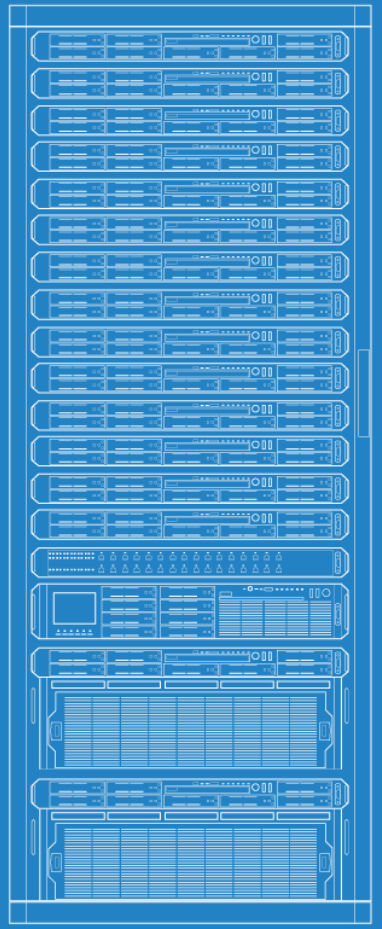
Save All or Save Costs?

Big Data Universe 2018
Peter Czanik / Balabit



BALABIT

ABOUT ME

- Peter Czanik from Hungary
- Evangelist at Balabit: syslog-ng upstream
- syslog-ng packaging, support, advocacy

Balabit is now part of One Identity

It’s business as usual

OVERVIEW

- The “save all” approach to Big Data
- The four roles of syslog-ng
- Parsing & enriching logs (data)
- How to save costs
- Compliance: anonymization

SAVE ALL?

- Big Data vendors often recommend a “save all” and “save raw” approach
- Users often do not know which part of their data is actually useful

- Expectation: storage is practically free compared to value of data

REAL LIFE

Resources are not free

- Hardware and storage
- Software licenses

GOALS

- Reduce the amount of data
- Bring data to a common format
- Process as much as possible in real-time

- syslog-ng can do all of this for logs (and other text data)

syslog-ng

Logging
Recording events, such as:

> Jan 14 11:38:48 linux-0jbu sshd[7716]: Accepted publickey for root from 127.0.0.1 port 48806 ssh2

syslog-ng
Enhanced logging daemon with a focus on portability and high-performance central log collection.

MAIN SYSLOG-NG ROLES

collector

processor

filter

storage
(or forwarder)

ROLE: DATA COLLECTOR

Collect system and application logs together:
contextual data for either side

A wide variety of platform-specific sources:

- /dev/log & co
- Journal, Sun streams

Receive syslog messages over the network:

- Legacy or RFC5424, UDP/TCP/TLS

Logs or any kind of text data from applications:

- Through files, sockets, pipes, application output, etc.

ROLE: PROCESSING

Classify, normalize, and structure logs with built-in parsers:

- CSV-parser, DB-parser (PatternDB), JSON parser, key=value parser, Python parser and more to come

Rewrite messages:

- For example: anonymization

Reformatting messages using templates:

- Destination might need a specific format (ISO date, JSON, etc.)

Enrich data:

- GeoIP
- Additional fields based on message content

ROLE: DATA FILTERING

Main uses:

- Discarding surplus logs (not storing debug-level messages)
- Message routing (login events to SIEM)

Many possibilities:

- Based on message content, parameters, or macros
- Using comparisons, wildcards, regular expressions, and functions
- Combining all of these with Boolean operators

ROLE: DESTINATIONS

“TRADITIONAL”

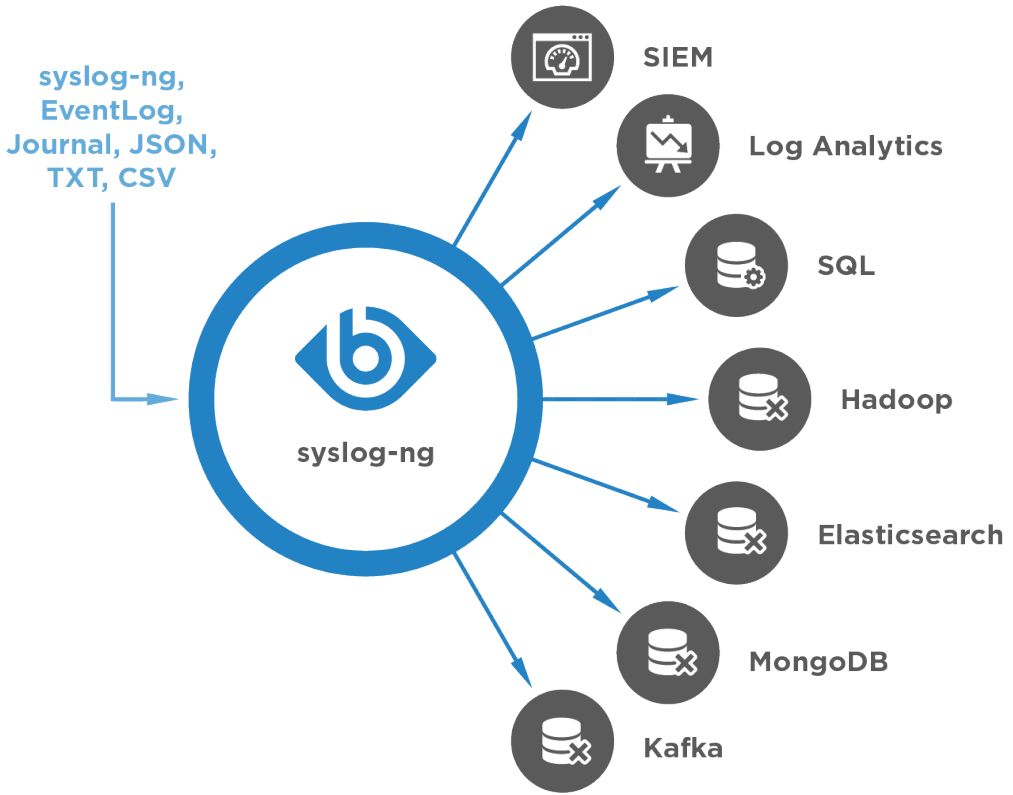
- File, network, TLS, SQL, etc.

