Monitoring von Oracle-Datenbank-Servern mit Nagios

Gerhard Laußer, ConSol* Software GmbH

Nagios ist das bekannteste und am weitesten verbreitete Open-Source-Monitoring-Tool. Längst dient es nicht mehr nur der Überwachung von Netzwerk-Infrastrukturen und Servern, sondern auch von Web-Applikationen, SAP, Virtualisierung, RZ-Infrastruktur und natürlich Datenbanken. Dank seiner flexiblen Erweiterbarkeit durch Plug-ins sind Nagios im Grunde keine Grenzen gesetzt. Jedes Gerät, jede Software, deren Zustand mithilfe dieser Scripts abgefragt werden kann, lässt sich so ins Monitoring einbinden. In diesem Artikel geht es natürlich um das Thema „Oracle-Datenbanken".

Nagios allein ist lediglich ein Scheduler, der mithilfe anderer Programme die Verfügbarkeit von Servern und Diensten ermittelt, über die Fehlerzustände Buch führt und gegebenenfalls Alarme verschickt. Besagte Programme können beliebige Skripte sein, die ermitteln, ob ein überwachtes System funktioniert oder ausgefallen ist (beziehungsweise im Begriff ist, auszufallen). Mittels definierter Exitcodes („0 = OK", „1 = WARNING", „2 = CRITICAL", „3 = UNKNOWN") wird diese Information an Nagios übergeben.

Zum Check-Resultat gehört auch ein erläuternder Text beziehungsweise eine Fehlermeldung, die in der Nagios-Web-GUI dann grün, gelb oder rot unterlegt ist. Neben der reinen Alarmierung können mit Nagios auch Messwerte, die bei jedem Check anfallen, gespeichert und als Langzeit-Graphen gezeichnet werden. „Dashboards" und „Reporting" sind weitere Themen, für die es Add-ons gibt.

Vor drei Jahren schuf eine Gruppe von Monitoring-Consultants die „Open Monitoring Distribution" (OMD). Das ist eine Sammlung von Nagios und den nützlichsten Tools dazu, zusammengefasst zu einem einzigen Installationspaket.

Nagios-Konfiguration und -Philosophie

In den Konfigurationsdateien von Nagios unterscheidet man mehrere Typen von Objekten. Zunächst gibt es das Host-Objekt, das eine IP-Adresse und einen Namen besitzt. Dies kann ein physikalischer oder virtueller Server sein oder aber auch eine virtuelle Cluster-Adresse. Dem Host zugeordnet sind Service-Objekte. Diese können beliebige Zustände abfragen, etwa ob die Hardware des Servers fehlerfrei funktioniert, wie viel Speicher zur Verfügung steht, binnen wie vieler Sekunden eine Webseite geladen wird und ob im Inhalt ein bestimmtes Schlüsselwort vorkommt, welche Temperatur im Inneren eines Storage-Schranks herrscht etc. Hier sind der Phantasie keine Grenzen gesetzt. Dies ist möglich, weil Nagios die Ermittlung der Zustände an die Plug-ins delegiert.

Bei OMD sind bereits zahlreiche Plug-ins mitgeliefert, unter anderem zur Überwachung von Basis-IT, also von Unix-/Windows-Servern, Festplatten oder Netzwerk-Geräten. Weitere Anforderungen können jedoch leicht selbst implementiert werden. Plug-ins sind nichts weiter als Scripts, die definierte Werte an Nagios zur Weiterverarbeitung übergeben.

Üblicherweise wird man nicht nur den Datenbank-Server überwachen, sondern auch das darunterliegende Betriebssystem. Die gebräuchlichen Checks umfassen den Füllstand der File-Systeme, die Auslastung der Netzwerk-Interfaces, Hauptspeicher, CPU und natürlich den Zustand der Hard-

ware (Temperaturen, Raid, Stromversorgung). Dieser Artikel geht darauf nicht näher ein, da der Schwerpunkt bei der Datenbank liegt. Das Plug-in der Wahl ist hier „check_oracle_health“, das OMD beiliegt.

