

 **ONE IDENTITY™**

Syslog-ng 101, part 10: Parsing

Peter Czanik

Parsing

- Creating name-value pairs from log messages using parsers
- RFC 3164 syslog parsing by default
- Parsers usually work on the MESSAGE macro
- PatternDB for unstructured logs
- JSON, XML, CSV, etc. parsers for structured log messages
- Advantages:
 - More precise filtering (alerting)
 - Save only relevant data

PatternDB parser

- Extracts information from unstructured messages into name-value pairs
- Add status fields based on message text
- Message classification (like LogCheck)
- Needs XML describing log messages
- Example: an ssh login failure:
 - Parsed: app=sshd, user=root, source_ip=192.168.123.45
 - Added: action=login, status=failure
 - Classified as "violation"

JSON parser

- Turns JSON-based log messages into name-value pairs

```
{"PROGRAM":"prg00000","PRIORITY":"info","PID":"1234",  
"MESSAGE":"seq: 00000000000, thread: 0000, runid:  
1374490607, stamp: 2013-07-22T12:56:47 MESSAGE...  
","HOST":"localhost","FACILITY":"auth","DATE":"Jul 22  
12:56:47"}
```

CSV parser

- Parses columnar data into fields

```
parser p_apache {  
  csv-parser(columns("APACHE.CLIENT_IP", "APACHE.IDENT_NAME",  
    "APACHE.USER_NAME",  
      "APACHE.TIMESTAMP", "APACHE.REQUEST_URL", "APACHE.REQUEST_STATUS",  
      "APACHE.CONTENT_LENGTH", "APACHE.REFERER", "APACHE.USER_AGENT",  
      "APACHE.PROCESS_TIME", "APACHE.SERVER_NAME")  
    flags(escape-double-char,strip-whitespace) delimiters(" ") quote-pairs('"'[']')  
    );  
};  
destination d_file { file("/var/log/messages-${APACHE.USER_NAME:-nouser}"); };  
log { source(s_local); parser(p_apache); destination(d_file);};
```

Key=value parser

- Finds key=value pairs in messages
- Introduced in version 3.7.
- Typical in firewalls, like:

```
Aug  4 13:22:40 centos kernel: IPTables-Dropped: IN= OUT=em1  
SRC=192.168.1.23 DST=192.168.1.20 LEN=84 TOS=0x00  
PREC=0x00 TTL=64 ID=0 DF PROTO=ICMP TYPE=8 CODE=0  
ID=59228 SEQ=2
```

```
Aug  4 13:23:00 centos kernel: IPTables-Dropped: IN=em1 OUT=  
MAC=a2:be:d2:ab:11:af:e2:f2:00:00 SRC=192.168.2.115  
DST=192.168.1.23 LEN=52 TOS=0x00 PREC=0x00 TTL=127  
ID=9434 DF PROTO=TCP SPT=58428 DPT=443 WINDOW=8192  
RES=0x00 SYN URGP=0
```

Further parsers

- XML
- Linux Audit
 - /var/log/audit/audit.log
 - MSG often parsed further for extra info
- Date
 - Uses templates
 - Saves to sender date

SCL: syslog-ng configuration library

- Apache access logs
 - Combines CSV and date parsers
- Cisco
 - Cisco logs are similar to syslog messages
 - Can parse many but not all Cisco logs

Python parser

- Released in syslog-ng 3.10
- Parse complex data formats
- Enrich logs from external data sources, like SQL, whois, etc.
- Slower than C
- Does not need compilation or a development environment

Application adapters, Enterprise wide message model

- Application adapters
 - Parse messages automatically
 - Syslog and a growing list of parsers (sudo, Cisco, etc.)
 - Enabled by default from 3.13 for the system() source
- Enterprise wide message model
 - Forward name-value pairs between syslog-ng instances (JSON)
 - Can preserve original message

/etc/syslog-ng/syslog-ng.conf: application adapter

```
@version:3.21
```

```
@include "scl.conf"
```

```
source s_sys { system(); internal();};
```

```
destination d_mesg { file("/var/log/messages"); };
```

```
log { source(s_sys); destination(d_mesg); };
```

```
filter f_sudo {program(sudo)};
```

```
destination d_test {
```

```
    file("/var/log/sudo.json"
```

```
    template("${format-json --scope nv_pairs --scope dot_nv_pairs --scope rfc5424}\n\n");
```

```
};
```

```
log {
```

```
    source(s_sys);
```

```
    filter(f_sudo);
```

```
    if (match("czanik" value(".sudo.SUBJECT"))) {
```

```
        destination { file("/var/log/sudo_filtered"); };
```

```
    };
```

```
    destination(d_test);
```

```
};
```

 **ONE IDENTITY™**