SAP HANA auf Power installieren und konfigurieren

Jochen Hein

10. April 2019

Vor etlichen Jahren hat SAP mit SAP HANA (ursprünglich High Performance Analytic Appliance) eine in Memory Datenbank und Entwicklungsumgebung vorgestellt. Zunächst war die Software nur auf Intel Systemen in Form einer Appliance mit vordefinierter Ausstattung an CPU und RAM zu bekommen. Seit einigen Jahren wird parallel zur Intel-Plattform auch SAP HANA auf der Power-Plattform angeboten. In beiden Fällen kommt Linux (SUSE Linux Enterprise Server for SAP Applications oder Red Hat Enterprise Linux) als Betriebssystem zum Einsatz.

1 SAP HANA auf Power - was spricht dafür?

In der Vergangenheit haben wir für unseren Kunden SAP Systeme mit DB2 als Datenbank überwiegend unter AIX betrieben. Mit dem Wunsch des Kunden zukünftig eine Vielzahl an Systemen auf SAP HANA umzustellen standen wir vor der Entscheidung, welche Hardware-Plattform wir in Zukunft vorrangig einsetzen wollen. Wir haben uns aus folgenden Gründen für Power entschieden:

- Unsere Erfahrungen mit der Redundanz, Ausfallsicherheit und Virtualisierung mit AIX auf Power sind sehr gut.
- Die Power-Architektur bietet für analytische Applikationen erhebliche Vorteile.
- Auf der Power-Plattform sind Scale-Up Systeme (mit nur einem Host) erheblich größer möglich als auf Intel.
- Durch die Virtualisierung mit PowerVM können LPARs dynamisch mit mehr oder weniger CPUs und RAM konfiguriert werden.
- Durch den Taylored Datacenter Integration Ansatz ist der Kunde in der Lage die Hardware gut und auf seine Anforderungen angepasst zu nutzen.
- Neue HANA Releases sind für beide Plattformen zeitgleich und mit gleichen Features verfügbar.
- Linux ist auf beiden Plattformen stabil und gut unterstützt.
- Linux HA und System-Replication sind für beide Plattformen verfügbar.

Als Nachteil sehen wir, dass wir mit PowerVM eine dem normalen Linux-Administrator unbekannte Virtualisierungslösung verwenden. Es gibt zwar KVM für Power, aber die Freigaben von SAP verwenden PowerVM - und aufgrund von den guten Erfahrungen mit AIX-LPARs sehen wir diesen Nachteil nicht als gravierend an, es bedeutet aber etwas Lernaufwand.

Mit Hilfe der PowerVM Virtualisierung können den einzelnen LPARs Resourcen wie CPUs, Memory oder Adapter (dynamisch) zugewiesen werden. Das hat in der Vergangenheit zu vielen Adaptern in den Systemen geführt, die verwaltet werden müssten. Seit vielen Jahren gibt es daher das Konzept der Virtual IO Server (VIOS), denen die Adapter zugewiesen werden und die dann den eigentlichen LPARs virtuelle Adapter zur Verfügung stellen.

Das geht sowohl für LAN- als auch für Fiber-Channel-Adapter (SAN). Das Konzept Shared-Ethernet-Adapter (SEA) belastet dabei die CPU des VIOs und für 10GBit/s muss nicht nur die LPAR, sondern auch der VIO geeignet konfiguriert sein. Für moderne Power-Systeme werden auch spezielle SRV-IO-Karten angeboten, die eine Hardware-Vrtualisierung unterstützen und den VIO entlasten sollen.

Jedes Power-System wird mit zwei VIOS installiert, die Adapter auf beide Systeme verteilt. Jeder VIO stellt dann entsprechende virtuelle Adapter den LPARs zur Verfügung. Für die Ausfallsicherheit wird jeder VIO z.B. über zwei Switches mit dem LAN bzw. SAN verbunden und stellt diese dann virtuell den LPARs zur Verfügung. Die LPAR fügt dann die virtuellen Ethernet-Adapter mittels Bonding zusammen, für SAN-LUNs verwendet man multipath. Damit ist jede LPAR redundant mit LAN und SAN versorgt und es kann ein VIO im laufenden Betrieb gewartet und durchgestartet werden.

