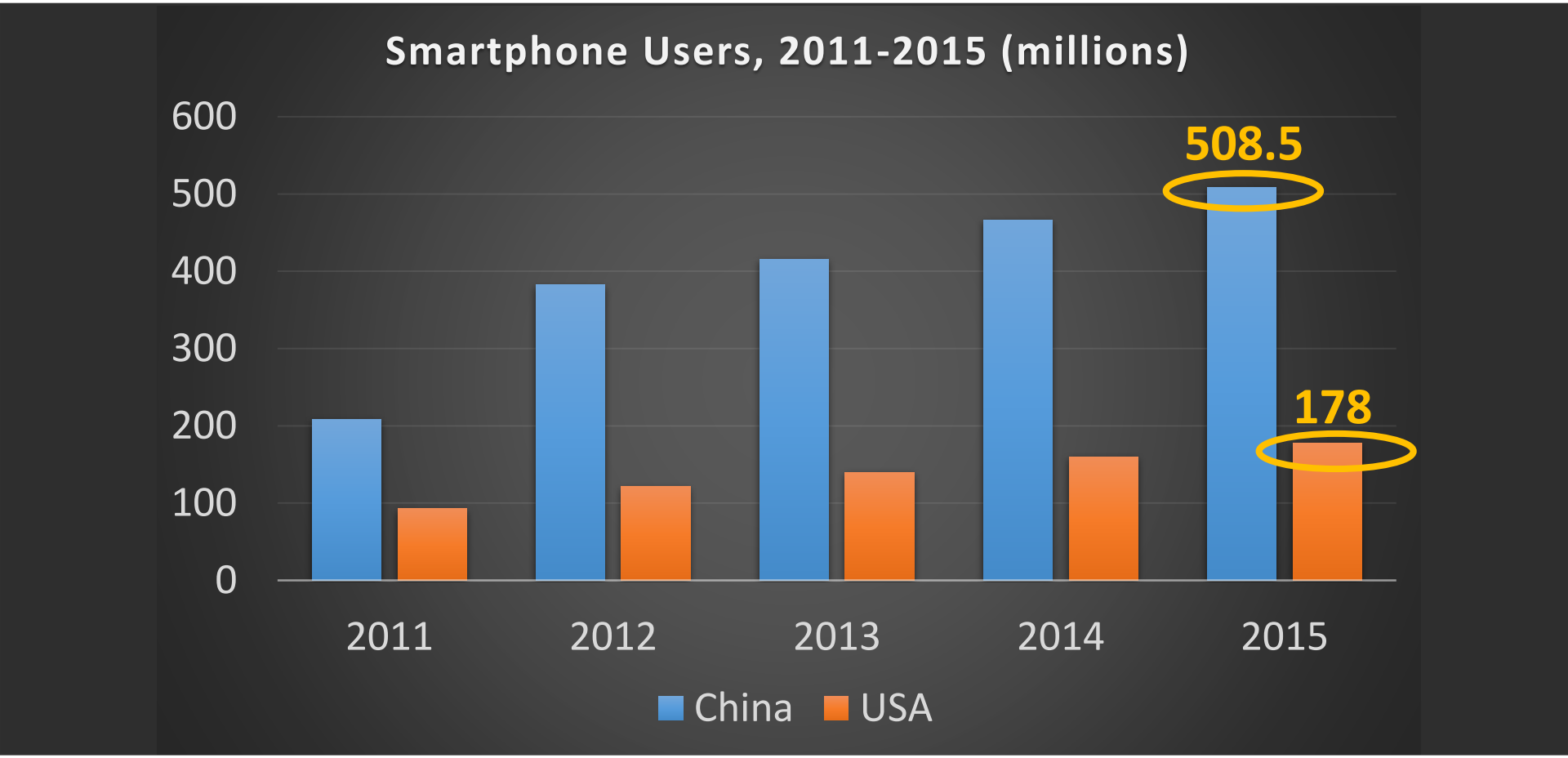
Exploring Cross-Application Cellular Traffic Optimization with Baidu TrafficGuard

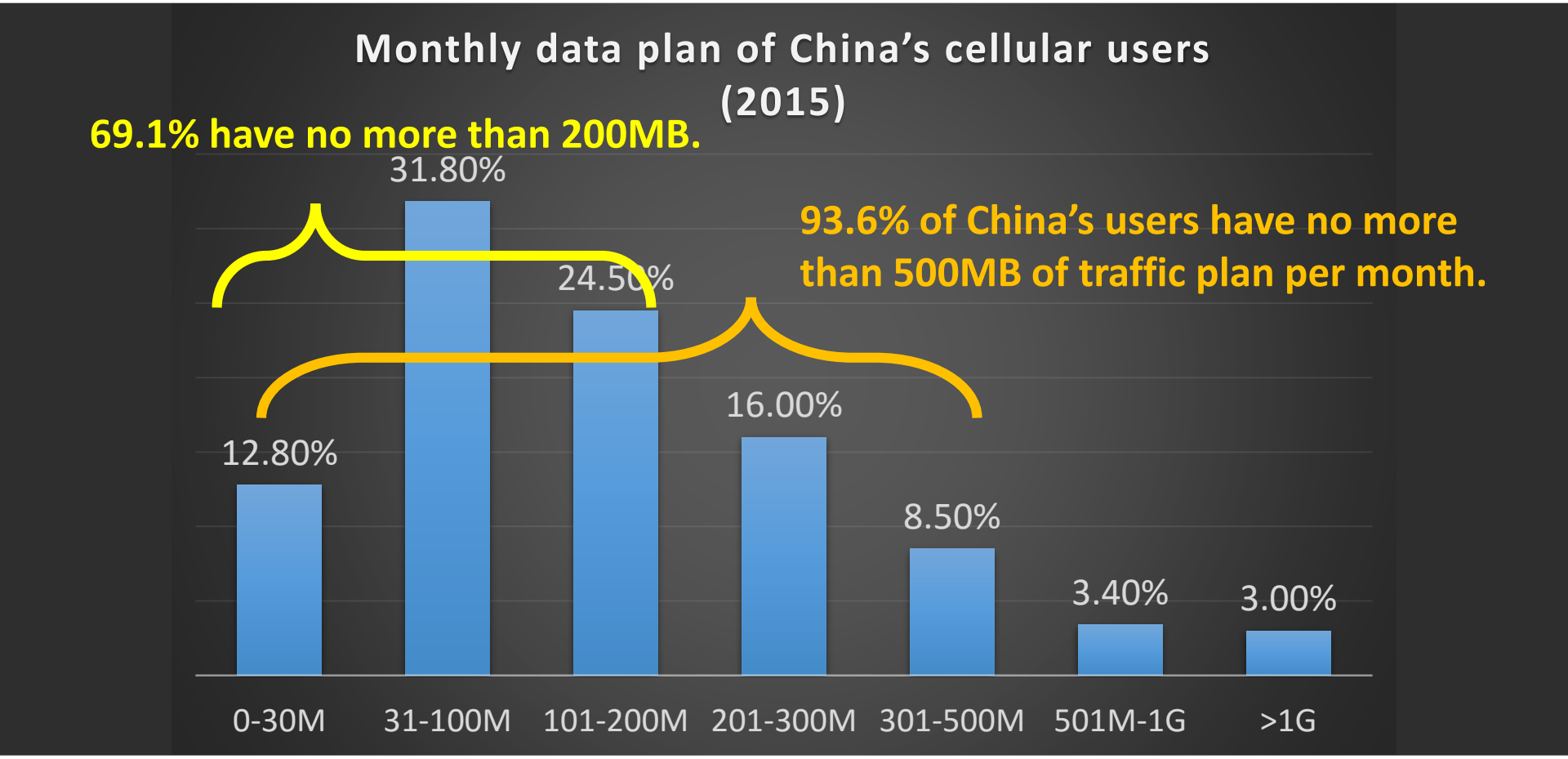
Zhenhua Li, Weiwei Wang, Tianyin Xu, Xin Zhong,
Xiang-Yang Li, Yunhao Liu, Christo Wilson, and Ben Y. Zhao