Bei der Definition eines Service gibt man an, in welchen zeitlichen Abständen das dazugehörige Plug-in aufgerufen werden soll und, falls ein Fehler gemeldet wird, wie oft dieser Check wiederholt werden soll, bevor ein Alarm verschickt wird. Wohin dieser geht, legt Nagios mithilfe zweier weiterer Objekt-Arten fest: „Contacts“ und „Contactgroups“. Dahinter verbergen sich entweder reale Personen beziehungsweise deren Nutzer-Accounts, aber auch anonyme Empfänger von E-Mail oder SMS. Auf diese Art kann Nagios ein Bereitschaftshandy abbilden.

Die Contacts sind Services und Hosts zugeordnet. Damit erreicht man, dass beim Login (Single Sign-on mit AD ist möglich) eines Contact an der Web-Oberfläche dieser ausschließlich die Hosts und Services zu sehen bekommt, die ihm per Konfiguration zugewiesen sind. Außerdem ist damit auch festgelegt, wer bei Ausfall eines Service benachrichtigt wird.

Die Konfiguration von Nagios basiert auf Konfigurationsdateien. Mit der Zeit sind aber auch ein paar Tools entstanden, die das Editieren der Konfiguration im Web-Browser möglich machen. In OMD ist die Web-GUI „Thruk“ dabei, mit deren Config-View bequem Objekte angelegt und gepflegt werden können.

Vorbereitungen

Nach Installation des OMD-Pakets erzeugt der Administrator mit dem Kommando „omd create oramon“ eine sogenannte „Site“. Diese besteht aus einer Laufzeit-Umgebung und dem Benutzer „oramon“, der automatisch zusammen mit einer gleichnamigen Gruppe angelegt wird. Ab hier meldet man sich als User „oramon“ an, um die Site weiter zu konfigurieren. Das Site-Konzept von OMD erlaubt parallele Nagios-Laufzeit-Umgebungen auf einem Monitoring-Server. Das ist sehr praktisch, wenn man neue Versionen erst einmal testen will, bevor man die produktive Site migriert.

```
create user nagios identified by oradbmon;
grant create session to nagios;
grant select any dictionary to nagios;
grant select on V_$SYSSTAT to nagios;
grant select on V_$INSTANCE to nagios;
grant select on V_$LOG to nagios;
grant select on SYS.DBA_DATA_FILES to nagios;
grant select on SYS.DBA_FREE_SPACE to nagios;
```

Listing 1

```
$ check_oracle_health --mode connection-time \
    --connect NAPRAX --user nagios --password oradbmon
OK - 0.11 seconds to connect as NAGIOS |
connection_time=0.1068;1;5
```

Listing 2

Im Standardumfang von OMD ist das Plug-in „check_oracle_health“, das eine ganze Reihe von Prüfungen gegen einen Datenbank-Server ausführen kann, enthalten. Diese reichen vom einfachen Connect-Check über das Ermitteln von Tablespace-Füllständen bis hin zu selbstdefinierten SQL-Statements, die applikationsspezifischen Status liefern. Dazu ist natürlich Oracle-Client-Software auf dem Nagios-Server erforderlich, da die Kommunikation zwischen „check_oracle_health“ und der Oracle-Instanz üblicherweise über ein Netzwerk stattfindet.

Falls es kein einheitliches Installationspaket gibt, reicht auch der „Instant Client“ aus (wichtig ist das „sqlplus“-Paket). Diesen entpackt man einfach im Homeverzeichnis des Users „oramon“ und setzt anschließend die nötigen Environment-Variablen. Technisch wäre es kein Problem, den Instant Client bereits als Bestandteil von OMD mitzuliefern, aus lizenzrechtlichen Gründen ist das jedoch nicht möglich. Wer vorhat, viele Systeme mit OMD auszustatten oder häufig Software-Updates durchzuführen, dem ist das Erstellen eines eigenen OMD-Database-AddOns-RPM empfohlen, in das lizenzierte Clientsoftware gepackt wird. Die Komplettinstallation besteht dann aus zwei RPM-Paketen.

Für die Überwachung eines Datenbank-Servers, die über ein simples Anpingen hinausgeht, ist es natürlich erforderlich, sich dort anzumelden und aus den internen System-Tabellen Informationen auszulesen. Dazu legt man sich am besten einen eigenen User an, der die entsprechenden Rechte erhält (siehe Listing 1). Mit diesen Privilegien ist der Benutzer „nagios“ in der Lage, alle Features von „check_oracle_health“ zu nutzen.

