MARS – Dokumentation - Benutzerhandbuch

Letztes Update - 20.03.2014 - joerg.mann@1und1.de

Dokumentation aus Basis der Versionen:

Mars – 2014-03-07

Marsadm – 2014-03-07

Mars-Status – 2014-03-07 / 0.073

Vom Leser dieser Dokumentation wird erwartet, dass er grundlegendes Wissen im Umgang mit Filesystemen, den hier verwendeten Administrationstools hat und vor allem das er sicher im Umgang und dem Wissen um DRBD ist.

MARS, die hier beschrieben Usertools MARSADM und MARS-STATUS, sowie die Templates für das Monitoring mit Zabbix sind als OpenSource unter der URL https://github.com/schoebel/mars frei verfügbar.

Innerhalb der 1&1 wird MARS aktuell in den folgenden Bereichen eingesetzt:

- Shard-Hosting-Linux (infong, istore, ovzrdb, ovzxxx)
- ePages / Shop4
- statistik.schlund.de

Dokument History

24.02.2012 – erste Version

27.02.2012 – Syntax, Schreibfehler, Anmerkungen Holger

29.02.2012 – Notizen von Daniel

10.04.2012 – Überarbeitung mit aktuellen Informationen, MARS-STATUS hinzugefügt

28.09.2012 – Anpassungen für aktuelle Version von MARS Modul, MARSADM und MARS-STATUS

11.02.2014 – Überarbeitung auf aktuelle Versionen MARS und Usertools

12.03.2014 – Anpassungen für aktuelle Version von MARS Modul, MARSADM und MARS-STATUS

Inhaltsverzeichnis

0. Warum MARS ?

Die Entwicklung von MARS wurde begonnen da aktuell im Unternehmen eingesetzte Softwarepakete zur Replikation zahlreiche Schwachpunkte und Unzulänglichkeiten enthalten. Hauptsächlich trifft diese auf DRBD zu. Die wichtigsten Kritikpunkte sind hier:

- keine Möglichkeiten zur asynchronen Replikation
- keine Replikation über längere Distanzen (als BS – BAP / > 50km)
- keine Replikation von Datenträgern über 4 TB
- I/O Verhalten bei größeren Lasten unzulänglich
- maximal 2 Partner möglich

MARS soll diese Problematik zusammen mit weiteren wichtigen Funktionseigenschaften beheben.

0.1 MARS Meilensteine

Für das Projekt MARS sind folgende Meilensteine vorhanden:

- 19.09.2011 - erster Code MARS-Kernel, MARS Modul und MARSADM zum Testen verfügbar
- 21.10.2011 - erster Code MARS-STATUS verfügbar
- 01.09.2011 - Start Projekt TEC.1603 - MARS Light 1.0
- 07.02.2012 - Release Beta1
- 13.02.2012 - Inbetriebnahme Pilotsystem statistik.schlund.de
- 27.07.2012 - Start Projekt TECITO.1735 – MARS Light 2.0
- 01.06.2013 – Start Pilotsystem ShoLin
- 01.02.2014 – Start Projekt epages
- 20.03.2014 – RC1 MARS

0.2 Grundfunktionen MARS-Cluster

Der Aufbau von MARS ist im Vergleich zu DRBD grundlegend anders. Während DRBD vollkommen synchron betrieben wird, arbeitet MARS asynchron. Synchron heißt dabei, dass ein „OK“ an die Applikation erst zurück gegeben wird, wenn die Daten auf beiden Seiten geschrieben wurden. Bei MARS (asynchron) werden die zu schreibenden Daten auf dem aktiven Node (Primary-System) in ein Transaktionslogfile geschrieben. Danach wird das „OK“ zur Applikation gemeldet. Im Folgenden wird auf dem aktiven Node das Transaktionslogfile abgespielt. Weiterhin wird das Transaktionslogfile (bzw. dessen Änderungen) an die inaktiven Nodes (Partner) übertragen und dort ebenfalls abgespielt. Der hier entstehende Zeitverzug hat jedoch keinen Einfluss auf die Applikationen.

MARS nutzt (im Vergleich zu DRBD) verschiedene Kommunikationswege um Daten zwischen den Nodes eines Clusters auszutauschen. Diese Wege können entsprechend Betriebsmodus getrennt verwendet und gesteuert werden. Im Einzelnen sind dies:

Cluster Statusinformationen

- Diese Daten werden fortlaufend zwischen allen Nodes im Cluster ausgetauscht, auch unabhängig davon ob Ressourcen zwischen den Nodes repliziert werden oder nicht

- Der Austausch von Statusinformationen im Cluster kann während des Betriebes von MARS nicht unterbrochen werden (ausgenommen sind Netzwerkstörungen, Firewalleinstellungen und entladenes MARS-Modul)

  → Siehe dazu auch 2.1 Systemdaten /mars

Sync-Daten

- Sync-Daten werden zum Abgleich der Ressourcen zwischen dem aktiven und dem (den) inaktiven Nodes Blockweise übertragen
- Die Übertragung der Sync-Daten kann – getrennt für alle Nodes - pausiert und wieder gestartet werden

  → Siehe dazu auch 3.3 Ressource pause-sync / resume-sync

Transaktionslogfile Transfer

- Transaktionslogfiles enthalten die fortlaufenden Schreibzugriffe auf die Ressource
- Getrennt für jeden Node kann die Übertragung der Transaktionslogfiles pausiert und auch wieder gestartet werden

  → Siehe dazu auch 3.7 Ressource log-rotate / log-delete / log-delete-all

1. Störungen im Betrieb - WTW ?

Eine Beschreibung bekannter Probleme und Störungen ist im Wiki unter

http://wiki.intranet.1and1.com/bin/view/PO/ProjektMars

hinterlegt. Ein Kurzhilfe ist den Man-Pages der Scripte bzw. mit der Option „*--help*“ verfügbar.

