TCP Improvements for Intermittent Connectivity

Simon Schütz, Lars Eggert, Stefan Schmid and Marcus Brunner
NEC Network Labs
Heidelberg, Germany

Friday, October 14, 2005

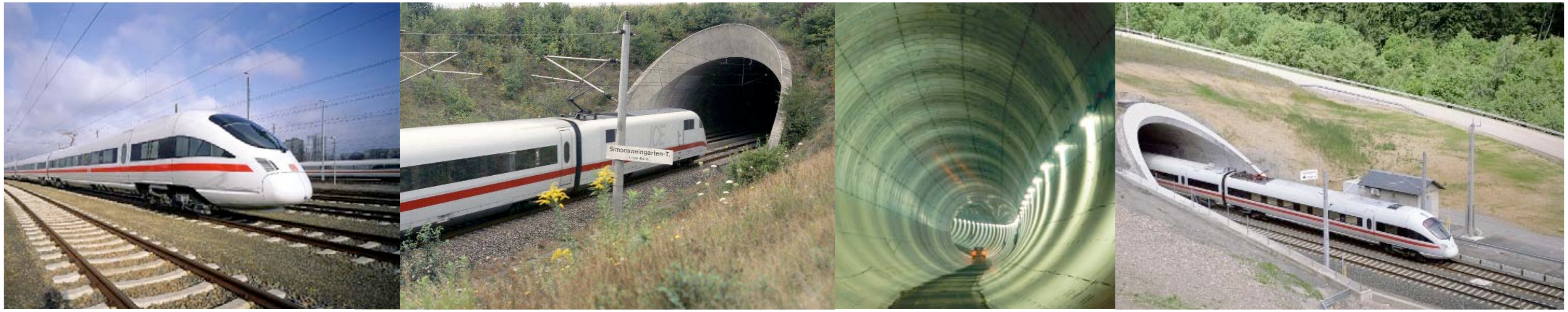
NEC Network Labs

- ~40 research staff + ~12 students
- next-generation internetworking, ad hoc & sensor security, car-to-car communication, mobile services
- EU and national German research projects
- IETF, 3GPP and OMA standardization
- Heidelberg, Germany and London, UK

Outline

- motivation
- TCP issues
- solution idea & components
- experimental evaluation of benefits
- ongoing & future work

Motivation

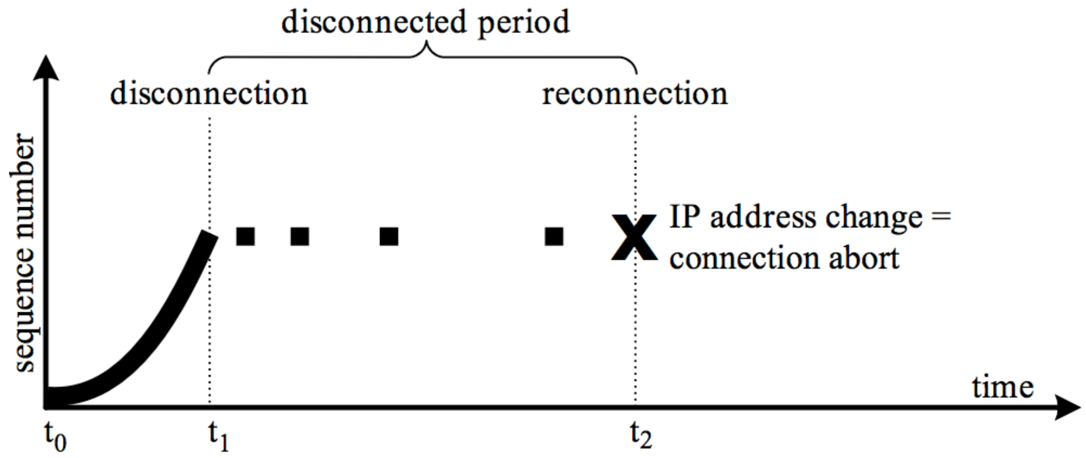


- connectivity disruptions can occur along an end-to-end path
- node mobility, equipment failure, nomadic use
- TCP operates inefficiently under intermittent end-to-end connectivity or can even fail

TCP Problems

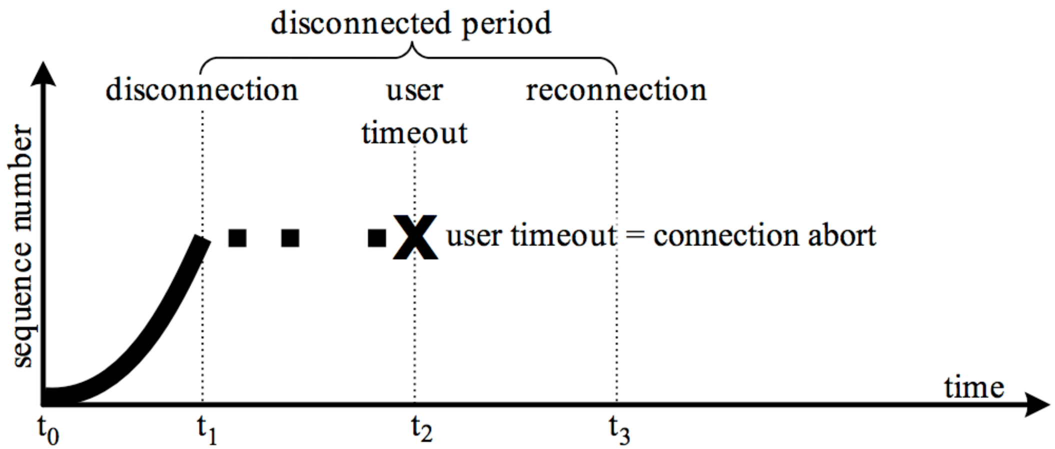
- connection aborts
  - IP address changes after mobility
  - prolonged absence of connectivity
- poor performance
  - retransmission behavior inefficient or too aggressive

IP Address Changes



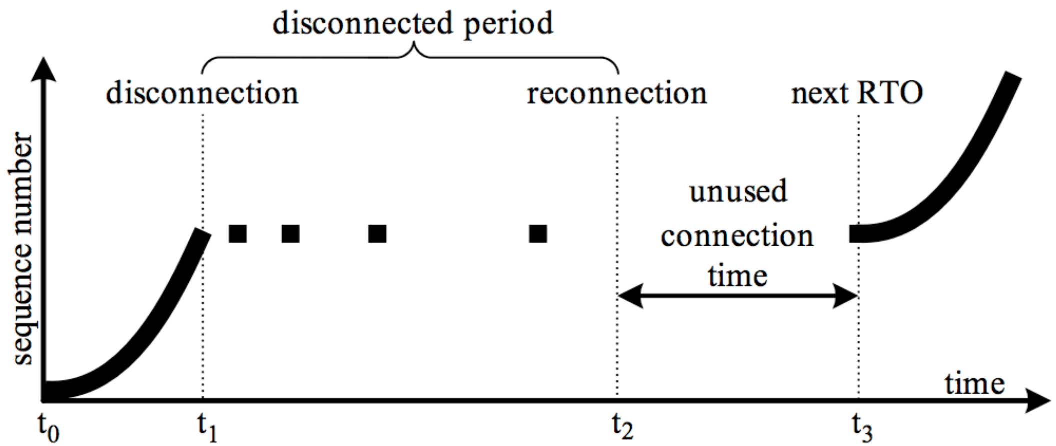
- connection endpoints bind to IP addresses
- IP addresses can change, e.g., due to mobility
- connection aborts

