



Stolpersteine bei der Überwachung von RAC- Umgebungen

Thorsten Bruhns

Senior Solution Architect

1 Grundlagen RAC + Grid-Infrastructure

2 Monitoringarchitekturen

3 Herausforderungen beim Monitoring

4 Fazit

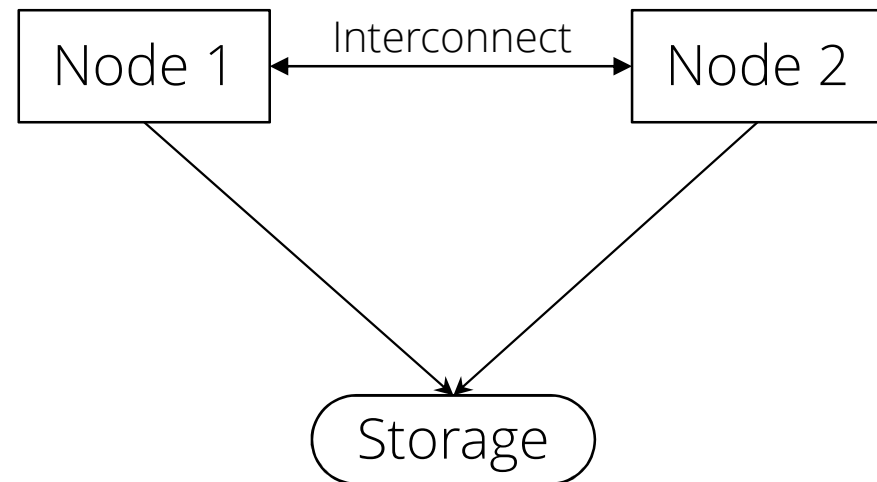


Grundlagen RAC + Grid-Infrastructure



Grundlagen RAC

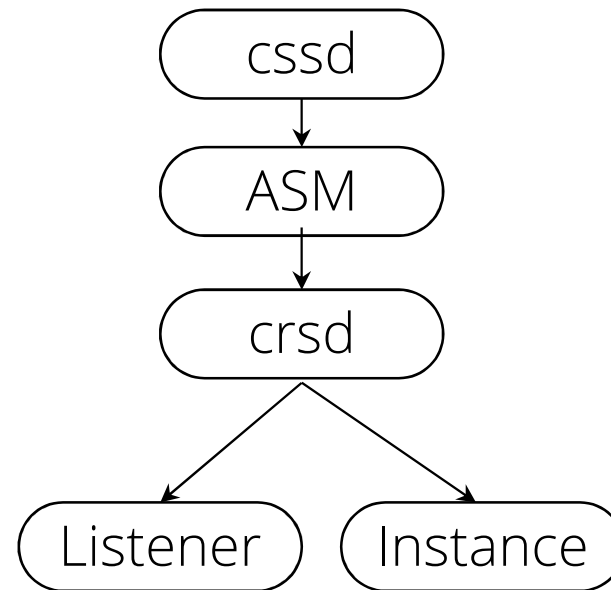
- Einführung mit 9i
- Active/Active Cluster
 - RAC OneNode ist auch Active/Active
 - Pro Clusterknoten 1 Instance mit Shared Storage
- Gute Skalierung
- Schneller Failover
- ‚ausgereifte Lösung‘
 - Nicht ganz preiswert aber gut einsetzbar



Grundlagen Grid-Infrastruktur

Wesentliche Prozesse

- Clusterware von Oracle
 - Früher CRS (Cluster Ready Services)
 - Einführung mit 10g
- Änderungen mit 11.2
 - Änderung in Grid-Infrastruktur
 - ASM wird Bestandteil der Clusterware
 - Voting in ASM oder auf NFS
 - Bei ASM Henne/Ei-Frage wegen Voting
 - Einführung policy managed Clusterware
- Vortrag betrachtet nur crsd-
Ressourcen!



Grundlagen Grid-Infrastructure

Administrative based

- Clusternode
 - Datenbankinstance
- Klassische Methode
- Einfacher administrierbar
 - Vorhersehbar wo was läuft
 - Cron-Jobs einfach realisierbar
- Monitoring deutlich einfacher

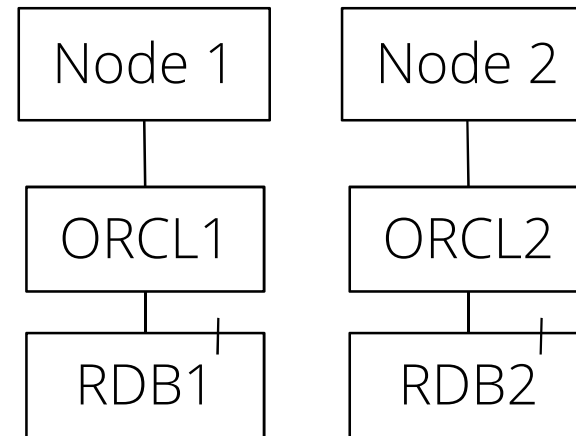
Policy based

- Clusternode
 - Serverpool
 - Datenbankinstance
- Flexible Verteilung
 - Hilfreich in großen Cluster mit vielen Nodes
- cron-Jobs viel komplizierter
- Empfehlung
 - Nur einsetzen wenn nötig
 - Nur sinnvoll bei Clustern > 2 Nodes
 - Bei RAC OneNode Pflicht...

