HP Asset Manager

Push/Generic Adapter in Parallel Mode on MS SQL Server

Legal Notices

Warranty

The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

Restricted Rights Legend

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright Notice

Trademark Notices

Adobe™ is a trademark of Adobe Systems Incorporated.

Microsoft® and Windows® are U.S. registered trademarks of Microsoft Corporation.

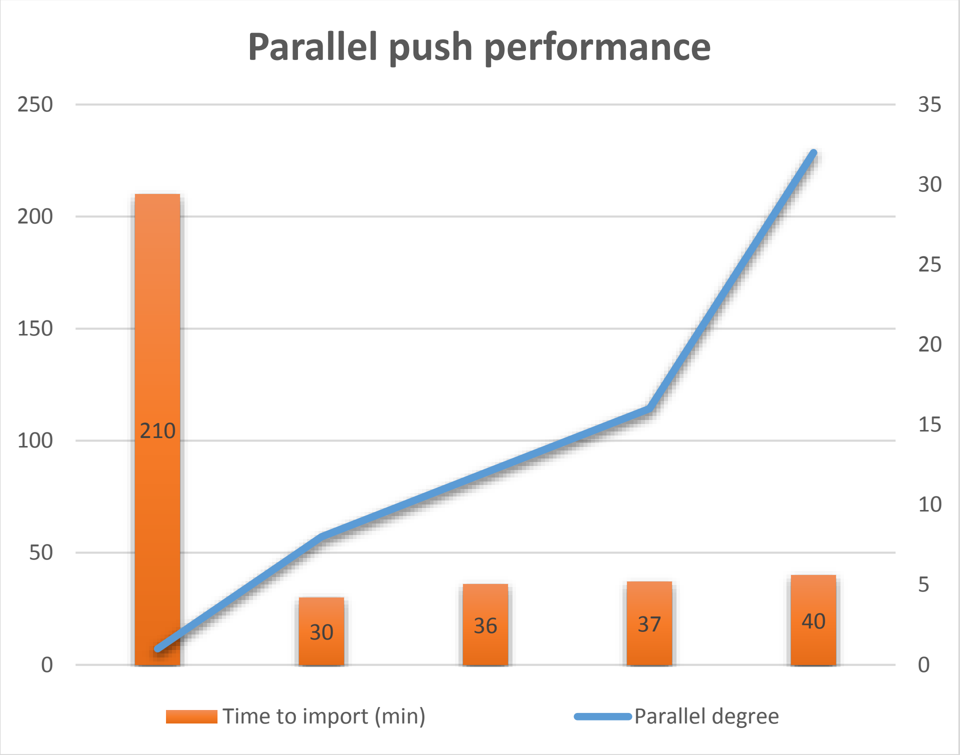
This guide describes monitoring parameters for an HP software Asset Manager instance with Tomcat 7.

Contents

Summary

This document describes the recommended configuration for the Asset Manager (AM) Push/Generic Adapter that runs in parallel mode.

Parallel data push can significantly improve performance results. In this test, it increases the performance by 700% on an Asset Manager/Microsoft SQL Server database.



Acronyms:

- AM (HP Asset Manager),
- UCMDB (HP Universal Configuration Management Database)

This test was done after several changes were implemented in Asset Manager 9.41:

Defects	Description
QCCR1E113177	When activating the parallel push and modifying the isolation level for Push Adapter, push job will fail
QCCR1E118467	Error (12,001): The maximum number of active connections (64) has been reached
QCCR1E118984	AM Push Adapter tries to perform an insert instead of update
QCCR1E119786	Some contents missing in ADB log in multi-thread environment

QCCR1E120402	Stored Procedure UP_GETID for MS SQL Server may cause dead lock on concurrent environment
QCCR1E121193	When calling AmGetConnection through AM Push adapter parallel mode, it gives the error: "Unable to complete operation in current state"
QCCR1E121194	When running AM Push adapter in parallel mode, it gives the error occasionally: "Failed to generate random number from SSL library"
QCCR1H98679	NullPointerException occurs when run AM Push Adapter in parallel mode.
QCCR1E122535	Running parallel push in the AM Push Adapter fails on quantity of the batch check

To deploy AM Push/Generic Adapter, you need to install UCMDB 10.20 CP14 Update5, CP15 Update2, or a greater version. For Asset Manager, you need to install 9.41.11083, 9.50.11741, or a greater version which has these fixes and is optimized for running in parallel mode.