F & A:

Frage: Was für Kommandos können ohne geladenes Mars Modul ausgeführt werden?
Antwort: create-cluster und join-cluster
→ Siehe auch unter 2.2 Ressource / Cluster einrichten

Frage: Kommunizieren die Nodes über SSH?! Wozu -A? No go.
Antwort: In der aktuellen Betaversion / Pilotbetrieb ja, später wird dies (sicher) überarbeitet

Frage: Kommando verify ?
Antwort: Diese Kommando ist in MARS nicht vorhanden. Im normalen Betriebsfall sind die Ressourcen nicht identisch, da MARS ja asynchron arbeitet und eine Verzögerung beim abspielen der Logfiles vorhanden ist. Als „Abhilfe“ kann hier der normale „Sync“ verwendet werden. Dieser arbeitet sozusagen als „Verify mit Repair“.
→ Siehe dazu auch unter 3.3 Ressource pause-sync / resume-sync

Frage: Kann ein fake-sync rückgängig gemacht werden?
Antwort: Ja, die Ressource auf invalidate schalten.
→ Siehe auch unter 3.4 Ressource fake-sync

Frage: Ist es eine gute Idee einen Port über 1024 zu nehmen?
Antwort: Default ist Port 7777, ab der aktuellen Version aber zur Compiletime einstellbar.

Frage: Kann der Zugriff auf die LV (Daten-Device) gesperrt werden?
Antwort: Ja, unter Verwendung des „detach“ Kommandos.
→ Siehe auch unter 3.2 Ressource attach / detach

Frage: Gibt es ein Rollback der Transaktionslogfiles?
Antwort: In der aktuell geplanten Light-MARS-Version nicht, erst in einer der späteren Versionen.

Frage: Wie entferne ich einen Node aus dem Cluster der physikalisch nicht mehr vorhanden ist ?
Antwort: Unter Verwendung des „leave-resource“ Kommandos.
→ Siehe auch unter 2.3 Ressource / Cluster entfernen

Frage: Wie vergrößere ich das Daten- und/oder das MARS-Daten-Volume ?
Antwort: Unter Verwendung des „resize“ Kommandos.
→ Siehe auch unter 3.9 Ressource resize

Frage: Wie kann ich die Verbindungen des Nodes überprüfen ?
Antwort: Kommandos „*netstat -tulen*“ | grep 7777 und „*ls -l /mars/ips/**“

2. Einrichtung

Die nachfolgende Dokumentation geht davon aus, dass die Software MARS als Kernelmodul in der aktuellen Version auf den Systemen installiert ist. Dies gilt ebenso für die verwendeten Hilfsprogramme von MARS.

In den aktuell verwendeten Version von MARS werden einige Parameter für die Betriebsumgebung in den Konfig-Files des MARS Modules verwaltet. Diese Parameter sind für die derzeit verwendeten Systeme angepasst und hier auch allgemein gültig.

2.1 Systemdaten /mars

Zur Einrichtung von MARS ist auf den System eine Partition /mars (vorzugsweise – siehe 2.0) einzurichten. Auf dieser werden interne Files und Systemlinks (als Sym-Link-Tree) zu Steuerung von MARS, sowie die Transaktionslogfiles abgespeichert. Diese Partition sollte nach Möglichkeit nicht auf dem gleichen Volumes/Platten/Enclosures liegen, wo auch die Ressourcen liegen. Zu empfehlen sind hier getrennt Disk und Devices, mit einem entsprechenden Raidset um einen optimale Wirkungsgrad und eine ordnungsgemäße Funktion von MARS zu gewähreleisten. Die Größe der Partition sollte ausreichend sein um eine möglichst große Anzahl von Transaktionslogfiles zu speichern. Zu beachten ist dabei, dass mit hoher der Anzahl der Transaktionslogfiles mehr Möglichkeiten zur eventuellen Fehlerbehebung im „Fall der Fälle“ zur Verfügung stehen.

Die vom MARS angelegten Systeminformationen werden ständig zwischen allen im Cluster bekannten Nodes ausgetauscht. Dieser Austausch läuft parallel auf allen bekannten Nodes die lokalen und kann nicht angehalten werden. Zu den Informationen gehören Daten zu den verwendeten Ressourcen, deren Größe und Zustand. Ebenfalls werden hier Informationen zum Zustand des Primary/Secondary, des Sync- und Replay-Zustandes übertragen. Durch den Austausch der Statusinformationen sind auf allen im Cluster bekannten Nodes ständig alle Informationen zu allen beteiligten Nodes und Ressourcen vorhanden.

Ein Eingriff in die Systemdaten unter /mars ist für den Benutzer in der Regel nicht erforderlich. Die hier hinterlegten Daten und Informationen werden selbstständig durch MARS bzw. die Usertools verwaltet.

→ Erfahrungen (der statistik.schlund.de) zeigen, dass 15GB pro Stunden geschrieben werden können.

→ In der aktuellen Version soll für die Systemdaten und das Daten-Device ein unterschiedlicher Filesystemtyp verwendet werden. Vorzugsweise ext4 für /mars und XFS für das Daten-Device.

2.2 Ressource / Cluster einrichten

Die Einrichtung der Ressourcen (Daten-Devices) erfolgt nach den Vorgaben des Systems. Es sind entsprechende PV's / LV's bzw. Disk's bereit zu stellen. Das Volume ist auf dem aktiven System (Primary) und den inaktiven Systemen (Secondary) identisch anzulegen. In Beispiel dieser Dokumentation legen wir in der Volume Group „vg-test“ ein Logical Volume „Mars-ResA“ mit 100GB an.

lvcreate vg-test -L 100G -n LV-ResA

Als nächstes wird das aktive System in Betrieb genommen. Dazu wird in Abfolge der Cluster und die Ressource innerhalb von MARS angelegt, sowie das Modul geladen.

marsadm create-cluster

modprobe mars

marsadm create-resource Mars-ResA /dev/vg-test/LV-ResA

→ Das „create-cluster“ Kommando darf immer nur **einmal** auf **einem** Node ausgeführt werden, da sonst die Struktur des Clusters zerstört wird.

-> Zu beachten ist, dass eine entsprechende Freischaltung und ein Zugriff mit dem Benuter „root“ zwischen den Nodes möglich sein muss (ssh -A verwenden).

-> marsadm geht im normalen Betrieb davon aus, dass die lokale IP-Adresse des Interfaces eth0 verwendet wird. Sofern hier eine andere IP-Adresse verwendet werden soll, ist beim Kommando „join-cluster“ die Option --ip=x.x.x.x zusätzlich zu verwenden.

