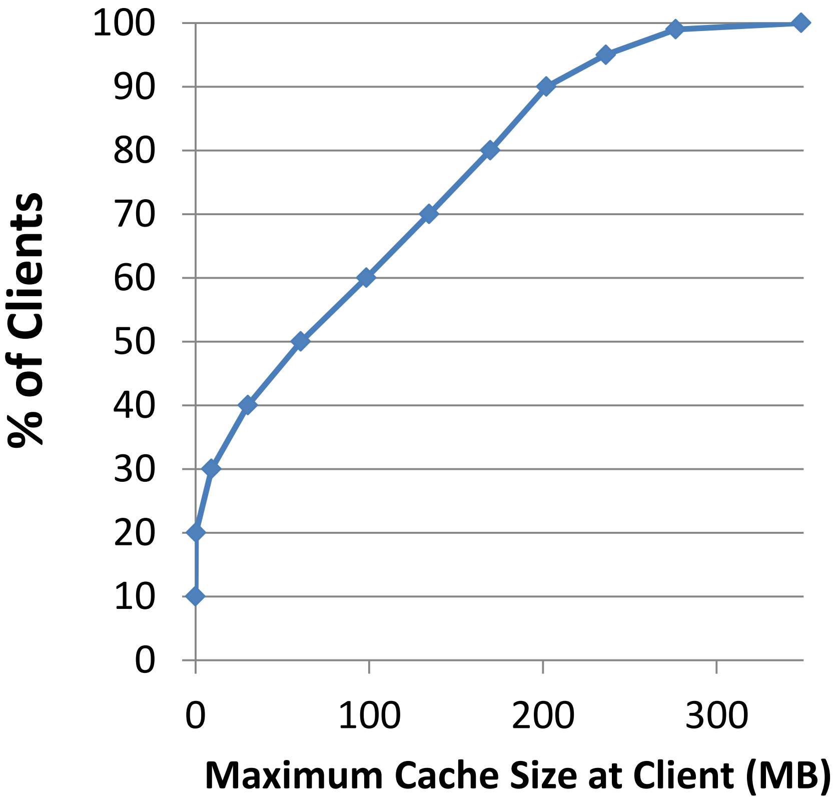
# Fingerprinting Algorithms: Comparison

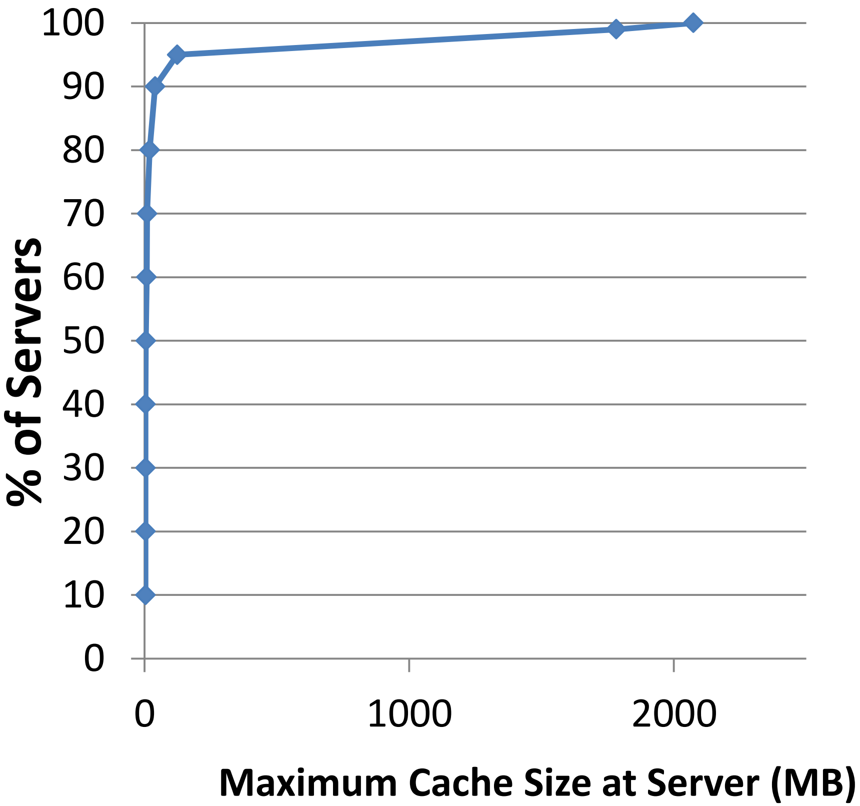




- SAMPLEBYTE delivers bandwidth savings similar to MODP while operating at speeds similar to FIXED