Die Abbildung 1 zeigt schematisch eine entsprechende LAN-Anbindung.

Im Rahmen der ersten HANA-Umstellung der ersten Systemlandschaft in 2016 haben wir die ersten Erfahrungen mit Linux auf Power sammeln können. Zunächst haben wir nur eine handvoll SAP ABAP Systeme von DB2 auf HANA umgestellt. In der Folge haben wir dann auch weitere Systeme (SAP Java und SAP BO) dieser Landschaft mittels Multi-Tenant-Database-Containern (MDC) auf HANA migriert. Mit MDC können in einer SAP HANA Instanz (Installation) mehrere voneinander getrannte Datenbanken betrieben werden.

Derzeit betreiben wir die SAP Application Server noch mit AIX. Mittelfristig planen wir jedoch wo möglich auch Linux auf Power für die SAP Instanzen zu verwenden.

2 Installationen und Migrationen in großer Zahl

Bisher haben wir die Systeme als „Pets“ betrieben - jede LPAR wurde manuell installiert und konfiguriert. Bei nur einer handvoll Systeme ist das noch machbar (auch wenn sich die Systeme immer wieder in Kleinigkeiten voneinander unterscheiden) - nun kommen aber ca. 100 HANA Systeme zum Einsatz. Damit war für uns klar, dass wir die Systeme mit erheblich mehr Automatisierung installieren, konfigurieren und betreiben müssen. Tools dafür gibt es etliche (z.B. ansible, chef oder puppet) - was also nehmen?

Auf den DSAG Technologietagen der Deutschen SAP Anwender Gruppe wurde der SUSE Manager vorgestellt und von anderen SAP Kunden empfohlen. Da unser Linux Admin Team diesen schon im Einsatz hatte haben wir uns damit beschäftigt und entschieden ihn für uns zu nutzen. Im SUSE Manager sind verschiedene Tools integriert, die zur Installation und Konfiguration verwendet werden können:

- Autoyast für die automatische Installation
- Configuration Management mit SALT
- Patch-Managment für verschiedene Systemgruppen

Wir werden uns hier mit den ersten beiden Themen beschäftigen, also der Installation und dem Configuration Management.

3 Installation einer LPAR

In der Vergangenheit haben wir LPAR per „Formular“ bestellt, das dann manuell für die Installation und Grundkonfiguration verwendet wurde. Das hat für erhebliche Unterschieden zwischen den LPARs und zu Fehlern geführt. Daher haben wir uns entschieden die Installation mit einem Template-System zu unterstützen. Für eine LPAR legen wir z.B. fest, dass dort ein HANA-System `SID` laufen soll. Daraus generieren wir z.B. die Anforderung von Dateisystemen mit Vorschlagsgrößen, etwa wie in Tabelle 1.

Dateisystem	Größe
`/hana/data/SID`	9999
`/hana/log/SID`	9999
`/hana/shared/SID`	60

Tabelle 1: Vorschlag für Dateisysteme

Aus der Eingabe erzeugen wir SALT-Skripte, die entsprechende Logical Volumes und Dateisysteme anlegen. Für `/hana/data` und `/hana/log` wird von SAP erwartet, dass diese sehr schnell sein sollen. Dazu werden die Logical Volumes mit vier Stripes angelegt - das ging in der Vergangenheit oft schief. Entweder hat es der Anforderer vergessen zu beschreiben, oder es ging unterwegs unter. Oder es wurde nicht mit Stripes gearbeitet sondern die LUNs einfach hintereinander gepackt.

Die passende Anforderung von vier gleich großen Stripes für ein Physical Volume wird jetzt per Beschreibungsdatei angefordert und kann mit Skripten im SAN einfach angelegt werden. Hier ein statisches Beispiel:

```
01 lv_hana_sid:
02   lvm.lv_present:
03     - vgname: sidvg
04     - size: 1000G
05     - stripes: 4
06     - stripesize: 256K
```

4 Integration in den SUSE Manager

Im Unternehmen wurde der SUSE Manager als zentrales Admin-Tool für die Linux-Systeme (auch unabhängig von den HANA-Systemen) ausgewählt. Zunächst haben wir damit die Installation von LPARs mittels Autoyast durchgeführt und z.B. eigene/zusätzliche Pakete nachinstalliert oder Konfigurationsskripte ausgeführt. Mit der Integration in SALT kann der SUSE Manager aber viel mehr.

