29th IEEE/ACM International Conference on Program Comprehension

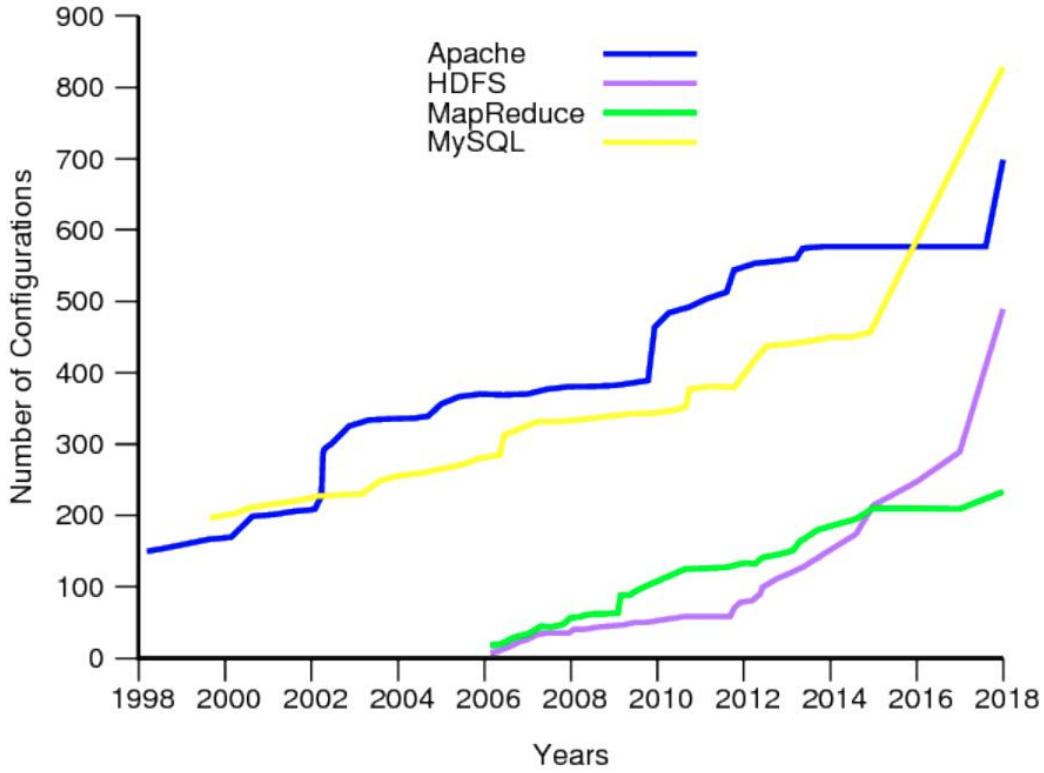
ConfInLog: Leveraging Software Logs to Infer Configuration Constraints

Shulin Zhou, Xiaodong Liu, Shanshan Li, Zhouyang Jia,
Yuanliang Zhang, Teng Wang, Wang Li, Xiangke Liao