Grundlagen Grid-Infrastructure

Administrative based

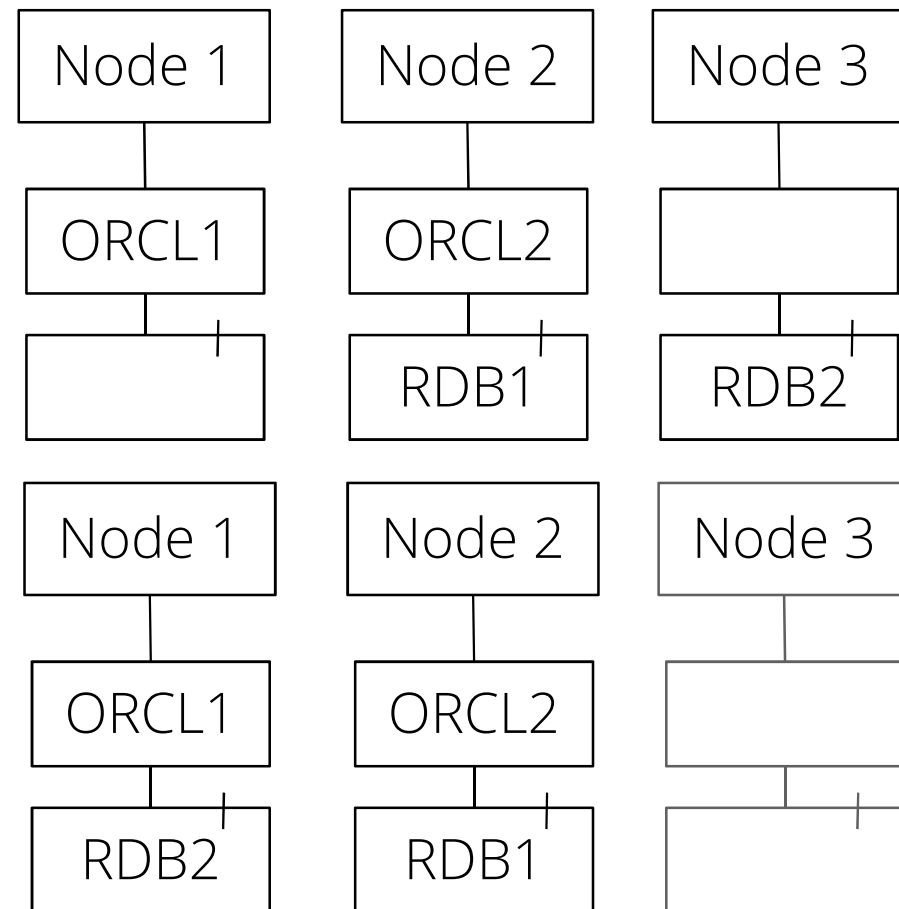
- Statische Instancennamen
 - Einfache Zuordnung von Instance => Clusternode
 - Einfache Administration der Umgebung
 - Zuordnung nach DBCA nicht änderbar
- Pflicht bei Clustern mit 2 Knoten
 - Warum es unnötig kompliziert machen?
 - Ausnahme RAC OneNode
 - Wer hat einen RAC OneNode?
 - Meine persönliche Meinung



Grundlagen Grid-Infrastructure

Policy based

- Regelbasierte Zuweisung
 - Server werden in Pools gruppiert
 - Datenbankinstanzen werden Pools zugewiesen
 - Instance_id wird durch Clusterware bestimmt
 - Kann sich bei Ausfällen von Knoten ändern!



Monitoringarchitekturen



Monitoringarchitekturen

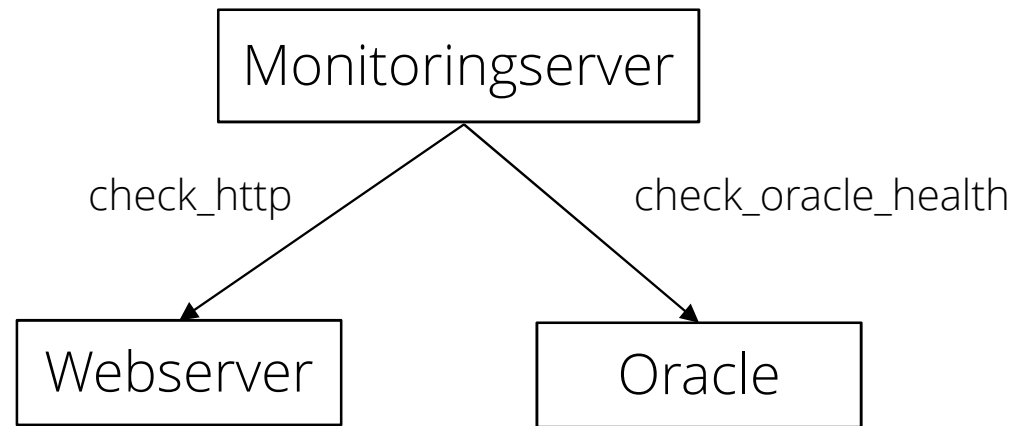
- Aktive Checks vom Monitoringhost
- Agentbasiert auf Zielsystem
- Spezialagent auf Monitoringhost



