Networking Enhancements for a Dynamic Internet

Lars Eggert
lars.eggert@netlab.nec.de
NEC Network Labs
Heidelberg, Germany

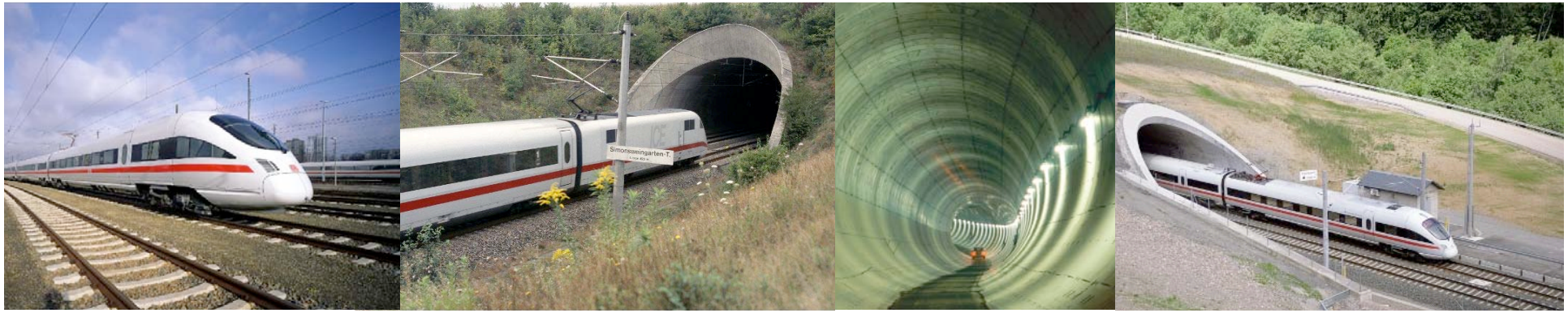
Friday, May 19, 2006

NEC Network Labs

- ~45 research staff + ~15 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

Part I

Motivation



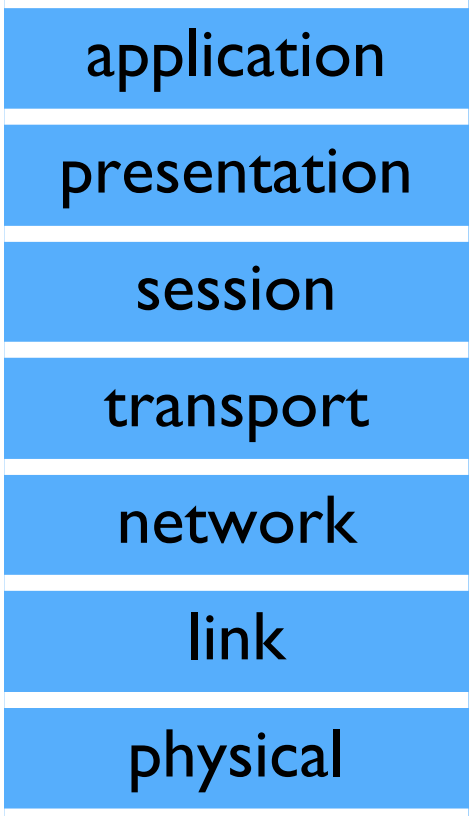
- connectivity disruptions can occur along an end-to-end path
- node mobility, equipment failure, nomadic use
- Internet protocols operate inefficiently under intermittent end-to-end connectivity or can even fail

Why?

- evolution
- network is much more dynamic now than when Internet protocols were designed
- mobile nodes, links of vastly different characteristics, many more services, etc.
- original abstractions have started to limit performance and operation of the Internet

Abstractions

- layers in the network stack can be seen as "virtual machines"
- expose well-defined set of operations & information
- hide intricacies of a layer (& layers below) to its users
- this is generally good!



Example: Network Layer

- abstraction is something like
  - "will deliver your packets in some order"
  - "may deliver multiple copies of some packets"
  - "may not deliver some others"
- hides other network-layer functionality, such as
  - packet fragmentation/reassembly
  - route computation and forwarding

But in Practice

- users of the network-layer abstraction have made additional assumptions about it
  - and in the past, they have been true
- these assumptions are the basis of many key transport-layer mechanisms, such as
  - congestion control
  - flow control
  - reliability mechanisms

Assumptions

- hosts remain at the network port identified by an IP address for long times
- packets between the same src/dst addresses mostly follow the same path
- paths change on time scales that are orders of magnitude greater than the RTT
- path characteristics change on similarly large time scales
- connectivity along a path is very rarely disrupted

Reality Check

- many of these assumptions are no longer generally true
- especially with recent/proposed network layer extensions, such as MIP, HIP, SHIM6, NEMO, etc.
- but also simply because recent link technologies are different
  - network-based mobility
  - link-layer retransmissions
  - non-congestion packet loss

Consequence

- traditional transport mechanisms are performing less well than in the past
- this is not news: gazillion of "optimize transport protocol X for scenario Y" approaches
  - where X is mostly TCP
  - and Y = satellites, 802.11, GSM, 3G, ad hoc net, high bit-error links, etc.
- but vast majority of these are band aids
  - specific fixes for limited scenarios
  - not appropriate for a general-purpose Internet

What Is Appropriate?

- idea: extend the "virtual machine" abstraction that the network layer provides to its users
- but do it in a way that is generic
  - independent of specific network layer extensions
  - independent of specific link technologies

Approach

- extension consists of additional pieces of information or notification about network-layer events
- should be advisory and optional: transports shouldn't depend on them
- new transport mechanisms could then act on this information to improve operation and performance

Not a New Idea

- other proposals are already enhancing the network-layer "virtual machine" abstraction
  - ECN ("I'm about to start dropping these packets")
  - Quickstart ("you may send me packets at rate n")
  - TRIGTRAN (but this is broader)
- and don't forget about ancient stuff like ICMP
  - unreachables: "this host/network is not here"
  - original source quench: "stop sending so fast"

Generalize!

- “virtual machines” - or communication primitives provided by different transport protocols to the apps - are also restricted
- richer transport “virtual machines” can improve app operation and performance
- similarly for the interface below the network layer (towards the link layers)

Why Is This Hard?

