Gaining Control of Cellular Traffic Accounting by Spurious TCP Retransmission

Younghwan Go, Jongil Won, Denis Foo Kune*,
EunYoung Jeong, Yongdae Kim, KyoungSoo Park

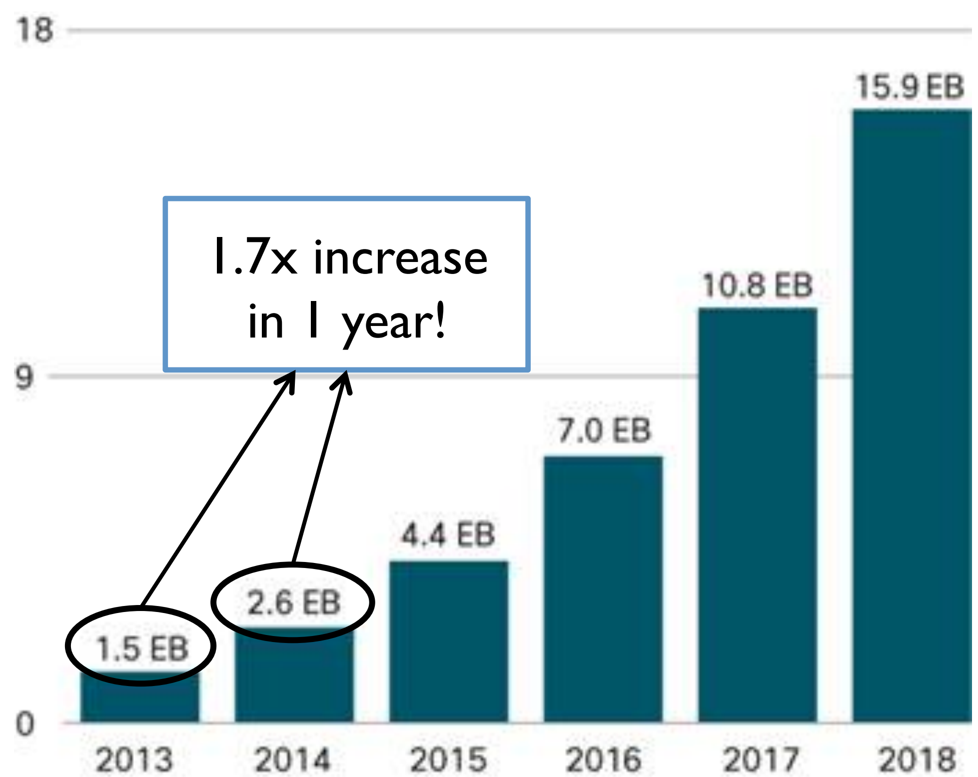
KAIST University of Michigan*

Mobile Devices as Post-PCs

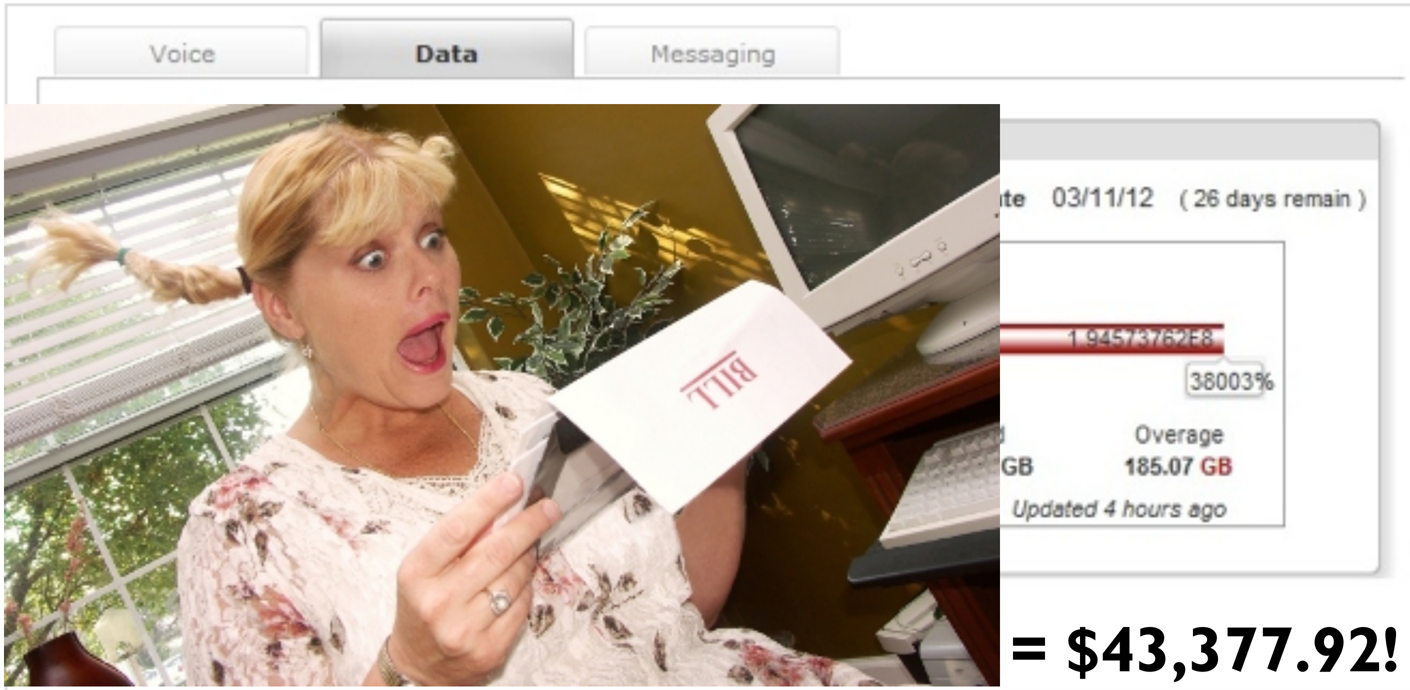
- Smartphones & tablet PCs for daily network communications

Mobile Devices as Post-PCs

- Smartphones & tablet PCs for daily network communications
  - Massive growth in cellular data traffic (Cisco VNI Mobile, 2014)



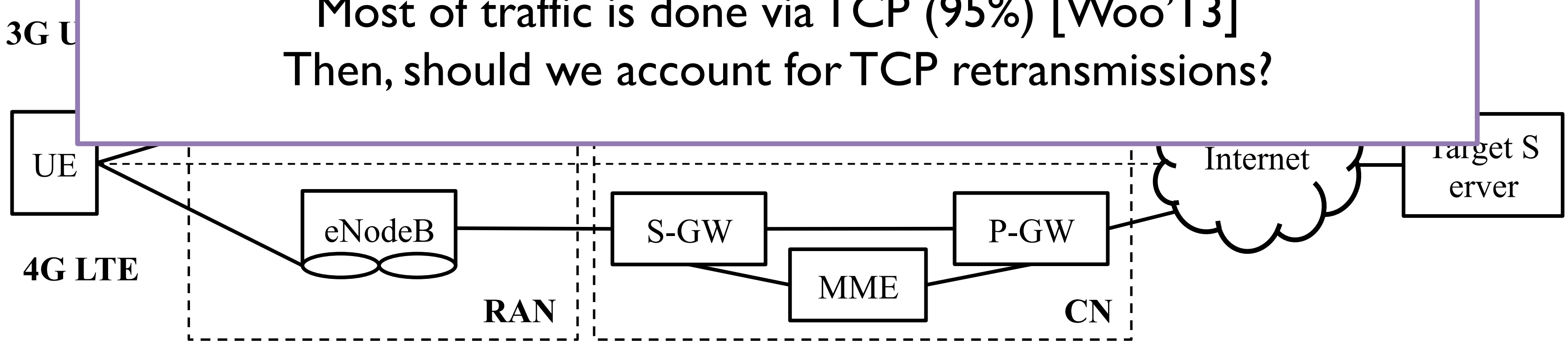
Cellular Traffic Accounting



Cellular network subscribers want accurate accounting!