Monitoringarchitekturen

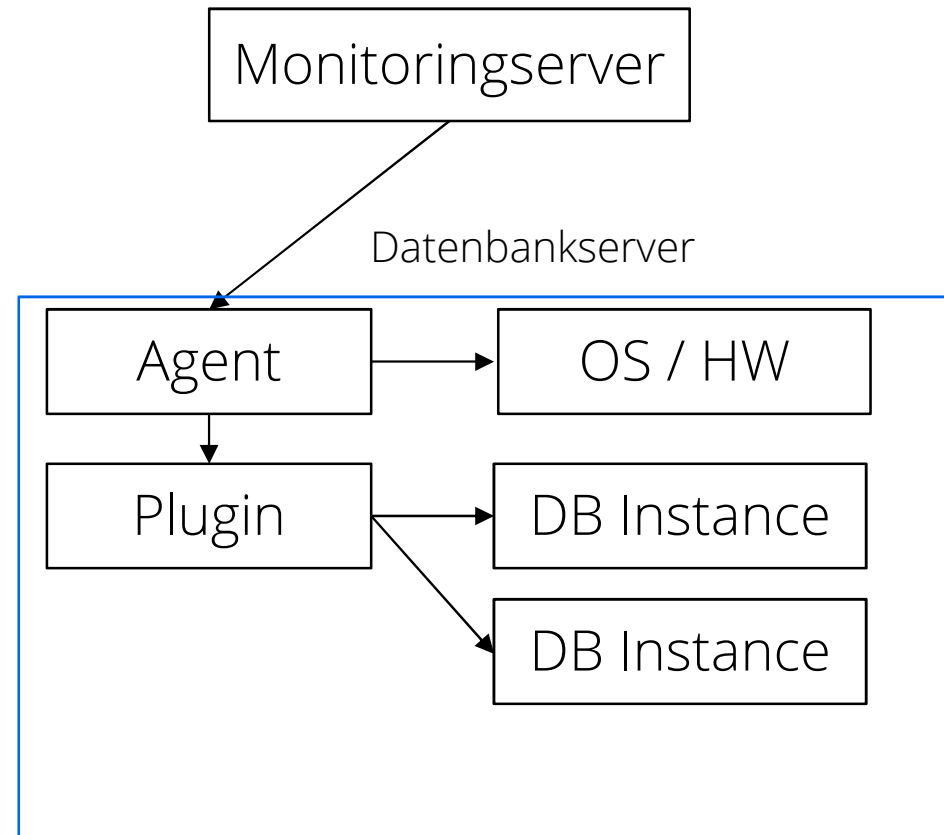
Aktive Checks

- Vom Monitoringserver gestartet
 - check_http
 - Mittels wget/curl vom Monitoringhost
 - check_oracle_health
 - Oracle Überwachungsskript per SQLNet
- Keine Änderungen auf Zielsystem
 - Schnell implementierbar
 - Aufwändig in der Wartung
 - Aus Nagioswelt viele Checks verfügbar
 - Polling kann Last verursachen
 - Abfrage jeder DB und/oder Tablespace
- Einsatz nur wenn wirklich benötigt



Monitoringarchitekturen agentbasiert

- Monitoringserver pollt Agent
 - Wesentlicher Teil der Logik im Agent
 - Plugin in vielen Lösungen optional
 - Agent liefert viel Basischecks
 - Automatische Erkennung von Zielen
 - Pluginentwicklung kann komplex werden
 - Kostenfaktor
 - Plugins in kommerziellen Lösung gerne mal teuer...
 - Agenten plattformabhängig
 - Nur bei exotischen Plattformen ein Thema
 - Security des Agenten bewerten

