Kopfschmerzen mit ZFS

Roman Gächter
Trivadis AG
Schweiz

Was sie von Oracle über ZFS nicht hören werden

Wenn sich ein Solaris Sysadmin an die Worte erinnert „don't touch a running system" ist es meistens schon zu spät. Eine einfache Umkehr nicht immer möglich. Der Autor hat routinemässig ein Solaris „CPU OS Patchset" eingespielt und ist danach in Teufels Küche geraten.

Einleitung

In diesem Artikel beschreibt der Autor die Erfahrungen, welche er bei einem Kunden mit dem ZFS „Zetabyte File System" gemacht hat. ZFS wurde von Sun Microsystems für Solaris entwickelt und gehört heute Oracle. Der Kunde muss anonym bleiben und kann nicht erwähnt werden. ZFS ist unbestritten ein außerordentlich gutes Produkt. Eine der innovativsten Erfindungen in diesem Bereich der letzten Jahre. Auch wenn das Aufsetzen von ZFS aus der Sicht des Administrators unglaublich leicht von Statten geht darf nicht vergessen werden, dass in einem Enterprise Datenbank Umfeld zusätzliches Tuning notwendig ist und man von Anfang an ein gutes Konzept für den Storage Layout ausarbeiten muss.

Situation

Eine Bankenplattform, die seit knapp einem Jahr in Betrieb war und problemlos lief, bekundete 3 Wochen nach dem einspielen des Solaris „CPU OS Patchset" vom Januar 2012 massive Performance Probleme. Diese führten zu Verbindungsabbrüchen zum SWIFT Netzwerk, dem „worst case" für das Banken Business. SWIFT steht für „Society for Worldwide Interbank Financial Telecommunication" und ist eine Plattform, über welche Banken untereinander Finanztransaktionen abwickeln können.

Die einzige Änderung, welche am System vorgenommen wurde, war der Solaris Upgrade. Für das Business, den Software Lieferanten und das Management war somit die Ursache des Problems gefunden. Nun wurde der Ball den Solaris System Administratoren zugespielt, welche gewaltig unter Druck gerieten das Performance Problem zu lösen.

System Architektur

Im folgenden wird die System Architektur grob aufgezeigt.

Verteilt auf zwei Rechenzentren ist die Plattform redundant aufgebaut. Es handelt sich um mehrere SPARC Enterpise M4000 Server auf denen Solaris 10 installiert ist. Es wird die Solaris Virtualisierungs-Lösung mit Solaris Zonen genutzt. Die verschiedenen Systeme der Applikationen laufen alle in „Solaris Containern".

Die Daten der Bankenapplikationen befinden sich in einer Oracle 11G RDBMS. Die gesamte Oracle DB Installation ist in einem einzelnen Zpool konzentriert. Dieser Zpool wird mit der „SNDR" Replikations-Lösung von Solaris synchron in das andere Rechenzentrum repliziert.

(Es ist bekannt, dass sich für solche Zwecke Oracle Dataguard besser eignen würde, die Lösung mit SNDR wurde aber vom Softwarehersteller empfohlen und vom Kunden gewünscht).

Die Daten liegen auf einem SAN. Es wird ausschließlich das „Zetabyte File System“ ZFS verwendet.

Die zu replizierende Datenmenge ist relativ klein (50GB). Die Verteilung zwischen Schreib- und Leseoperationen auf der DB ist ausgeglichen. Die Antwortzeiten der Applikationen für die Benutzer verhalten sich in etwa linear zu den Antwortzeiten der DB.

Die Systeme sind in Bezug auf CPU nur wenig ausgelastet. Aufgrund der synchronen Replikation war der Flaschenhals früher die Netzwerkverbindung zwischen den Rechenzentren.