Das Mars-Device der angelegten Ressource steht jetzt unter „/dev/mars/Mars-ResA“ zur Verfügung und kann verwendet (gemountet) werden.

Im nächsten Schritt werden nun die entsprechenden inaktiven Nodes (Secondary) in Betrieb genommen. Dazu werden die Nodes dem Cluster hinzugefügt. Somit wird zunächst sicher gestellt, dass alle Statusinformationen des Cluster auf dem Node verfügbar sind. Nachdem der inaktive Node mit dem Cluster verbunden ist, wird die Ressource hinzugefügt.

marsadm join-cluster $hostname-primary

modprobe mars

marsadm join-resource Mars-ResA /dev/vg-test/LV-ResA

Entsprechend Anzahl der inaktiven Nodes und Ressourcen sind die Schritte zu wiederholen. Die Installation bzw. Vorbereitung des Clusters ist jetzt abgeschlossen, die inaktiven Nodes sind jetzt betriebsbereit.

Alle im Cluster vorhandenen Nodes tauschen untereinander fortlaufend selbstständig Systemdaten aus. Dieser Mechanismus ist unabhängig von den verbunden Ressourcen.

2.3 Ressource / Cluster entfernen

Damit auch Nodes und Ressourcen wieder aus einem Cluster entfernt werden können, stehen über das Usertool MARSADM entsprechende Kommandos zur Verfügung.

marsadm leave-cluster Mars-ResA	→ Node aus dem Cluster entfernen
marsadm leave-resource Mars-ResA	→ Ressource aus dem Cluster entfernen

Vor dem entfernen eines Nodes oder einer Ressource aus einem Cluster ist diese auf dem Node entsprechend zu stoppen:

marsadm pause-replay Mars-ResA	→ Replay auf der lokalen Resource anhalten
marsadm pause-sync Mars-ResA	→ Sync auf der lokalen Ressource anhalten
marsadm detach Mars-ResA	→ lokale Ressource detachen
marsadm disconnect Mars-ResA	→ lokale Ressource disconnecten

3. MARSADM

Die Steuerung eines Mars-Clusters erfolgt über das Usertool „MARSADM“. Die Anzeige der verschiedenen Zustände des Clusters, der Nodes und der Transaktionslogfiles erfolgt über das Usertool „MARS-STATUS“. Eingriffe von „Hand“ sollten generell vermieden werden. Dies auch aus dem Grund, dass verschiedene Funktionen aus technischen Gründen und bedingt durch den fortlaufenden Entwicklungsprozess der Software ständig verändert und angepasst werden. Bei der Entwicklung von MARS wurde darauf geachtet, dass hinsichtlich der implementierten Kommandos, immer die gleichen Funktionen angewandt und hinterlegt werden.

3.1 Ressource connect / disconnect

Nach der Einrichtung der Nodes sind diese in der Regel „connected“. In „disconneted“ Zustand werden keine Transaktionslogfiles, sondern ausschließlich Statusinformationen des Clusters übertragen. Mit den folgenden Kommandos kann dieser Zustand verändert werden:

marsadm connect Mars-ResA	→ die Ressource wird lokal connected
marsadm disconnect Mars-ResA	→ die Ressource wird lokal disconnected

→ Die Kommandos „Connect / Disconnect“ haben keine direkte Auswirkung auf einem Primary-Node, sondern nur auf den Secondary Nodes.

→ Anstelle des Ressourcennamens kann auch der Zusatz „all“ verwendet – der sich dann auf alle Ressourcen die mit dem Node verbunden sind auswirkt.

→ Der Status des „Connect-Schalter“ ist über MARS-STATUS einsehbar.

→ Siehe dazu auch unter 2.1 Systemdaten /mars

3.2 Ressource attach / detach

Mit den Kommandos „*detach*“ kann eine Ressource vollständig vom Betrieb abgetrennt werden. Damit werden alle Schreib- und Leseoperation auf die Ressource unterbunden. Über das Kommando „*attach*“ kann dieser Zustand wieder aufgehoben werden.

marsadm detach Mars-ResA	→ Zugriffe auf die Ressource Mars-ResA werden unterbunden
marsadm attach Mars-ResA	→ Zugriffe auf die Ressource Mars-ResA werden wieder erlaubt

→ Anstelle des Ressourcennamens kann auch der Zusatz „all“ verwendet – der sich dann auf alle Ressourcen des Nodes auswirkt.

→ Der Status des „Attach-Schalter“ ist über MARS-STATUS einsehbar.

→ Siehe dazu auch unter 2.1 Systemdaten /mars

Während des „*detach*“ Betriebes werden nur die Statusinformationen des Clusters übertragen. Eine Wiedergabe von Transaktionslogfiles oder ein laufender Sync-Prozesse wird angehalten.

3.3 Ressource pause-sync / resume-sync

Analog dem vom DRBD bekannten Verhalten müssen die Ressourcen bei (bzw.) vor der ersten Inbetriebnahme oder für den Fall einer vollständigen Wiederherstellung gesynct (abgeglichen) werden. Dabei wird die betreffende Ressource blockweise vom aktiven Node auf den zu syncenden Node übertragen. Während des laufenden Sync-Prozesses arbeiteten alle anderen Nodes uneingeschränkt weiter.

Mit den folgenden Kommandos kann der Sync-Prozess angehalten bzw. wieder gestartet werden.

marsadm pause-sync Mars-ResA	→ der Sync-Prozess für die Ressource wird auf dem lokalen Node angehalten
marsadm resume-sync Mars-ResA	→ der Sync-Prozess für die Ressource wird auf dem lokalen Node wieder gestartet

→ Die Kommandos haben keine Wirkung auf dem Primary-Node !

→ Anstelle des Ressourcennamens kann auch der Zusatz „all“ verwendet – der sich dann auf alle Ressourcen des Nodes auswirkt.

→ Der Status des Sync ist über MARS-STATUS einsehbar.