- it's easy to optimize for one particular lower layer ("TCP over 802.11" hacks)
- it is hard to identify a small (minimal?) set of generic pieces of information or signals that:
  - can be provided by many underlying technologies (in different ways)
  - are expressive enough to allow significant performance improvements for many different uses
- there is some research left to be done

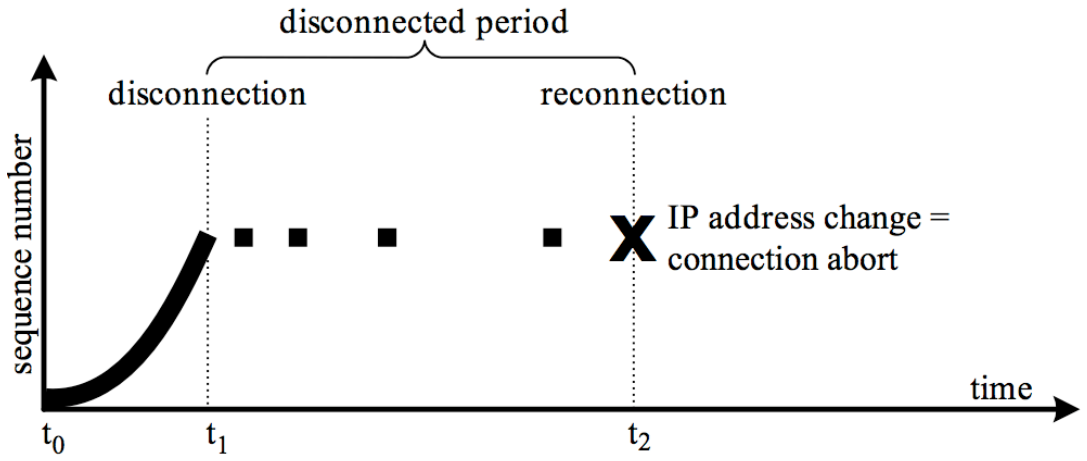
Part 2

- Experiments with a TCP Enhanced for Operation across Intermittently-Connected Paths

TCP Problems

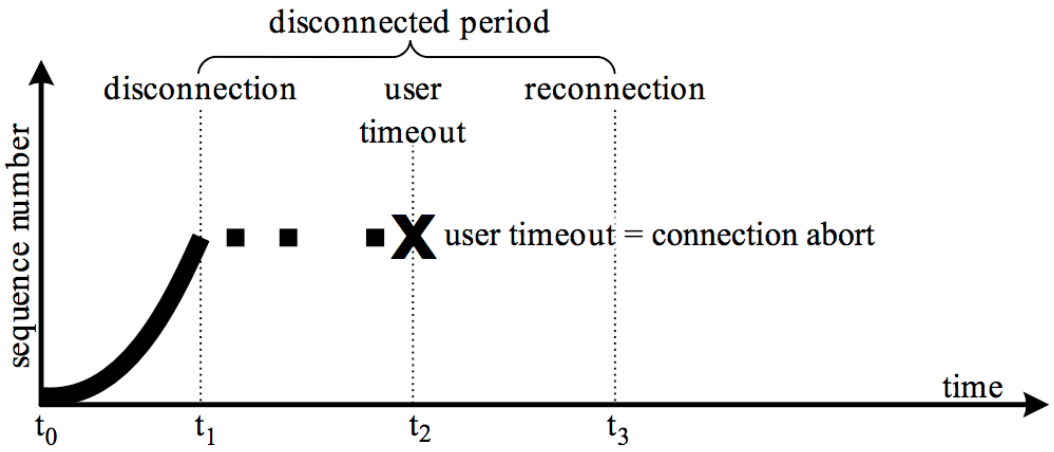
- intermittent connectivity breaks TCP
- connection aborts
  - IP address changes after mobility
  - prolonged absence of connectivity
- poor performance
  - retransmission behavior inefficient or too aggressive

IP Address Changes



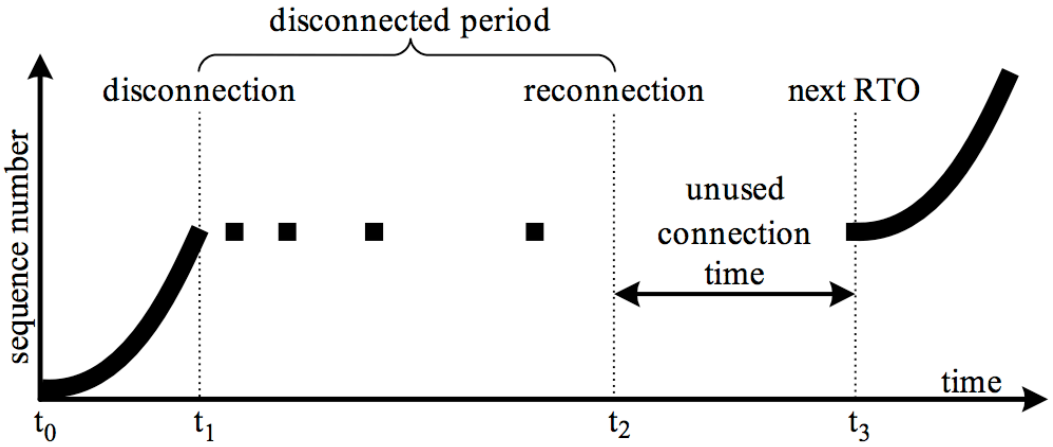
- connection endpoints bind to IP addresses
- IP addresses can change, e.g., due to mobility
- connection aborts

Prolonged Disconnection



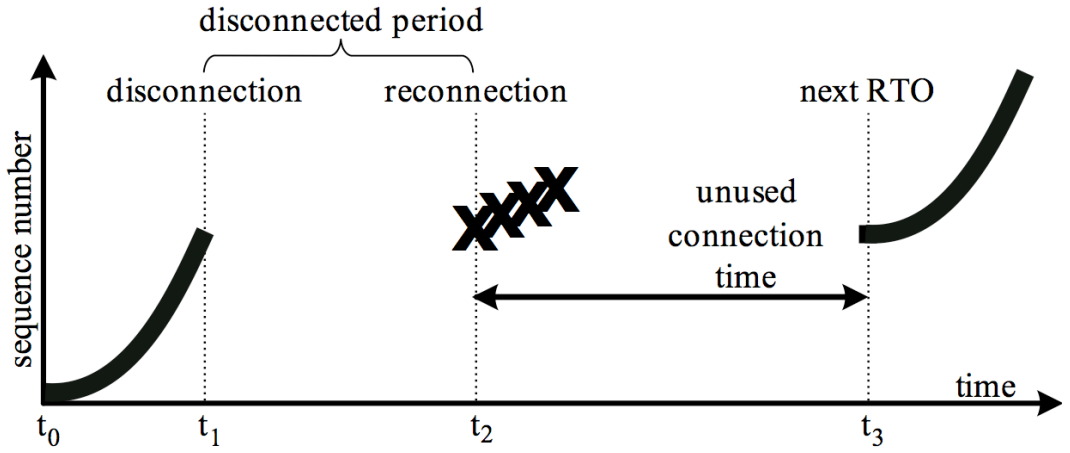
- RFC spec defines "user timeout" as max. time sent data may remain un-ACK'ed
- default is O(minutes)
- connections abort during longer disruptions

Inefficient Retransmit (1)



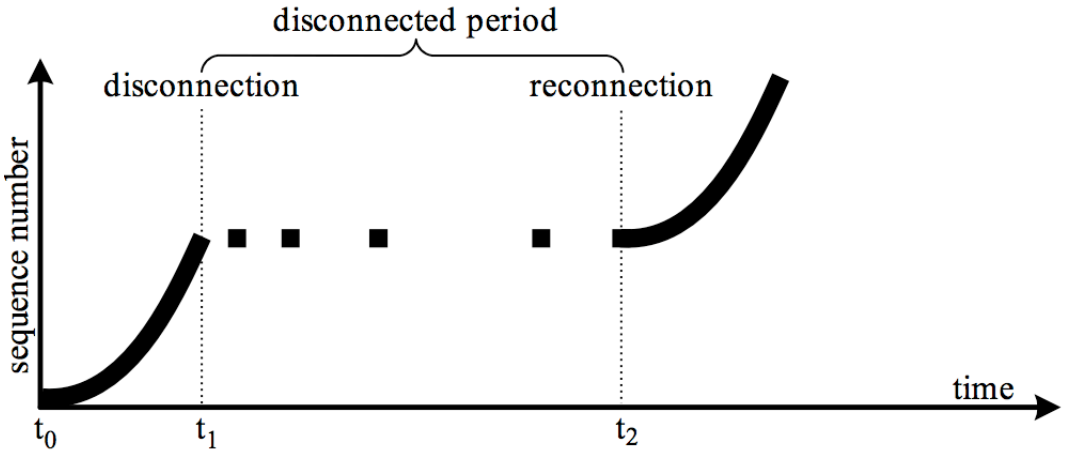
- during (longer) disconnections, TCP periodically attempts retransmit
- attempts are exponentially timed
- inefficient! wastes connectivity time after reconnect, which may be short