China has the world's largest population of mobile cellular users.



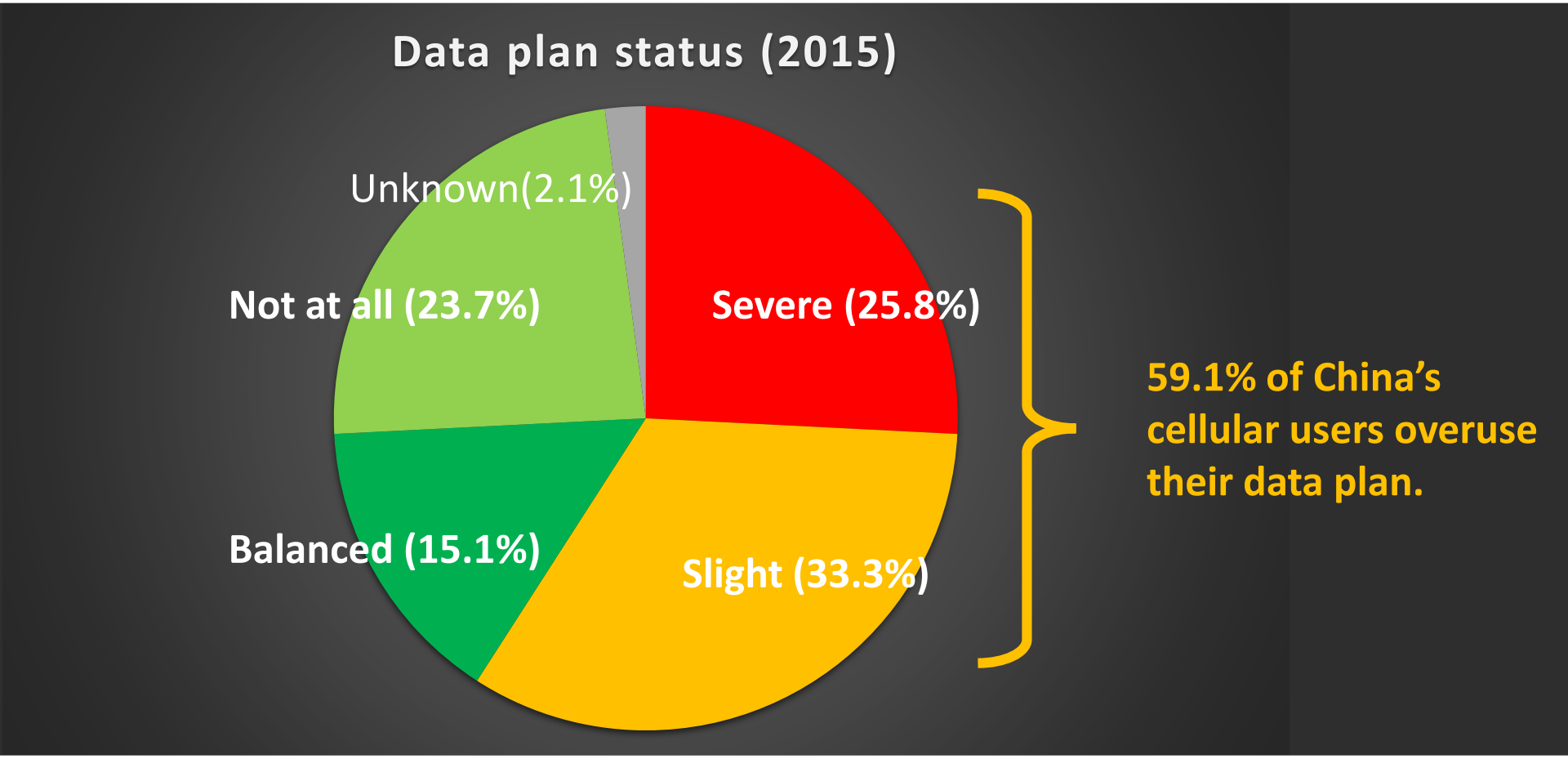
Source: www.eMarketer.com

China’s mobile users have notably small cellular data plan.