Zu beachten ist, dass der Sync-Prozess als Fast-Full-Sync im MARS implementiert ist. Dieser Modus impliziert einerseits das syncen der Ressourcen und andererseits das abspielen von Transaktionslogfiles. Beide Funktionen laufen dabei parallel. Im Ergebnis findet kein wirklich voller Sync statt, es werden nur veraltete Daten neu geschrieben.

3.4 Ressource fake-sync

Unter bestimmten Umständen – wie z.B. zu Testzwecken oder bei einer Erstinbetriebnahme von leeren Ressourcen – kann der Sync-Prozess gefakt (simuliert) werden. Damit erklärt MARS den Datenbestand als valide (gültig) ohne die Daten abzugleichen. Zur Steuerung wird das nachfolgende Kommando verwendet:

marsadm fake-sync Mars-ResA	→ die Ressource wird als gesynct betrachtet

→ Das Kommando prüft nicht ob bereits Daten auf dem Device vorhanden sind ! Es kann daher nur zu Testzwecken oder vorher entsprechend gelöschten Devices verwendet werden.

→ Anstelle des Ressourcennamens kann auch der Zusatz „all“ verwendet – der sich dann auf alle Ressourcen des Nodes auswirkt.

→ Der Status des Sync ist über MARS-STATUS einsehbar.

3.5 Ressource invalidate

Entsprechend Betriebsmodus ist es notwendig, dass Daten einer Ressource für ungültig erklärt werden müssen, um so z.B. einen erneuten Fast-Full-Sync zu starten. Die Steuerung erfolgt mit den folgenden Kommandos:

marsadm invalidate Mars-ResA	→ die Ressource wird auf dem aktuell Node ungültig

→ Anstelle des Ressourcennamens kann auch der Zusatz „all“ verwendet – der sich dann auf alle Ressourcen im Cluster auswirkt.

3.6 Ressource pause-replay / resume-replay

Ebenso wie der Sync-Prozess kann auch der Replay-Prozess – also das Abspielen der Transaktionslogfiles – pausiert und wieder neu gestartet werden.

marsadm pause-replay Mars-ResA	→ für die Ressource wird nur lokal kein Logfile mehr abspielen
marsadm resume-replay Mars-ResA	→ Logfile für die Ressource wird nur lokal wieder abspielen

→ Anstelle des Ressourcennamens kann auch der Zusatz „all“ verwendet – der sich dann auf alle Ressourcen im Cluster auswirkt.

→ Der Status des „Replay-Schalter“ ist über MARS-STATUS einsehbar.

3.7 Ressource log-rotate / log-delete / log-delete-all

Im Betrieb werden alle Änderungen der Ressource (Schreibzugriffe auf das Daten-Device) in Transaktionslogfiles abgespeichert. Diese werden auf alle Nodes im Cluster übertragen, die mit der Ressource verbunden sind (join-ressource). Für den Betrieb ist eine Verwaltung der Transaktionslogfiles über den Userspace (Cron-Job) zwingend notwendig. Die Verwaltung ist entsprechend den Betriebsparametern anzupassen. Gelöscht werden können im Betrieb nur die nicht mehr benötigten und bereits abgespielten Transaktionslogfiles. Nicht mehr benötigt heißt in diesen Fall, dass eine Reihe von Bedingungen erfüllt sein müssen:

- das Transaktionslogfile muss auf alle aktiven und inaktiven Nodes übertragen sein
- die Transaktionslogfiles müssen auf den beteiligtem Node abgespielt sein.
- das Transaktionslogfile muss auf allen Systemen den gleichen Zustand (siehe TODO) haben

Die Transaktionslogfiles werden (wie auch andere interne Files) mit einer fortlaufenden Versionsnummer versehen. Durch eine Rotation der Transaktionslogfiles, wird eine neue Version des Transaktionslogfiles erzeugt. MARS erkennt selbstständig was für Transaktionslogfiles noch benötigt oder gelöscht werden können. Mit dem Löschen nicht mehr benötigter Transaktionslogfiles werden gleichzeitig automatisch weitere nicht mehr benötigte Daten auf dem System entfernt.

marsadm log-rotate Mars-ResA	→ für die Ressource das Transaktions-Logfile rotieren
marsadm log-delete Mars-ResA	→ für die Ressource nächstes inaktive Transaktions-Logfile löschen
marsadm log-delete-all Mars-ResA	→ für die Ressource alle nicht mehr benötigten Transaktions-Logfiles löschen

→ Anstelle des Ressourcennamens kann auch der Zusatz „all“ verwendet – der sich dann auf alle Ressourcen im Cluster auswirkt.

→ Der Status der Logfiles ist über MARS-STATUS einsehbar.

3.8 Ressource Primary

Mit den Kommando „*primary*“ kann ein Node in den aktiven Modus umgeschaltet werden. Im Cluster kann es für jede Ressource nur einen aktiven Node (Primary) geben. Die Anzahl der inaktiven Nodes (Secondarys) ist nicht beschränkt. Die Umschaltung des Modus wirkt sich auf alle Systeme die mit dieser Ressource verbunden sind aus. Dies kann (zB. wegen einem gemouteten Ressource auch erst zu einem späterem Zeitpunkt erfolgen).

marsadm primary Mars-ResA	→ für die Ressource wird in den aktiven Modus geschaltet

→ Zu beachten ist, dass die Umschaltung des Modus sich auch auf alle anderen Nodes im Cluster auswirkt.

→ Der Status der Nodes ist über MARS-STATUS einsehbar.

3.9 Ressource up / down

Die Kommandos „up“ und „down“ sind nur aus Gründen der Kompatibilität zu DRBD in MARS eingefügt wurden. Im normalen Betrieb sind diese Kommandos eigentlich nicht notwendig. Beim Aufruf von MARS mit dem Kommando „down“ werden die folgenden Aktionen ausgeführt:

- pause-replay -> Abspielen des Transaktionslogfiles anhalten
- pause-sync -> laufenden Sync-Prozess anhalten
- disconnect -> keine Logfiles/Sync-Daten mehr übertragen
- detach -> Ressource vom System trennen

→ Anstelle des Ressourcennamens kann auch der Zusatz „all“ verwendet – der sich dann auf alle Ressourcen im Cluster auswirkt.

Das Kommando „up“ führt die entsprechenden Kommandos in umgekehrter Reihenfolge (attach, connect, resume-sync, resume-replay) aus.