Eingrenzung des Problems

Es kristallisierte sich schnell heraus, dass der Engpass auf der I/O Seite der Oracle DB lag. Die aufgezeichneten „System Activity Report“ Daten (5 Minuten Mittelwerte) zeigten zum Teil 100% busy Werte auf dem Devicefiles des Oracle Zpool’s und entsprechend auf dem Replikations-Device von SNDR. Obwohl der gesamte Datenduchsatz bescheiden war, zeigten die Oracle „AWR“ Reports sehr schlechte Antwortzeiten (AVERAGE WAIT) – zum Teil über 50ms. Auch mit iostat (Messintervall 1 Sekunde) beobachteten wir viele hohe durchschnittliche Antwortzeiten. (Werte von „asvc_t“ – „average service time of active transactions in milliesconds“ – von über 50). Weil für die Plattform eine dedizierte Netzwerkverbindung zwischen den Rechenzentren verfügbar war und die gemessenen Bandbreiten nur einen Teil der möglichen Kapazität ausmachten, war ein Problem mit der synchronen Replizierung unwahrscheinlich. Man konnte die weitere Analyse also auf Oracle und das ZFS fokussieren.

Übersicht ZFS

Auszug aus Wikipedia:

> ZFS ist ein von Sun Microsystems entwickeltes transaktionales Dateisystem, welches zahlreiche Erweiterungen für die Verwendung im Server- und Rechenzentrumsbereich enthält. Hierzu zählen die enorme maximale Dateisystemgrösse, eine einfache Verwaltung selbst komplexer Konfigurationen, die integrierte RAID-Funktionalität, das Volume-Management sowie der prüfsummenbasierte Schutz vor Datenübertragungsfehlern.

Die Vorteile von ZFS sind unbestritten und der Autor möchte sie nicht mehr missen. Insbesondere die einfache Administration, die optimale Integration mit Solaris und die eingebaute Funktionalität von Snapshots sind von grossem Nutzen.

Man sollte sich den diversen ZFS Filesystem Chaches bewusst sein. Diese können durch geschickte Konfiguration und Nutzung von schnellen Devices wie „solide state disks“ die Performance erheblich verbessern. Sie können sich aber auch kontraproduktiv auswirken:

- Memory Verbrauch des ARC Caches kann andere Applikationen behindern
- Verdoppelung der Schreiboperationen für „synchrone writes“, ZIL Log
- Abbremsen von Datenbanken durch „file level prefetching“ und „device level read ahead“ Mechanismen beim Lesen

Im folgenden eine Liste von ZFS Caches:

- Der “first level cache” (ARC cache) befindet sich im Memory
    - Es handelt sich um eine Variante des ARC Algorithmus (Adaptive Replacement Cache)
- Optional können “second level disk caches” definiert warden
    - Dafür eignen sich schnelle Disks wie SSD
    - Der “read cache” wird als L2ARC bezeichnet
        - Wird über das zpool property cachefile aktiviert
        - Wird beeinflusst über das ZFS property scondarycache (all | none | metadata)
    - Der “write cache” wird als ZIL (ZFS Intent Log) bezeichnet
        - befriedigt POSIX Requirements for synchrone Transaktionen
        - wenn kein separates ZIL Device definiert wird ist der ZIL ein Teil des Zpool’s
    - Mit “zpool status” warden die “log devices” angezeigt
    - Die “second level” Chaches kann man einfach während des Betrieb hinzufügen, konfigurieren oder entfernen

ZFS und Oracle

In den Anfangszeiten von ZFS gab es Statements von Sun Microsystems, welche darauf hinausliefen ZFS nicht für Oracle RDBMS zu verwenden. Dies hat sich schon seit einiger Zeit geändert. Heute ist ZFS offiziell von Oracle für RDBMS supportet und wird auch empfohlen. Wichtig ist jedoch, das ZFS nach „best practice“ aufzusetzen. Oracle hat diverse Whitepapers dazu verfasst – und diese sollten auch berücksichtigt werden. Die wichtigsten Punkte sind:

- ZFS „record size“ an „db block size“ anpassen
- Überwachen des „ARC cache“ im Memory und wenn nötig begrenzen
- ZIL (ZFS intend log) für „Oracle data files“ umgehen
- Überwachen des ZFS Füllgrades (Usage), immer mindestens unter 80% bleiben oder besser noch tiefer
- Für Oracle RDBMS wenn möglich immer dedizierte Zpool’s für „redo logfiles“, „data files“ und „archivelog files“ verwenden, mit dedizierten SAN LUN’s
- Für Zpool’s nur ganze LUN’s verwenden, keine Partitionen einer LUN

Analyse von ZFS Performance Problemen

Wie bereits erwähnt startet man hier am besten damit, die „best practice“ Whitepapers von Oracle zu studieren und zu untersuchen, ob die eigene Installation davon abweicht.

