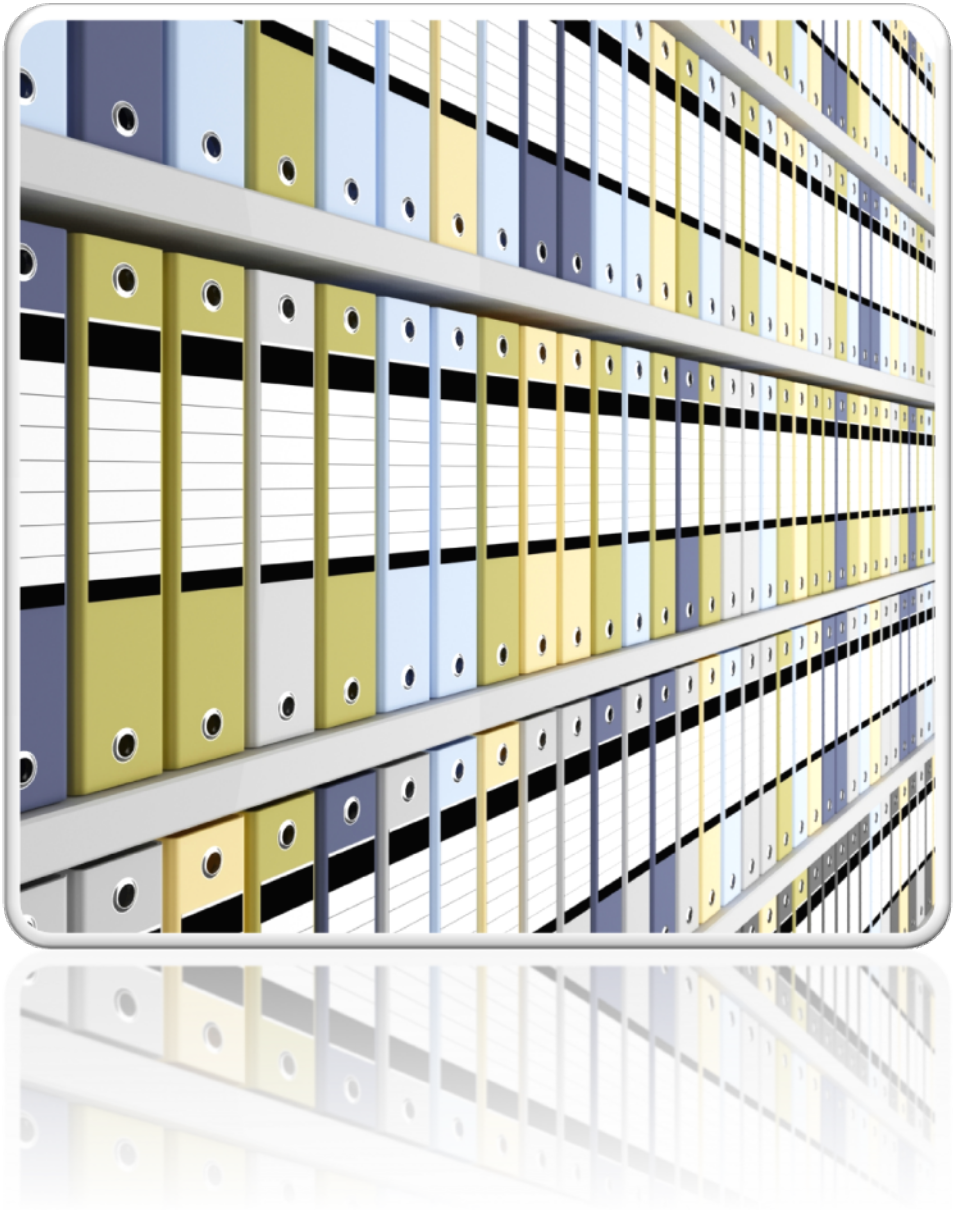
Syslog-ng 3
A step towards log processing