Die Verfügbarkeit überwachen

Als Erstes will man natürlich wissen, ob eine Oracle-Instanz überhaupt läuft. Zu diesem Zweck ruft man „check_oracle_health“ mit dem Kommandozeilen-Parameter „--mode connection-time“ auf. Der Exit-Code richtet sich danach, ob ein Login überhaupt möglich war, und wenn ja, ob der ganze Vorgang länger als eine beziehungsweise fünf Sekunden gedauert hat. Beim Überschreiten dieser Schwellwerte zeigt Nagios einen Warning- beziehungsweise Critical-Zustand an. So einen Basis-Check könnte man auch mit dem Parameter „--mode tnsping“ durchführen, jedoch ist ein echter Login wesentlich aussagekräftiger. Was nützt es, wenn der Listener antwortet, die Instanz aber nicht läuft beziehungsweise sich niemand anmelden kann?

Es wurde bereits beschrieben, wie man einen geeigneten Monitoring-User in der Datenbank anlegt. Beim Aufruf von „check_oracle_health“

```
$ check_oracle_health --mode connection-time \
    --connect 10.73.9.102:1523/orcl12 \
    --user nagios --password oradbmon
$ check_oracle_health --mode connection-time \
    --connect nagios/oradbmon@10.73.9.102:1523/orcl12
```

Listing 3

```
check_oracle_health --mode connected-users
OK - 38 connected users | connected_users=38;50;100
```

Listing 4

```
$ check_oracle_health --mode tablespace-free --name USERS
OK - tbs USERS has 99.98% free space left |
'tbs_users_free_pct'=99.98%;5:;2:
'tbs_users_free'=32761MB;1638.40:;655.36:;0;32767.98
```

Listing 5

muss man jeweils Username, Passwort und Connect String angeben (siehe Listing 2). Hier wurde dem Parameter „—connect“ der TNS-Name mitgegeben. Alternativ kann auch die Schreibweise „Easy Connect“ verwendet werden, wobei Username und Passwort Bestandteil des Connect Strings sein können (siehe Listing 3).

Bereits dieser einfache Check kann mithilfe einer Aufzeichnung und der grafischen Darstellung der gemessenen Antwortzeit eine Aussage über das Verhalten eines Datenbank-Servers im Tagesverlauf machen (siehe Abbildung 1).

Ebenfalls von Interesse kann die Anzahl der eingeloggten User sein (siehe Listing 4). In den Beispielen wurde aus Gründen der Übersichtlichkeit die Parameter „connect“, „user“ und „password“ weggelassen.

Das Beispiel trifft sicher weniger auf Datenbanken zu, die Backends von Web-Applikationen sind. Laufen aber auf zahlreichen PC-Arbeitsplätzen Fat Clients, von denen jeder eine Verbindung zur Datenbank öffnet, ist diese Messung schon sinnvoller.

Tablespaces

Vollgelaufene Tablespaces sollten mit einem Monitoring-System der Vergangenheit angehören. Hier kann man sich frühzeitig alarmieren lassen, wenn der Platz langsam knapp wird. Mit „--mode tablespace-free“ ermittelt „check_oracle_health“, wie viel Prozent noch verfügbar sind (siehe Listing 5).

Sind in diesem Beispiel weniger als fünf beziehungsweise zwei Prozent (Default) des USERS-Tablespace frei, verschickt Nagios einen Alarm an den DBA. Weil bei den heute üblichen Platten und Storage-Systemen wenige Prozent auch große Datenmengen ausmachen, kann man alternativ mit absoluten Werten arbeiten. Die Kommandozeile „--warning 12: --critical 8: --units GB“ würde also ausdrücken: „Warning“, wenn weniger als zwölf, und „Critical“, wenn weniger als acht Gigabyte freier Speicherplatz im Tablespace verfügbar sind. Lässt man den Parameter „--name“ weg, bezieht sich der Check auf sämtliche Tablespaces. Alarmiert wird dann, wenn irgendeiner von ihnen die Schwellwerte unterschreitet.

Diese Vorgehensweise spart einem zwar Tipparbeit bei der Erstellung der Nagios-Konfiguration, wird jedoch nicht empfohlen. Wird der Check etwa wegen Tablespace1 rot, so kann man es leicht übersehen, wenn kurz darauf auch ein Tablespace 2 vollläuft. Ein Statuswechsel (so wie am Anfang von grün nach rot) findet nämlich dann nicht mehr statt.

