

# Zentraler Logging-Server auf dem Raspberry Pi

Als Nebenprodukt meines Jobs durfte ich mich in den letzten Wochen mit einigen Logging-Lösungen auseinandersetzen. Im Speziellen traf ich auf [syslog-ng](#), ein weit verbreiteter Systemlogger, der gerne unter Linux eingesetzt wird.

Während man sich in der Regel unter einem Systemlogger aber nun einen kleinen Daemon vorstellt, der alle Systemevents in eine Datei wirft, ist syslog-ng deutlich komplexer. Es eignet sich unter Anderem bestens dazu, einen zentralen Server aufzusetzen, der die Logs sämtlicher Computer im Netzwerk aggregiert.

Im Folgenden werden wir einen **Raspberry Pi** mit aktuellem Arch Linux genau hierzu nutzen. In einem älteren [Artikel](#) haben wir bereits andere Verwendungszwecke für den kleinen Computer mit ARM-CPU vorgestellt.

Zuerst benötigen wir eine Standard-Installation von syslog-ng. Diese ist ganz normal per **pacman** erhältlich.

Sollte syslog-ng bereits installiert sein, meldet der folgende Befehl dies und wir können einfach fortfahren.

Als nächsten Schritt müssen wir syslog-ng mitteilen, dass wir Meldungen von anderen Systemen erwarten und was genau mit diesen passieren soll. Zu Beginn erscheint die Konfiguration überwältigend, ist aber in mehrere Abschnitte aufgeteilt und relativ selbsterklärend.

Da es potentiell multiple Maschinen im Netzwerk gibt, die wir nicht unbedingt vorab kennen, müssen wir im *options* Abschnitt der Datei `/etc/syslog-ng/syslog-ng.conf` sicherstellen, dass die Option *create\_dirs* vorhanden und aktiviert ist.

Als Beispiel hier mein eigener *options* Block:

```
1 options {
2     stats_freq (0);
3     flush_lines (0);
4     time_reopen (10);
5     log_fifo_size (10000);
6     chain_hostnames (on);
7     use_dns (no);
8     use_fqdn (no);
9     create_dirs (yes);
10    keep_hostname (yes);
```

```
11  perm(0640);
12  group("log");
13  };
```

Als nächstes erstellen wir unterhalb der Einstellung *source local* eine Zeile, die dafür sorgt, dass syslog-ng auch Verbindungen von außen akzeptiert. Damit Absender nicht auf ACK Signale warten müssen, nutzen wir UDP statt TCP. Außerdem erstellen wir einen Filter, der ankommende Nachrichten filtert, sodass nur lokale Hosts akzeptiert werden. So sorgen wir dafür, dass niemand von außen die Logs aufbläst (obwohl natürlich anderweitig ebenfalls dafür gesorgt werden sollte, z.B. mit einer Firewall).

```
source remote { udp(ip(0.0.0.0) port(514)); };
...
filter f_remote { netmask("192.168.0.0/255.255.255.0"); };
```

Die eigentliche Arbeit, das Sortieren der Logs nach Hostname, Datum und Typ, findet in der Zielangabe statt:

```
destination d_remote { file("/var/log/remote/$HOST-
$SOURCEIP/$R_YEAR/$R_MONTH/$R_DAY/$FACILITY.log"); };
```

Nun sehen wir auch, warum syslog-ng in der Lage sein sollte, Ordner zu erstellen: Logs sind nicht nur nach Hosts, sondern auch per Datum sortiert. So stellen wir sicher, dass die Dateien niemals zu groß werden. Zudem sind die Logs anhand der Variable *\$FACILITY* in verschiedene Bereiche aufgeteilt; Kernel-, Benutzer-, Authentifizierungsnachrichten und andere Typen werden so unterschieden.

Dies könnte beispielsweise so aussehen:

### *Log Liste*

An das Ende der Konfiguration fügen wir eine Zeile ein, die nun alles zusammenfügt:

```
log { source(remote); filter(f_filter); destination(d_remote); };
```

Nun ist der Server praktisch komplett konfiguriert. Der folgende Befehl sollte syslog-ng stoppen und mit der neuen Konfiguration wieder neu laden:  
(Wenn dies nicht funktioniert, hilft ein Reboot)

```
$ systemctl restart syslog-ng
```

```
[root@Pi1x1] ~# ls -l /var/log/remote/linac-wifi-192.168.0.13/2013/04/03/
total 32
dmesg----- 2 root root 4896 Apr 3 18:32 ..
dmesg----- 4 root root 4896 Apr 3 18:32 ..
----- 1 root Log 2683 Apr 3 18:34 authpriv.log
----- 1 root Log 1338 Apr 3 18:32 dmesg.log
----- 1 root Log 9678 Apr 3 18:45 kern.log
----- 1 root Log 2167 Apr 3 18:43 user.log
[root@Pi1x1] ~#
```

Abschließend müssen wir nur noch unserem Client mitteilen, dass Logs an den Raspberry Pi gesendet werden sollen! Unter OSX ist dies besonders einfach, da der mitgelieferte *syslogd* direkt mit *syslog-ng* kompatibel ist.

