Beyond "Regular" Regular Expressions

Cary Petterborg

Senior Engineer, LDS Church

.conf2016

splunk>

Boiler Plate

Disclaimer

My Disclaimer

During the course of this presentation, I may make references to my employer, the Church of Jesus Christ of Latter-day Saints. This should not be taken as an endorsement of Splunk or Splunk products by the LDS Church.

About Me

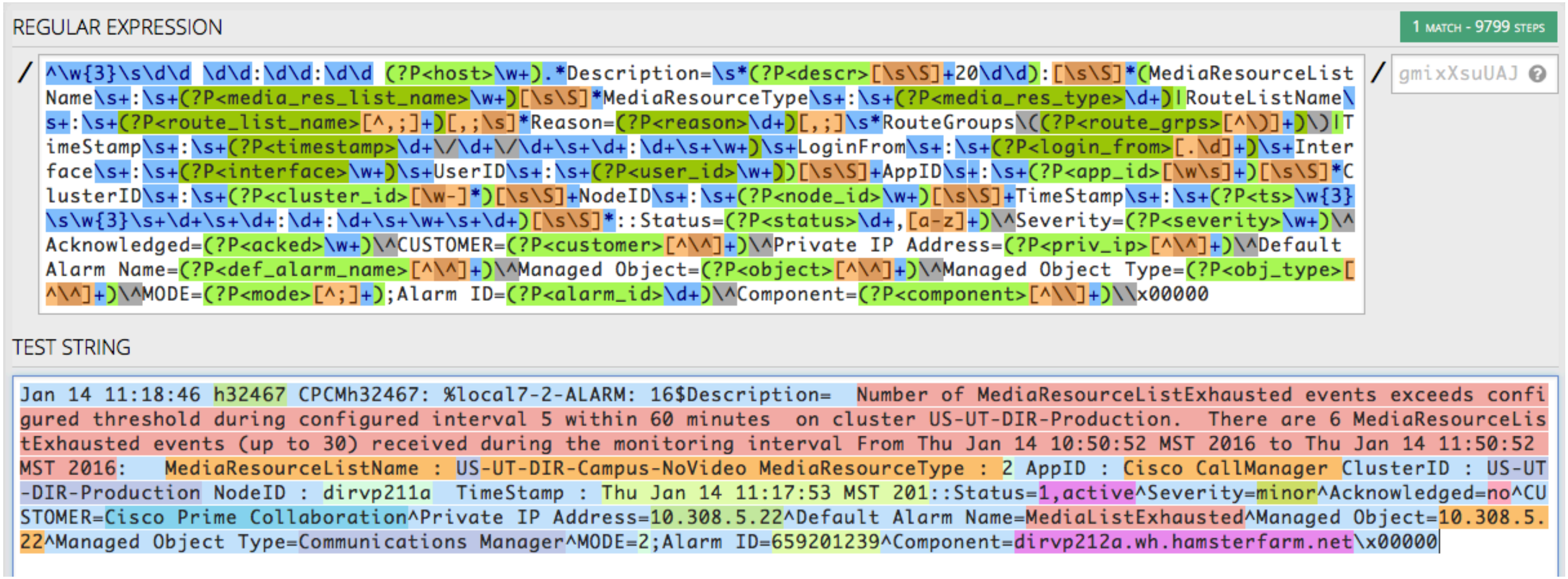
Who is Cary Petterborg?

- Splunk user and administrator for 4.5 years
- Monitoring Engineer for 9 years
- Web developer for 22 years
- Software engineer for 36 years
- Many languages from assembly to Ruby
- Application development including Flight Sim, DB systems, and Web
- Works for the LDS Church in Salt Lake City
- Favorite cologne while in Florida – Deep Woods OFF

Why Regular Expressions?

- Using Regular Expressions since the mid 80’s
- Started using regex with lex/yacc/sed/grep for software development
- Realized the power of regex quickly
- Taught classes on regex
- Love working with regex stuff in Splunk and other utilities
- Regex is an important skill, and I want to share my skill
- While everyone was playing Pokemon GO on their phone this last summer, I was having **fun with regular expressions**

One day, you too…



REGULAR EXPRESSION

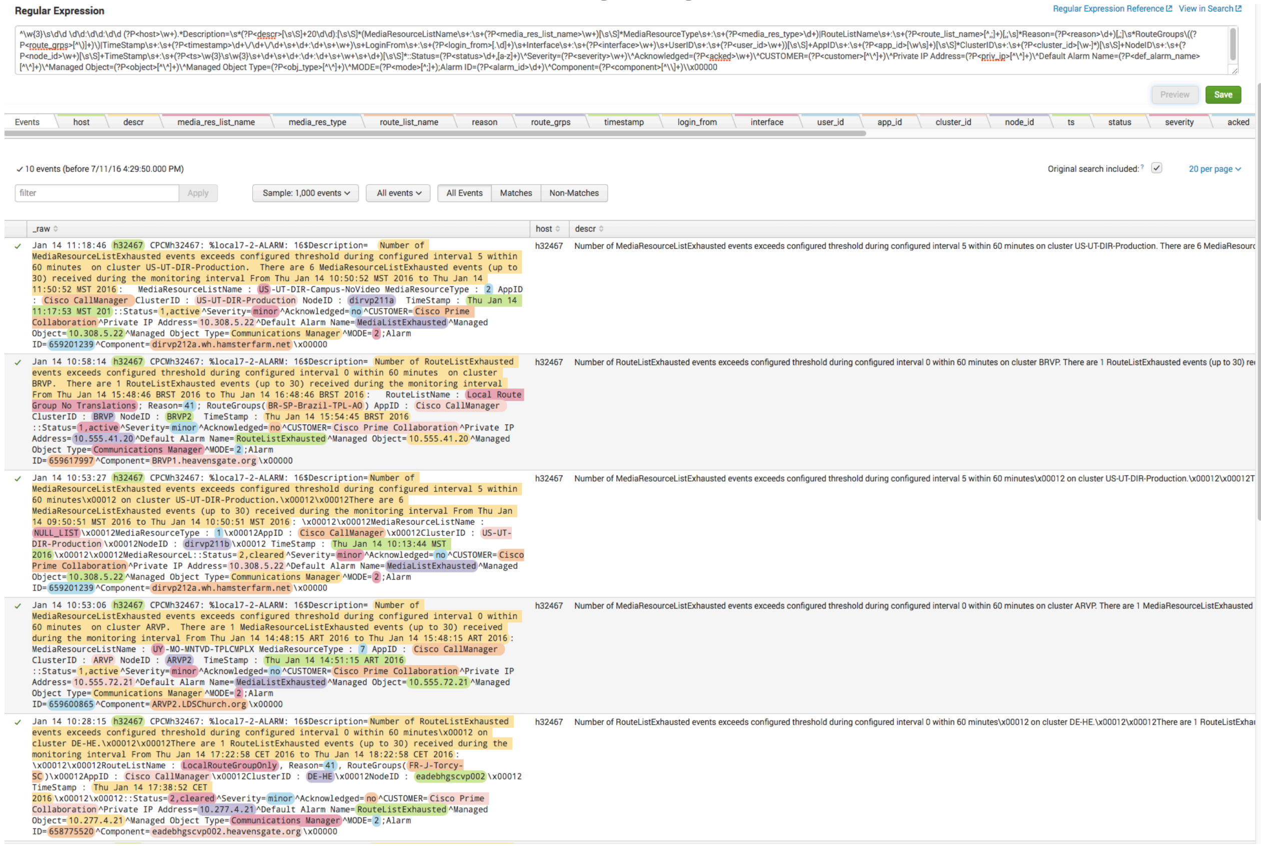
1 MATCH - 9799 STEPS

```
/ ^\w{3}\s\d\d \d\d:\d\d:\d\d (?P<host>\w+).*Description=\s*(?P<descr>[\s\S]+20\d\d):[\s\S]*(MediaResourceList
Name\s+:\s+(?P<media_res_list_name>\w+)[\s\S]*MediaResourceType\s+:\s+(?P<media_res_type>\d+)|RouteListName\
s+:\s+(?P<route_list_name>[^,;]+)[,;\s]*Reason=(?P<reason>\d+)[,;]\s*RouteGroups\((?P<route_grps>[^\)]+)\)|T
imeStamp\s+:\s+(?P<timestamp>\d+\/\d+\/\d+\s+\d+:\d+\s+\w+)\s+LoginFrom\s+:\s+(?P<login_from>[.\d]+)\s+Inter
face\s+:\s+(?P<interface>\w+)\s+UserID\s+:\s+(?P<user_id>\w+))[\s\S]+AppID\s+:\s+(?P<app_id>[\w\s]+)[\s\S]*C
lusterID\s+:\s+(?P<cluster_id>[\w-]*)[\s\S]+NodeID\s+:\s+(?P<node_id>\w+)[\s\S]+TimeStamp\s+:\s+(?P<ts>\w{3}
\s\w{3}\s+\d+\s+\d+:\d+:\d+\s+\w+\s+\d+)[\s\S]*::Status=(?P<status>\d+,[a-z]+)\^Severity=(?P<severity>\w+)\^
Acknowledged=(?P<acked>\w+)\^CUSTOMER=(?P<customer>[^\^]+)\^Private IP Address=(?P<priv_ip>[^\^]+)\^Default
Alarm Name=(?P<def_alarm_name>[^\^]+)\^Managed Object=(?P<object>[^\^]+)\^Managed Object Type=(?P<obj_type>[
^\^]+)\^MODE=(?P<mode>[^;]+);Alarm ID=(?P<alarm_id>\d+)\^Component=(?P<component>[^\\]+)\\x00000 / gmixXsuUAJ
```

TEST STRING

```
Jan 14 11:18:46 h32467 CPCMh32467: %local7-2-ALARM: 16$Description=  Number of MediaResourceListExhausted events exceeds confi
gured threshold during configured interval 5 within 60 minutes  on cluster US-UT-DIR-Production.  There are 6 MediaResourceLis
tExhausted events (up to 30) received during the monitoring interval From Thu Jan 14 10:50:52 MST 2016 to Thu Jan 14 11:50:52
MST 2016:   MediaResourceListName : US-UT-DIR-Campus-NoVideo MediaResourceType : 2 AppID : Cisco CallManager ClusterID : US-UT
-DIR-Production NodeID : dirvp211a  TimeStamp : Thu Jan 14 11:17:53 MST 201::Status=1,active^Severity=minor^Acknowledged=no^CU
STOMER=Cisco Prime Collaboration^Private IP Address=10.308.5.22^Default Alarm Name=MediaListExhausted^Managed Object=10.308.5.
22^Managed Object Type=Communications Manager^MODE=2;Alarm ID=659201239^Component=dirvp212a.wh.hamsterfarm.net\x00000
```

One day, you too…



Regular Expression

Regular Expression Reference View in Search

```
^\w{3}\s\d\d \d\d:\d\d:\d\d (?P<host>\w+).*Description=\s*(?P<descr>[\s\S]+20\d\d):[\s\S]*(MediaResourceListName\s+:\s+(?P<media_res_list_name>\w+)[\s\S]*MediaResourceType\s+:\s+(?P<media_res_type>\d+)|RouteListName\s+:\s+(?P<route_list_name>[^,;]+)[,;\s]*Reason=(?P<reason>\d+)[,;]\s*RouteGroups\((?
P<route_grps>[^\)]+)\)|TimeStamp\s+:\s+(?P<timestamp>\d+\/\d+\/\d+\s+\d+:\d+\s+\w+)\s+LoginFrom\s+:\s+(?P<login_from>[.\d]+)\s+Interface\s+:\s+(?P<interface>\w+)\s+UserID\s+:\s+(?P<user_id>\w+))[\s\S]+AppID\s+:\s+(?P<app_id>[\w\s]+)[\s\S]*ClusterID\s+:\s+(?P<cluster_id>[\w-]*)[\s\S]+NodeID\s+:\s+(?
P<node_id>\w+)[\s\S]+TimeStamp\s+:\s+(?P<ts>\w{3}\s\w{3}\s+\d+\s+\d+:\d+:\d+\s+\w+\s+\d+)[\s\S]*::Status=(?P<status>\d+,[a-z]+)\^Severity=(?P<severity>\w+)\^Acknowledged=(?P<acked>\w+)\^CUSTOMER=(?P<customer>[^\^]+)\^Private IP Address=(?P<priv_ip>[^\^]+)\^Default Alarm Name=(?P<def_alarm_name>
[^\^]+)\^Managed Object=(?P<object>[^\^]+)\^Managed Object Type=(?P<obj_type>[^\^]+)\^MODE=(?P<mode>[^;]+);Alarm ID=(?P<alarm_id>\d+)\^Component=(?P<component>[^\\]+)\\x00000
```

Preview Save

Events | host | descr | media_res_list_name | media_res_type | route_list_name | reason | route_grps | timestamp | login_from | interface | user_id | app_id | cluster_id | node_id | ts | status | severity | acked

✓ 10 events (before 7/11/16 4:29:50.000 PM)

Original search included: ✓ 20 per page

filter Apply | Sample: 1,000 events | All events | All Events | Matches | Non-Matches