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



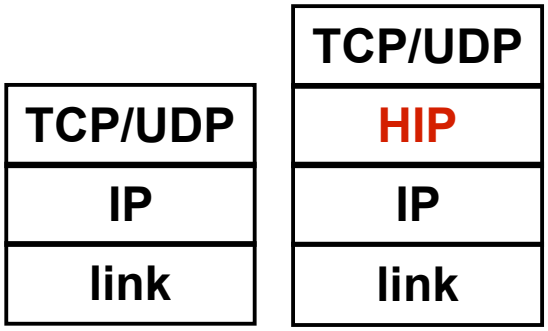
- ideally, TCP would not abort and efficiently and conservatively resume transmission immediately upon reconnection

Solution Components

- tolerate IP address changes:
  mobility management solution (we use HIP)
- user timeout:
  new TCP option to exchange UTO information
- response mechanism to lower-layer information:
  make TCP act on "path connectivity has changed" triggers

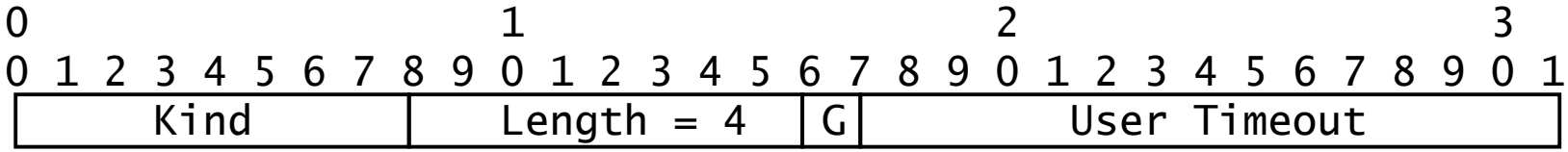
Host Identity Protocol

- 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



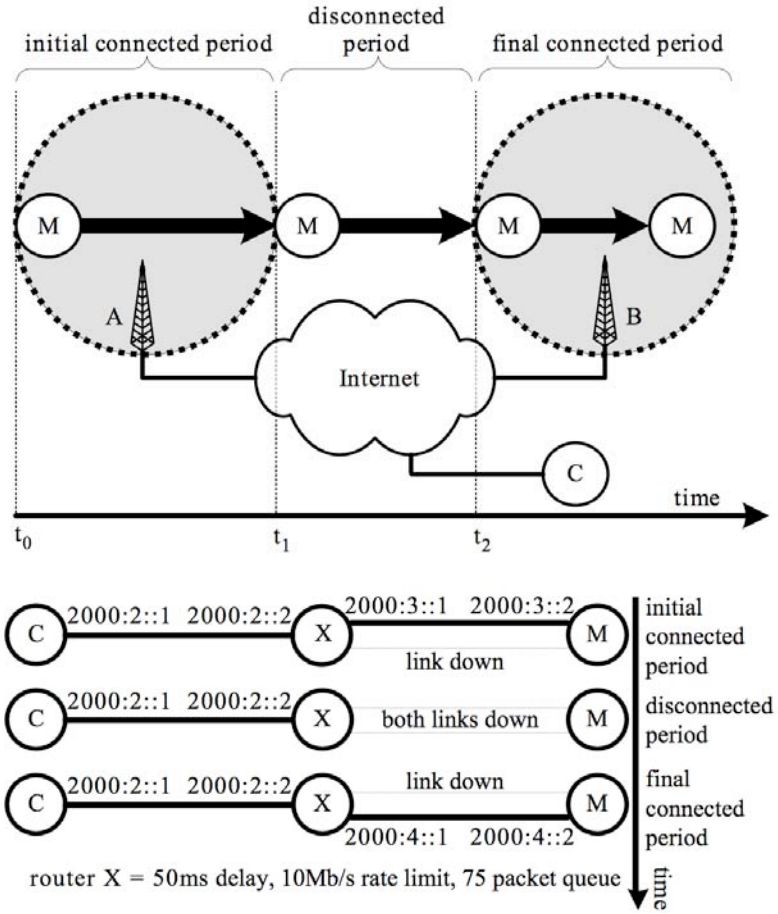
- enable per-connection user timeouts instead of system-wide default of O(minutes)
- exchange user timeout values between peers
- shorter- and longer-than-default timeouts
  - maximum is 2^{15} minutes > 22 days, minimum can be O(seconds)

Retransmit Improvements

- idea: add speculative retransmission attempt on "path connectivity has changed" indicator
- when disconnected, may mean that connectivity to the peer is restored
  - link-layer events on end hosts
  - MobileIP binding update, HIP readdressing
- (other meaning & response in steady state: reset congestion state and force slow-start re-probing)