National University of Defense Technology, China

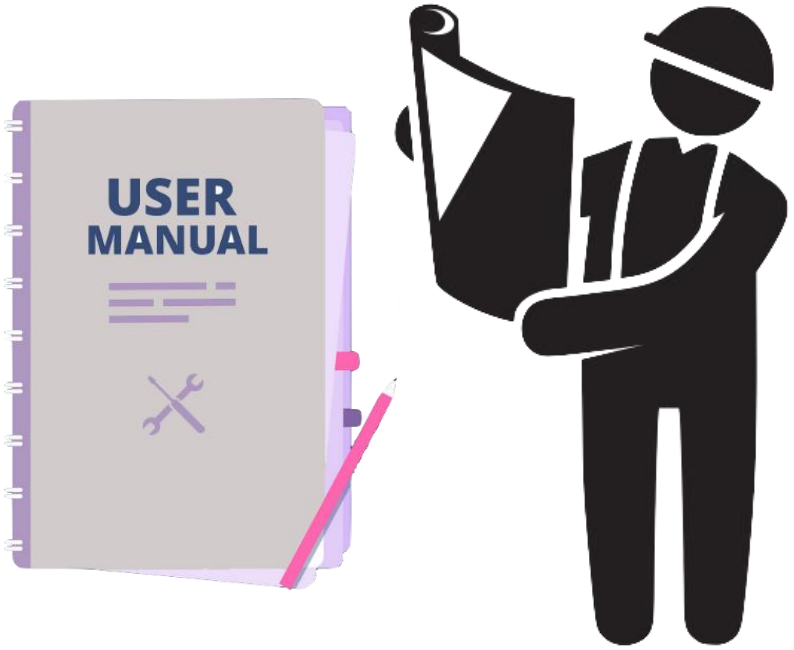
Background

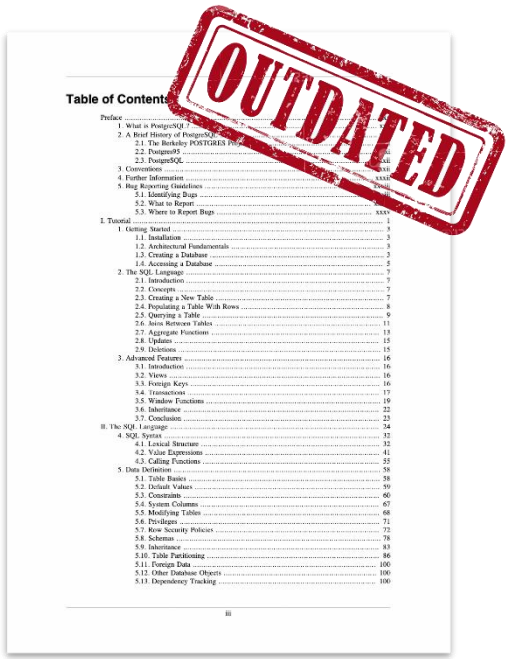
Misconfigurations are Severe and Prevalent



Configuration complexity is ever-increasing

Background





Documents are the main source to comprehend configuration constraints

The documents are often incomplete or outdated

[1] Z. Yin, et.al. An empirical study on configuration errors in commercial and open source systems. **SOSP 2011**
[2] A Rabkin, et.al. Static extraction of program configuration options. **ICSE 2011**
[3] X. Liao, et.al. Do you really know how to configure your software? configuration constraints in source code may help. **Transaction on Reliability 2018**.

Related Work & Challenges

- Inferring configuration constraints from software source code
  - SPEX - SOSP 2013
  - CDep - FSE 2020

Related Work & Challenges

- Inferring configuration constraints from software source code
  - SPEX - SOSP 2013
  - CDep - FSE 2020

Challenge 1

The barrier of pointer usages in static analysis.

Challenge 2

The quality of predefined code patterns that describes constraints.

Motivating Example

```
…
…
48  Mutex default:logs
49
50  VirtualDocumentRoot ../test/
51
52  <IfModule unixd_module>
…     …
```