Das beste Solaris Bordmittel für die Analyse der I/O Performance ist iostat. Man sollte ein Scripting aufsetzen, welches die „extended“ iostat Werte im 1 Sekunden Intervall rund um die Uhr aufzeichnet. Zudem sollte man sich aufzeichnen lassen, wie viel Memory sich der ZFS ARC Cache reserviert und

ob das Memory auch schnell wieder freigegeben wird sofern anderweitig gebraucht. Man bewegt sich auf der sicheren Seite, wenn man den ARC Cache beschränkt.

Befehl um sich den ARC Cache anzeigen zu lassen:

```
root# echo "::memstat" | mdb -k
```

Wichtige Informationen gewinnt man durch Tracen mit „dtrace“. Das im Solaris Betriebssystem eingebaute Dtrace (Dynamic Tracing) ist ein sehr mächtiges Tool. Es bietet die Möglichkeit in laufenden Prozessen den Arbeitsspeicher, die Prozessorzeit, das Dateisystem und Netzwerkressourcen zu untersuchen. Was im Zusammenhang mit ZFS Performance von Dtrace interessiert:

- Welche Programme verursachen die „top writes“, „top reads“?
- Wie sieht die „byte size“ Verteilung aus?
- Sind „ganging“ Operationen zu beobachten? Dies wäre ein Zeichen von fragmentierten Zpool’s.

In der Oracle RDBMS kommt man nicht darum herum AWR Reports – AWR steht für „Automatic Workload Repository“ - zu generieren, um aussagekräftige Bewertungen zur I/O Performance machen zu können.

Bei ZFS spricht man von „ganging“, wenn die Daten nicht mehr an einem zusammenhängenden Platz im ZFS geschrieben werden können und kleinere Gaps verwendet werden müssen. Die Anzahl der „ganging“ Operationen beim Schreiben ist linear zum Fragmentierungsgrad eines Zpool’s. Besteht der Verdacht auf eine Fragmentierung in einem Zpool sollte man das entsprechende Tracing durchführen.

Mit der folgenden Dtrace Syntax kann man sich die „ganging“ Operationen in einem System anzeigen lassen:

```
dtrace -qn 'fbt::zio_gang_tree_issue:entry { @[pid]=count();  }' -c "sleep 300"
```

Problemlösung

Schritt 1:

Wir mussten feststellen, dass sich die über die Netzwerkverbindung zum anderen Rechenzentrum replizierte Datenmenge nach dem Solaris Update sprunghaft, fast um den Faktor zwei erhöht hatte. Die Abbildung 1 zeigt den Replikations-Durchsatz auf der Leitung zwischen den Rechenzentren. Grüne Kurve „inbound“, blaue Kurve „outbound“. Vor und nach dem Upgrade wurden Failover Test durchgeführt und jeweils eine volle Replikation (Spitzen) gefahren. Nach dem Upgrade liefen die Applikationen auf der anderen Seite also unter der blauen Kurve, die Richtung der Replikation hatte sich geändert. Durch die Kernel Patches 147440-10 und 144500-19 des Januar Patch Bundle wurden im ZFS neue Properties eingeführt. Unter anderem das Property „logbias“. Für Datenbank-Files sollte dieses auf „throughput“ gesetzt werden, war jedoch nach der Patch Installation auf dem Default Wert „latency“. Throughput bedeutet: Der ZFS Intend Log wird für synchrones Schreiben nicht verwendet,

womit sich die „writes“ quasi um die Hälfte reduzieren. (In Oracle Solaris 10 10/08 bis 10/09 Installationen wurde ein ähnliches Verhalten durch das Setzen des Kernel Parameter im /etc/system

„set zfs:zfs_immediate_write_sz=8000“ erreicht. (Dieser Kernel Parameter war bei uns jedoch nicht gesetzt). Nach der Änderung von „logbias“ auf „througput“ im ZFS mit den Oracle Daten-Files hat sich bei uns auch tatsächlich die replizierte Bandbreite wieder auf den normalen Wert eingestellt. Leider war das Performance Problem damit aber noch nicht gelöst.

