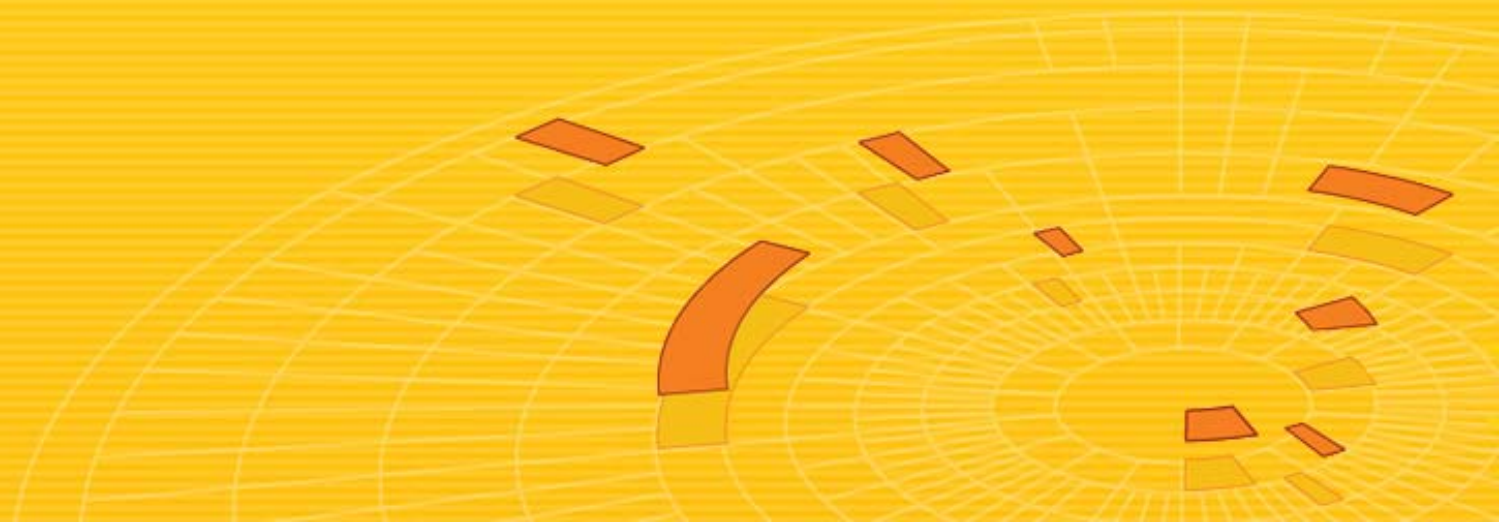
ConSol
Consulting & Solutions
www.consol.de

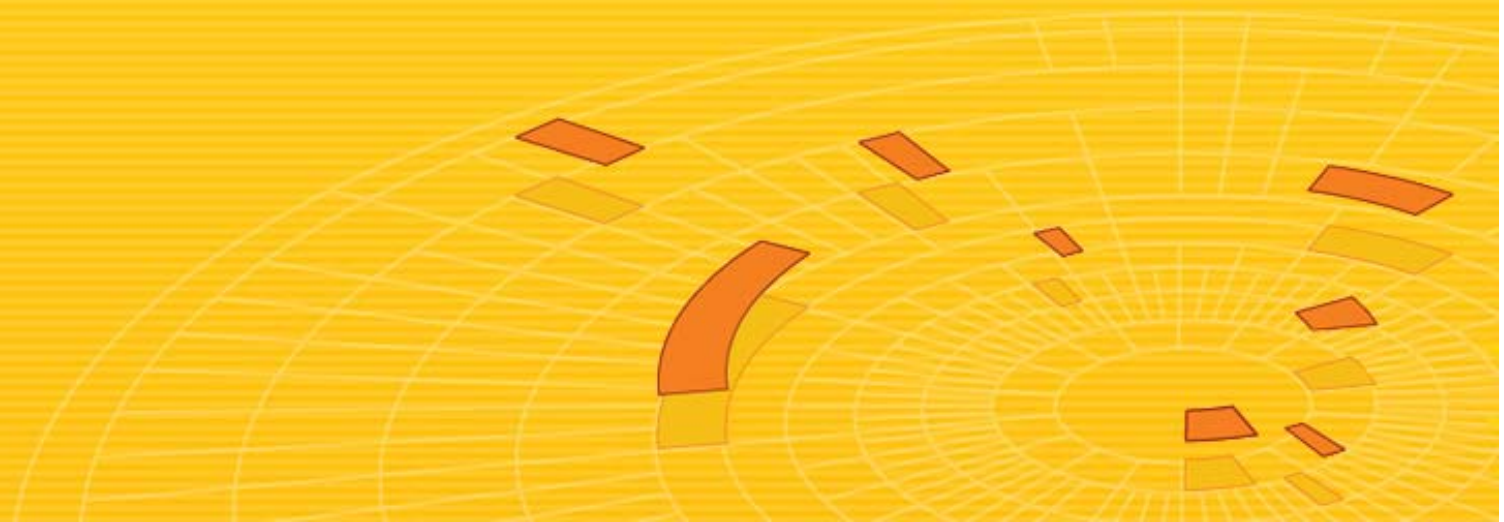
Überwachung von Logfiles mit check_logfiles