“BIG DATA”

- Hadoop
- MongoDB
- Elasticsearch
- Kafka

“OTHERS”

- HTTP(S)
- Java / Python



BALABIT

FREE-FORM LOG MESSAGES

Most log messages are: date + hostname + text

Mar 11 13:37:56 linux-6965 sshd[4547]: Accepted keyboard-interactive/pam for root from 127.0.0.1 port 46048 ssh2

- Text = English sentence with some variable parts
- Easy to read by a human
- Difficult to create alerts or reports

SOLUTION: STRUCTURED LOGGING

- Events represented as name-value pairs
- Example: an ssh login:

  app=sshd user=root source_ip=192.168.123.45
- syslog-ng: name-value pairs inside
  - Date, facility, priority, program name, pid, etc.
- Parsers in syslog-ng can turn unstructured and some structured data (CSV, JSON) into name-value pairs

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: 0000000000, 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

OTHER PARSERS

XML, Linux Audit, Date

XML

Linux Audit

- /var/log/audit/audit.log
- MSG often parsed further for extra info

Date

- Uses templates
- Saves to sender date

SCL

- Combines multiple parsers: Cisco, Apache, etc.

PARSERS WRITTEN IN PYTHON

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

- Jolly Joker :-)

ENRICHING LOG MESSAGES

Additional name-value pairs based on message content

PatternDB

GeoIP: find the geo-location of an IP address

- Country name or longitude/latitude
- Detect anomalies
- Display locations on a map

Add metadata from CSV files

- For example: host role, contact person
- More accurate alerts or dashboards

GOALS

- Reduce the amount of data
- Bring it to a common format
- Process as much as possible in real-time

HOW?

Parsing & Enrichment

- Turn raw data into information
- Facilitate filtering

Reformatting

- Common format, for example JSON
- Ready for further processing

Filtering

- Reduces amount of data

All in real-time with a single application

ANONYMIZING MESSAGES

Many regulations about what can be logged:

- PCI-DSS: credit card numbers
- GDPR: IP addresses, user names

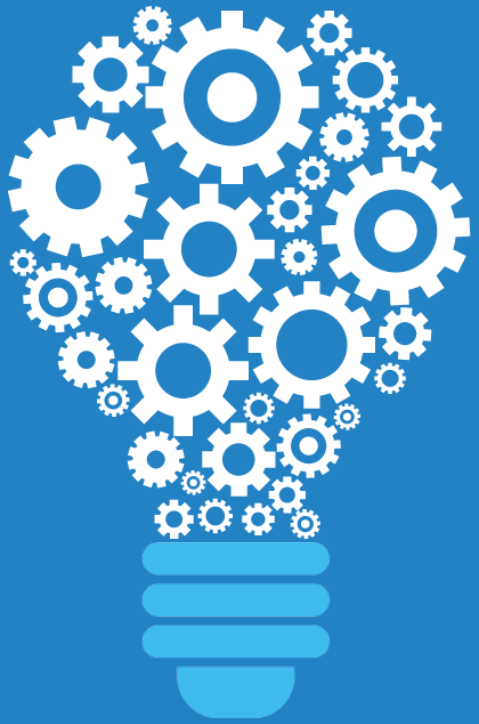
Locating sensitive information:

- Regular expression: slow, works also in unknown logs
- Patterndb, CSV parser: fast, works only in known log messages

Anonymizing:

- Overwrite it with a constant
- Overwrite it with a hash of the original

WHAT IS NEW IN SYSLOG-NG



- Disk-based buffering
- Grouping-by(): generic correlation
- Parsers written in Python
- Elasticsearch REST API support
- HTTP(s) destination
- Wildcard file source
- Performance and memory usage improvements
- Many more :-)

SYSLOG-NG BENEFITS

High-performance reliable log collection

Simplified architecture

Single application for both syslog and application data

Easier-to-use data

Parsed and presented in a ready-to-use format

Lower load on destinations

Efficient message filtering and routing

JOINING THE COMMUNITY

- syslog-ng: http://syslog-ng.org/
- Source on GitHub: https://github.com/balabit/syslog-ng
- Mailing list: https://lists.balabit.hu/pipermail/syslog-ng/
- Gitter: https://gitter.im/balabit/syslog-ng

QUESTIONS?

My blog: https://syslog-ng.com/blog/author/peterczanik/

My e-mail: peter.czanik@balabit.com

Twitter: https://twitter.com/PCzanik

syslog-ng stickers