For more information about how to deploy adapter package and configure the integration point and integration job, see UCMDB Discovery and Integration Content Guide – HP Integrations using the following URL:

https://softwaresupport.hp.com/group/softwaresupport/search-result/-/facetsearch/document/KM01367254

Configuration for Parallel Push

Prepare Asset Manager Database for Parallel Push

For DB2 and Oracle databases, follow the instructions in the *Eliminating locks and deadlocks* chapter of the Asset Manager Tuning Guide. See *https://softwaresupport.hp.com/group/softwaresupport/search-result/-/facetsearch/document/KM00975188*

For SQL Server, follow these steps:

1. Alter the SQL Server schema isolation level:

   Run the following command on the database, replace <AMSchema> with the actual schema name:

   ```
   ALTER DATABASE <AMSchema> SET READ_COMMITTED_SNAPSHOT ON

   ALTER DATABASE <AMSchema> SET ALLOW_SNAPSHOT_ISOLATION ON

   GO
   ```

   Note: If the operation takes too long to complete, you need to disconnect all connections to the database. One possible way is to restart the database service, and then run the command. If you decide to restart SQL Server, and an Integration Point has already been created in the UCMDB, you should restart the UCMDB Probe as well to avoid the dead connections.

2. Alter Asset Manager database options:
   a. Open the Asset Manager Client and connect to the appropriate database schema.
   b. On the **Administration** menu, click **Database options**.
   c. For option “Sql Server specifics’|‘Isolation command before starting a write transaction”, change the current value to set the transaction isolation level snapshot.
   d. The mechanism that generates counter values in Asset Manager needs to be made compatible with parallel push. By default, counters are locked until each transaction is committed. This prevent concurrent transactions using counters from being done in parallel OOB. The Tuning guide lists a way to generate this counter mechanism on a production database that might have been customized. The following example can be taken as a simple template on a demo database on which all counter values will start with value 100,000, for the counters present in the demo database used at test time:

```
go
CREATE TABLE [itam].[cnt_amComputer_Group] ([lValue] [int] IDENTITY (100000,1) NOT NULL ,[luserspid] [int] NOT NULL)  ON [PRIMARY]
CREATE TABLE [itam].[cnt_amComputer_Domain] ([lValue] [int] IDENTITY (100000,1) NOT NULL ,[luserspid] [int] NOT NULL)  ON [PRIMARY]
CREATE TABLE [itam].[cnt_amComputer_Vm] ([lValue] [int] IDENTITY (100000,1) NOT NULL ,[luserspid] [int] NOT NULL)  ON [PRIMARY]
CREATE TABLE [itam].[cnt_amComputer_MD] ([lValue] [int] IDENTITY (100000,1) NOT NULL ,[luserspid] [int] NOT NULL)  ON [PRIMARY]
CREATE TABLE [itam].[cnt_amComputer_Name] ([lValue] [int] IDENTITY (100000,1) NOT NULL ,[luserspid] [int] NOT NULL)  ON [PRIMARY]
CREATE TABLE [itam].[cnt_amSoftInstall_Code] ([lValue] [int] IDENTITY (100000,1) NOT NULL ,[luserspid] [int] NOT NULL)  ON [PRIMARY]
CREATE TABLE [itam].[cnt_amAsset_AssetTag] ([lValue] [int] IDENTITY (100000,1) NOT NULL ,[luserspid] [int] NOT NULL)  ON [PRIMARY]
CREATE TABLE [itam].[cnt_amBrand_BarCode] ([lValue] [int] IDENTITY (100000,1) NOT NULL ,[luserspid] [int] NOT NULL)  ON [PRIMARY]
CREATE TABLE [itam].[cnt_amModel_BarCode] ([lValue] [int] IDENTITY (100000,1) NOT NULL ,[luserspid] [int] NOT NULL)  ON [PRIMARY]
CREATE TABLE [itam].[cnt_amModel_ModelRef] ([lValue] [int] IDENTITY (100000,1) NOT NULL ,[luserspid] [int] NOT NULL)  ON [PRIMARY]
CREATE TABLE [itam].[cnt_amAssignment_Code] ([lValue] [int] IDENTITY (100000,1) NOT NULL ,[luserspid] [int] NOT NULL)  ON [PRIMARY]
CREATE TABLE [itam].[cnt_amMonitor_Serial] ([lValue] [int] IDENTITY (100000,1) NOT NULL ,[luserspid] [int] NOT NULL)  ON [PRIMARY]