Experimental Evaluation

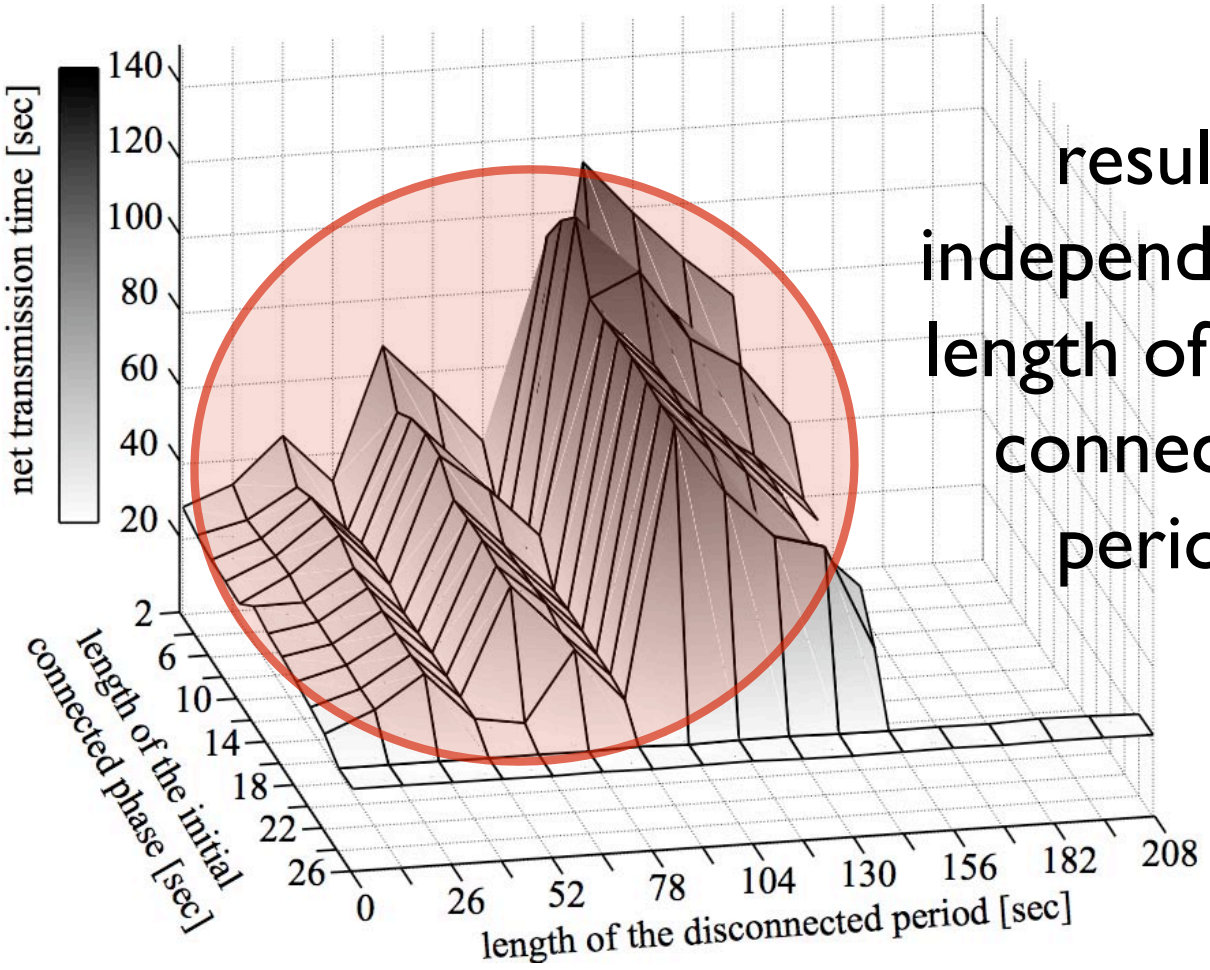
- single bulk data transfer between M and C (25 MB in ~22 sec)
- M "moves" from access point A to B, then stops
- emulate mobility through dynamic reconfiguration of Ethernet interfaces



Parameters & Metric

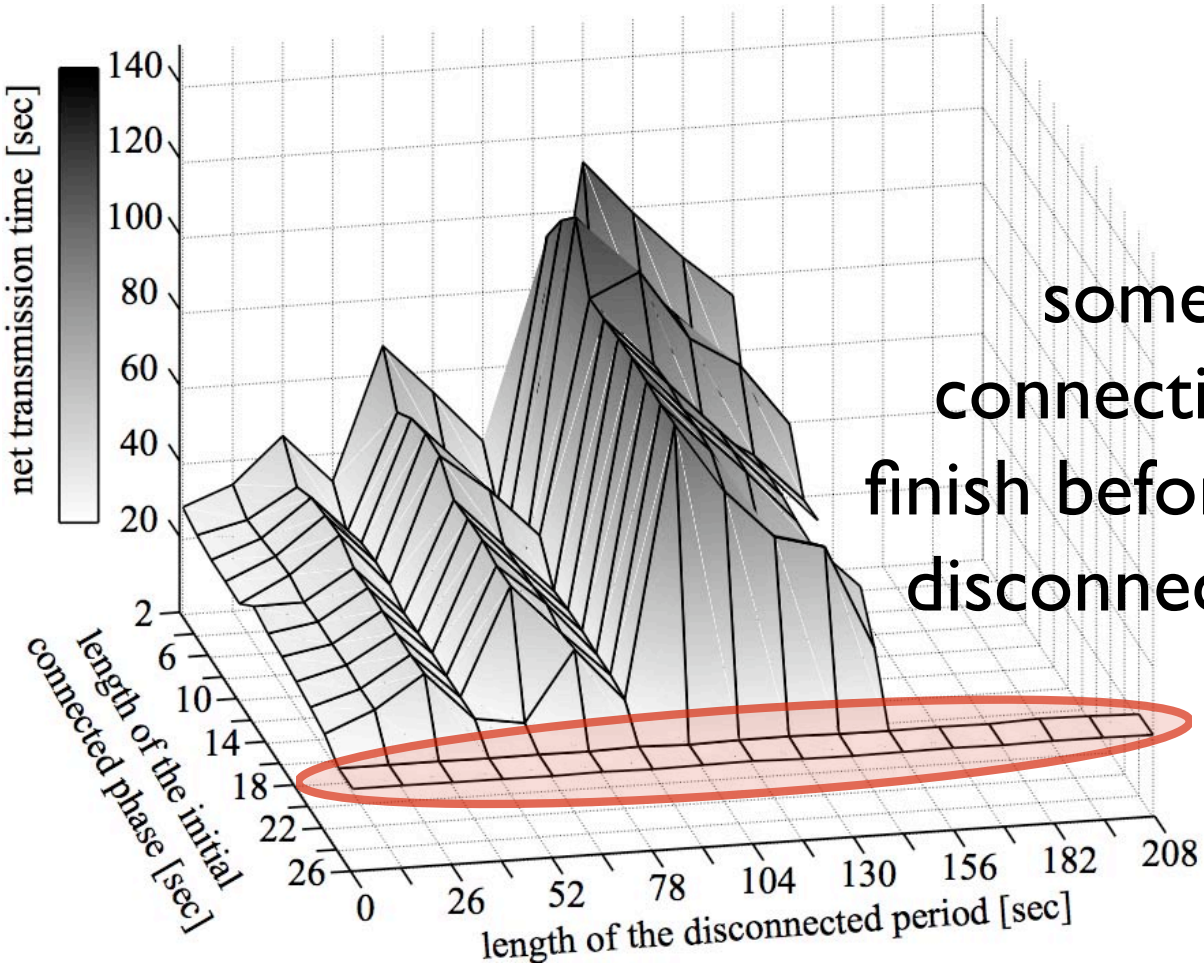
- parameters
  - length of initial connected period: 2-26 sec
  - length of disconnected period: 0-208 sec
- metric: net transmission time
  - factor out disconnected periods
  - compare efficiency during connected periods

