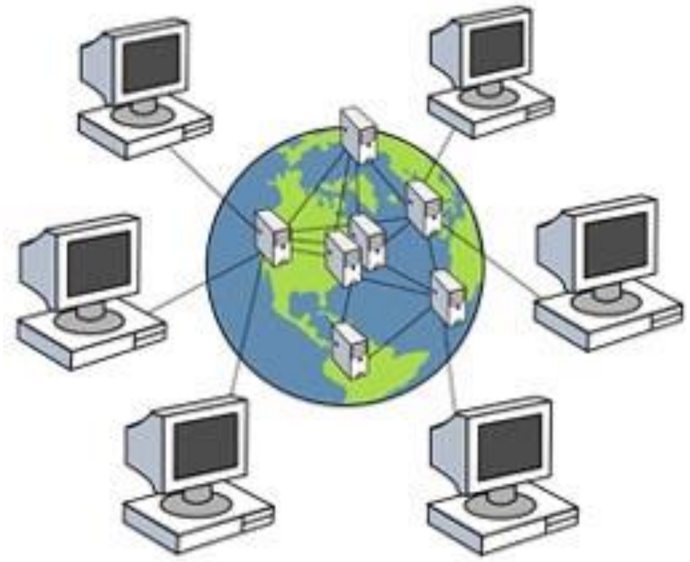
EndRE: An End-System Redundancy Elimination Service For Enterprises

Bhavish Aggarwal, Aditya Akella, Ashok Anand1, Athula Balachandran1, Pushkar Chitnis, Chitra Muthukrishnan, Ramachandran Ramjee and George Varghese
Microsoft Research India; University of Wisconsin-Madison; CMU; UCSD

SLIDES BY SHIMON AZULAY & AYAL MITTELMAN

Introduction

- Nowadays, network services have reached a global scale in enterprise space
- Data between clients is transferred over WAN
- Data should be transferred from end to end clients quickly and efficiently for better user experience



Globalization And Network Service Example

- In a global scale corporation, branch offices can be found all over the globe
- Where should they locate their servers?

What is better?

Example – cont.

Tradeoff:

- Servers that are located near the clients are much more efficient in data exchange, but the operational costs for them are high.
- Servers at a small number of locations can lower administration costs, but increase network costs and latency

Middlebox & Protocol Independent redundancy elimination

We use Middleboxes

- Performance in WAN communication
- One box detects chunks of data that match entries in its cache (by computing fingerprints), it encodes matches using tokens
- Box at the far end reconstructs original data using its own cache and the tokens

Middle box Drawbacks

- Encrypted data:
  - Encrypted data could not be found in the middlebox cache, although the decrypted data exists in the cache.
  - Data encrypted, then was sent to middlebox, which need to decrypt it- Not safe and redundant.

- Usage of mobile devices:
  - Token reached the middlebox, found in the cache and data and was reconstructed. Now the bottleneck is between the router and the mobile phone.

End-System Redundancy Elimination

In this presentation we will explore end-system redundancy elimination service called EndRE.

EndRE could supplement or supplant middle box-based techniques while addressing their drawbacks.

We will examine the changes in design and implementation in order to support EndRE .

EndRE– Design Goals

We will examine five design goals for the new approach:

- ❑Transparent operation
- ❑Fine-grained operation
- ❑Simple decoding at clients
- ❑Fast and adaptive encoding at servers
- ❑Limited memory footprint at servers and clients

EndRE design-
Server And Client

EndRE Design is divided in to two modules:

- Server
  - Encoding the redundant data with shorter meta-data
  - Meta-data is essentially a set of <offset, length>
  - The meta-data computed with respect to the client-side cache

EndRE design-
Server And Client

EndRE Design is divided in to two modules:

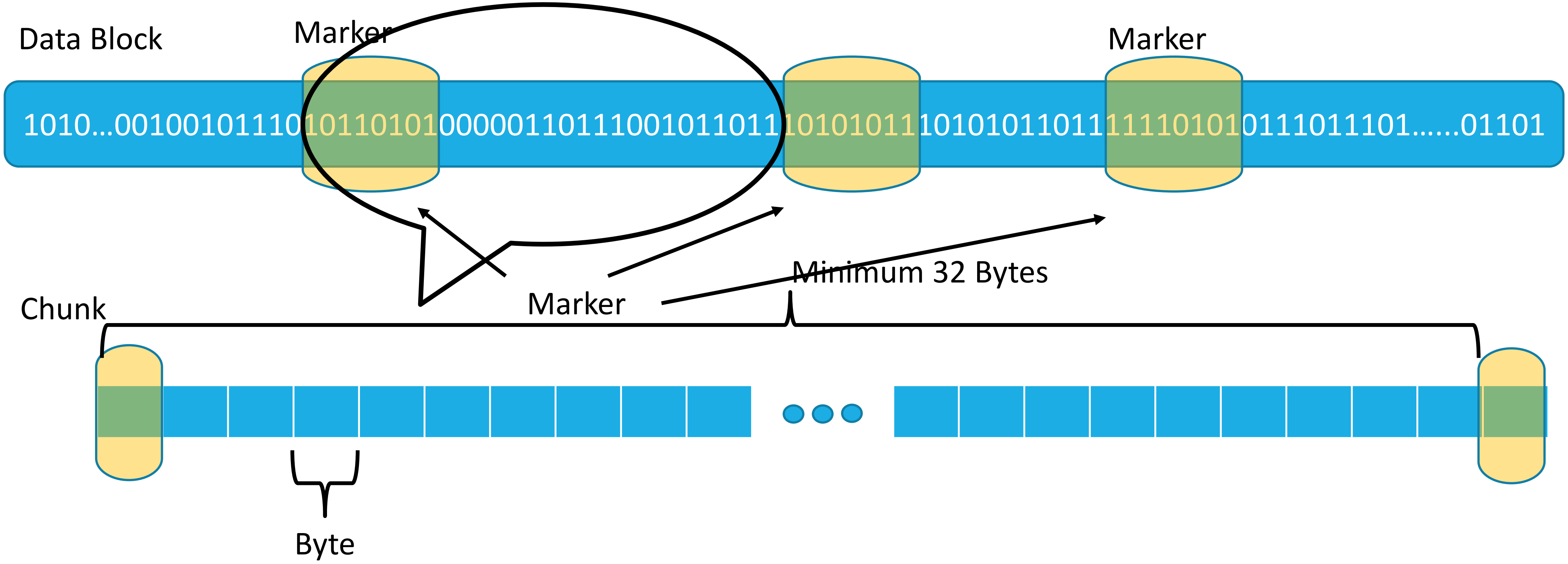
- Client
  - Consist Simple logic to decode the meta-data by “de-referencing” the offsets sent by the server.