	_raw	host	descr
✓	`Jan 14 11:18:46 h32467 CPCMh32467: %local7-2-ALARM: 16$Description= Number of MediaResourceListExhausted events exceeds configured threshold during configured interval 5 within 60 minutes on cluster US-UT-DIR-Production. There are 6 MediaResourceListExhausted events (up to 30) received during the monitoring interval From Thu Jan 14 10:50:52 MST 2016 to Thu Jan 14 11:50:52 MST 2016: MediaResourceListName : US-UT-DIR-Campus-NoVideo MediaResourceType : 2 AppID : Cisco CallManager ClusterID : US-UT-DIR-Production NodeID : dirvp211a TimeStamp : Thu Jan 14 11:17:53 MST 201::Status=1,active^Severity=minor^Acknowledged=no^CUSTOMER=Cisco Prime Collaboration^Private IP Address=10.308.5.22^Default Alarm Name=MediaListExhausted^Managed Object=10.308.5.22^Managed Object Type=Communications Manager^MODE=2;Alarm ID=659201239^Component=dirvp212a.wh.hamsterfarm.net\x00000`	h32467	Number of MediaResourceListExhausted events exceeds configured threshold during configured interval 5 within 60 minutes on cluster US-UT-DIR-Production. There are 6 MediaResourc
✓	`Jan 14 10:58:14 h32467 CPCMh32467: %local7-2-ALARM: 16$Description= Number of RouteListExhausted events exceeds configured threshold during configured interval 0 within 60 minutes on cluster BRVP. There are 1 RouteListExhausted events (up to 30) received during the monitoring interval From Thu Jan 14 15:48:46 BRST 2016 to Thu Jan 14 16:48:46 BRST 2016: RouteListName : Local Route Group No Translations; Reason=41; RouteGroups(BR-SP-Brazil-TPL-AO) AppID : Cisco CallManager ClusterID : BRVP NodeID : BRVP2 TimeStamp : Thu Jan 14 15:54:45 BRST 2016 ::Status=1,active^Severity=minor^Acknowledged=no^CUSTOMER=Cisco Prime Collaboration^Private IP Address=10.555.41.20^Default Alarm Name=RouteListExhausted^Managed Object=10.555.41.20^Managed Object Type=Communications Manager^MODE=2;Alarm ID=659617997^Component=BRVP1.heavensgate.org\x00000`	h32467	Number of RouteListExhausted events exceeds configured threshold during configured interval 0 within 60 minutes on cluster BRVP. There are 1 RouteListExhausted events (up to 30) re
✓	`Jan 14 10:53:27 h32467 CPCMh32467: %local7-2-ALARM: 16$Description=Number of MediaResourceListExhausted events exceeds configured threshold during configured interval 5 within 60 minutes\x00012 on cluster US-UT-DIR-Production.\x00012\x00012There are 6 MediaResourceListExhausted events (up to 30) received during the monitoring interval From Thu Jan 14 09:50:51 MST 2016 to Thu Jan 14 10:50:51 MST 2016: \x00012\x00012MediaResourceListName : NULL_LIST\x00012MediaResourceType : 1\x00012AppID : Cisco CallManager\x00012ClusterID : US-UT-DIR-Production\x00012NodeID : dirvp211b\x00012 TimeStamp : Thu Jan 14 10:13:44 MST 2016\x00012\x00012MediaResourceL::Status=2,cleared^Severity=minor^Acknowledged=no^CUSTOMER=Cisco Prime Collaboration^Private IP Address=10.308.5.22^Default Alarm Name=MediaListExhausted^Managed Object=10.308.5.22^Managed Object Type=Communications Manager^MODE=2;Alarm ID=659201239^Component=dirvp212a.wh.hamsterfarm.net\x00000`	h32467	Number of MediaResourceListExhausted events exceeds configured threshold during configured interval 5 within 60 minutes\x00012 on cluster US-UT-DIR-Production.\x00012\x00012T
✓	`Jan 14 10:53:06 h32467 CPCMh32467: %local7-2-ALARM: 16$Description= Number of MediaResourceListExhausted events exceeds configured threshold during configured interval 0 within 60 minutes on cluster ARVP. There are 1 MediaResourceListExhausted events (up to 30) received during the monitoring interval From Thu Jan 14 14:48:15 ART 2016 to Thu Jan 14 15:48:15 ART 2016: MediaResourceListName : UY-MO-MNTVD-TPLCMPLX MediaResourceType : 7 AppID : Cisco CallManager ClusterID : ARVP NodeID : ARVP2 TimeStamp : Thu Jan 14 14:51:15 ART 2016 ::Status=1,active^Severity=minor^Acknowledged=no^CUSTOMER=Cisco Prime Collaboration^Private IP Address=10.555.72.21^Default Alarm Name=MediaListExhausted^Managed Object=10.555.72.21^Managed Object Type=Communications Manager^MODE=2;Alarm ID=659600865^Component=ARVP2.LDSChurch.org\x00000`	h32467	Number of MediaResourceListExhausted events exceeds configured threshold during configured interval 0 within 60 minutes on cluster ARVP. There are 1 MediaResourceListExhausted
✓	`Jan 14 10:28:15 h32467 CPCMh32467: %local7-2-ALARM: 16$Description=Number of RouteListExhausted events exceeds configured threshold during configured interval 0 within 60 minutes\x00012 on cluster DE-HE.\x00012\x00012There are 1 RouteListExhausted events (up to 30) received during the monitoring interval From Thu Jan 14 17:22:58 CET 2016 to Thu Jan 14 18:22:58 CET 2016: \x00012\x00012RouteListName : LocalRouteGroupOnly, Reason=41, RouteGroups(FR-J-Torcy-SC)\x00012AppID : Cisco CallManager\x00012ClusterID : DE-HE\x00012NodeID : eadebhgscvp002\x00012 TimeStamp : Thu Jan 14 17:38:52 CET 2016\x00012\x00012::Status=2,cleared^Severity=minor^Acknowledged=no^CUSTOMER=Cisco Prime Collaboration^Private IP Address=10.277.4.21^Default Alarm Name=RouteListExhausted^Managed Object=10.277.4.21^Managed Object Type=Communications Manager^MODE=2;Alarm ID=658775520^Component=eadebhgscvp002.heavensgate.org\x00000`	h32467	Number of RouteListExhausted events exceeds configured threshold during configured interval 0 within 60 minutes\x00012 on cluster DE-HE.\x00012\x00012There are 1 RouteListExha

Where are regular expressions useful?

.conf2016

splunk>

What do Regular Expressions do to you?

In Splunk

- The **rex** and **regex** search commands
- In *props.conf, transforms.conf* and other *.conf* files
- Field extractions
- Data feeds
- Splunk regular expressions are PCRE (Perl Compatible Regular Expressions) and use the PCRE C library.

Elsewhere

Though mostly similar, there are differences in the various implementations, but many of the concepts carry across from one form to another:

- Shell commands
- Utilities like *sed, lex, grep, egrep*
- Programming languages (*perl, python, php, C*, etc.) through libraries or built-in functionality
- **Sed** mode available in Splunk commands like **rex** and in *transforms.conf* files

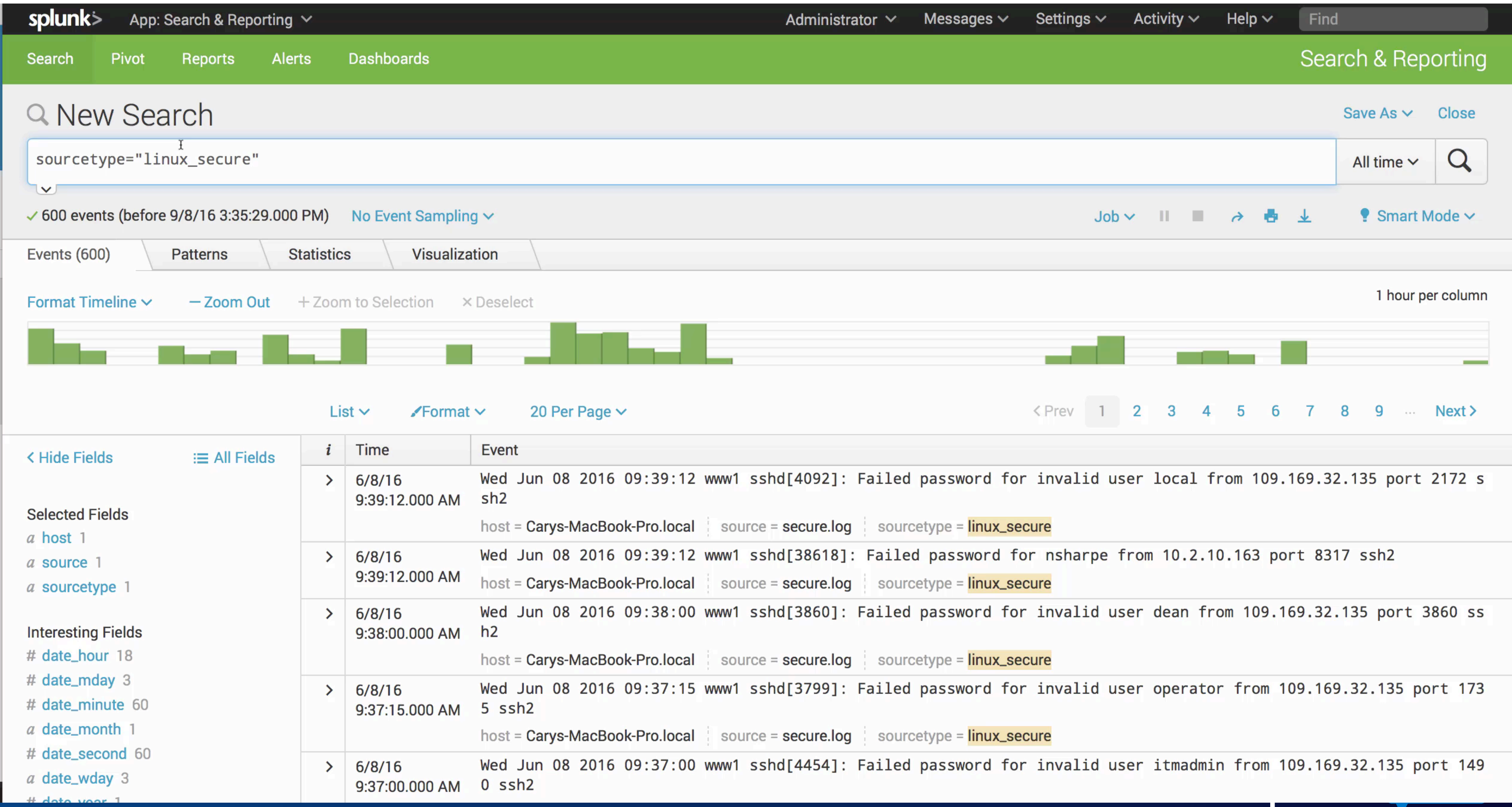
The Splunk Field Extraction Tool

.conf2016

splunk>

Field Extraction Tool

- GUI tool in the web UI of Splunk
- Simple to use
- Pretty good *for a start*, but not *always* good for a final result
- Not able to optimize or do anything complex
- Limited in order or number of named capture groups
- Makes mistakes if you don’t have regular data
- **Demo**



```
sourcetype="linux_secure"
```

✓ 600 events (before 9/8/16 3:35:29.000 PM)

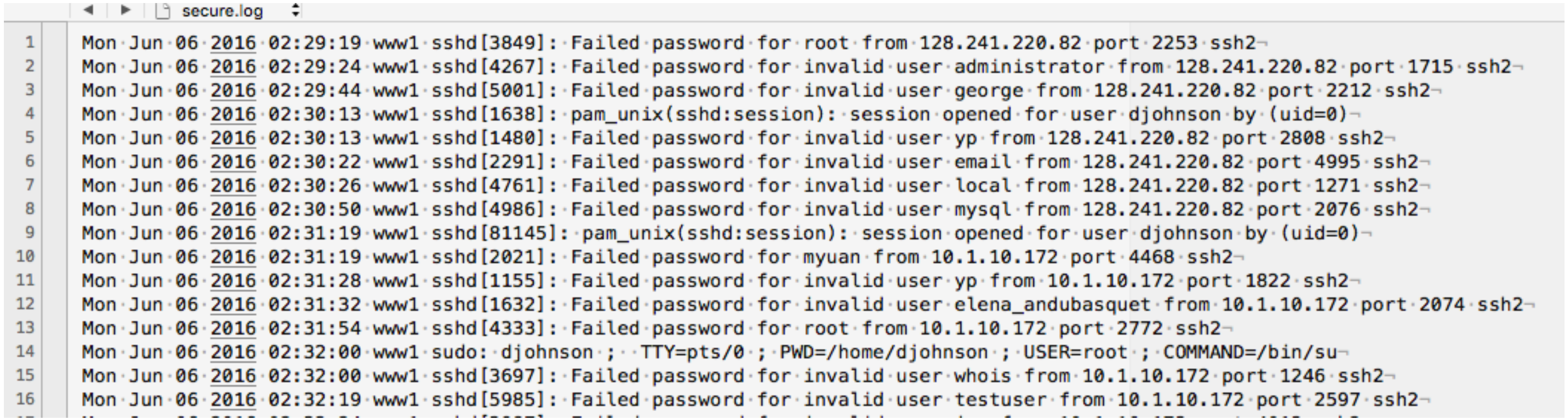
Selected Fields
a host 1
a source 1
a sourcetype 1

Interesting Fields
# date_hour 18
# date_mday 3
# date_minute 60
a date_month 1
# date_second 60
a date_wday 3
# date_year 1

i	Time	Event
>	6/8/16 9:39:12.000 AM	Wed Jun 08 2016 09:39:12 www1 sshd[4092]: Failed password for invalid user local from 109.169.32.135 port 2172 ssh2 host = Carys-MacBook-Pro.local source = secure.log sourcetype = linux_secure
>	6/8/16 9:39:12.000 AM	Wed Jun 08 2016 09:39:12 www1 sshd[38618]: Failed password for nsharpe from 10.2.10.163 port 8317 ssh2 host = Carys-MacBook-Pro.local source = secure.log sourcetype = linux_secure
>	6/8/16 9:38:00.000 AM	Wed Jun 08 2016 09:38:00 www1 sshd[3860]: Failed password for invalid user dean from 109.169.32.135 port 3860 ssh2 host = Carys-MacBook-Pro.local source = secure.log sourcetype = linux_secure
>	6/8/16 9:37:15.000 AM	Wed Jun 08 2016 09:37:15 www1 sshd[3799]: Failed password for invalid user operator from 109.169.32.135 port 1735 ssh2 host = Carys-MacBook-Pro.local source = secure.log sourcetype = linux_secure
>	6/8/16 9:37:00.000 AM	Wed Jun 08 2016 09:37:00 www1 sshd[4454]: Failed password for invalid user itmadmin from 109.169.32.135 port 1490 ssh2