Zentral ist im SUSE Manager der Configuration Channel. Dieser kann an einzelne Systeme oder an eine Systemgruppe gehängt werden. In einem SALT Channel kann dann ein SALT-State definiert werden, der dann für diese Systeme verwendet wird. Aus den angehängten Configuration Channels wird dann der sogenannte „High State“ dynamisch erzeugt und kann dann auf die Systeme angewendet werden. Dabei werden Änderungen automatisch versioniert. Bei der Aufnahme eines Systems in den SUSE Manager wird ein „Activation Key“ verwendet, der auch wieder auf einen Configuration Channel verweisen kann. Dieser High State wird dann direkt angewendet...

Sehr hilfreich war die gute Dokumentation der SALT-Module bzw. -States, aber auch die „Best Practices SUSE Manager“ Dokumentation. Nach der grundsätzlichen Entscheidung für den SUSE Manager haben wir uns die verschiedenen Schritte beim Aufbau einer LPAR angesehen und versucht zu automatisieren. Glücklicherweise kann praktisch man jeden Themenkomplex

relativ unabhängig betrachten, so dass man sich nur jeweils mit einem überschaubaren Bereich auseinandersetzen muss. Unter anderem haben wir uns mit folgenden Bereichen beschäftigt:

- Konfiguration von Dateisystemen mittels `multipath` und LVM
- Einrichten der Netzwerk-Konfiguration mittels bonding und IP Aliases
- Installation und Konfiguration von Agenten zum Monitoring
- Anlegen von technischen Benutzern für die SAP HANA-Installation

5 Anlegen von Dateisystemen im SAN

Als Beispiel für eine komplexere Lösung mittels SALT werden wir uns die Dateisysteme genauer ansehen. Der Ablauf beginnt mit der Konfiguration der LPAR und dem Zuweisen der virtuellen Fiber-Channel-Adapter an die LPAR. Die Identifikation der Adapter (WWPNs) wird dann für das Zoning der LUNs an die richtige LPAR benötigt. Bisher wurde diese Information im Aufbau-Change dokumentiert und mittels Cut&Paste übertragen.

Im zweiten Schritt wurden mit Hilfes eines Excel-Formulars die einzelnen SAN-LUNs bestellt. Nach der Anlage der SAN-LUNs wurden die WWIDs wiederum im Change dokumentiert und manuell weiter verarbeitet. Aus den WWIDs wurde die passende multipath-Konfiguration erstellt und festgelegt, welche SAN-LUNs im LVM welches Physical Volume bereitstellt. Die Datei `multipath.conf` zu erstellen ist manuell aufwändig und fehleranfällig.

Wie geht's besser? Wir haben uns zunächst überlegt, dass die Daten zentral abgelegt werden sollen, damit alle beteiligten Gruppen unmittelbar darauf zugreifen können. Dann ist es sinnvoll die Daten sowohl maschinell zu erzeugen als auch zu verarbeiten. Beispielsweise legen wir die WWPNs der virtuellen Fiber-Channel-Adapter in dem Format ab, wie sie die Hardware-Management-Console (HMC) bei der LPAR-Definition anzeigt.

```
01 #!py
02 import csv
03 def run():
04     lines = []
05 ...
06     lines.append('multipaths {')
07     datei = __salt__['grains.get']('host') + '-uid.txt'
08     with open(datei) as f:
```

```
09          read_data = csv.reader(f, delimiter=';')
10          next(read_data)
11          for row in read_data:
12              name = row[0].split(',')[0]
13              pv = name.split('_')[1]
14              if pv == 'system':
15                  continue
16              wwid = row[0].split(',')[1]
17              lines.append('  multipath {')
18              lines.append('    wwid  3' + str.lower(wwid))
19              lines.append('    alias ' + pv)
20              lines.append('  }')
21
22      lines.append('  }')
23     return '\n'.join(lines)
```

Im Beispiel verwenden wir nicht den „normalen“ jinja-Renderer, sondern den Python-Renderer (`#!py`. Dabei wird diese Datei geladen und die Funktion `run()` aufgerufen. Als Rückgabewert wird die generierte Datei ausgegeben. Mit `__salt__['grains.get']('host')` kann auf ein Grain (hier der Hostname) zugegriffen werden. Damit ist es einfach möglich auch komplexere Dateien zu generieren. Leider ist diese Möglichkeit in der SALT-Dokumentation recht gut versteckt...