EndRE design-
Handle Redundancy

For handling the redundancy we need to do two steps:

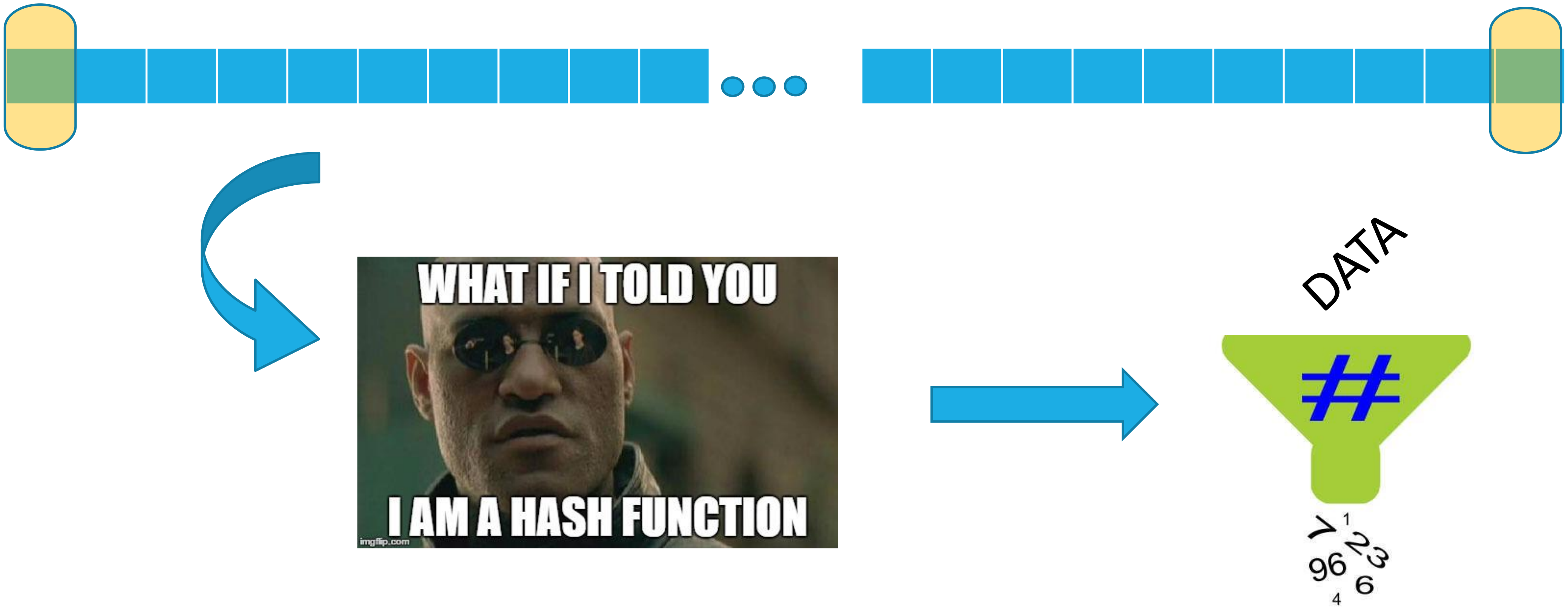
- Fingerprinting (4 Approaches):
  - MODP
  - MAXP
  - FIXED
  - SAMPLEBYTE
- Matching and Encoding (2 Approaches):
  - Chunk-Match
  - Max-Match