3G/4G Accounting System Architecture

- Charging Data Record (CDR)
  - Billing information (e.g., user identity, session elements, etc.)
- Record traffic volume in IP packet-level



Question:

Most of traffic is done via TCP (95%) [Woo'13]
Then, should we account for TCP retransmissions?

Cellular Provider’s Dilemma: Charging TCP Retransmissions

- Subscriber’s stream of consciousness

What’s TCP ret
ransmission?

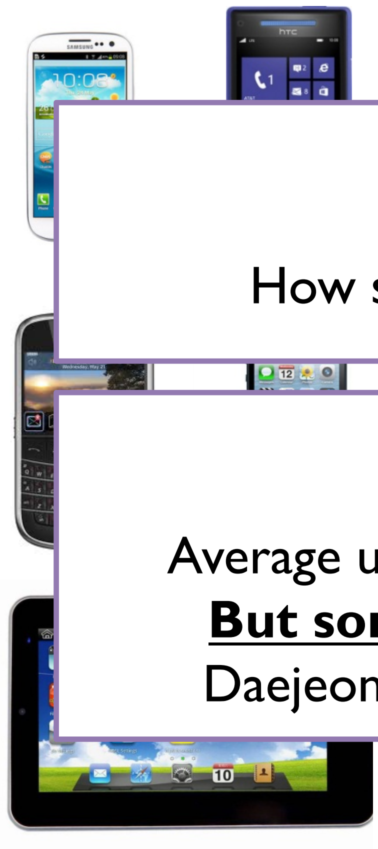
Network conditi
on is not my pro
blem

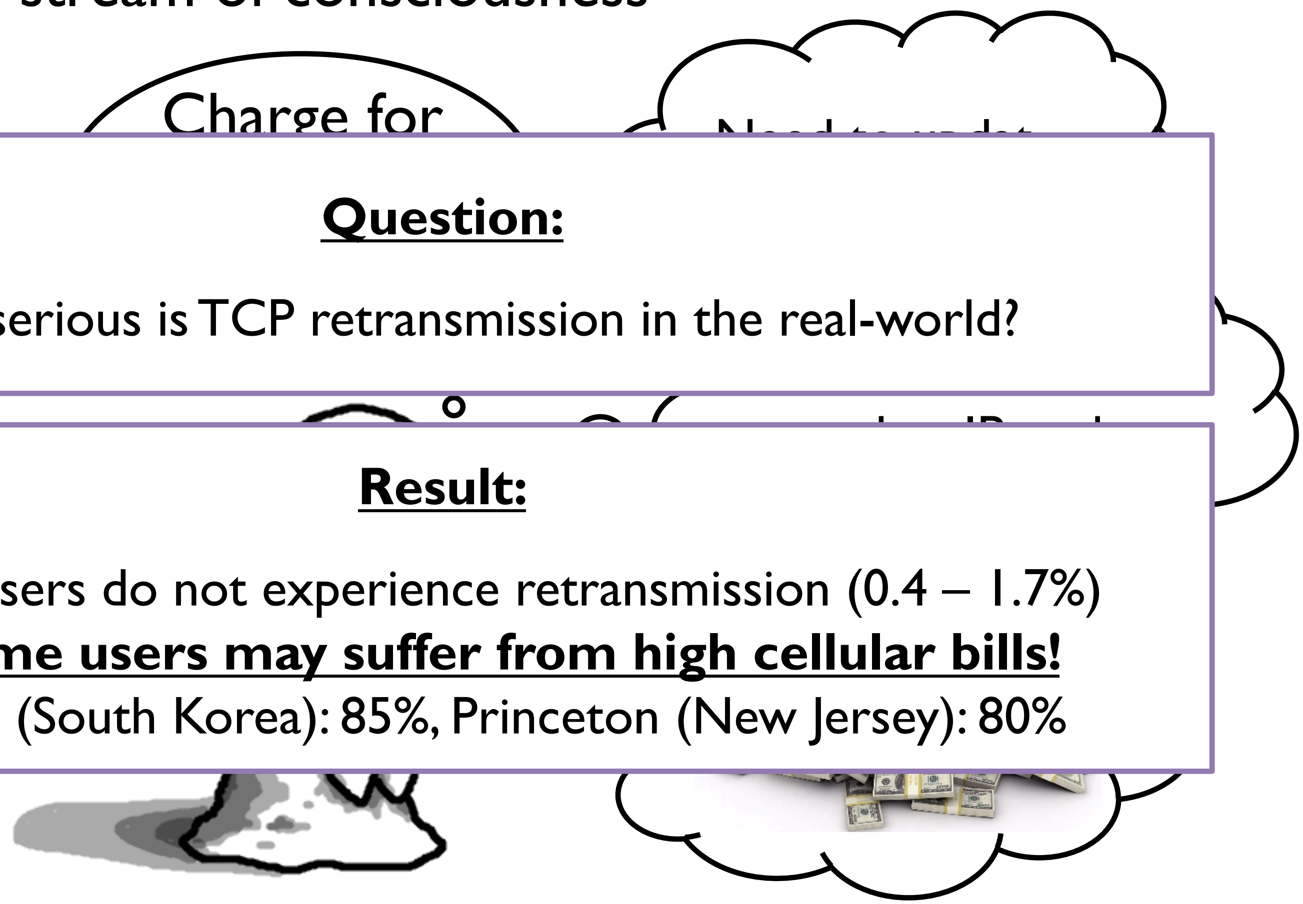
Charge volume
= file size

Pay for ap
plication d
ata only!

Cellular Provider's Dilemma: Charging TCP Retransmissions

- Cellular ISP's stream of consciousness





Question:

How serious is TCP retransmission in the real-world?