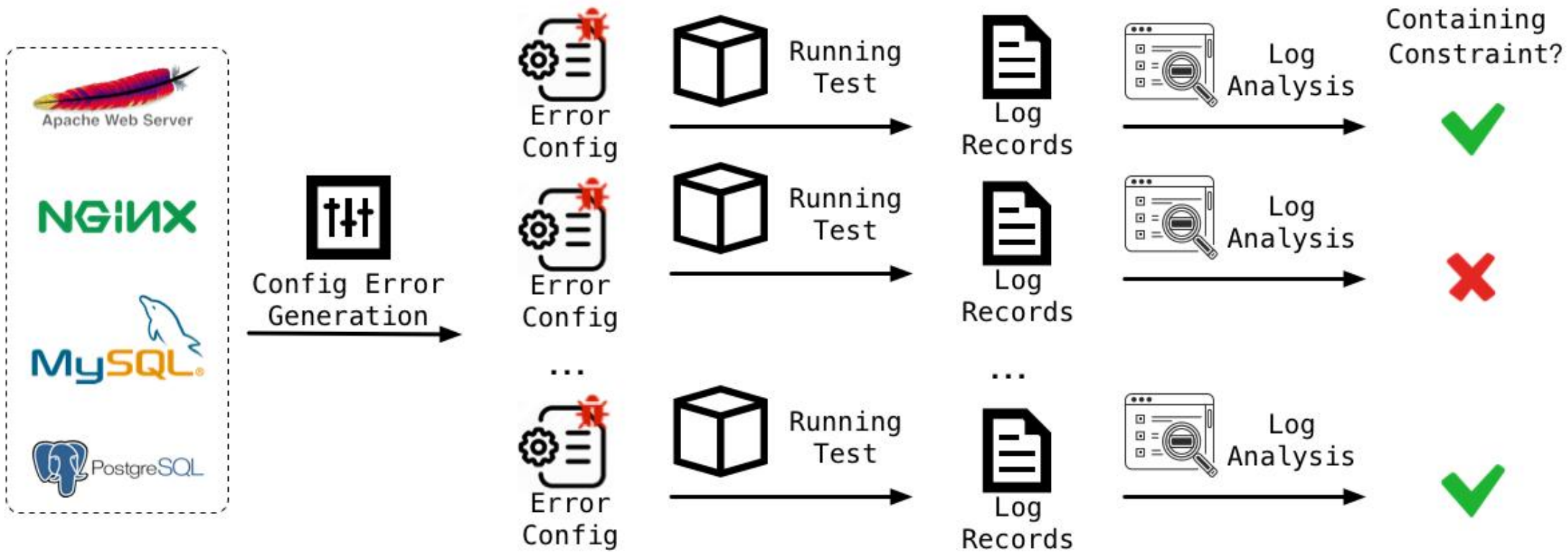
```
AH00526: Syntax error on line 50
of /usr/local/httpd/conf/
httpd.conf:
format string must be an absolute
path, or 'none'
```

```
static const char *vhost_alias_set(..., const
char *map){
   ...
   if(!ap_os_is_path_absolute(cmd->pool, map)){
    if (strcasecmp(map, "none")) {
       return "format string must be an
        absolute path, or 'none'";
    }
   ...
}
```

[1] Log record: *referring to the text outputs at runtime*.

Empirical Study

- Mutation-based Configuration Error Injection



[1] S. Zhang, et.al. Proactive detection of inadequate diagnostic messages for software configuration errors. **ISSTA 2015**

Empirical Study

- Different reaction of configuration options in log records[1].

Software	# of Config Options Under Test	# of Config Options with Related Log Records		
		Total	w/ Constraints	w/o Constraints
Httpd	46	43	34	9
Nginx	28	25	9	16
MySQL	23	23	11	12
PGSQL	58	58	31	27
Total	**155**	**149**	**85**	**64**

[1] Log record: *referring to the text outputs at runtime*.

Empirical Study

- Different reaction of configuration options in log records[1].

Software	# of Config Options	# of Config Options with Related Log Records		
PGSQL	58	58	31	27
Total	**155**	**149**	**85**	**64**

Finding 1: Up to **57.0%** of the configuration options have log records containing constraints description, which provides opportunity to infer configuration constraints.

[1] Log record: *referring to the text outputs at runtime*.

Empirical Study

- Logs in source code:
  - **Log statement**: *Referring to the program statements that perform logging behaviors.*
  - **Log message**: *Referring to the string constants that will be printed by log statements.*

Empirical Study

- How to identify configuration-related log messages:

Directly Match

```
if (ConvertToXSegs(min_wal_size_mb, wal_segment_size) < 2)
   ereport(ERROR, (errcode(ERRCODE_INVALID_PARAMETER_VALUE),
            errmsg("\"min_wal_size\" must be at least twice \"wal_segment_size\"")));
```