EndRE-Terminology

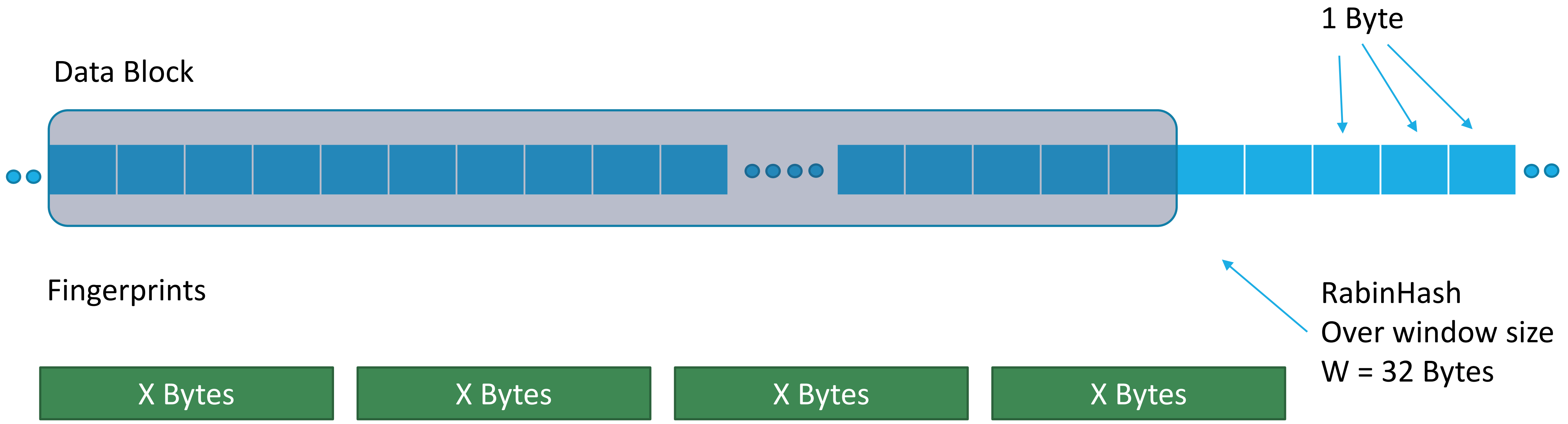


EndRE- Terminology

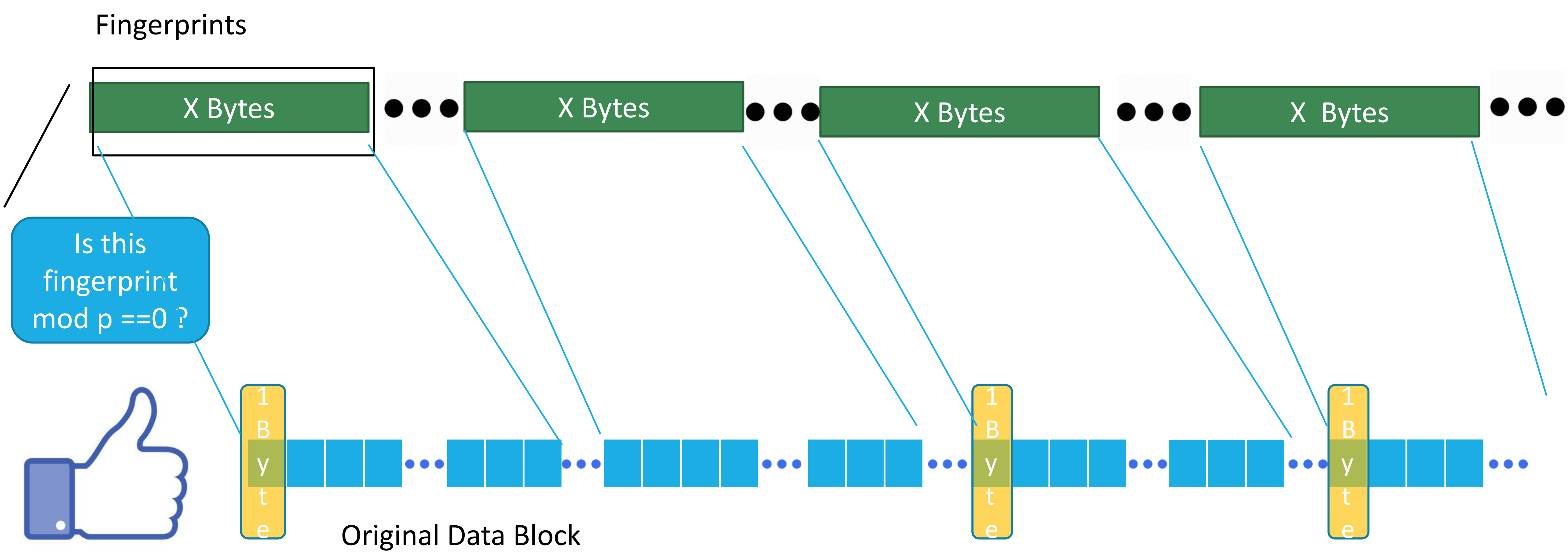
Chunk



EndRE design- MODP Fingerprinting



EndRE design-
MODP Fingerprinting

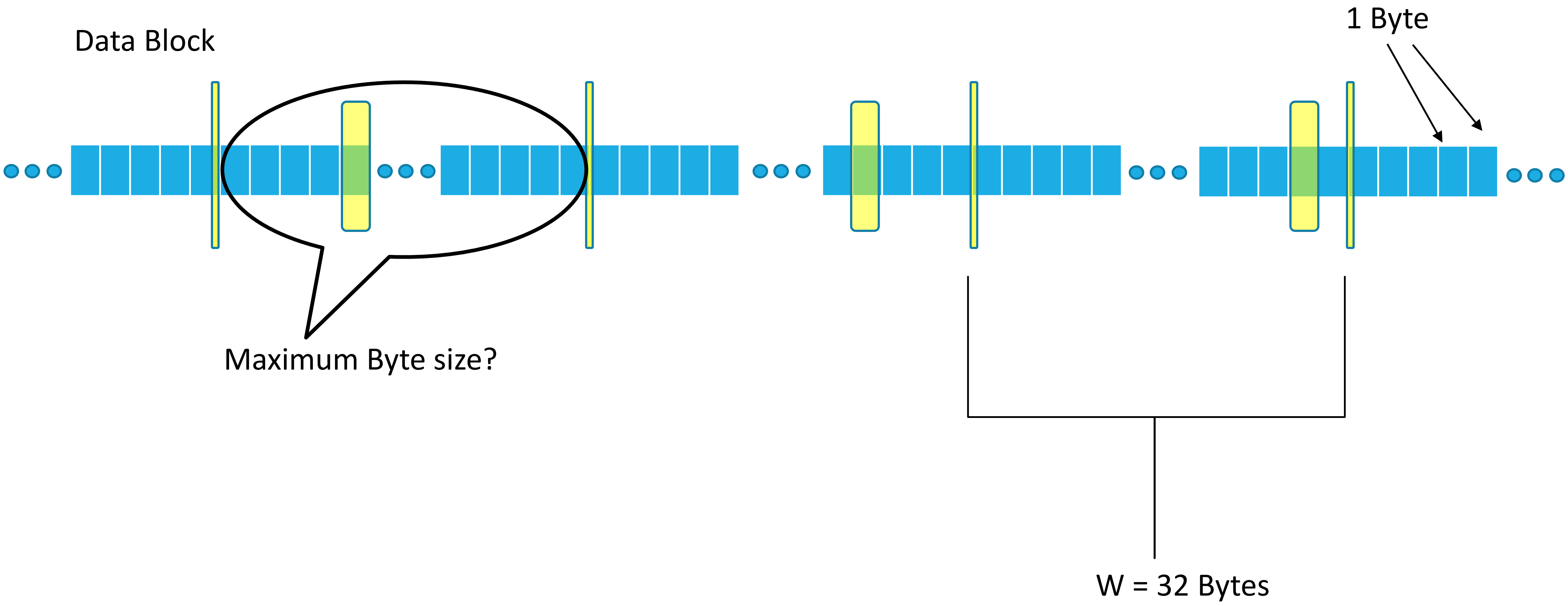


EndRE design-
MODP Fingerprinting – Cont.

- ❑ Content based

- ❑ Expensive computational operations
- ❑ Over/under sampling

EndRE design-
MAXP Fingerprinting

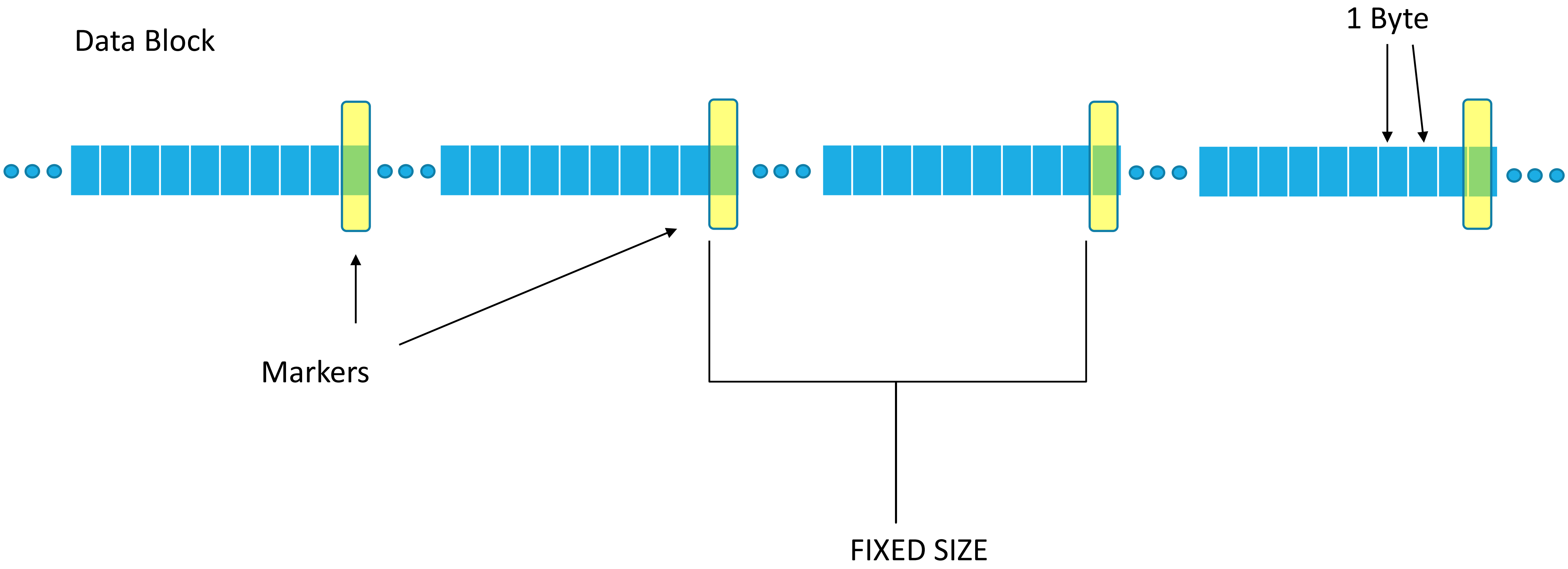


EndRE design- MAXP Fingerprinting – Cont.

- ❑ Content based
- ❑ No over/under sampling

- ❑ Expensive computational operations

EndRE design- FIXED Fingerprinting

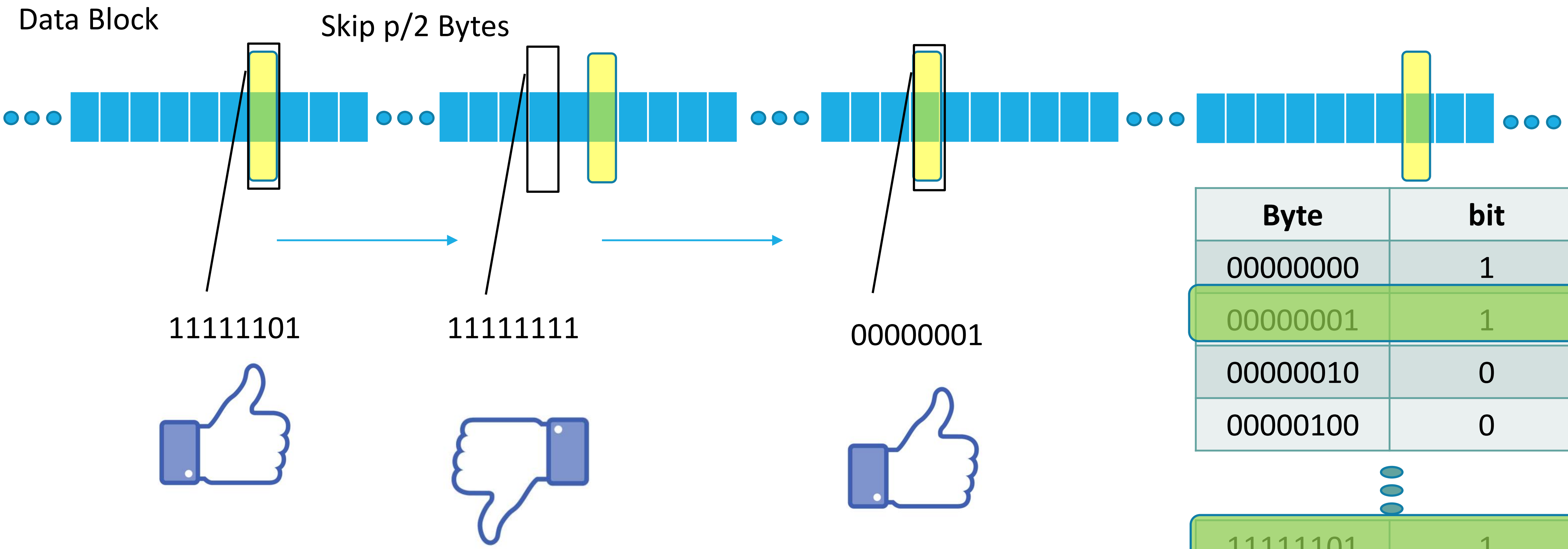


EndRE design-
MODP Fingerprinting – Cont.