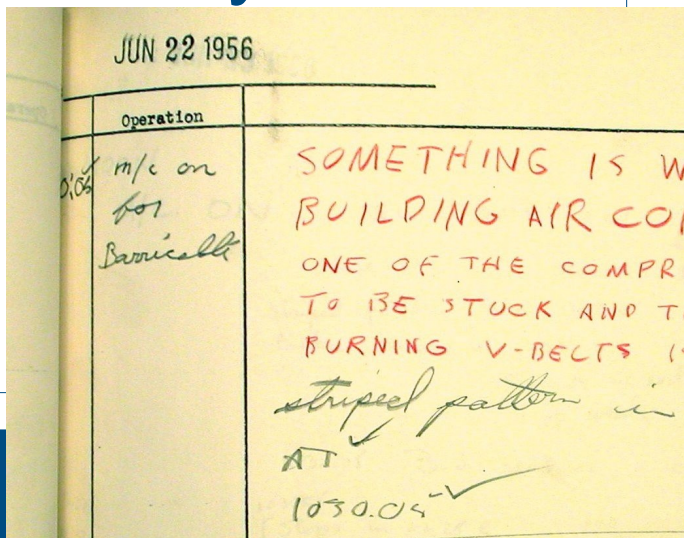
Márton Illés
marton.illes@balabit.com

Contents

- Short introduction to syslog
- The syslog-ng story
- New trends in log collection
- New vision of syslog-ng
- syslog-ng 3.0
- Log processing with syslog-ng

Syslog 101

- Spin-off of sendmail by Eric Allmann
- Describing simple events in plain English
- Easy to use API: syslog()
- Messages are stored in files or sent over the network using UDP transport
- Some application simply store messages directly in files, in SQL database or in proprietary format
- Still the most widespread solution
- Only UNIX and network devices



Problems with the syslog protocol

- No structure at all: hard to parse!
  - Priority and facility is very limited
- Need for central collection, but...
  - No authentication, no encryption, no integrity check, no digital signature
  - No flow-control
  - UDP based transfer with high message loss

```
Jul  3 22:45:21 octane sshd[18206]:
Accepted publickey for marci from 127.0.0.1 port 37126 ssh2
```

The syslog-ng story...