Source: iiMedia Research

Traffic overage is pervasive in China.



Source: iiMedia Research

Efforts towards optimizing mobile traffic

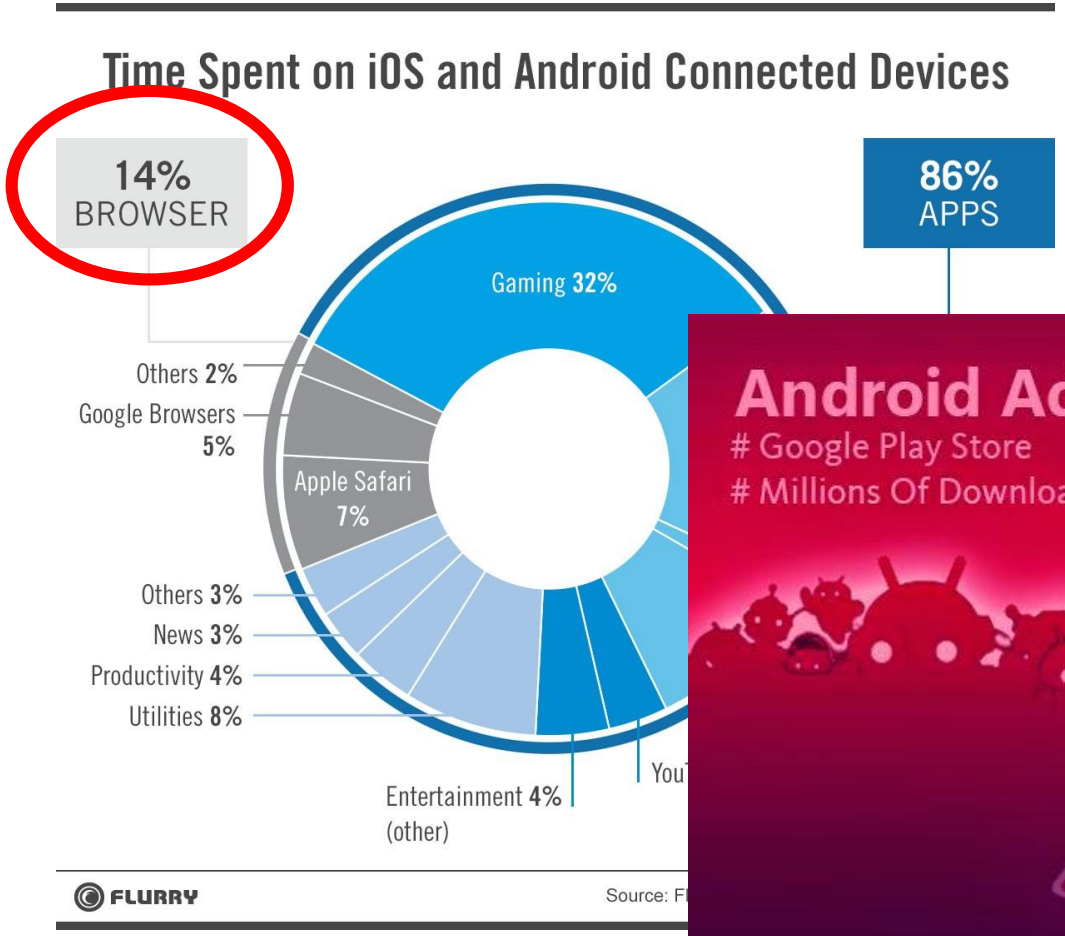
- **Application-specific** data compression

Flywheel: Google's data compression proxy (NSDI'15)

→ transcode images to the WebP format
→ minify and gzip text content

50+% of mobile traffic reduction with moderate latency penalty.

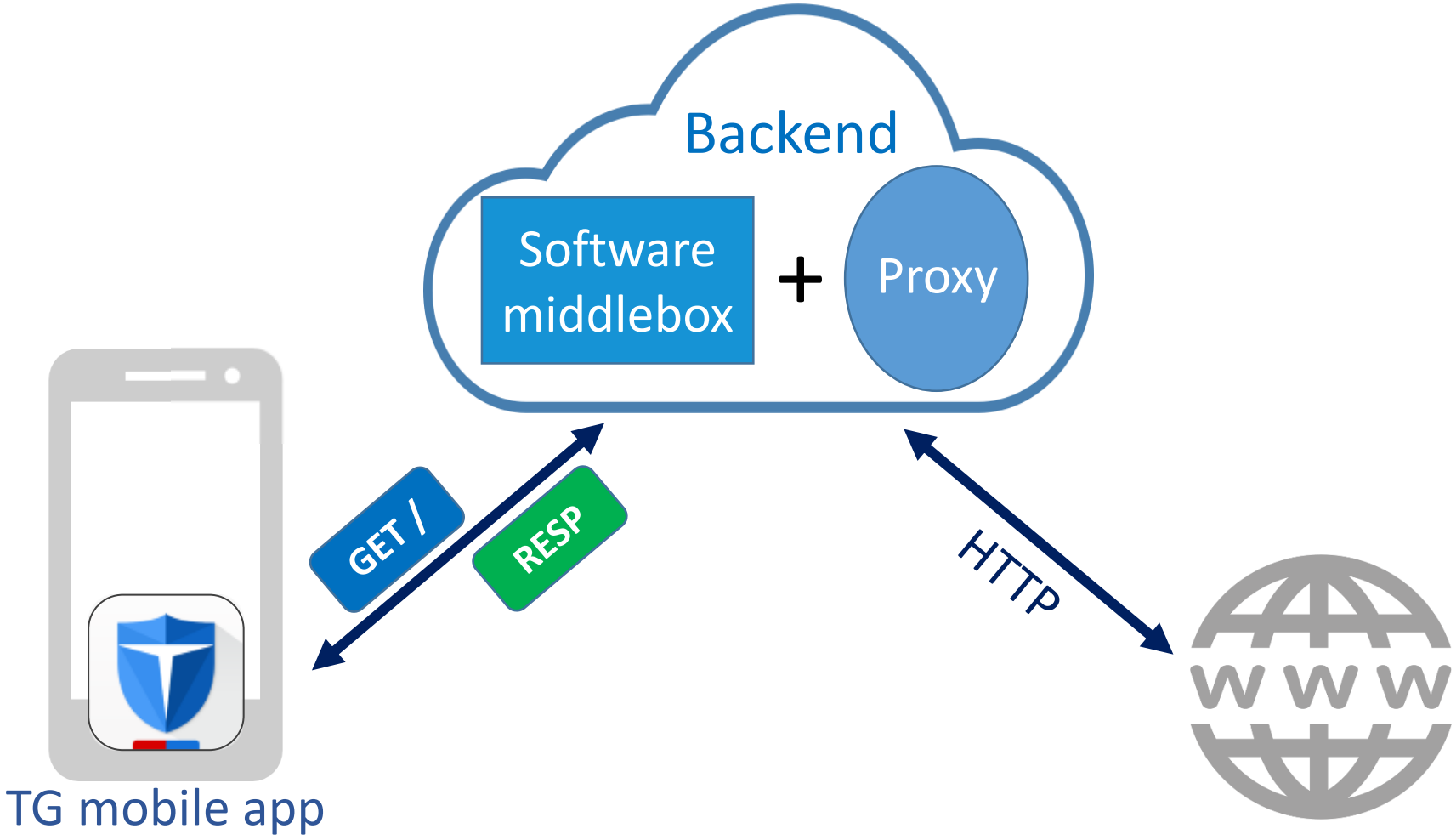
Mobile web browser traffic is only a (small) portion of today's cellular traffic.