Eine sinnvolle Ergänzung zur Tablespace-Überwachung ist das Monitoring der Filesysteme oder der Volumes im Storage. Die Alarme bei Unterschreiten des verfügbaren Speichers sind wichtig. Sie sagen dem DBA, dass es Zeit wird, aufzuräumen oder für neuen Platz zu sorgen.

Genauso nützlich ist allerdings auch die Aufzeichnung und Speicherung der gemessenen Werte durch Nagios. Mithilfe des in OMD enthaltenen „AddOns PNP4Nagios“ lässt sich de-

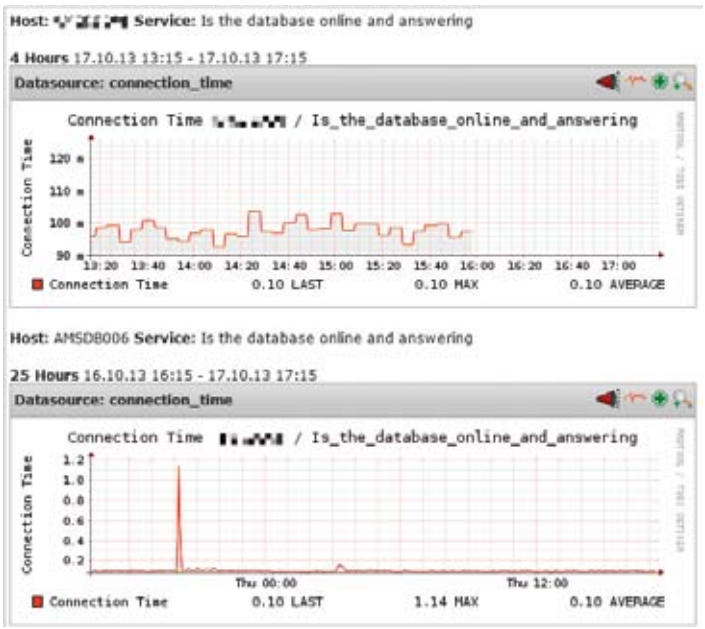

Abbildung 1: Dauer des Logins während der letzten 4 beziehungsweise 25 Stunden

```
$ check_oracle_health --mode switch-interval
OK - Last redo log file switch interval was 12 minutes. Next
interval presumably >418 minutes |
redo_log_file_switch_interval=756s;600:;60:
```

Listing 6

ren Verlauf grafisch anzeigen. So sieht man, wie sich der Platzbedarf langfristig entwickeln wird.

Performance-Monitoring

Zu jedem Monitoring gehört nicht nur die Überwachung auf produktionskritische Ausfälle, sondern auch ein Blick auf die Performance des Systems. In den letzten Versionen von Oracle hat sich ja in Sachen automatischer Optimierung von Speicher und Caches einiges getan.

Das Plug-in „check_oracle_health“ bietet in diesem Zusammenhang „--mode sga-data-buffer-hit-ratio“ an, allerdings dürfte der ermittelte Wert bei einem modernen Oracle stärkeren Schwankungen unterliegen. Er ist daher mit Vorsicht zu genießen. Eine sinnvolle Metrik ist hingegen die Länge der verstrichenen Zeit zwischen zwei Redo-Log-Switches. Folgen diese in kurzen Abständen aufeinander, deutet das auf hohe Schreiblast hin (siehe Listing 6).

Monitoring der Integrität

Der Modus „--mode corrupted-blocks“ warnt, wenn die Hardware-Infrastruktur Probleme bereitet. Innerhalb der Datenbank kann es aber auch zum Verfall kommen. Indexe und Views können beispielsweise ungültig werden, wenn sich die zugrunde liegenden Tabellen ändern. Um so etwas festzustellen, benutzt man „--mode invalid-objects“. Eine feinere Eingrenzung ist noch möglich, wenn der Parameter „--name2“ mit einem der Argumente „dba_objects“, „dba_indexes“, „dba_ind_partitions“ oder „dba_registry“ angehängt wird. Es wird dann nur auf fehlerhafte Objekte einer Kategorie geprüft.