Prolonged Disconnection



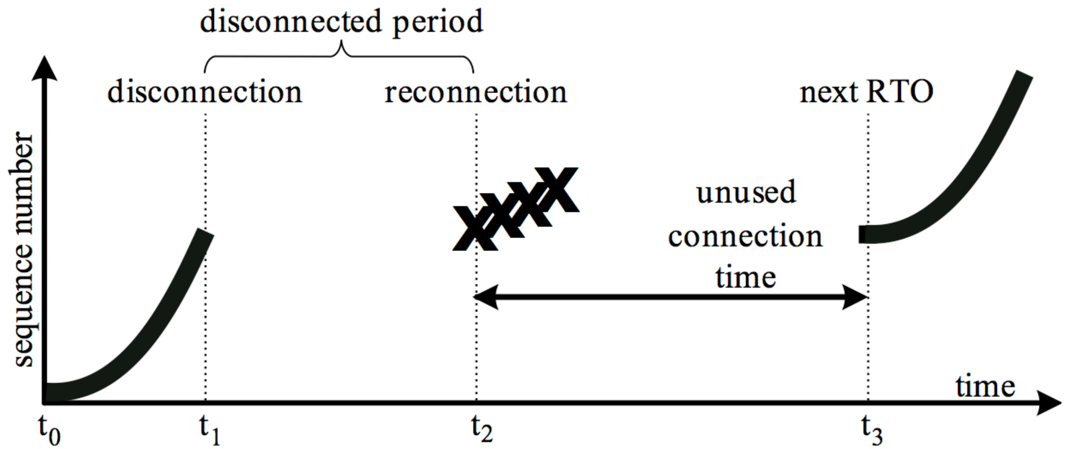
- RFC spec defines "user timeout" as max. time sent data may remain un-ACK'ed
- default is O(minutes)
- connections abort during longer disruptions

Inefficient Retransmit (1)



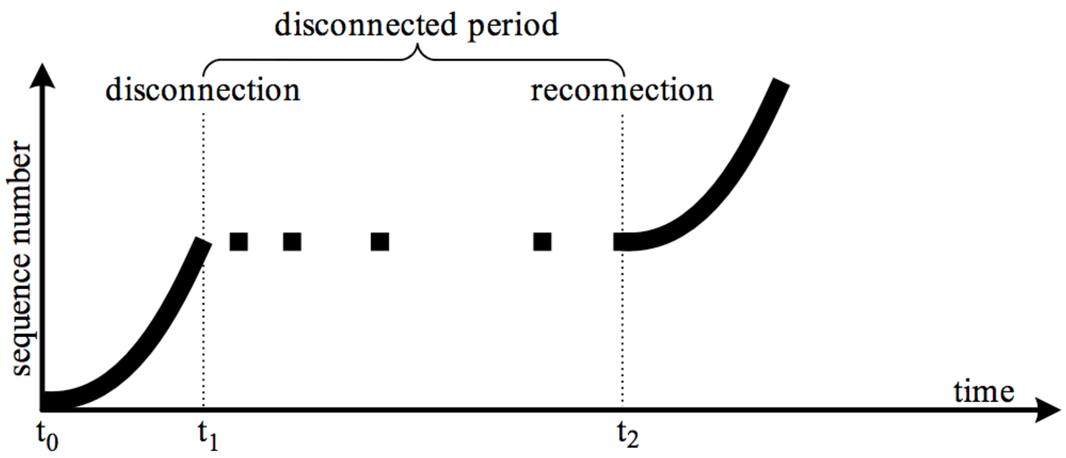
- during (longer) disconnections, TCP periodically attempts retransmit
- attempts are exponentially timed
- inefficient! wastes connectivity time after reconnect, which may be short

Inefficient Retransmit (2)



- TCP may be too aggressive when resuming transmission after reconnection, if the path characteristics have changed
- may interfere with concurrent traffic and cause additional delays due to self-induced loss

Ideal Behavior



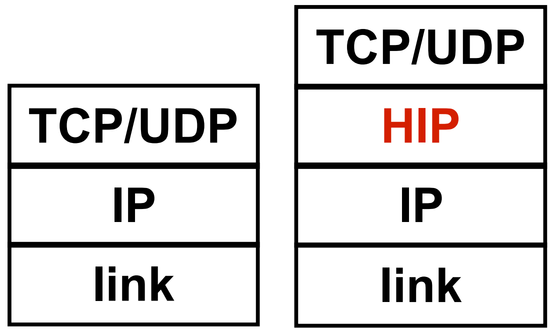
- ideally, TCP would not abort and efficiently and conservatively resume transmission immediately upon reconnection

Solution Components

- <u>tolerate IP address changes:</u>
  mobility management solution (we use HIP)
- <u>user timeout:</u>
  new TCP option to exchange UTO information
- <u>retransmission behavior:</u>
  make TCP aware of connectivity indications and reset congestion control state

Host Identity Protocol

- new layer between network and transport layers
- connections bind to host identifiers instead of IP addresses



- mobility mechanism: dynamic HIP→IP mapping
- intrinsic security: host identifiers are cryptographic hashes of public keys
- authentication and IPsec for encryption

