

Syslog

Martin Schütte

17. April 2005



Gliederung

Einführung

Sinn und Ziel von Logfiles

Syslog

BSD Syslog

Loggen im Netzwerk

syslog-ng und Tools

syslog-ng

Hilfsprogramme

Analyse

Klassifizierung

automatische Reaktion

Sinn von Logfiles

- Fehlersuche
- Systemauslastung
- Protokollierung/Zurechenbarkeit von Aktivitäten
- Warnung vor laufenden Angriffen
- Untersuchung nach Angriffen

Was und wieviel loggen?

Wieviel?

- lieber mehr als weniger
- Beschränkung durch Plattenplatz, Bandbreite, Analysemöglichkeit
- genug Reservern für Spitzenlast berücksichtigen

Was loggen?

- Änderung der Systemkonfiguration
- Privilegien-Wechsel (login, su, sudo, ...)
- Umgehungsversuche bei Sicherheitsmechanismen
- Zugriff auf vertrauliche Daten

dabei immer: genaue Uhrzeit

Exkurs: Datenschutz

- Logdaten sind i. d. R. personenbezogene Daten
- Zielkonflikt:
 - Zurechenbarkeit bei Mißbrauch
 - Interesse des Einzelnen an Datenschutz/Anonymität
- Bei Zugriff mehrerer Personen:
 - Aufgabe eines Ziels
 - Kompromiss durch Pseudonymisierung

BSD Syslog

- ursprünglich von Eric Allman für sendmail geschrieben
- de facto Standard für Log-Files unter Unix
- Systemweit einheitlicher Log-Mechanismus
- Informationen über Teilsystem und Dringlichkeit
- Loggen in Datei, auf Konsole oder per UDP ins Netz

Facility & Priority

Facility:

0. kernel messages
1. user-level messages
2. mail system
3. system daemons
4. security/authorization messages
5. messages generated internally by syslogd
6. line printer subsystem
7. network news subsystem
8. UUCP subsystem
9. clock daemon
10. security/authorization messages
11. FTP daemon
12. NTP subsystem
13. log audit
14. log alert
15. cron daemon
16. local use 0 (local0)
17. local use 1 (local1)
18. local use 2 (local2)
19. local use 3 (local3)
20. local use 4 (local4)
21. local use 5 (local5)
22. local use 6 (local6)
23. local use 7 (local7)

Priority:

0. Emergency: system is unusable
1. Alert: action must be taken immediately
2. Critical: critical conditions
3. Error: error conditions
4. Warning: warning conditions
5. Notice: normal but significant condition
6. Informational: informational messages
7. Debug: debug-level messages

Aufbau einer Syslog-Zeile

```
<38> Mar 17 21:57:57 frodo sshd[27010]: Connection from 211.174.53.81 port 55991
```

```
<52> Mar 17 13:54:30 192.168.0.42 printer: paper out
```

- PRI: Facility*8 + Priority
- Header
 - Timestamp
 - Hostname
- Message
 - Tag (optional)
 - Content (7-bit ASCII)

Gesamtlänge einer Syslog-Zeile: bis zu 1023 Zeichen

Nachrichten senden

- In Programmen:

```
#include <syslog.h>
openlog("my_tool", LOG_PID, 0);
setlogmask(LOG_UPTO(LOG_INFO));
syslog(LOG_LPR|LOG_ALERT, "printer on fire");
closelog();
```

- Von der Shell:

```
/usr/bin/logger -t my_tool -i -p lpr.alert \  
"printer on fire"
```

syslog.conf (1)

Jede Regel hat das Format: Selektor <Tab> Aktion

Selektoren:

- facility.level
- facility1,facility2.level
- facility1.level1;facility2.level2
- *.level
- *.level;badfacility.none

Verknüpfung mit „oder“. Ausnahme: none deselektiert eine facility.