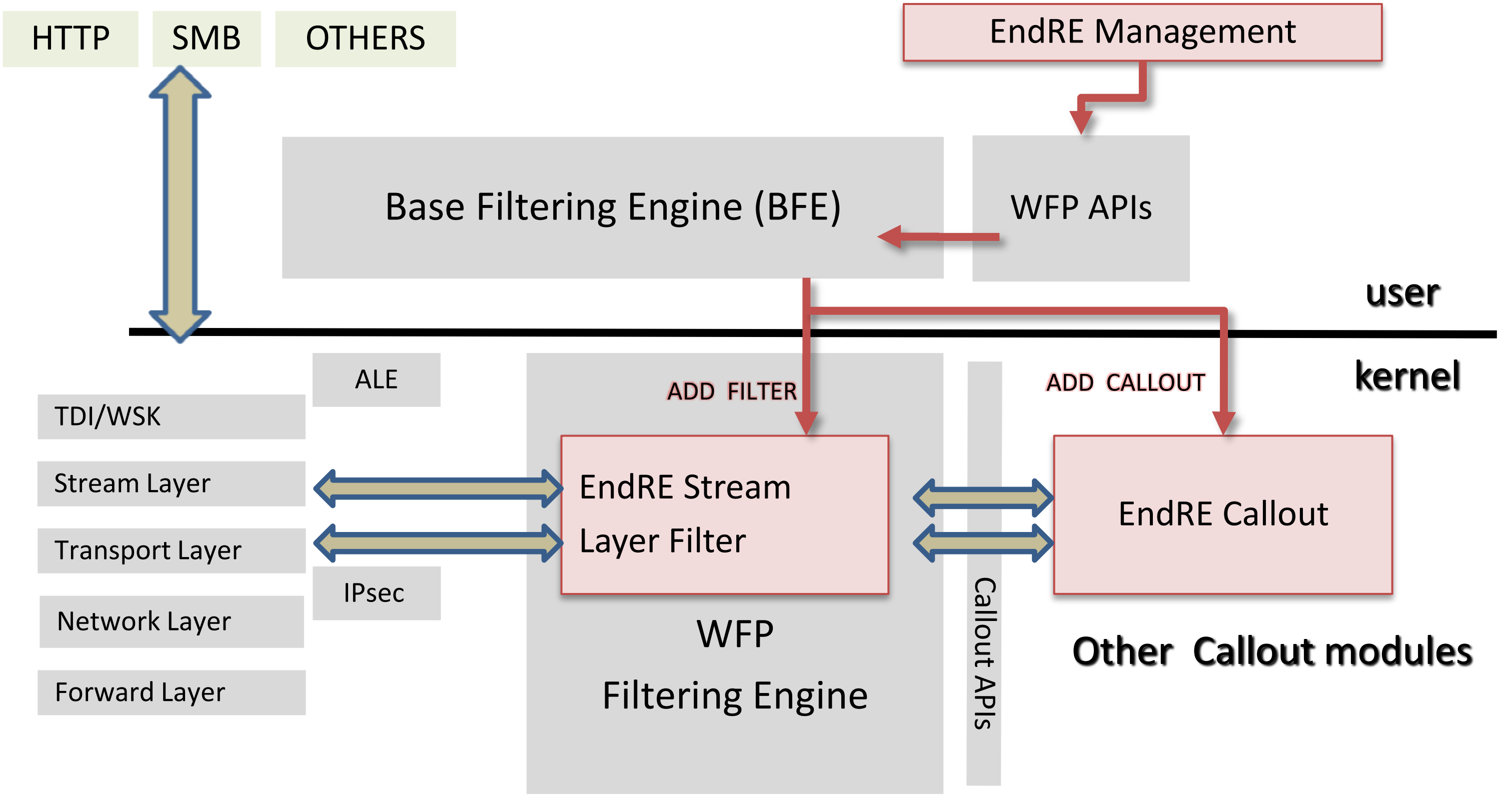
# EndRE Memory Requirements: 44-day 11-enterprise Trace Analysis