ConSol
Consulting & Solutions
www.consol.de

Wer bin ich?

Gerhard Laußer
aus München
arbeite bei der Firma ConSol
betreue Nagios bei ***



ConSol
Consulting & Solutions
www.consol.de

Entstehungsgeschichte

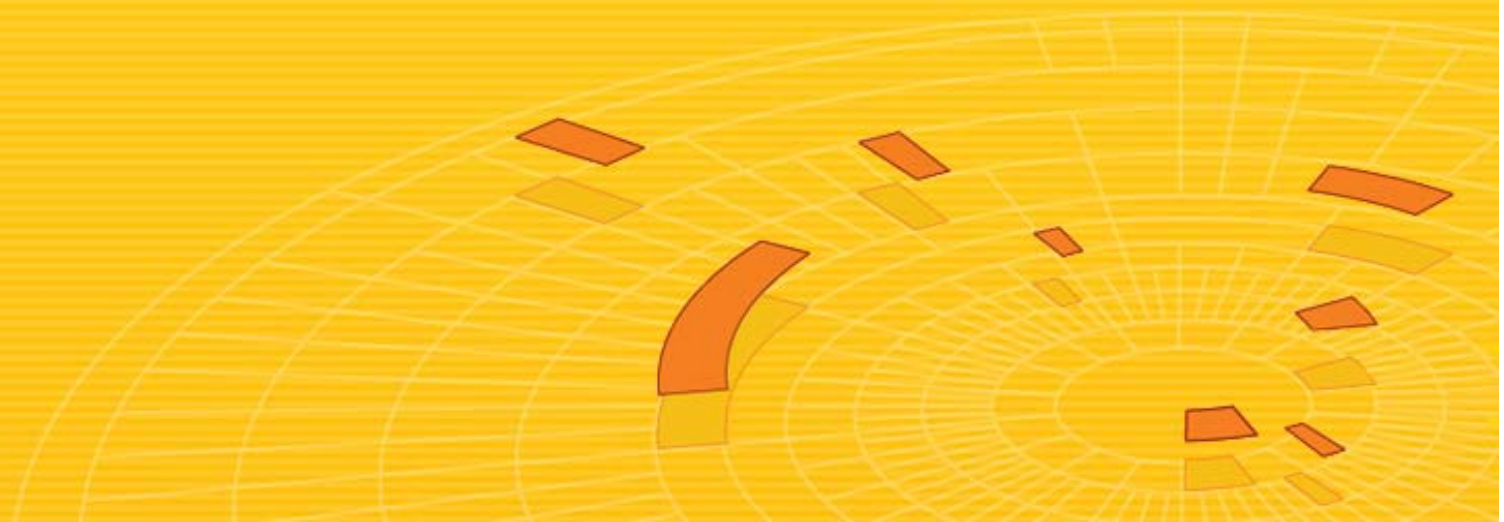
- Tivoli musste abgelöst werden.
- Tivoli hat einen sog. Logfile-Adapter.
- Programme müssen z.T. in Pipes schreiben.
- Ein Leseprozess muss ständig laufen.
- Schon wieder etwas, das kaputt gehen kann.
- Vorhandene Plugins lassen Lücken zu.
- Erste Version von Anfang 2006 für den Eigenbedarf.
- Stetige Weiterentwicklung und Zufluss von Ideen.