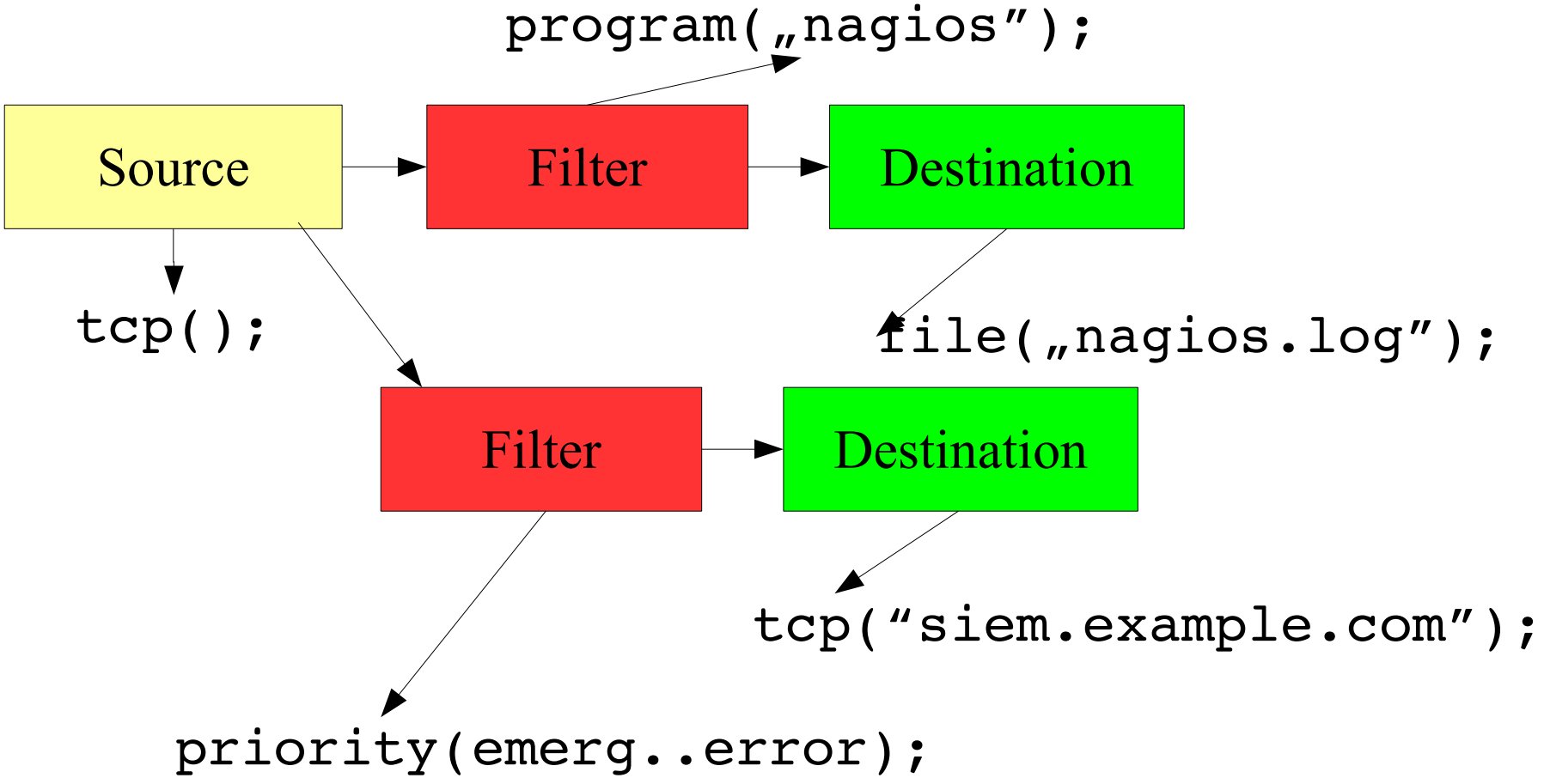
- Designed for central log collection since the beginning
- First release in 1998, now part of most Linux distribution and available for most UNIX flavours
- Operates in multiple global networks serving thousands of devices
- Development funded by BalaBit
  - Open Source Edition, released under GPL
  - Commercial “Premium” and appliance (SSB) editions since 2007/2008

Main features of syslog-ng

- Support for TCP based message transport
  - Understands different syslog flavors (eg: Cisco)
  - Converting between UDP/TCP transports
- Flexible filtering capabilities
- Different, customizable log destinations
  - Message forwarding using TCP
  - File, pipe, program, fifo destinations
  - Utilizing macros and templates
- “Log router” utilizing filters and destinations

The “log router”

Log statement:



New trends in log collection

- Earlier, logs were collected for IT management
  - Troubleshooting, accounting
  - Forensics situations (mainly detective situation)
- The focus and use-cases are changing
  - Security incident and event mgmt. (SIEM)
  - Various regulations
  - Real-time alerting and correlation
  - More messages coming from applications, not just from the infrastructure
- Logs are to be processed automatically