Aktionen:

- filename in Datei schreiben
- @host an Host weiterleiten
- user1,user2,... Usern aufs Terminal schreiben
- * allen Usern aufs Terminal schreiben

syslog.conf (2)

- BSD Erweiterungen:
 - level-Selektion mit =, =>, =<, !
 - Blöcke für Hosts (+host) oder Programme (!tag)
 - Aktion „| command“ – Meldungen an Befehl pipen
- Linux Erweiterungen:
 - level-Selektion mit =, !
 - Aktion „| /some/fifo“ – Meldungen an benannte Pipe

Beispiel:

```
security,auth.*      /var/log/messages
*.info               @192.168.5.23
```

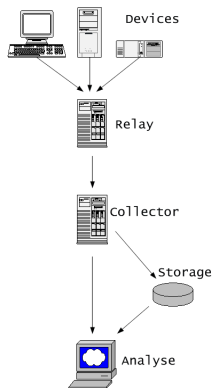
```
*.=notice           /var/log/notice
mail.!=notice       /var/log/messages
```

Beschränkungen von Syslog/Logfiles

Logs sind kein Audit!

- Kein Anspruch auf Vollständigkeit
- Nachrichten können verloren gehen
- Reihenfolge der Nachrichten unsicher
- Alle Programme und User können beliebige Logmeldungen schreiben
- keine Manipulationssicherheit bei lokalen Dateien
⇒ idealerweise auf write-once Medium loggen

Loggen im Netzwerk



verschieden Rollen:

- *Devices* erzeugen Meldungen
- *Relays* empfangen und leiten weiter
- *Collectors* sammeln Meldungen
- *Storage* zur Archivierung
- *Analyzer* zum Auswerten der Meldungen

BSD Syslog über UDP

Gefahren:

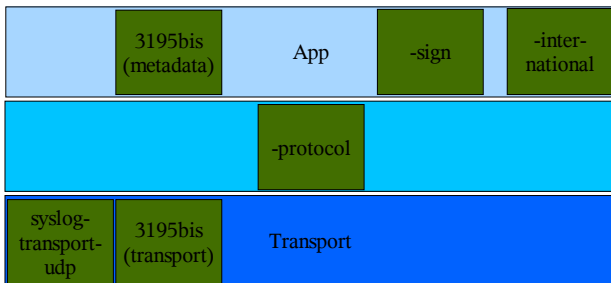
- Paketverlust
- Absenderidentität unsicher
- Fälschen von Paketen
- Überfluten mit falschen Paketen

Vorteile (?):

- einfach und effizient
- „Stealth Logging“ möglich

IETF Working Group

- RFCs 3164 (Syslog) und 3195 (Reliable Syslog)
- aktuelle Drafts:
 - syslog-sign – kryptographisch signierte Nachrichten
 - syslog-protocol – Formaterweiterungen und Schichtenmodell



Reliable Syslog – RFC 3195

- BEEP als Transportprotokoll
- XML-Datensätze
- RAW-Mode ähnlich wie BSD Syslog
- COOKED-Mode:

```
C: MSG 2 3 . 4360 250
C: Content-Type: application/beep+xml
C:
C: <entry facility='24' severity='5'
C:   timestamp='Oct 27 13:24:12'
C:   deviceFQDN='screen.lowry.records.example.com'
C:   deviceIP='10.0.0.47'
C:   pathID='173'
C:   tag='dvd'>
C:     Job paused - Boss watching.
C: </entry>
C: END
```

```
S: RPY 2 3 . 1383 45
S: Content-Type: application/beep+xml
S:
S: <ok/>
S: END
```

Syslog über TCP

- zur Zeit die beste Alternative
- verschiedene Implementierungen
- kein fester Standard – ggf. nicht untereinander kompatibel
- TCP kann mit SSH oder SSL (stunnel) abgesichert werden

syslog-ng

- Alternativer syslogd von Balázs Scheidler
- mehr Filteroptionen
- Logdateien über Makros wählen
- Syslog über TCP
- basiert auf Logpfaden:

```
log { source(s1); source(s2);  
      filter(f1); filter(f2);  
      destination(d1); destination(d2);  
      flags(flag1); };
```

syslog-ng.conf (1)

Sources:

- `file("/dev/kmsg");` Kernel-Nachrichten; kein `tail -f`
- `pipe("/var/tmp/mypipe");` benannte Pipes
- `unix-dgram("/var/run/log");` Sockets
- `udp();` Netzwerk
- `tcp(ip(192.168.1.2) port(1234));` Netzwerk
- `internal();` syslog-ng selbst

Destinations:

- `file("/var/log/$HOST/$YEAR$MONTH/messages");` In Textdatei
- `pipe("/var/tmp/mypipe");` benannte Pipes
- `unix-dgram("/var/tmp/socket");` Sockets
- `tcp("192.168.2.3" port(514));` Netzwerk
- `usertty(root)` Nachricht an User senden
- `program("/bin/logsurfer");` Nachrichten an ein Programm senden

syslog-ng.conf (2)