Result:

Average users do not experience retransmission (0.4 – 1.7%)
But some users may suffer from high cellular bills!
Daejeon (South Korea): 85%, Princeton (New Jersey): 80%

Contributions

- Identify current TCP retransmission accounting policies of 12 cellular ISPs in the world
  - Some ISPs account for retransmissions (blind), some do not (selective)
- Implement and show TCP retransmission attacks in practice
  - Blind → "Usage-inflation" attack
    - Overcharge a user by 1 GB in just 9 minutes without user's detection!
  - Selective → "Free-riding" attack
    - Use the cellular network for free without ISP's detection!
- Design an accounting system that prevents "free-riding" attack
  - Accurately identify all attack packets
  - Works for 10 Gbps links even with a commodity desktop machine

TCP Retransmission Accounting Policy

- Tested 12 ISPs in 6 countries

ISPs (Country)	Policy
AT&T, Verizon, Sprint, T-Mobile (U.S.)	Blind
Telefonica (Spain)	Blind
[illegible]	Blind
T-Mobile (England)	Blind
China Unicom, CMCC (China)	Blind
SKT, [illegible]	Selective

Vulnerable to "usage-inflation" attack!

Vulnerable to "free-riding" attack!

Related Works

- Peng et. al. [MobiCom'12, CCS'12]
  - Toll-free data access attack
    - Bypass cellular accounting via DNS port, which used to be free-of-service
    - U.S. ISPs now account for all packets going through DNS port
    - South Korean ISPs verify DNS packets
  - Stealth-spam attack
    - Inject large volume of spam data via UDP after the connection is closed
    - Attack limited as most of traffic is TCP (95%)
- Tu et. al. [MobiSys'13]
  - Inject large volume of spam data via UDP while the user is roaming
    - Packet drops during handoffs (e.g., 2G↔3G, 3G↔LTE)
  - Attack not so severe in real life since TCP is most dominant

Usage-inflation Attack

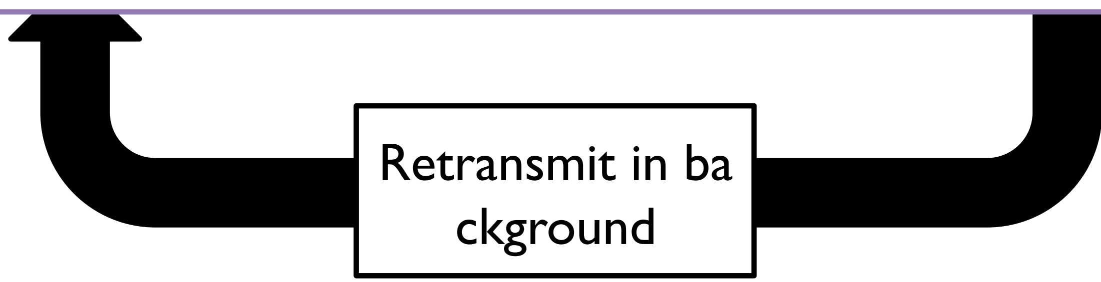
- Intentionally retransmit packets even without packet losses
  - ISPs with blind accounting policy charge for all packets

<u>Strength:</u>

No need to compromise the client

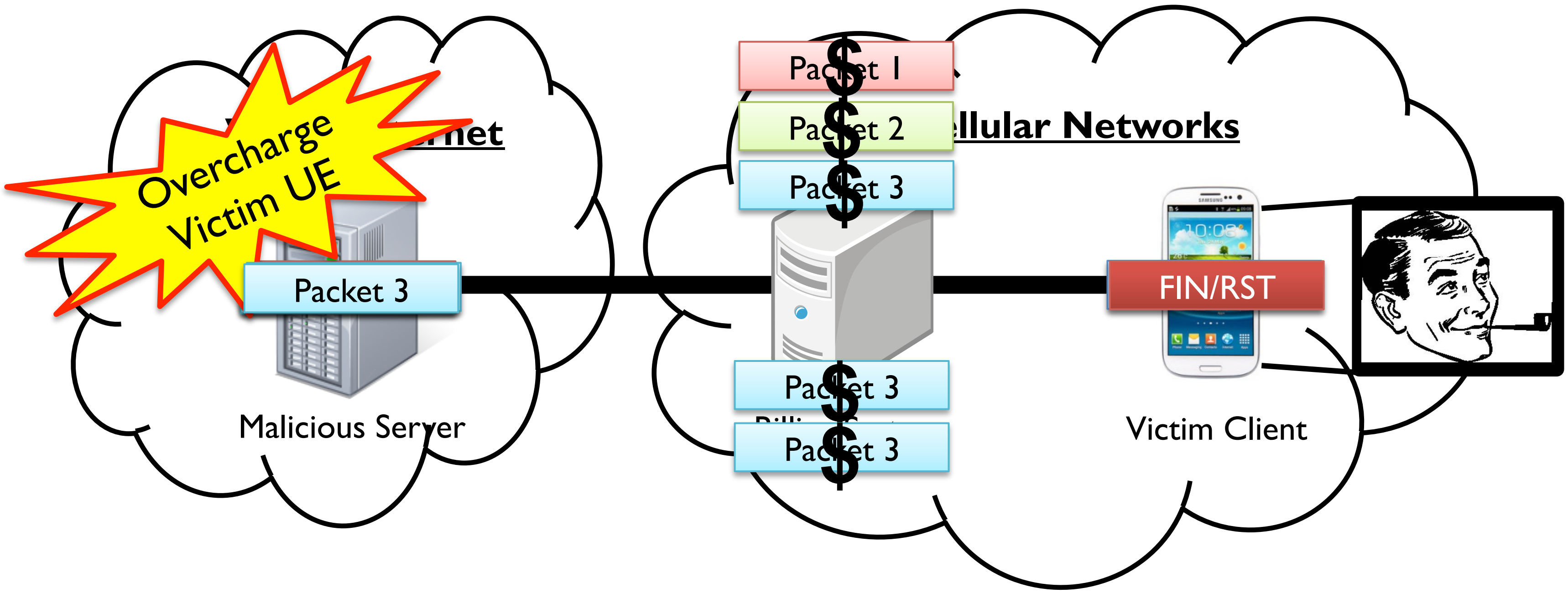
User does not notice an attack

Inflate more than 1GB in just 9 minutes!



