Linux 2.6.20: Current Status of DCCP

Gerrit Renker – Ian McDonald

with code from
Arnaldo De Melo Andrea Bittau
Outline

1. Status so far:
   - RFC Compliance / Gaps
   - Recent updates

2. Experiences (additional information)
   - Performance tests
   - Scheduling limitations
   - Idle Periods
   - Accumulation of send credits

3. Next Steps & Conclusions
Scope of Work

• Our work so far has focused on matching
  – RFC compliance to specifications
  – TFRC performance to theory
  – behaviour to user expectation

Areas of the code

• Arnaldo's DCCP framework: very mature and high quality – few changes required
• CCID 2 (Andrea) – seems to work / not touched
• CCID 3 – code & specification needs work
RFC Compliance

- Original merged DCCP code based on revision 00 of *DCCP Internet Drafts*!
- Combines and integrates latest updates from
  - RFC 3448
  - rfc3448bis
  - RFC 4340/2 + errata
- Numerous *bug fixes* (total of > 100 patches)
Additions to match RFC

- **Service Codes** and **Partial Checksums**
- **Larger initial windows** *(RFC 4342, 5.)*
- **Idle and application-limited periods** *(RFC 4342)*
  - rfc3448bis provides more detailed information on the `how' and is used as basis of implementation
- **Use RTT estimate from Request exchange**
  - as suggested in erratum to RFC 4342
  - again detailed documentation is missing, so rfc3448bis is used as basis of implementation
RFC Compliance Gaps

- **ECN** support (globally)
- **CCID3**
  - Loss Intervals Option (RFC 4342)
  - History Discounting (RFC 3448 optional)
  - Preventing Oscillations (RFC 3448 optional)
- **Need to complete gap analysis** with RFC
  - to show what is still missing
  - or wrong.
Next Steps

• documentation & extension for socket API
• e.g. changing CCIDs via socket options
• ...

Availability

• merging of further fixes planned for March
• about 75 patches in the pipeline
• on list/website, but won't be in 2.6.21
Experiences
(Additional Information)
Built-in Kernel Instrumentation

- **shown**: plot from a kernel run using `dccp_probe`
- for **performance analysis** and **fine-tuning**
Limits of Scheduling Granularity

- **shown**: throughput = f(fixed \( t_{ipi} \))
- no control below \( t_{ipi} = 1000\text{ms} = t_{gran} \).
Uncontrollable Speeds in CCID 3

- admits of GB speed, but no (congestion) control
- limit of controllable speed given by $t_{gran}$
- live with this limitation or use (real-time) fix?
Idle Periods

- RFC3448: if $p > 0$ then
  
  $$X = \max(\min(X_{\text{calc}}, 2X_{\text{recv}}), s/t_{\text{mbi}})$$

- If feedback given once per RTT then
  - after 1 RTT of no transmission $X_{\text{recv}}$ close to 0;
  - therefore $X$ becomes close to 0

- RFC3448 says
  - feedback rate is at least once per RTT or if interval is slower then one packet per RTT, feedback every packet
  - So if application idle occurs, basically start again

- TFRC Faster-Restart will help
Open Issues

• Accumulation of Send Credits
  – Basically $t_{nom_{n+1}} = t_{nom_n} + t_{ipi}$
  – If idle or not sending slower than allowed
    • then $t_{nom}$ will be way behind current time
    • which allows unlimited sending for a period
  – Proposals discussed on list ==> rfc3448bis

• Window Counter RTT Sampling (RFC 4342)
  – RTT needed for computing first loss interval / $X_{recv}$
  – receive times differ from send times (high variance)
  – time (ACK) compression (Zhang '91 / Mogul '92) ?
Conclusions

• Many bug fixes so far
  – not all committed to mainline yet
  – latest patch sets kept online
  – see mailing list (dccp@vger.kernel.org)

• User / application experience missing
  – but interest is perceivable (growing?)
  – paraslash audio streamer runs on dccp

• In good shape and getting better