ConSol
Consulting & Solutions
www.consol.de

Her damit!

http://www.consol.de/opensource/nagios/check-logfiles

http://sourceforge.net/projects/check-logfiles



ConSol
Consulting & Solutions
www.consol.de

Auspacken und Zusammenbauen

```
tar zxvf check_logfiles-2.3.tar.gz
cd check_logfiles-2.3
./configure –help
   .....
   --libexecdir=DIR         program executables [PREFIX/libexec]
   ......
  --with-nagios-user=USER   set user name to run nagios
  --with-nagios-group=GROUP set group name to run nagios
  --with-seekfiles-dir=PATH sets directory for the state files
                            (default=/tmp)
  --with-protocols-dir=PATH sets directory for the protocol files
                            (default=/tmp)
  --with-trusted-path=PATH  sets trusted path for executables called by
                            scripts
(default=/bin:/sbin:/usr/bin:/usr/sbin)
  --with-perl=PATH          sets path to perl executable
  --with-gzip=PATH          sets path to gzip executable
   ......
```

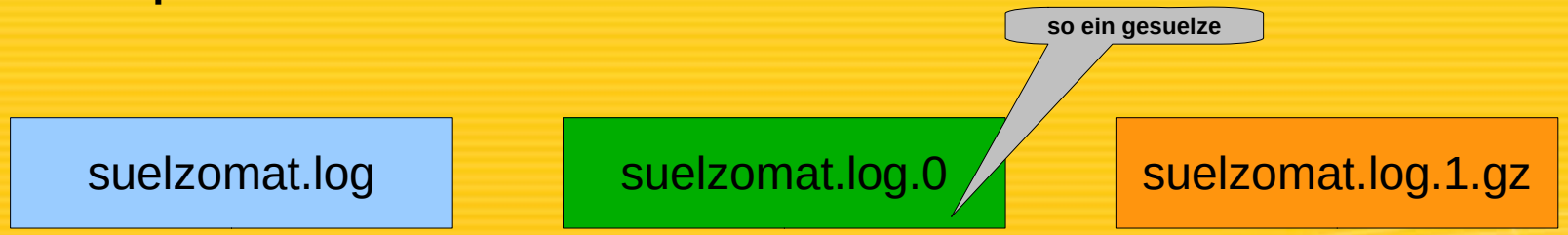
ConSol
Consulting & Solutions
www.consol.de

So funktioniert's

suelzomat.log

suelzomat.log.0

```
$ check_logfiles \
    --logfile=/var/log/suelzomat.log \
    --tag=gesuelz --criticalpattern='.*suelz.*'\
    --rotation=debian
OK - no errors or warnings |gesuelz_lines=31 gesuelz_warnings=0
gesuelz_criticals=0 gesuelz_unknowns=0
$ suelzomat –log-msg=”so ein gesuelze”
$ sleep 999999
```




```
$ check_logfiles \
    --logfile=/var/log/suelzomat.log \
    --tag=gesuelz --criticalpattern='.*suelz.*'\
    --rotation=debian
CRITICAL - (1 errors in check_logfiles.protocol-2007-10-10-14-59-05) - so
ein gesuelze |gesuelz_lines=9371 gesuelz_warnings=0 gesuelz_criticals=1
gesuelz_unknowns=0
$
```

ConSol
Consulting & Solutions
www.consol.de

So funktioniert's

suelzomat.log

suelzomat.log.0

Hier endet der erste Lauf.
Merke: Offset im Logfile und Modification Time

Einige Zeit vergeht.....
Eine Fehlermeldung wird erzeugt
Mehr Zeit vergeht.....
suelzomat.0 wird zu suelzomat.1.gz
suelzomat.log wird zu suelzomat.log.0
Ein neues suelzomat.log wird angelegt
Es wurden insges. 9371 Meldungen vom Suelzomat erzeugt

suelzomat.log

suelzomat.log.0

suelzomat.log.1.gz

ConSol
Consulting & Solutions
www.consol.de

So funktioniert's

suelzomat.log

suelzomat.log.0

suelzomat.log.1.gz

2.Lauf: Wo waren wir stehengeblieben? Zeitstempel und Offset

Such mir jetzt alle Logfilekandidaten, die sich seit Zeitstempel geändert haben.

suelzomat.log

suelzomat.log.0

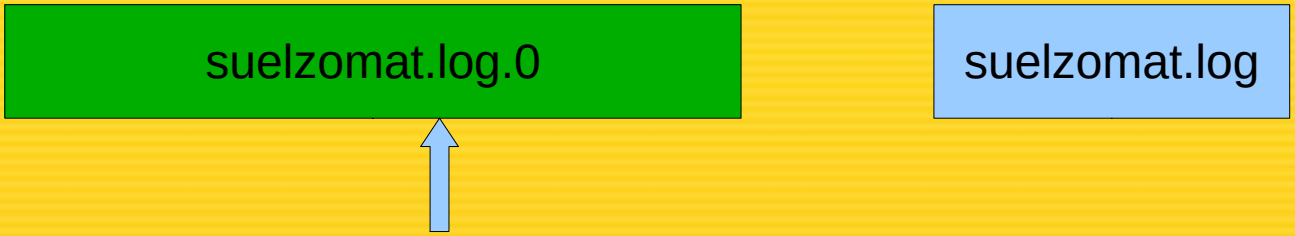
Sortier sie so, daß die älteste an erster Stelle kommt.
Das ist dann das ehemalige Logfile, in dem wir ans Dateiende gestossen sind.