Filters:

- `facility(mail);`
- `priority(crit..emerg);`
- `host(frodo);` Regex für Hostnamen
- `netmask(192.168.4.0/255.255.255.0);` IP eines Clients
- `program(postfix/.*);` Regex für Tag
- `match(mail);` Regex für Content
- `filter(mein_filter);` Auswerten eines anderen Filters

Flags:

- `final` Nachricht nicht weiterverarbeiten
- `fallback` Nur Nachrichten annehmen, die in keinem andern Pfad matchen
- `catchall` Nachrichten aus allen Quellen verarbeiten

syslog-ng.conf (3)

```
source src { unix-dgram("/var/run/log"); internal(); };
source remote { udp(); tcp(port(514)); };

destination messages { file("/var/log/messages"); };
destination notice   { file("/var/log/notice"); };
destination loghost  { udp("192.168.5.23" port(514)); };

filter f_security { facility(security) or facility(auth); };
filter f_mail     { facility(mail); };
filter f_info     { level(info..emerg); };
filter f_eq_notice { level(notice); };

log { source(src); filter(f_security); destination(messages); };
log { source(src); filter(f_info);      destination(loghost); };

log { source(src); filter(f_eq_notice);
      destination(notice); flags(final); };
log { source(src); filter(f_mail); destination(messages); };
```

SSH

Einfaches Sichern einer TCP-Verbindung über SSH:
Lokales Port-Forwarding von localhost:1514 nach loghost:514

- sshd_config:

```
LocalForward 1514 loghost:514
```

- Shell:

```
ssh -L 1514:loghost:514 loghost
```

stunnel.conf

```
client = yes

cert = /usr/local/etc/my.crt
key = /usr/local/etc/my.key
CAfile = /usr/local/etc/my.cacert
verify = 2

chroot = /var/tmp/stunnel
# PID is created inside chroot
pid = /stunnel.pid
setuid = stunnel
setgid = stunnel

#debug = 7
#output = /var/tmp/stunnel/stunnel.log

# Service-level configuration
[syslog-ssl]
accept = 127.0.0.1:1514
connect = 192.168.5.23:514
```

Logfiles rotieren

Problem:

- Logs sollen lange aufbewahrt werden
- Logdateien sollen nicht unbeschränkt wachsen

Lösung:

```
#!/bin/sh
cd /var/log
mv logfile.2.bz2 logfile.3.bz2
mv logfile.1.bz2 logfile.2.bz2
mv logfile logfile.1
cat /dev/null > logfile
kill -HUP `cat /var/run/syslogd.pid`
bzip2 logfile.1
```

Programme: unter BSD newsyslog, unter Linux logrotate

Beispielkonfiguration

newsyslog.conf:

```
# logfilename      [owner:group] mode count size  when  flags [/pid_file] [sig_num]
/var/log/messages  600    50  2048 $W5D0   JO
/var/log/apache/*.log  www:www 644    2    * $M1D0   GB /var/run/httpd.pid 30
```

/etc/logrotate.conf oder /etc/logrotate.d/syslog:

```
errors sysadmin@my.org
compress
compresscmd /usr/bin/bzip2

/var/log/messages /var/log/localmessages {
    weekly
    rotate 50
    size 2M
    create 640 root root
    postrotate
        /etc/init.d/syslog reload
    endscript
}
```

syslog-ng

Ähnliche Möglichkeiten in syslog-ng

- Jeden Monat neue Datei schreiben

```
destination archiv { file("/var/log/$YEAR/$MONTH"); };
```

- wöchentlich rotieren

```
destination weekly { file("/var/log/messages.$WEEKDAY");  
                    remove_if_older(172800); };
```

Logs normalisieren

Konvertieren verschiedener Log-Formate und -mechanismen zur gemeinsamen Auswertung

Beispiele:

- Recodierung Unicode → ASCII/ISO-8859-x
- Angleichung von Logs verschiedener Implementierungen
- DJBs multilog an Syslog weiterleiten
- Windows Eventlog an Syslog weiterleiten
- SNMP Traps an Syslog weiterleiten