Retransmit after FIN

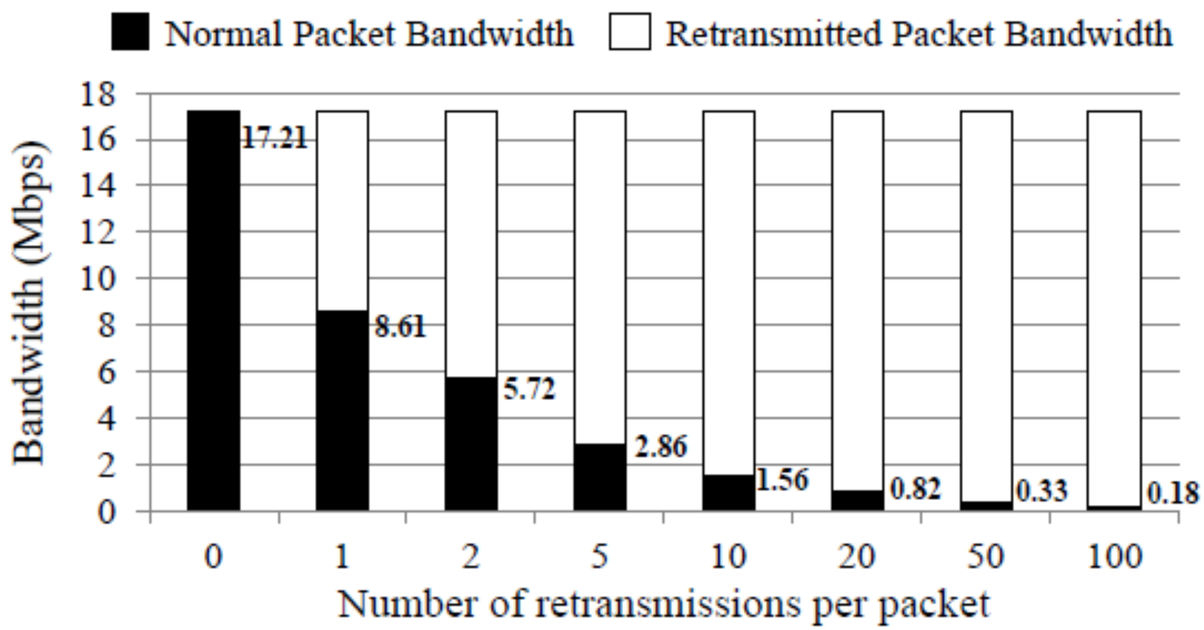
- Ignore client's FIN/RST to prevent TCP teardown
  - Utilize full bandwidth to overcharge the usage



- Some ISPs allow attacks even after 4 hours!

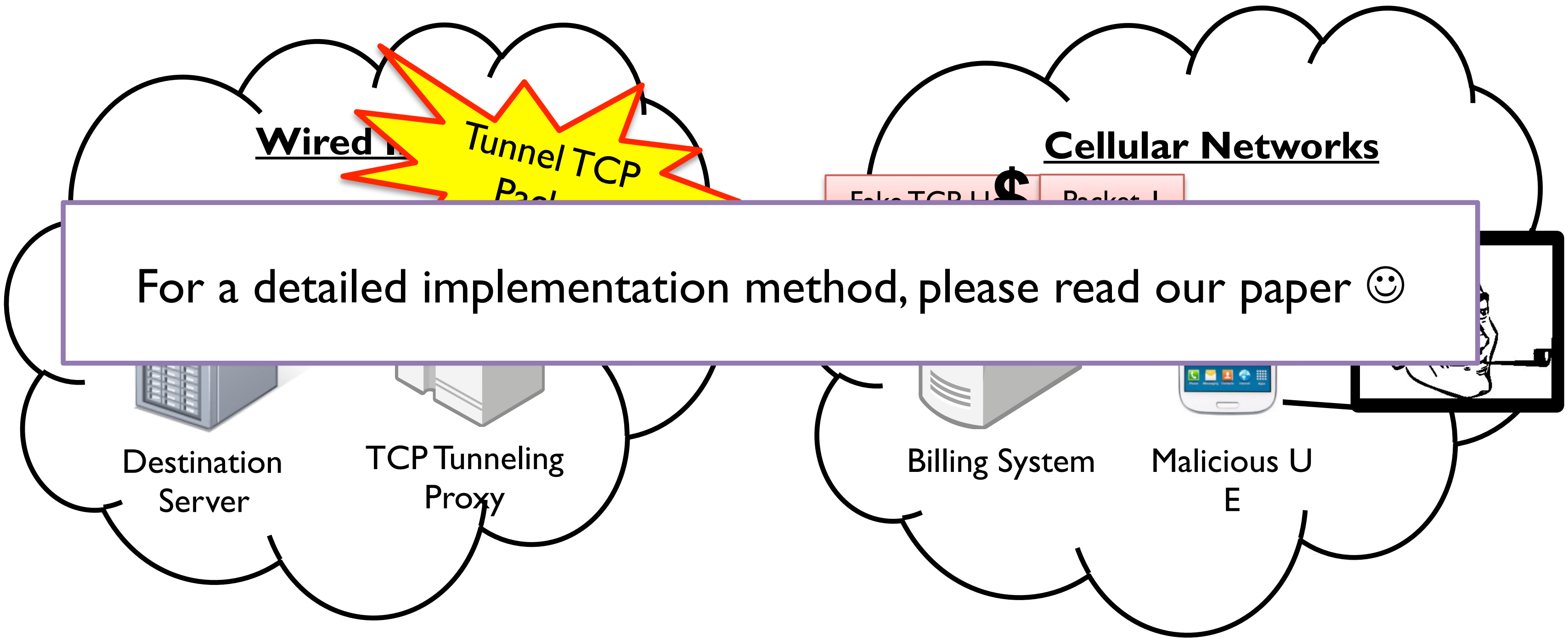
Retransmit during Normal Transfer

- ISP may block data packet retransmissions after FIN/RST
- Embed retransmission packets in stream of normal packets
  - Guarantee minimum goodput for interactive content



Free-riding Attack

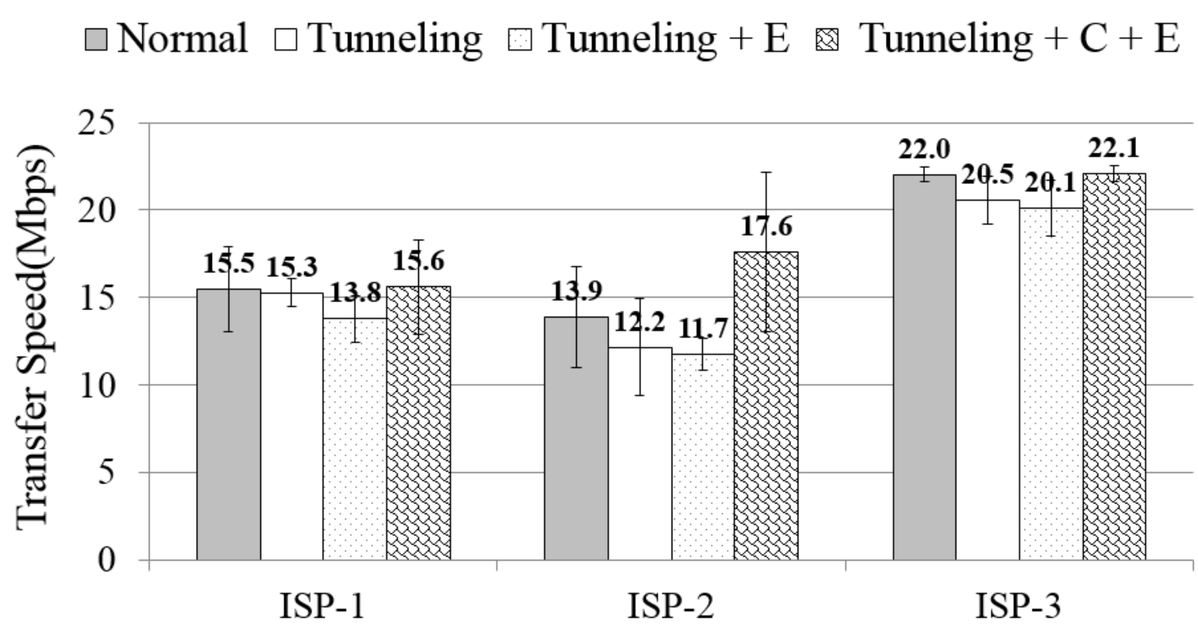
- Tunnel payload in a packet masquerading as a retransmission
  - ISPs with selective accounting policy inspects TCP header only