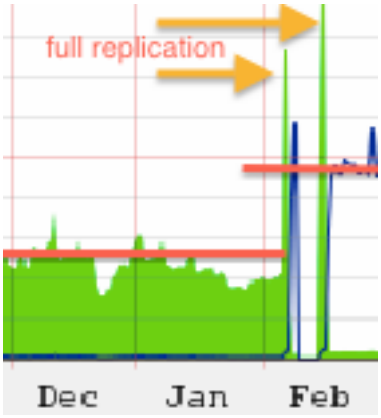

Abb. 1: Replikationsdurchsatz

Schritt 2:

Wir haben festgestellt, dass das ZFS mit den „redolog files“ fälschlicherweise auf eine „recordsize“ von 8k anstatt 128k eingestellt war. Zudem waren die Messungen der „bitesize“ Verteilung im Oracle Zpool anders als erwartet. Wir erwarteten die grösste Verteilung bei 8k, entsprechend zur „recordsize“ des ZFS mit den Oracle „datebase files“.Die Tabelle 2 zeigt aber ein ganz anderes Bild. Gemessen wurde mit dem Dtrace Toolkit Programm „bitesize.d“.

```
6. May
     371  zpool-Oracle_R\0
           value  ------------- Distribution ------------- count
             256 |                                         0
             512 |@@@                                      4498
            1024 |@@@@@                                    7236
            2048 |@@@@@@@@@@                   14340
            4096 |@@@@@@                                   8388
            8192 |@@@@@                                    7461
           16384 |@@@@@                                    7710
           32768 |@@@@@                                    7068
           65536 |                                         425
          131072 |                                         40
          262144 |                                         0
```

Tabelle 2 bytesize Verteilung am 6. Mai

Wir haben also den „recordsize“ Wert im „redolog file“ ZFS auf 128k geändert und mussten danach in einem Servicefenster den ganzen Oracle Zpool sichern und restoren, um diese Änderung wirksam zu machen. Danach kam das grosse Aufatmen! Die Performance bewegte sich wieder in einem Bereich wie vor dem Upgrade. (AWR Werte für „AVERAGE WAIT„ von 14 ms). Nun sahen auch die Resultate der Bitesize Verteilung besser aus. Zu sehen in Tabelle 3.

```
11. May
    371  zpool-Oracle_R\0
           value  ------------- Distribution ------------- count
             256 |                                         0
             512 |@                                        119079
            1024 |@                                        146371
            2048 |@@                                       311663
            4096 |@@@@@                                    886102
            8192 |@@@@@@@@@@@@@@@@    2623578
           16384 |@@@@@@                                   1076413
           32768 |@@@@@@                                   1042969
           65536 |@@                                       319642
          131072 |@@                                       380511
          262144 |                                         0
```

Tabelle 3 bytesize Verteilung 11. Mai

Leider erlebten wir schon bald eine bittere Enttäuschung, wir hatten die Lösung immer noch nicht gefunden. 3 Wochen später war das Performance Problem zurückgekehrt. Wir kannten nun den „work-around“: Servicefenster beantragen, Datensicherung des Oracle Zpool’s und restore der Daten. Das gab uns wieder für 3 Wochen Ruhe.