Baidu TrafficGuard

- **Optimizing cross-app cellular traffic**
  - data compression
  - content validation
  - cross-app caching
- **Filtering out unwanted traffic**
  - malicious/ads/overnight traffic
- **Two years of deployment experience**
  - 10 millions of users & 0.2 millions of active users daily
  - support all Android 4.0+ devices

TrafficGuard Overview



TrafficGuard Overview

① **Handle all cellular traffic across all apps**

TG mobile app

Backend

Software middlebox + Proxy

GET /

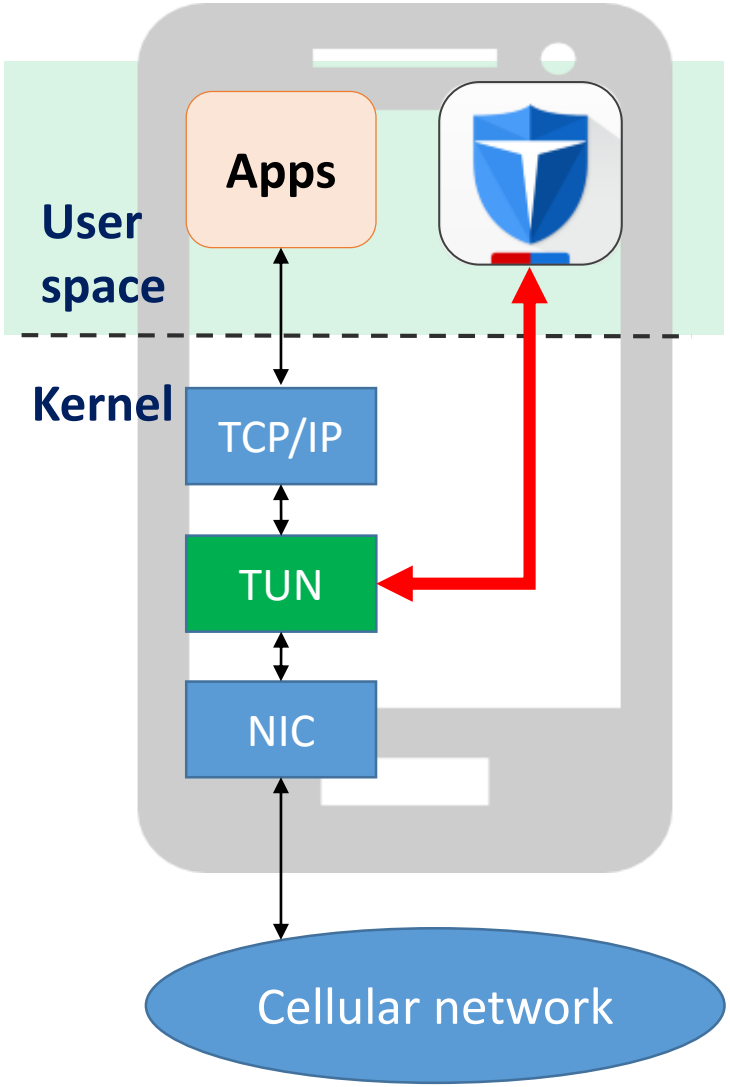
RESP

HTTP

② **Filter out unwanted traffic (e.g., malicious)**

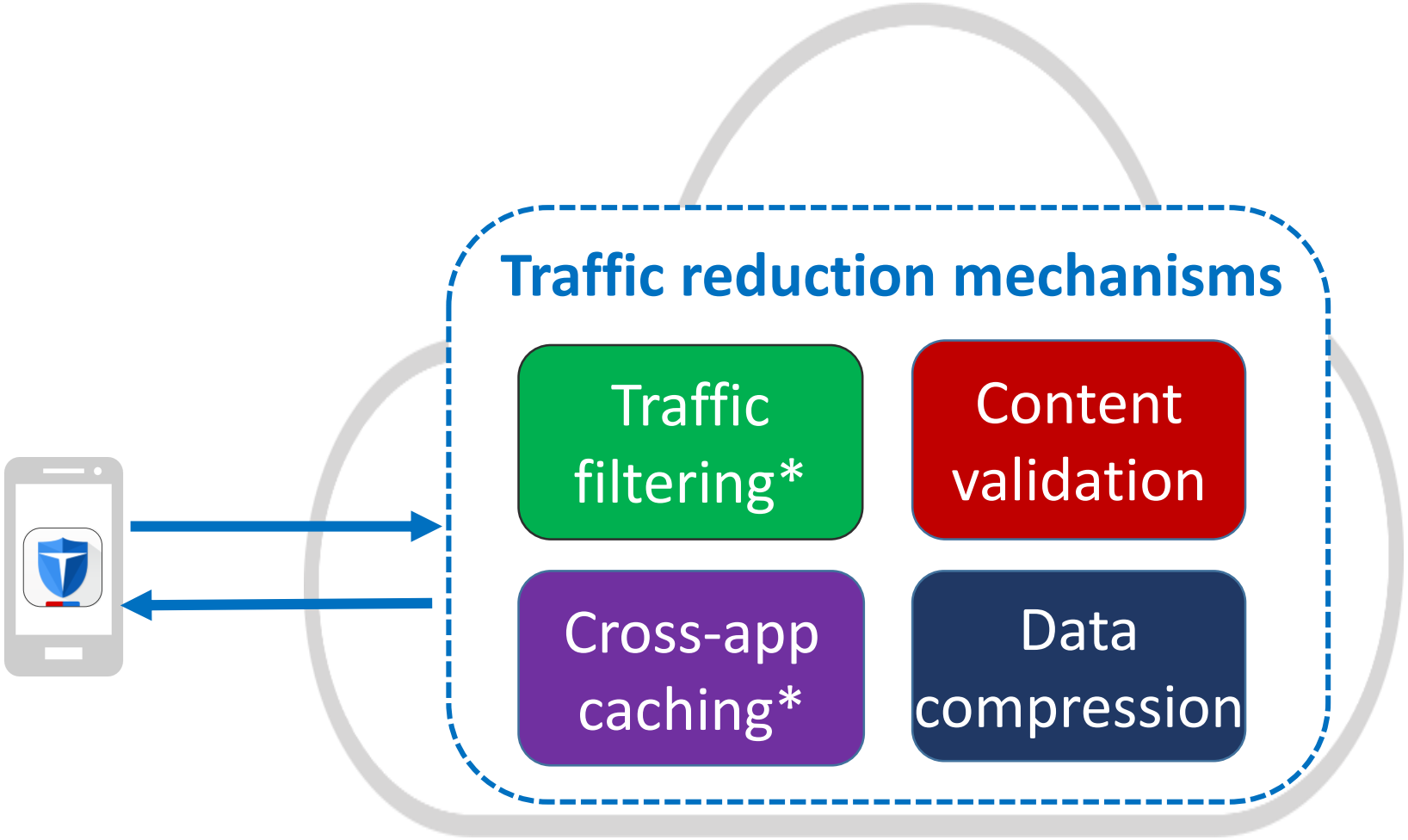
In China, HTTP requests dominate the mobile web requests (80%).

TG mobile app (client-side support)



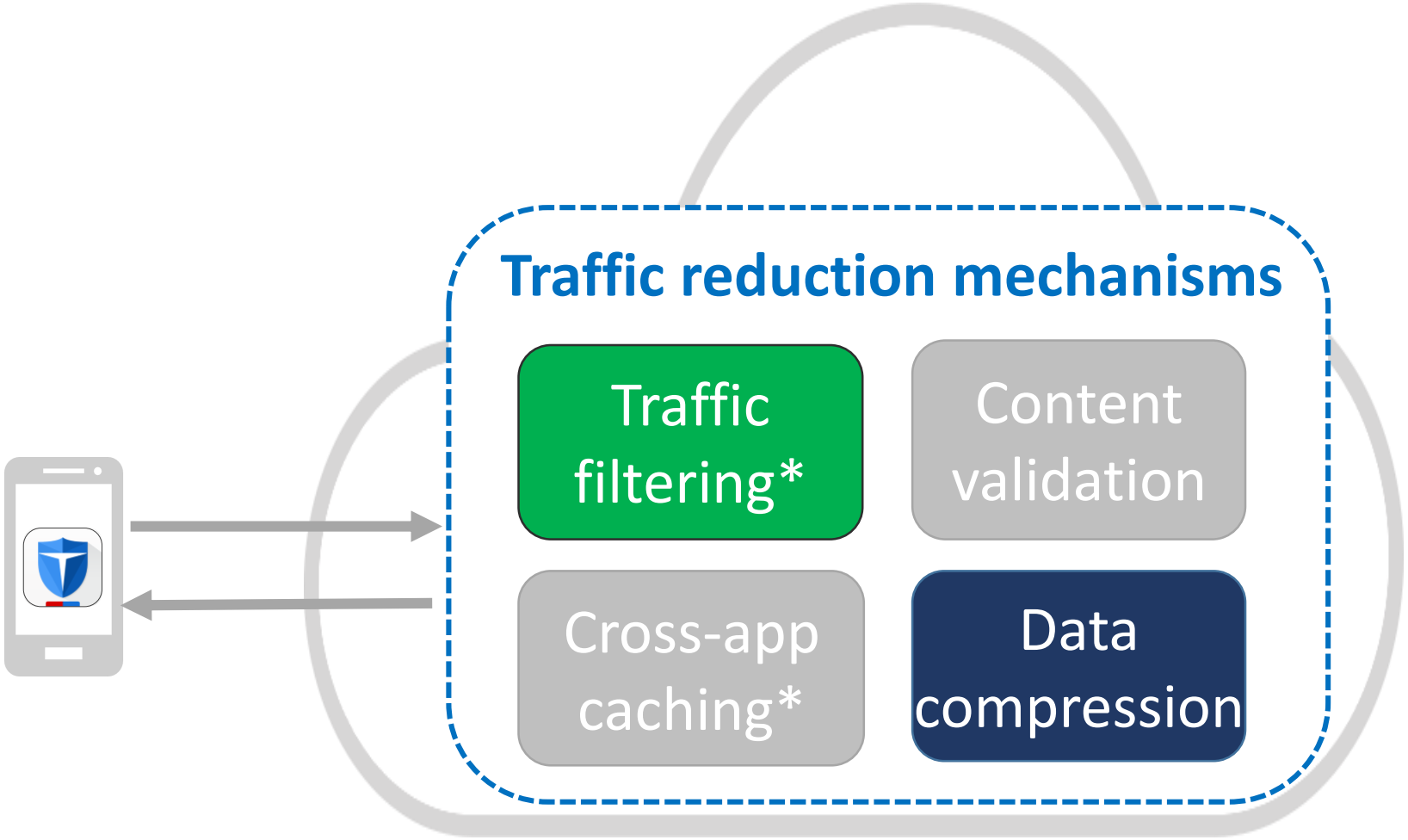
- **Customized VPN**
  - based on TUN
  - manipulate layer-3 packets at user space
  - no root privilege required
- **Cooperation w/ the backend**
  - Local filter for malicious links
  - Value-based cross-app caching