6 Beispiel für generierte SALT-Skripte

Im ersten Schritt war uns nicht klar, wie wir unsere Systeme dynamisch beschreiben können. Also haben wir erstmal mit einer möglichst einfachen Version begonnen. Hier ein vereinfachtes Beispiel für die Eingabe-Dateie zu Dateisystemen.

```
01 # mount;lv;vg;type;sizeg;stripes
02 /hana/data/SID;datalv,sidvg,xfs;1000,4
```

Daraus haben wir etwa folgendes SALT-Skript generiert:

```
01 lv_hana_sid:
02   lvm.lv_present:
03     - vgname: sidvg
04     - size: 1000G
05     - stripes: 4
```

```
06     - stripesize: 256K
07
08 fs_hana_sid:
09   blockdev.formatted:
10     - fs_type: xfs
11
12 /hana/SID:
13   mount.mounted:
14    - device: /dev/sidvg/lv_hana_sid
15    - fstype: xfs
16    - mkmnt: True
17    - persist: True
18    - mount: True
19    - opts:
20      - defaults
```

Dieses Verfahren hilft erheblich bei der Qualität der installieren Systeme und auch in der Reduktion des Aufwandes bei der Installation. Dennoch wollten wir versuchen, den Zwischenschritt des Erzeugens der SALT-Skripte zu vermeiden.

7 Kür: dynamische SALT-Skripte mit Python

In der statischen Darstellung verwendet SALT YAML als Format. Dieses wird durch den Standard-Renderer in JSON umgewandelt, dass dann von der SALT-Engine entsprechend verarbeitet wird. Alternativ kann auch hier ein Python-Renderer verwendet werden, der direkt das passende JSON-Format als Rückgabewert der Funktion `run()` liefert.

In der SALT-Dokumentation ist gut versteckt, dass man sowohl SALT-States als auch Template Dateien mittels Python „rendern“ kann. Wie sieht das aus? Der Trick ist `#!py` in der ersten Zeile. Die Python-Funktion `run()` erzeugt dann aus beliebigen Daten die entsprechende SALT-Konfiguration. Hier ein Template:

```
01 #!py
02 def run():
03     config = {}
04     ...
05     return config
```

Die Konfiguration für die logical Volumes erzeugen wir aus der Eingabedatei. In jeder Zeile wird ein logical Volume beschrieben. Im Python-Dictionary

sammeln wir die passende Konfiguration, die dann durch SALT auf dem System angewendet wird.

```
01 with open('os_host_filesystems.csv') as f:
02   read_data = csv.reader(f, delimiter=';')
03   next(read_data)
04   for row in read_data:
05      mount, lv, vg, type, sizeg, stripes = row
06      config['lv_' + lv ] = {
07        'lvm.lv_present': [
08          { 'name': lv },        { 'vgname': vg },
09          { 'size': sizeg + 'G' },
10          { 'stripes': stripes },{ 'stripesize': 256 }
```

Damit konnten wir die vorhandenen Daten zur Anlage eines Logical Volumes und des Dateisystems verwenden, ohne dass dazu ein erneuter Aufruf unseres Template-Systems notwendig war.