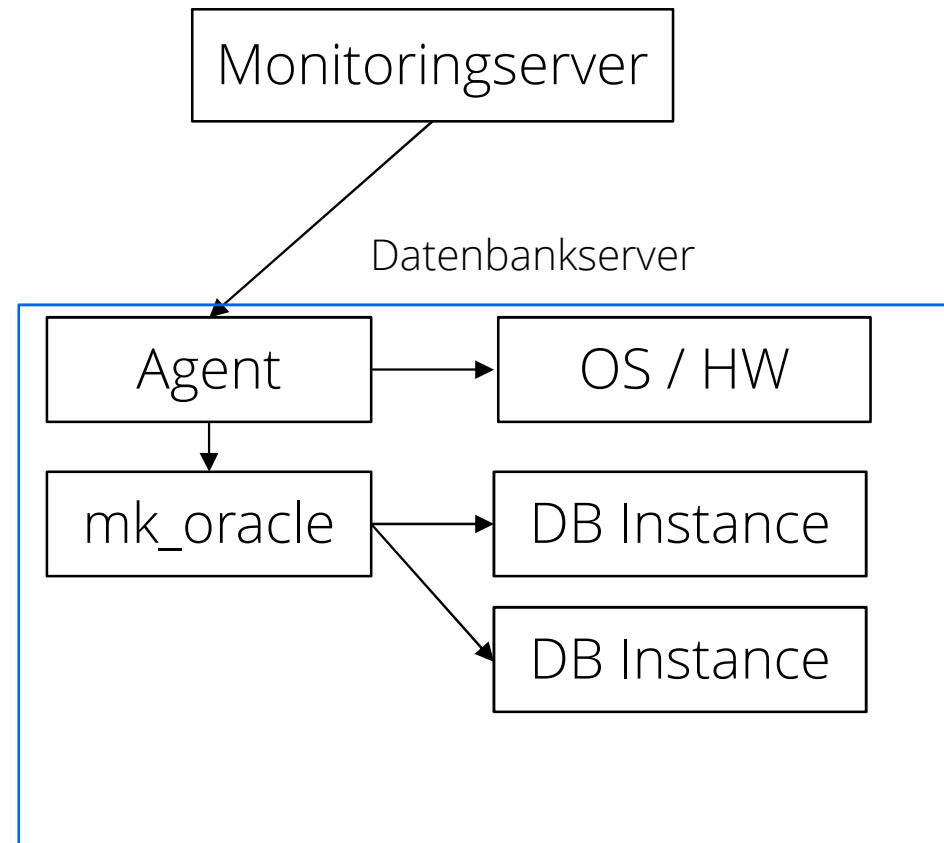


Monitoringarchitekturen

Oracle agentbasiert mit Check_MK

- Plugin mk_oracle verfügbar
- Automatische Erkennung
 - Instancen + Tablespaces + Backups
 - ORACLE_SID wesentlicher Faktor bei Erkennung
 - ‚Designschwäche‘ bei policy managed
- Wenig Last für DB
 - alle SQLs mit 1 Login
 - 1 SQL für alle Tablespaceabfragen
 - Gilt auch für Backups
- Overhead im Cluster
 - Plugin wird auf jedem Node ausgeführt