Backend: web proxy + software middleboxes



*cooperate w/ the client-side TG mobile app

Backend: web proxy + software middleboxes



*cooperate w/ the client-side TG mobile app

Data compression

- Characteristics of HTTP content types

Content Type	% HTTP Requests	% HTTP Traffic	Size (KB) Median	Size (KB) Mean
Text	**49.0%**	**15.7%**	**0.2**	**2.2**
Image	**32.0%**	**71.0%**	**5.7**	**15.5**
Octet-stream	10.0%	5.5%	0.4	3.8
Zip	8.1%	5.1%	0.5	4.3
Audio & video	0.03%	2.6%	407	614
Other	0.87%	0.1%	0.3	0.7

*Text: HTML, CSS, JSON, XML, JS, etc

Text compression

- Text compression is not worthwhile.
  - → very small in size (mean: 0.2KB)
  - → reduce HTTP traffic by 1.4%
  - → considerable computation overhead at both client and cloud side.

TrafficGuard does not compress text objects.

Image compression

- 71% of HTTP traffic are images, and 40% of these images are oversized.

- Challenges:
  → **cannot** transcode all the images to WebP (Flywheel's approach) because not all existing apps support WebP
  → **cannot** use WebP as the transfer format as the client-side transcoding overhead is not acceptable
  → **cannot** simply rescale images which may distort UI layout and degrade UX

Image compression

- Adjusting the **quality** of the images by tuning QF (Quality Factors) of JPEG images

JPEG: 500×375

75	60	45
QF	30	15

Image compression

- Adjusting the **quality** of the images by tuning QF (Quality Factors) of JPEG images

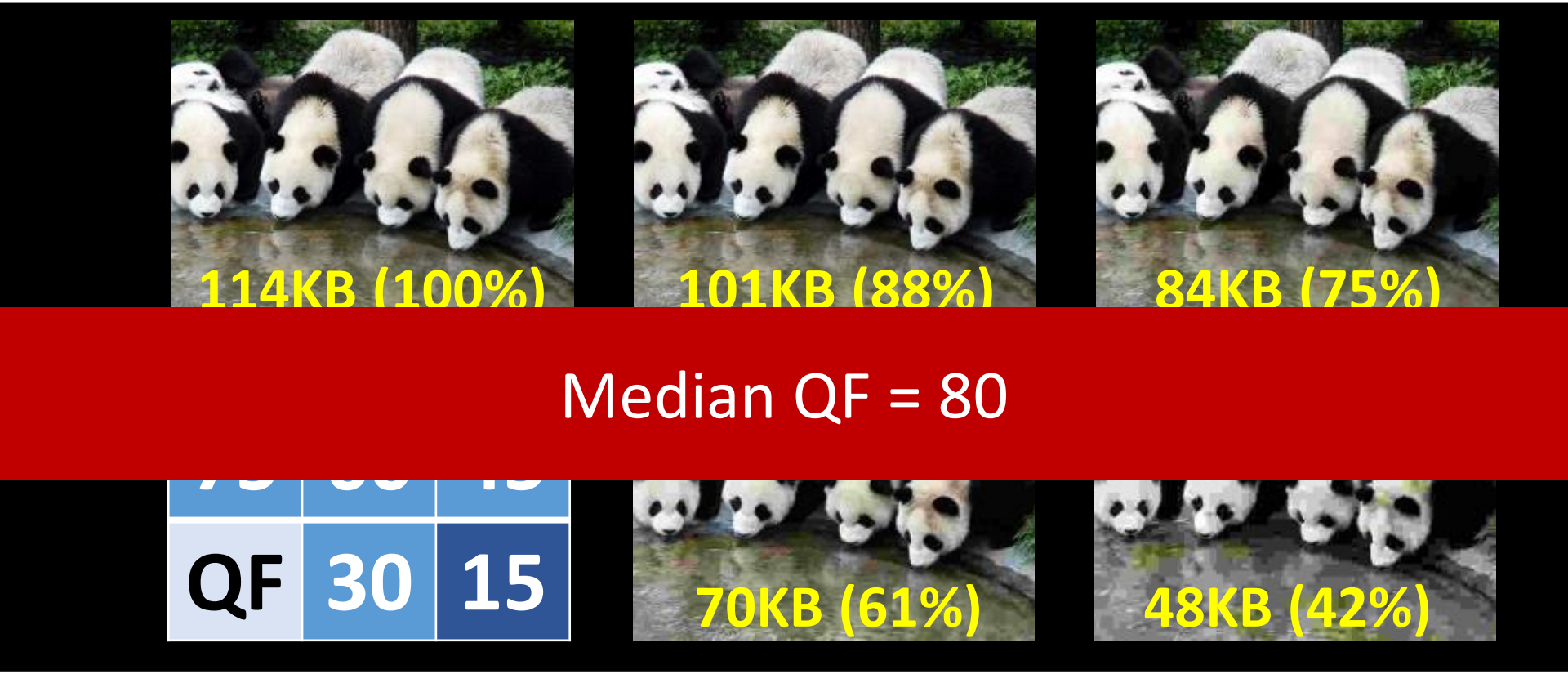


Image compression

- Adjusting the **quality** of the images by tuning QF (Quality Factors) of JPEG images

→ categorize images into *Large*, *Medium*, *Small* & *Tiny*
- based on the Ziproxy standard