For a detailed implementation method, please read our paper ☺

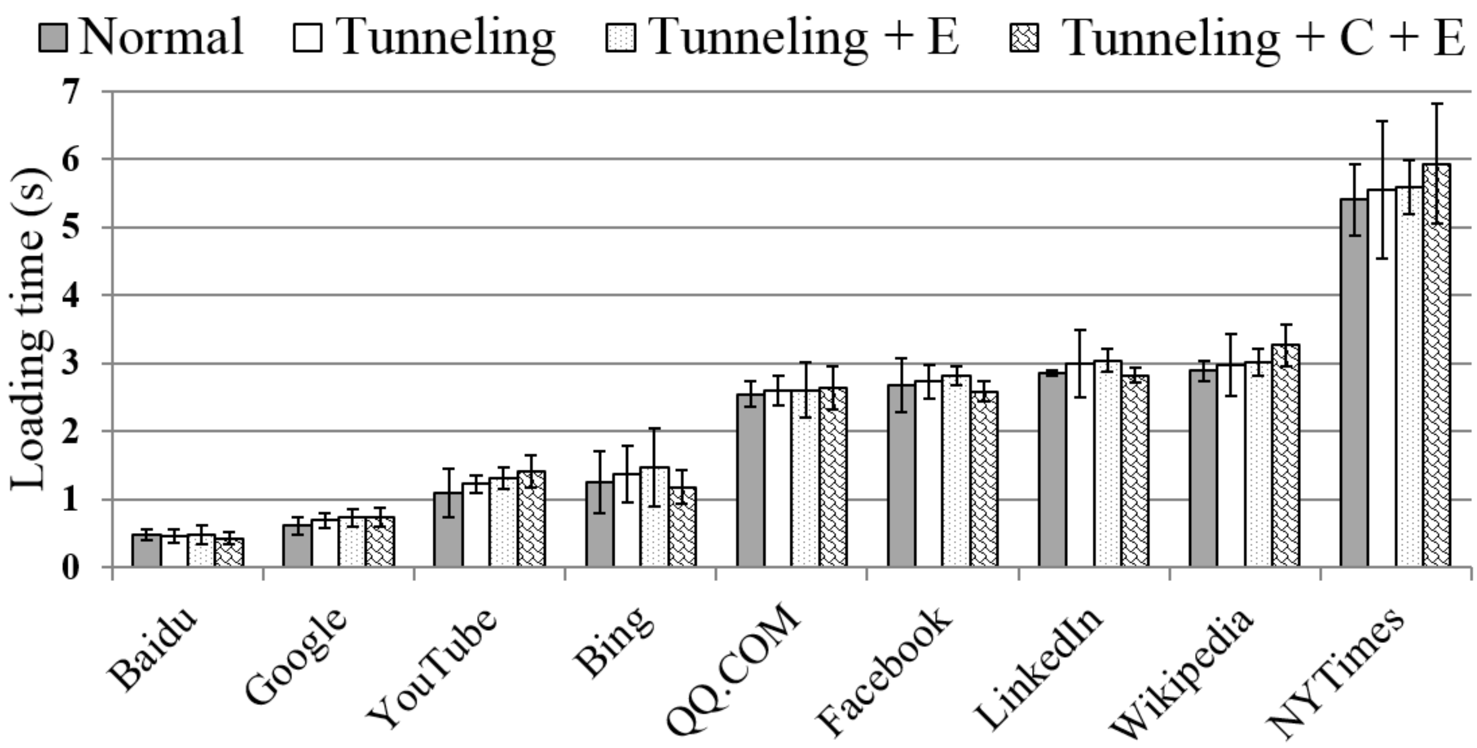
Free-riding Attack in Practice

- Attack successful in all 3 South Korean ISPs
  - http://abacus.kaist.edu/free_riding.html
- Packet encryption → evade tunnel header detection
- Packet compression → increase data transfer speed