suelzomat.log.0

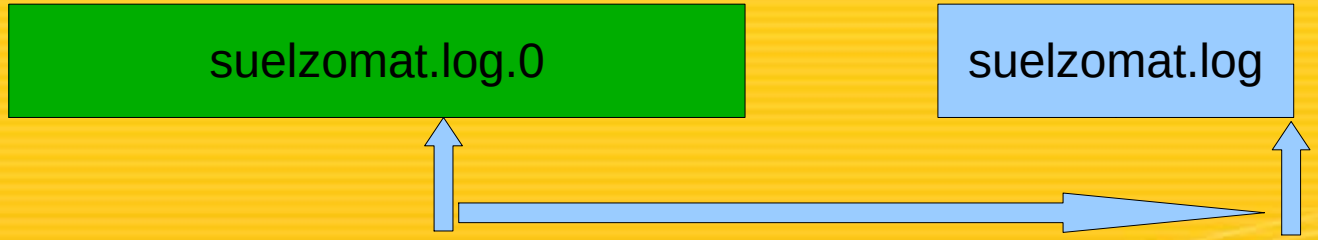
suelzomat.log

ConSol
Consulting & Solutions
www.consol.de

So funktioniert's



Und jetzt durchsuchen wir alles, was seitdem an Meldungen entstanden ist.

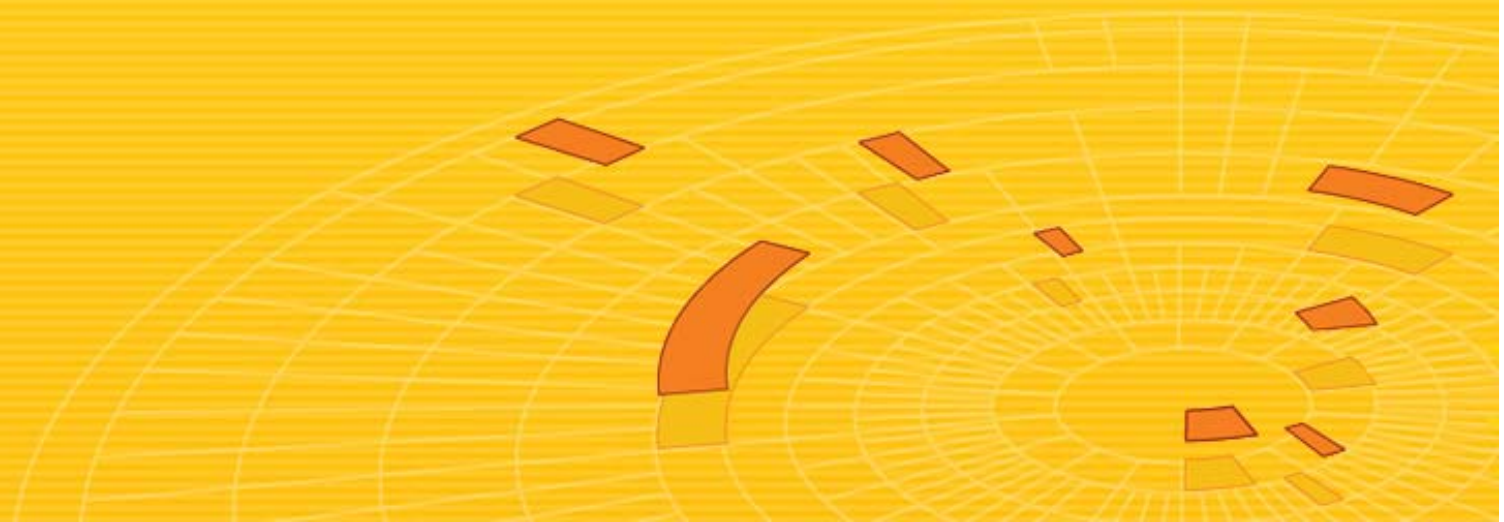


Bis das es nichts mehr zu durchsuchen gibt.
Der Zeitpunkt der letzten Modifikation
und die Position innerhalb des Logfiles
werden wieder gespeichert.