Variable Related

```
if (clcf->disable_symlinks == NGX_CONF_UNSET_UINT) {
   ngx_conf_log_error(NGX_LOG_EMERG, cf, 0, "\"%V\" must have \"off\", \"on\" "
                                "or \"if_not_owner\" parameter", &cmd->name);
   return NGX_CONF_ERROR;
}
```

Function Related

```
AP_INIT_TAKE1("AllowEncodedSlashes", set_allow2f, NULL,RSRC_CONF,
  "Allow URLs containing '/' encoded as '%2F'"),

static const char *set_allow2f(cmd_parms *cmd, void *d_, const char *arg){
   .......
   else {
      return apr_pstrcat(cmd->pool, cmd->cmd->name, " must be On, Off, or NoDecode", NULL);
   }
}
```

Empirical Study

- How to identify configuration-related log messages:

Directly Match

```
if (ConvertToXSegs(min_wal_size_mb, wal_segment_size) < 2)
    ereport(ERROR, (errcode(ERRCODE_INVALID_PARAMETER_VALUE),
                errmsg("\"min_wal_size\" must be at least twice \"wal_segment_size\"")));
```

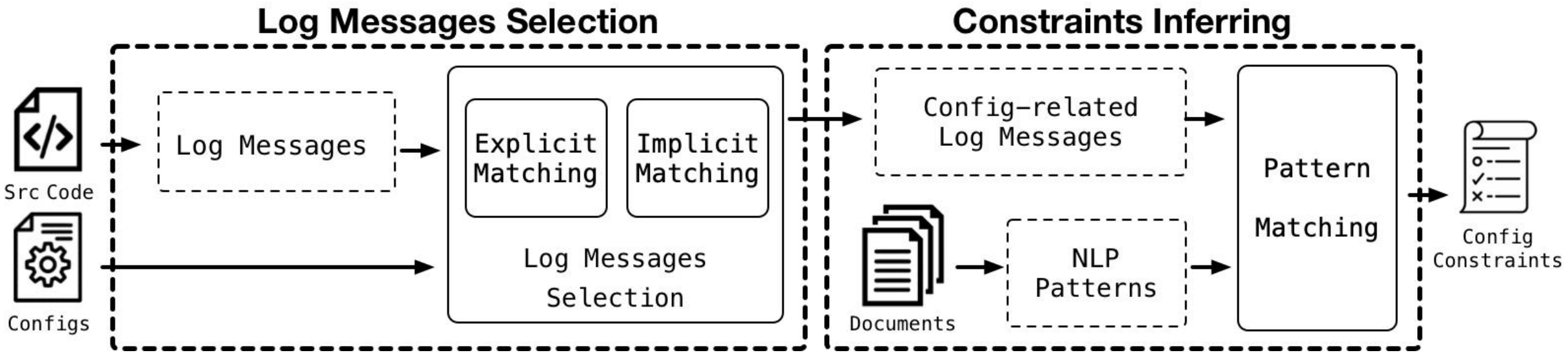
Finding 2: 51.8% of configuration-related log messages could be identified by directly matching option names in log messages, or finding relevant functions or variables.

