System Level Performance Management with PCP

Nathan Scott
<nathans@debian.org>
Overview

• What is PCP?
  • Open source toolkit for system level performance analysis
  • Live and historical
  • Extensible (monitors, collectors)
  • Distributed
Architecture

pmlogger
kmchart
pmie
Data Model

- Metrics come from one **source** (host / archive)
- Source can be queried at any interval by any monitor tool
- Hierarchical metric names
  - e.g. disk.dev.read  and  aconex.response_time.avg
- Metrics are singular or set-valued ("instance domain")
- Metadata associated with every metric
  - Data type (int32, uint64, double, ...)
  - Data semantics (units, scale, ...)
  - Instance domain
Performance Timeline

- Where *does* the time go?
- Where’s it going now?
- Where will it go?
Performance Timeline - PCP Toolkit

- Archives

- Live monitoring

- Modelling and statistical prediction
Performance Timeline - PCP Toolkit

- Yesterday, last week, last month, ...
- All starts with pmlogger
  - Arbitrary metrics, intervals
  - One instance produces one PCP archive for one host
  - An archive consists of 3 files
    - Metadata, temporal index, data volume(s)
- pmlogger_daily, pmlogger_check
  - Ensure the data keeps flowing
- pmlogsummary, pmwtf, pmdumptext
- pmlogextract, pmlogreduce
Performance Timeline - PCP Toolkit

- Graphical tools – kmchart, kmtime
  - Strip charts – align data from different subsystems on a single time axis
  - Time controls
    - VCR paradigm
    - Multiple tools can share

- [ Demo ]
  Tempdb growth (1)
Inference Engine - pmie
- Evaluates arithmetic, logical and rule expressions at arbitrary frequencies
- Scan historical data looking for given conditions

Archive mode uses:
- Data reduction
- Alarm verification

[ Demo ]
Tempdb growth (2)
Performance Timeline - PCP Toolkit

- What’s happening right now?
- Hardware, kernel, services, databases, ... application PMDAs.
  - PCP toolkit provides many PMDAs and APIs for customisation
- Important to be able to match user-perceived response time back to system activity

[ Demo ]
Kernel, pmcd, shping PMDAs
pmchart, pmval - monitor tools
Custom Instrumentation (Applications)
Establish performance baselines
  - Setup constant logging
  - Automate detection of known issues (pmie)
Monitor end-user perceived response time
  - Custom collectors
  - Generic collectors – shping, dbping
Understand where that time is spent
  - Distributed systems, distributed queues
  - Monitor for transient / unexpected events
**Tricks and Tips**

- Have a model of performance in your head, and evaluate new information against it

<table>
<thead>
<tr>
<th>Subsystem</th>
<th>Measured Time</th>
<th>Scaled Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU cycle</td>
<td>0.31 nanosec</td>
<td>0.31 sec</td>
</tr>
<tr>
<td>L1 cache</td>
<td>0.31 nanosec</td>
<td>0.31 sec</td>
</tr>
<tr>
<td>L2 cache</td>
<td>1.25 nanosec</td>
<td>1.25 sec</td>
</tr>
<tr>
<td>Memory bus</td>
<td>2 nanosec</td>
<td>2 sec</td>
</tr>
<tr>
<td>DRAM chip</td>
<td>60 nanosec</td>
<td>1 min</td>
</tr>
<tr>
<td>Disk seek</td>
<td>3.5 millisecond</td>
<td>1.35 months</td>
</tr>
<tr>
<td>NFS3 read</td>
<td>32 millisecond</td>
<td>1.01 years</td>
</tr>
<tr>
<td>RDBMS update</td>
<td>0.5 sec</td>
<td>15.85 years</td>
</tr>
<tr>
<td>Tape access</td>
<td>5 sec</td>
<td>1.59 century</td>
</tr>
</tbody>
</table>

*Source:
Analysing Computer System Performance with Perl PDQ, NJ Gunther (2005)*

• units(1)
Tricks and Tips

- Use the “Scientific Method”
  - Postulate, test hypotheses
  - Record results, iterate

- Find good (user) response time metrics
  - Drive analysis based on issues they detect
  - CPU and disk utilisation (time based metrics)
Tricks and Tips

- Regularly apply “Little’s Law” to all data... generic (queueing theory) form:

\[ Q = \lambda R \]

- Length = Arrival Rate x Response Time
- e.g. 10 MB = 2 MB/sec x 5 sec
- Utilisation = Arrival Rate x Service Time
- e.g. 20% = 0.2 = 100 msec/sec x 2 sec
Present and future

- Recent past
  - Development moved to git
  - kmchart developed independently to PCP
  - Included in Debian and Ubuntu (SuSE for ages)

- Moving toward PCP 3.0
  - Native Windows version, Perl APIs, new PMDAs
  - New archive temporal reduction tool

- Longer term
  - SGI releasing 3D visualisation code
  - Many more kmchart features planned
  - Capacity planning ... PCP meets R? PDQ?
Finally...

- Major corporate sponsors of PCP development

- Thanks!

  - Docs, tutorials, git repos, mailing list, IRC channel
  - Binaries - Mac, Windows, RPMs
Questions...?