ConSol

Consulting & Solutions

www.consol.de



ConSol
Consulting & Solutions
www.consol.de

08/15 Beispiel

```
$ check_logfiles -V
check_logfiles v2.3
$
$ check_logfiles --logfile=/var/log/messages \
    --tag=0815 --criticalpattern='.*0815.*'
OK - no errors or warnings |0815_lines=0 0815_warnings=0 0815_criticals=0
0815_unknowns=0
$
$ logger "test1 das ist doch 0815"
$ logger "das nicht, weil 0916"
$
$ check_logfiles --logfile=/var/log/messages \
    --tag=0815 --criticalpattern='.*0815.*'
CRITICAL - (1 errors in check_logfiles.protocol-2007-10-10-15-10-02) - Oct
10 15:09:56 localhost lausser: test1 das ist doch 0815  |0815_lines=2
0815_warnings=0 0815_criticals=1 0815_unknowns=0
$
$ echo $?
2
```

ConSol
Consulting & Solutions
www.consol.de

Mehr 08/15 Logging

```
$ logger "gelaber"
$ logger "test2 das ist auch 0815"
$ logger "gefasel"
$
$ check_logfiles --logfile=/var/log/messages \
    --tag=0815 --criticalpattern='.*0815.*'
CRITICAL - (1 errors in check_logfiles.protocol-2007-10-10-15-12-02) - Oct
10 15:11:51 localhost lausser: test2 das ist auch 0815  |0815_lines=3
0815_warnings=0 0815_criticals=1 0815_unknowns=0
$
$ echo $?
2
$
$ logger "geblubber"
$ check_logfiles --logfile=/var/log/messages \
    --tag=0815 --criticalpattern='.*0815.*'
OK - no errors or warnings |0815_lines=1 0815_warnings=0 0815_criticals=0
0815_unknowns=0
$ echo $?
0
```

ConSol
Consulting & Solutions
www.consol.de

Wie sieht die Servicedefinition aus?

```
define service {
  service_description       check_0815msgs
  host_name                 logserver
  max_check_attempts        1
  is_volatile               1
  check_command             check_logfiles!0815!/var/adm/messages!.*0815.*
}

define command {
  command_name check_logfiles_critical
  command_line $USER1$/check_logfiles --logfile=”$ARG2$ --criticalpattern="
$ARG3$" --tag=" $ARG1$"
}
```

max_check_attempts darf nur 1 sein, weil der gleiche Fehler bei einem erneuten Check nicht wieder auftaucht.

ConSol
Consulting & Solutions
www.consol.de

Wie werden überhaupt Rotationen erkannt?

```
$ cat /tmp/check_logfiles._tmp_suelzomat.log.suelz
$state = {
  logoffset => 10,
  logtime => 1182640189,
  devino => '773:510722',
  logfile => '/tmp/suelzomat.log',
};
1;
```

Wie weit haben wir das Logfile gelesen? (= wie groß war es?)
Wann wurde zuletzt was reingeschrieben?
Welche Inode-Nummer hatte es?

Dann schaut man sich das aktuelle Logfile an.
Wie groß ist es und welche Inode-Nr. hat es?

Damit lässt sich rausfinden, ob es eine Rotation gab und welche der wegrotierten Dateien in die Suche einbezogen werden müssen.

ConSol
Consulting & Solutions
www.consol.de

Configfiles

Das war Kinderkram:
check_logfiles --wo? --was?

Configfiles bieten wesentlich mehr :
check_logfiles -f waskompliziertes.cfg

ConSol
Consulting & Solutions
www.consol.de

Voriges Beispiel mit einem 08/15 Configfile

```
$ cat gesuelze.cfg
@searches = ({
    tag             => '0815',
    logfiles        => '/var/log/messages',
    criticalpatterns => '.*0815.*',
    rotation        => 'debian',
    options         => 'noprotocol'
});
$
$ check_logfiles -f gesuelze.cfg
CRITICAL - (4 errors) - error: msg-0815 ... |0815_lines=39
0815_warnings=0 0815_criticals=4 0815_unknowns=0
```

ConSol
Consulting & Solutions
www.consol.de

was bedeutet dieses noprotocol?

Die Treffer im Logfile werden normalerweise in eine Protokolldatei geschrieben.
/tmp/check_logfiles.protocol.<datum-zeit>

$options => “noprotocol”
schaltet das ab.