In der Abbildungen 4 sieht man wie sich die Performance Werte innerhalb von 2 Wochen verschlechtert haben. Die Graphik zeigt die Werte für „AVERAGE WAIT (ms) for LOG FILE SYNC (time to wait before writing into RedoLog files)“

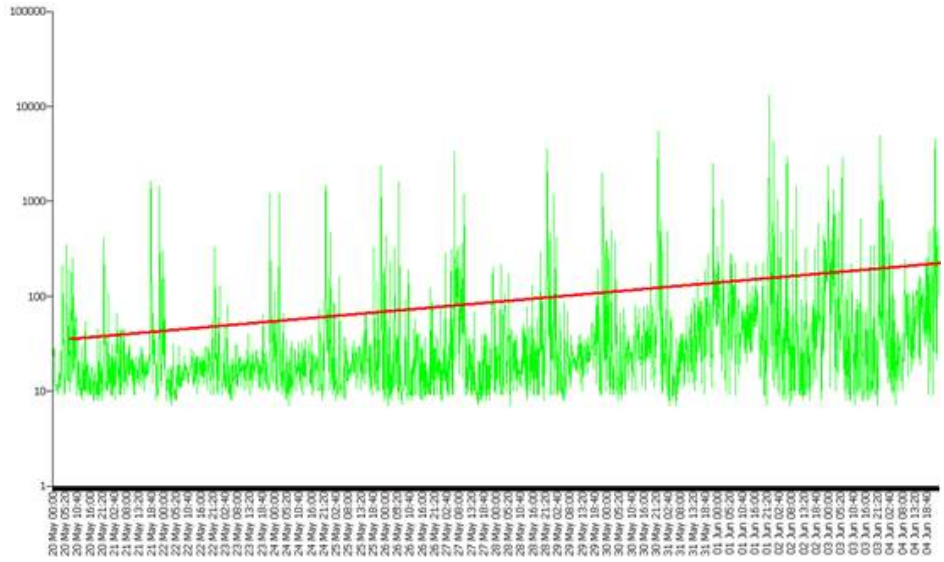

Abb. 4 „AVERAGE WAIT (ms) for LOG FILE SYNC

Schritt 3:

Ein Oracle Consultant, der bei uns vor Ort war erklärte, dass unser Performance Problem durch die fortschreitende ZFS Fragmentierung des Oracle Zpool's verursacht wird. Verstärkt wird das Problem, weil der Storage Layout nicht den „best practice" Richtlinien folgt und keine dedizierten zpool's verwendet werden für eine Separierung von „redolog files" und „database files".

Bei unserer Umgebung wird wegen der SNDR Replikation absichtlich nur ein Oracle Zpool verwendet. (Die Dauer einer vollständigen SNDR Replikation ist abhängig von der Anzahl und der Grösse der zu replizierenden Zpool's).

ZFS überschreibt nie Daten sondern kennt das „copy on write" Prinzip. Dies ist optimale für die Datenintegrität und auch notwendig damit Techniken wie „snapshots, cloning, shadow copy und zfs send / recieve" überhaupt funktionieren. Leider handelt sich ZFS dabei das Fragmentierungsproblem ein sobald in einem Zpool regelmässig geschrieben wird.

Die Empfehlung von Oracle war:

Storage Layout ändern und trennen respektive verteilen von Oracle „redolog files" und „database files" auf mehrere Zpool's mit dedizierten LUN's.

Diese Umorganisation haben wir in die mittelfristige Planung aufgenommen.

Schritt 4:

Durch Benchmarks auf unseren Test Systemen und „ganging" Messungen mit Dtrace haben wir festgestellt, dass sich die Fragmentierungsproblematik in unserer Umgebung durch die Erhöhung des ZFS „freespaces" entschärfen liess.

Wir haben also zuerst mal die Oracle Administratoren gebeten die Daten im Oracle Zpool so weit wie möglich zu reduzieren. Dadurch konnten wir die Zeitspanne auf 2 Monate verlängern bis eine neue Defragmentierungsaktion mit Service Fenster notwendig wurde.

Mit einer Usage von 82% im Oracle Zpool dauerte es 3 Wochen bis die Fragmentierung die Performance beeinträchtigte, mit 67% Usage 8 Wochen.

Die Abbildung 5 zeigt sehr schön die Auswirkung der schleichenden Fragmentierung. Erster Knick am 6 Juni durch Defragmentierung, zweiter Knick am 13. Juni durch Defragmentierung und mehr freien Platz im Zpool.

