What if we’d designed measurement as a first-order service?

Mirja Kühlewind and Brian Trammell, ETH Zürich
RIPE 72 - Copenhagen
May 24, 2016
Overview

- Network measurement is hard.
  - Which tool? What to measure? How often?
- Getting it right is even harder.
  - „Wer misst, misst Mist“ * \text{misst} = \text{measure} \& \text{Mist} = \text{bullshit}
- Why is it so hard?
  - “Big five” metrics (loss, latency, jitter, rate, reordering)
- How hard can it be?
  - Path layer providing explicit in-band measurement!
Example: latency/RTT

- Ping?
  - IMCP often blocked
  - Differential Treatment possible
- TCP TSOPT timestamps for latency/jitter
  - Only works with TCP, enabled on about 30% of hosts
  - No application hooks for *explicit* enablement
  - Need heuristics to estimate sender clock rate
Example: Loss/reordering

- TCP throughput testing… is hard to get right [1]
- High network load and unwanted interference
- Ping Mesh?
  - Overhead is not applicable for Internet measurement
  - Do we really measure what we want to measure?
- TCP seq/ack number analysis for loss/reorder?
  - Always exposed, and roundly abused in the Internet
  - Only works with TCP
Everything after ping is a hack

- And even ping doesn’t work that well:
  - ICMP blocked, different codepaths, ECMP routing.
- Traceroute: overload ICMP Time Exceeded messages to infer Layer 3 topology
  - Same problems as ping, but ECMP is worse.

- Passive measurement, e.g. Netflow/IPFIX:
  - Passive RTT measurement [2] broken by ACK optimizations [3], etc.
  - Inflexible, low-rate sampling, even though we know better [4].
What do we really need?

• “Big five” metrics: loss, latency, jitter, rate, reordering
  • as socket properties, with API for access
  • exposed to the network, explicitly for measurability

• Transport-independent header fields explicitly defined for measurability
  • Constant-rate timestamps for latency/jitter
  • Exposure of loss/reordering
  • Detection of header manipulation (required for dynamic transport selection)

• Explicit endpoint control over measurement exposure
• Exposure in header allows passive as well as endpoint measurement
Sounds great. Let’s do it!

Now we just have to find the bits…

- IPv6 Destination Options [5]?
  - not very deployable, may be nearing deprecation, v6 only.
- IPv4 options?
  - even less deployable, v4 only.
- in the TCP header?
  - TCP only; options hard to deploy
  - HICCUPS [6] reclaimed a few bits from the header itself
A Measurement Layer

... for explicit exposure of information as part of normal protocol exchanges!

You don’t have to instrument every packet, every endpoint, or every router to get much better information than we have today.
Adding new layers to the stack for fun and profit

Our “measurement layer” is a special case of a more general problem [7]:
• Where do all of the complex, stateful, not necessarily end-to-end functions we’ve built go?

Solution: “Path layer”
• Encryption of transport layer and above to enforce end-to-end-ness
• Explicit exposure from endpoints to the path of appropriate information
Path layer requirements

- Packets grouping for property binding, on-path state management
- Efficient per-packet signaling
- Integrity protection for exposed headers, allowing modification with endpoint permission
- Protection against trivial abuse of UDP
- Work in progress: draft-trammell-spud-req [8], spud@ietf.org
Will it deploy?

- You can’t add a new layer that today’s routers won’t route.
  - NAT: hard* to deploy protocols other than TCP or UDP

Conclusion: “path layer” headers as shim over UDP

- Initial findings: 3-6% of Internet hosts may have broken or no UDP connectivity, so we’ll need a backup.
- See presentation by Brian Trammell in MAT WG
Concussion

- Yes, measurement is hard.
- Let’s make it better!

**Missing:**
Path layer for explicit exposure of traffic and measurement information
Path Transparency Observatory

- Observatory (public release end 2016) to derive common *observations* about *conditions* on a given *path* at a given *time*
- Active measurements, made by the project
- External measurements (e.g. traceroutes, BGP, traces)
- Combining disparate measurements leads to better insight
- How likely is it that a certain path impairment impacts my traffic?

Follow [http://mami-project.eu](http://mami-project.eu) for updates on data model & availability!
References

[7] draft-kuehlewind-spud-use-cases (IETF individual Internet-draft)
[8] draft-trammell-spud-req (IETF individual Internet-draft)
The MAMI Project
Measurement and Architecture for a Middleboxed Internet

- Strong interaction with relevant standards organizations for impact on deployment
- FIRE testbed (MONROE) support for measurement as well as experimentation, especially on mobile broadband access networks
- Learn more at http://mami-project.eu/
How close are we to the goal?

Modern networking stacks are heavily instrumented
- netstat -s -p tcp on OSX yields 82 event counters.

Application instrumentation also includes collection
- e.g. telemetry.mozilla.org

Phase 1: generalizing and standardizing access to data we already have.
- e.g. mPlane [4]
Why a new shim layer?

- Transport layer: end-to-end sockets
  - flow information
- Internet layer: hop-by-hop handling at the
  - per-packet information
  - stateless and simple processing in the middle
  - stateful and 'smart' processing at the edge

**Path layer** for explicit cooperation with middleboxes instead of implicit assumptions
Implementing an Explicit Path Interface

- Application can directly indicate requirements to path layer
- Transport can use the path layer to expose parts of its functionality/intentions to the network
- *Middlebox Cooperation protocol* (MCP) signals these information appropriately to on-path middleboxes

⇒ Minimize the information exposed!

![Diagram showing the relationship between applications, protocols, and middleboxes.]
How to implement a new path layer?

- Transport-layer **encapsulation over UDP**
  - Need ports for NAT
  - Impossible to deploy with new protocol number across the Internet
  - Userspace (and kernelspace) implementation possible
- **Magic number** for easy recognition, protection against reflection
- **Flags** for “SYN/ACK” condition for state decision delegation to endpoint
  - All traffic bidirectional
  - Data in first packet possible
- Signals fit in a single packet (**no segmentation or reliability**)
- **Checksum** for error detection, cryptographic integrity checks available
Why should I trust what you say about your flows?

- **Default**: *trust but verify*
  - declarative signaling: no negotiation, no guarantees
  - the best way to prevent cheating is to make it useless to do so
  - minimize the information exposed!

- Leverage existing trust relationships for higher-assurance declarations
  - e.g. your enterprise firewall, access network middleboxes, etc.
A Measurement Layer

- Insight: shifting the burden to analysis-time reduces the runtime burden.
- Cumulative nonce \((n_{tx} + \sum n_{rx})\) added to each / sample of packets [8] allows loss rate estimation.
- Timestamp echo \((t_{tx}, t_{rx}, t_{\Delta rx})\) with constant-rate clock [7] and remote delta allows latency and jitter estimation.
- Protected header hash echo \((h_{tx}, h_{rx})\) allows detection of header manipulation [6].
  - Shared-secret protected hashes allow secure detection by endpoints
  - Unprotected hashes detect only accidents
- Insight: Each of these can work at low sampling rates for large flows.
  - How much smarter can we be for less than one bit per packet?