- ❑ Cheap computational operations
- ❑ No over/under sampling

- ❑ Not robust to small changes

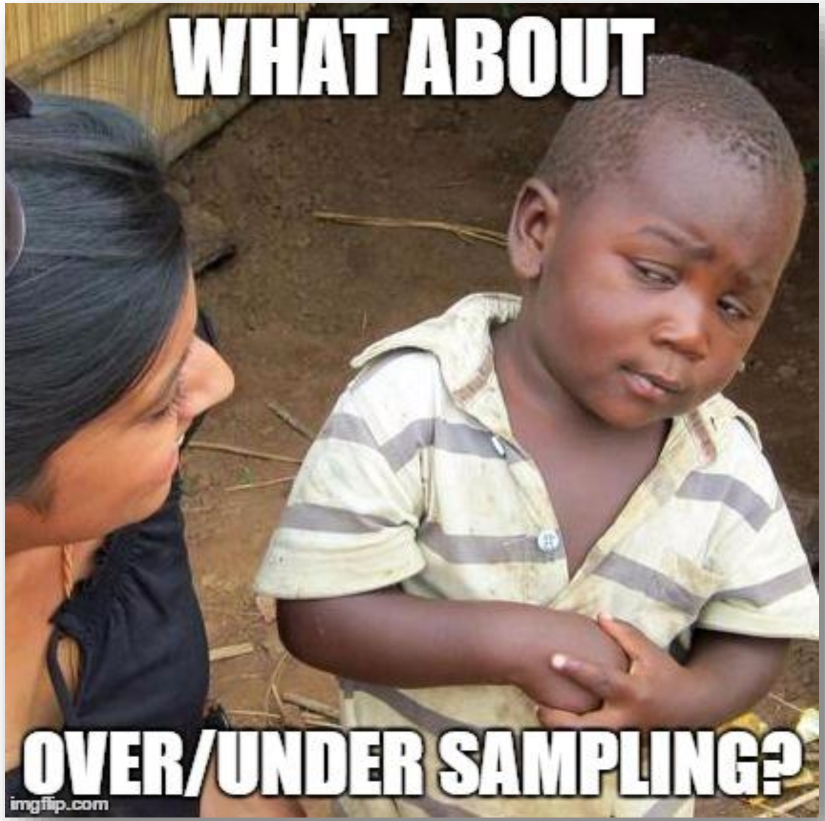
EndRE design- SAMPLEBYTE Fingerprinting



Byte	bit
00000000	1
00000001	1
00000010	0
00000100	0
...	
11111101	1
11111110	0
11111111	0

EndRE design-
SAMPLEBYTE Fingerprinting – Cont.

- ✓ Content-Based
- ✓ Computationally efficient



SAMPLEBYTE Fingerprinting Over/Under sampling

- Skips P/2 bytes after match
- Table is built in a way that match will be occur every 1/P bytes

SAMPLEBYTE Fingerprinting- Creating The Entry Table

We build static lookup table:

- Use Network traces from one of the enterprise sites
- Run MAXP to identify redundant content
- Sort characters in descending order of their presence in the identified redundant content
- Set the first x to 1

EndRE design-
Matching And Encoding

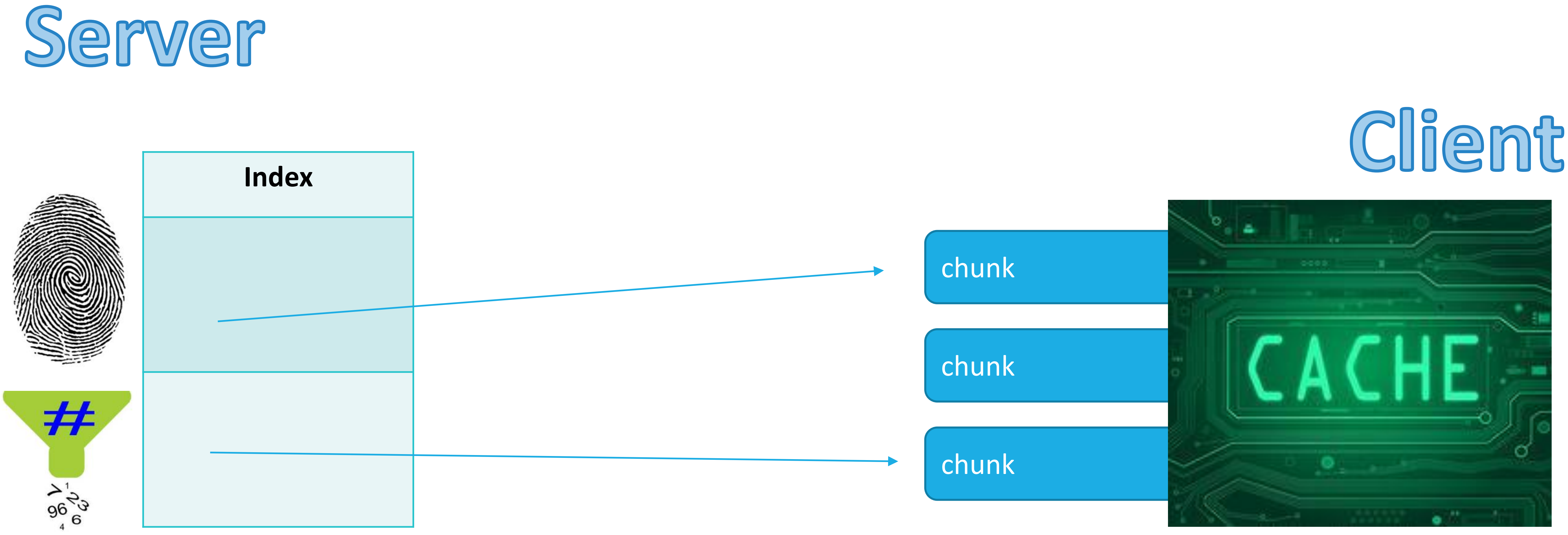
We examine 2 Approaches:

- Chunk-Match
- Max-Match

For both approaches we will try to:

- Move computationally operations & memory management tasks to the server
- Exploit inherent structure within the data to optimize memory usage