TCP User Timeout Option

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---------------+---------------+-+-----------------------------+
|     Kind      |  Length = 4   |G|        User Timeout         |
+---------------+---------------+-+-----------------------------+
```

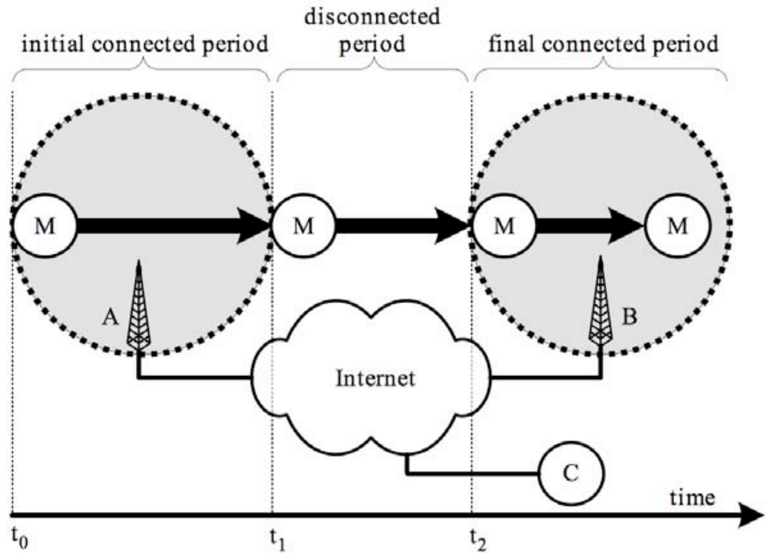
- enable per-connection user timeouts instead of system-wide default of O(minutes)
- exchange user timeout values between peers
- shorter- and longer-than-default timeouts
  - maximum is 2^{15} minutes > 22 days, minimum can be O(seconds)
- draft-ietf-tcp-uto currently under IETF discussion

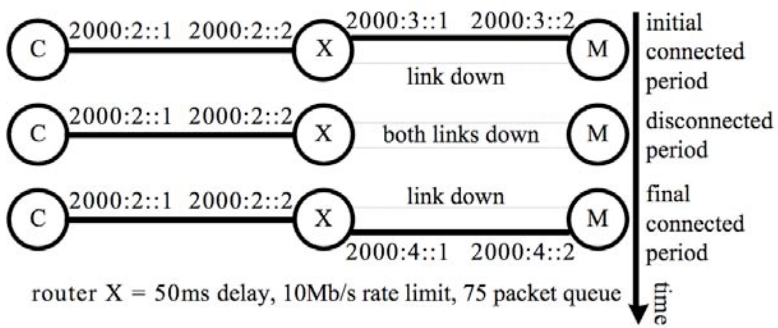
Retransmit Improvements