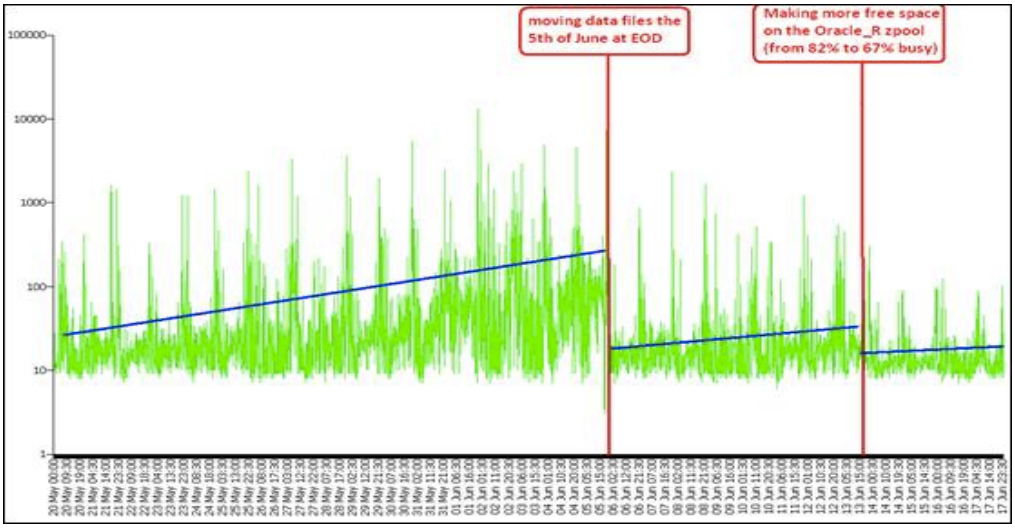

Abb. 5 schleichende Fragmentierung

Die Abbildung 6 zeigt die „ganging“ Operationen vor (rote Kurve) und nach einer Defragmentierung des Oracle Zpool’s.

Abb. 6 „ganging“ Operationen

Schritt 5:

Nun entschlossen wir uns, den Oracle Zpool in dem Masse zu vergrössern, dass die Zeit für eine volle Replikation gerade noch akzeptabel war. Wir erhöhten den Freespace auf 50%. Mit dieser Aktion konnten wir das Fragmentierungsproblem lösen.

Einerseits sind keine „ganging“ Operationen mehr zu beobachten andrerseits hat sich auch die Perfomance massiv verbessert. Wir haben nun AWR Werte „AVERAGE WAIT“ von 6ms.

Die Abbildung 7 zeigt die Entwicklung der „average wait times“:

Links bis zum 13.6 mit einer ZFS Usage von 80%, in der Mitte nach dem Löschen von Daten und „file relocation“ und ZFS Usage von 67% und ganz rechts nach einer weiteren „file relocation“ und nun nur noch mit einer Usage von 50%.