Free-riding Attack in Practice

- Practical even for normal web usage

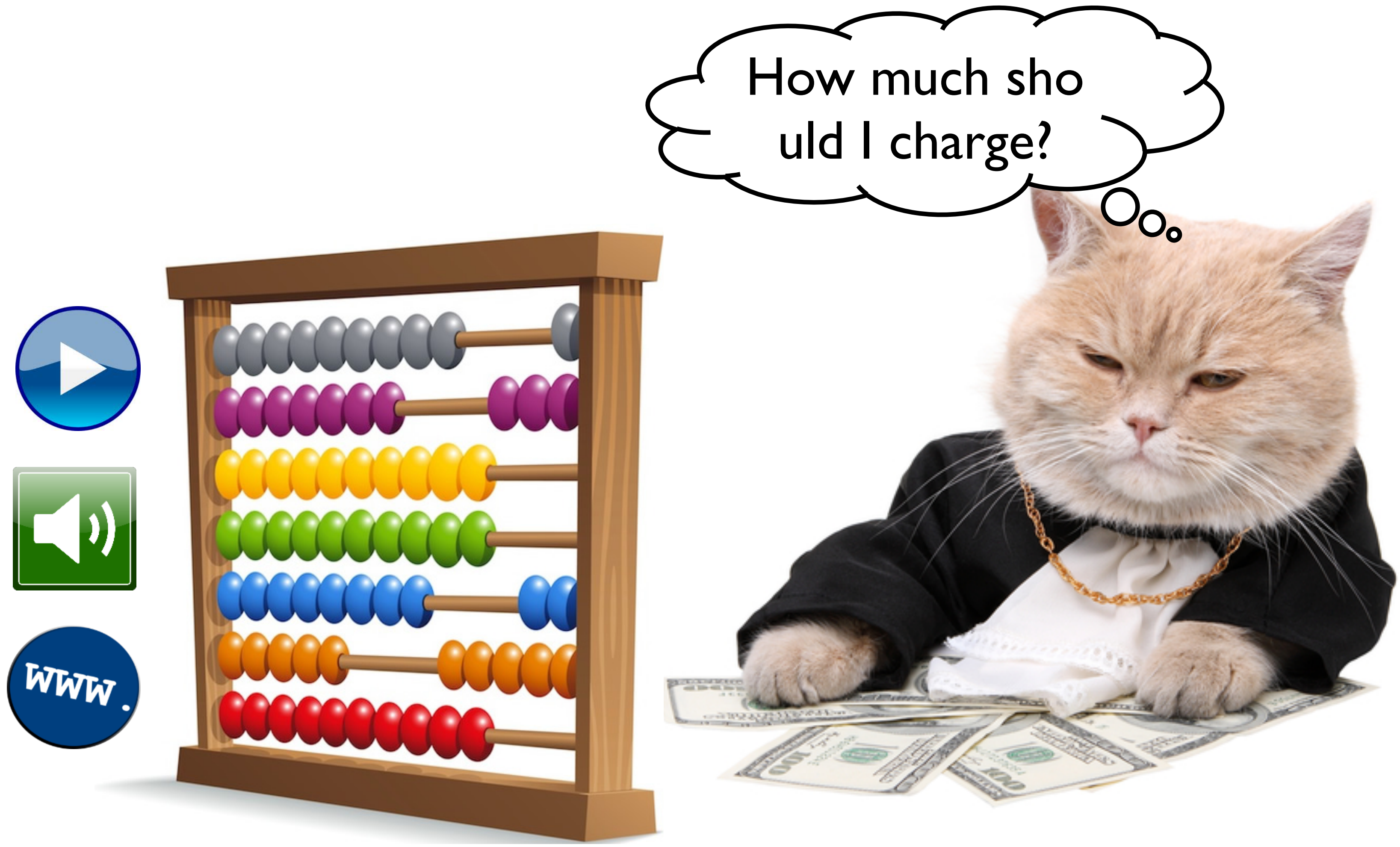


Defending against Retransmission Attacks

- Difficult to fundamentally defend against "usage-inflation" attack
  - Detect attack by a retransmission rate threshold
    - 85% retransmission ratio for legitimate flows → lead to false positives
  - Monitor TCP sender behavior

ISPs should not charge for retransmissions
but defend against "free-riding" attack!

- Reasonable to defend against "free-riding" attack
  - Attacker can simulate behavior of poorly-provisioned environment
  - Accurately identify retransmission tunneled packets via DPI



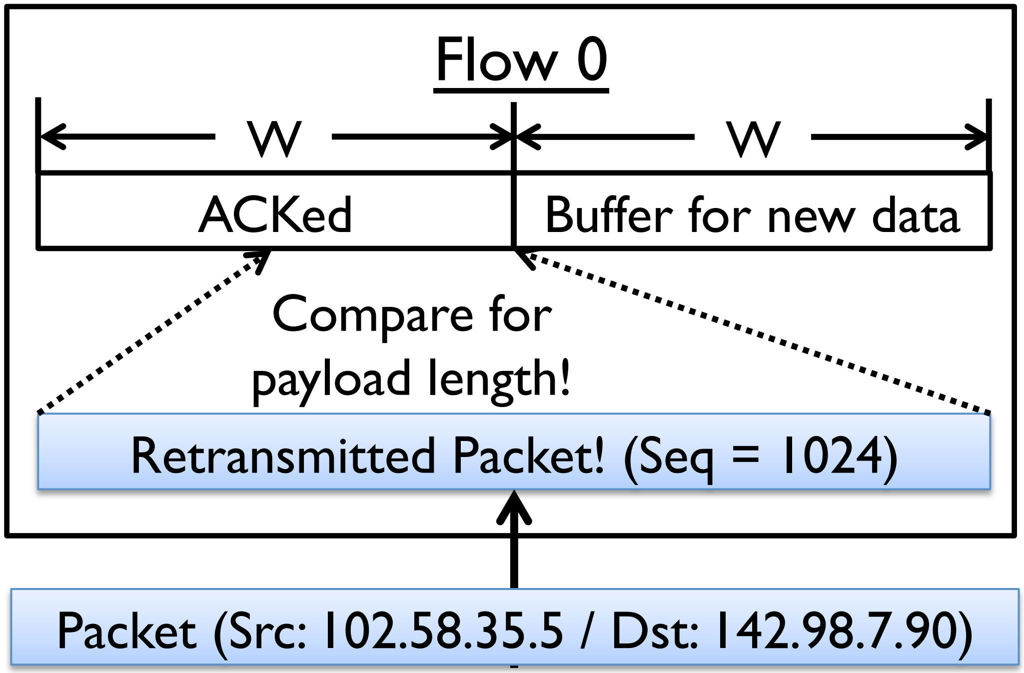
Abacus: Cellular Data Accounting System

Abacus: Deterministic DPI

- Byte-by-byte comparison of original vs. retransmitted packets
- Buffer size: 2 x Receive Window Size
- Accounting process
  - Head seq: 0
  - Window: 2KB
  - Next expected seq: 2048

Strength:
No false-positives!

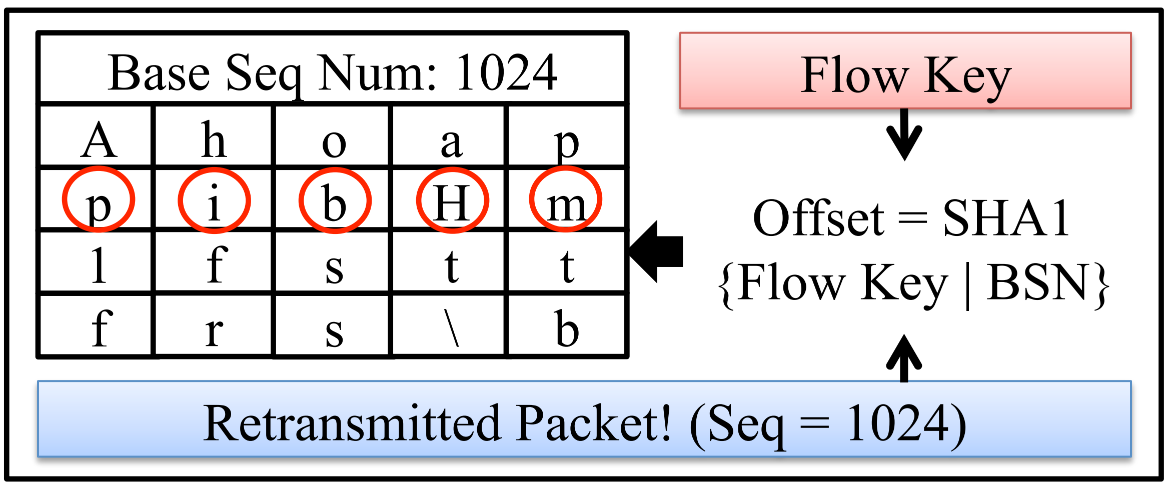
Weakness:
Require large memory!