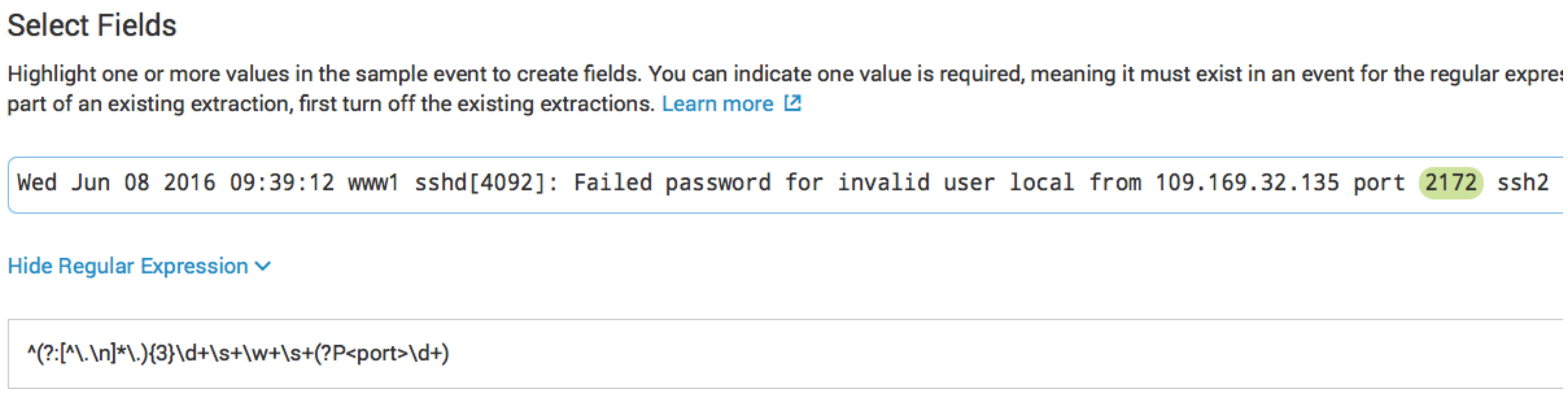
Examples of Field Extractor Deficiencies

Multi-format Security Data in…



```
Mon Jun 06 2016 02:29:19 www1 sshd[3849]: Failed password for root from 128.241.220.82 port 2253 ssh2
Mon Jun 06 2016 02:29:24 www1 sshd[4267]: Failed password for invalid user administrator from 128.241.220.82 port 1715 ssh2
Mon Jun 06 2016 02:29:44 www1 sshd[5001]: Failed password for invalid user george from 128.241.220.82 port 2212 ssh2
Mon Jun 06 2016 02:30:13 www1 sshd[1638]: pam_unix(sshd:session): session opened for user djohnson by (uid=0)
Mon Jun 06 2016 02:30:13 www1 sshd[1480]: Failed password for invalid user yp from 128.241.220.82 port 2808 ssh2
Mon Jun 06 2016 02:30:22 www1 sshd[2291]: Failed password for invalid user email from 128.241.220.82 port 4995 ssh2
Mon Jun 06 2016 02:30:26 www1 sshd[4761]: Failed password for invalid user local from 128.241.220.82 port 1271 ssh2
Mon Jun 06 2016 02:30:50 www1 sshd[4986]: Failed password for invalid user mysql from 128.241.220.82 port 2076 ssh2
Mon Jun 06 2016 02:31:19 www1 sshd[81145]: pam_unix(sshd:session): session opened for user djohnson by (uid=0)
Mon Jun 06 2016 02:31:19 www1 sshd[2021]: Failed password for myuan from 10.1.10.172 port 4468 ssh2
Mon Jun 06 2016 02:31:28 www1 sshd[1155]: Failed password for invalid user yp from 10.1.10.172 port 1822 ssh2
Mon Jun 06 2016 02:31:32 www1 sshd[1632]: Failed password for invalid user elena_andubasquet from 10.1.10.172 port 2074 ssh2
Mon Jun 06 2016 02:31:54 www1 sshd[4333]: Failed password for root from 10.1.10.172 port 2772 ssh2
Mon Jun 06 2016 02:32:00 www1 sudo: djohnson ;  TTY=pts/0 ; PWD=/home/djohnson ; USER=root ; COMMAND=/bin/su
Mon Jun 06 2016 02:32:00 www1 sshd[3697]: Failed password for invalid user whois from 10.1.10.172 port 1246 ssh2
Mon Jun 06 2016 02:32:19 www1 sshd[5985]: Failed password for invalid user testuser from 10.1.10.172 port 2597 ssh2
```

Simple FET extraction of Port



Select Fields

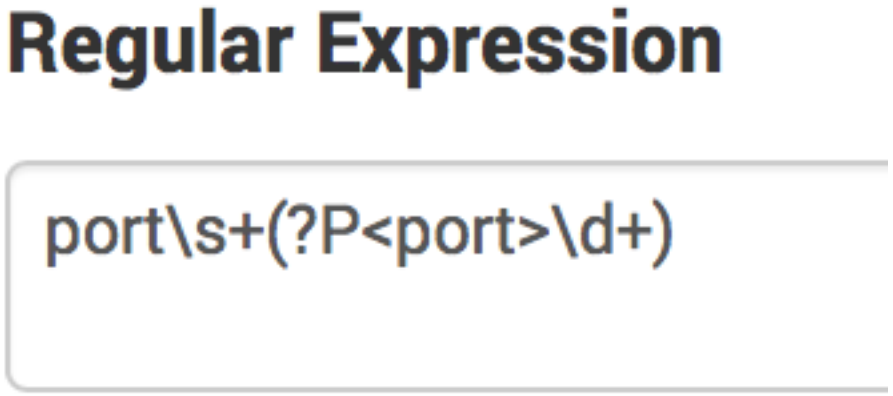
Highlight one or more values in the sample event to create fields. You can indicate one value is required, meaning it must exist in an event for the regular expres
part of an existing extraction, first turn off the existing extractions. Learn more

```
Wed Jun 08 2016 09:39:12 www1 sshd[4092]: Failed password for invalid user local from 109.169.32.135 port 2172 ssh2
```

Hide Regular Expression

```
^(?:[^\.\n]*\.){3}\d+\s+\w+\s+(?P<port>\d+)
```

Intelligent extraction of Port



Regular Expression

```
port\s+(?P<port>\d+)
```

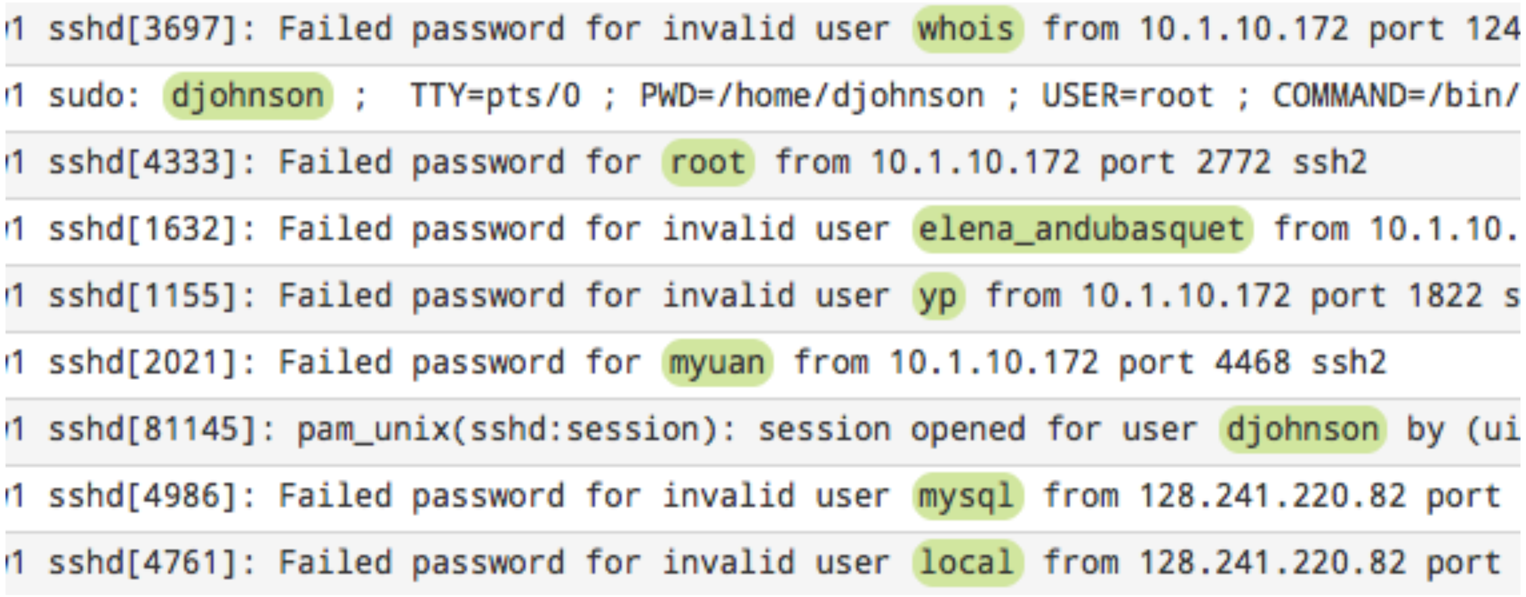
Named Capture Groups

- (?P<name>…) === (?<name>...)
- The **P** is optional (came from **P**ython), but it is usually considered more correct
- Splunk FET will use (?P<name>...), so why not make things similar?

BUT

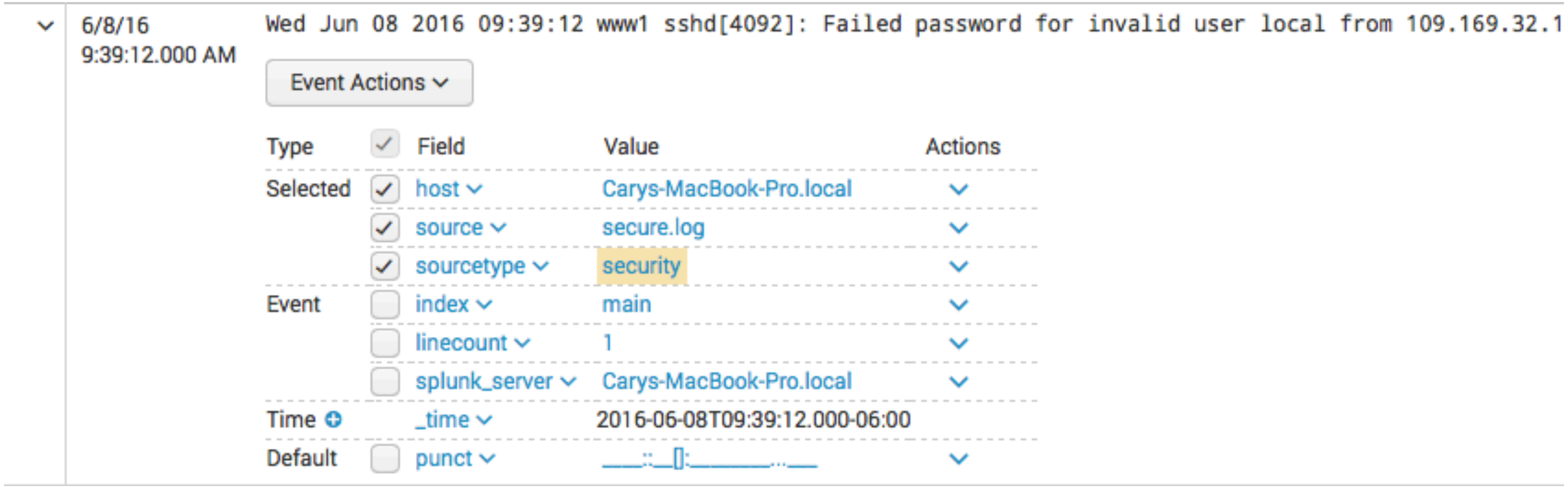
- Do it the way you feel most comfortable

Goal: ***user*** from all entries using one regex



```
1 sshd[3697]: Failed password for invalid user whois from 10.1.10.172 port 124
1 sudo: djohnson ;  TTY=pts/0 ; PWD=/home/djohnson ; USER=root ; COMMAND=/bin/
1 sshd[4333]: Failed password for root from 10.1.10.172 port 2772 ssh2
1 sshd[1632]: Failed password for invalid user elena_andubasquet from 10.1.10.
1 sshd[1155]: Failed password for invalid user yp from 10.1.10.172 port 1822 s
1 sshd[2021]: Failed password for myuan from 10.1.10.172 port 4468 ssh2
1 sshd[81145]: pam_unix(sshd:session): session opened for user djohnson by (ui
1 sshd[4986]: Failed password for invalid user mysql from 128.241.220.82 port
1 sshd[4761]: Failed password for invalid user local from 128.241.220.82 port
```

There is no user automatically extracted



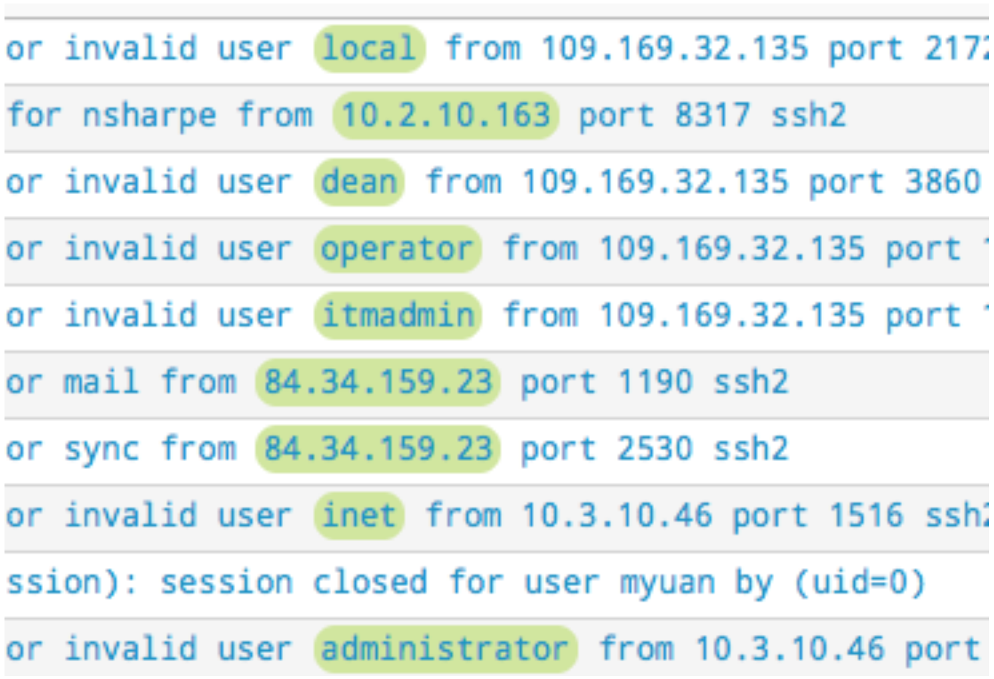
6/8/16
9:39:12.000 AM

```
Wed Jun 08 2016 09:39:12 www1 sshd[4092]: Failed password for invalid user local from 109.169.32.1
```