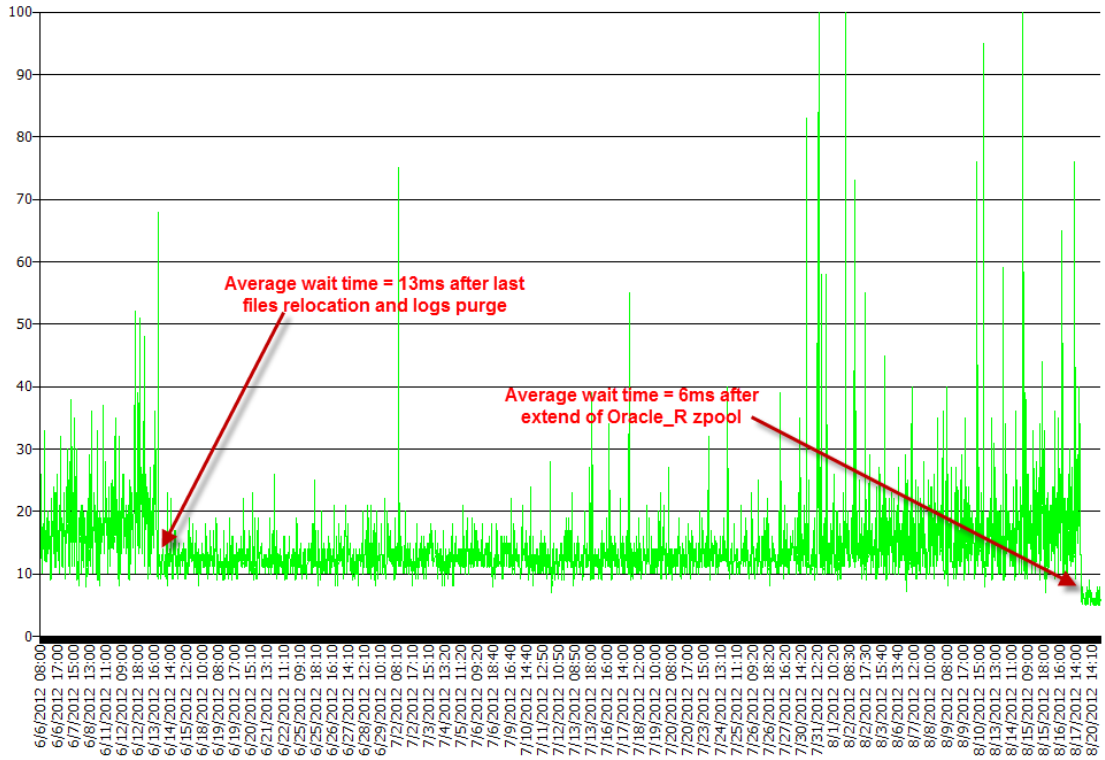

Abb. 7 Entwicklung der „average wait times“

Fazit

Die Gretchenfrage, welche sich stellt: Hatten unsere Performance Probleme einen Zusammenhang mit den applizierten Solaris Patches oder entstanden sie durch eine langsame Reduktion des freien Platzes im Zpool?

Gemäss unseren Aufzeichnungen hat sich der Füllgrad des Zpool’s nur minimal verändert. Die Vermutung liegt nahe, dass mit den Patches 147440-10 und 144500-19 Änderungen im ZFS

vorgenommen wurden, welche die Fragmentierungsproblematik in unserer spezifischen Umgebung akzuentiert haben. Im Nachhinein betrachtet ist die Lösung unseres ZFS Performance Problems simpel. „Reduktion des Füllgrades von ZFS von 80% auf unter 50%“. Damit hat man die vorher immer wiederkehrende Fragmentierung des Oracle Zpools, verbunden mit Performance Problemen in der spezifischen Umgebung gänzlich eliminiert und macht damit regelmässige Defragmentierungs-Aktionen mit Servicefenstern unnötig. Leider mussten wir die Lösung selber finden. Wir eröffneten mehrere SR’s beim Oracle Support – kein Wort über ZFS Fragmentierung. Erst ein Oracle Consultant vor Ort nahm das unschöne Wort in den Mund. Des weiteren muss man sehr lange suchen, um in der Oracle Dokumentation die Fragmentierungsproblematik zu finden – in den ZFS Manuals herrscht hier das grosse Schweigen. Schön wäre es, wenn Oracle ein Tool bereitstellen könnte, welches ähnlich wie ein ZFS „Scrubing“ im Hintergrund laufen könnte - im on-line Betrieb - und den Zpool automatisch defragmentieren würde.

Quellen

Oracle ZFS Whitepaper:

http://www.oracle.com/technetwork/server-storage/solaris/config-solaris-zfs-wp-167894.pdf

Dtrace:

http://www.brendangregg.com/dtrace.html

ZFS Fragmentation:

http://wildness.espix.org/index.php?post/2011/06/09/ZFS-Fragmentation-issue-examining-the-ZIL

AWR:

http://www.oracle-base.com/articles/10g/automatic-workload-repository-10g.php

Kontaktadresse:

Roman

Gächter

Europastrasse 5

CH-8152 Glattbrugg

Telefon:	+41 44 808 70 20
Fax:	+41 44 808 70 21
E-Mail	roman.gaechter@trivadis.com
Internet:	www.trivadis.com