3.9 Ressource resize

Das Kommando *„resize“* wird benötigt um vorhandene Devices nachträglich zu vergrößern. Ein Resize ist nur möglich wenn der Sync- und der Replay nicht angestoppt sind.

marsadm resize Mars-ResA → die Größe Ressource wird angepasst

→ Zu beachten ist, dass nur Auswirkungen hat wenn alle Nodes entsprechend veränderte Devices haben.

→ Der Resize-Status der Nodes ist über MARS-STATUS einsehbar.

Ein typisches Vorgehen zum Resize wäre, zuerst auf den Secondary-Nodes das Device (LVM) zu vergrößern. Im nächsten Schritt wäre das Device auf dem Primary anzupassen. Nun kann das Kommando „*resize*“ ausgeführt werden. Im folgenden wäre beispielsweise bei verwendeten XFS-Filesystem noch ein „*xfs_growfs*„ notwendig.

4. MARS-STATUS

Während des Betriebes von MARS werden zwischen allen Nodes im Cluster Statusinformationen ausgetauscht, die von den beteiligten Nodes ausgewertet und verarbeitet werden. Diese Informationen, sowie weitere Parameter aus der Betriebsumgebung werden durch das Usertool MARS-STATUS ausgelesen, aggregiert und entsprechend formatiert angezeigt.

MARS-STATUS kann ohne zusätzliche Optionen aufgerufen werden. Das Programm arbeitet ausschließlich im Read-Only Modus, Änderungen an den Datenbeständen oder den Einstellungen werden nicht vorgenommen.

→ Das Programm MARS-STATUS befindet sich aktuell in der Entwicklung. Demnach kann eine fehlerfreie Erkennung der verschiedenen Zustände innerhalb von MARS – insbesondere bei der Erkennung des Status – nicht garantiert werden.

4.1 Optionen

Das MARS-STATUS Programm kann entsprechend Anforderungen zusätzlichen mit verschiedenen Optionen aufgerufen werden:

-- ressource x

Mit dieser Option wird die Ausgabe der Statusinformation auf eine bestimmte Ressource beschränkt.

-- interval x

Die Anzeige rotiert selbstständig in x Sekunden.

-- history

Neben den Statusinformationen werden Angaben zu den Transaktionslogfiles, deren Versionen und dem Status ihrer Verarbeitung und die Verknüpfung der Transaktionslogfiles angezeigt.

-- system

Die Ausgabe wird um die einstellbaren Systemoptionen erweitert.

-- debug

Es werden zusätzlich interne Ausgaben des MARS Modules aus dem Kernel ausgelesen.

Eine Kombination der verschiedenen Parameter ist möglich, gewollt und sinnvoll.

4.2 Monitor Funktion

Für den Betrieb im Normal-Modus wird „MARS-STATUS“ ohne Parameter aufgerufen. Vom Programm wird (sofern vorhanden) ein entsprechendes Exit-Code und eine Fehlermeldung ausgegeben.

4.3 Anzeige

In den Ausgaben von MARS-STATUS werden an verschiedenen Stellen zusätzliche Angaben hinzugefügt, die einen Hinweis auf mögliche notwendige Arbeiten und Zustände anzeigen. Unterschieden werden hier:

- WORK: System arbeitet (zB. Sync / Replay läuft)
- TODO: System ist aktuell mit einem Fehler versehen (zB. Logfiles sind nicht identisch)
- HINT: System benötigt Eingriff (zB. Ältere Logfiles können gelöscht werden)

(*Im folgenden sind rote Texte als Kommentar zu den Ausgaben von MARS-Status angegeben*)

a) MARS-STATUS ohne Optionen :

Die Darstellung in MARS-STATUS ist wie folgt aufgebaut:

1) Statusinformation zu Kernel, Mars, Marsadm und Mars-Status

2) Systemzustand (Diskspace, Emergency)

3) Ressourcen
   3.1) Übersicht Node

   3.2) Device: physikalisches und logisches Device der Resource, Resourcenname und Check nach Resize und Mountpoints

   3.3) Sync: Größe des Devices und der Zustand der gesyncten Daten, sofern ein Sync läuft und Werte berechenbar sind, verbleibende Größe und Restzeit

   3.4) Logfile: Größe und Name des Logfiles

   3.5) Replayed: Größe vom abgespielten Logfile, sofern ein Replay läuft und Werte berechenbar sind, verbleibende Größe und Restzeit

   3.6) Actual: Secondary/Primary, Sync aktiv/inaktiv, Logfileupdate aktiv/inaktiv

   3.7) Switches: Aattch, Connect, Sync, Allow-Replay

   3.8) Zusammenfassung der Ressource

Beispiel:

MARS Status - logs1ba01, 0.073
MARS Admin - /usr/local/bin/marsadm $Id: 32e134c96048a95ece20c3fbdd239a12b06f3e6f $
MARS Module - 196ba5e71f09315b7222358acd0e3f0d3070f29d (root@ 2014-03-07 01:48:28)
MARS Kernel - 3.2.46_WebSrv08