New vision of syslog-ng

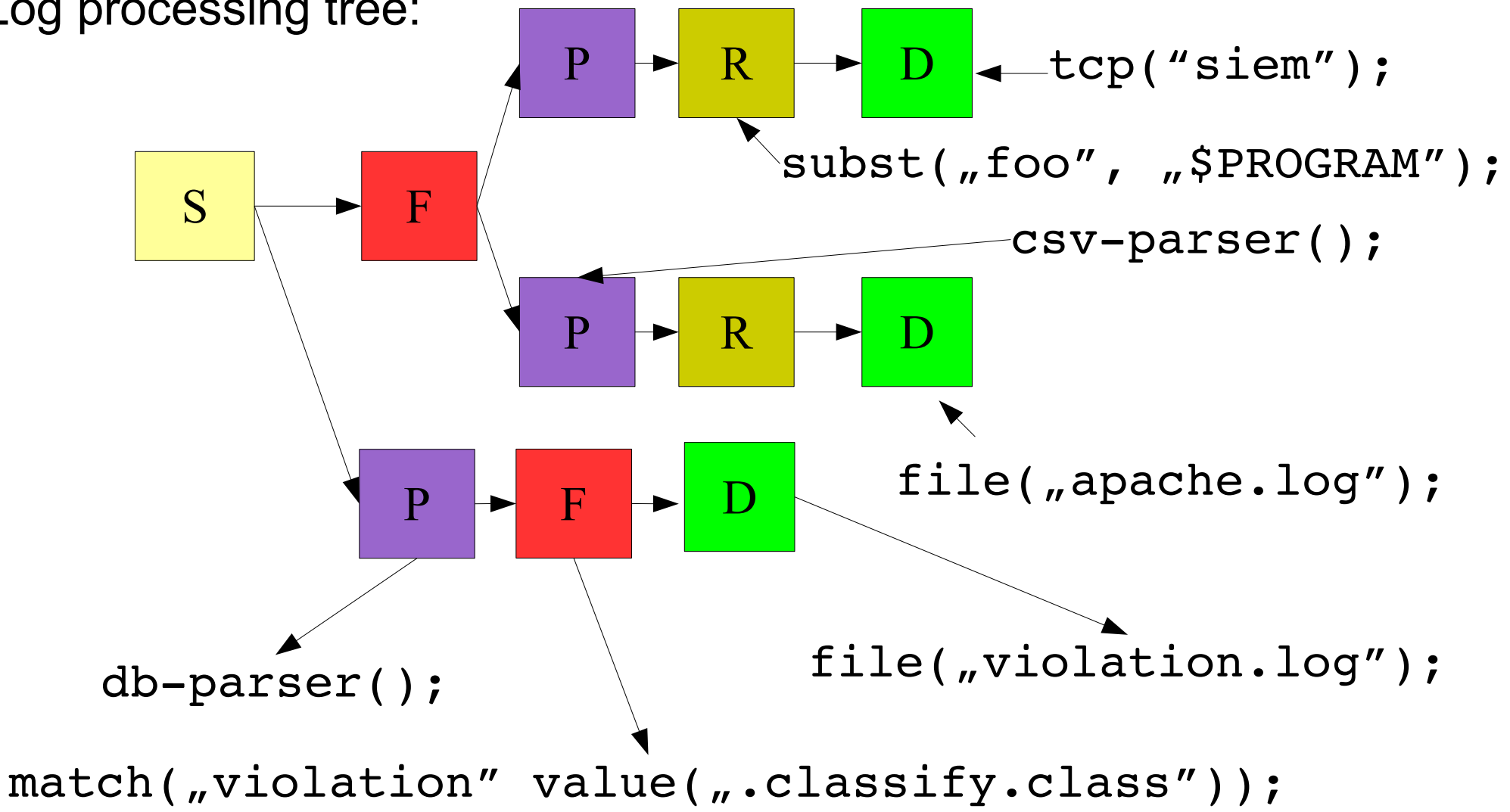
- Acting as a “log router” is not enough anymore
- Syslog-ng needs to aid message analysis
  - Pre-parse message and move them to a common base
  - Extract information from messages
  - Forward messages based on the message content/type/classification
- Syslog-ng is a great integration platform
  - A good position to influence message flow

Syslog-ng 3.0

- Enhanced transport infrastructure
  - Support for new RFC-5426 syslog protocol
  - TLS encrypted transport
- About 70% improved performance over 2.0
- Content related functions
  - Message parsers to extract information into name-value pairs and to classify messages
  - Rewrite framework to fixup messages before analysis
- Native SQL destination support!