EndRE design-
Matching Overview



Matching And Encoding –

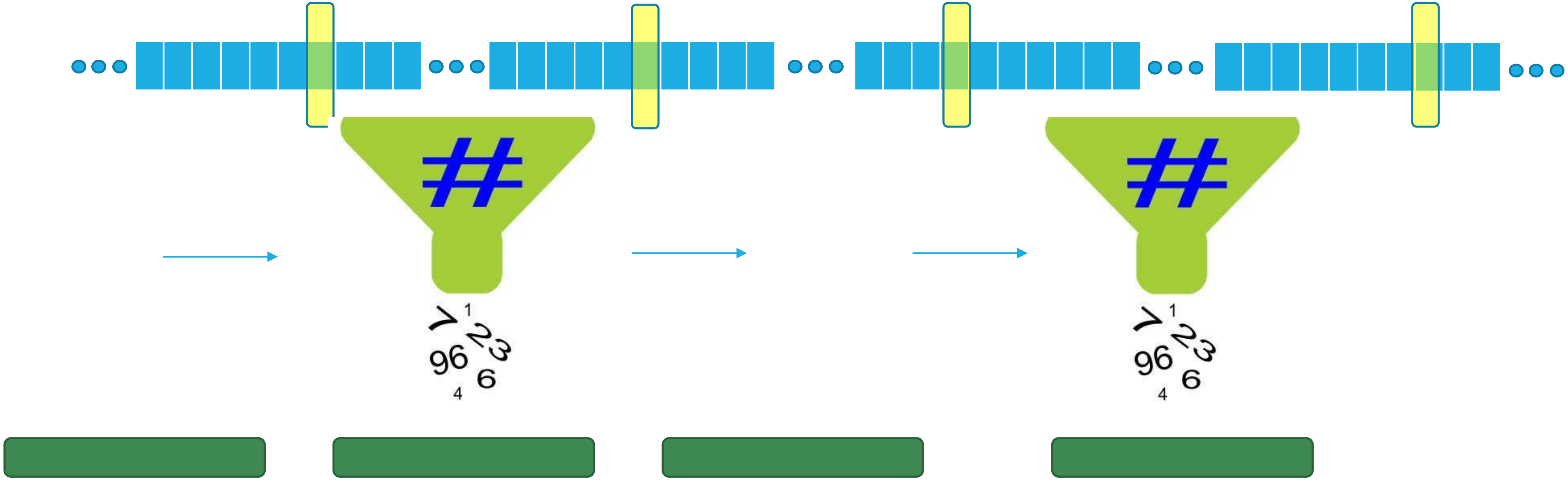
In other systems:

- Client saves hash chunk mapping
- Server sends the hash to the client
- Server holds chunks

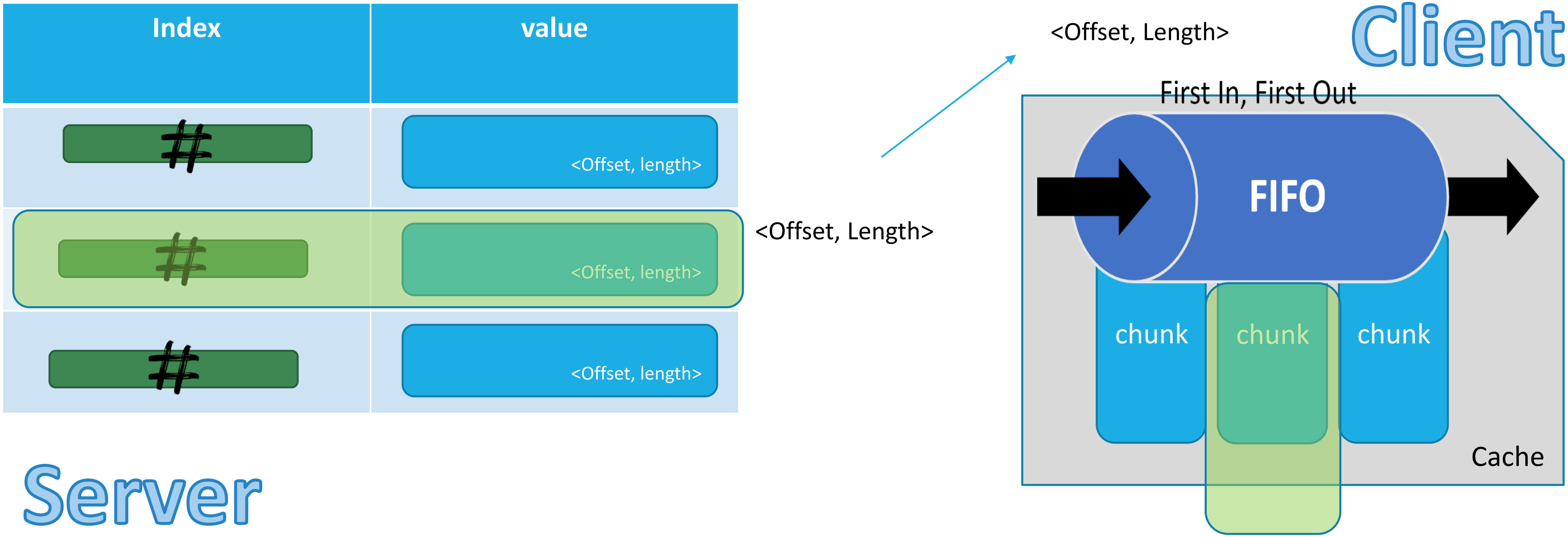
In EndRE:

- Client hold simple circular FIFO cache. Doesn’t hold the hash function
- Server hold hash - <offset, length> table
- Server sends <offset, length> tuple

Matching And Encoding – Chunk-Match



Matching And Encoding – Chunk-Match



Chunk Match - Optimization

Assume:

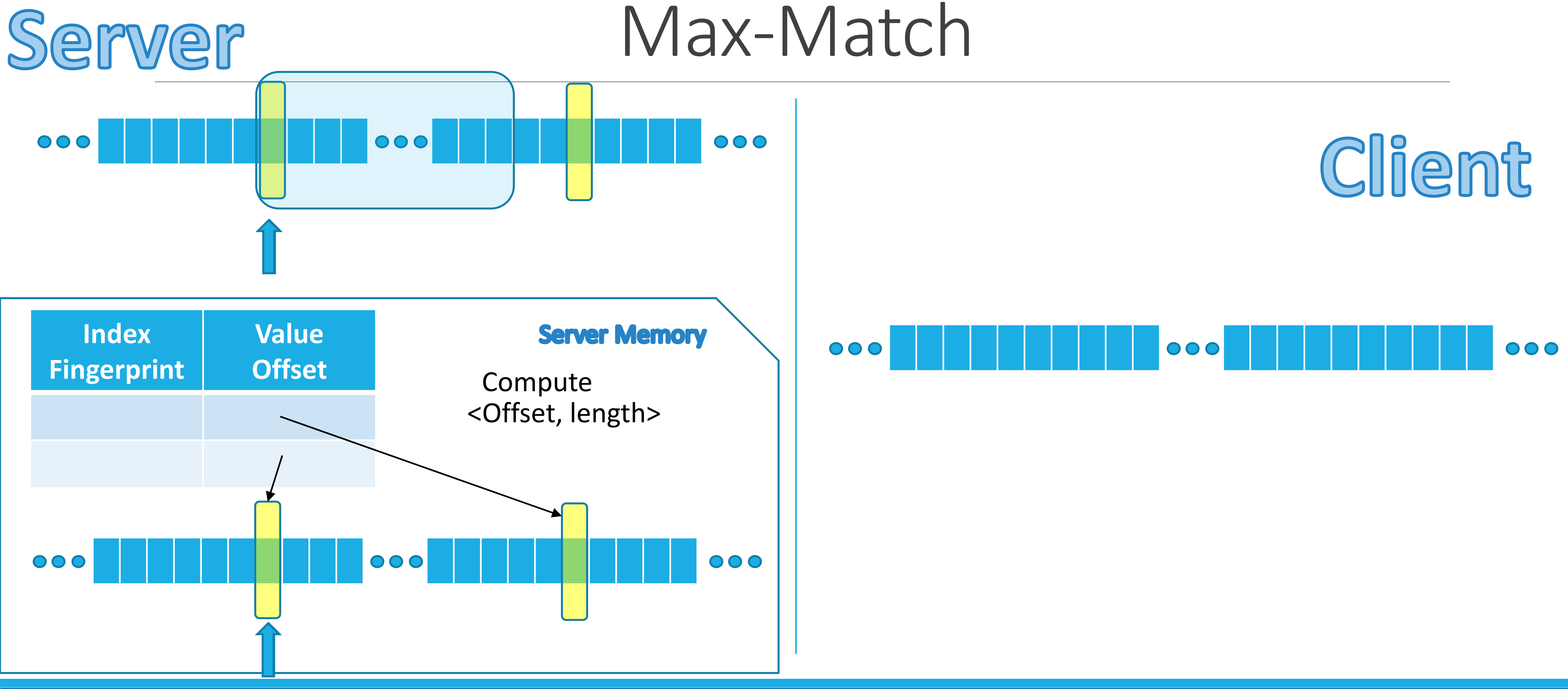
- P = 64
- Cache Size (client) = 16MB = 2^24 Bytes
- Maximum Chunk Size = 256 Bytes