-> Cluster Diskspace: smoothly Transaktions: smoothly Connects: TODO Synclimit: smoothly
(Anzeige der Zusammenfassung des Systemstatus)
---> Resources <---
-> check resource data, with 0.244TB, Primary Node is logs1ba01
(Anzeige der Zusammenfassung des Devices der Resource)
-> local Node (logs1ba01 [10.72.54.16]) as Primary, System alive
(Anzeige lokaler Node)
Device : Disk-Device /dev/vg01/data, not enlarged, used as Mars-Device /dev/mars/data
---> WORK: mounted as /mnt/data
(verwendetes Disk- und Daten-Device, check mount (nur primary!) und check resize)
Sync   : 268435456000 bytes (0.244TB) synced = 100.00%
(Status des Sync, Anzeige Fortschritt und evt. Noch verbleibende Restzeit)
Logfile : 373579748 bytes (0.348GB) in log-000000728 active
(Versionsnummer, Größe des Logfile)
Replayed: 372618096 bytes (0.347GB) now replayed, Todo 0, completed 99.74%
---> WORK: Replay in progress = (99.74% < 100.00%)
(Fortschritt, Restwert und verbleibende Restzeit des des Replay)
Actual : Status=Primary, used Device=on, Attached=on
(Primary: Status des Nodes und des Devices)
Switches: Attach=on [masked: Connect=on Sync=on AllowReplay=on]
(Primary: inaktiv)
-> remote Node (logs1bs01 [10.72.53.16]) as Secondary, System alive
(Anzeige weiterer Node)
Device : Disk-Device /dev/vg01/data, not enlarged
Sync   : 268435456000 bytes (0.244TB) synced = 100.00%
Logfile : 373579748 bytes (0.348GB) in log-000000728 active
Replayed: 372618096 bytes (0.347GB) now replayed, Todo 0, completed 99.74%
---> WORK: Replay in progress = (99.74% < 100.00%)
Actual : Status=Secondary, Syncstatus=off, Logfileupdate=on, Attached=on
(Secondary: Status des Nodes, Sync und Logfile)
Switches: Attach=on Connect=on Sync=on AllowReplay=on
(Secondary: Status der Switches)
-> modus for resource data is clustered (2 nodes)
(Anzeige Zusammenfassung Resource)

b) MARS-STATUS mit Optionen „--system“:

Mit Verwendung der *„system“* Option werden verschiedene Werte und Ist/Soll des Nodes angezeigt. Der System-Block wird vor dem Ressourcen-Block angezeigt.

Beispiel:

---> Systemdata <---
-> AVG Limit is now unsed,
-> Memory Limit is set to 20 %, actualy used 8536 kb,
-> Traffic Limit Client is set to 0 kb/s, actualy used 0 kb/s,
-> Traffic Limit Server is set to 0 kb/s, actualy used 3989 kb/s,
-> Traffic Writeback is set to 0 kb/s, actualy used 4100 kb/s,
-> I/O Writeback is set to 0 kb/s, actualy used 4100 kb/s,
-> Server-IO Limit is set to 0 kb/s, actualy used 0 kb/s, Flying IO is actualy null
-> Copy Read: Prio is now unsed, Flying IO is actualy null
-> Copy Write: Prio is now unsed, Flying IO is actualy null
-> LoggerMemory is actualy 40 kb,
-> FreeSpaceLimit LogRotate is set to 32 gb,
-> Network-IO-Timeout is set to 30 sec,
-> Clear Page Cache is actualy 10 sec,
-> Statusfile Rollover is actualy 3 sec,
-> Modus: Fast Full Sync is actualy on, AIO Sync is actualy on, Delay say Overflow is actualy off, Emergency is actualy off, Logger Resume is actualy off,
-> LamportClockDifferenz 0.00 sec, Mars Port is set to 7777 ,
-> Free-Space-Limit on /mars is set to 2 mb (actualy 201586.68 mb used),

(Anzeige des aktuell eingestellte Systemparameter, soweit verfügbar (vom MARS Modul bereit gestellt) werden die SOLL und IST Werte mit angezeigt. Entsprechend der eingesetzten Version kann die Anzeige abweichen)

c) MARS-STATUS mit Optionen „--debug“:

Mit Verwendung der *„debug“* Option werden Debug Informationen für die Ressourcen und des Systems angezeigt. Der Debug-Block für die Ressource wird unter dem Ressourcen-Block angezeigt, der Allgemeine am Ende der Anzeige.

Beispiel:

-> Debug for resource-data
Fri Mar 7 08:17:01 2014: 1394176621.960258021 MARS mars_handler367[0] old/sy_generic.c:530 get_inode(): cannot stat mars/resource-data/log-000000572-logs1ba01', status = -2
---> Debug <---
-> Main-Debug:
Fri Mar 7 14:12:40 2014: 1394197960.911247186 MARS mars_handler418[0] ernel/mars_net.c:566 mars_recv_raw(): #418494 got EOF from socket (done=0, req_size=24)
Fri Mar 7 14:12:40 2014: 1394197960.911247188 MARS mars_handler418[0] el/mars_server.c:308 handler_thread(): #418494 recv cmd status = -32
Fri Feb 7 15:01:48 2014: 1391781708.191851535 MARS mars_handler191[6] el/lib_mapfree.c:164 mapfree_get(): can't open file '/mars/resource-data/log-000000031-logs1ba01' status=-2
Fri Feb 7 15:01:48 2014: 1391781708.191851536 MARS mars_handler191[6] ernel/mars_aio.c:1112 aio_switch(): could not en file = '/mars/resource-data/log-000000031-logs1ba01' flags = 32770
Fri Feb 7 15:01:48 2014: 1391781708.191851537 MARS mars_handler191[6] ernel/mars_aio.c:1138 aio_switch(): status = -2

d) MARS-STATUS mit Optionen „--history“:

Mit Verwendung der *„history“* Option werden die vorhandenen Versionen des Logfiles und deren Zustand - bezogen auf Verarbeitung, Replay, Größe und Prüfsummen - angezeigt. Der History-Block wird am Ende der Anzeige eingefügt.

Beispiel:

```
-> History Replay/Status
   |
   |=> Vers.000000008 ==> old Version, Logfile is deleted ...
   |  |                  |=> Node: searchs1bs02          Check: xxx    Size: 2144636 bytes (0.002GB)   (no longer available logfile)
   |  |                  |=> Node: webfiless1bs01        Check: xxx    Size: 2144636 bytes (0.002GB)   (no longer available logfile)
   |  |                  |=> Node: webfiless1bs02        Check: xxx    Size: 2144636 bytes (0.002GB)   (no longer available logfile)
   |  |=> WORK: Version is deleted the next log-rotate ...
   |
   |=> Vers.000000009 ==> Logfile Size: 2142848 bytes (0.002GB) by webfiless1bs01 from Fri Mar  7 10:45:12, include hypothetically 0 sec
   |  |                  |=> Secondary Node: searchs1bs02   Check: xxx   Size: 2142848 bytes (0.002GB) ---> replay OK, verify OK
   |  |                  |=> Primary Node: webfiless1bs01   Check: xxx   Size: 2142848 bytes (0.002GB) ---> replay OK, verify OK
   |  |                  |=> Secondary Node: webfiless1bs02 Check: xxx   Size: 2142848 bytes (0.002GB) ---> replay OK, verify OK
   |  |=> WORK: Version are actual and used. Wait for start replay ...
```