Baseline: Vanilla HIP



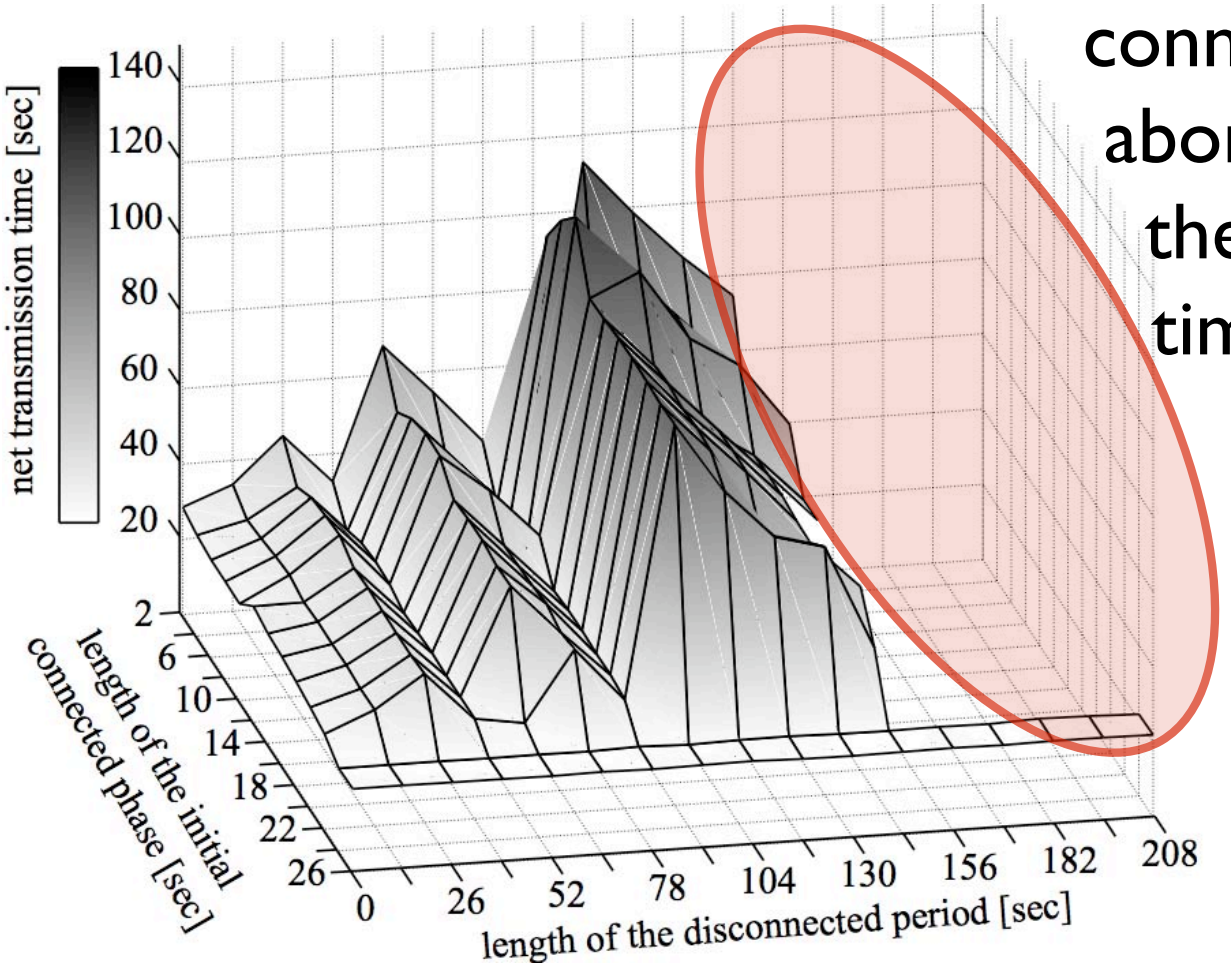
results independent of length of initial connected period

Baseline: Vanilla HIP



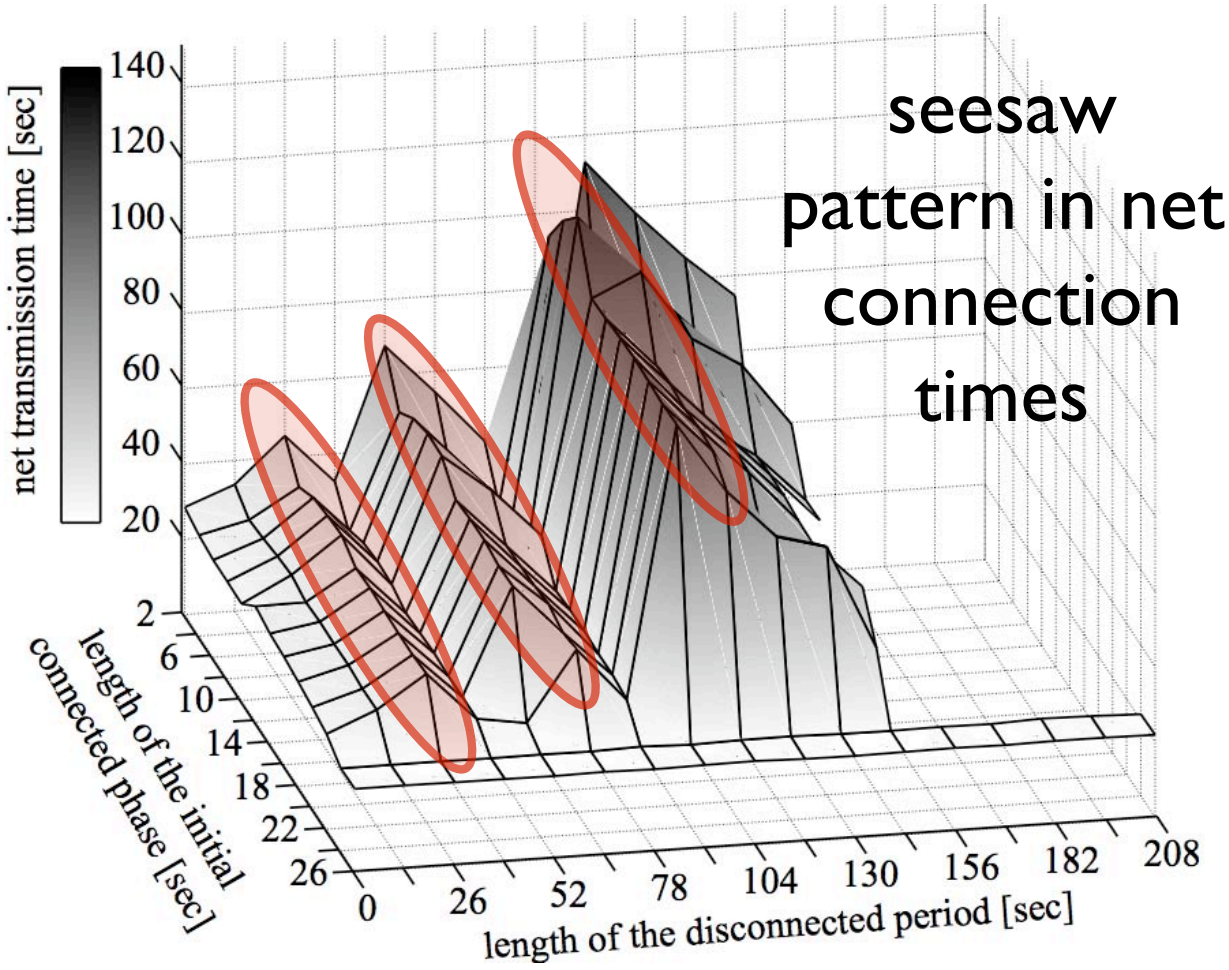
some connections finish before the disconnection