Event Actions

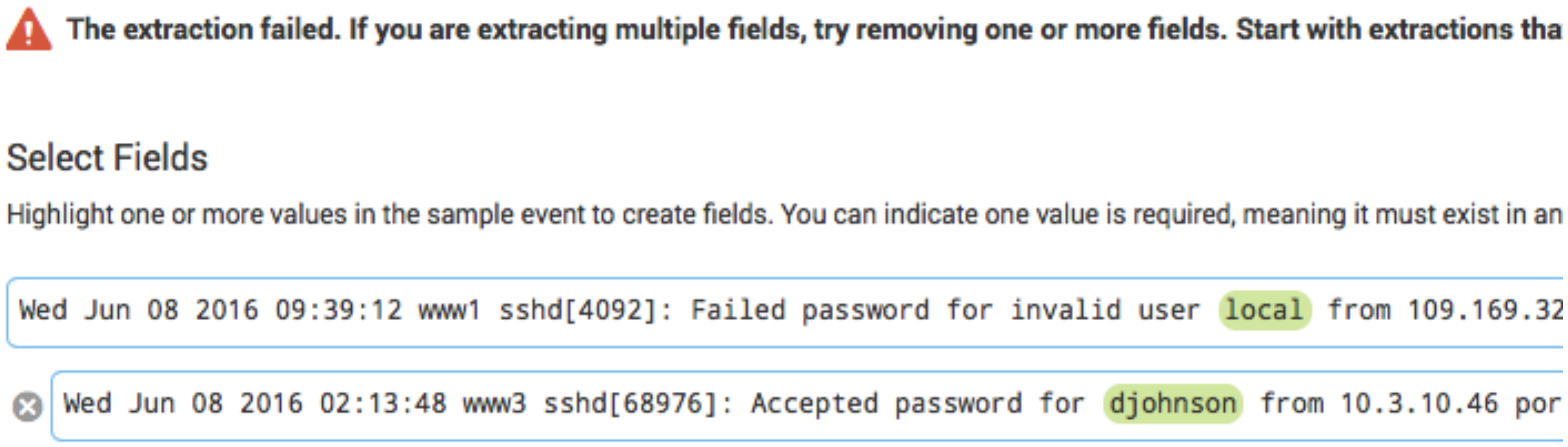
Type	✓	Field	Value	Actions
Selected	✓	host	Carys-MacBook-Pro.local	
	✓	source	secure.log	
	✓	sourcetype	security	
Event		index	main	
		linecount	1	
		splunk_server	Carys-MacBook-Pro.local	
Time		_time	2016-06-08T09:39:12.000-06:00	
Default		punct	____::__[]:_________...___	

Field Extractor Failed



```
or invalid user local from 109.169.32.135 port 217
for nsharpe from 10.2.10.163 port 8317 ssh2
or invalid user dean from 109.169.32.135 port 3860
or invalid user operator from 109.169.32.135 port
or invalid user itmadmin from 109.169.32.135 port
or mail from 84.34.159.23 port 1190 ssh2
or sync from 84.34.159.23 port 2530 ssh2
or invalid user inet from 10.3.10.46 port 1516 ssh
ssion): session closed for user myuan by (uid=0)
or invalid user administrator from 10.3.10.46 port
```

Field Extractor Failed After More Lines Added



The extraction failed. If you are extracting multiple fields, try removing one or more fields. Start with extractions tha

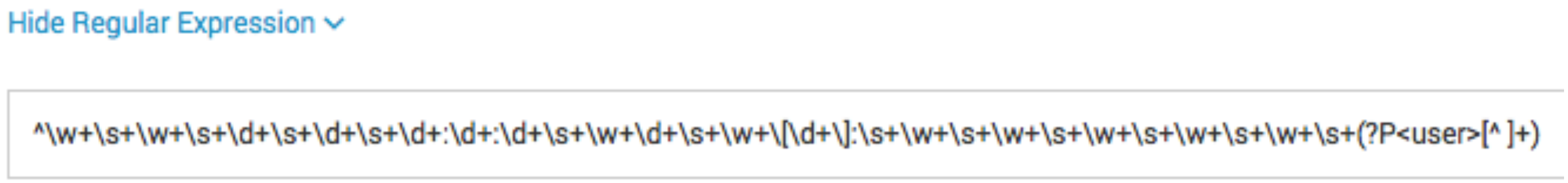
Select Fields

Highlight one or more values in the sample event to create fields. You can indicate one value is required, meaning it must exist in an

```
Wed Jun 08 2016 09:39:12 www1 sshd[4092]: Failed password for invalid user local from 109.169.32
```

```
Wed Jun 08 2016 02:13:48 www3 sshd[68976]: Accepted password for djohnson from 10.3.10.46 por
```

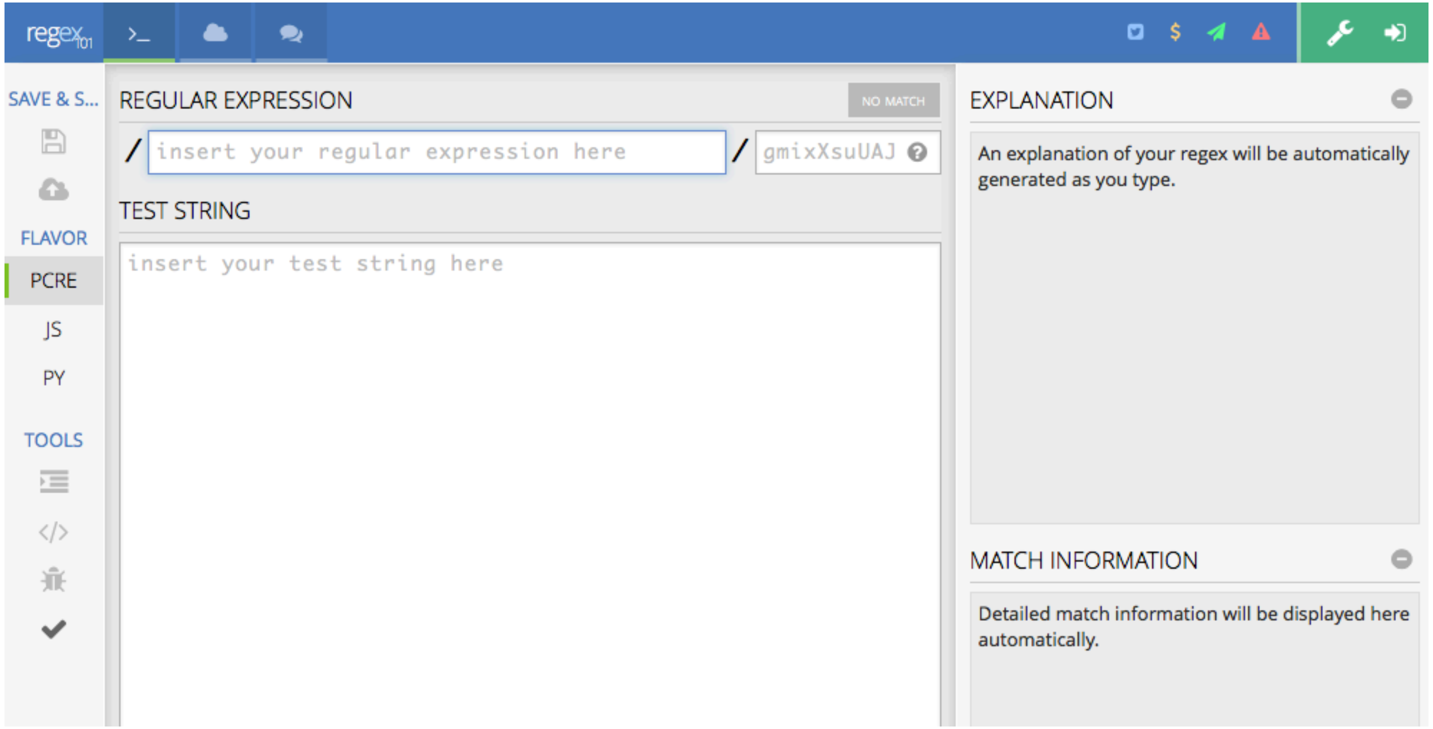
Let’s look at the generated REGEX



Hide Regular Expression

```
^\w+\s+\w+\s+\d+\s+\d+\s+\d+:\d+:\d+\s+\w+\d+\s+\w+\[\d+\]:\s+\w+\s+\w+\s+\w+\s+\w+\s+\w+\s+(?P<user>[^ ]+)
```

My tool of choice: regex101.com



REGULAR EXPRESSION

/ insert your regular expression here / gmixXsuUAJ

TEST STRING

insert your test string here

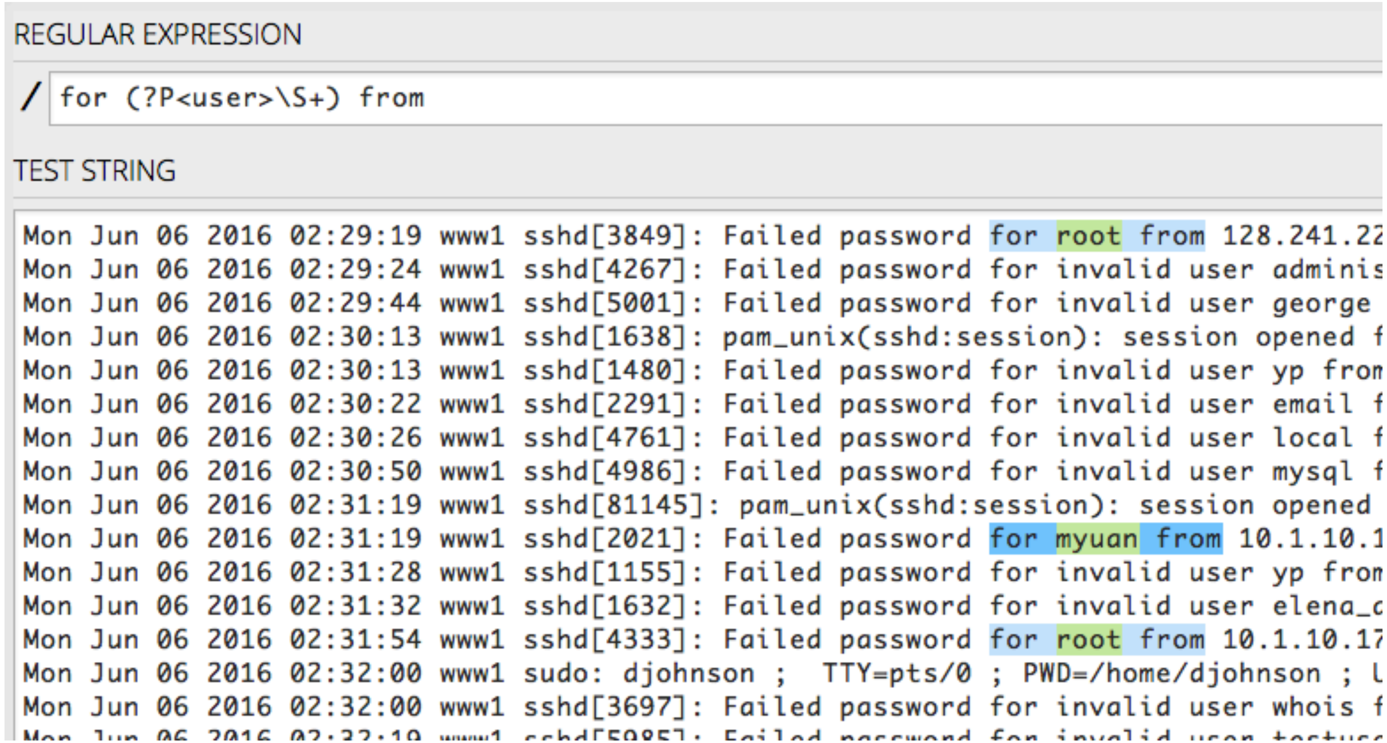
EXPLANATION

An explanation of your regex will be automatically generated as you type.

MATCH INFORMATION

Detailed match information will be displayed here automatically.