Mit „--name <Ausdruck> --regexp“ kann man noch feiner aussieben. Es ist damit möglich, gezielt nach Objekten eines bestimmten Schemas zu suchen. Beispielsweise sorgt „--name ‚!(SYS|SYSTEM|DBSNMP|....)‘ –regexp“ dafür, dass System-Objekte ignoriert und nur solche überwacht werden, die zu Applikationen gehören.

Alertlog

Neben der Überwachung durch SQL-Befehle ist es auch wichtig, das Logfile von Oracle im Auge zu behalten. Das übernimmt das Plug-in „check_logfiles“. Diesem gibt man eine Liste von Fehler-Pattern mit, die den relevanten Meldungen entsprechen.

In regelmäßigen Abständen sorgt Nagios dafür, dass „check_logfiles“ überprüft, ob eine dieser Meldungen aufgetaucht ist. Stimmt eine Zeile mit einem der Fehler-Pattern überein, wird der DBA alarmiert. Dazu ein Auszug aus der Konfig-Datei für „check_logfiles“ (siehe Listing 7).

Die Trefferzeile aus dem Alertlog wird unverzüglich per E-Mail verschickt. In der Web-Oberfläche ist sie ebenfalls zu sehen, hier wird sie auch gleich rot (oder gelb, falls es nur eine Warnung ist) unterlegt (siehe Abbildung 2).

Auf der Webseite von „check_logfiles“ ist beschrieben, wie die Alert-

```
@searches = ({
  tag => 'oraalerts',
  logfile => '......../alert.log',
  criticalpatterns => [
      'ORA\-0*204[^\d]',        # error in reading control file
      'ORA\-0*206[^\d]',        # error in writing control file
      'ORA\-0*210[^\d]',        # cannot open control file
      'ORA\-0*257[^\d]',        # archiver is stuck
...
      'ORA\-19502[^\d]',        # write error on datafile
      'ORA\-27063[^\d]',        # number of bytes read/written is incorrect
      'ORA\-0*4031[^\d]',       # out of shared memory.
      'No space left on device',
      'Archival Error',
  ],
  warningpatterns => [
      'ORA\-0*3113[^\d]',        # end of file on communication channel
      'ORA\-0*6501[^\d]',        # PL/SQL internal error
      'Archival stopped, error occurred. Will continue retrying',
  ]
});
```

Listing 7

Datei in eine External Table eingebunden werden kann. In Umgebungen, in denen es unmöglich ist, als Nagios-Benutzer Zugriff ins Filesystem des Datenbank-Servers zu erhalten, kann „check_logfiles“ auch wie ein Client arbeiten und die Zeilen per SQL-Statements auslesen.

Applikations-Monitoring

Bisher wurde gezeigt, wie man die Grundfunktionen eines Oracle-Servers überwacht. Damit ist sichergestellt, dass er seinen Dienst erbringen kann. Eine Ebene höher sind Applikationen, die die Datenbank benutzen. Auch hier kann es zu Störungen kommen. Sofern sich diese durch SQL-Abfragen ermitteln lassen, hilft wieder „check_oracle_health“.

Mit dem Modus „sql“ werden beliebige Statements zur Ausführung gebracht. Die Bewertung, ob ein Fehler vorliegt, kann auf mehrere Arten erfolgen. Ist das Ergebnis des Statements eine Zahl, wird diese mit Warning- und Critical-Schwellwerten verglichen. Über- beziehungsweise Unterschreitung zieht einen Nagios-Alarm nach sich (siehe Listing 8).

Ist das Ergebnis ein String, sind die Parameter „--name2“ und „—regexp“ erforderlich. Das Argument des ersteren ist ein regulärer Ausdruck, der mit besagtem String verglichen wird. Bei Übereinstimmung gibt es einen Alarm.

Wenn nicht das Ergebnis eines bestimmten SQL-Statements relevant ist, sondern die Zeit, die zu seiner Ausführung benötigt wird, verwendet man „--mode sql-runtime“. Damit hat man ein mächtiges Werkzeug zur Hand, falls es häufig Beschwerden bezüglich der Antwortzeiten einer Applikation gibt. Man identifiziert die aufwändigsten oder am häufigsten aufgerufenen Datenbank-Anfragen und bindet diese ins Monitoring ein. So erhält man mithilfe von Nagios eine Aufzeichnung der Performance-Indikatoren über den ganzen Tag hinweg. Diese objektiven Zahlen sind eine gute Argumentationshilfe, wenn es „Heute ist es wieder besonders langsam“, heißt, was nicht selten nur aus einem subjektiven Eindruck heraus so empfunden wird.

