Øvelse 12 – Linux log system

Information

Formålet med denne øvelse er at introducere Linux-logsystemet og præsentere de grundlæggende koncepter. Hver Linux-distribution har sine egne variationer i logsystemet, men grundlæggende fungerer de ens. I dette fag arbejdes der med logsystemet på en Ubuntu-server.

> En log er en registrering af hændelser på et system. Logs bruges til fejlfinding, overvågning og sikkerhedsundersøgelser.

I Linux findes der overordnet to logsystemer: **rsyslog** og **journald**.
rsyslog er en moderne implementering af det klassiske syslog-system.

> Syslog-formatet blev oprindeligt beskrevet i RFC 3164 og senere standardiseret i RFC 5424.

Begge systemer bruges til at håndtere logfiler, men der er forskelle:

- rsyslog skriver logs til filer i klar tekst
- journald håndterer logs via systemd's journal og skriver logs i binært format

I denne øvelse fokuserer vi på rsyslog og logfilerne i Ubuntu (baseret på Debian).

Logfiler i Ubuntu

Systemets logfiler opbevares i mappen `/var/log/`. To af de vigtigste logfiler er:

- **syslog** – Den primære logfil, der indeholder systemets aktiviteter
- **auth.log** – Log over autentificering og sikkerhedshændelser

---

rsyslog vs. journalctl

rsyslog:

- Filbaseret logning
- Skriver tekstlogs til `/var/log/`

- Let at analysere med standard Unix-værktøjer
- Velegnet til integration med eksterne logsystemer (fx SIEM)

journalctl:

- Binær logning via systemd
- Læses med `journalctl`
- Mere metadata (PID, UID, service)
- Avanceret filtrering

Mange moderne Linux-systemer anvender begge mekanismer samtidigt.

Logniveauer i Linux

Logsystemer som **syslog/rsyslog** bruger forskellige **logniveauer** til at angive hvor alvorlig en hændelse er.
Logniveauet hjælper administratorer og sikkerhedsanalytikere med hurtigt at identificere vigtige hændelser i store mængder logs.

De mest almindelige niveauer er:

Niveau	Betydning
debug	Meget detaljerede oplysninger til fejlsøgning
info	Normal systemaktivitet
warning	Noget uventet skete, men systemet fungerer stadig
error	En fejl er opstået
critical	En alvorlig fejl der kan påvirke systemets stabilitet

I praksis bruges logniveauer til at:

- prioritere hvilke hændelser der skal undersøges
- filtrere logs i analyseværktøjer
- identificere potentielle sikkerhedshændelser

Instruktioner

Husk at notere dine observationer i dit Linux cheat sheet.

---

1 Primær logfil: syslog

I dette trin skal du undersøge indholdet af **syslog-filen** og identificere logformatet.

1. Udskriv de seneste 20 linjer af syslog: `tail -n 20 /var/log/syslog`
2. Studér logformatet. En typisk linje ser sådan ud: `Mar 12 10:15:03 hostname process[1234]: This is a log message`
3. I log linjen, identificer følgende: | tidsstempel | værtsnavn | process | logbesked |
4. Lav et generisk Linux-logformat til dit cheat sheet.
5. Find systemets tidszone med kommandoen: `timedatectl`

Nu burde du have et overblik over det generelle format af loglinjer i **syslog**, samt hvordan de kan læses.
Du har også set, hvordan systemets lokale tidszone kan verificeres.

---

2 Authentication log: auth.log

I dette trin skal du undersøge **auth.log**, som indeholder loghændelser relateret til login, autentificering og brug af privilegerede kommandoer.

1. Udskriv de seneste 20 linjer: `tail -n 20 /var/log/auth.log`
2. Identificér brugernavnene i loglinjerne.
3. Skift til root og gentag:

```
sudo -i
tail -n 20 /var/log/auth.log
exit
```

4. Udskriv auth.log igen som almindelig bruger.
5. Hvilke nye linjer er tilføjet?
6. Kan du se autentificeringsforsøget?

Nu burde du kunne identificere login- og autentificeringshændelser i **auth.log**, samt se hvordan brug af privilegerede kommandoer registreres i loggen.