Function Related

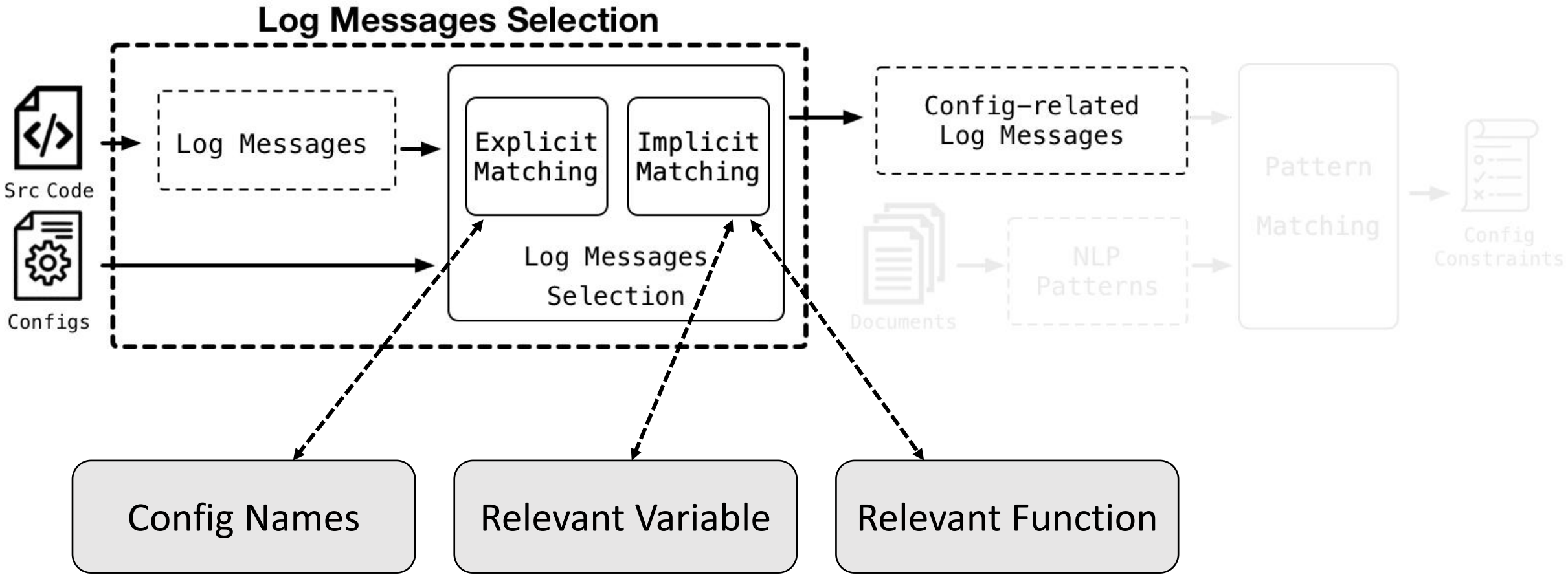
```
  "Allow URLs containing '/' encoded as '%2F'"),

static const char *set_allow2f (cmd_parms *cmd, void *d_, const char *arg){
    .......
    else {
        return apr_pstrcat(cmd->pool, cmd->cmd->name, " must be On, Off, or NoDecode", NULL);
    }
}
```

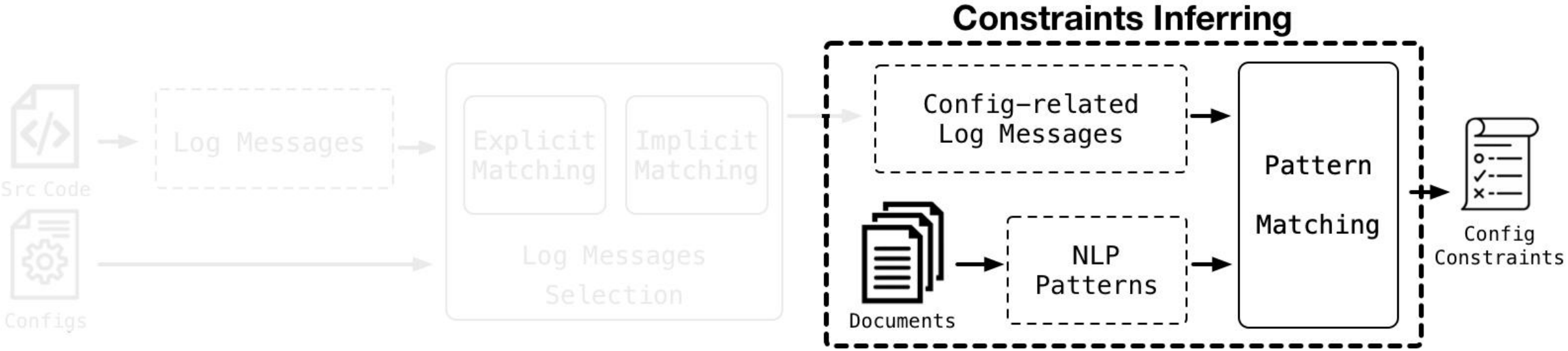
ConfInLog: Configuration Constraints In Log Messages