mk_oracle

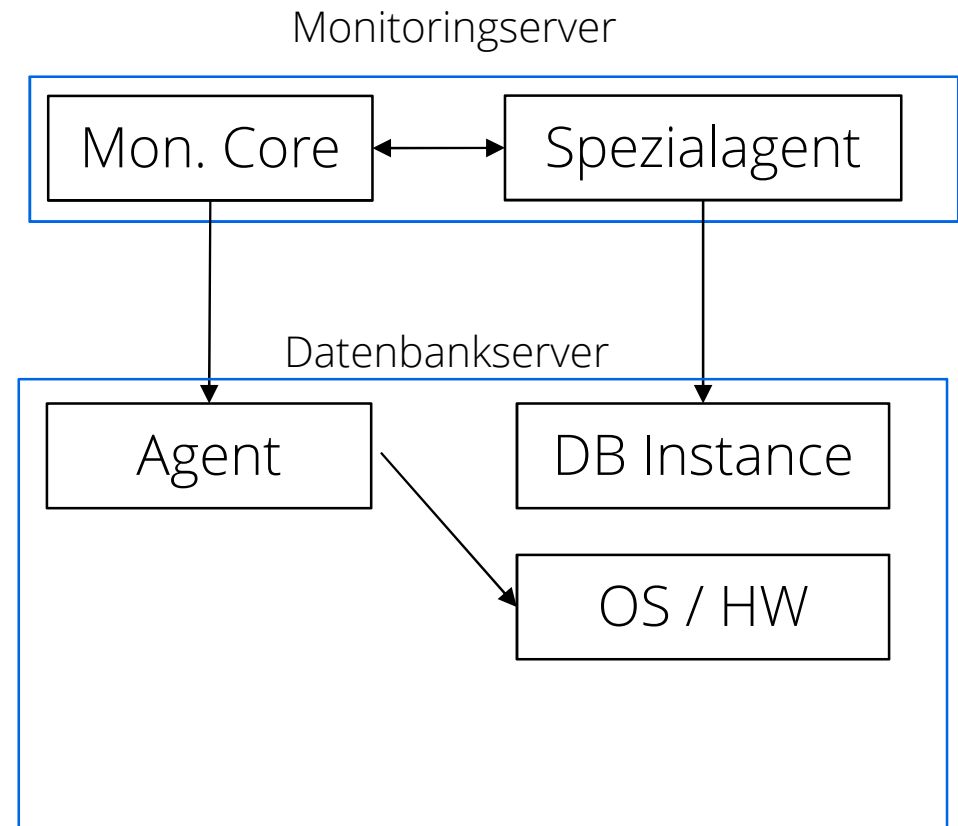


Monitoringarchitekturen

Spezialagent

- Besondere Lösung in Check_MK
 - Ähnlich zu aktive Checks
 - Für Komplexe Anforderungen an Check
 - Installation / Änderungen am Zielsystem verboten/nicht möglich
 - Für Spezialgebiete konzipiert
 - VMware vSphere
 - Stagesysteme
 - NetApp, 3par, IBM SVC
 - Oracle
 - Policy managed Cluster
 - Noch in Entwicklung
 - Spezialagent + agentbasierte Überwachung parallel möglich

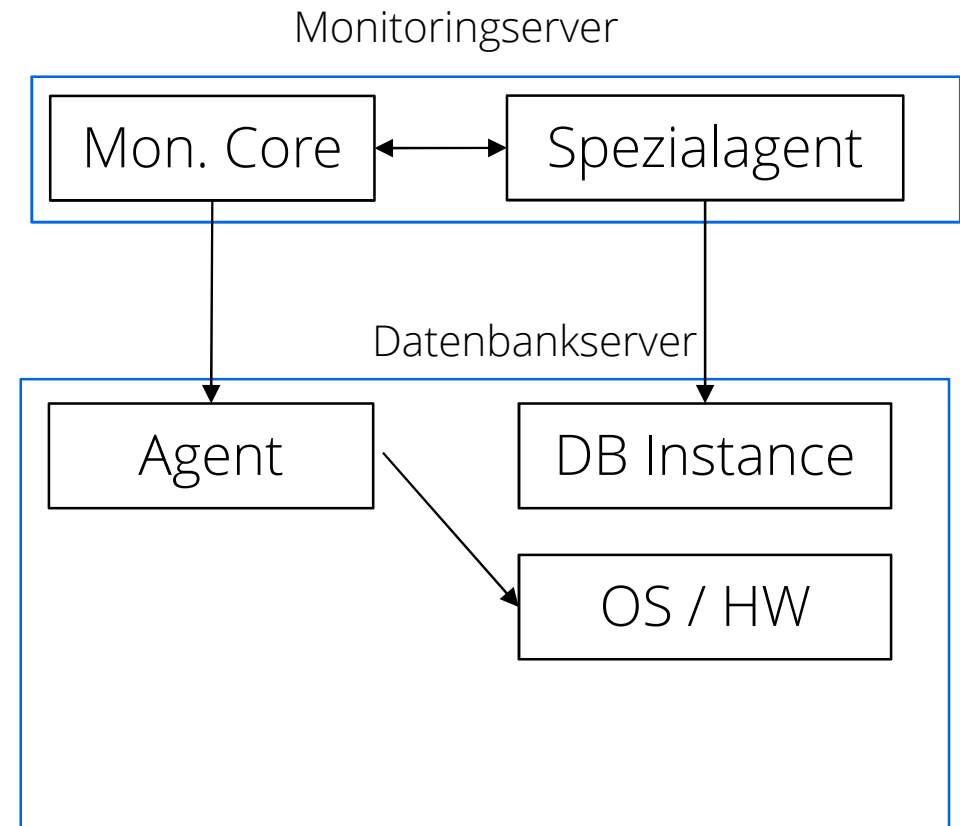
Oracle mit Spezialagent



Monitoringarchitekturen

Spezialagent für Oracle

- Verbindung mittels SQLNet
- Keine automatische Erkennung
 - Keine Erkennung von neuen Instanzen
- Kann über SCAN-Listener arbeiten
 - Unabhängig von der Zuordnung ORACLE_SID ↔ Node



Herausforderungen beim Monitoring



Herausforderungen beim Monitoring

Anforderungen Single-Instance

- Einfache Konfiguration
 - Idealerweise nur Username + Paßword für Plugin
- Automatische Erkennung von Instanzen
 - Prozeßliste (pmon) oder oratab
- Darstellung in der Oberfläche
 - Hostname
 - Servicename
 - Was soll bei Datenbanken genutzt werden?
 - db_name bei single Instance
 - db_name und instance_name im RAC
- Cloud-Control speziell dafür gebaut
 - Dedizierter Bereich für Datenbankmonitoring

Provider, Local site ocicest, vsllgm156	
State	Service
OK	ORA SMA Instance
OK	ORA SMA Locks
OK	ORA SMA Logswitches
OK	ORA SMA Long Active Sessions
OK	ORA SMA Performance
OK	ORA SMA Processes
OK	ORA SMA Recovery Area
OK	ORA SMA Recovery Status

Herausforderungen beim Monitoring Anforderungen RAC

- Ganze einfach

- Genauso wie bei single Instance. :-)
 - Leider geht das nicht...