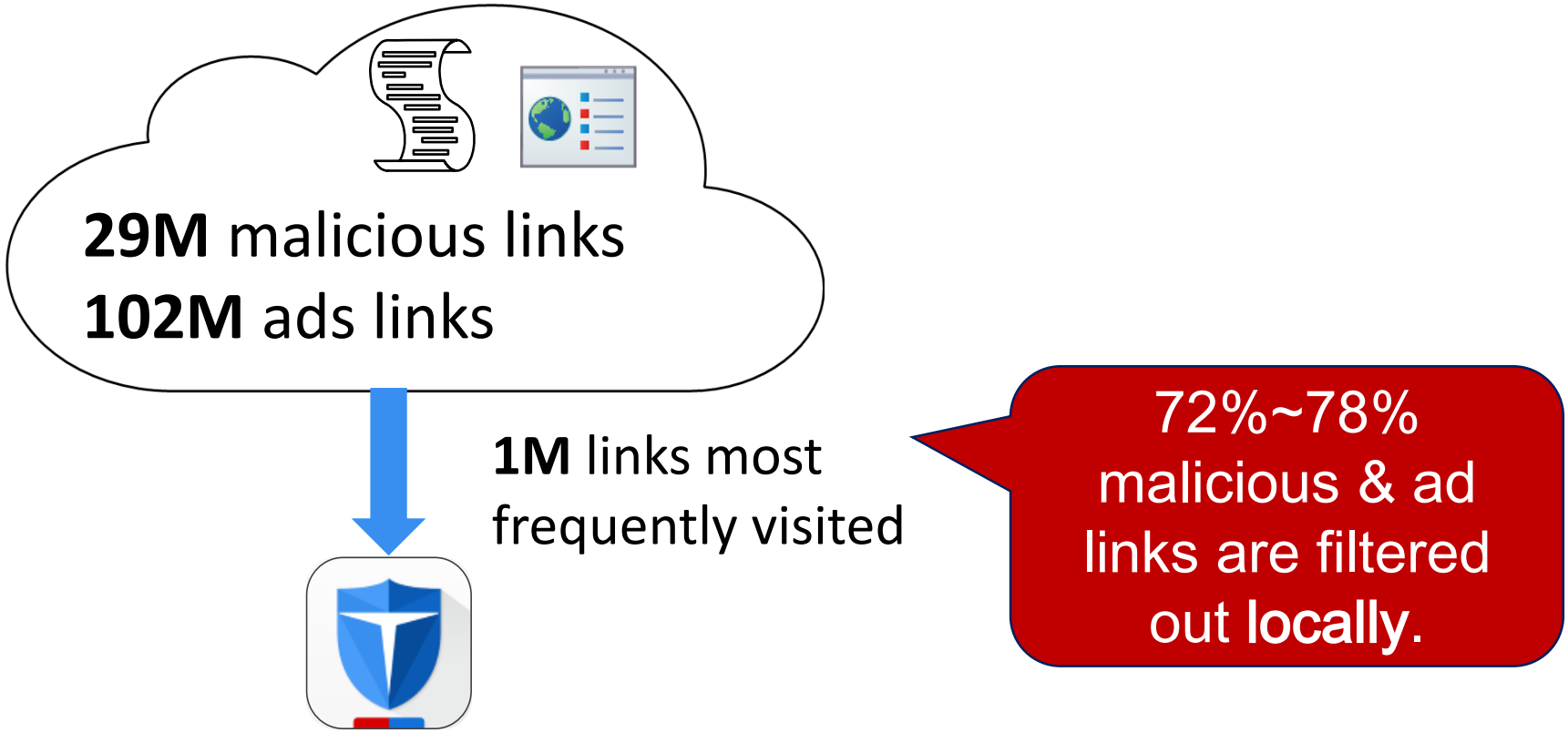
→ tune QF to balance compression ratio and quality
- quality is measured by SSIM (Structural Similarity)

→ transcode PNG & GIF to JPEG if possible
- **59.9%** of the images are in JPEG
- **8-10** times smaller overhead than WebP transcoding

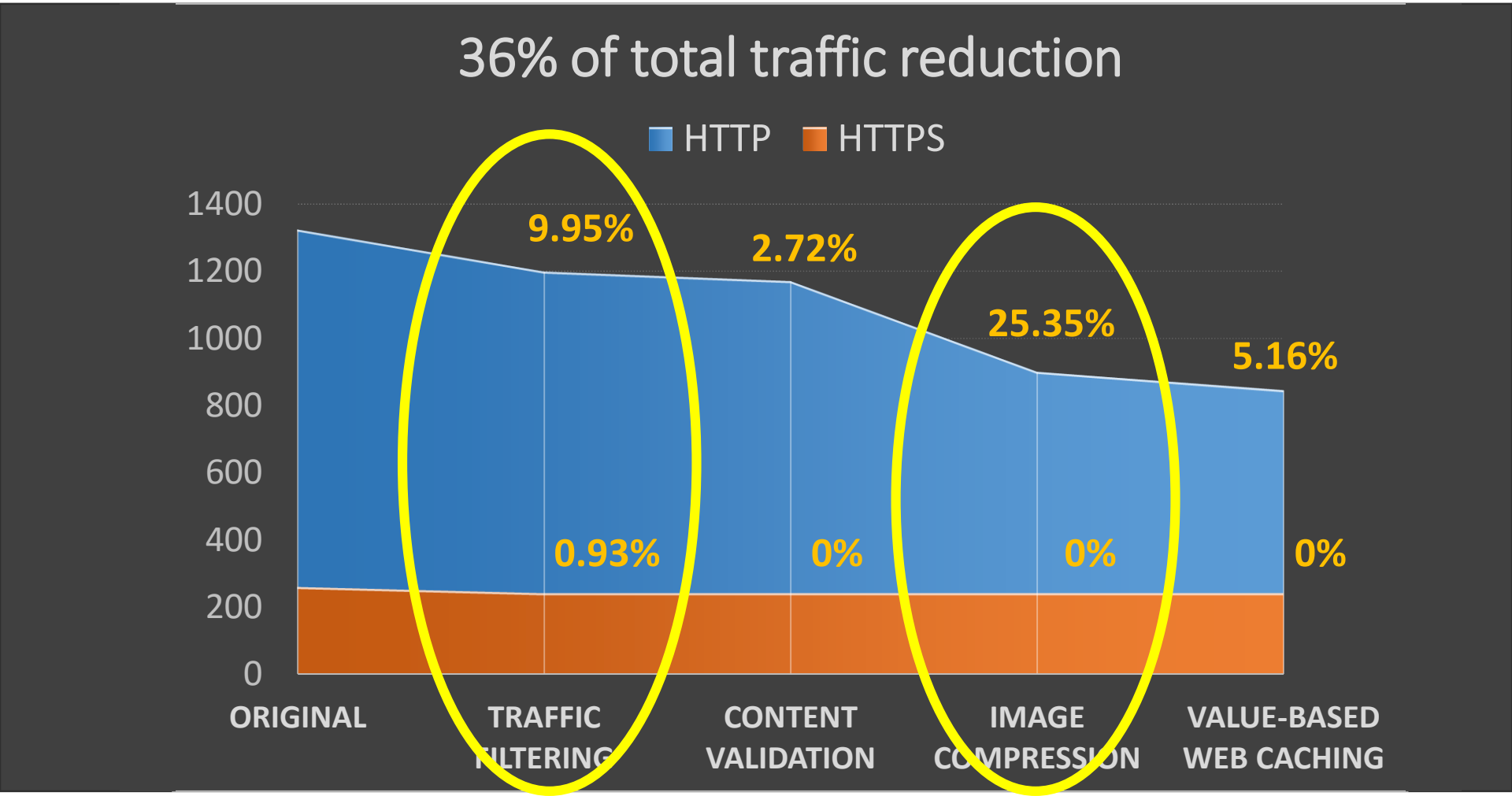
25.4% of traffic is reduced by such image compression!