8 Installation von SAP HANA

Die Installation von SAP HANA erfolgt mit dem HANA Lifecycle Manager `hdblcm`. Von dem Tool gibt es mehrere Versionen mit verschiedenen Oberflächen. Wir verwenden in der Regel den Text-Modus von `hdblcm`. Das Tool läßt sich mit Parametern so aufrufen, dass eine unattended-Installation durchgeführt werden kann. Die Installation der Software erfolgt als `root`.

```
01 .../hdblcm -sid=<SID> --hostname=<host> \
02   -configfile=...
```

9 SAP HANA Grundkonfiguration

Nach der Installation der HANA Software und dem Anlegen der Tenants sind noch etliche Dinge zu tun:

- Anlegen Backup-Konfiguration für backint/TSM
- Import von Rollen und Rechten
- Anlegen von Benutzern zur Administration, Monitoring etc.
- Anlegen von Audit-Policies

- Standard-Konfiguration: Isolation-Level der Tenants, SQL-Parameter
- Jobs für Konsistenzprüfung der Datenbank
- ...

Für die weitere Konfiguration verwenden wir aktuell ein größeres Shell-Skript, dass die notwendigen Tenants anlegt, mit den notwendigen Security-Einstellungen versorgt und die technischen Benutzer in den Tenants anlegt, die benötigt werden. Die Laufzeit des Skriptes für drei Tenants beträgt weniger als 45 Minuten und es sind keine weiteren Nacharbeiten notwendig.

Bei Github findet man SALT-Module für SAP HANA, die wir eventuell in Zukunft verwenden wollen aber noch in Entwicklung sind. Wir hoffen, dass diese Module bald den Weg in den SUSE Manager finden. Damit wird sich dieser Teil nochmal erheblich vereinfachen und auch das zentrale Ausrollen neuer Konfigurationsparameter könnte sich deutlich einfacher gestalten. Auch für die Konfiguration von Cluster-Systemen mittels `crm` befinden sich SALT-Module in der Entwicklung.

10 Zusammenfassung

Die Entscheidung für die Power-Plattform hat sich für uns als sinnvoll erwiesen. Performance, Stabilität und Ausfallsicherheit sind sehr gut. Auch für Linux-Administratoren mit vorranging Intel-Erfahrung ist die Verwaltung der Systeme kein Problem. Lediglich der Umgang mit den VIOs und der darauf basierenden Konfiguration unterscheiden sich von der Intel-Plattform.

Mit der Verwendung von Autoyast verkürzt sich die Installationszeit des Betriebssystems auf wenige Stunden. „Schwierig“ ist hier das Bereitstellen der ISO-Datei der Installations-DVD. Bereits hier können Pakete zusätzlich installiert werden und mittel Skripten können beliebige Einstellungen vorgenommen werden.

Mit der Einbindung in den SUSE Manager wird das System einem „Activation Key“ zugeordnet. Dieser kann mit Hilfe von „Configuration Channels“ zum Customizing der LPAR verwendet werden. Zur Zeit haben wir hier für alle Systeme einheitliche hinterlegt. Damit erhalten wir innerhalb weniger Minuten eine Grundkonfiguration der LPAR. Interessant wird hier in der Zukunft die Gruppierung der Systeme nach verschiedenen Kriterien wie Produktion/Vorproduktion oder installierten Systemen (SAP HANA, SAP Netweaver oder andere).

Aus unserem Template-System generieren wir u.a. SALT-Skripte zur Anlage der notwendigen Dateisystem. In der Vergangenheit war dies mit erheblichem manuellem Aufwand verbunden. Mit Hilfe von SALT werden die SAN

LUNs automatisch mittels multipath zu Physical Volumes verbunden. Diese werden in Volume Groups aufgenommen und darauf dann Logical Volumes definiert. Diese werden mit einem Dateisystem formatiert und eingehängt. Aktuell dauert das Anlegen der Dateisysteme für eine SAP HANA Instanz ca. 20 Minuten, wobei die meiste Zeit für die Formatierung mittels `mkfs` draufgeht.

Die Installation der HANA Software, das Anlegen der Tenants und die Grundkonfiguration der HANA Datenbanken dauert per Skript nur noch ca. eine Stunde. Auch die Backup-Konfiguration erzeugen wir per SALT - alleine das erspart den Backup Kollegen nochmals etwa vier Stunden Aufwand.

Lohnt sich der Aufwand für die Automatisierung und Skript-Erstellung? Noch haben wir den Return-of-Invest nicht erreicht, aber alleine die verbesserte Qualität und einheitliche Umgebung erleichtern dem SAP Administrator die Arbeit ungemein.