ConSol
Consulting & Solutions
www.consol.de

Welche Parameter gibt es noch?

```
$ cat gesuelze.cfg
@searches = ({
    tag                 => '0815',
    logfiles            => '/var/log/messages',
    criticalpatterns    => '.*0815.*',
    criticalexceptions  => '.*0815 macht aber nix.*',
    warningpatterns     => ['.*failure.*', '!successful'],
    warningthreshold    => 10,
    okpatterns          => '.*cleared.*',
    rotation            => 'debian',
    options             => 'case,script'
    script              => 'restart_suelzomat'
});
$
```

ConSol
Consulting & Solutions
www.consol.de

Nochmal von Anfang an

- Wir durchsuchen ein oder mehrere Logfiles oder einfach irgendwelche Dateien
- Und zwar nach regular Expressions
- Manche sollen als CRITICAL und manche als WARNING gewertet werden.
- Von jeder Kategorie können mehrere angeben werden, auch solche die auftauchen müssen(!)
- Logfiles werden gelegentlich rotiert, aber auch die wegrotierten (und evt. komprimierten) Dateien werden durchsucht, so daß keine Lücken in der Überwachung entstehen.

ConSol
Consulting & Solutions
www.consol.de

Globale Variablen im Konfigfile

- $protocolsdir – wohin mit den Protokolldateien. (def. /tmp)
- $protocolretention – Maximales Alter von Protokolldateien in Tagen. Nach Ablauf werden sie gelöscht.
- $seekfilesdir – wohin mit den seekfiles (def. /tmp. !!Solaris)
- @searches – Das Kernstück: wonach wird wo gesucht?
- $prescript – Ein Kommando, das ausgeführt wird, bevor die Sucherei anfängt.
- $postscript – dto. Wird ausgeführt, nachdem alle Logfiles durchsucht wurden. Kann Endergebnis bzw. Ausgabe beeinflussen.

ConSol
Consulting & Solutions
www.consol.de

Globale Variablen im Konfigfile II

- $MACROS – Vordefinierte Strings, die man nachher in Dateinamen und Pattern wiederverwenden kann.
  $MACROS = { KAPUTTEPLATTE => 'scsi01lun02' },
  ...
  criticalpatterns => 'SCSI error: ',
  criticalexceptions => '$KAPUTTEPLATTE$'

  Es gibt eine Fülle von Macros, die intern definiert wurden. (Uhrzeit, Hostname, Domain,...)

ConSol
Consulting & Solutions
www.consol.de

Wichtig ist: Wo und was

```
@searches = (
{
  tag => 'lamp-apache'
  logfile => '/var/log/apache/error.log',
  criticalpatterns => ['.*error.*, '.*fatal.'],
  rotation => 'solaris'
},
{
 tag => 'lamp-mysql',
 logfile => '/var/log/mysql.log',
 criticalpatterns => ['corruption', 'you hit a bug']
}
);
```

ConSol
Consulting & Solutions
www.consol.de

Und sowas kommt raus

```
CRITICAL - (1 errors in check_logfiles.protocol-2007-10-10-16-21-09) –
InnoDB: Database page corruption on disk or a failed |lamp-mysql_lines=12
   lamp-mysql_warnings=0 lamp-mysql_criticals=1 lamp-mysql_unknowns=0
```

Wieviele Criticals und Warnings
Wenn der Admin zu faul ist, das Logfile zu durchsuchen, dann gibt's ein Protokollfile, wo die ganzen Fehlermeldungen drinstehen.
Die aktuellste Fehlermeldung steht im Plugin-Output.

ConSol
Consulting & Solutions
www.consol.de

Performancedaten

```
<tag>_lines  die Anzahl der Zeilen, die durchsucht wurden
<tag>_warnings  die Anzahl der gefundenen Warningpatterns
<tag>_criticals  die Anzahl der gefundenen Criticalpatterns
<tag>_unknowns   die Anzahl der gefundenen Unknownpatterns
```

```
z.b: lamp-mysql_lines=8 lamp-mysql_warnings=1 lamp-mysql_criticals=0 lamp-
mysql_unknowns=0 lamp-apache_lines=377 lamp-apache_warnings=0 lamp-
apache_criticals=0 lamp-apache_unknowns=0 lamp-php_lines=19 lamp-
php_warnings=3 lamp-php_criticals=0 lamp-php_unknowns=0
```

ConSol
Consulting & Solutions
www.consol.de

Rotation

Wenn die Durchsuchung von wegrotierten Logfiles gewünscht ist, dann muss man dem check_logfiles eine Hilfestellung geben, wie diese heissen könnten. Dazu gibt es das Feld “rotation” in einer search-definition.
Der Wert dieses Feldes ist entweder eine vordefinierte Rotationsmethode oder ein regulärer Ausdruck.