- idea: add <u>speculative retransmission attempt</u> on “connectivity indicator” (CI)
- CIs signal that connectivity to the peer may be restored
  - link-layer events on end hosts
  - MobileIP binding update, HIP readressing
- combine with mechanism to reset congestion state and force slow-start path reprobing
- draft-eggert-tcpm-retransmit-now & draft-swami-tcp-lmdr; being merged

Experimental Evaluation

- single bulk data transfer between M and C (25 MB in ~22 sec)
- M "moves" from access point A to B, then stops
- emulate mobility through dynamic reconfiguration of Ethernet interfaces



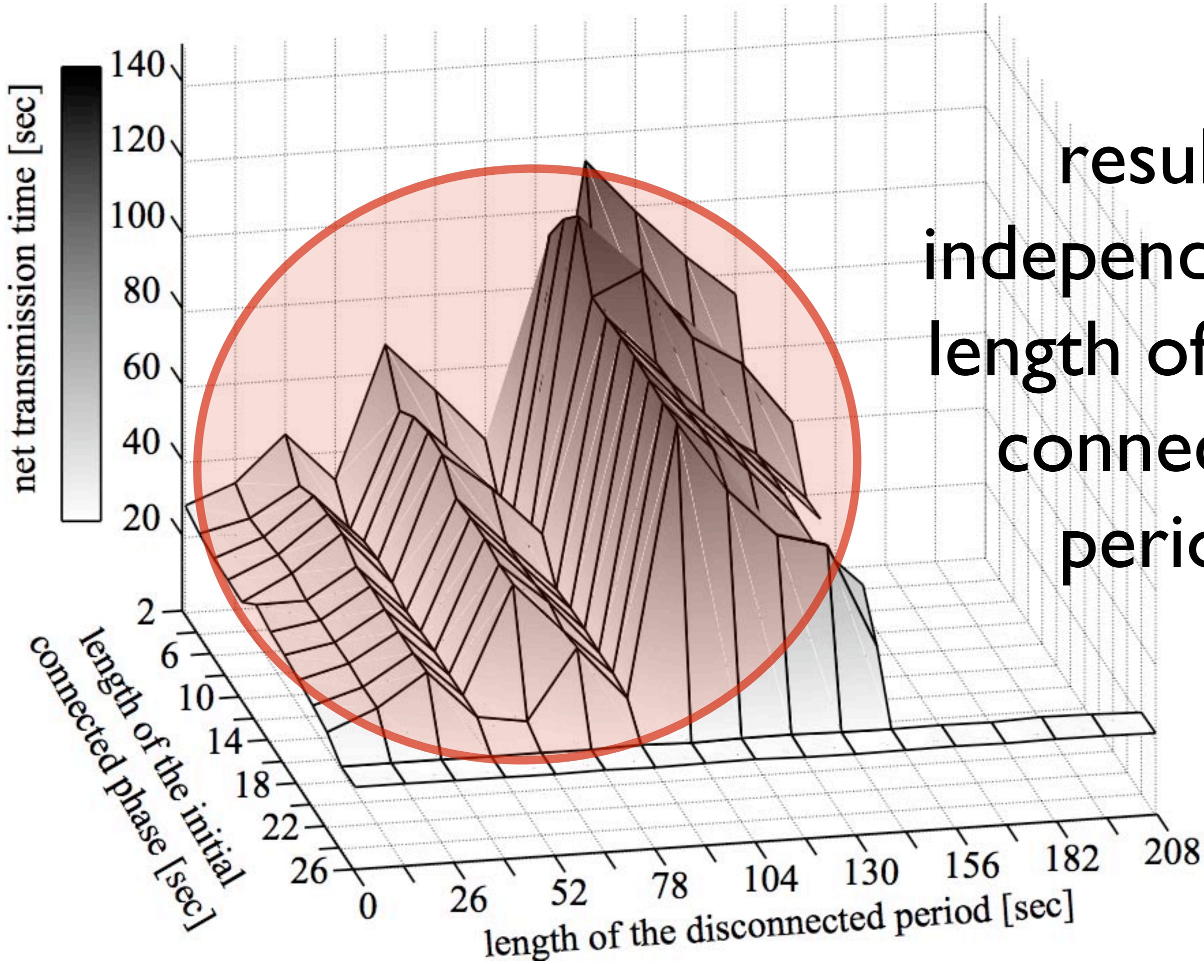


Empowered by Innovation NEC

Parameters & Metric