Dies waren jetzt nur wenige Beispiele für Metriken, die zur Überwachung herangezogen werden können. Auf den Webseiten der Plug-ins finden sich weitere Anregungen. Sinnvolle Werte sind unter anderem „Library Cache Hitratio“, der Prozentsatz der Sort-Operationen, die im Hauptspeicher ausgeführt werden, oder „Soft Parse Ratio“. Mit Nagios, „check_oracle_health“ und „check_logfiles“ (die ja alle in OMD enthalten sind) lässt sich recht schnell ein umfassendes Monitoring für Oracle-Datenbanken aufbauen. Firmenspezifische Abfragen lassen sich ebenfalls leicht implementieren, wie man gesehen hat (siehe Abbildung 3).

Alarmierung

Wenn es um das Versenden von Alarm-Meldungen geht, ist Nagios genauso flexibel konfigurierbar wie bei den Plug-ins für die Checks. Man muss nur einstellen, wer bei welchem Fehler zu welchen Zeiten auf welchem Weg benachrichtigt werden soll. Das Warum kann beispielsweise sein: „Plug-in check_oracle_health hat festgestellt, dass der Datenbank-Server XY nicht antwortet, selbst nach drei Wiederholungen nicht.“ Wer benachrichtigt wird, ergibt sich aus der Zuordnung von Contacts und Contactgroups zu den betroffenen Host- oder Service-Objekten.

Service State Information

Current Status:	CRITICAL (for 0d 0h 2m 7s)
Status Information:	CRITICAL - (5 errors) - ORA-06512: at "SYS.DBMS_STATS", line 23461 ...
	tag alertlog_bba
	ORA-00600: internal error code, arguments: [ktsircinfo_num1], [1], [2], [82139], [], [], [], []
	ORA-06512: at "SYS.DBMS_STATS_INTERNAL", line 67
	ORA-06512: at "SYS.DBMS_STATS_INTERNAL", line 134
	ORA-06512: at "SYS.DBMS_STATS_INTERNAL", line 2468
	ORA-06512: at "SYS.DBMS_STATS", line 23461
Performance Data:	alertlog_lines=6 alertlog_warnings=0 alertlog_criticals=5 alertlog_unknowns=0
Current Attempt:	1/1 (HARD state)
Last Check Time:	02-20-2009 20:04:44
Check Type:	ACTIVE
Check Latency / Duration:	2.146 / 1.612 seconds
Next Scheduled Check:	02-20-2009 20:09:44
Last State Change:	02-20-2009 20:02:49
Last Notification:	N/A (notification 0)
Is This Service Flapping?	YES (26.12% state change)
In Scheduled Downtime?	NO
Last Update:	02-20-2009 20:04:49 (0d 0h 0m 7s ago)