Traffic filtering

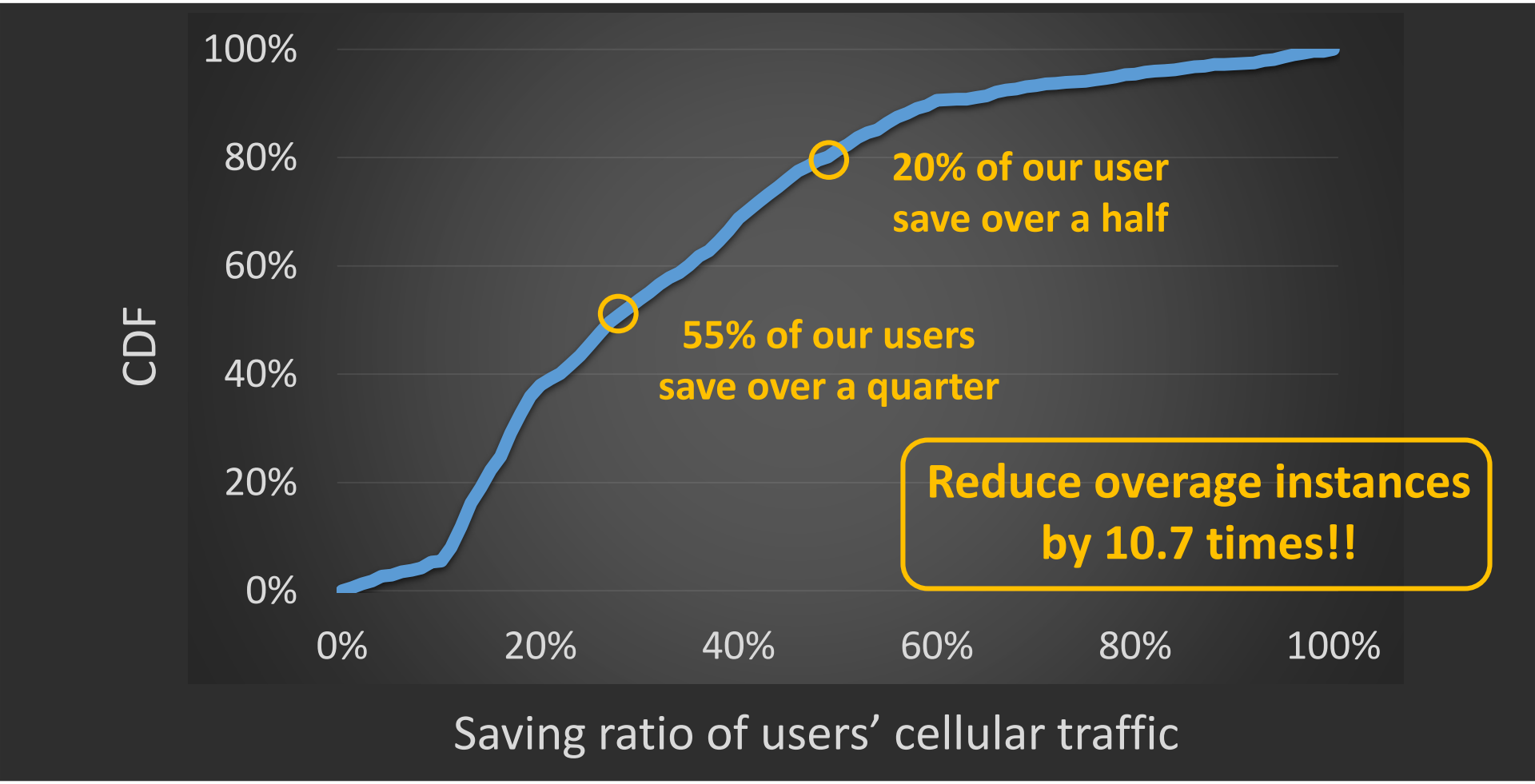
- Two-lever filtering cooperated by both the client-side mobile app and the backend



Total cellular traffic reduction



Per-user traffic saving



Latency penalty & battery overhead

- End-to-end latency penalty
  - worst case: 474ms
  - mean: **282ms** & median: **53ms**

- Battery consumption
  - negligible (93mW on average)
  - sometimes reduce battery consumption

Conclusion

- TrafficGuard achieves **36%** of cellular traffic reduction across different apps without degrading user experiences.

- Baidu is working on integrating TrafficGuard with cellular carrier infrastructure, which can further reduce end-to-end latency.