CREATE TABLE [itam].[cnt_amEmplDept_BarCode] ([lValue] [int] IDENTITY (100000,1) NOT NULL ,[luserspid] [int] NOT NULL)  ON [PRIMARY]
go

SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [itam].[up_GetCounterVal] @CounterName nvarchar(64), @CounterIncrement int AS
BEGIN
--Frequency Order: tuning DDMI
 IF @CounterName =N'amAssignment_Code'
  BEGIN
   insert into  cnt_amAssignment_Code (luserspid) values(@@SPID)
   goto RETURN_IDENTITY
  END
 IF @CounterName =N'amSoftInstall_Code'
  BEGIN
   insert into  cnt_amSoftInstall_Code (luserspid) values(@@SPID)
   goto RETURN_IDENTITY
  END
 IF @CounterName =N'amAsset_AssetTag'
  BEGIN
   insert into  cnt_amAsset_AssetTag (luserspid) values(@@SPID)
   goto RETURN_IDENTITY
  END
 IF @CounterName =N'amModel_ModelRef'
  BEGIN
   insert into  cnt_amModel_ModelRef (luserspid) values(@@SPID)
   goto RETURN_IDENTITY
  END
  IF @CounterName =N'amModel_BarCode'
    BEGIN
      insert into  cnt_amModel_BarCode (luserspid) values(@@SPID)
      goto RETURN_IDENTITY
    END
  IF @CounterName =N'amComputer_Name'
    BEGIN
      insert into  cnt_amComputer_Name (luserspid) values(@@SPID)
      goto RETURN_IDENTITY
    END
  IF @CounterName =N'amMonitor_Serial'
    BEGIN
      insert into  cnt_amMonitor_Serial (luserspid) values(@@SPID)
      goto RETURN_IDENTITY
    END
  IF @CounterName =N'amBrand_BarCode'
    BEGIN
      insert into  cnt_amBrand_BarCode (luserspid) values(@@SPID)
      goto RETURN_IDENTITY
    END
  IF @CounterName =N'amComputer_Group'
    BEGIN
      insert into  cnt_amComputer_Group (luserspid) values(@@SPID)
      goto RETURN_IDENTITY
    END
  IF @CounterName =N'amComputer_Domain'
    BEGIN
      insert into  cnt_amComputer_Domain (luserspid) values(@@SPID)
      goto RETURN_IDENTITY
    END
  IF @CounterName =N'amComputer_Vm'
    BEGIN
      insert into  cnt_amComputer_Vm (luserspid) values(@@SPID)
      goto RETURN_IDENTITY
    END
  IF @CounterName =N'amComputer_MD'
    BEGIN
      insert into  cnt_amComputer_MD (luserspid) values(@@SPID)
      goto RETURN_IDENTITY
    END
  IF @CounterName =N'amEmplDept_BarCode'
    BEGIN
      insert into  cnt_amEmplDept_BarCode (luserspid) values(@@SPID)
      goto RETURN_IDENTITY
    END
RETURN_IDENTITY:
  SELECT @@IDENTITY
  END
GO

DROP PROCEDURE itam.up_GetId
GO

CREATE PROCEDURE itam.up_GetId
```

```
AS
DECLARE @id INT
DECLARE @count INT

SET @id = 0

DECLARE @result INT;
DECLARE @NewId AS TABLE (Id INT)

BEGIN TRAN tran1
    INSERT INTO LastId (Value, IRemain, IInUse) OUTPUT Inserted.IdSeed INTO @NewId VALUES (@@SPID, 32, 1)
    SELECT @id = Id FROM @NewId
COMMIT TRAN tran1

RETURN @id
GO

DROP procedure itam.up_GetIndepId
GO
CREATE procedure itam.up_GetIndepId as
declare @id int
set @id = 0
DECLARE @result int;
declare @NewId as table(Id int)

begin tran tran1
    EXEC @result = sp_getapplock @Resource = 'up_GetIndepId', @LockMode = 'Exclusive', @LockTimeout = '0';
    if (@result >= 0) begin
        select @id = Min (IdSeed) from IndependentLastId where IRemain > 0 and IInUse = 0
        if (@id > 0) update IndependentLastId set IInUse = 1 where IdSeed = @id
        else begin
            insert into IndependentLastId(Value, IRemain, IInUse) output Inserted.IdSeed INTO @NewId values(@@SPID,32, 1)
            select @id = Id from @NewId
        end
        EXEC @result = sp_releaseapplock @Resource = 'up_GetIndepId'
    end
    else begin
        insert into IndependentLastId(Value, IRemain, IInUse) output Inserted.IdSeed INTO @NewId values(@@SPID,32, 1)
        select @id = Id from @NewId
    end
commit tran tran1
return @id
GO
```

Performance Test Result and Recommended Thread Number

Performance Test Result

This performance test is intended to compare the performance between parallel push and single thread push. It tries to find out the optimal thread number for parallel push.