ConfInLog

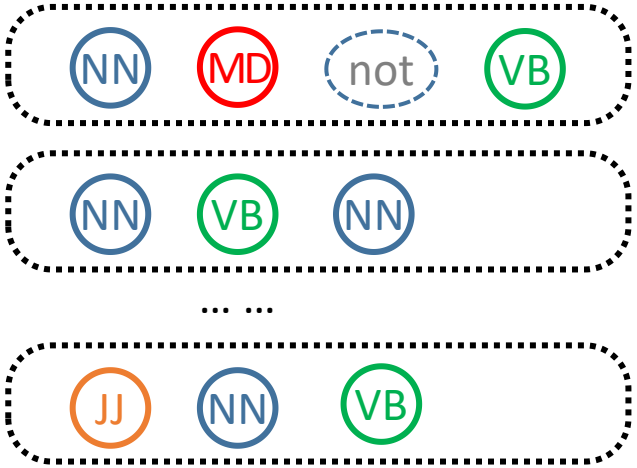

Log Messages Selection
Src Code
Configs
Log Messages
Explicit Matching
Implicit Matching
Log Messages Selection
Config-related Log Messages
Pattern Matching
Config Constraints
Documents
NLP Patterns
Config Names
Relevant Variable
Relevant Function

ConfInLog


Src Code
Log Messages
Configs
Explicit Matching
Implicit Matching
Log Messages Selection
Constraints Inferring
Config-related Log Messages
Documents
NLP Patterns
Pattern Matching
Config Constraints


ALLUXIO
AOLserver
HYPERTABLE INC
hadoop
YARN
cassandra
SUBVERSION
HDFS
APACHE Spark
mongoDB
OPEN LDAP
Lightweight Directory Access Protocol


spaCy


NN
MD
not
VB
NN
VB
NN
... ...
JJ
NN
VB

Evaluation

- Effectiveness of Inferring Configuration Constraints.
- Comparison with the state-of-the-art.

Evaluation - RQ1

- Effectiveness on Inferring Configuration Constraints

Software	# of Config Constraints		
	True Positive	False Positive	False Negative
Httpd	164	41	23
Nginx	80	23	8
MySQL	33	9	10
PGSQL	35	6	7
Lighttpd	63	19	12
Squid	30	13	8
Postfix	22	8	12
Total	**427**	**119**	**80**

Evaluation - RQ1

- Effectiveness on Inferring Configuration Constraints

Software	# of Config Constraints		
	True Positive	False Positive	False Negative
Lighttpd	63	19	12
Squid	30	13	8
Postfix	22	8	12
Total	**427**	**119**	**80**