Analyse von Logfiles

99,9% der Logs sind unwichtig

```

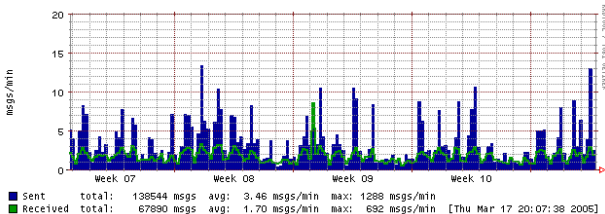
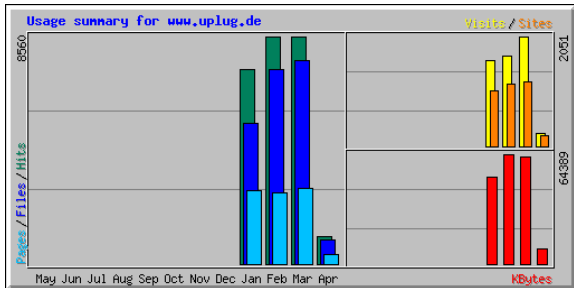
Jul 9 09:37:58 localhost SystemStarter: Willkommen!
Jul 9 09:37:58 localhost lookupd[122]: lookupd (version 324.2.1) starting - Fri Jul 9 09:37:58 2004
Jul 9 09:37:58 localhost ConsoleMessage: Starting Apple Multicast DNS Responder
Jul 9 09:37:58 localhost ConsoleMessage: Starting kernel event agent
Jul 9 09:37:58 localhost ConsoleMessage: Starting timed execution services
Jul 9 09:37:58 localhost ConsoleMessage: Starting SecurityServer
Jul 9 09:38:01 localhost SystemStarter: ?Apple Multicast DNS Responder? wird gestartet
Jul 9 09:38:02 localhost SystemStarter: ?Kernel Event Agent? wird gestartet
Jul 9 09:38:02 localhost SystemStarter: ?Timed Execution Services? werden gestartet
Jul 9 09:38:04 localhost kernel: ApplePMUUserClient::setProperties WakeOnACChange 0
Jul 9 09:38:06 localhost ConsoleMessage: Initializing network
Jul 9 09:38:06 localhost mDNSResponder[155]: mDNSResponder-58.8 (Apr 24 2004 20:38:40) starting
Jul 9 09:38:06 localhost SystemStarter: Starting SecurityServer
Jul 9 09:38:06 localhost SystemStarter: Netzwerkdienste werden initialisiert
Jul 9 09:38:06 localhost ConsoleMessage: Checking disks
Jul 9 09:38:06 localhost SystemStarter: Volumes werden ?berprft
Jul 9 09:38:08 localhost kernel: ATY,Bee_A: vram [9c000000:02000000]
Jul 9 09:38:08 localhost kernel: ATY,Bee_B: vram [98000000:02000000]
Jul 9 09:38:08 localhost syslogd: /dev/console: Input/output error
Jul 9 09:38:08 localhost init: kernel security level changed from 0 to 1
Jul 9 09:38:09 localhost DirectoryService[186]: Launched version 1.8 (v256.6)
Jul 9 09:38:09 localhost loginwindow[182]: Sent launch request message to DirectoryService mach_init port
Jul 9 09:38:16 localhost SystemStarter: Warten auf ?Netzwerkdienste werden initialisiert?
Jul 9 09:38:22 localhost last message repeated 2 times
Jul 9 09:38:23 localhost config[87]: executing /System/Library/SystemConfiguration/Kicker.bundle/Contents/Resources/set-hostname
Jul 9 09:38:24 localhost set-hostname[195]: setting hostname to ivich.local
Jul 9 09:38:25 localhost SystemStarter: Warten auf ?Netzwerkdienste werden initialisiert?
Jul 9 09:38:26 localhost ConsoleMessage: Loading Shared IP extension
Jul 9 09:38:28 localhost SystemStarter: Shared-IP-Erweiterung wird geladen
Jul 9 09:38:28 localhost /usr/libexec/crashreporterd: get_exception_ports() failed: (ipc/send) invalid destination port
Jul 9 09:38:28 localhost ConsoleMessage: Starting Apple File Service
Jul 9 09:38:28 localhost ConsoleMessage: Starting printing services
Jul 9 09:38:29 localhost SystemStarter: ?Apple File Services? werden gestartet
Jul 9 09:38:29 localhost SystemStarter: Die Druckerdienste werden gestartet
Jul 9 09:38:30 localhost ConsoleMessage: Loading IP Firewall extension
Jul 9 09:38:30 localhost SystemStarter: IP-Firewall-Erweiterung wird geladen
Jul 9 09:38:32 localhost kernel: IP packet filtering initialized, divert enabled, rule-based forwarding enabled, default to accept, logging
Jul 9 09:38:32 localhost kernel: IPv6 packet filtering initialized, default to accept, logging disabled
Jul 9 09:38:32 localhost kernel: IP firewall loaded
Jul 9 09:38:32 localhost ConsoleMessage: Starting internet services
Jul 9 09:38:32 localhost SystemStarter: ?Internet Services? werden gestartet
Jul 9 09:38:32 localhost xinetd[257]: xinetd Version 2.3.11 started with libwrap options compiled in.
Jul 9 09:38:32 localhost xinetd[257]: Started working: 3 available services
Jul 9 09:38:33 localhost SystemStarter: Systemstart beendet.
Jul 9 09:38:44 localhost diskarbitrationd[88]: disk1s2 hfs ED021A74-9EEF-3AA5-9B54-5B703F7D0D71 ute [not mounted]
Jul 9 09:38:45 localhost diskarbitrationd[88]: disk1s2 hfs ED021A74-9EEF-3AA5-9B54-5B703F7D0D71 ute /Users/ute