5. Betriebsparameter

Während des Betriebes von MARS werden zwischen allen Nodes im Cluster automatisch Statusinformationen ausgetauscht, die von den beteiligten Nodes ausgewertet und verarbeitet werden. Zusätzlich werden verschiedene Parameter aus der Betriebsumgebung (/proc/sys/mars/*) ausgelesen, auswertet und ebenfalls verarbeitet. Im Ergebnis erfolgt ein speziell auf den Node angepasster Betrieb des MARS Modules. Die Änderung dieser Werte gehen nach einem Neuladen des Modules jedoch wieder verloren (/proc Filesystem!). Eine Anpassung der Werte ist im Normalbetrieb nicht notwendig. Im Störungs- und Testfällen kann es jedoch hilfreich sein zB. den Load oder den Traffic auf dem System zu limitieren.

5.1 Load AVG

- Wert in der Datei unter „*./proc/sys/mars/loadavg_limit*“
- Parameter mit dem maximalen Load-AVG des lokalen Node
- bei Erreichen dieses Wertes wird ein disconnect der Ressource durchgeführt

5.2 Network-Traffic-Limit

- Anzeige für das Limit für den ausgehenden Netzwerk-Traffic insgesamt

5.3 Server-IO-Limit

- Anzeige für das Limit für das eingehenden Netzwerk-Traffic insgesamt

5.3 Memory-Limit

- Anzeige für das Limit des zur Verfügung stehenden Speichers

5.4 Free-Space-Limit

- Parameter mit dem maximalen Füllstand des /mars Devices

5.5 Free-Space-Limit Auto-Log-Delete

- Wie vor, jedoch Speicherplatz bis ein automatisches Log-Delete durchgeführt wird

5.6 Free-Space-Limit Auto-Log-Rotate

- Maximale Größe die ein Transaktionslogfile haben darf, bis ein automatisches Log-Rotate durchgeführt wird

5.7 Sync-Limit

- Einstellung für den gleichtigen Sync unterschiedlicher Ressourcen

6. Debug

In der aktuellen Mars-Version sind folgende Debug-Optionen nutzbar:

MARS-STATUS –debug	= Auswertung der Dateien *„/proc/sys/errors“* und *„…/warnings*“
	= Diese Datei muss angelegt werden (touch /mars/log.txt) und enthält Debug-Informationen des Mars Modules. Die Datei kann sehr groß werden!

7. Beispiele für den laufenden Betrieb

Für die folgenden Beispiele wird eine MARS-CLUSTER mit zwei Nodes angenommen. Als Datenvolumen wird der Name DATA verwendet, dass als Device /dev/sdb1 verfügbar ist.

→ Primary erstellen und starten:
marsadm create-cluster
modprobe mars
marsadm create-ressource dara /dev/sdb1

→ Ressource verwenden:
mkfs.xfs /dev/mars/data
mount /dev/mars/data /mnt/data
Dienste/Anwednungen starten

→ Secondary einrichten und starten:
marsadm jon-cluster primary_hostname
modprobe mars
marsadm join-ressource data /dev/sdb1

→ Primary / Secondary umschalten:
* auf dem Primary System:
Dienste/Anwednungen stoppen
umount /dev/mars/data
* auf dem Secondary System:
marsadm primary data
mount /dev/mars/data /mnt/data
Dienste/Anwednungen starten

→ Logfiles rotieren und löschen:
marsadm log-rotate data
marsadm log-delete-all data

→ Resize von Devices

1) auf allen Nodes mit des Clusters das Device für die Ressource vergrößeren
(zB. lvresize -L+100G /dev/vg00/ressource)
→ mars-status zeigt jetzt für alle Nodes an das ein Resize möglich ist

2) auf allen Nodes des Clusters laufende syncs abschliessen und dann stoppen
marsadm pause-sync $resssource

3) auf dem Primary des Clusters das Resize starten
marsadm resize $resssource
→ jetzt wird der Sync des Devices gestartet
→ mars-status zeigt jetzt den Sync auf allen Nodes an

4) das Filesystem des Devices anpassen
growxfs /dev/mars/$ressource

8. Betriebs-(un)-fälle

8.1 Mars-Systemdirectory voll – Emergency Modus

Im Betrieb kann es vorkommen, dass das Mars-Systemdirectory (/mars) voll wird. Dies wird in der Praxis beispielsweise durch zu große oder nicht gelöschte Transaktionslogfiles vorkommen.

Für das kontinuierliche rotieren und löschen der Transaktionslogfiles sind hier entsprechende Cron-Jobs auf dem Node einzurichten. Sollte das Mars-Systemdirectory trotzdem „überlaufen“, schaltet MARS in den Notbetrieb (Emergency Modus). Dieser Zustand wird im MARS-STATUS angezeigt. Auf dem Primary Node werden dabei folgende Aktionen ausgeführt:

1. es wird versucht Logfiles zu rotieren und zu löschen um wieder Speicherplatz zur Verfügung zu haben

2. sofern 1. immer noch nicht ausreicht, schaltet der Primary auf „Durchgang“. Es werden keine Transaktionslogfiles mehr geschrieben, die Daten werden direkt auf das Daten-Device geschrieben. Weiterhin werden die Secondary Nodes als „*invalidate*“ gekennzeichnet.

Um diesen Fall wieder aufzulösen, ist im ersten Schritt der Primary Node wieder in einen Zustand zu bringen, mit dem wieder Transaktionslogfiles geschrieben werden können. Dazu ist als erstes das System wieder mit ausreichend Platz zu versehen. Danach ist der Primary Node kurzzeitig in den Secondary Modus zu schalten.

Im nächsten Schritt sind die Secondary Nodes wieder in Betrieb zu setzen. Hierfür ist ein Sync-Prozess zu starten.