Baseline: Vanilla HIP



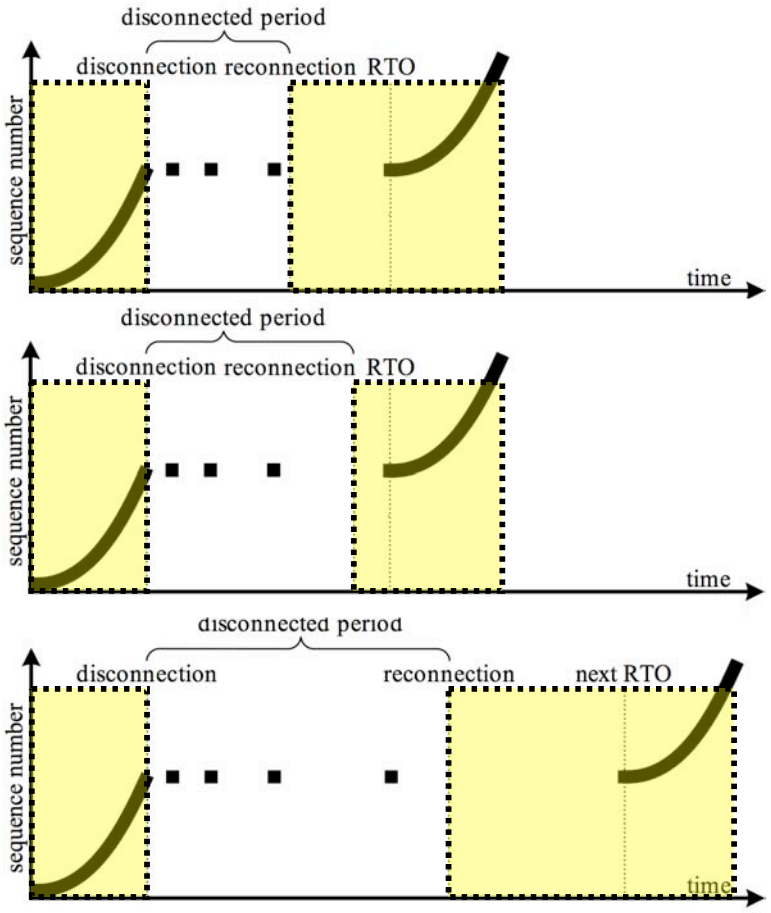
connections abort after the user timeout