Abbildung 2: Im Alertlog wurden Fehlermeldungen entdeckt

Host	Service	Status	Last Check	Duration	Attempt	Status Information
dbsrv1	app_oracle_common_NAPRAX_check_login	OK	11-14-2008 20:39:58	0d 4h 18m 40s	1/4	OK - 3.39 seconds to connect as NAGIOS
	app_oracle_common_NAPRAX_check_ping	OK	11-14-2008 20:40:30	0d 4h 25m 8s	1/4	OK - connection established to NAPRAX.
	app_oracle_logs_NAPRAX_check_alertlog	OK	11-14-2008 20:41:01	0d 0h 6m 36s	1/1	OK - no errors or warnings
	app_oracle_perf_NAPRAX_check_databuf_hitratio	OK	11-14-2008 20:37:33	0d 0h 15m 4s	1/4	OK - SGA data buffer hit ratio 99.97%
	app_oracle_perf_NAPRAX_check_dictcache_hitratio	OK	11-14-2008 20:42:04	0d 5h 37m 22s	1/4	OK - SGA dictionary cache hit ratio 100.00%
	app_oracle_perf_NAPRAX_check_inmemory_sorts	OK	11-14-2008 20:37:36	0d 4h 23m 2s	1/4	OK - PGA in-memory sort ratio 100.00%
	app_oracle_perf_NAPRAX_check_latches_hitratio	OK	11-14-2008 20:38:08	0d 22h 40m 33s	1/4	OK - SGA latches hit ratio 100.00%
	app_oracle_perf_NAPRAX_check_libcache_hitratio	OK	11-14-2008 20:38:39	0d 1h 3m 59s	1/4	OK - SGA library cache hit ratio 100.00%
	app_oracle_perf_NAPRAX_check_redo_io	OK	11-14-2008 20:39:11	0d 4h 19m 27s	1/4	OK - Redo log io is 0.000228 MB/sec
	app_oracle_perf_NAPRAX_check_softparse	OK	11-14-2008 20:39:42	0d 4h 18m 56s	1/4	OK - Soft parse ratio 100.00%
	app_oracle_perf_NAPRAX_check_switchinterval	OK	11-14-2008 20:40:14	0d 4h 23m 24s	1/4	OK - Last redo log file switch interval was 591 minutes
	app_oracle_tbs_NAPRAX_SYSAUX_check_usage	OK	11-14-2008 20:40:45	0d 4h 22m 53s	1/4	OK - tbs SYSAUX usage is 1.90%
	app_oracle_tbs_NAPRAX_SYSTEM_check_usage	OK	11-14-2008 20:41:17	0d 5h 37m 22s	1/4	OK - tbs SYSTEM usage is 2.12%
	app_oracle_tbs_NAPRAX_TEMP_check_usage	OK	11-14-2008 20:41:49	0d 5h 37m 21s	1/4	OK - tbs TEMP usage is 0.00%
	app_oracle_tbs_NAPRAX_TEST_TBS_check_usage	OK	11-14-2008 20:42:20	0d 5h 37m 22s	1/4	OK - tbs TEST_TBS usage is 19.50%
	app_oracle_tbs_NAPRAX_UNDOTBS1_check_usage	OK	11-14-2008 20:37:52	0d 4h 22m 46s	1/4	OK - tbs UNDOTBS1 usage is 0.03%
	app_oracle_tbs_NAPRAX_USERS_check_usage	OK	11-14-2008 20:38:23	0d 6h 32m 42s	1/4	OK - tbs USERS usage is 0.00%
	app_ticker_check_noticks	OK	11-14-2008 20:40:55	0d 0h 1m 42s	1/4	OK - noticks: 26s

Abbildung 3: Umfangreiches Monitoring eines Datenbank-Servers

```
$ check_oracle_health … --mode sql \
    --warning 8 -critical 15 \
    --name “select count(*) from app1.processes where status =
’failed‘ “ --name2 failedprocs
CRITICAL - failedprocs: 29 | failedprocs=29;8;15
```

Listing 8

Üblicherweise gibt es einen Contact namens „db-admins“, der allen datenbankrelevanten Services zugeordnet ist. Dessen Mailadresse ist eine Sammeladresse für das Datenbank-Team.

Ebenso konfigurierbar ist das Wann, also zu welchen Uhrzeiten überhaupt Alarme verschickt werden. Die angesprochene Flexibilität zeigt sich aber am deutlichsten beim Wie. Analog zu den Checks, die von beliebigen Skripten (den Plug-ins) ausgeführt werden, kommen auch auf der Alarmierungsseite Notification-Scripts zum Einsatz. Nagios kümmert sich lediglich um den Aufruf der Skripte mit passenden Kommandozeilen-Parametern. Was diese Scripts letztlich tun, bleibt dem Administrator überlassen.

Natürlich ist bei OMD ein Skript enthalten, das für den Versand von E-Mails sorgt. Damit dürften für die meisten Installationen bereits die Anforderungen abgedeckt sein. Wünscht man jedoch den Versand einer SMS oder das automatische Öffnen eines Tickets in einem entsprechenden Tool, kann dies durch geeignete Notification-Skripte geschehen. Im Internet gibt es zahlreiche solcher Programme, die man herunterladen und in sein Monitoring einbinden kann.

Um das Thema anschaulicher zu beschreiben, folgendes Szenario: Bei jeder Störung einer produktiven Datenbank sollen von 8 bis 18 Uhr die DBAs eine E-Mail bekommen. Ab 18 Uhr soll eine SMS an das Bereitschaftshandy geschickt werden. Bleibt ein Fehlerzustand länger als zwei Stunden unbearbeitet, soll auch der Teamleiter eine SMS bekommen (sogenannte „Notification Escalation“). Parallel dazu ist bei jedem Incident auch automatisch ein Ticket anzulegen. Störungen von Testsystemen sollen lediglich per E-Mail gemeldet werden. Bei Ende des Ausfalls sollen alle wieder eine „OK“-Mail erhalten und das zugehörige Ticket automatisch geschlossen werden.