- ➢ Median/Max memory requirement at Client is 60/360MB
- ➢ Memory requirement at server tunable, at cost of reduced savings

# Implementation



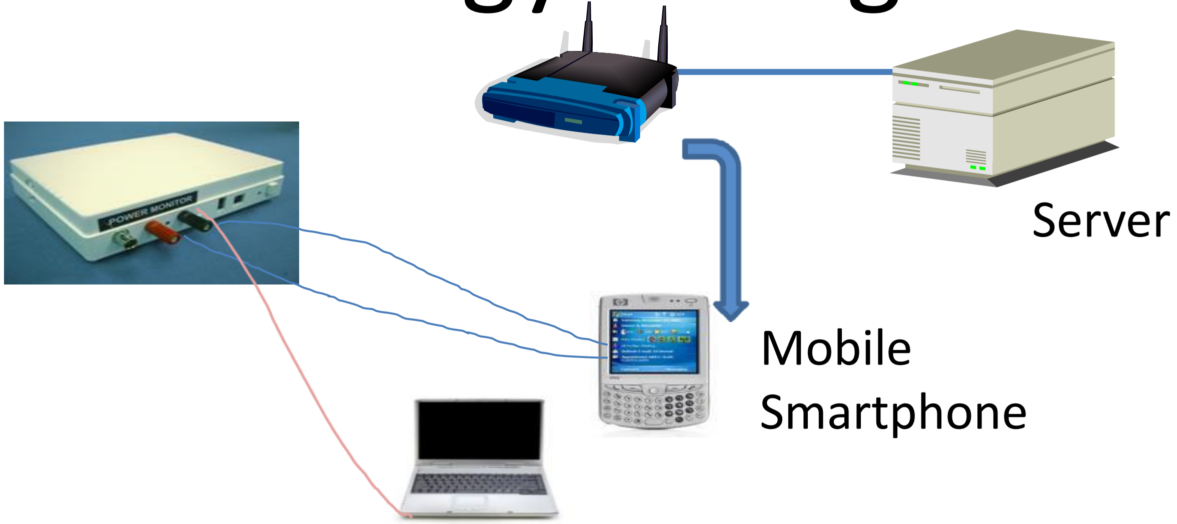
- EndRE pilot deployment on laptops/desktops over one week with 11 users for HTTP traffic (1.7GB) delivered bandwidth savings of 31%

# Bandwidth Savings (~2 weeks)

| Enterprise Site | Trace Size (GB) | Middle (2GB) % savings | EndRE (1-10 MB) % savings | Middle + large-files %savings | EndRE + large-files % savings |
|---|---|---|---|---|---|
| 1. | 173 | 71 | 47 | 72 | 56 |
| 2. | 8 | 33 | 24 | 33 | 33 |
| 3. | 71 | 34 | 26 | 35 | 32 |
| 4. | 58 | 45 | 24 | 47 | 30 |
| 5. | 69 | 39 | 27 | 42 | 37 |
| 6. | 80 | 34 | 22 | 36 | 28 |
| 7. | 80 | 31 | 26 | 33 | 33 |
| 8. | 142 | 34 | 22 | 40 | 30 |
| 9. | 198 | 44 | 16 | 46 | 26 |
| 10. | 117 | 27 | 21 | 30 | 30 |
| **Avg/Site** | **100** | **39** | **26** | **41** | **34** |

➢ EndRE delivers average bandwidth savings of 26-34%, a significant portion of the 39-41% savings of middlebox