The new style “log router”

Log processing tree:



Message parsing

- A parser is an element in the processing tree:
  - Analyzes the content of the message
  - Extracts variable information from messages into name-value pairs which could be used in templates latter
  - Classify/tag messages for further filtering
- Two kind of parsers support as of now: csv and db based
- There are other special requirements for parsing
  - XML based messages (eg: new Cisco IOS logs)
  - New RFC-5426 structured data handling

csv-parser()

- Simple parser to handle “comma separated values”
  - Each column is parsed into name-value pairs
  - Not limited to just “commas”
  - It only recognizes one specific format so messages needs to be filtered before to match the right csv-parser()
- Typical use-cases:
  - Apache, Squid, Nagios logs

Unstructured message parsing

- Parsing unstructured, badly formated messages requires a pattern database
- Most text/message parsing utilizes regular expressions, however...
  - Regexps are hard to write (eg: IPv6 address)
  - Regexps are hard to understand
  - Regexps do not scale to a large number of patterns
  - Regexps do not scale to a high message rate

db-parser()

- Syslog-ng parser to parse messages based on a pattern database
  - Recognize, classify, tag messages
  - Extract information from messages
  - Easy to use unlike the csv-parser()
- Performance:
  - Pattern matching costs about 10-20% of performance relative to storing into files
  - Algorithm is close to O(1) on the number of patterns and depends on the length of the msg

The pattern database and matching

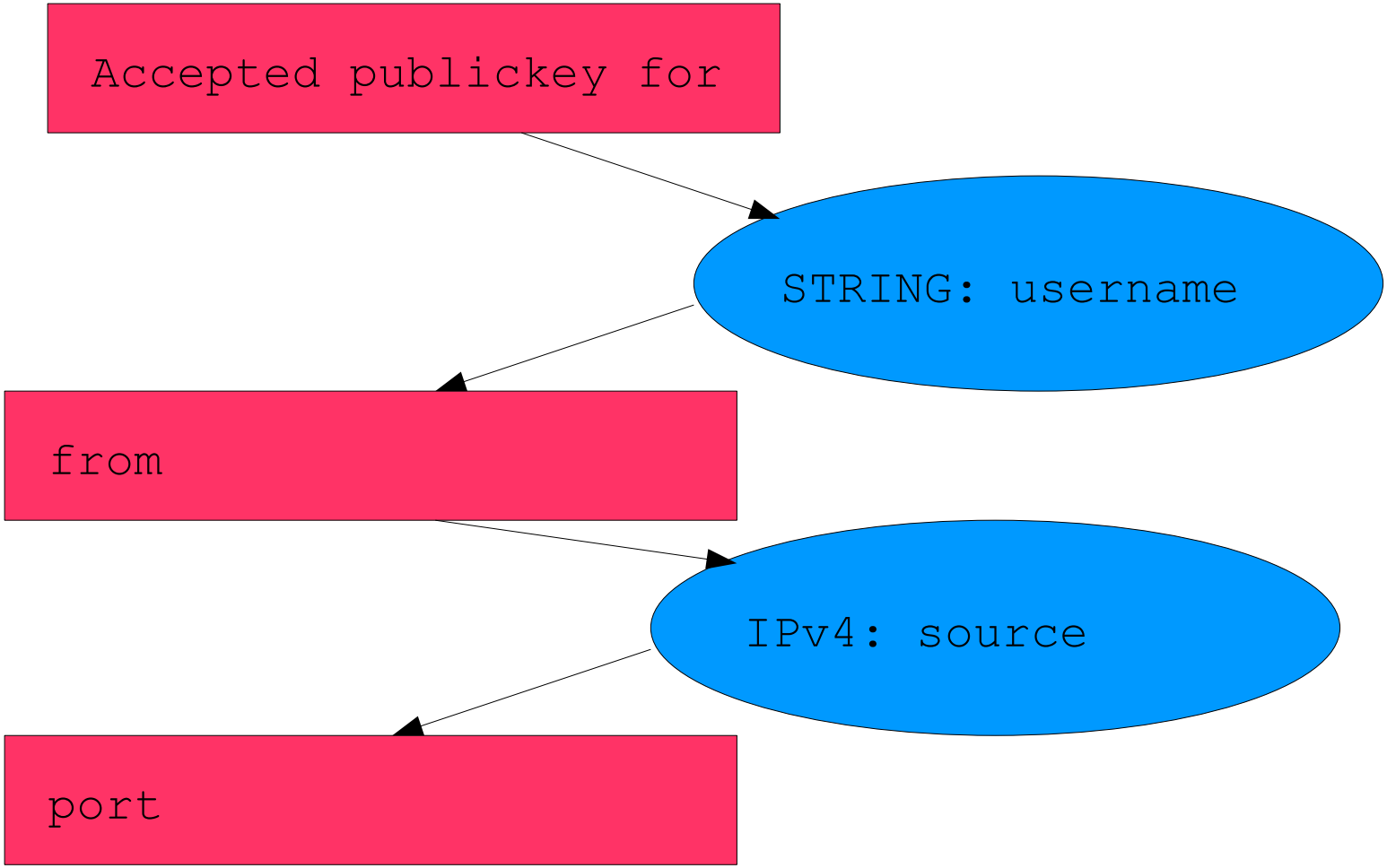
- The on-disk format is XML
- The in-memory format is a radix like tree structure
  - Literal and special “parser” nodes
  - Predefined “parser” nodes to match variable parts:
    - IP addresses (IPv4, IPv6)
    - Strings, quoted-strings, numbers
  - “Parser” matches are stored in name-values
  - Longest prefix matching