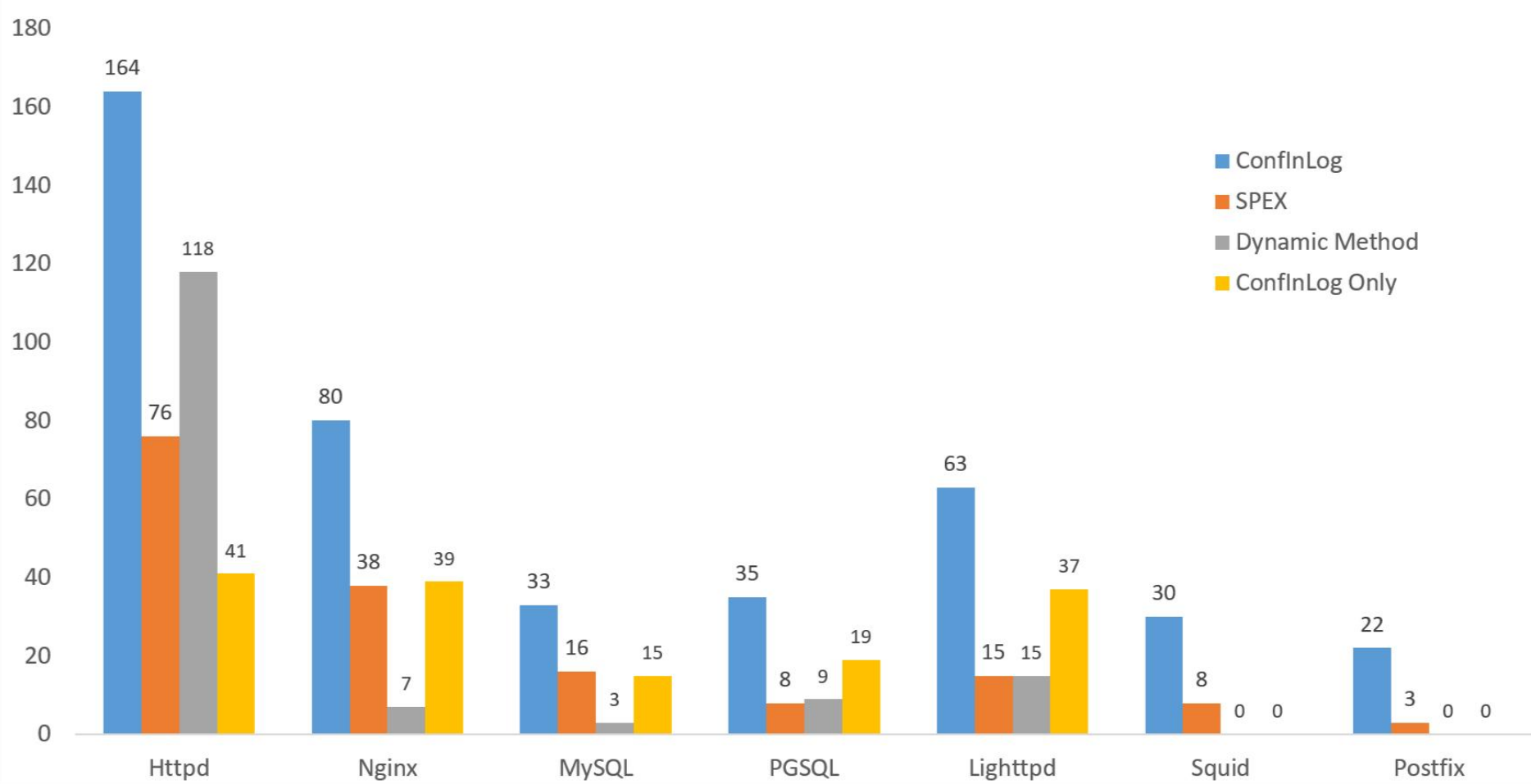
Submitted 67 patches to enhance the documentation. 29 patches have been confirmed by the developers, among which 10 patches have been accepted.

Evaluation - RQ2

- Comparison with the State-of-the-art
  - **SPEX**: Pattern-based constraints inferring tool
  - **Dynamic Method**: Mutation-based configuration error injection method

