Logging IoT
Know what your IoT devices are doing

FOSDEM 2018
Peter Czanik / Balabit
ABOUT ME

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

- Balabit is an IT security company with development HQ in Budapest, Hungary
- Over 200 employees: the majority are engineers
- Balabit is now a One Identity company
OVERVIEW

- What is syslog-ng
- The four roles of syslog-ng
- Why structured data
- IoT devices: consumer, networking, industrial
- syslog-ng on the server side
- Configuring syslog-ng
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.
WHY CENTRAL LOGGING?

**EASE OF USE**
one place to check instead of many

**AVAILABILITY**
even if the sender machine is down

**SECURITY**
logs are available even if sender machine is compromised

Diagram showing syslog-ng Relay in Data Center (New York), syslog-ng Central Server at Company HQ (Singapore), and syslog-ng Relay in Data Center (Chicago), with TCP/Encrypted Log Traffic connecting these components.
Why syslog-ng on IoT devices?

- Portable (x86, ARM, POWER, MIPS, etc.)
- Small footprint (written in C)
- Can perform complex processing & filtering
  - Send / save only relevant logs
  - In a ready-to-use format
- Use the same software on the client and server side
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 data from applications:
- Through files, sockets, pipes, etc.
- Application output
ROLE: PROCESSING

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

- CSV-parser, DB-parser (PatternDB), JSON parser, key=value 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”

- Distributed file systems:
  - Hadoop
- NoSQL databases:
  - MongoDB
  - Elasticsearch
- Messaging systems:
  - Kafka
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 search and report on
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, program ID, etc.

- Parsers in syslog-ng can turn unstructured and some structured data into name-value pairs
  - CSV-parser, JSON parser, key=value parser
  - DB-parser (PatternDB),
  - Python parser
WHICH SYSLOG-NG VERSION IS THE MOST USED?

- Project started in 1998
- RHEL EPEL has version 3.5
- Latest stable version is 3.13, released two months ago
Greenland or right-whale, he is the best existing authority. But Scooby knew nothing and says nothing of the great sperm whale, compared with which the Greenland whale is almost unworthy mentioning. And here be it said, that the Greenland whale is an usurper upon the throne of the seas. He is not even by any means the largest of the whales. Yet, owing to the long priority of his claims, and the profound ignorance which, till some seventy years back, invested the then fabulous or utterly unknown sperm-whale, and which ignorance to this present day still reigns in all but some few scientific retreats and whale-portns; this usurpation has been every way complete. Reference to nearly all the leviathanic allusions in the great poets of past days, will satisfy you that the Greenland whale, without one rival, was to them the monarch of the seas. But the time has at last come for a new proclamation. This is Charing Cross; hear ye! good people all,—the Greenland whale is deposed,—the great sperm whale now reigns!

There are only two books in which all pretend to put the living sperm whale before you, and at the same time, in the remotest degree succeed in the attempt: those books are Beale’s and Bennett’s, both in their time surgeons to English South-Sea whale-ships, and both exact and reliable men. The original matter touching the sperm whale to be found in their volumes is necessarily small; but so far as it goes, it is of excellent quality, though
IoT: consumer devices

Where:
- Kindle
- BMW i3 electric car

How:
- Embedded, user is not aware

Why:
- Usage information
- Troubleshooting
IoT: NAS, network devices

Where:
- Synology, FreeNAS, etc.
- Turris Omnia

How:
- Usually just CLI
- Some provide rich GUI

Why:
- Troubleshooting, security
- Central logging for SOHO network
IoT: industrial

Where:
- National Instruments real-time Linux devices
- Control and automation

How:
- Configuration through CLI
- GUI for browsing the logs

Why:
- Troubleshooting
IoT and central logging

Where:
- Car industry
- Smart metering

How:
- Sending log and data through syslog
- Processing and storing to Big Data

Why:
- Usage data
- Troubleshooting
- Metering
CONFIGURATION

- “Don't Panic”
- Simple and logical, even if it looks difficult at first
- Pipeline model:
  - Many different building blocks (sources, destinations, filters, parsers, etc.)
  - Connected into a pipeline using “log” statements
syslog-ng.conf: global options