Herausforderungen beim Monitoring

Probleme beim RAC

- Beispiel: db_name: INFRAP
 - Node1 INFRAP1
 - Node2 INFRAP2
- Wohin mit den Tablespaces, Jobs und RMAN?
 - INFRAP1 oder INFRAP2 werden gebraucht
 - Lösung:
 - Services unter virtuellen Host (Clustered Host) zusammen fassen
- Das funktioniert wunderbar mit administrative managed
- Was ist mit policy managed?

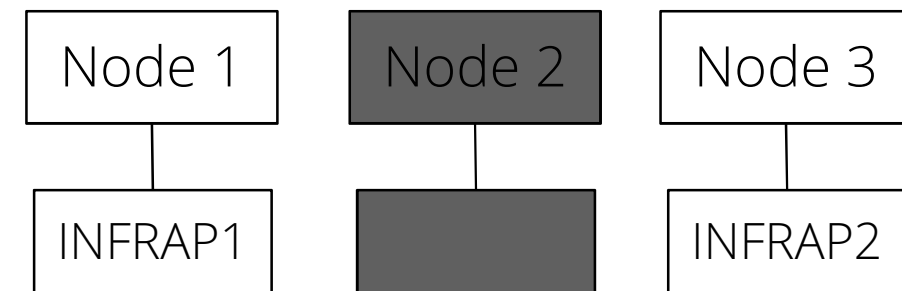
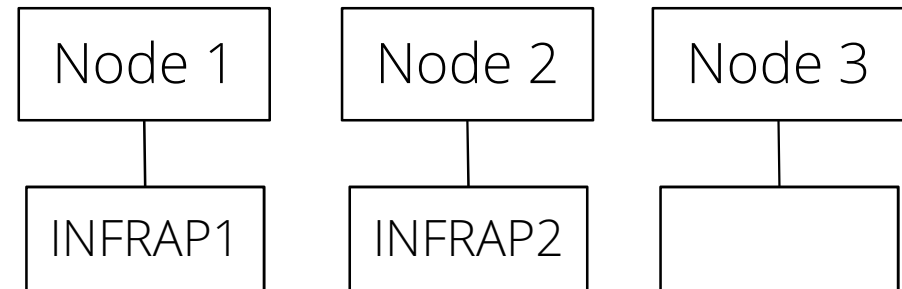
OK	ORA INFRAP1 Instance
OK	ORA INFRAP1 Locks
OK	ORA INFRAP1 Logswitches
OK	ORA INFRAP1 Long Active Sessions
OK	ORA INFRAP1 Performance
OK	ORA INFRAP2 Instance
OK	ORA INFRAP2 Locks
OK	ORA INFRAP2 Logswitches
OK	ORA INFRAP2 Long Active Sessions
OK	ORA INFRAP2 Performance
OK	ORA INFRAP.DB_INCR_1 RMAN Backup
OK	ORA INFRAP.JMSSTORE Tablespace
OK	ORA INFRAP.PERFSTAT Tablespace

Herausforderungen beim Monitoring

Probleme beim RAC

- Annahmen:
 - Serverpool mit 3 Nodes
 - INFRAP soll auf 2 Nodes aktiv sein
- Beispiel oben
 - sieht wie bei administrative managed aus
- Beispiel unten
 - Hier ist INFRAP2 auf Node 3 gewandert
 - Kann passieren, wenn Node 2 ausfällt
- Problem
 - Services wandern von Node 2 => Node 3

Policy managed

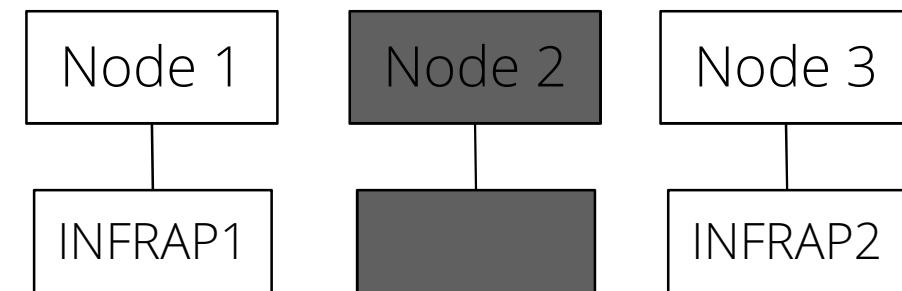
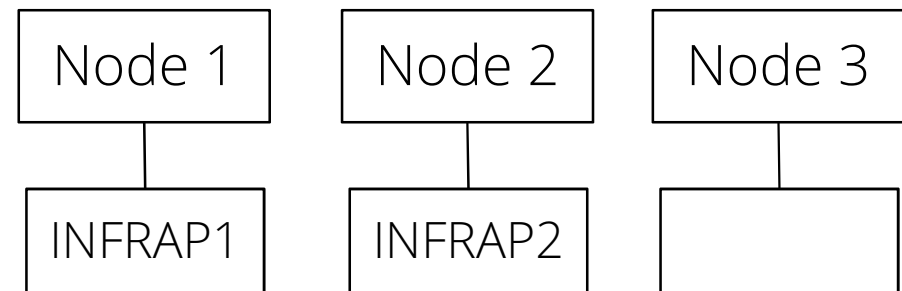


Herausforderungen beim Monitoring

Probleme beim RAC

- Lösungsmöglichkeit:
 - Services von Instanzen an clustered Hosts
 - Damit können sie nicht mehr wandern
 - Unübersichtlich
- Wie sieht das eigentlich im RAC OneNode aus?