Evaluation - RQ2

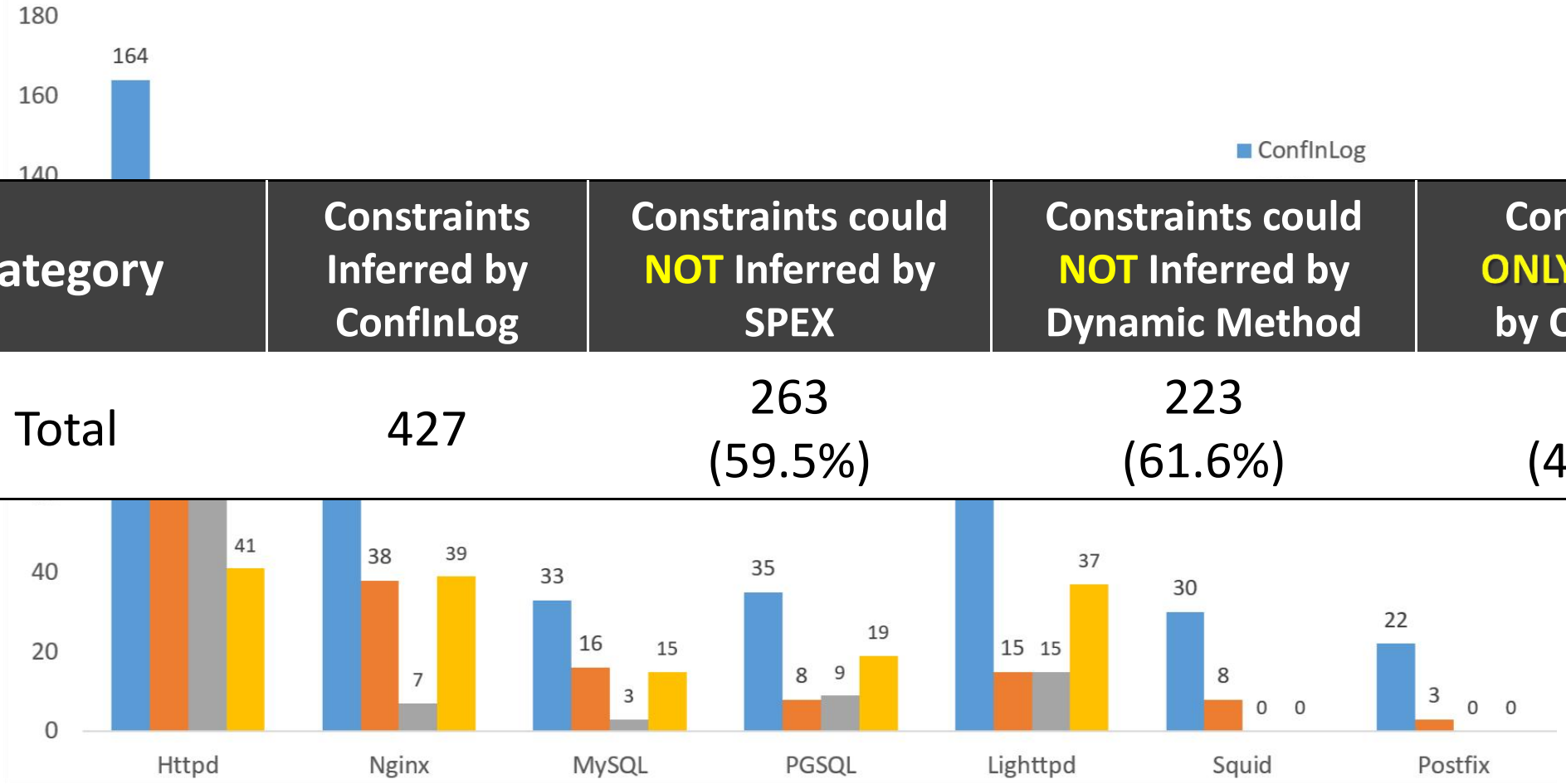
- Comparison with the State-of-the-art



Evaluation - RQ2

- Comparison with the State-of-the-art

Category	Constraints Inferred by ConfInLog	Constraints could NOT Inferred by SPEX	Constraints could NOT Inferred by Dynamic Method	Constraints ONLY Inferred by ConfInLog
Total	427	263 (59.5%)	223 (61.6%)	151 (40.3%)



Conclusion

- We conducted an **empirical study** on 4 popular open-source software systems and summarized patterns of configuration-related log messages.
- We designed and implemented **ConfInLog**, a static tool to infer configuration constraints from log messages.
- We evaluated ConfInLog on 7 software systems. ConfInLog could infer **22~ 163** constraints for the software systems, in which **59.5%~ 61.6%** could not be inferred by the state-of-the-art work.
- We submitted **67 documentation patches** regarding the constraints inferred by ConfInLog.

Thank You