Baseline: Vanilla HIP

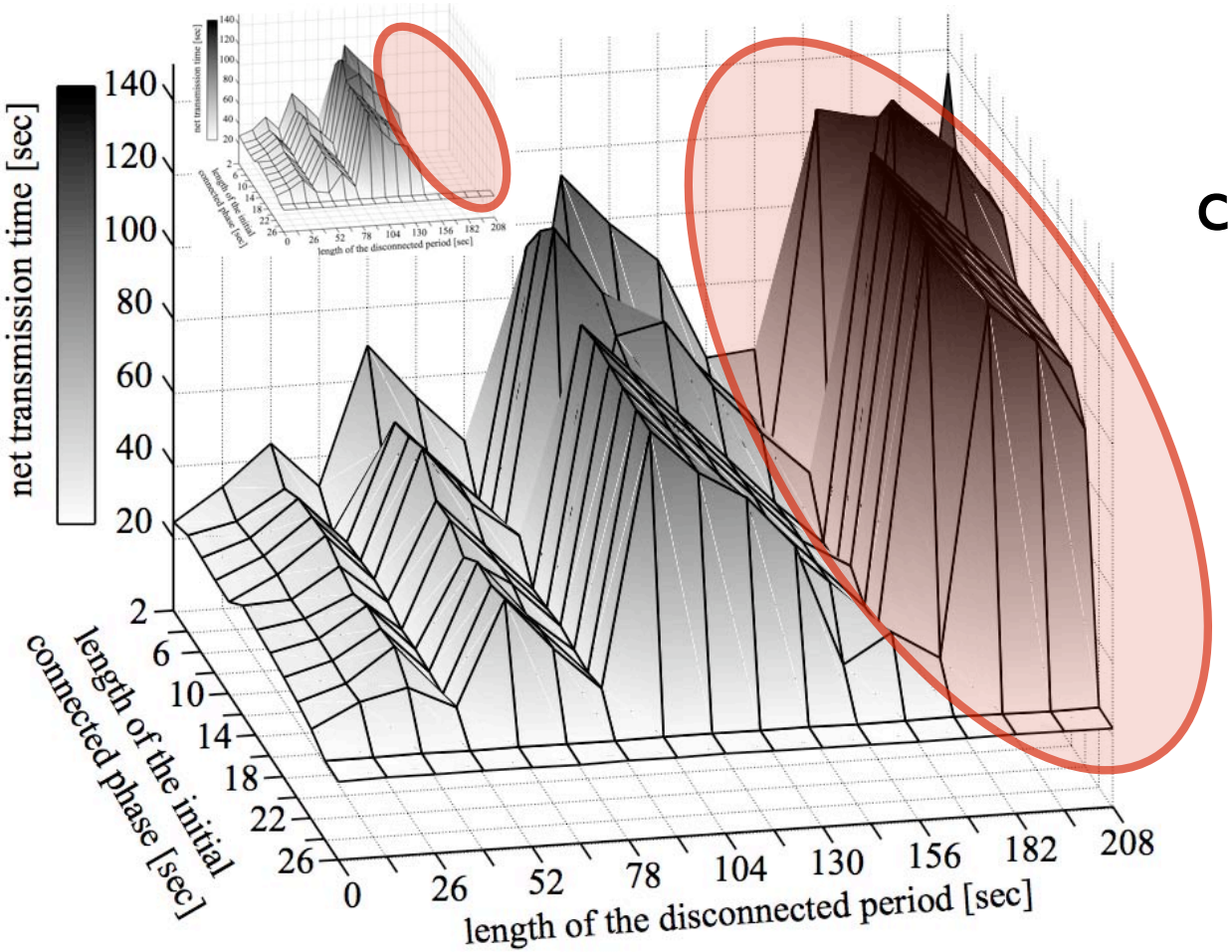


Seesaw Effect

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



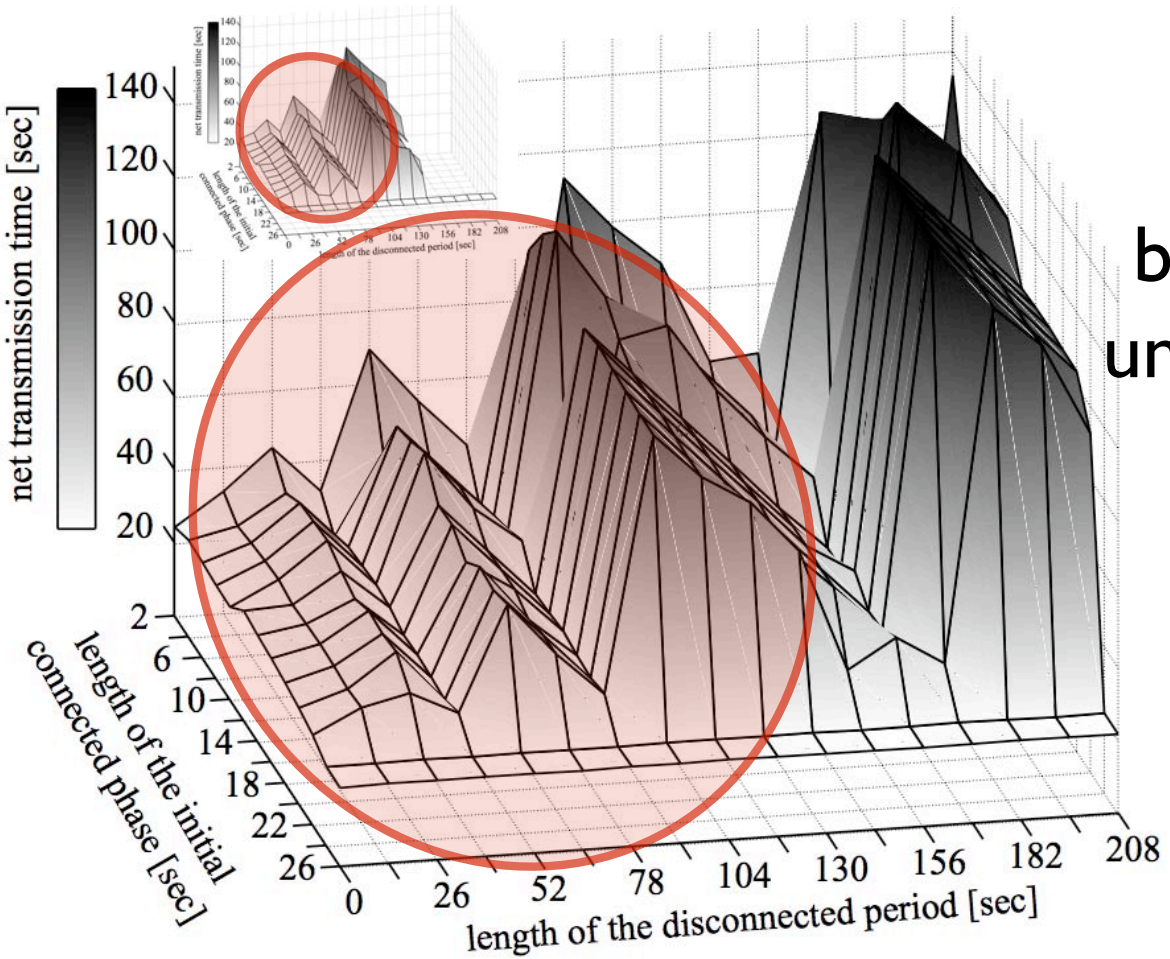
HIP + TCP/UTO



no connection aborts

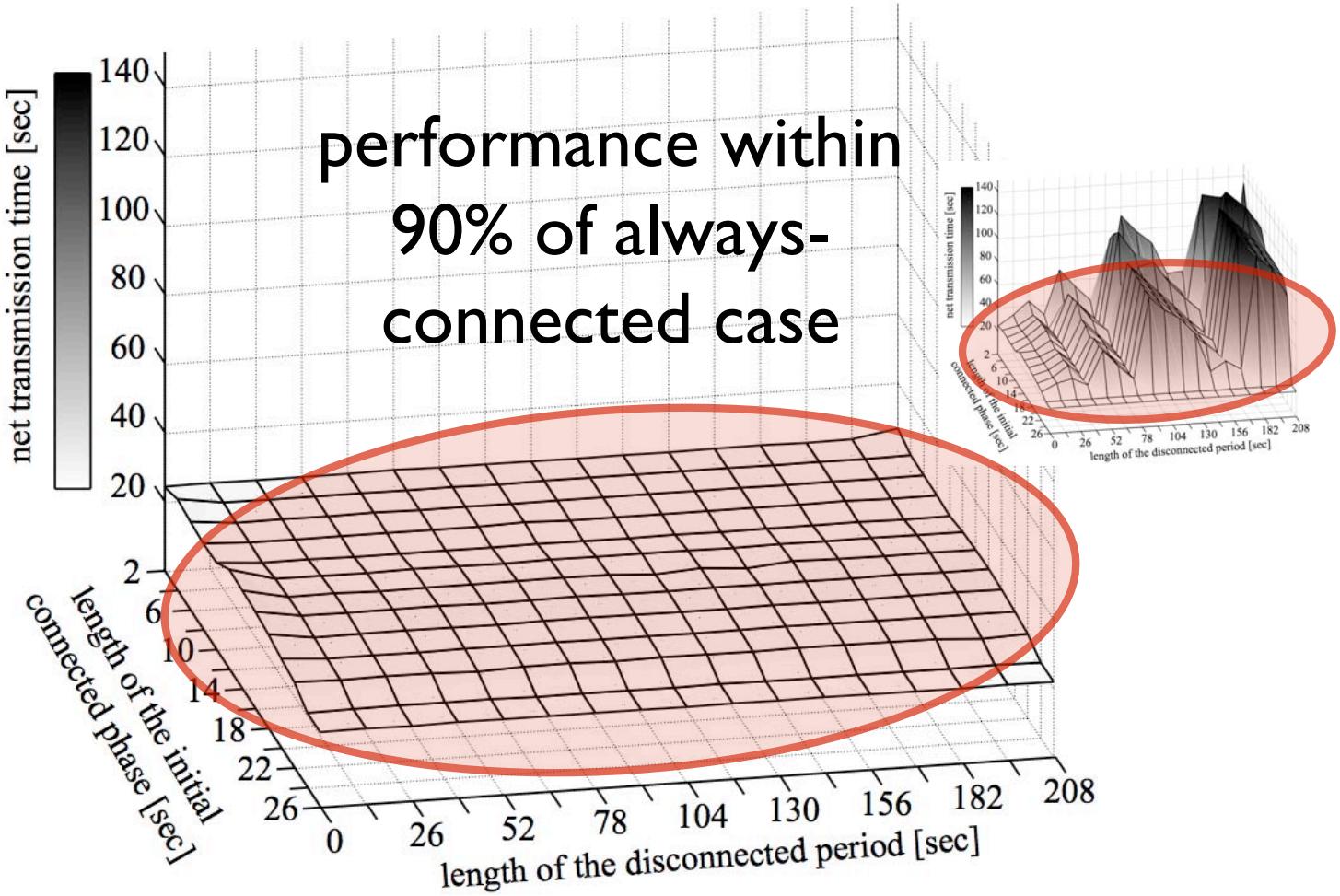
(but long transmit times)