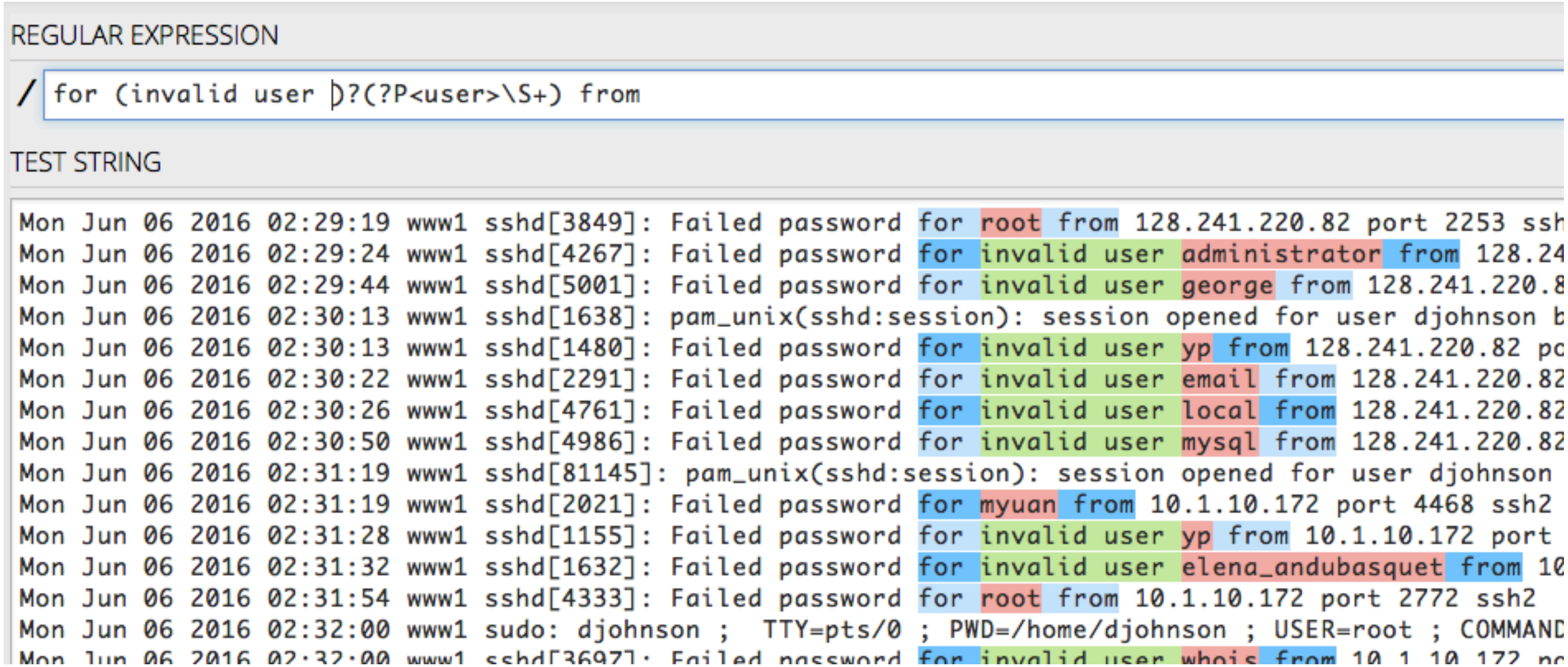
Add the data and a regex:



```
/ for (?P<user>\S+) from
```

```
Mon Jun 06 2016 02:29:19 www1 sshd[3849]: Failed password for root from 128.241.22
Mon Jun 06 2016 02:29:24 www1 sshd[4267]: Failed password for invalid user adminis
Mon Jun 06 2016 02:29:44 www1 sshd[5001]: Failed password for invalid user george
Mon Jun 06 2016 02:30:13 www1 sshd[1638]: pam_unix(sshd:session): session opened f
Mon Jun 06 2016 02:30:13 www1 sshd[1480]: Failed password for invalid user yp from
Mon Jun 06 2016 02:30:22 www1 sshd[2291]: Failed password for invalid user email f
Mon Jun 06 2016 02:30:26 www1 sshd[4761]: Failed password for invalid user local f
Mon Jun 06 2016 02:30:50 www1 sshd[4986]: Failed password for invalid user mysql f
Mon Jun 06 2016 02:31:19 www1 sshd[81145]: pam_unix(sshd:session): session opened
Mon Jun 06 2016 02:31:19 www1 sshd[2021]: Failed password for myuan from 10.1.10.1
Mon Jun 06 2016 02:31:28 www1 sshd[1155]: Failed password for invalid user yp from
Mon Jun 06 2016 02:31:32 www1 sshd[1632]: Failed password for invalid user elena_c
Mon Jun 06 2016 02:31:54 www1 sshd[4333]: Failed password for root from 10.1.10.17
Mon Jun 06 2016 02:32:00 www1 sudo: djohnson ;  TTY=pts/0 ; PWD=/home/djohnson ; U
Mon Jun 06 2016 02:32:00 www1 sshd[3697]: Failed password for invalid user whois f
```

Refine the regex – better matches, but not all



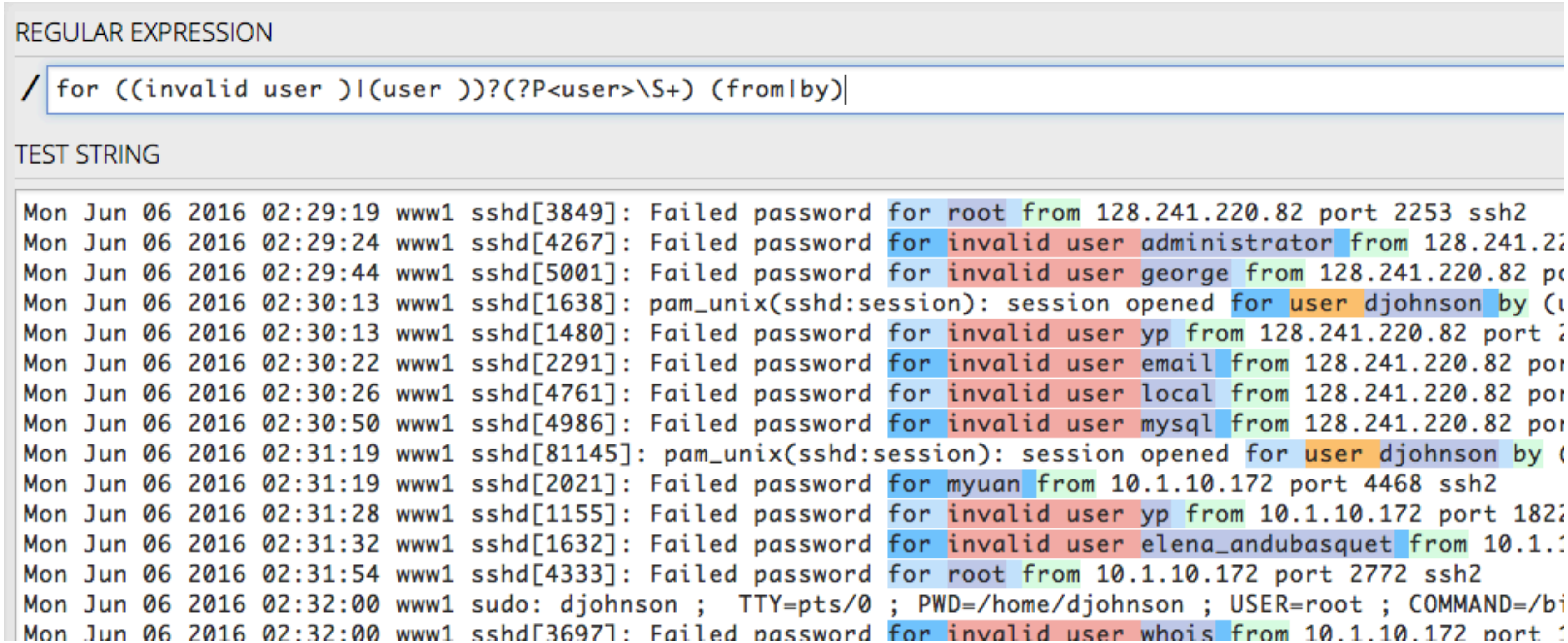
REGULAR EXPRESSION

```
/ for (invalid user )?(?P<user>\S+) from
```

TEST STRING

```
Mon Jun 06 2016 02:29:19 www1 sshd[3849]: Failed password for root from 128.241.220.82 port 2253 ssh
Mon Jun 06 2016 02:29:24 www1 sshd[4267]: Failed password for invalid user administrator from 128.24
Mon Jun 06 2016 02:29:44 www1 sshd[5001]: Failed password for invalid user george from 128.241.220.8
Mon Jun 06 2016 02:30:13 www1 sshd[1638]: pam_unix(sshd:session): session opened for user djohnson b
Mon Jun 06 2016 02:30:13 www1 sshd[1480]: Failed password for invalid user yp from 128.241.220.82 pc
Mon Jun 06 2016 02:30:22 www1 sshd[2291]: Failed password for invalid user email from 128.241.220.82
Mon Jun 06 2016 02:30:26 www1 sshd[4761]: Failed password for invalid user local from 128.241.220.82
Mon Jun 06 2016 02:30:50 www1 sshd[4986]: Failed password for invalid user mysql from 128.241.220.82
Mon Jun 06 2016 02:31:19 www1 sshd[81145]: pam_unix(sshd:session): session opened for user djohnson
Mon Jun 06 2016 02:31:19 www1 sshd[2021]: Failed password for myuan from 10.1.10.172 port 4468 ssh2
Mon Jun 06 2016 02:31:28 www1 sshd[1155]: Failed password for invalid user yp from 10.1.10.172 port
Mon Jun 06 2016 02:31:32 www1 sshd[1632]: Failed password for invalid user elena_andubasquet from 10
Mon Jun 06 2016 02:31:54 www1 sshd[4333]: Failed password for root from 10.1.10.172 port 2772 ssh2
Mon Jun 06 2016 02:32:00 www1 sudo: djohnson ;  TTY=pts/0 ; PWD=/home/djohnson ; USER=root ; COMMAND
Mon Jun 06 2016 02:32:00 www1 sshd[3697]: Failed password for invalid user whois from 10.1.10.172 pc
```

Refine the regex again – almost there



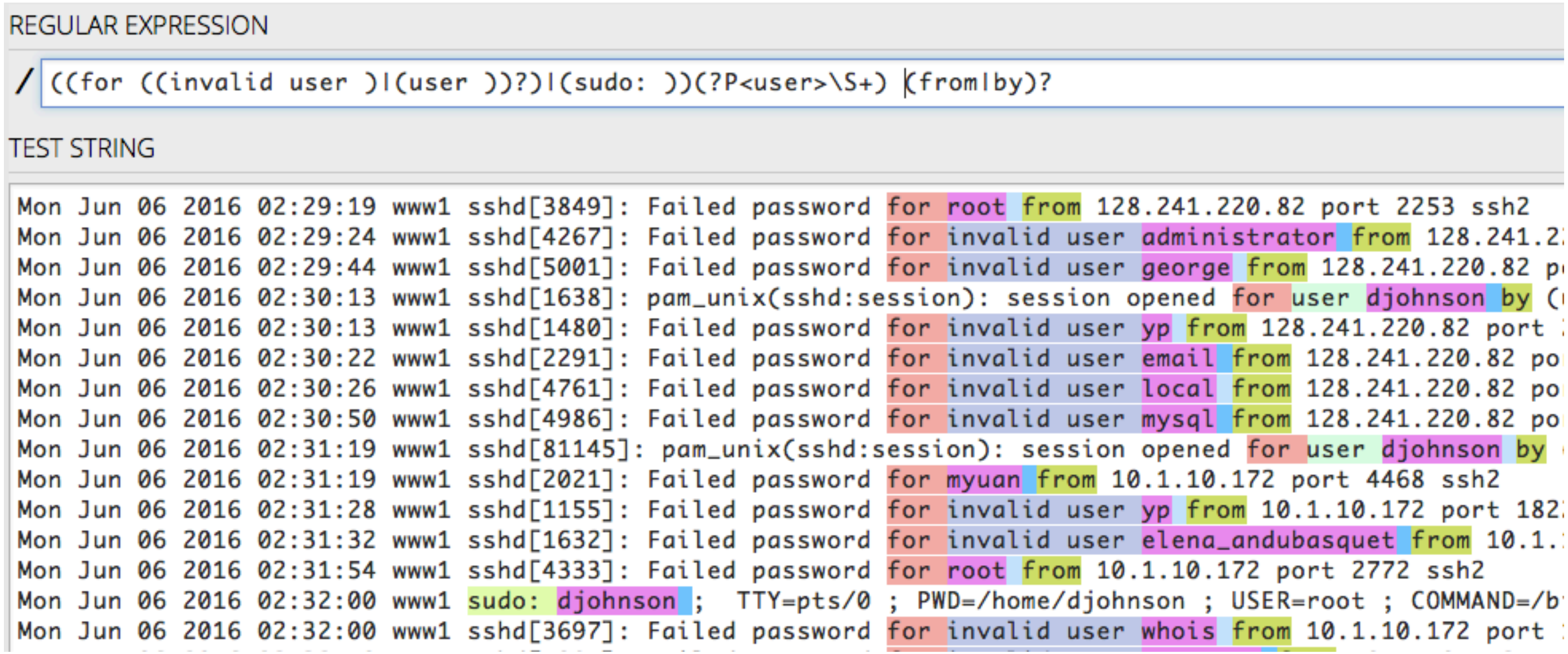
REGULAR EXPRESSION

```
/ for ((invalid user )|(user ))?(?P<user>\S+) (from|by)
```

TEST STRING

```
Mon Jun 06 2016 02:29:19 www1 sshd[3849]: Failed password for root from 128.241.220.82 port 2253 ssh2
Mon Jun 06 2016 02:29:24 www1 sshd[4267]: Failed password for invalid user administrator from 128.241.22
Mon Jun 06 2016 02:29:44 www1 sshd[5001]: Failed password for invalid user george from 128.241.220.82 po
Mon Jun 06 2016 02:30:13 www1 sshd[1638]: pam_unix(sshd:session): session opened for user djohnson by (u
Mon Jun 06 2016 02:30:13 www1 sshd[1480]: Failed password for invalid user yp from 128.241.220.82 port 2
Mon Jun 06 2016 02:30:22 www1 sshd[2291]: Failed password for invalid user email from 128.241.220.82 por
Mon Jun 06 2016 02:30:26 www1 sshd[4761]: Failed password for invalid user local from 128.241.220.82 por
Mon Jun 06 2016 02:30:50 www1 sshd[4986]: Failed password for invalid user mysql from 128.241.220.82 por
Mon Jun 06 2016 02:31:19 www1 sshd[81145]: pam_unix(sshd:session): session opened for user djohnson by (
Mon Jun 06 2016 02:31:19 www1 sshd[2021]: Failed password for myuan from 10.1.10.172 port 4468 ssh2
Mon Jun 06 2016 02:31:28 www1 sshd[1155]: Failed password for invalid user yp from 10.1.10.172 port 182
Mon Jun 06 2016 02:31:32 www1 sshd[1632]: Failed password for invalid user elena_andubasquet from 10.1.1
Mon Jun 06 2016 02:31:54 www1 sshd[4333]: Failed password for root from 10.1.10.172 port 2772 ssh2
Mon Jun 06 2016 02:32:00 www1 sudo: djohnson ;  TTY=pts/0 ; PWD=/home/djohnson ; USER=root ; COMMAND=/bi
Mon Jun 06 2016 02:32:00 www1 sshd[3697]: Failed password for invalid user whois from 10.1.10.172 port 1
```