---

3 Filtrering af logs

I dette trin skal du lære at filtrere logfiler for at finde specifikke hændelser.
Logfiler kan hurtigt blive meget store, og derfor er værktøjer som `grep` vigtige til at finde relevante informationer.

1. Søg efter fejlbeskeder i syslog: `grep -i "error" /var/log/syslog | tail -n 10`

   > **Bemærk:**
   > Kommandoen filtrerer kun efter loglinjer der indeholder ordet "**error**".
   > Det betyder ikke nødvendigvis, at loglinjen har logniveauet *error*.
   > Filtreringen er altså baseret på tekstindhold – ikke på syslog-logniveauer.

2. Søg efter autentificeringsfejl i auth.log: `grep -i "failed" /var/log/auth.log | tail -n 10`

Nu burde du kunne bruge simple filtreringsværktøjer til at finde specifikke hændelser i logfiler, såsom fejl eller mislykkede loginforsøg.

---

4 Find system-genstartere

I dette trin skal du undersøge systemets logs for at finde information om tidligere genstarter.
Systemgenstarter registreres i flere forskellige logkilder og kan bruges til at forstå ændringer i systemets tilstand eller administrative handlinger.

> **Tip:**
> Hvis du ikke kan finde nogen genstarter i loggene, kan du selv oprette nogle.
> Prøv at genstarte systemet et par gange med kommandoen `reboot`, og kør derefter øvelsen igen.
>
> Det vil sikre, at der findes relevante loglinjer i både `syslog` og `auth.log`.

1. Søg efter genstarter i syslog: `grep -i "reboot" /var/log/syslog`
2. Notér tidspunkterne for de registrerede genstarter.
3. Identificér hvilken bruger der eventuelt har startet genstarten: `grep -i "sudo" /var/log/auth.log | grep -E "reboot|shutdown"`
4. Alternativt kan du bruge kommandoen: `last reboot`

Nu burde du kunne identificere tidligere systemgenstarter samt finde ud af, hvornår de fandt sted, og om de blev udført af en bruger via administrative kommandoer.

Ved at sammenholde information fra flere forskellige logkilder kan man opbygge en **tidslinje over systemhændelser**.
Denne metode kaldes **logkorrelation** og bruges ofte i fejlfinding, incident response og efterforskning.

5⃣ Brug af journalctl

I dette trin skal du arbejde med **journalctl**, som bruges til at læse logs fra systemd's journal. I modsætning til traditionelle logfiler kan journalctl filtrere logs baseret på services, tidsperioder og logniveauer.

1. Udskriv de seneste logs: `journalctl -n 20`
2. Filtrér logs for SSH-servicen: `journalctl -u ssh --no-pager | tail -n 10`
3. Vis logs siden sidste boot: `journalctl -b`

Nu burde du kunne bruge **journalctl** til at filtrere og analysere systemlogs.
Værktøjet giver mulighed for mere avanceret filtrering end almindelige tekstværktøjer som `grep`, fordi logs indeholder struktureret metadata såsom service, proces-id og logniveau.

6⃣ Filtrering efter logniveau

I dette trin skal du bruge **journalctl** til at filtrere logs efter logniveau.
I modsætning til tekstfiltrering med `grep`, kan journalctl filtrere direkte på systemets registrerede logniveauer.

1. Vis de seneste loglinjer med logniveau **error** eller højere: `journalctl -p err -n 20`
2. Undersøg hvilke typer fejl der optræder i systemets logs.
3. Prøv eventuelt også at vise **warning** eller højere: `journalctl -p warning -n 20`

Nu burde du kunne filtrere logs baseret på **logniveauer**, hvilket gør det lettere at finde kritiske hændelser i store logmængder.

🤔 Refleksion

- Hvad kan du lære om systemets aktivitet ud fra disse logs?
- Hvorfor er logfiler vigtige for fejlfinding og sikkerhed?
- Hvordan kan flere logkilder kombineres til at forstå en systemhændelse?

🔗 Links

Linux Rsyslog Linux Journalctl værktøj RFC 5424 – The Syslog Protocol

Last update: 2026-03-20 13:58:28