ConSol
Consulting & Solutions
www.consol.de

Beispiel

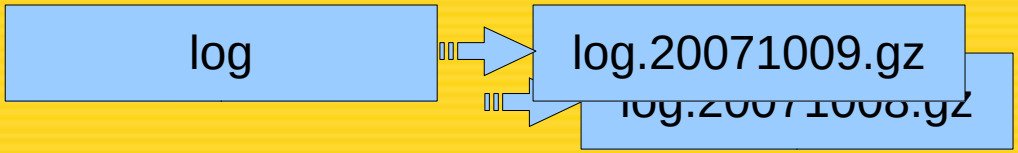
```
$ find /var/log/nagios
/var/log/nagios
/var/log/nagios/archives
/var/log/nagios/archives/nagios-10-04-2007-00.log
/var/log/nagios/archives/nagios-10-05-2007-00.log
/var/log/nagios/archives/nagios-10-06-2007-00.log
/var/log/nagios/nagios.log

@searches = {
   tag => 'naglog',
   logfile => '/var/log/nagios/nagios.log',
   archivedir => '/var/log/nagios/archive',
   rotation => 'nagios\-\d\d\-\d\d\-\d\d\d\d\-\d\d\.log',
   ....
}
```

ConSol
Consulting & Solutions
www.consol.de

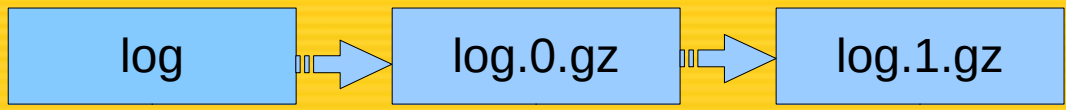
Vordefinierte Rotationen

suse

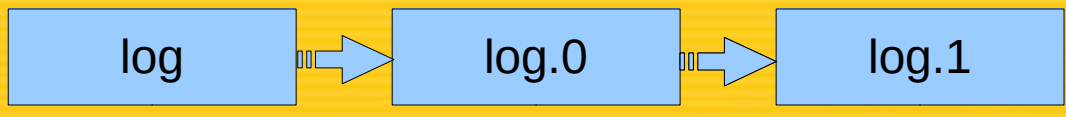


debian

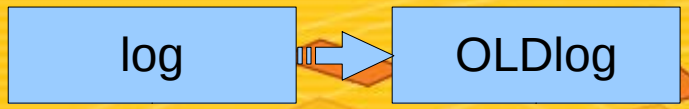
logrotate



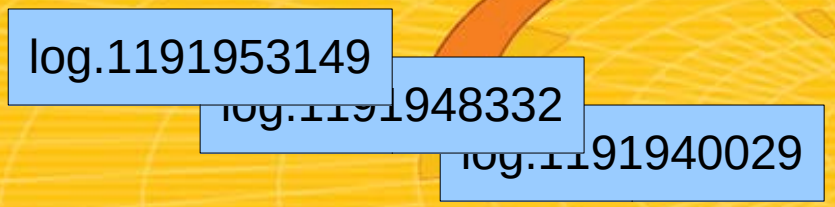
solaris



hpux



mod_log_rotate



ConSol
Consulting & Solutions
www.consol.de

Der Parameter type

- Was ist das überhaupt für eine Art von Logfile?
- Wenn rotation angegeben ist ,dann ist type implizit “rotation”.
- Wenn nicht, dann ist type implizit “simple”. Das nimmt man her, wenn Logfiles einfach gelöscht und neu angelegt werden, oder wenn man in Kauf nimmt, daß die letzten Zeilen in einer wegrotierten Logdatei nicht durchsucht werden.

ConSol
Consulting & Solutions
www.consol.de

Krasse Typen

- type => 'virtual' für Linux-proc-file-device-status-oder-so-Dateien. z.b. /proc/scsi/lun/id/bla/online, wo 0 oder 1 drinsteht. Oder /proc/fcal/host/status, wo “Link: Up” drinsteht. Damit kann man schnell eine Hardwareüberwachung bauen. Diese Logfiles werden jedesmal ganz von Anfang an durchsucht.
- type => 'errpt' zum Durchsuchen des AIX Error Report. Die Ausgabe des errpt-Kommandos wird nach Pattern durchsucht, als wäre sie eine gewöhnliche Logdatei.
- type => 'psloglist' (experimental) zum Durchsuchen des Windows EventLog.

ConSol
Consulting & Solutions
www.consol.de

Weitere Optionen

- noperfdata – Anzeige von Performancedaten wird unterdrückt
- syslogserver – im Logfile stehen Messages, die von allen möglichen Servern stammen. Mit dieser Option werden nur Messages vom lokalen Host untersucht.
- syslogclient – dto., aber diesmal werden nur Messages untersucht, die von einem bestimmten Client stammen. ...,syslogclient=hostname,...
- nologfilenocry – Normalerweise ist es ein Grund für ein UNKNOWN, wenn das Logfile nicht existiert. Mit dieser Option ist es einfach egal.