We only need to store:

< SHA1 ,	Offset ,	Length >
20 Bytes	3 Bytes	1 Bytes

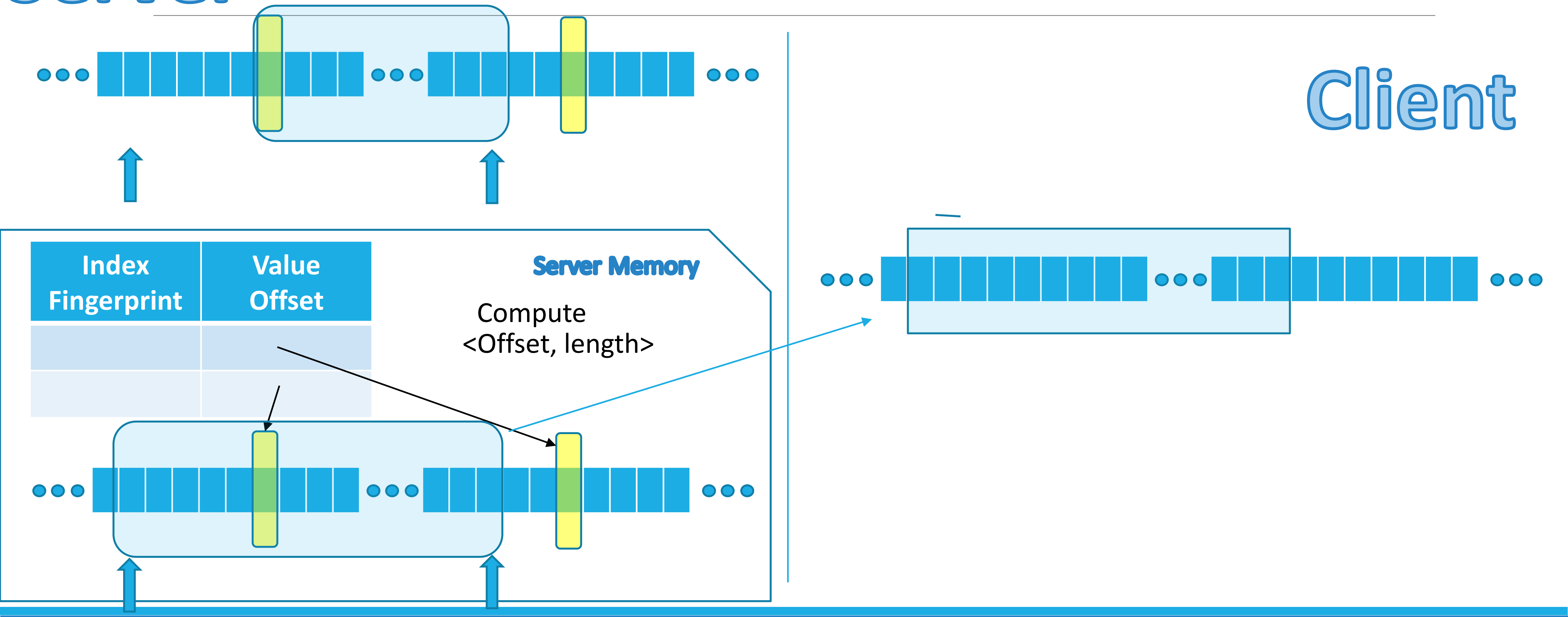
Server Holds 38% of the client cache size

Matching And Encoding – Max-Match



Matching And Encoding – Max-Match

Server



Max-Match - Optimization

- Client cache size of 16MB = 2^24
- P = 64 = 2^6 bytes
- 2^18 fingerprints
- Add additional 8 bits to fingerprint index column
- Server holds table of size 6% of the client cache size
- Server holds in total 106% of the client cache size

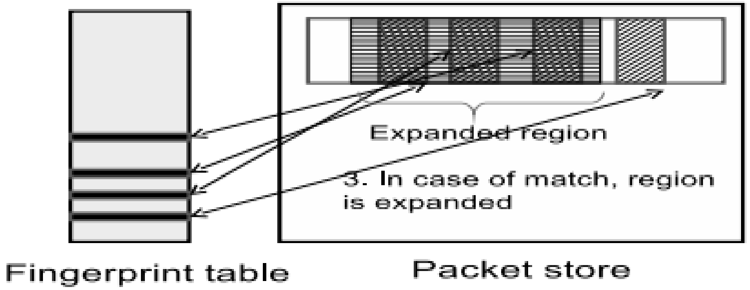


Figure 5: Max-Match: matched region is expanded

index (implicit fingerprint, 18 bits)	fingerprint remainder (8 bits)	offset (24 bits)
0	...	...
...	...	...
$2^{18} - 1$	...	...

Implementation - Socket Layer Above TCP

Benefits of implementing EndRE at the socket layer above TCP:

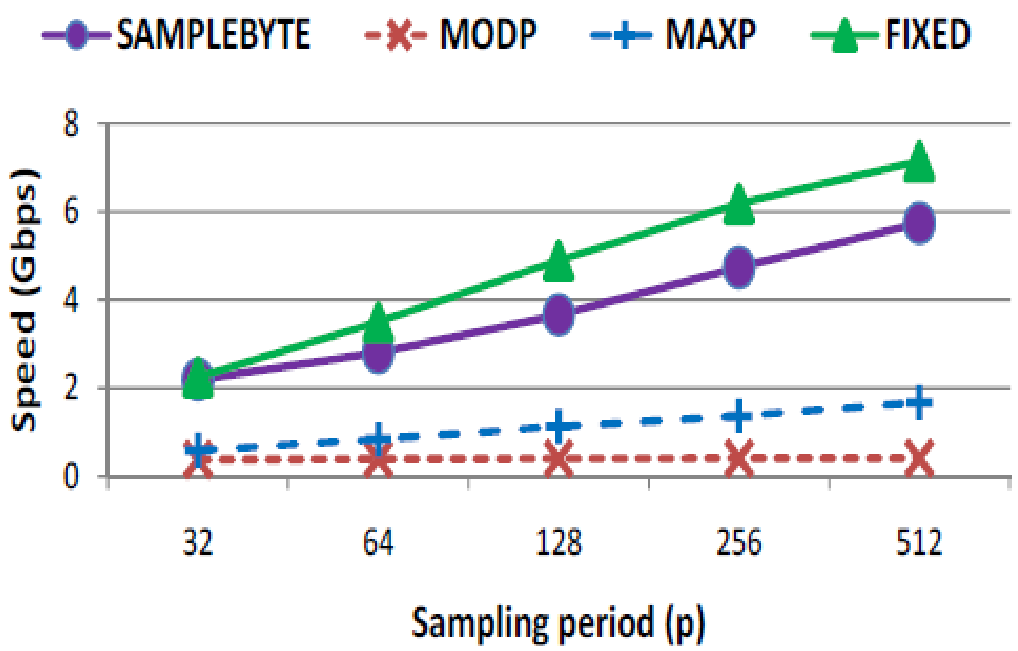
- Latency – Reduce the number of packets
- Encryption – Can be compressed before encryption
- Cache Synchronization: TCP ensure reliable in-order delivery. However, TCP connections may get reset in the middle of a transfer. 2 Solutions:
  - Pessimistic
  - Optimistic