And FINALLY – we got them all



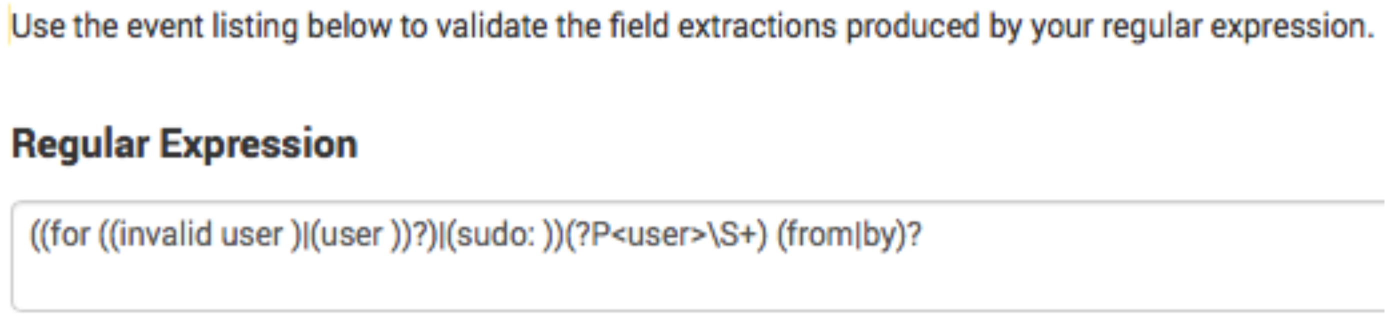
REGULAR EXPRESSION

```
/ ((for ((invalid user )|(user ))?)|(sudo: ))(?P<user>\S+) (from|by)?
```

TEST STRING

```
Mon Jun 06 2016 02:29:19 www1 sshd[3849]: Failed password for root from 128.241.220.82 port 2253 ssh2
Mon Jun 06 2016 02:29:24 www1 sshd[4267]: Failed password for invalid user administrator from 128.241.2
Mon Jun 06 2016 02:29:44 www1 sshd[5001]: Failed password for invalid user george from 128.241.220.82 p
Mon Jun 06 2016 02:30:13 www1 sshd[1638]: pam_unix(sshd:session): session opened for user djohnson by (
Mon Jun 06 2016 02:30:13 www1 sshd[1480]: Failed password for invalid user yp from 128.241.220.82 port
Mon Jun 06 2016 02:30:22 www1 sshd[2291]: Failed password for invalid user email from 128.241.220.82 po
Mon Jun 06 2016 02:30:26 www1 sshd[4761]: Failed password for invalid user local from 128.241.220.82 po
Mon Jun 06 2016 02:30:50 www1 sshd[4986]: Failed password for invalid user mysql from 128.241.220.82 po
Mon Jun 06 2016 02:31:19 www1 sshd[81145]: pam_unix(sshd:session): session opened for user djohnson by
Mon Jun 06 2016 02:31:19 www1 sshd[2021]: Failed password for myuan from 10.1.10.172 port 4468 ssh2
Mon Jun 06 2016 02:31:28 www1 sshd[1155]: Failed password for invalid user yp from 10.1.10.172 port 182
Mon Jun 06 2016 02:31:32 www1 sshd[1632]: Failed password for invalid user elena_andubasquet from 10.1.
Mon Jun 06 2016 02:31:54 www1 sshd[4333]: Failed password for root from 10.1.10.172 port 2772 ssh2
Mon Jun 06 2016 02:32:00 www1 sudo: djohnson ;  TTY=pts/0 ; PWD=/home/djohnson ; USER=root ; COMMAND=/b
Mon Jun 06 2016 02:32:00 www1 sshd[3697]: Failed password for invalid user whois from 10.1.10.172 port
```

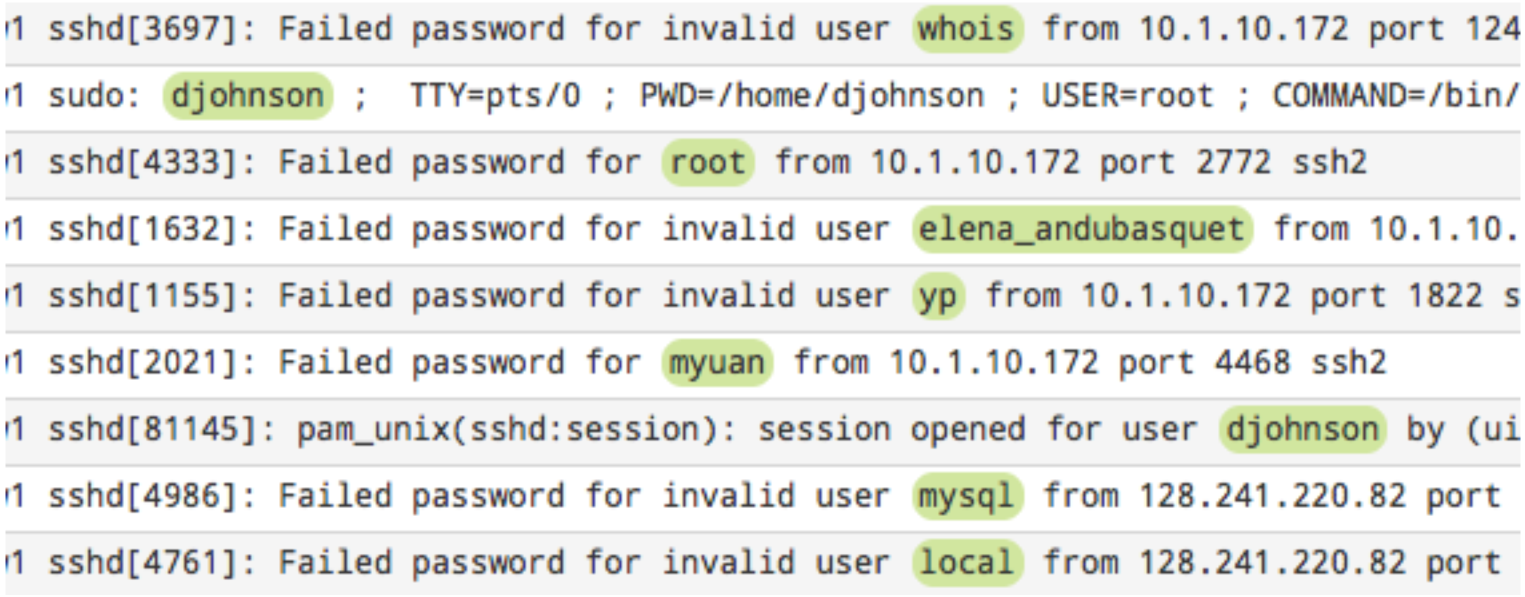
Back to the Field Extractor – use our regex


Use the event listing below to validate the field extractions produced by your regular expression.

Regular Expression

```
((for ((invalid user )|(user ))?)|(sudo: ))(?P<user>\S+) (from|by)?
```

And the results are MUCH better



```
1 sshd[3697]: Failed password for invalid user whois from 10.1.10.172 port 124
1 sudo: djohnson ;  TTY=pts/0 ; PWD=/home/djohnson ; USER=root ; COMMAND=/bin/
1 sshd[4333]: Failed password for root from 10.1.10.172 port 2772 ssh2
1 sshd[1632]: Failed password for invalid user elena_andubasquet from 10.1.10.
1 sshd[1155]: Failed password for invalid user yp from 10.1.10.172 port 1822 s
1 sshd[2021]: Failed password for myuan from 10.1.10.172 port 4468 ssh2
1 sshd[81145]: pam_unix(sshd:session): session opened for user djohnson by (ui
1 sshd[4986]: Failed password for invalid user mysql from 128.241.220.82 port
1 sshd[4761]: Failed password for invalid user local from 128.241.220.82 port
```

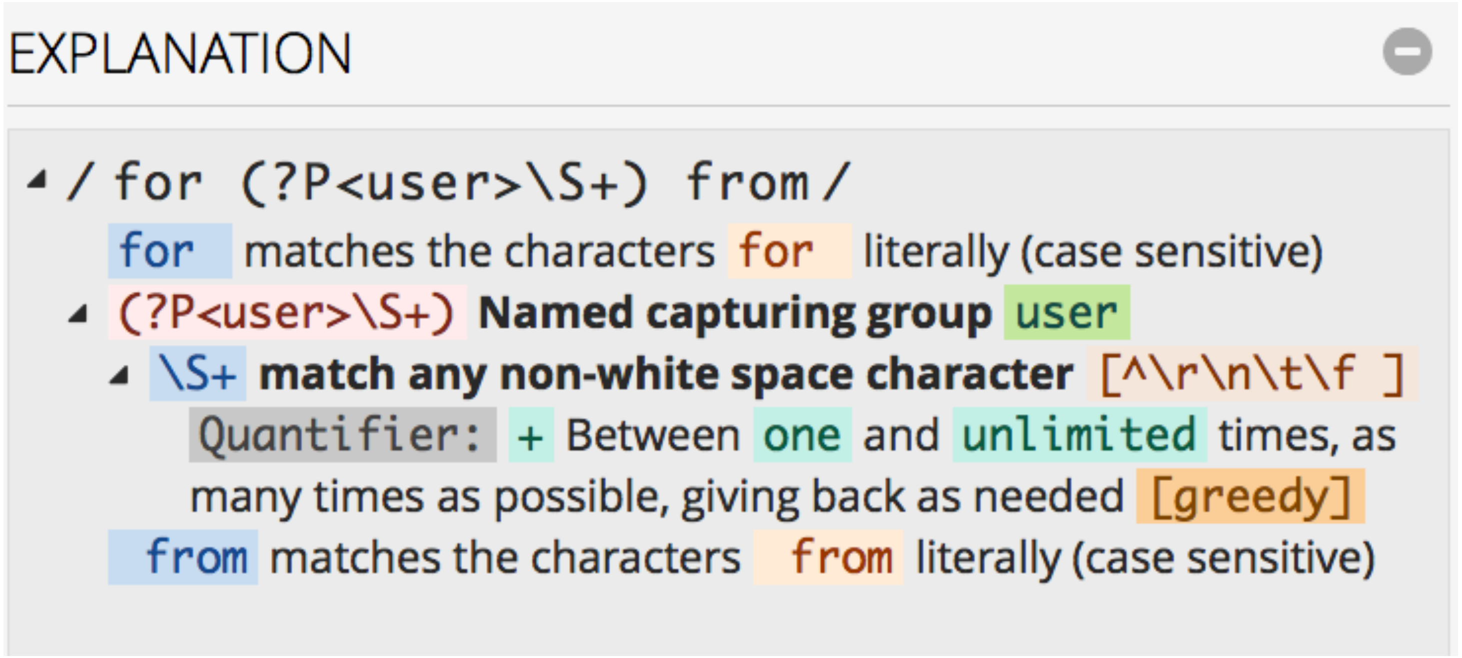
Dissecting the Path to Success

Note the progression in the regex

- for (?P<user>\S+) from
- for (invalid user)?(?P<user>\S+) from
- for ((invalid user)|(user))?(?P<user>\S+) (from|by)
- ((for ((invalid user)|(user))?)|(sudo:))(?P<user>\S+) (from|by)?