```

grep

- `grep "wichtige Meldung" logdatei | less`
- `grep -v "unwichtige Meldung1" logdatei | \
grep -v "unwichtige Meldung2" | \
grep -v "unwichtige Meldung3" | less`
- ineffizient
- nur interaktiv, nicht automatisch
- kein Auslösen von Aktionen

Ereignisse zählen



logtool

- bunte Logs
- lenkt Aufmerksamkeit auf „wichtige“ Zeilen
- zeigt trotzdem viel Kontext

```

Nov 17 17:29:12 minideb snort: WEB-IIS cmd exe access [1:1002:5] [Web Application Attack] [Pri: 1]: {TCP}
dummyxjack.org(192.168.1.9).3992 -> dummy.xjack.org(192.168.1.2).80
Nov 21 18:39:06 minideb snort: WARNING: Not IPv4 datagram! (snort_decoder) [116:1:1] {TCP}
dummy.xjack.org(192.168.1.1).0 -> dummy.xjack.org(192.168.1.2).0
Nov 10 15:15:39 max snort: Portscan detected from 192.168.1.2: 6 targets 6 ports in 95 seconds (spp_portscan2) [117:1:1]
{ICMP} dummy.xjack.org(192.168.1.2) -> dummy.xjack.org(192.168.1.7)
Nov 7 12:00:00 max snort: Possible RETRANSMISSION detection (spp_stream4) [111:3:1] {TCP} 192.168.9.10.49905 ->
dummy.xjack.org(192.168.1.2).80
Mar 5 03:00:01 192.168.1.9 PAM_unix[16545]: (cron) session opened for user root by (uid=0)
Mar 5 03:00:01 192.168.1.9 logger: Hourly Check In!!!
Mar 5 02:48:09 192.168.1.9 iptables: -j LOG: Bad packet on pub int: {UDP} dummy.xjack.org(192.168.1.1).1028 ->
dummy.xjack.org(192.168.1.2).137
  
```

Grundprinzip: Artificial Ignorance

Ziel:

- Automatisches Filtern der uninteressanten Log-Daten
- schnell auf „ungewöhnliche“ Meldungen aufmerksam werden

Vorgehensweise:

1. vorhandene Logdaten auswerten
2. häufigste normale Meldungen mit Regex ausfiltern
3. das was übrigbleibt regelmäßig ansehen
4. Filter ggf. nachjustieren

logcheck.sh

- Klassifiziert alle Zeilen mit Regex nach
 - hacking
 - violations (ohne violations.ignore)
 - ignore
 - alles übrige
- schickt Report per E-Mail
- merkt sich letzte Position in der Logdatei – dadurch beliebig oft aufrufbar

Swatch

- prüft zeilenweise nach Regex und führt Aktionen aus
- kann als daemon laufen und Logs zeitnah verfolgen

Aktionen:

- `echo bold` Zeile ausgeben
- `bell 3` Terminal piept
- `write root` Zeile auf Terminal schreiben
- `mail addresses=root,subject=swatch` E-Mail senden
- `exec "/bin/programxy"` Programm ausführen
- `pipe "/bin/report"` Zeile an Programm weiterleiten
- `throttle 0:30:00` nur alle 30 min ausführen
- `threshold 5:60` nur ab 5 Ereignissen in 60 Sekunden ausführen
- `continue` auch folgende Regeln testen
- `quit` swatch beenden
- `perlcode` beliebigen Perl-Code ausführen

swatch.conf

Regeln haben die Form:

```
ignore regex
```

```
watchfor regex
```

```
    aktion1
```

```
    aktion2
```

```
    ...
```

Beispiel:

```
watchfor /\,'su root\' failed/
```

```
    echo
```

```
    bell
```

```
    exec echo $0 | mail -s\"security alert\" root\@irgendwo.de
```

logsurfer

wie swatch; außerdem:

- dynamisches Erzeugen/Löschen von Regeln
- Sammeln mehrerer Zeilen in Kontexte
- komplizierte Konfigurationsdatei
- absichtlich keine default-Konfiguration

Aktionen:

- ignore Zeile verwerfen
- exec programm parameter Programm ausführen
- pipe programm Zeile an Programm weiterleiten
- open kontext Kontext öffnen
- delete kontext Kontext löschen
- report programm kontext Kontext an Programm geben
- rule {before,after,top,bottom} regel neue Regel erzeugen

logsurfer.conf

Jede Regel hat die Form:

```
match-regex not-match-regex
  stop-regex not-stop-regex timeout [continue] action
```

Beispiel:

```
'on (/.*): file system full$' - - - 0 continue
  rule top
  "on $2: file system full$" - - - 500 ignore

'on (/.*): file system full$' - - - 0 continue
  pipe "mail -s file system full root"

'on (/.*): file system full$' '/root/mp3sammlung' - - 0
  exec "/usr/bin/find" "$2 -name *.mp3 -exec rm {} ;"

'.*' - - - 0 pipe "/bin/cat"
```

Kontexte

```
'ftpd\[0-9]+\]: \(\?\@\([0-9\.]+\)\) \[INFO\] Neue Verbindung von \1\.$' - - - 0
  open
  "ftpd\[0-9]+\]: \(\.*\@$2\)" - 2000 720 20 ignore

'ftpd\[0-9]+\]: \(\.*\@(\.*)\) \[ERROR\]' - - - 0
  rule before
  "ftpd\\\\[0-9]+\:\\\]: \\(\.*\@(\$2)\)\\\) \\\[INFO\\\] Logout." - - - 300
  report "/bin/cat" "ftpd\[0-9]+\]: \(\.\.*\@$2\)"
```

erzeugte Kontexte:

```
ftpd\[0-9]+\]: (. *@192.168.0.2)
```

ggf. eingefügte dynamische Regeln:

```
ftpd\[0-9]+\]: \(\.*@(192.168.0.2)\) \[INFO\] Logout. -
- - 298
  REPORT "/bin/cat" "ftpd\[0-9]+\]: \(\.\.*\@$2\)"
```

Beschränkungen von logsurfer

- logsurfer kann nicht zählen
- Kontexte benötigen *eine* gemeinsame regex
- kompliziertes Quoting
- kein Reagieren auf Ausbleiben von Meldungen
- flacher Namensraum für Kontexte und Regeln

Spezialwerkzeug ist oft besser:

- Verfolgen eine E-Mail
- Netzwerk-Management/IDS
- Statistiken führen u. visualisieren

Welche Aktionen ausführen?

- Vorsicht bei Aktionen, die neue Logmeldungen generieren
- dito für E-Mails
- Log-Meldungen sind wie Benutzereingaben zu behandeln und zu filtern
- Umgebung, Aktionen und mögliche Auswirkungen müssen eindeutig definiert sein

Links

Informationen

- [LogAnalysis.org](#)
- [IETF Syslog Working Group Home Page](#)
- [Syslog-ng FAQ und Loghost HOWTO](#)
- [Artificial Ignorance: How-To Guide](#)

beschriebene Programme

- [syslog-ng](#)
- [Swatch](#)
- [Logsurfer](#)

weitere Programme

- [Logwatch](#)
- [LoGS](#)
- [SEC - simple event correlator](#)