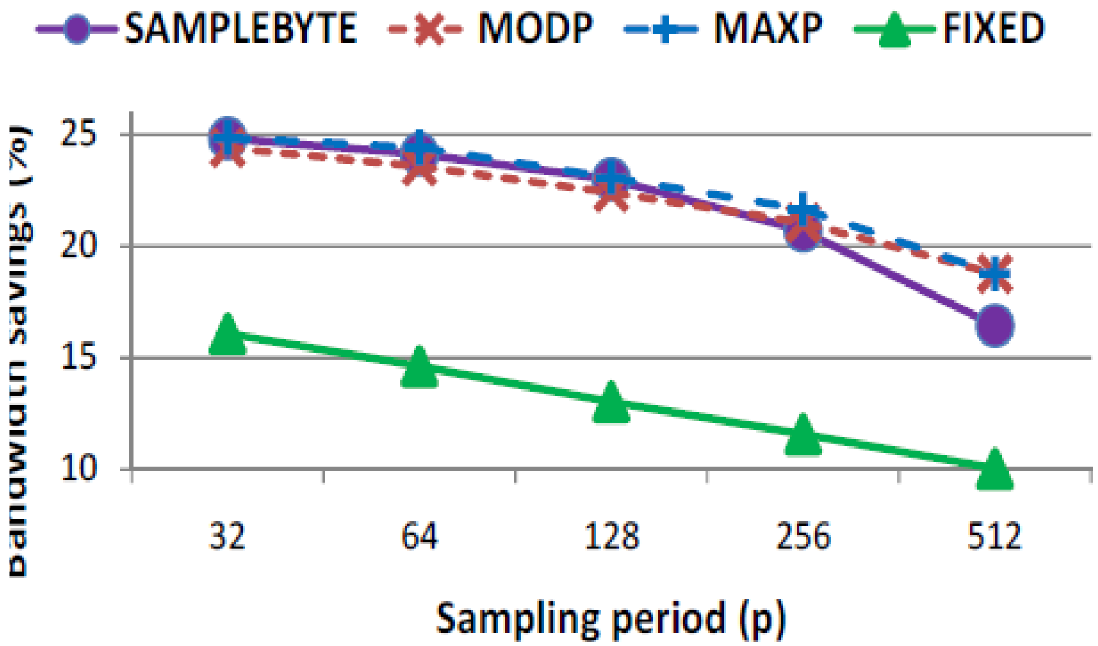
Evaluation

- 11 corporate enterprise locations (classified as small, medium or large)
- Small pilot deployment (15 laptops) in their lab

Trace Name (Site #)	Unique Client IPs	Dates (Total Days)	Size (TB)
Small Enterprise (Sites 1-2)	29-39	07/28/08 - 08/08/08 (11) 11/07/08 - 12/10/08 (33)	0.5
Medium Enterprise (Sites 3-6)	62-91	07/28/08 - 08/08/08 (11) 11/07/08 - 12/10/08 (33)	1.5
Large Enterprise (Sites 7-10)	101-210	07/28/08 - 08/08/08 (11) 11/07/08 - 12/10/08 (33)	3
Large Research Lab (Site 11, training trace)	125	06/23/08 - 07/03/08 (11)	1

Server CPU And Bandwidth Costs





CPU Costs - Server

Max-Match $p \rightarrow$	Fingerprint		InlineMatch		Admin	
	32	512	32	512	32	512
MODP	526.7	496.7	9.6	6.8	4.8	0.6
MAXP	306.3	118.8	10.1	7.7	5.2	0.5
FIXED	69.4	14.2	7.1	4.7	4.7	0.4
SAMPLEBYTE(SB)	76.8	20.2	9.5	6.1	3.0	0.7

CPU Time(s) for different algorithms

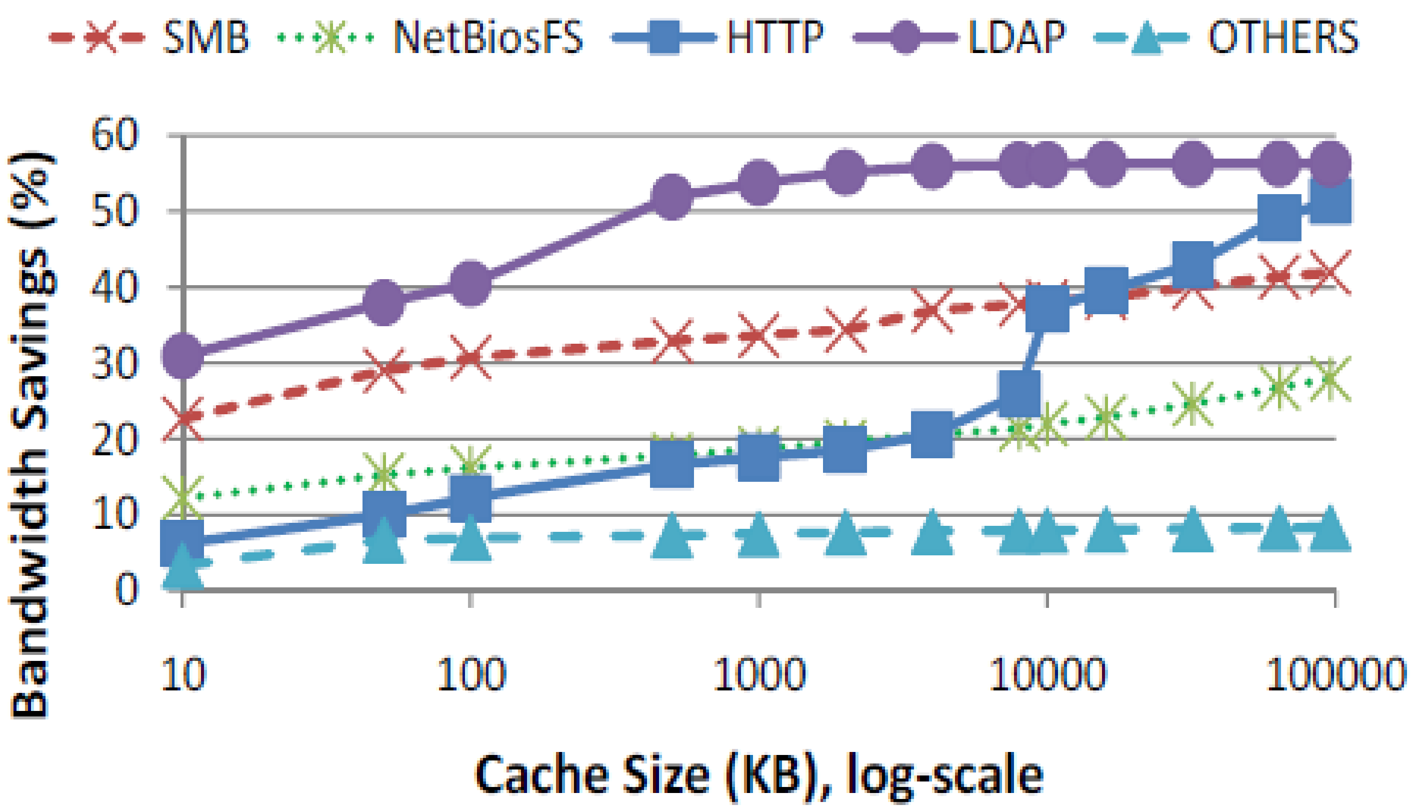
Memory Costs

Two key questions:

1. What is the cache size limit between a single client – server pair?
2. Given the cache size limit for one pair, what is the cumulative memory requirement at clients & sever