Explanation

- for (?P<user>\S+) from

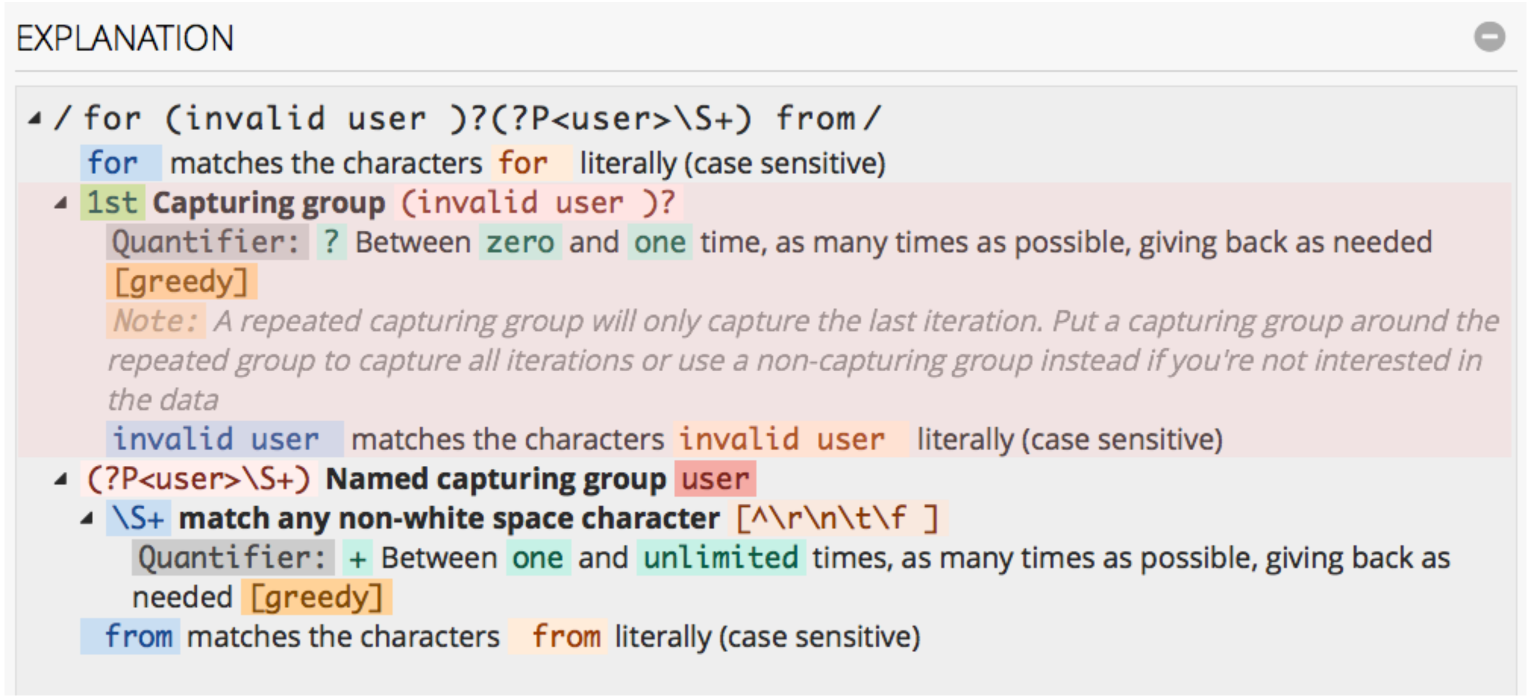


EXPLANATION

```
/ for (?P<user>\S+) from /
    for  matches the characters  for  literally (case sensitive)
    (?P<user>\S+) Named capturing group user
        \S+ match any non-white space character [^\r\n\t\f ]
            Quantifier: + Between one and unlimited times, as
            many times as possible, giving back as needed [greedy]
    from matches the characters  from literally (case sensitive)
```

Explanation (cont)

- for (invalid user)?(?P<user>\S+) from



EXPLANATION

/ for (invalid user)?(?P<user>\S+) from/

for matches the characters for literally (case sensitive)

1st Capturing group (invalid user)?

Quantifier: ? Between zero and one time, as many times as possible, giving back as needed [greedy]

Note: *A repeated capturing group will only capture the last iteration. Put a capturing group around the repeated group to capture all iterations or use a non-capturing group instead if you're not interested in the data*

invalid user matches the characters invalid user literally (case sensitive)

(?P<user>\S+) **Named capturing group** user

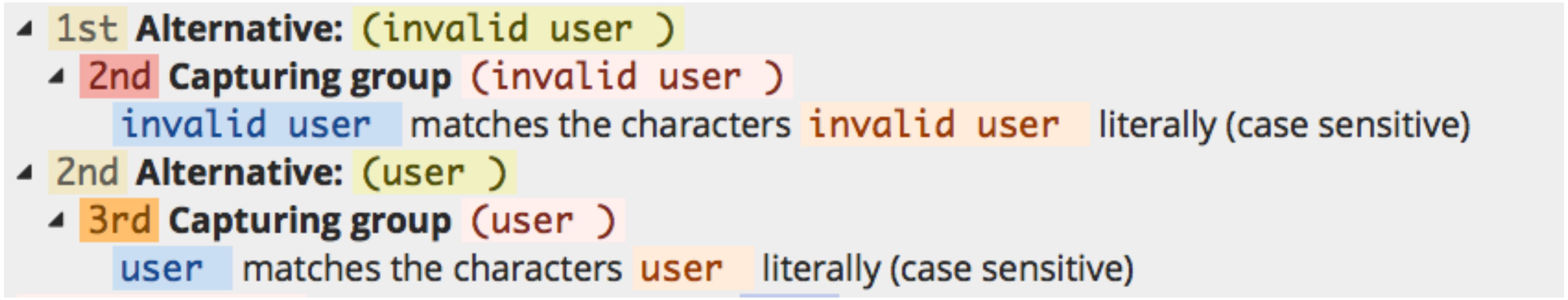
\S+ **match any non-white space character** [^\r\n\t\f]

Quantifier: + Between one and unlimited times, as many times as possible, giving back as needed [greedy]

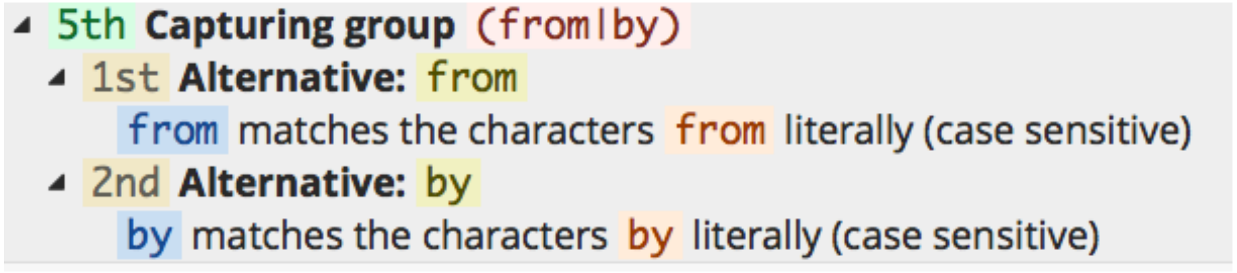
from matches the characters from literally (case sensitive)

Explanation (cont)

- for ((invalid user)|(user))?(?P<user>\S+) (from|by)



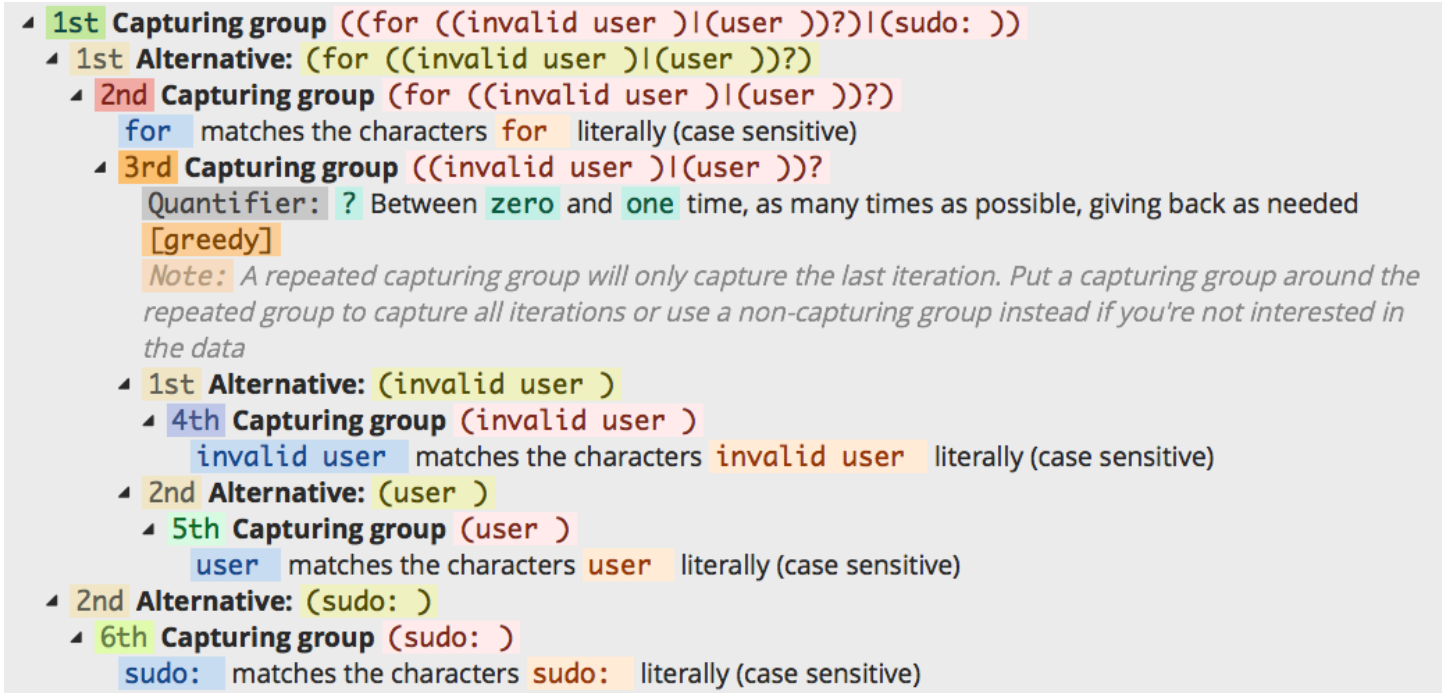
```
◢ 1st Alternative: (invalid user )
  ◢ 2nd Capturing group (invalid user )
      invalid user  matches the characters invalid user  literally (case sensitive)
◢ 2nd Alternative: (user )
  ◢ 3rd Capturing group (user )
      user  matches the characters user  literally (case sensitive)
```



```
◢ 5th Capturing group (from|by)
  ◢ 1st Alternative: from
      from matches the characters from literally (case sensitive)
  ◢ 2nd Alternative: by
      by matches the characters by literally (case sensitive)
```

Explanation (cont)

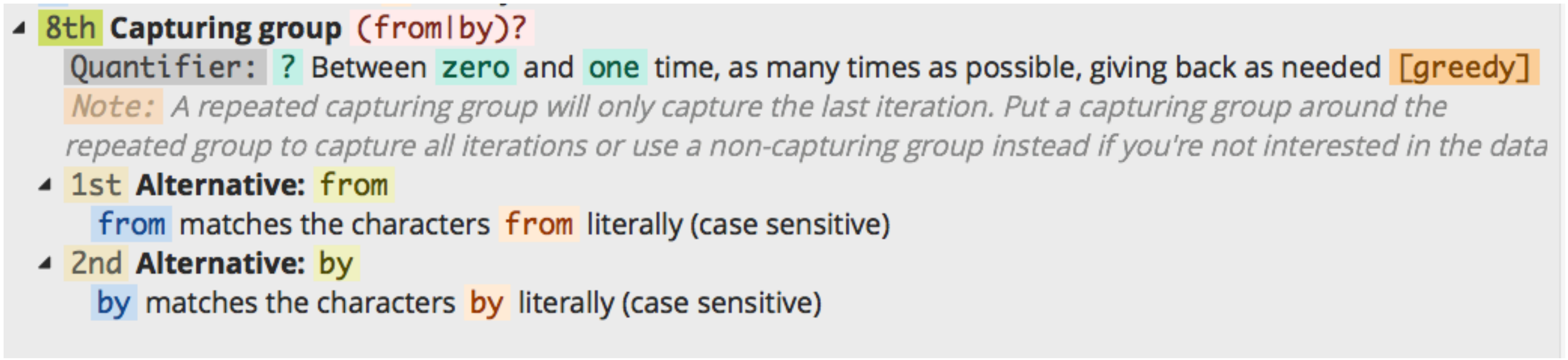
- ((for ((invalid user)|(user))?)|(sudo:))(?P<user>\S+) (from|by)?



```
◢ 1st Capturing group ((for ((invalid user )|(user ))?)|(sudo: ))
  ◢ 1st Alternative: (for ((invalid user )|(user ))?)
    ◢ 2nd Capturing group (for ((invalid user )|(user ))?)
        for matches the characters for literally (case sensitive)
      ◢ 3rd Capturing group ((invalid user )|(user ))?
          Quantifier: ? Between zero and one time, as many times as possible, giving back as needed
          [greedy]
          Note: A repeated capturing group will only capture the last iteration. Put a capturing group around the
          repeated group to capture all iterations or use a non-capturing group instead if you're not interested in
          the data
        ◢ 1st Alternative: (invalid user )
          ◢ 4th Capturing group (invalid user )
              invalid user matches the characters invalid user literally (case sensitive)
        ◢ 2nd Alternative: (user )
          ◢ 5th Capturing group (user )
              user matches the characters user literally (case sensitive)
  ◢ 2nd Alternative: (sudo: )
    ◢ 6th Capturing group (sudo: )
        sudo: matches the characters sudo: literally (case sensitive)
```

Explanation (cont)

- ((for ((invalid user)|(user))?)|(sudo:))(?P<user>\S+) (from|by)?