HIP + TCP/UTO

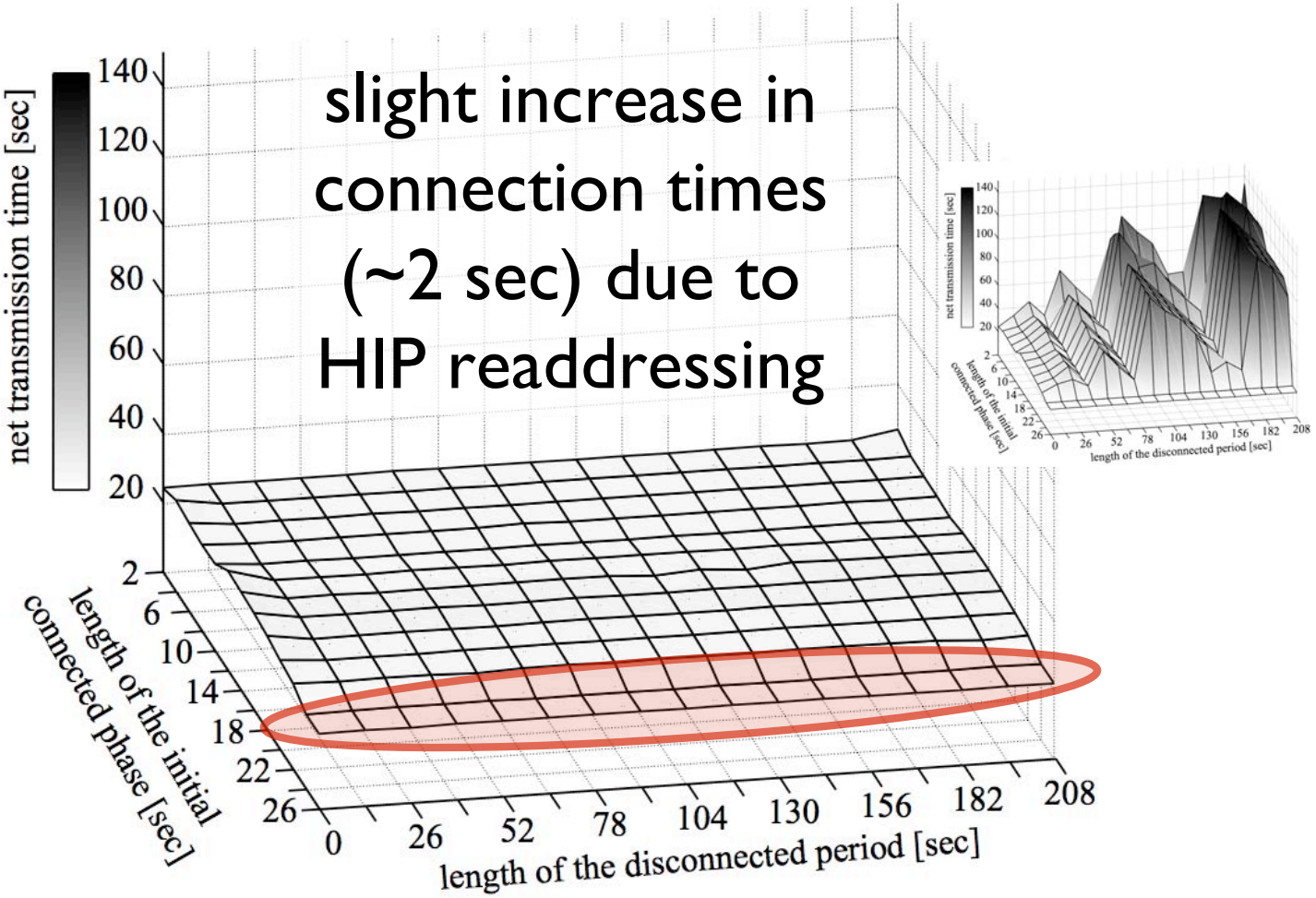


other behavior unchanged

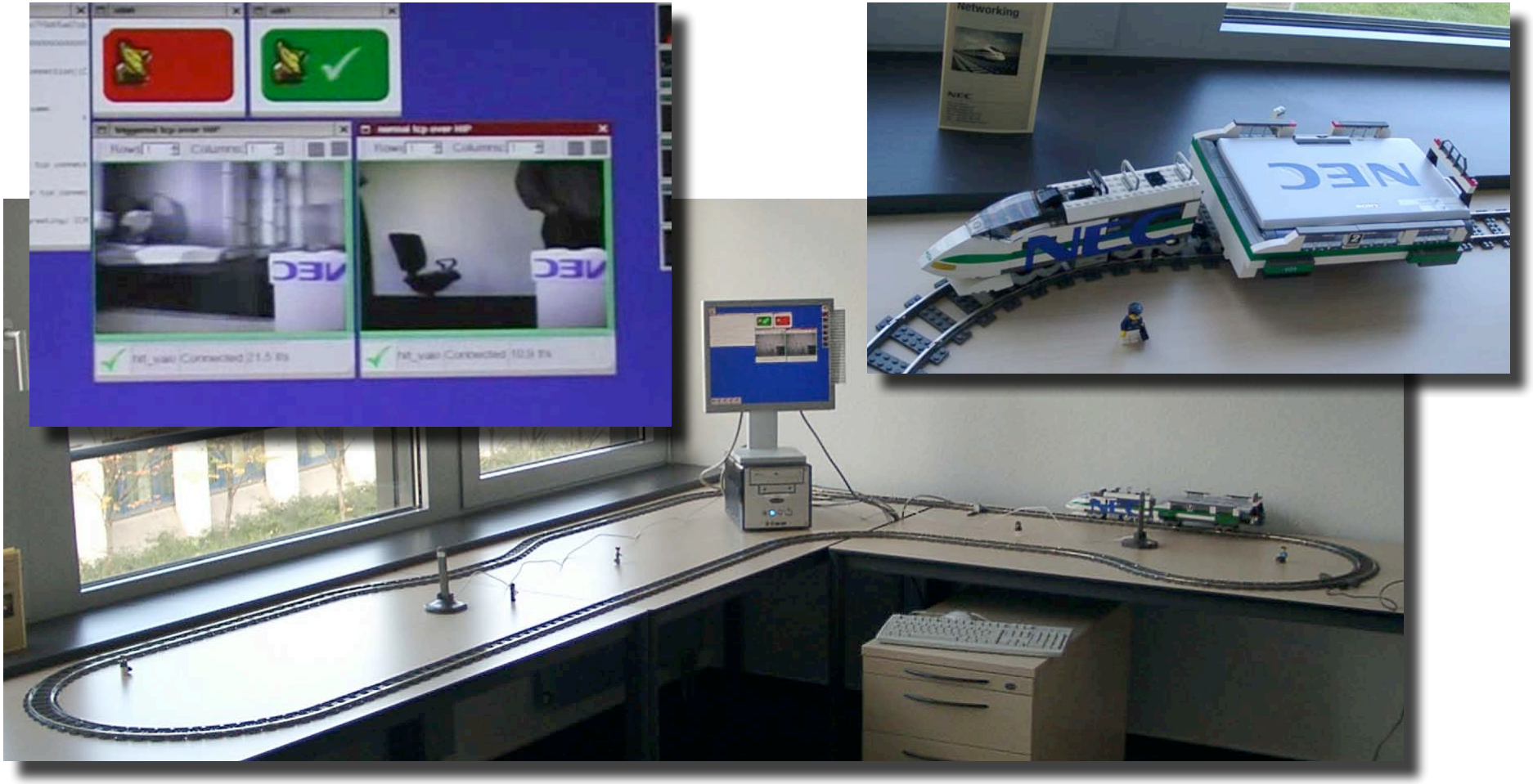
Response Mechanism



Response Mechanism



Demonstrator



Summary

- generic, technology-independent extensions to the virtual machines implemented by layers in the network stack
  - goal: improve operation and performance
- proof-of-feasibility
  - TCP enhancements for operation under intermittent connectivity
  - 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-02, Work in Progress, October 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-07, Work in Progress, February 2006.
- TCP Response to Lower-Layer Connectivity-Change Indications. Simon Schütz, Lars Eggert, Wesley Eddy, Yogesh Swami and Khiem Le. Internet Draft draft-schuetz-tcpm-tcp-rlci-00, Work in Progress.