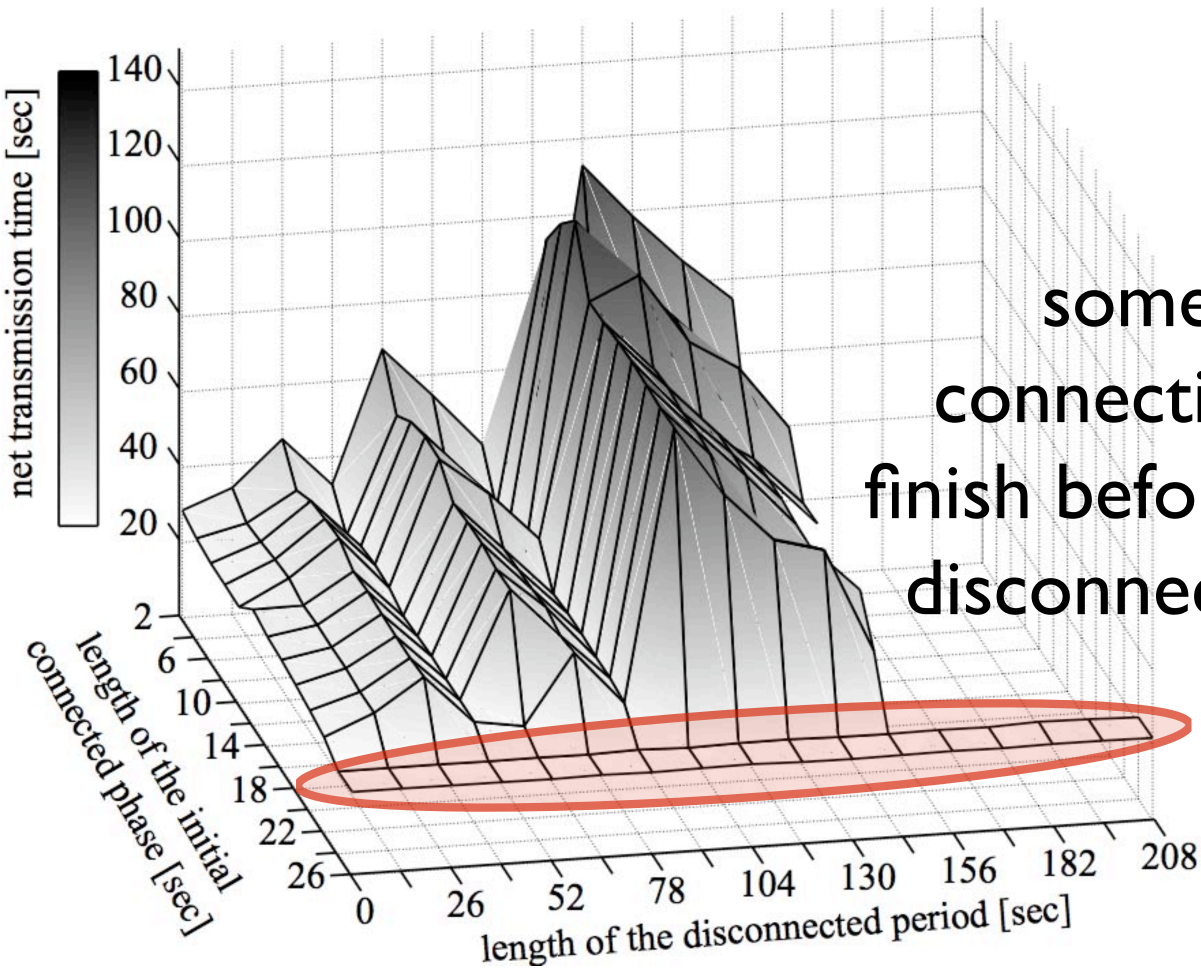
- parameters
  - length of initial connected period: 2-26 sec
  - length of disconnected period: 0-208 sec
- metric: net transmission time
  - factor out disconnected periods
  - compare efficiency during connected periods

Baseline: Vanilla HIP



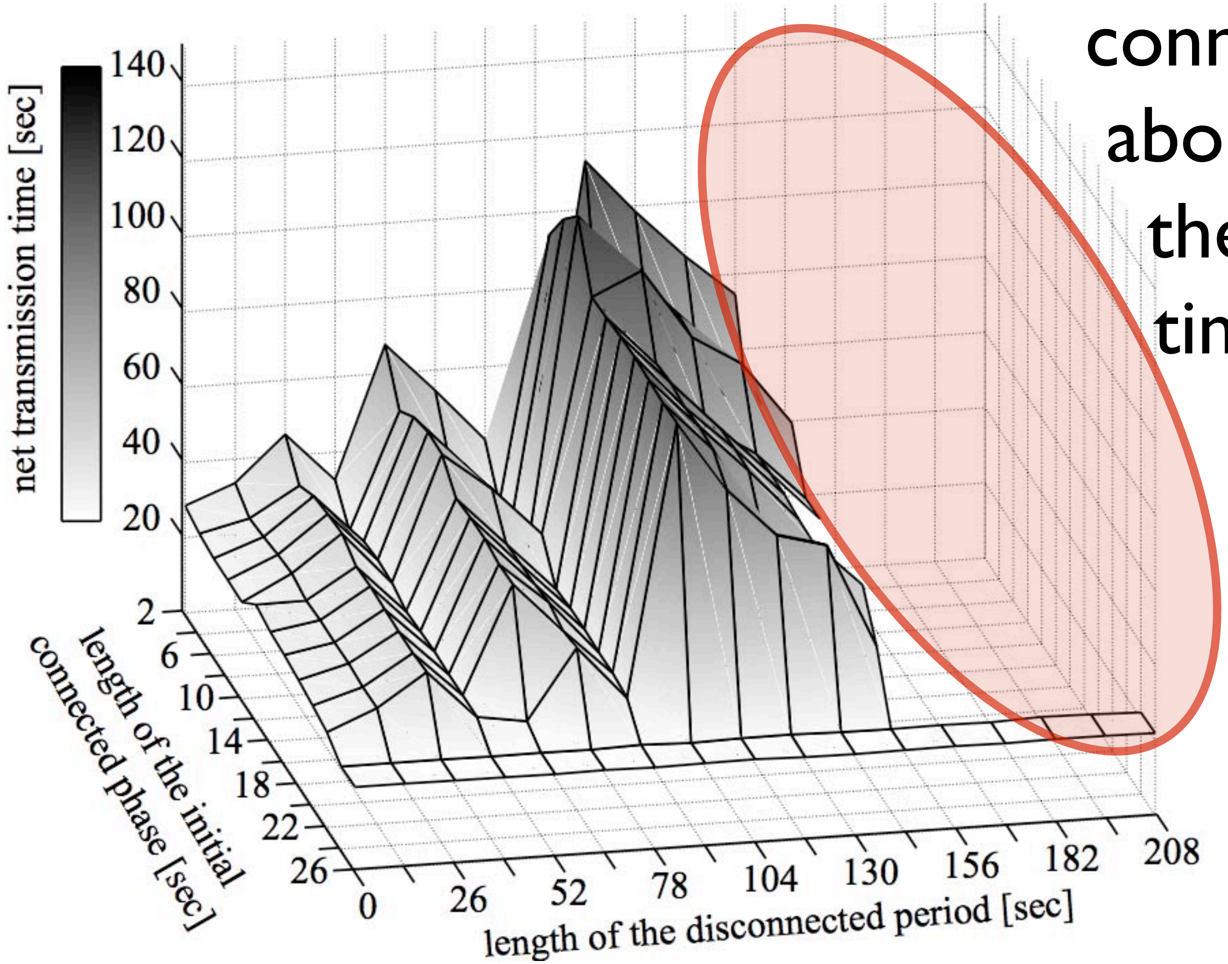
results independent of length of initial connected period

Baseline: Vanilla HIP



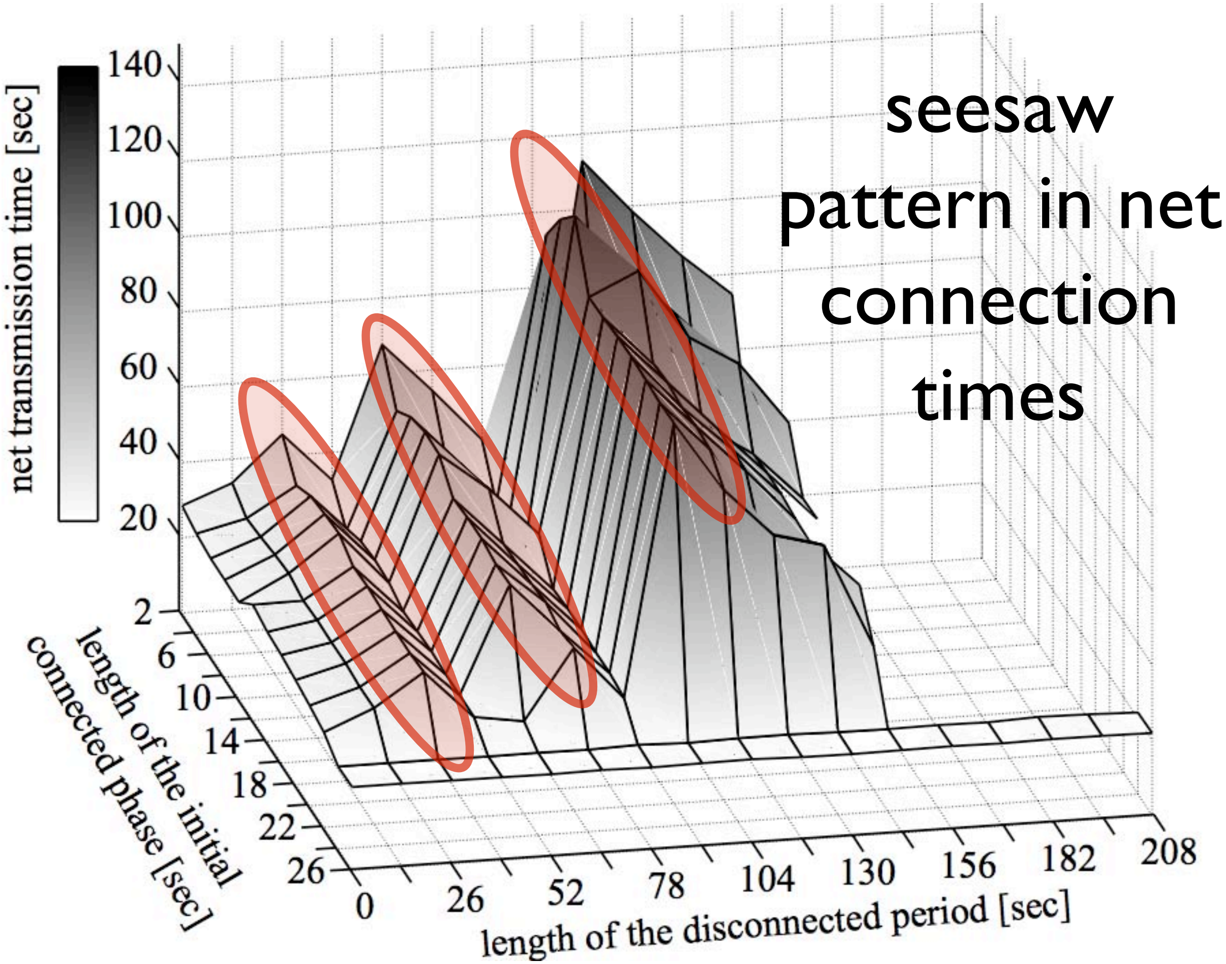
some connections finish before the disconnection

Baseline: Vanilla HIP



connections abort after the user timeout

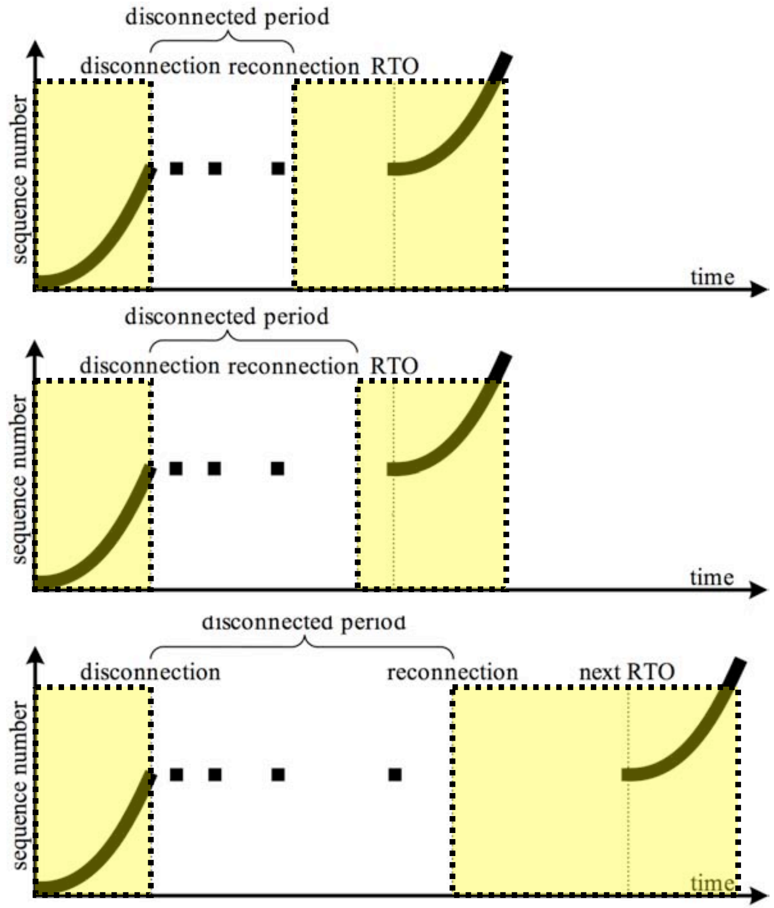
Baseline: Vanilla HIP



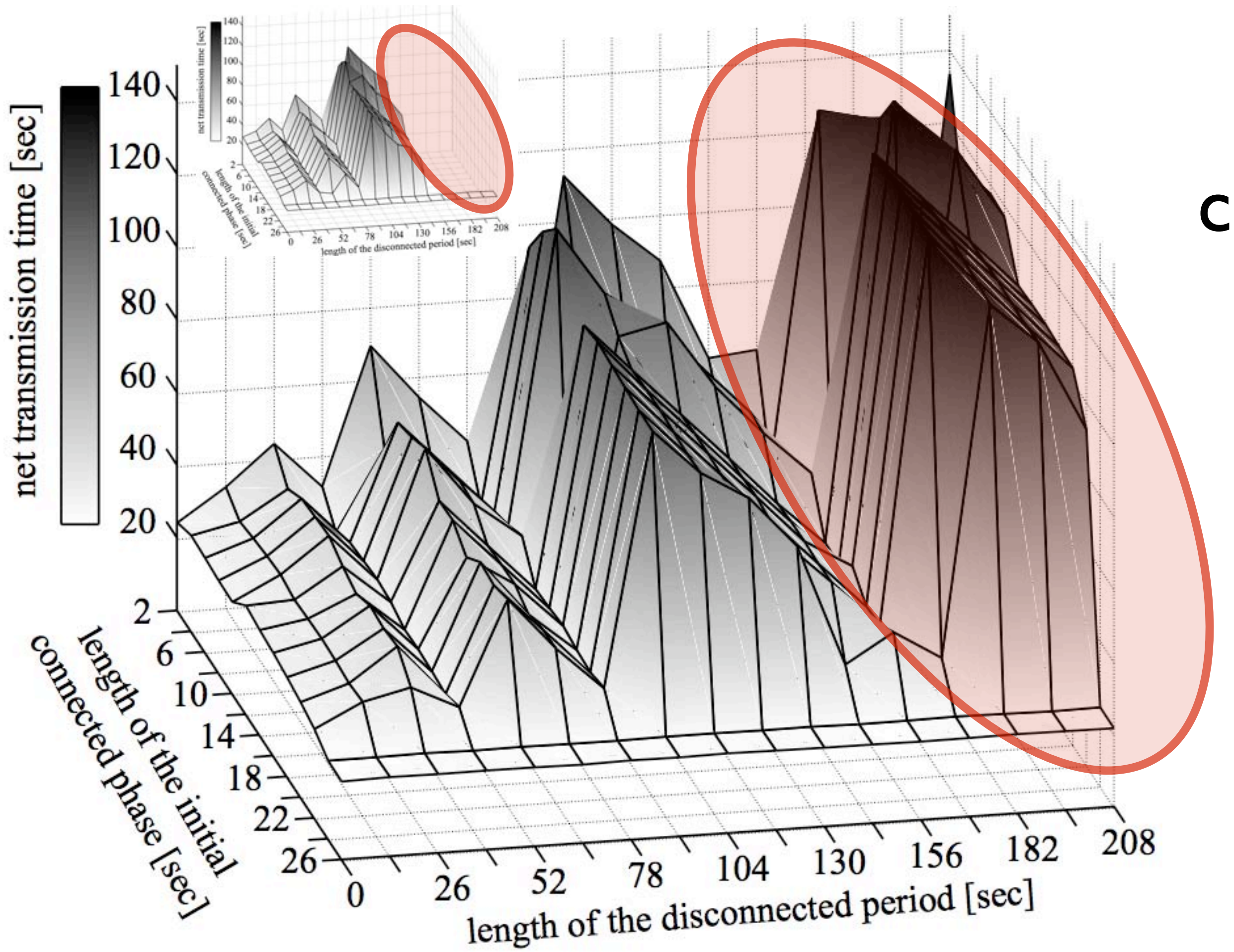
seesaw pattern in net connection times