@version:3.13
@include "scl.conf"

# this is a comment :)  

options {
  flush_lines (0);
  keepHostname (yes);
};
syslog-ng.conf: sources

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

source s_net {
    udp(ip(0.0.0.0) port(514));
};
syslog-ng.conf: destinations

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

destination d_es {
    elasticsearch(
        index("syslog-ng_${YEAR}.${MONTH}.${DAY}")
        type("test")
        cluster("syslog-ng")
        template("$(format-json --scope rfc3164 --scope nv-pairs --exclude R_DATE --key ISODATE)\n"));
    };
}
syslog-ng.conf: filters, parsers

filter f_nodebug    { level(info..emerg); };

filter f_messages    { level(info..emerg) and
    not (facility(mail)
    or facility(authpriv)
    or facility(cron)); };

parser pattern_db {
    db-parser(file("/opt/syslog-ng/etc/patterndb.xml") );
};
syslog-ng.conf: logpath

```conf
log { source(s_sys); filter(f_messages); destination(d_mesg); }

log {
    source(s_net);
    source(s_sys);
    filter(f_nodebug);
    parser(pattern_db);
    destination(d_es);
    flags(flow-control);
}
```
PatternDB & Elasticsearch & Kibana
ANONYMIZING MESSAGES

Many regulations about what can be logged

- PCI-DSS: credit card numbers
- Europe: 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
GeoIP

- parser p_kv { kv-parser(prefix("kv.'"); ); }
- parser p_geoip { geoip( "$\{kv.SRC\}'"", prefix( "geoip." ) database( "/usr/share/GeoIP/GeoLiteCity.dat" ) ); }
- rewrite r_geoip {
  set(
    "$\{geoip.latitude}'"","\{geoip.longitude}'",
    value( "geoip.location" ),
    condition(not "$\{geoip.latitude}'" == '"
  )
  );
}
- log {
  source(s_tcp);
  parser(p_kv);
  parser(p_geoip);
  rewrite(r_geoip);
  destination(d_elastic);
}
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 :-)

BALABIT
SYSLOG-NG BENEFITS FOR IoT AND BIG DATA

High-performance reliable log collection
Simplified architecture
Easier-to-use data
Lower load on destinations

High-performance reliable log collection
Simplified architecture
Easier-to-use data
Lower load on destinations

Single application for both syslog and application data
Parsed and presented in a ready-to-use format
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
SAMPLE XML

- `<?xml version='1.0' encoding='UTF-8'?>`
- `<patterndb version='3' pub_date='2010-07-13'>`
- `<ruleset name='opensshd' id='2448293e-6d1c-412c-a418-a80025639511'>`
- `<pattern>sshd</pattern>`
- `<rules>`
  - `<rule provider="patterndb" id='4dd5a329-da83-4876-a431-ddcb59c2858c" class="system">`
    - `<patterns>`
      - `<pattern>`Accepted @ESTRING:usracct.authmethod: @for @ESTRING:usracct.username: @from @ESTRING:usracct.device: @port @ESTRING:: @@ANYSTRING:usracct.service@</pattern>`
    - `<examples>`
      - `<example>`
        - `<test_message program="sshd">Accepted password for bazsi from 127.0.0.1 port 48650 ssh2</test_message>`
        - `<test_values>`
          - `<test_value name="usracct.username">bazsi</test_value>`
          - `<test_value name="usracct.authmethod">password</test_value>`
          - `<test_value name="usracct.device">127.0.0.1</test_value>`
          - `<test_value name="usracct.service">ssh2</test_value>`
        - `<test_values>`
      - `<example>`
        - `<test_values>`
          - `<test_value name="usracct.type">login</test_value>`
          - `<test_value name="usracct.sessionid">$PID</test_value>`
          - `<test_value name="usracct.application">$PROGRAM</test_value>`
          - `<test_value name="secevt.verdict">ACCEPT</test_value>`
        - `<values>`
          - `<value name="secevt.verdict">ACCEPT</value>`
        - `<values>`
      - `<example>`
      - `<example>`
    - `<examples>`
  - `<rules>`