- We will we examine the trade-off between cache sizes and bandwidth savings

Memory Costs



Memory Costs

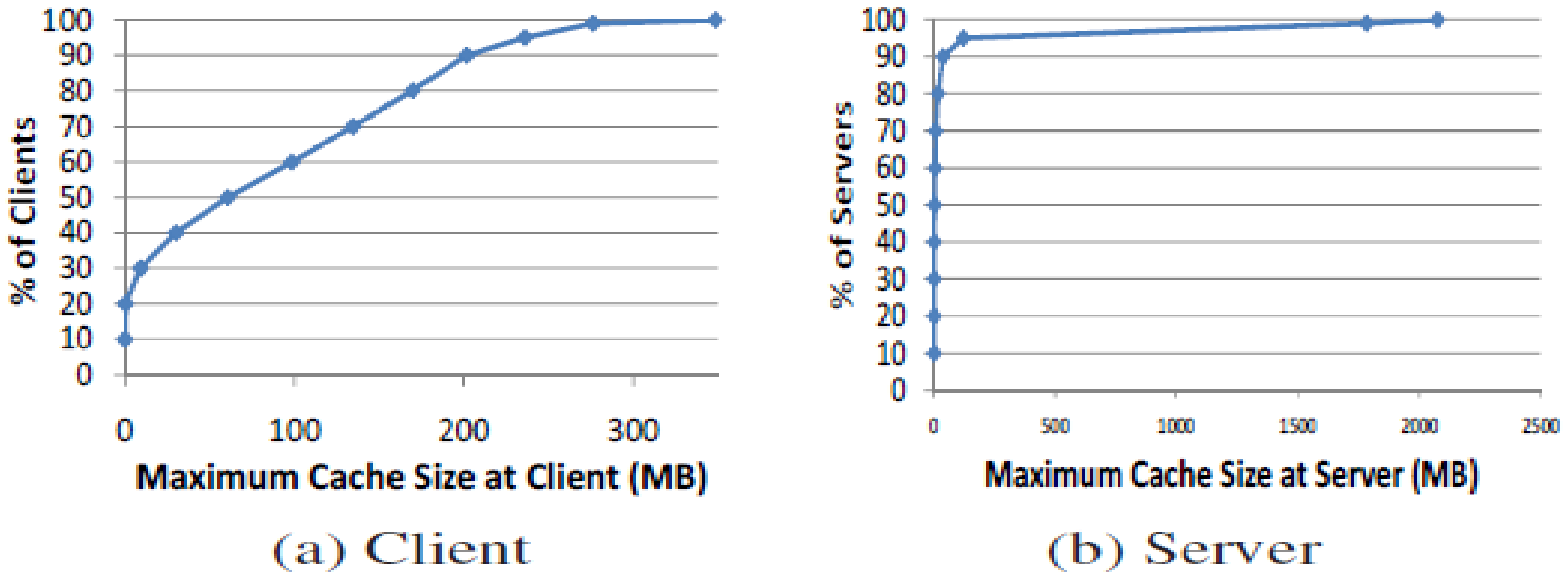
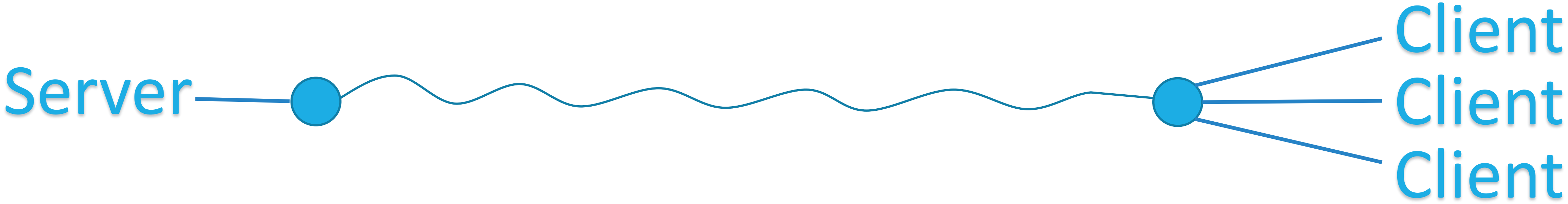


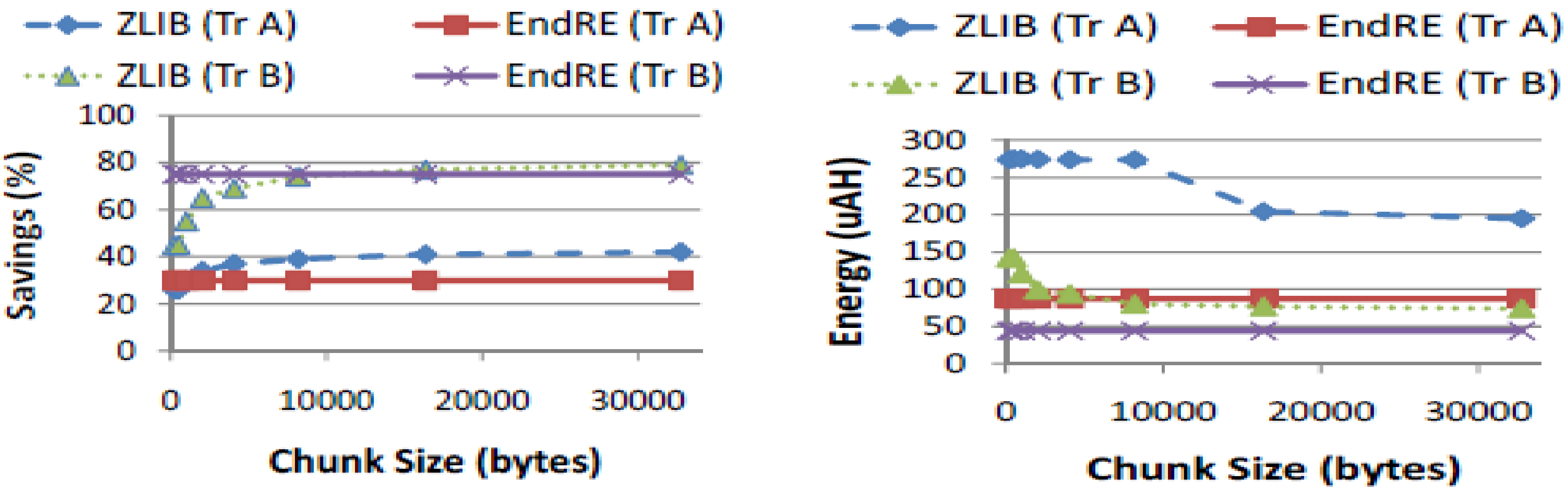
Figure 10: Cache scalability

Bandwidth Savings

Site	Trace Size GB	GZIP 10ms	EndRE Max-Match 10MB				EndRE Max-Match+GZIP 10MB	EndRE Chunk-Match 10MB	EndRE Max-Match + DOT 10MB	IP WAN-Opt Max-Match 2GB	IP WAN-Opt Max-Match + DOT 2GB
		% savings									
			MODP	MAXP	FIXED	SB	SB	MODP	SB	SB	SB
1	173	9	47	47	16	47	48	46	56	71	72
2	8	14	24	25	19	24	28	19	33	33	33
3	71	17	25	26	23	26	29	22	32	34	35
4	58	17	23	24	20	24	31	21	30	45	47
5	69	15	26	27	22	27	31	21	37	39	42
6	80	12	21	21	18	22	26	17	28	34	36
7	80	14	25	25	22	26	30	21	33	31	33
8	142	14	22	23	18	22	28	19	30	34	40
9	198	9	16	16	14	16	19	15	26	44	46
10	117	13	20	21	17	21	25	17	30	27	30
Avg/site	100	13	25	26	19	26	30	22	34	39	41



Energy Savings



(a) Compression Savings

(b) Energy

Questions?