Policy managed

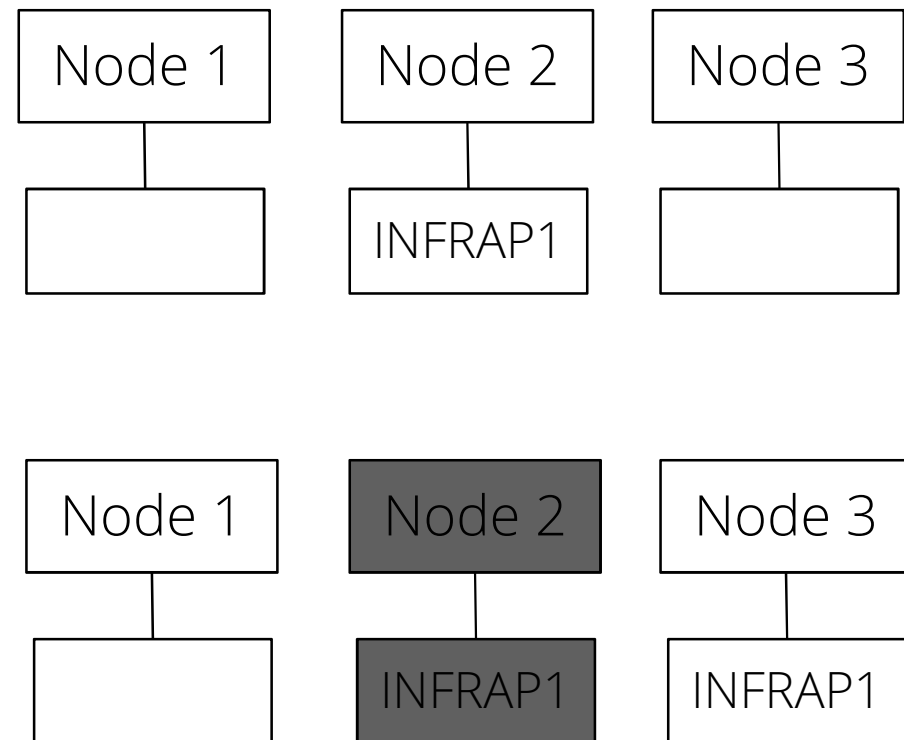


Herausforderungen beim Monitoring

Probleme beim RAC OneNode

- Beispiel oben
 - IFRAP läuft auf 1 Node im Cluster
 - Per srvctl veränderbar
- Achtung
 - Unterschied zwischen Failover und Switchover
- Failover
 - Genauso wie policy managed RAC
 - Failover kann auch auf Node 1 erfolgen