Pattern database example


```
<patterndb version="2" pub_date="2009-07-01">
 <ruleset name="sshd">
  <rules>
   <rule id="1" class="login">
    <patterns>
     <pattern>Accepted publickey for @STRING:username@ from
@IPv4:source@ port @NUMBER:port@ ssh2</pattern>
    </patterns>
   </rule>
  </rules>
 </ruleset>
</patterndb>
```

```
destination d_sql {
   sql(type(mysql) host(dbhost) database(logs)
table("login_$R_YEAR_$R_MONTH_$R_DAY) columns("date
timestamp", "username", "source)
values("$R_UNIXTIME", "$username", "$source"));};
```

Pattern database in memory



Pattern database use-cases

- Artificial ignorance with log classification
  - Similar to the “logcheck” project
  - Real-time alerting and reporting
  - “logcheck” converted patterndb is available on BalaBit website
- Extracting information from messages and storing into different customized SQL tables
  - Easier to aggregate and report
- Pre-processing messages for correlation
  - Maybe “in-syslog-ng” correlation one day...

Other noteworthy features in 3.0

- BalaBit supported free binary packages to free UNIX platforms (Linux, *BSDs)
- Support for different character encoding
- Configuration include file support
- Timezone as names (eg: Europe/Budapest)
- PCRE and glob based filters
- Extended statistics framework over unix-socket
- Self-monitoring and automatic restart

Further plans

- Community built pattern database
- Transport improvements: application layer ACKs
- Extended classification with TAGs, TAG-clouds
  - Dynamic SQL schemas based on tags
- New release model
  - Smaller “feature” releases
  - Longer supported “stable” releases
- More transparent development process
  - Public bugzilla, git repo etc.

Summary

- There are severe problems how logging is done today
- More logs are coming from more applications
- Log processing and analysis must be done automatically
  - We need structured log messages
  - CEF, Cisco's XML, RFC-5426 are some steps
- syslog-ng vision has been adjusted
  - Not a mere log transport infrastructure anymore
  - Helping log processing and analysis
- New regexp free message parser

Questions and answers

Thank You for Your Attention!