Hierzu editieren wir die Datei */etc/syslog.conf* und fügen entweder eine Zeile hinzu oder ersetzen die existierende Zeile. In unserem Beispiel fügen wir eine zweite Zeile hinzu, damit wir sowohl

lokale Logs wie auch solche auf dem Server haben.

```
linac-wifi:~$ cat /etc/syslog.conf
# Note that flat file logs are now configured in /etc/asl.conf
*. * @127.0.0.1:52376
*. * @raspi-ether
linac-wifi:~$
```

### OSX Log Routen

Man beachte, dass in der Regel eine IP an Stelle eines Hostnamens sinnvoller ist. In meinem Fall wird der Name *raspi-ether* jedoch direkt in der Datei */etc/hosts* aufgelöst.

Sobald der Mac nun neu gestartet wird, werden Logs an den Raspberry Pi gesendet.

Um tatsächlich zu sehen, dass Logs an den Server übertragen werden, können wir den Port

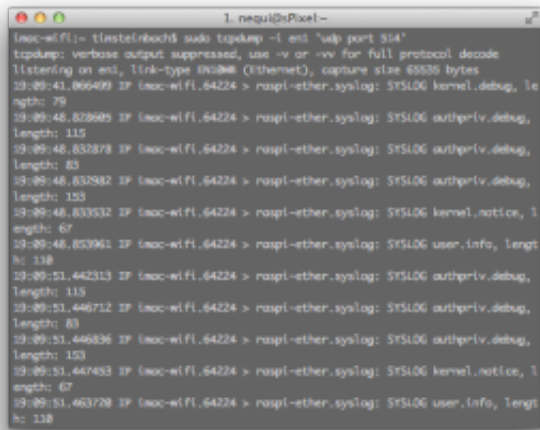
vom Mac aus beobachten:

```
$ sudo tcpdump -i en1 'udp port 514'
```

Dies sieht normalerweise wie folgt aus:

### TCP Dump

Sollten keinerlei Logs übertragen werden – in der Regel werden alle paar Sekunden Informationen ins Log geschrieben – ist höchstwahrscheinlich eine Firewall der Übeltäter. Die Standard-Firewall unter OSX sollte keinerlei Probleme bereiten.



```
1. nequi@PiPixel~$
1. nequi@PiPixel~$ sudo tcpdump -i eth0 'port 314'
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
09-09-41.096499 IP 192.168.1.64224 > rasp1-ether.syslog: SYSLOG kernel.debug, le
ngth: 75
09-09-48.829685 IP 192.168.1.64224 > rasp1-ether.syslog: SYSLOG authpriv.debug,
length: 115
09-09-48.832878 IP 192.168.1.64224 > rasp1-ether.syslog: SYSLOG authpriv.debug,
length: 83
09-09-48.832982 IP 192.168.1.64224 > rasp1-ether.syslog: SYSLOG authpriv.debug,
length: 153
09-09-48.833532 IP 192.168.1.64224 > rasp1-ether.syslog: SYSLOG kernel.notice, l
ength: 67
09-09-48.833961 IP 192.168.1.64224 > rasp1-ether.syslog: SYSLOG user.info, lengt
h: 118
09-09-51.442313 IP 192.168.1.64224 > rasp1-ether.syslog: SYSLOG authpriv.debug,
length: 115
09-09-51.446712 IP 192.168.1.64224 > rasp1-ether.syslog: SYSLOG authpriv.debug,
length: 83
09-09-51.446836 IP 192.168.1.64224 > rasp1-ether.syslog: SYSLOG authpriv.debug,
length: 153
09-09-51.447493 IP 192.168.1.64224 > rasp1-ether.syslog: SYSLOG kernel.notice, l
ength: 67
09-09-51.463728 IP 192.168.1.64224 > rasp1-ether.syslog: SYSLOG user.info, lengt
h: 118
```

Mithilfe dieser Informationen können wir nun sämtliche Linux / Unix / OSX Clients auf dem Raspberry Pi loggen lassen. Microsoft Windows kann ohne zusätzliche Applikation nicht direkt in syslog loggen. Dies ist jedoch ein Thema für einen zukünftigen Artikel 😊