ConSol
Consulting & Solutions
www.consol.de

Aktionen bei einem Treffer ausführen

Mit der Option script kann man dafür sorgen, daß Code ausgeführt wird, sobald ein Pattern im Logfile gefunden wurde.

script => 'name_eines_programms'

oder seit neuestem

script => sub { perl }

ConSol
Consulting & Solutions
www.consol.de

Warum?

Restart von Applikationen
Versenden von SNMP-Traps
Versenden von NSCA-Messages

Dadurch kann check_logfiles auch als Standalone-Script laufen.

ConSol
Consulting & Solutions
www.consol.de

Nochmal

```
@searches =(
{
  tag => 'suelz',
  logfile => '/var/log/suelzomat.log',
  criticalpatterns => ['ERROR', 'crashed'],
  script => 'restart_suelzomat',
  scriptparams => '--suelzprefix=bla',
  options => 'script'
});
```

ConSol
Consulting & Solutions
www.consol.de

Und wenn das Script fehlschlägt?

Dann muss man das hinnehmen, weil es ein dämliches Script ist.
In jedem Fall ist der Exitcode von check_logfiles CRITICAL.
Auch wenn der Restart geklappt hat und eigentlich wieder alles in Ordnung ist.

ConSol
Consulting & Solutions
www.consol.de

Nehmen wir also das Script ernst

```
@searches =(
{
  tag => 'suelz',
  logfile => '/var/log/suelzomat.log',
  criticalpatterns => ['ERROR', 'crashed'],
  script => 'restart_suelzomat',
  scriptparams => '--suelzprefix=bla',
  options => 'smartscript'
});
```

ConSol
Consulting & Solutions
www.consol.de

Ein smartscript hat einen Exitcode

Und zwar von 0..3 wie man das so kennt. Ein Exitcode 2 von restart_suelzomat wird behandelt, als ob ein criticalpattern im Logfile gefunden worden wäre. Die entsprechende Message ist die erste Zeile des Outputs von restart_suelzomat.

ConSol
Consulting & Solutions
www.consol.de

Was bringt das?

Msg. Suelzomat abgeraucht -> Critical Nr.1
Restart-Script schlägt fehl -> Critical Nr.2
2 Criticals. Na ja, warum nicht?

Msg. Suelzomat abgeraucht -> Critical Nr.1
Restart erfolgreich -> OK
Bleibt immer noch 1 Critical. Suboptimal.

ConSol
Consulting & Solutions
www.consol.de

Siebengescheite Scripts

```
@searches =(
{
  tag => 'suelz',
  logfile => '/var/log/suelzomat.log',
  criticalpatterns => ['ERROR', 'crashed'],
  script => 'restart_suelzomat',
  scriptparams => '--suelzprefix=bla',
  options => 'supersmartscript'
});
```

ConSol
Consulting & Solutions
www.consol.de

Supersmart?

Supersmart Scripts ersetzen mit ihrem Exitcode und Output den auslösenden Treffer im Logfile, anstatt ihn zu ergänzen.

Msg. Suelzomat abgeraucht -> Critical Nr.1
Restart-Script schlägt fehl -> Critical Nr.1 neu
“CRITICAL – restart of suelzomat failed”

Msg. Suelzomat abgeraucht -> Critical Nr.1
Restart erfolgreich -> OK
“OK – restarted suelzmat”

ConSol
Consulting & Solutions
www.consol.de

Pre- und Postscripts

Supersmart Prescripts brechen den Lauf von check_logfiles komplett ab, wenn der Exitcode > 0 ist.

Wenn z.b. Ein Prozess nicht läuft, wozu dann noch im Logfile dieser Applikation nach Fehlern suchen?

ConSol
Consulting & Solutions
www.consol.de

Pre- und Postscripts

Supersmart Postscripts tauschen das Endergebnis von check_logfiles komplett aus, egal wieviele Error Messages vorher gefunden wurden.

ConSol
Consulting & Solutions
www.consol.de

Fehlermeldungen, die einem an der Backe kleben

```
@searches = ({
  tag => 'suelzfull',
  logfile => '/var/log/suelzomat.log',
  criticalpattern => 'oversuelzed',
  okpattern => 'restarted',
  options => 'sticky=36000'
});
```

Check_logfiles liefert so lange CRITICAL, bis die Applikation neu gestartet wurde, bzw. behält den kritischen Zustand 10 Stunden bei.

Sehr, sehr experimentell. Am besten Finger weg!!!!!!

ConSol
Consulting & Solutions
www.consol.de

FIN