Solche Abläufe sind bei Nagios ziemlich einfach zu konfigurieren. Man kann sogar noch einen Schritt weiter gehen. Als dritte Kategorie von Skripten, die aus Nagios heraus aufgerufen werden können, gibt es die Event-Handler. Damit kann man beispielsweise ein ausgefallenes System durchstarten lassen. Gelingt es auf diese Art, vor dem dritten Check wieder einen Normalzustand herzustellen, so wird kein Alarm ausgelöst. Den Bereitschafts-DBA wird es freuen. Dass etwas passiert ist, bleibt trotzdem nicht verborgen. In den Logs von Nagios wird ja jedes Ereignis protokolliert.

Reporting

Störungen und Ausfälle frühzeitig mitzubekommen, um größeren Schaden abwenden zu können, ist eine feine Sache. Die Vorgabe wird aber meistens lauten, dass es erst gar nicht zu solchen Situationen kommen soll. Am Ende des Monats will man daher wissen, ob die Systeme fehlerfrei durchgelaufen sind – und falls nicht, wie oft und wie lange nicht. Die Weboberfläche „Thruk“, Bestandteil von OMD, bietet zu diesem Zweck ein mächtiges Reporting-Tool an. Out-of-the-box lassen sich damit Verfügbarkeitsberichte erstellen sowie die Auskunft über die Erreichbarkeit einzelner Hosts, ganzer Host-Gruppen oder dedizierter Services geben (siehe Abbildung 3).

Ein Sonderfall ist SLA-Reporting. Hier gibt man beim erstmaligen Einrichten des Reports die vereinbarte Verfügbarkeit an und erhält dann im

Monatsbericht noch zusätzlich eine Liste der SLA-Verletzungen. Mit wenig Aufwand kann man Farben und Seitengestaltung so anpassen, dass das Aussehen der Reports dem firmentypischen Design entspricht.

Einmal angelegt kann man jederzeit einen aktuellen Report in der Thruk-GUI erzeugen und sich anzeigen lassen. Praktischer ist es allerdings, dies dem System zu überlassen. In Thruk muss man dazu nur einen Empfängerkreis und den Zeitpunkt des Versands angeben, dann werden die Reports automatisch beispielsweise zum Monatsende erstellt und per Mail als PDF verschickt.

Fazit

Nagios ist als lizenzfreies Monitoring-System so flexibel konfigurierbar, dass die meisten Anforderungen out-of-the-box umsetzbar sind. Fehlende Funktionalität kann mit wenig Aufwand dank der offenen Schnittstellen selbst implementiert werden. Für viele Sonderwünsche gibt es bereits existierende Add-ons, die man sich herunterladen kann. Das Budget, das ansonsten in Form von Lizenzgebühren verbraucht würde, investiert man besser in eigene Erweiterungen.

In Form von OMD steht ein Paket zur Verfügung, das den initialen Installationsaufwand zu einer Sache von Minuten werden lässt. Sicherlich bieten ein Enterprise Manager oder vergleichbare Kauftools dem DBA die Möglichkeit, seine Datenbanken auch bedienen zu können. Gleiches gilt für SAP, HP Insight Manager etc. Es empfiehlt sich jedoch, diese Programme ausschließlich als Management-Tools für die jeweiligen Spezialisten zu nutzen und die entsprechende Monitoring-Komponente an Nagios abzugeben oder zumindest zu koppeln. Auf diese Weise erhält man ein System (das dazugehörige Modewort lautet „Umbrella-Monitoring"), durch das sämtliche Alarmierungen laufen und das einen einheitlichen Blick auf alle IT-Objekte eines Unternehmens bietet.

Weiterführende Informationen

- http://www.nagios.org
- http://omdistro.org
- http://labs.consol.de/nagios/check_oracle_health
- http://labs.consol.de/nagios/check_logfiles
- http://www.thruk.org

Gerhard Laußer
gerhard.lausser@consol.de