Dynamic CPU Experience

Early Impressions of Dynamic CPU

Miles Nosler

VISA

Agenda

1. APAR functionality
2. Sponsor User / Beta (Stress Test 2017)
3. Phases of Implementation & Findings
4. Next Steps / Review
5. Q&A

APAR Review

Review basic functionality
Visa impacts
Sponsor User / Beta

APAR Functionality Review – PJ44591

Provides 3 parts of support

Dynamic CPU Capacity

Handle a sustained increase in workload without requiring an outage.

Expand/Collapse I-streams

- Reserve "extra" I-streams
- Step into extra capacity as needed in real-time.

Support HiperDispatch

Optimize CPU workload in shared LPAR environment

Minimize the # of I-streams

- Shared LPAR only
- Automatically Expand/Collapse when I-streams are shared

Low Priority Utilities

Selectively designate utilities as low priority and run them during peak traffic periods.

LOWPRIORITY attribute

- EASETC macro
- -LP prefix on Z-entries

https://www.ibm.com/developerworks/community/blogs/zTPF/entry/PJ44591_Dynamic_CPU_capacity

VISA

Dynamic CPU Capacity

The business case for expanding I-stream capacity in real time

Greater Flexibility in System Configuration

- Step into additional capacity on a day to day basis
- Seasonal peaks – sized all year or have an extra CPU for a day or few days?
- Failover capacity – bring up dynamic CPUs as backup capacity
- Add capacity in real-time with no outage.

Demand Forecasting

- Beyond the baseline transaction profile
- New projects/workload – add CPUs if market demand requires it

Sponsor User / Beta Testing

Findings during sponsor user testing

Tested at annual stress test - IBM Herndon

- Primarily tested on z13 with some testing on z14.

Testing focused almost entirely on expand/collapse of I-streams

- Code base changes – ISTUSEIS & ISTACTIS usage.
- Varied loads (70-80%+), varied number of I-stream expand/collapses
- Ex: 16 active I-streams and 12 in-use – ZMISS SET CAP-16 to add 4 I-streams

VCT List changes

- VCT count increased to 500 (from 50)
- Found CTL-10 in application with high VFA usage
- Change: VCT now set to 1 if running for 100ms. Next VFA access gives up control.

Phases of Implementation

Installing the APAR
Enabling Dynamic CPU
Low Priority Utilities

Implementation – Installing the APAR

Requirements for the APAR installation itself

Build the APAR and test everything!

Regression Test

- Long running utilities
- Complex applications
- Anything with special scheduling schemes
- Known high VFA users

Found 1 application error with VCT change.

Be aware of:

ECBs activated by INETD

- Daemon model is now switchable
- Update programs if they have I-stream dependencies

VCT List changes

- Potential impacts to scheduling.
- Found a utility did not have a proper resource check.

VISA

Installing the APAR – other notes

Utilization Config Update

/etc/ibm_utl_cfg.csv version 2 for Dynamic CPU

- Not required for APAR install by itself.
- Used version 1 for our install.

LODIC Utilization Class

New LODIC required for low-priority ECBs

- Not required for APAR install

New API:

- EASETC and tpf_easetc – SWITCHABLE, LOWPRIORITY attribute.
- EISAC updated to interact with SWITCHABLE
- Begin development with these as the APAR is brought into the codebase.

Installing the APAR – results

Installed successfully in March

- Included correcting APARs (PJ45231 etc.)

Some 31-bit memory freed up (control areas)

- Dispatch control records moved > 4G

CTL-1 catastrophic during LPAR changes on the same CEC

- Calculations in weighting shared CPs, divide by zero
- PJ45309 opened to address

The Dynamic Part – Changes Required!

Requirements for enabling the Dynamic CPU Capacity

Expanding/Collapsing I-streams

Problems:

- Logic Errors if based on highest I-stream
- I-stream unique data
- Outdated scheduling schemes
- Unauthorized ECB = collapsed I-stream

Remove hardcoded I-stream affinity

What to Look for:

- Logic based on ISTUSEIS & ISTACTIS
- SWISC targeting I-streams

Big impact, small changes

VISA

The Dynamic Part – Changes Required!

Examples

How to Remediate

I-stream targeting:

- Target the main I-stream

Code Example

Before (N-1):

```
LH R5, ISTUSEIS

BCTR R5,0

SWISC PROGRAM=QZZ1,IS=R5,TYPE=ENTER  Target highest I-stream
```

After:

```
SWISC QZZ1,IS=MAIN,TYPE=IMMEDIATE Target main I-stream
```

The Dynamic Part – Changes Required!

Examples

How to Remediate

Load Balancing:

- Use SWISC BALANCE (IS = 0)
- SWITCHABLE

Code Example

Before:

```
// spawn a number of ECBs
for (int i = 0; i < numECBs; i++)
{
   <…>
   create_parameters.program = "QZZ1";
   create_parameters.istream = (i % numIstreams) + 1;

   <…>
   swisc_create(&create_parameters); // create ECB
}
```

After:

```
// spawn a number of ECBs
tpf_easetc(TPF_EASETC_SWITCHABLE, TPF_EASETC_SET_ON+TPF_EASETC_INHERIT_YES);

for (int i = 0; i < numECBs; i++)
{
   <…>
   create_parameters.program = "QZZ1";
   create_parameters.istream = SWISC_IS_BALANCE;
   <…>
   swisc_create(&create_parameters); // create ECB
}
```

Low Priority Utilities

Priority class for low priority workloads

Run system “hotter” and these LOWPRIORITY ECBs will get out of the way if they need to.

Determine which long running utilities are candidates for this and implement EASETC macro/API

When these are updated they can be safely run during peak load if necessary.

Or with –LP/ZXXXX without code change.

Requires ZFMSG update – ZFMSG CHANGE ZXXXX LPA

Next Steps

Continuing Testing of Dynamic CPU capacity

- Testing application changes
- Additional Regression Testing
- Native hardware testing
- Stress Test 2018

Low Priority Utilities

- Identifying as needed throughout process
- -LP attribute on functional messages.

Review

APAR install

- Minimal change if any needed
- Test as much as possible

Application code base needs review for Dynamic CPU

- I-stream affinity
- SWITCHABLE for load balancing

Low Priority Utilities – roll out as needed

- LOWPRIORITY attribute via EASETC
- OR –LP prefix

Q&A