RAPID CAPACITY PLANNER FOR VPLEX WITH XML/PERL/SQLITE

Mintulal Debnath
Implementation Specialist, EMC Global Services

Salem Sampath
Advisory Solutions Architect, EMC Global Services

Table of Contents

The need for rapid capacity planner 3
Architecture of the tool 4
VPLEX Configuration file 5
How to export the VPLEX configuration data 5
Configuration details 5
Relational mapping for VPLEX configuration 7
Why a relational schema for VPLEX configuration? 7
Entity Relationship diagram for VPLEX 7
Explanation for some of the important tables 8
Sample Capacity reports 9
Overall Summary report 9
Server Allocation report 9
Front-end allocation 10
Back-end allocation 11
Conclusions and next steps 12
Bibliography 13

The need for rapid capacity planner

The ability to provide high availability storage for "big data" applications makes capacity planning a paramount task. This task is eased by EMC VPLEX®, a scalable, distributed storage federation solution. The storage allocated via VPLEX can be accessed by hundreds of hosts. In many customer installations, multiple business units share VPLEX-based storage, which poses some interesting questions to IT administrators, managers, and CIO's on the chargeback mechanism for VPLEX-based storage. Typical questions that our customers ask include:

1. What is the storage allocated and currently used for an application which has HOST 1, HOST 2, etc.?
2. We need to add 100 TB storage. How do we optimally find which port the storage should be added to?
3. Is the storage allocated optimally across all the front-end ports?
4. What is the capacity for finance application AF for the period between January 01, 2012 and February 28, 2012?

The current method of manually calculating allocated storage can be very time consuming. However, we have devised a methodology by reading the VPLEX (XML) configuration file using open source tools such Perl, XML, and SQLite. It is a simple, easy-to-use tool that can save hours of manual calculation. By using a relational database technology (RDBMS) using SQLite and scripting, standard and ad-hoc reports can be created in minutes.

In this article, we describe the architecture of the tool and explain in detail the technical components such as the input configuration dump file (XML format), the XML/Perl library that was used to parse the XML configuration dump file to load into a set of relational tables. Once the data is loaded into the relational tables, a SQL query is executed against the relational tables and the capacity report is generated in a .CSV or Excel workbook format.

Our solution has been successfully designed and implemented in Fortune 500 companies, saving several hours of work. The level of automation provided has an immense impact on total customer experience (TCE) with low or no learning curve to adopt this solution. The simplicity and the innovative method that we used can lead to wider use of this tool to rapidly generate capacity reports for VPLEX environments.

Architecture of the tool

The tool contains three main components:

1. Configuration dump file (XML) Perl Parser
2. SQLite based relational database
3. Report generator

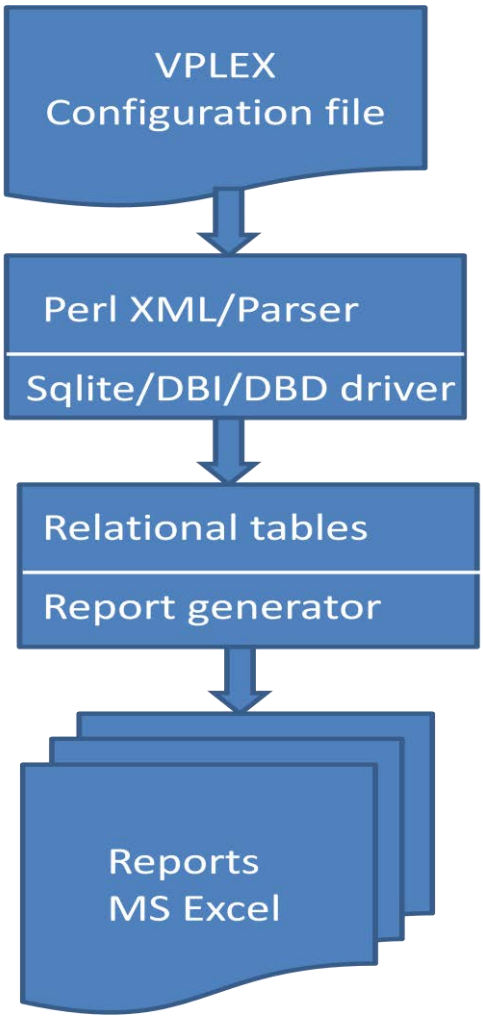

Figure 1

As you can see in Figure 1, the configuration file is a XML file obtained from the VPLEX cluster. The XML file is then parsed using Perl XML parser and the data generated from the XML parser is loaded into a set of relational tables. The next sections describe the configuration dump file and the relational schema used to hold the VPLEX configuration details. Once the configuration data is loaded into a set of relational tables, a set of SQL queries automatically generate capacity reports that can be easily imported to a Microsoft Excel workbook.

VPLEX Configuration file

How to export the VPLEX configuration data

Configuration details about the VPLEX array are stored persistently and can be easily exported to XML format. The configuration file can be exported using the Vplexcli command show below:

```
VPlexcli:/clusters/>configdump -c cluster-1 -f cluster1_config.xml
```

The XML configuration file can be browsed using any XML browser such as xmlnotepad. Below is a sample output:

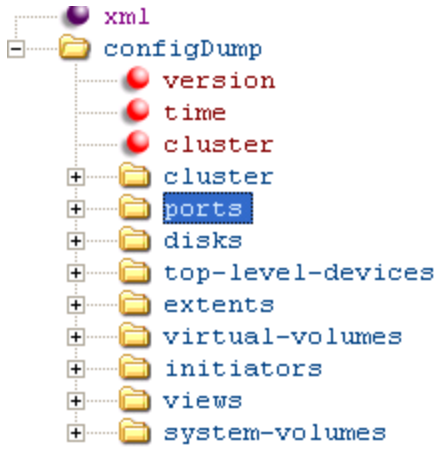

Figure 2

As you can see in Figure 2, the configuration file contains detailed information about various aspects of arrays such as ports, extents, disks, etc. In the next section, we will look at configuration data relevant for our analysis.

Configuration details

The Perl XML parser extracts the following data from the configuration dump file:

1. Cluster
   i. Version
   ii. ID
   iii. Plex director information
      1. Director name
      2. Host name
2. Ports. There are three types of ports in VPLEX:
   i. Front-End port - connects to SAN / Hosts
   ii. Back-End port - connects to EMC and heterogeneous arrays
   iii. The ports have the following attributes
      1. Status
      2. Role - front-end or back-end
      3. Associated director name

        4. World Wide Name
        5. Name
3. Disks
    i. Vpd identifier
    ii. Name of the imported disk
    iii. Size in bytes
    iv. Back-end target LUN
4. VPLEX device information
    i. VPLEX device name based on customer standards
    ii. Whether it is a local or remote device
    iii. Block size in bytes
    iv. Size in bytes
    v. Type (RAID 1, RAID-C, RAID 0)
5. Extents - How back-end LUN is presented to VPLEX
    i. Status: Alive / dead / degraded
    ii. Which front-end disk is using it
    iii. Extent_name
6. Virtual volumes
    i. Volume name
    ii. Device WWN
    iii. Cache mode - Sync or Async
7. Initiators
    i. Port WWN
    ii. Name
8. Views - three types (Storage port, HBA WWN, Virtual volumes)
    i. View name
    ii. Initiators HBA/WWN
    iii. Virtual volume name

The above mentioned configuration details are automatically loaded into a SQLite database schema as detailed in the next section. For example, extents-related data will be loaded into a table that keeps track of extent data and device-related data will be loaded into another table that keeps track of extent allocation for that device.

Relational mapping for VPLEX configuration

Why a relational schema for VPLEX configuration?

This is a valid and legitimate question. Creating a relational model to represent the VPLEX configuration data opens up avenues to query configuration data in an ad-hoc manner described earlier in Section 1.