Test Environment:

Hardware & OS

Usage	Configuration	OS
AM Database	Physical machine, Intel® Xeon® X5660@2.8GHZ*24 / 32GB RAM	Microsoft Windows Server 2008 R2, 64-bit
UCMDB Database	Physical machine, Intel® Xeon® X5660@2.8GHZ*24 / 32GB RAM	Microsoft Windows Server 2008 R2, 64-bit
UCMDB Server & Probe	Virtual machine Hosted on Intel® Xeon® X5670@2.93GHZ*24 / 16GB RAM	Microsoft Windows Server 2008 R2, 64-bit

Software

Usage	Software
AM Database	Asset Manager Version 9.41 on Microsoft SQL Server 2012 Enterprise Edition
UCMDB Database	Oracle 11g
UCMDB Server & Probe	UCMDB 10.20
AM API DLL	9.41.11083
AM Generic Adapter	CP15 Update2

Test Data

Push Job: AM Node Push, Full push (not differential)

Number of Computers Pushed into AM: 11290

Test Result

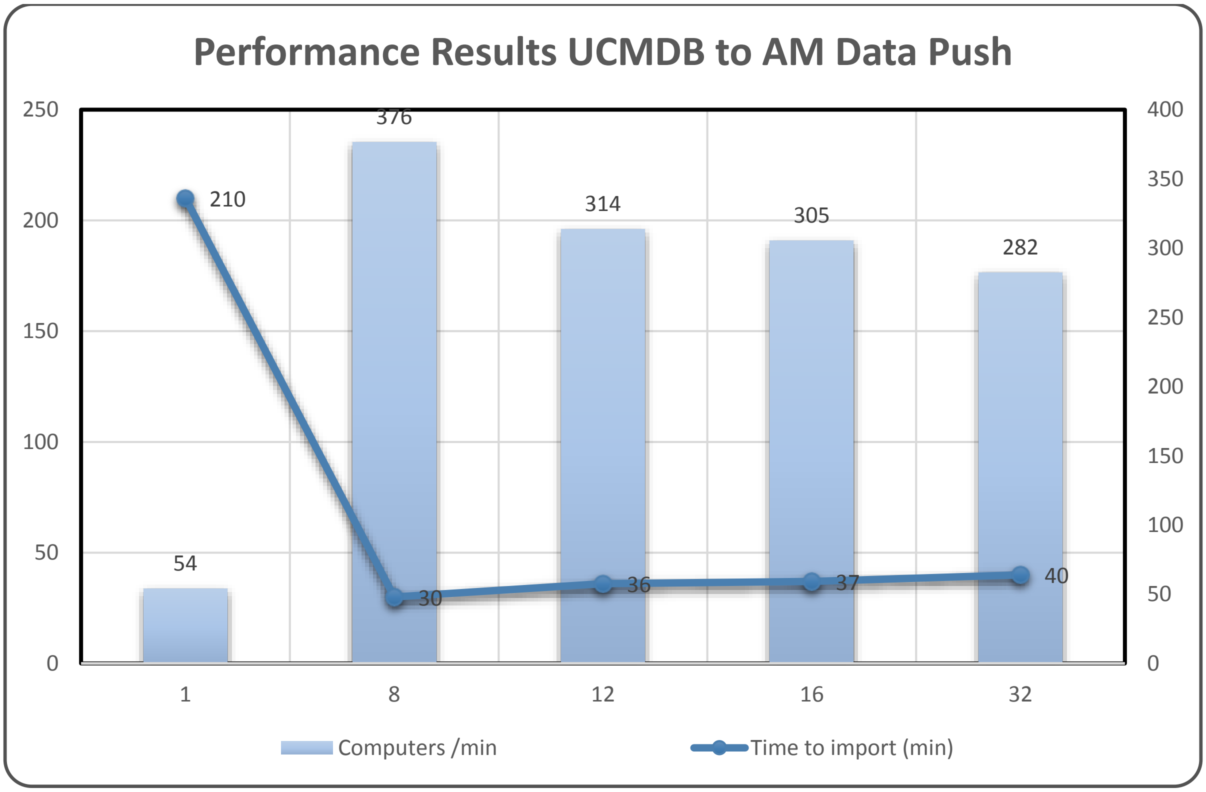
Parallel Mode	Threads	Time Cost
No	1 (single threaded push)	3hrs 30mins
Yes	8	30mins
Yes	12	36mins

Yes	16	37mins
Yes	32	40mins

Recommended Thread Number

The performance test result above indicates that the best results were produced with 8 threads in the parallel push.

Parallelization increases performance in a non-linear way, as per Amdahl's law. In this test, maximum performance is reached with 8 threads: more threads even get worse performance because of the overhead synchronization cost increase between threads, and possibly CPU workload capacity.



This test is in a specific environment, the test result may vary in your environment. In general, we recommend that you use the thread number by default.