Seesaw Effect

- net transmission times depend on the timing of reconnections and retransmission attempts
- <u>counter-intuitive:</u> longer disconnections can shorten net transmission times



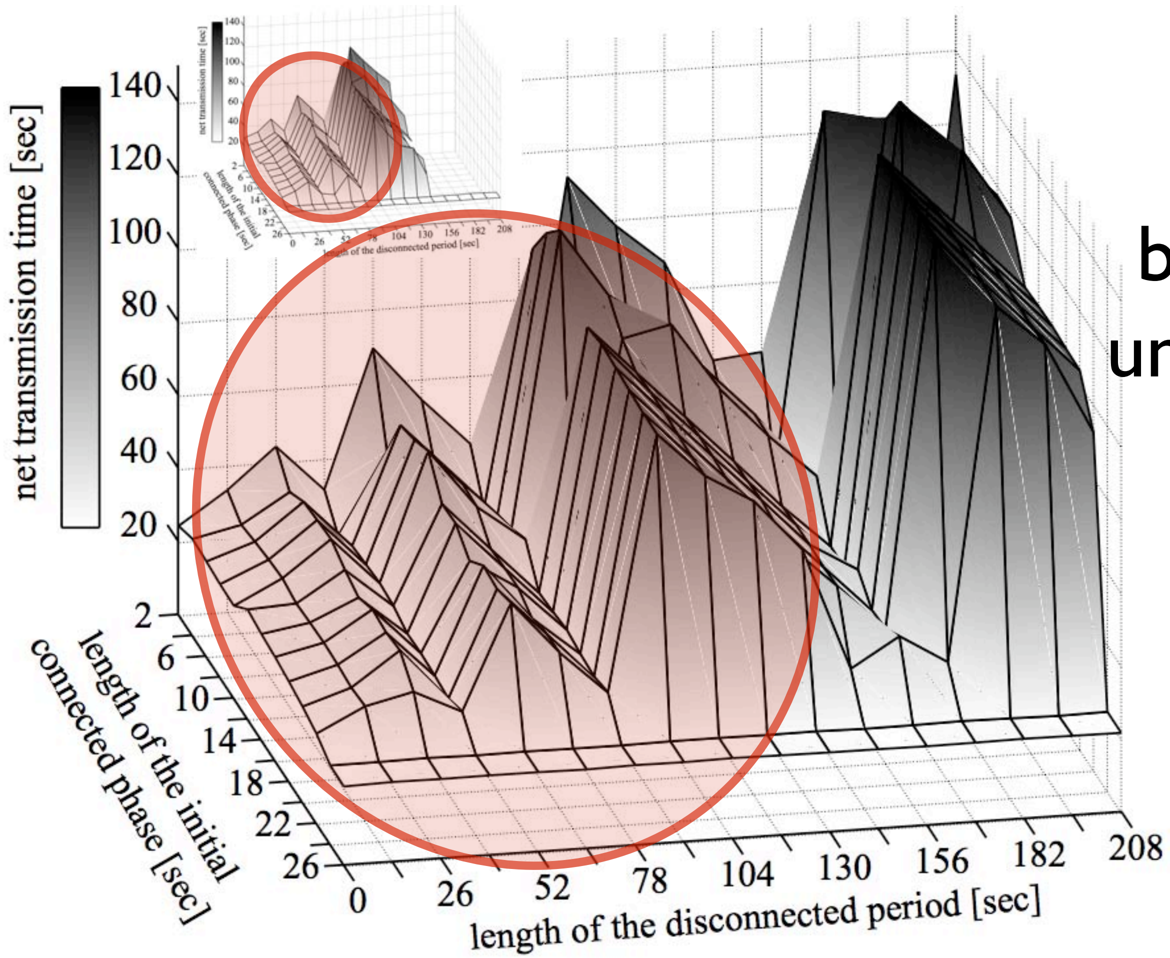
HIP + TCP/UTO



no connection aborts

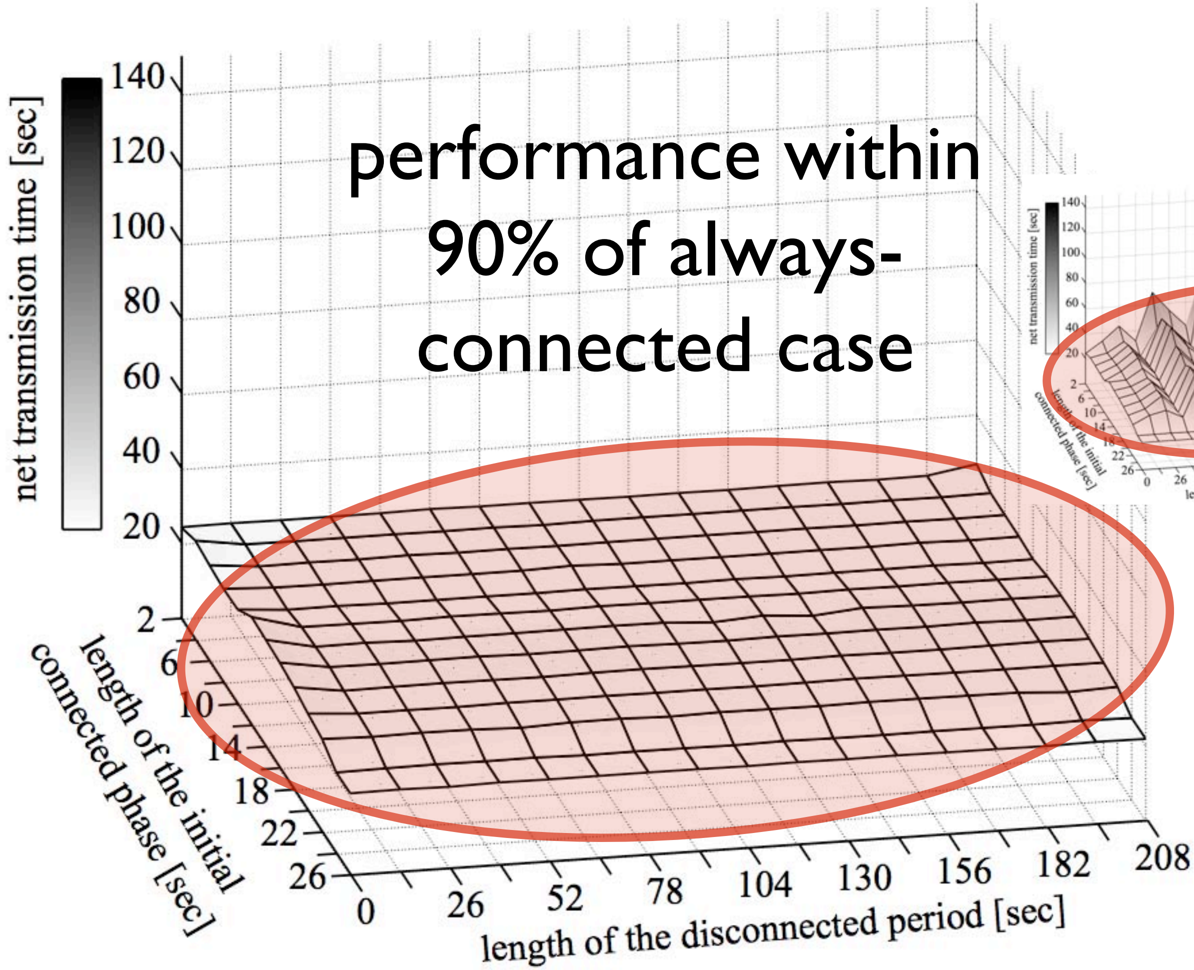
(but long transmit times)

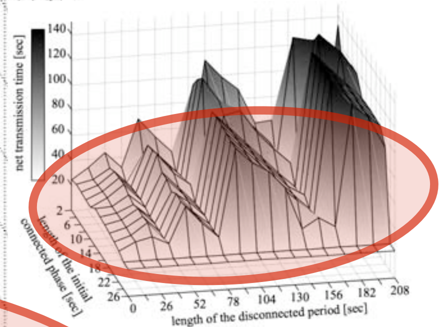
HIP + TCP/UTO



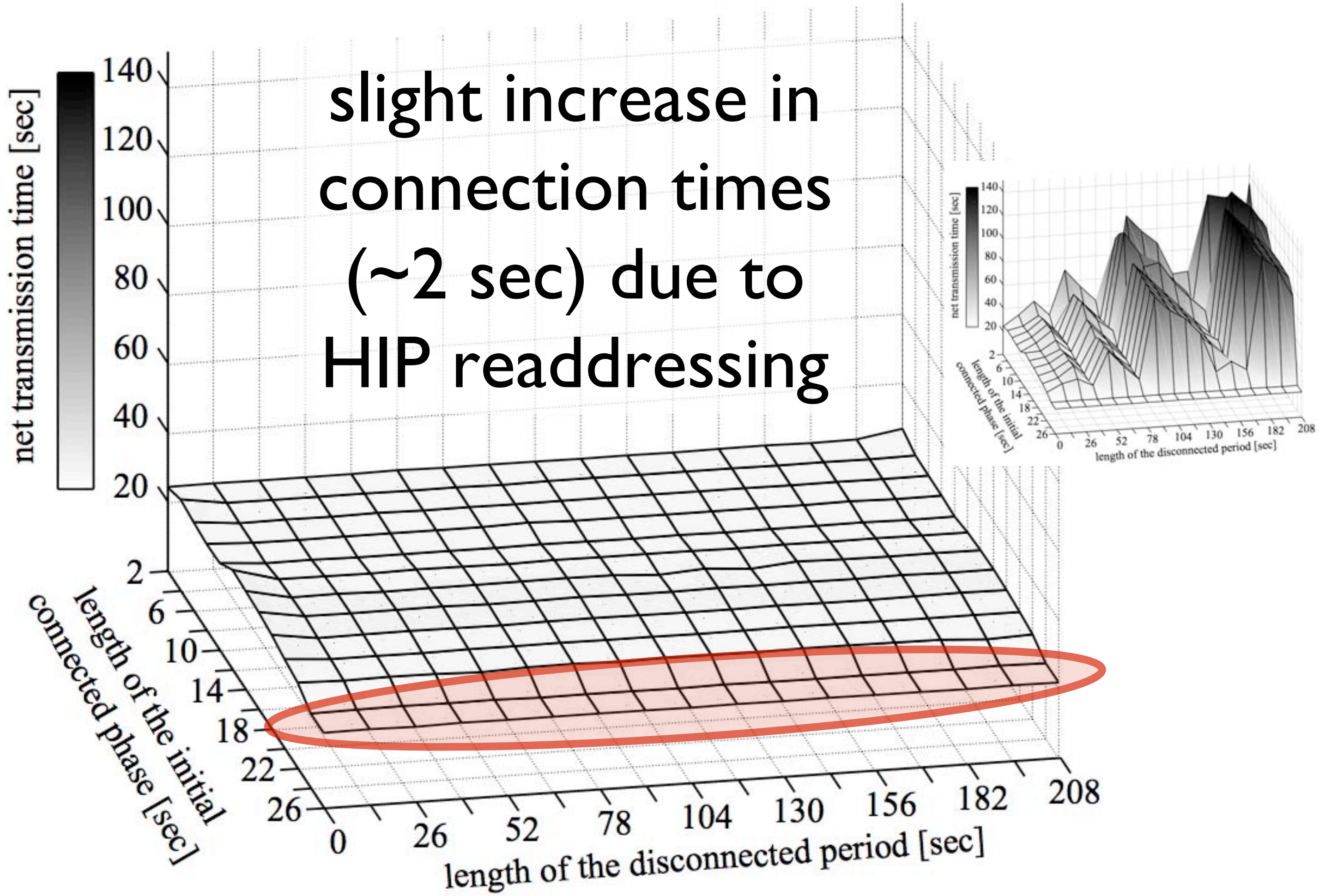
other behavior unchanged

“Retransmit Now”





“Retransmit Now”



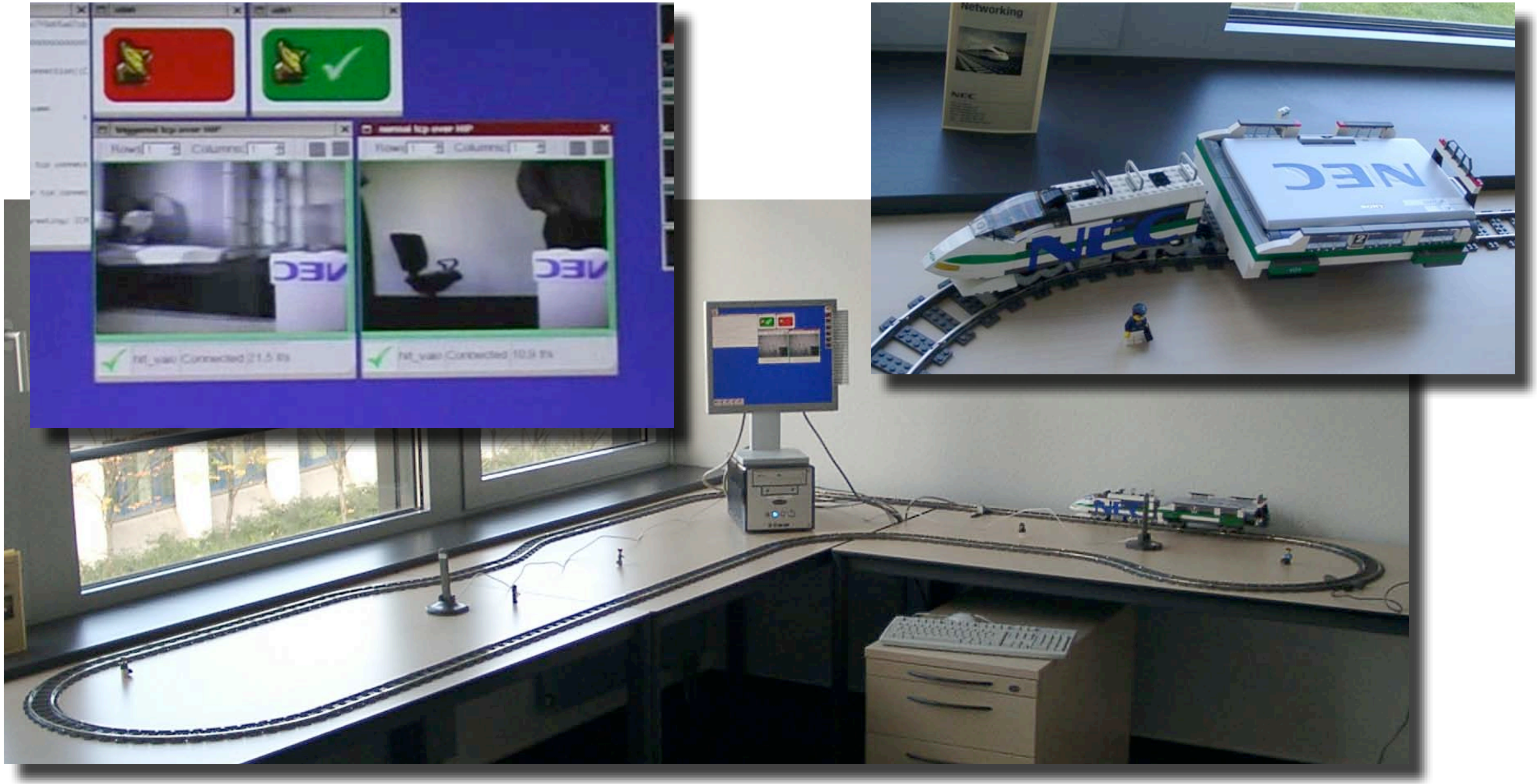
Related Work

- delay-tolerant networking
  - more complex: no end-to-end path even when connected
- disconnection tolerance