Entity Relationship diagram for VPLEX

Here is the Entity Relationship (ER) that maps various configurations:

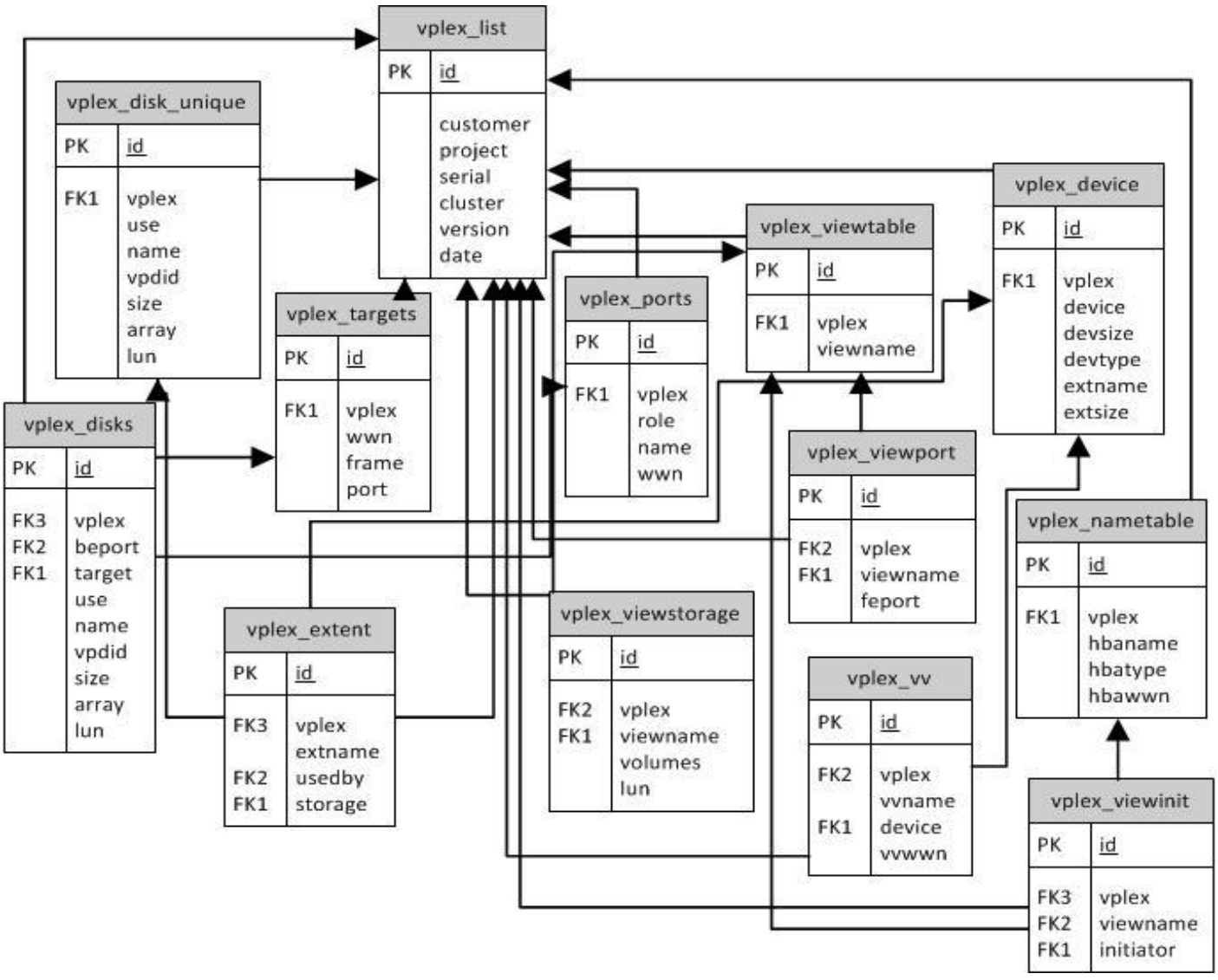

Figure 3

Explanation for some of the important tables

The schema above (Figure 3) lists 13 tables used to map the VPLEX configuration details. The following is the Data Definition Language for some of the important tables:

VPLEX_list

This is the master table. Every table in the schema is linked to id column of this table. A row is inserted nto this table for each report that was generated. Here is the schema for this table:

```
CREATE TABLE vplex_list
(
id        INTEGER PRIMARY KEY AUTOINCREMENT,    -- this is the master table and every table is linked through this column
customer  TEXT,                                 -- customer name
project   TEXT,                                 -- project name
serial    TEXT,                                 -- array serial number
[cluster] TEXT,                                 -- cluster #
version   TEXT,                                 -- vplex version
date      TEXT                                  -- data of run
)
```

VPLEX_device

This table contains information about device name, size, and so forth. Please note that device and extent sizes are indicated in bytes, respectively. Here is the schema for this table:

```
CREATE TABLE vplex_device
(
id      INTEGER PRIMARY KEY,    -- linked to ID in vplex_list table
vplex   INTEGER,
device  TEXT,                   -- device name
devsize REAL,                   -- device size in bytes
devtype TEXT,                   -- device type (RAID 0, RAID 1, RAID-C)
extname TEXT,                   -- extent name
extsize REAL,                   -- extent size in bytes
extvpd  TEXT                    -- extent vpd
)
```

VPLEX_extent

This table contains information about extents that are allocated. Here is the table schema:

```
CREATE TABLE vplex_extent
(
id       INTEGER PRIMARY KEY AUTOINCREMENT,      -- linked to ID in vplex_list table
vplex    INTEGER,
extname  TEXT,                                   -- extent name
usedby   TEXT,
storage  TEXT,
FOREIGN KEY ( vplex ) REFERENCES vplex_list ( id ) ON DELETE CASCADE,
FOREIGN KEY ( usedby ) REFERENCES vplex_device ( id ) ON DELETE CASCADE,
FOREIGN KEY ( storage ) REFERENCES vplex_disk_unique ( id ) ON DELETE CASCADE
)
```

VPLEX_ports

This table contains information about VPLEX ports and its worldwide names. Here is the table schema:

```
CREATE TABLE vplex_ports
(
id    INTEGER PRIMARY KEY AUTOINCREMENT,       -- linked to ID in vplex_list table
vplex INTEGER,
role  TEXT,                                    -- role (front-end, back-end, comm  etc.,)
name  TEXT,                                    -- port name
wwn   TEXT,                                    -- world wide name
FOREIGN KEY ( vplex ) REFERENCES vplex_list ( id ) ON DELETE CASCADE
)
```

Sample Capacity reports

The Capacity Reporter generates four reports:

1. Overall summary report
2. Server allocation report
3. Front-End allocation report
4. Back-end allocation report

Overall Summary report

This report shows allocation per storage array that was connected to the VPLEX based on how much is allocated and how much is used.

VPLEX Information

Customer Name	CSC
Project	Moto
Serial	FNM00115100247
VPLEX Cluster	cluster-1
VPLEX Geo Version	5.1.0.03.00.07
Config file Date	Thu, 20 Dec 2012 16:06:47 UTC

VPLEX Imported Capacity Summary

Array Type	Status	Capacity(GB)
VMAX-5052	used	20943.39844
VNX	used	35602.448
VMAX-5052	claimed	645.9814453
DMX-5405	unknown	0
VNX	meta-data	255.9980469
VMAX-5052	meta-data	255.9996338
VNX	unclaimed	10

Server Allocation report

This report shows storage allocated to each of the hosts that are connected to the VPLEX. This port is mainly used for chargeback purposes.