Abacus: Probabilistic DPI

- Store payload by sampling and compare for the sampled data
  - E.g., store 5 bytes out of 1,024-byte → reduce memory by ~200x

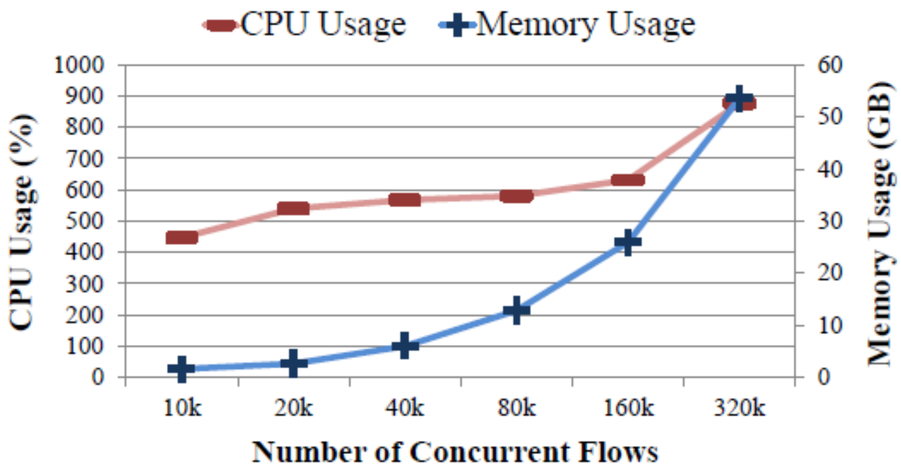
- Prevent attacker from guessing the sampled byte locations
  - Calculate byte location via per-flow key = $HMAC_{Secret\_Key}\{nonce\}$



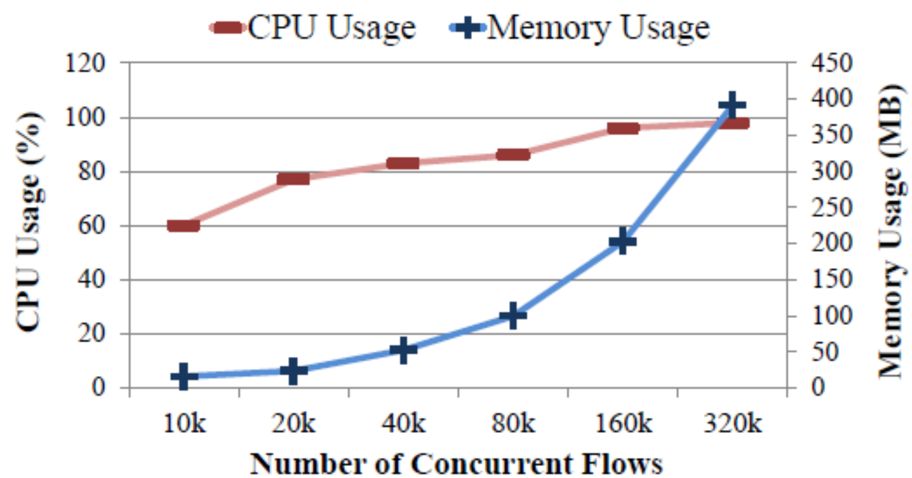
Evaluation

- Environment setup
  - Traffic generator (custom HTTP server) & client
    - Dual Intel Xeon E5-2690 CPU (2.90 GHz, 2 octacores)
    - 64GB RAM
    - Intel 10G NIC with 82599 chipsets
  - d-DPI Abacus
    - Same as traffic generator
  - p-DPI Abacus
    - Intel i7-3770 CPU (3.40 GHz, quadcore)
    - 16GB RAM
    - Intel 10G NIC with 82599 chipsets
- All machines are connected to 10 Gbps Arista 7124 switch
  - Abacus monitors all packets via port mirroring

Microbenchmark



(a) d-DPI



(b) p-DPI

- d-DPI requires large memory for buffering
  - 25.9GB @ 160K flows / 53.6GB @ 320K flows
  - Begins to drop packets 320K flows
- p-DPI requires small memory & CPU
  - 391MB @ 320K flows
  - CPU usage stays under 100% even @ 320K flows

Real Traffic Simulation

- Replay 3G cellular traffic logs
  - Measured in a commercial cellular ISP in South Korea [Woo'13]
  - 11PM – 12AM on July 7th, 2012
  - 61 million flows
  - 2.79 TB in volume
- Inject 100 "free-riding" attacks during replay

Result:

d-DPI & p-DPI accurately detect and report all of the attacks!

Conclusion

- Massive growth in cellular data usage
  - Importance of accurate accounting of cellular traffic
- Cellular ISP dilemma
  - Should we account for TCP retransmissions packets or not?
  - Accounting policies differ between countries
- Vulnerabilities in current accounting system
  - Usage-inflation attack
  - Free-riding attack
- Abacus
  - Manage 100Ks of concurrent flows with a small memory and CPU usage
  - Reliably detect free-riding attack

Thank You!
Any Questions?

http://abacus.kaist.edu

yhwan@ndsl.kaist.edu

Retransmission Rate Measurement

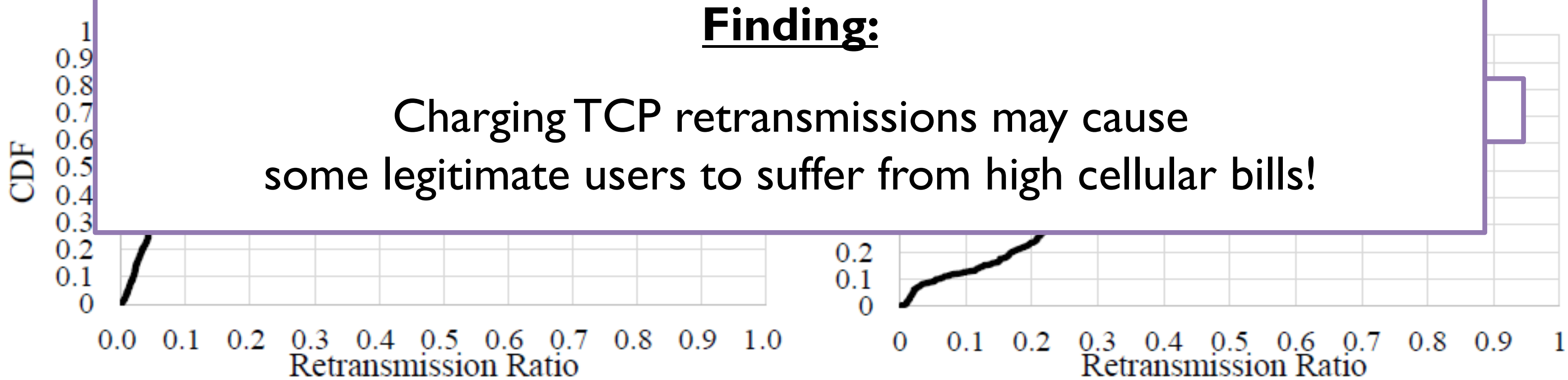
- Measurement environment
  - 11 volunteers (graduate students in KAIST)
  - 38 days (March 22nd – April 29th, 2013)
  - 151,469 flows (3.62GB)
- Packet analyzer
  - Process captured TCP flows
  - Calculate retransmission rate

Overall retransmission rate = 0.4 – 1.7%

Average users do not experience retransmission! But…

Some flows experience high retransmission rates

- CDF of flows with at least one retransmitted packet
  - Worst 10%
    - Daejeon: 40-85% / Princeton: 49-80%
  - Up to 93% retransmission in 3G cellular backhaul link [HotMobile'13]