  - mobile TCP socket, ROCKS/RACKS, migrate, drive-thru Internet
- performance enhancements
  - implicit link-up notification, smart link layer, TCP-F, ELFN

Ongoing & Future Work

- standardization (IETF TCPM WG)
- spec-conforming implementation revision
- real-life experimental evaluation
- meta: acquire more funding :-)

Demonstrator



Summary

- analysis of TCP under intermittent connectivity: connections break or show poor performance
- proposed solution
  - HIP (or mother mobility management scheme)
  - TCP User Timeout Option +
    TCP Retransmission Improvements
- illustrated benefit through experiments: improve performance to within 90% of constant connectivity

References

- Protocol Enhancements for Intermittently Connected Hosts. Simon Schütz, Lars Eggert, Stefan Schmid and Marcus Brunner. ACM Computer Communication Review (CCR), Vol. 35, No. 3, July 2005, pp. 5-18.

- TCP User Timeout Option. Lars Eggert and Fernando Gont. Internet Draft draft-ietf-tcpm-tcp-uto-01, Work in Progress, July 2005.

- TCP Extensions for Immediate Retransmission. Lars Eggert, Simon Schütz and Stefan Schmid. Internet Draft draft-eggert-tcpm-tcp-retransmit-now-02, Work in Progress, June 2005.

- Lightweight Mobility Detection and Response (LMDR) Algorithm for TCP. Yogesh Swami, Khiem Le and Wesley Eddy. Internet Draft draft-swami-tcp-lmdr-05, Work in Progress, August 2005.