→ siehe 3.5 Ressource invalidate / 3.3 Ressource pause-sync / resume-sync

8.2 Transaktionslogfiles

Im normalen Betrieb werden die Transaktionslogfiles vom Primary Node auf den Secondary Node übertragen. Sollten hier – oder auf anderen Wegen – Fehler entstehen, haben die Transaktionslogfiles unterschiedliche Prüfsummen. Die Versionen der Transaktionslogfiles, deren Prüfsummen und Größen werden mit MARS-STATUS –history angezeigt und ausgwertet.

Weiterhin kann es bei Problemfällen vorkommen, dass nicht alle benötigten Transaktionslogfiles in den entsprechenden Versionen auf den Secondary Nodes verfügbar sind. Dies kann zum Beispiel eintreten wenn der Primary Node ausgefallen und wieder in Betrieb genommen wurde, oder wenn auf dem Primary Node das Mars-Systemdirectory zugelaufen ist.

In den genannten Fällen kann ein geregelter Betrieb durch ein Kommando „invalidate“ und den nachfolgenden Sync wieder hergestellt werden. Entsprechend Ausfallzeit und Datenbestand ist die benötigte Zeit nicht sehr hoch (relativ), da ja ein Fast-Full-Sync verwendet wird.

8.3 Split-Brain

Unter bestimmten – ungewollten – Umständen kann es vorkommen das ein Split-Brain Zustand entsteht. Im Fall von MARS bedeutet dies, dass unterschiedliche Prüfsummen eines Transaktionslogfiles vorhanden sind. In der Praxis kann dies beispielsweise (siehe oben) vorkommen, wenn Fehler auf dem Primary Node auftreten.

Abhilfe kann hier nur geschaffen werden, in dem von „Hand“ der richtige Node auf Primary ausgewählt und gesetzt wird. Alle anderen (Secondary) Nodes müssen nachfolgend neu gesynct werden.

8.4 IP-Adressen

MARS geht im normalen Betrieb davon aus, dass die default IP-Adresse die erste IP-Adresse auf dem

Interface eth0 des Systems ist (ip addr sh dev eth0 | head -n 1). In bestimmten Setups ist dies jedoch nicht umsetzbar, da zB. anderen Interfaces verwendet werden sollen. Abhilfe schafft hier der Einsatz der Option „--ip=x.x.x.x“ bei Verwendung des Usertools MARSADM vorhanden.

Beispiel: ???

8.5 Cluster-Verbindungen

Die einzelnen passiven MARS Nodes (Secondarys) eines Clusters verbinden sich im normalen Betrieb immer mit dem aktiven Primary des Clusters. Dies trifft für den Sync des Datenbestandes und die Übertragung der Transaktionslogfiles zu. In bestimmten Fällen – zB. wenn sich mehrere Secondary-Nodes in der gleichen Lokation befinden – möchte man von diesem Schema jedoch abweichen.

MARS bietet dazu die Möglichkeit mehrere Nodes mit entsprechender Priorität zu hinterlegen. Sofern einer dieser Nodes nicht mehr erreichbar ist, wird von MARS auf den nächsten verfügbaren Node umgeschaltet.

Um den normalen Verbindungszustand zu verändern, ist auf den betreffenden Nodes der Connect-Link anzupassen. Der Connect-Link muss – mit Kommas getrennt – auf die entsprechenden Nodes angepasst werden.

Beispiel:

lrwxrwxrwx 1 root root 39 Mar 19 07:45 connect-webfiless1ba01 -> webfiless1bs01,webfiless1bs02
lrwxrwxrwx 1 root root 39 Mar 19 07:45 connect-webfiless1ba02 -> webfiless1ba01,webfiless1bs01
lrwxrwxrwx 1 root root 41 Mar 19 07:45 connect-webfiless1bs01 -> webfiless1ba01,webfiless1ba02
lrwxrwxrwx 1 root root 40 Mar 19 07:45 connect-webfiless1bs02 -> webfiless1bs01,webfiless1ba01

→ Bei Veränderung der Connect-Links ist zu beachten,dass die passende Reihenfolge auf allen Nodes gleichzeitig einzurichten ist. Es kann sonst vorkommen, dass eine Verbindung nicht mehr möglich ist oder das eine Übertragung der Daten im „Kreis“ (also gar nicht mehr!) erfolgt.

8.6 Notfall

Sollte es im Betrieb Störungen oder zu Notfällen kommen, ist es möglich auf das Daten-Volume - das von MARS genutzt wird – direkt zu zugreifen. Dazu ist das MARS Module zu entladen. Folgend kann das Daten-Volume gemountet werden.

→ Es ist bei der direkten Verwendungen des Daten-Volumes zu beachten, dass Zugriffe nicht mehr in den Transaktions-Logfiles gespeichert werden. Die MARS Volumes sind damit invalide und müssen neu gesynct werden.

8.7 Monitoring

Für das Monitoring des MARS Clusters steht ein XML-Template mit den entsprechenden Cron-Jobs und Konfigurationsdateien für Zabbix auf github zur Verfügung.

8.8 Störungen

In allen Fällen in den man unsicher über den Status der Nodes ist hilft es einen neuen Sync anzuschieben. Dazu wird auf den Secondarys das Kommando „*marsadm invalidate $ressource*“ aufrufen.

→ siehe 3.5 Ressource invalidate

9. Übersetzungen

Die folgenden Übersetzungen beziehen sich auf den Zusammenhang mit Mars.

Cluster	= Verbund aller Systeme, unabhängig ob Ressourcen verbunden oder gejoint werden.
Node	= einzelnes System das mit dem Cluster verbunden ist
Primary	= aktiver Node
Ressource	= Daten-Device auf die zu replizierenden Daten liegen
Sync-Daten	= Block-Daten die zum Sync einer Ressource notwendig sind
Secondary	= passiver Node
Transaktionslogfile	= Logfile, in dem die einzelnen Änderungen der Ressource hinterlegt sind
Version	= Nummer der fortlaufenden Transaktionslogfiles
Mars-Systemdirectory	= Directory auf dem Mars-Status-Informationen und Logfiles abgelegt werden typischer Weise „/mars“
Mars-Device	= Device das als Ressource von MARS bereit gestellt wird. typischer Weise „/dev/mars/resourcenname“
Daten-Device	= Datenvolumen das „unter“ dem Mars-devices liegt Typischer Weise „/dev/volumengroup/lvm-name“