# Energy Savings



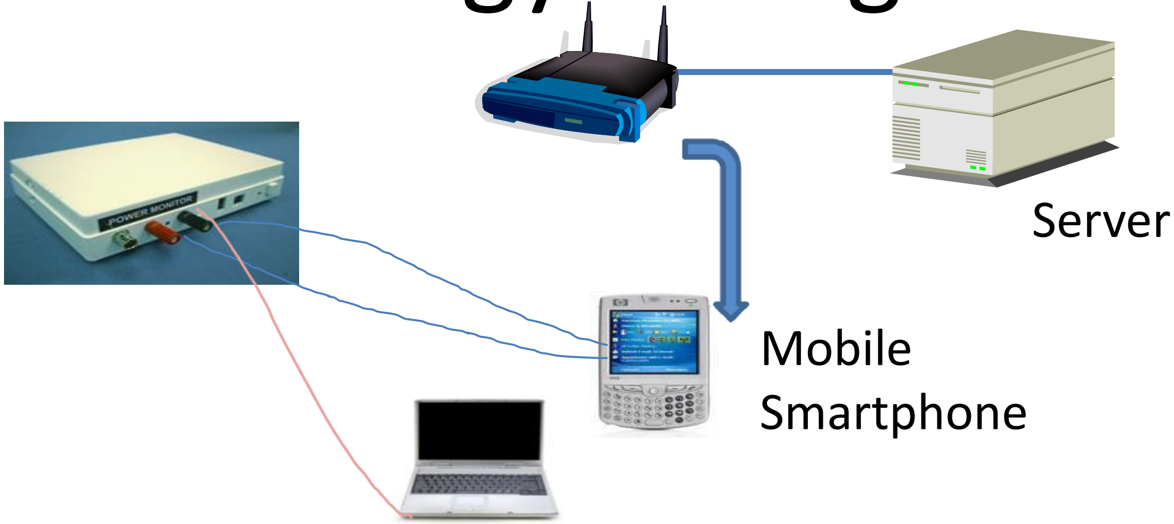
Server

Mobile Smartphone

| | None | ZLIB (LZ) | | | | EndRE | |
|---|---|---|---|---|---|---|---|
| | Energy uAh | Energy % savings | | Bandwidth % savings | | Energy %savings | Bandwidth %savings |
| Trace | | Packet | 32KB | Packet | 32KB | Packet | Packet |
| A | 2038 | -11 | 42 | 26 | 44 | 25 | 29 |
| B | 1496 | -11 | 68 | 41 | 75 | 70 | 76 |

- ZLIB works well for large chunk sizes but on a packet-by-packet basis may result in increased energy consumption

# Energy Savings



| | None | ZLIB (LZ) | | | | EndRE | |
|---|---|---|---|---|---|---|---|
| | Energy<br>uAh | Energy<br>% savings | | Bandwidth<br>% savings | | Energy<br>%savings | Bandwidth<br>%savings |
| Trace | | Packet | 32KB | Packet | 32KB | Packet | Packet |
| A | 2038 | -11 | 42 | 26 | 44 | 25 | 29 |
| B | 1496 | -11 | 68 | 41 | 75 | 70 | 76 |

- ➢ EndRE's bandwidth savings translate into equivalent savings in energy with no additional latency

# Related work

- Static content (e.g., large files)
  - *Host:* Disk De-Duplication
  - *Client and Server:* LBFS (SOSP’01), RSYNC/RDC
  - *Peer-to-Peer: DOT(NSDI’06),* SET (NSDI’07), BranchCache in Win7
- Dynamic content
  - Middlebox
    - Spring & Wetherall (SIGCOMM’00)
    - Products from Riverbed, Cisco, Juniper, etc.
- New architectures
  - Packet Caches: RE in routers (SIGCOMM’08)
  - Ditto: RE in wireless mesh networks (MobiCom’08)

# Summary

1. EndRE
   - SAMPLEBYTE fingerprinting algorithm supports processing speeds of 1.5-4Gbps/core
   - Data structure optimizations reduce server memory requirement by 33-75%
2. Costs
   - Client processing negligible; Server processing is load adaptive;
   - Median client requires only 60MB of memory; Server up to 2GB
3. Benefits
   - Avg. bandwidth savings of 26-34%
   - Bandwidth savings →equivalent energy savings on smartphones

➢ EndRE is a promising alternative to WAN optimizers

# Questions?