Capacity allocation per server

Serial	Location	Name	Server Name	Capacity(GB)	# of luns
FNM00115100247	mdc	vplex2	CT11CORPAPP551	68	1
FNM00115100247	mdc	vplex2	CT11FIL011	858.48	3
FNM00115100247	mdc	vplex2	CT11FIL013	951.97	1
FNM00115100247	mdc	vplex2	ct11fil012	849.98	2
FNM00115100247	mdc	vplex2	ct11mesx100	23552	23
FNM00115100247	mdc	vplex2	ct11mesx101	23552	23
FNM00115100247	mdc	vplex2	ct11mesx102	23552	23
FNM00115100247	mdc	vplex2	ct11mesx103	23552	23
FNM00115100247	mdc	vplex2	ct11mesx104	23552	23
FNM00115100247	mdc	vplex2	ct11mesx105	23552	23
FNM00115100247	mdc	vplex2	ct11mesx106	10240	10
FNM00115100247	mdc	vplex2	ct11mesx107	10240	10
FNM00115100247	mdc	vplex2	e08k223	509.99	15
FNM00115100247	mdc	vplex2	e08k225	305.99	9
FNM00115100247	mdc	vplex2	e08k26	475.99	14
FNM00115100247	mdc	vplex2	e08k36	985.97	29
FNM00115100247	mdc	vplex2	t54k609	1597.95	47
FNM00115100247	mdc	vplex2	x42k600	15605.55	465
FNM00115100247	mdc	vplex2	x42k601	15605.55	465
FNM00115100247	mdc	vplex2	x42k604	407.99	12
			Total	200015.4	1221

Front-end allocation

This report shows how much storage is allocated to a particular front-end port. It gives an idea if storage is uniformly allocated across multiple front-end ports.

Front-end load and total capacity exported from VPLEX

FE Port	Nos of Server	Exported Capacity(GB)
director-1-1-A_A0-FC00	9	173475.08
director-1-1-A_A0-FC01	9	25384.36
director-1-1-A_A0-FC02	2	1155.97
director-1-1-B_B0-FC00	9	25384.36
director-1-1-B_B0-FC01	9	173475.08
director-1-1-B_B0-FC03	2	1155.97

Back-end allocation

This report shows how storage is allocated per back-end port.

Back-end load and total capacity allocated

BE Port	Target	Type	Capacity
director-1-1-A_A1-FC00	VMAX-HK192605052-11E:0	used	19787.43
director-1-1-A_A1-FC00	0x500601603ea05a41	unclaimed	10
director-1-1-A_A1-FC00	0x500601603ea05a41	used	35602.45
director-1-1-A_A1-FC00	0x500601693ea05a41	unclaimed	10
director-1-1-A_A1-FC00	0x500601693ea05a41	used	35602.45
		Total Capacity	91012.33
director-1-1-A_A1-FC01	VMAX-HK192605052-12E:0	used	19787.43
director-1-1-A_A1-FC01	0x500601613ea05a41	unclaimed	10
director-1-1-A_A1-FC01	0x500601613ea05a41	used	35602.45
director-1-1-A_A1-FC01	0x500601683ea05a41	unclaimed	10
director-1-1-A_A1-FC01	0x500601683ea05a41	used	35602.45
		Total Capacity	91012.33
director-1-1-A_A1-FC02	VMAX-HK192605052-5F:0	claimed	645.98
director-1-1-A_A1-FC02	VMAX-HK192605052-5F:0	used	1155.97
director-1-1-A_A1-FC02	0x500601643ea05a41	unclaimed	10
director-1-1-A_A1-FC02	0x5006016d3ea05a41	unclaimed	10
		Total Capacity	1821.95

Conclusions and next steps

The tool and methodology clearly demonstrate how to use Perl /XML and SQLite database to map VPLEX configuration data into relational tables and generate ad-hoc capacity reports. Taking a relational approach provides greater flexibility to our customers and technical staff to generate capacity reports. The flexibility mentioned above greatly improves TCE. We intend to transform this tool into a web service hosted in an Apache server as depicted and described below.

Figure 4

Once the XML configuration data is extracted from the VPLEX array, the Perl/XML parser engine loads the data into appropriate tables. A simple web interface will be provided so that users can generate template-based reports such as Top 10 hosts consuming Tier 1 storage, Top 10 highly allocated ports, and so on.

Bibliography

1. VPLEX – Powerlink documentation – http://powerlink.emc.com
2. VPLEX central - http://one.emc.com/clearspace/community/active/vplex_central
3. SQLite database - http://www.sqlite.org/
4. Perl - http://www.perl.org/
5. Xmlnotepad - http://xmlnotepad.codeplex.com