Policy managed

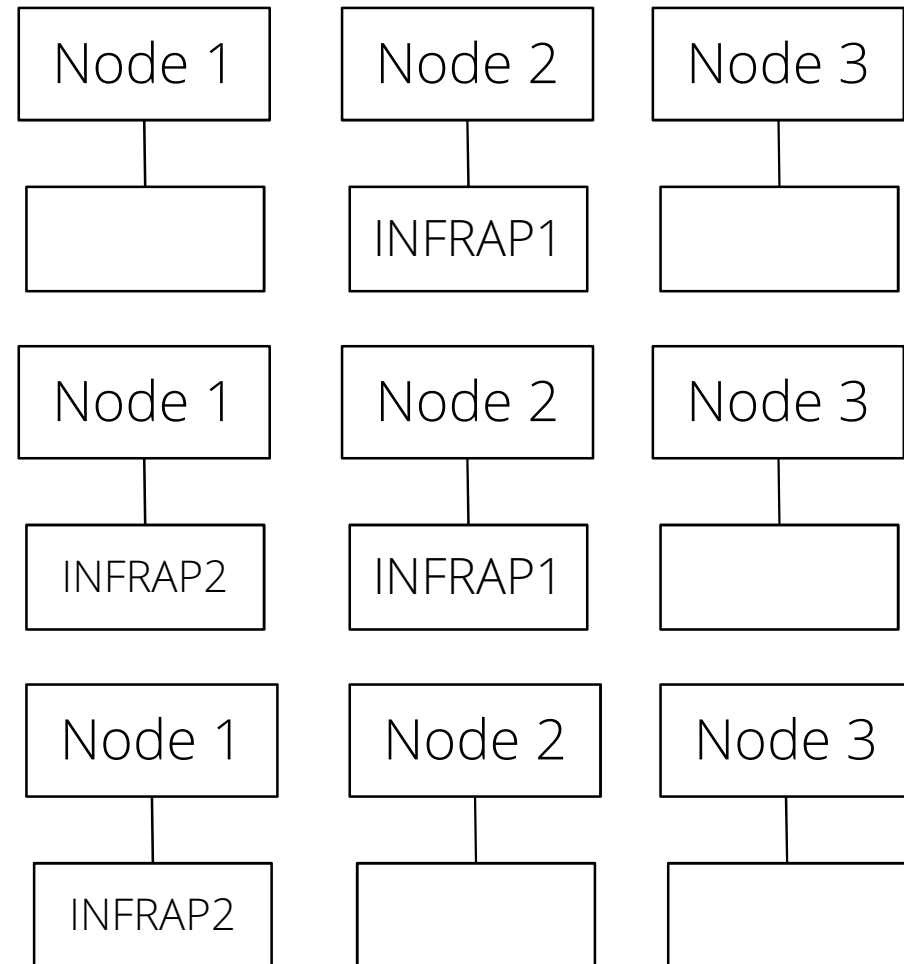


Herausforderungen beim Monitoring

Probleme beim RAC OneNode

- Beispiel oben
 - INFRAP läuft auf 1 Node im Cluster
- Switchover von Node 2 => Node 1
 - Starte auf Node 1 neue Instance
 - Hier wird eine ‚freie ORACLE_SID‘ gewählt
 - RAC OneNode ist nun active auf 2 Knoten
 - Shutdown transactional INFRAP1
 - INFRAP2 läuft auf Node 1
- Fazit
 - ORACLE_SID kann wandern
 - Extrem bei RAC OneNode wegen Switchover
 - Administrativ gewöhnungsbedürftig..
 - Monitoring muß darauf reagieren ...

Policy managed



Herausforderungen beim Monitoring

Lösungsvorschlag

- Aufbaue der festen Bindung ORACLE_SID => Node notwendig
 - Spannend, wenn Hostname + Service im Monitoringarchitektur vorgesehen ist
- Anzahl aktiver Instancen bei policy managed relevant
 - Logincheck kann nur eingeschränkt helfen
 - Abfrage von gv\$instance über SCAN-Listener mit Zählen aktiven Instancen
 - Mit Agent auf Node redundante Checks
 - Mk_oracle hat gecachte Abfrage an Oracle, die sich auf ORACLE_SID beziehen
 - Was ist wenn der Node nicht aktiv ist?
- Historie welche Instance wo lief
 - Kompliziert, da Statustext von Checks nicht historisiert wird
- Administrative managed ist einfach
 - Alle instancen müssen laufen
- Ist

Herausforderungen beim Monitoring

Lösungsvorschlag

- Wann lief welche Instance wo?
 - Herausforderung, weil Checks unter clustered Hosts gesammelt werden müssen
 - Historie des Instancecheck wird OK => CRITICAL => OK bei Fehlern sein
 - Jeder Wechsel ist mit Text in der Historie sichtbar, wenn Ausgabe Hostnamen enthält

Time	Event	Type	
2017-11-13 01:59:31 - 7 d	SERVICE ALERT	HARD	OK - Database Name PVS, Status OPEN, Force Logging yes
2017-11-13 01:32:22 - 7 d	SERVICE ALERT	HARD	CRIT - Database not running or login failed

Fazit



Fazit

- Administrative manage + Single Instance
 - Einfach
 - Node ↔ Datenbankinstancen
 - Wenn möglich nutzen, um sich Leben einfach zu machen
 - Monitoringsetzup einfach
 - Agent oder Spezialagent nutzbar
 - Automatisch Erkennung aller Instancen möglich
- Policy managed
 - Umdenken notwendig:
 - Node ↔ Serverpool ↔ Datenbankinstancen
 - Leider Pflicht bei RAC OneNode
 - Sehr praktisch bei Cluster > 3 Nodes

Fazit

■ Policy managed

- Umdenken notwendig:
 - Node ↔ Serverpool ↔ Datenbankinstanzen
- Leider Pflicht bei RAC OneNode
- Sehr praktisch bei Cluster > 3 Nodes
- Monitoring
 - Möglich, wenn die Verhaltensweisen bekannt
 - Erfordert ggf. spezielle Checks für Datenbankstatus
 - Check_MK: mit Spezialagent möglich (in Entwicklung)



Gemeinsam werden wir astreine Layouts bauen,
ich zähle auf euch!



Thorsten Bruhns

Senior Solution Architect

Norsk-Data-Straße
51647 Bad Homburg

thorsten.bruhns@opitz-consulting.com

+49 6172 6626 0 1541



WWW.OPITZ-CONSULTING.COM



[@OC_WIRE](https://twitter.com/OC_WIRE)



[OPITZCONSULTING](https://www.youtube.com/OPITZCONSULTING)



[opitzconsulting](https://www.linkedin.com/company/opitzconsulting)



[opitz-consulting-bcb8-1009116](https://twitter.com/opitz-consulting-bcb8-1009116)