(a) Daejeon in South Korea

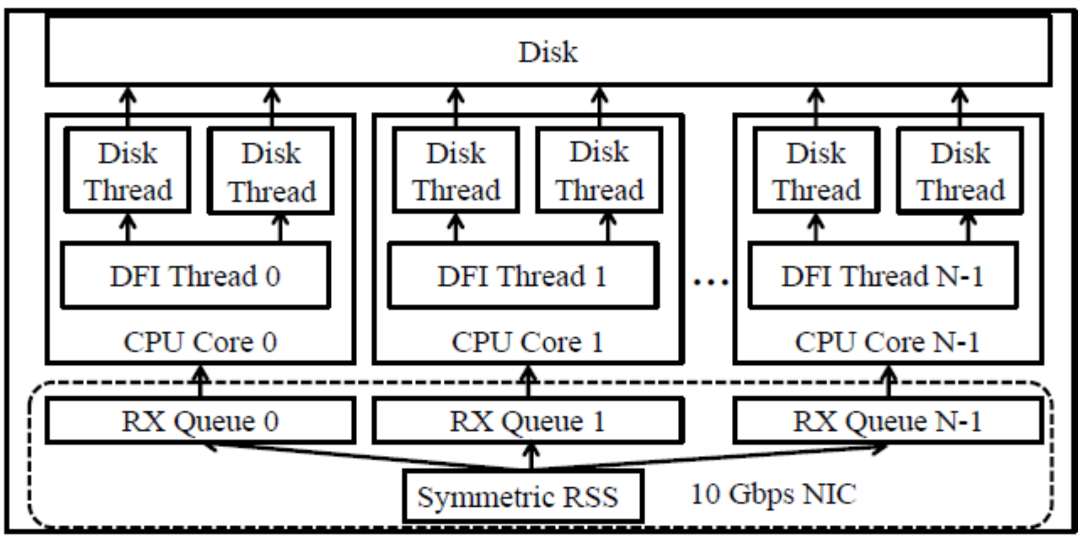
(b) Princeton, NJ in the U.S.

Finding:

Charging TCP retransmissions may cause
some legitimate users to suffer from high cellular bills!

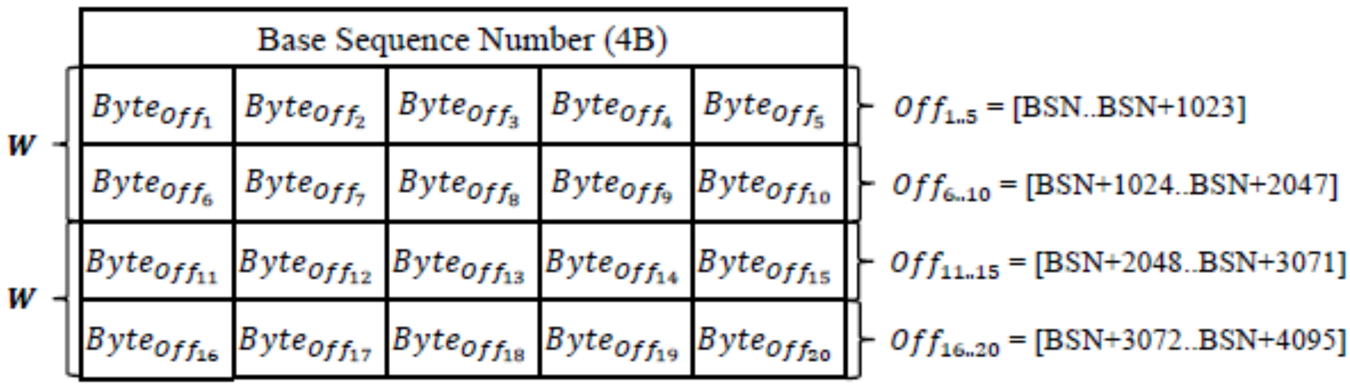
Monbot

- Highly-scalable flow monitoring system [Woo'13]
- PacketShader I/O (PSIO)
  - High-speed packet I/O
- Symmetric Receive-Side Scaling (S-RSS)
  - Map packets in same TCP connection to the same CPU core



Probabilistic DPI

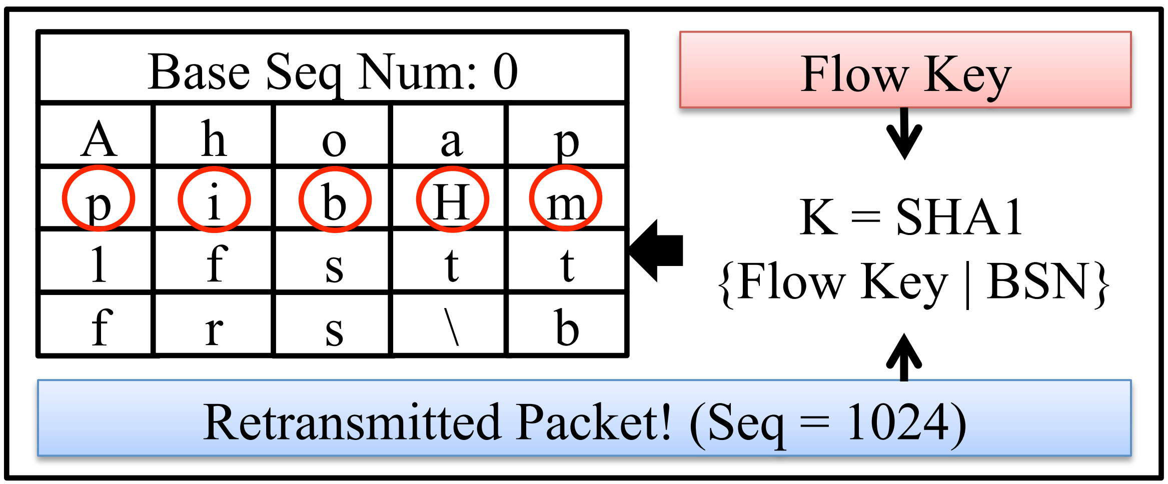
- Store payload by sampling and compare for the sampled data
  - E.g., store 5 bytes out of 1,000-byte → reduce memory by 200x



- 4-byte base sequence number
- Entry
  - Randomly sampled byte between [bsn, bsn + 1023]

p-DPI Byte Sampling

- Prevent attacker from guessing the sampled byte locations
- Random offset: K = SHA1{Flow Key | BSN}
  - Flow Key = $HMAC\downarrow Secret\_Key\{nonce\}$
  - Offset calculation per 1KB buffer → 10 bits to represent each offset
  - N = 5 → Bernstein hash function to produce 64-bit output



Choosing 'n'

- Choice of n-byte sampling
  - Memory space efficiency vs. attack detection accuracy
  - For 1000-byte size packet, attack detection probability: $1 - \frac{{}_{(1000-y)}C_x}{{}_{1000}C_x}$