10.1 Literatur/Links/Weitere Infos

SUSE Manager Best Practices
`https://www.suse.com/documentation/suse-manager-3/3.2/susemanager-best-practices/html/book.suma.best.practices/`
Python development for infrastructure management using Salt
`https://mirceaulinic.net/2017-12-19-salt-pure-python/`

10.2 Über Jochen

Jochen Hein ist SAP Basis Administrator seit 1988 und hat Erfahrung mit verschiedenen Plattformen vom Intel-System über diverse Unix-Plattformen bis zum Mainframe. Als Linux-Benutzer seit 1992 hat es lange gedauert bis zum Einsatz von Linux für die SAP-Systeme.

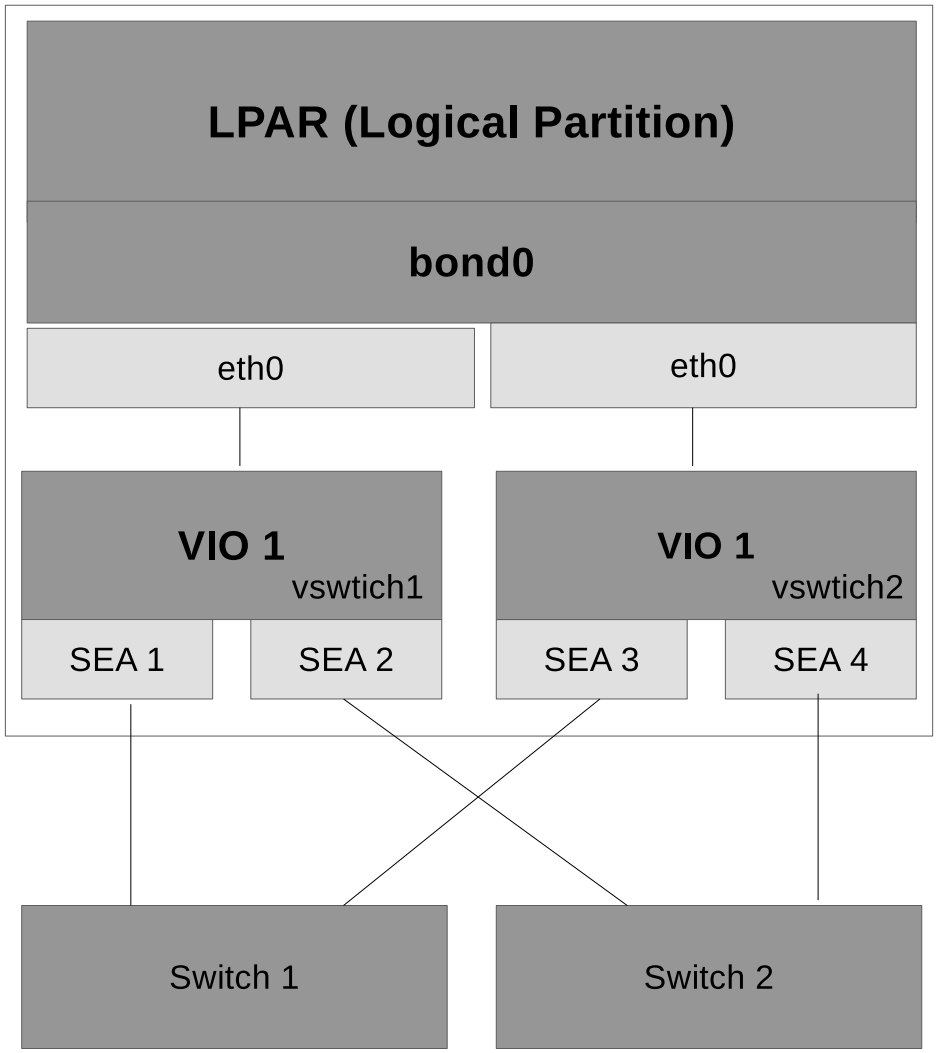

Abbildung 1: Schematische Darstellung der LAN-Virtualisierung