8th Capturing group (from|by)?
Quantifier: ? Between zero and one time, as many times as possible, giving back as needed [greedy]
Note: A repeated capturing group will only capture the last iteration. Put a capturing group around the repeated group to capture all iterations or use a non-capturing group instead if you're not interested in the data
1st Alternative: from
from matches the characters from literally (case sensitive)
2nd Alternative: by
by matches the characters by literally (case sensitive)

It’s not hard, it just takes understanding

- Start by giving defining a simple part of the regex that will work on one of the line ***types***.
- You might want to define another regex for a different type of line and try to combine the two regex’s after

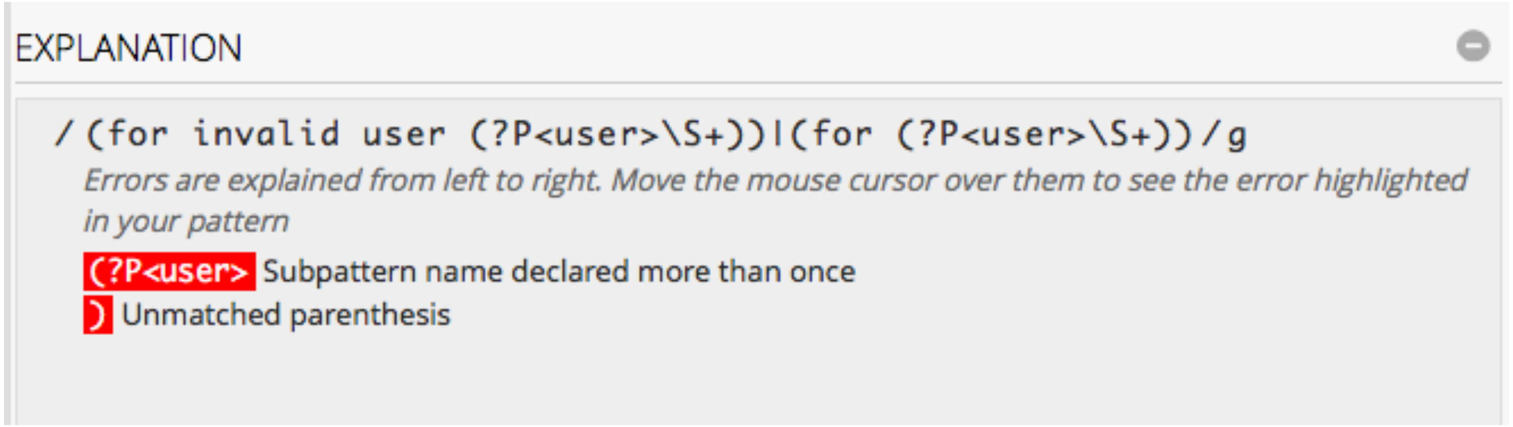
Or

- You can just modify the one that you had to find an additional instance
- Remember – You can only have ONE and only one named capture group *for a given name* **IN THE SAME REGULAR EXPRESSION**, but you can have multiple named capture groups with different names, or different regular expression with the same capture group name

One named capture group with a single name

- This will fail:

- (for invalid user (?P<user>\S+))|(for (?P<user>\S+))



EXPLANATION

```
/(for invalid user (?P<user>\S+))|(for (?P<user>\S+))/g
```

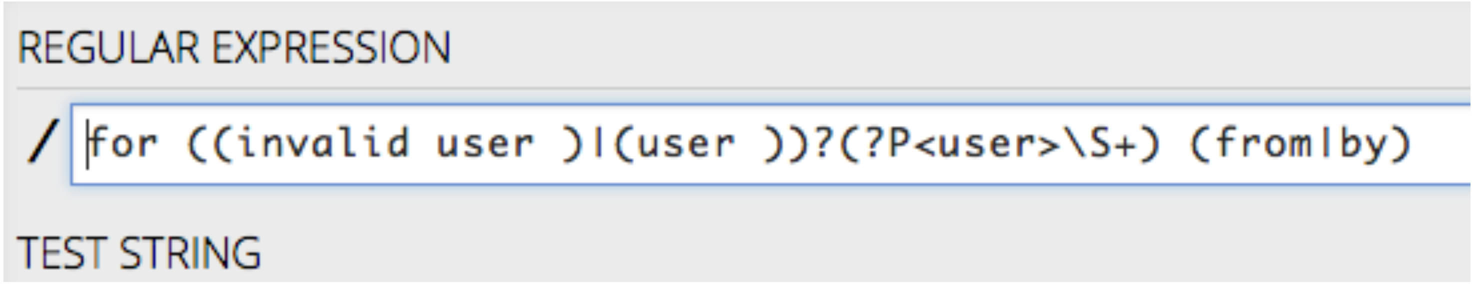
Errors are explained from left to right. Move the mouse cursor over them to see the error highlighted in your pattern

(?P<user> Subpattern name declared more than once

) Unmatched parenthesis

Use parenthesis!

- Use parenthesis to define capture groups to define groups of text that may be optional or when you want to pick from ***One Of***



REGULAR EXPRESSION

```
/ for ((invalid user )|(user ))?(?P<user>\S+) (from|by)
```

TEST STRING

- Not only does it make it more readable later, but this one wouldn't work without them
  - **(from|by) user** is not the same as **from|by user**

Use the best character class

- Sometime you need to make a field be able to match data that you haven’t seen in the data yet, so in this case be general
- The **\S** is best because **\s** will be the delimiter (a space in this case) because you want to catch any potential case that you don’t see in the data – yet.

```
\S+
```

- If you have a delimiter that you can count on, use something like this to match the field value (int this case be specific about what it is NOT):

```
[^,]+
```

Now you can wake up – seriously!

Transforms.conf Gotchas

Problem

You can’t index Social Security Numbers in your data, but phone numbers are okay.

- Use transforms.conf to clean up the data
- Leave the phone numbers alone, but take out the SSN’s

SSN vs Phone

Distinguishing in regex

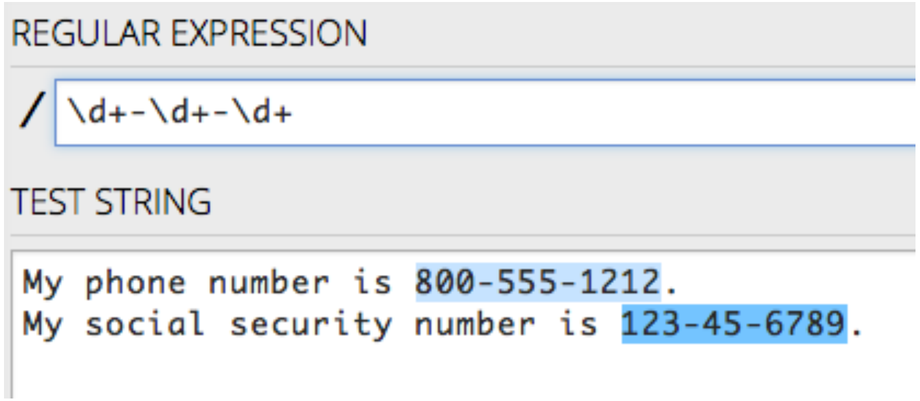
SSN	**Phone #**
123-45-6789	800-123-4567
• \d{3}-\d{2}-\d{4}	• \d{3}-\d{3}-\d{4}

Be as specific in your matches as possible

- You could use:

\d+-\d+-\d+

- But it will mistake a phone number for a SSN:



```
/ \d+-\d+-\d+
```

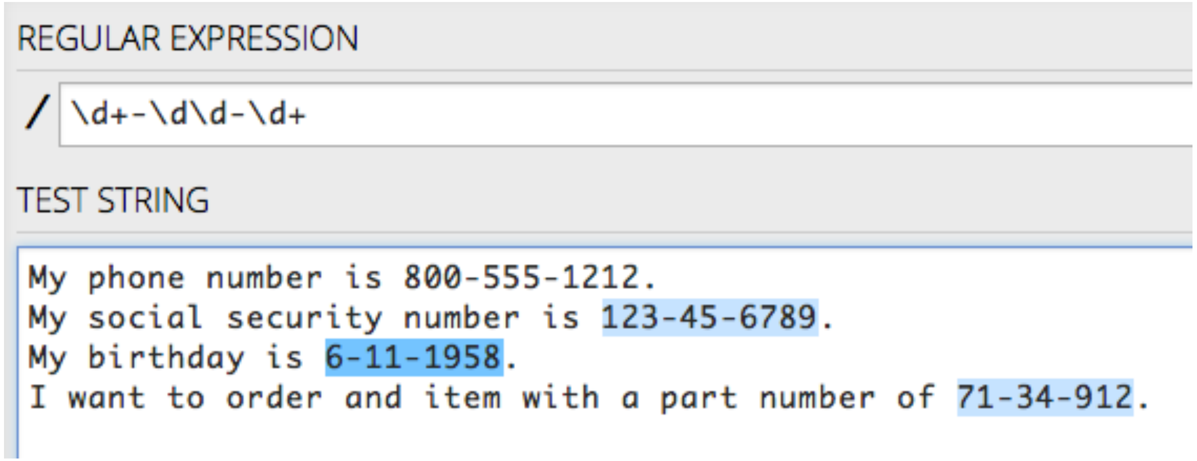
```
My phone number is 800-555-1212.
My social security number is 123-45-6789.
```

This is a better match

- You could use:

  \d+-\d\d-\d+

- Simple change, but it will NOT mistake a phone number for a SSN, but…:



REGULAR EXPRESSION

```
/ \d+-\d\d-\d+
```

TEST STRING

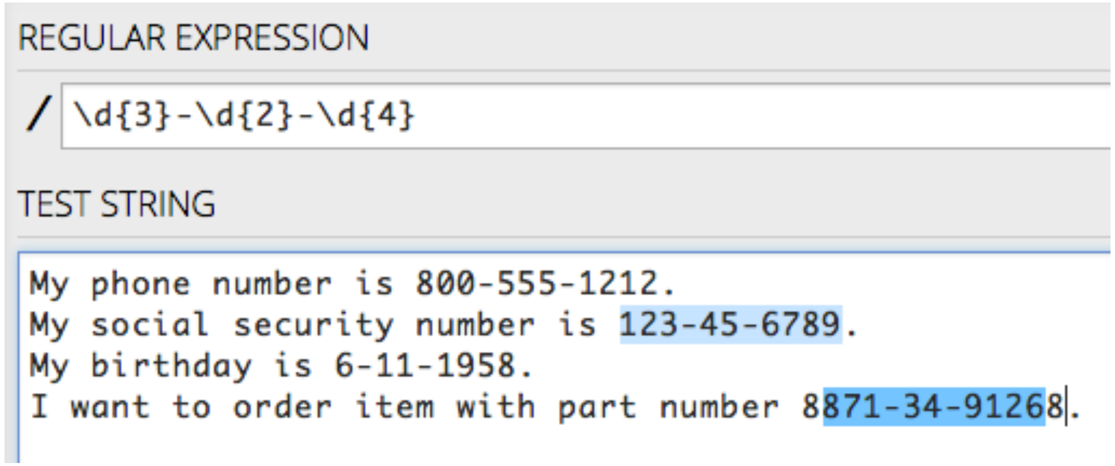
```
My phone number is 800-555-1212.
My social security number is 123-45-6789.
My birthday is 6-11-1958.
I want to order and item with a part number of 71-34-912.
```

We're so close

- This is exactly what every SSN looks like (in regex):

\d{3}-\d{2}-\d{4}

- This matches better, but it needs more work:



REGULAR EXPRESSION

```
/ \d{3}-\d{2}-\d{4}
```

TEST STRING

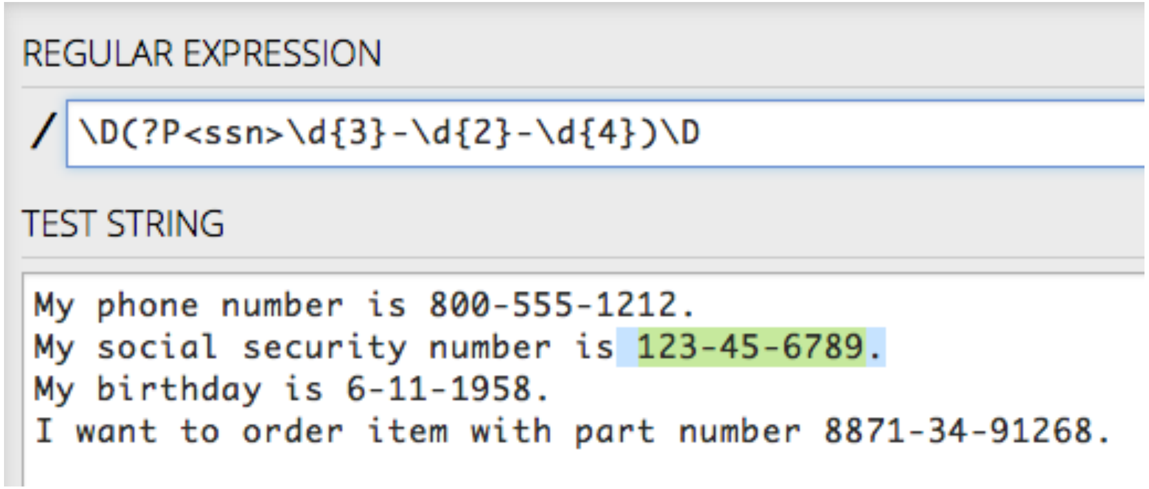
```
My phone number is 800-555-1212.
My social security number is 123-45-6789.
My birthday is 6-11-1958.
I want to order item with part number 8871-34-91268.
```

So let's make it specific

- This is exactly what every SSN looks like (in regex):

\D(?P<ssn>\d{3}-\d{2}-\d{4})\D

- This matches only the SSN:



REGULAR EXPRESSION

```
/ \D(?P<ssn>\d{3}-\d{2}-\d{4})\D
```

TEST STRING

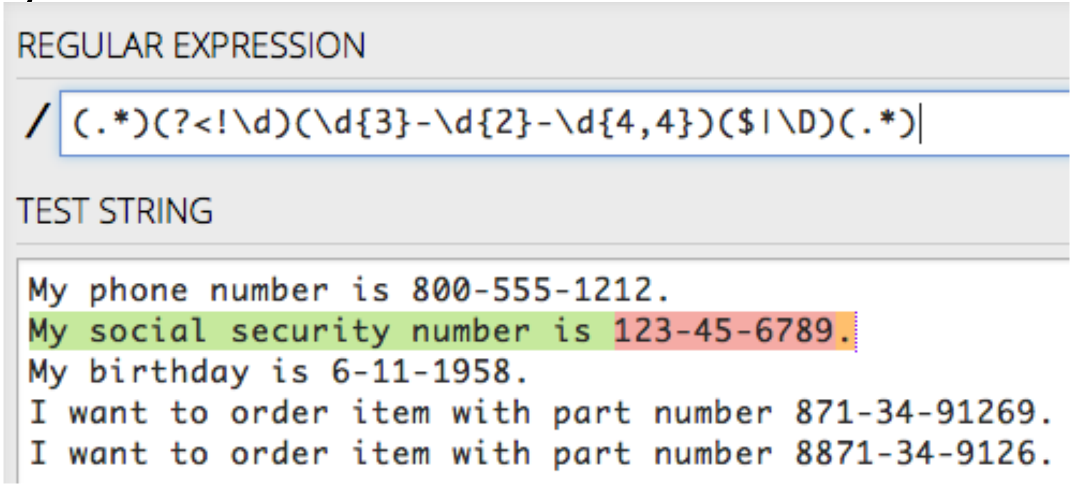
```
My phone number is 800-555-1212.
My social security number is 123-45-6789.
My birthday is 6-11-1958.
I want to order item with part number 8871-34-91268.
```

And finally:

- Don’t let it match when there are digits at the front or the end of the group:

(.*)(?<!\d)(\d{3}-\d{2}-\d{4,4})($|\D)(.*)

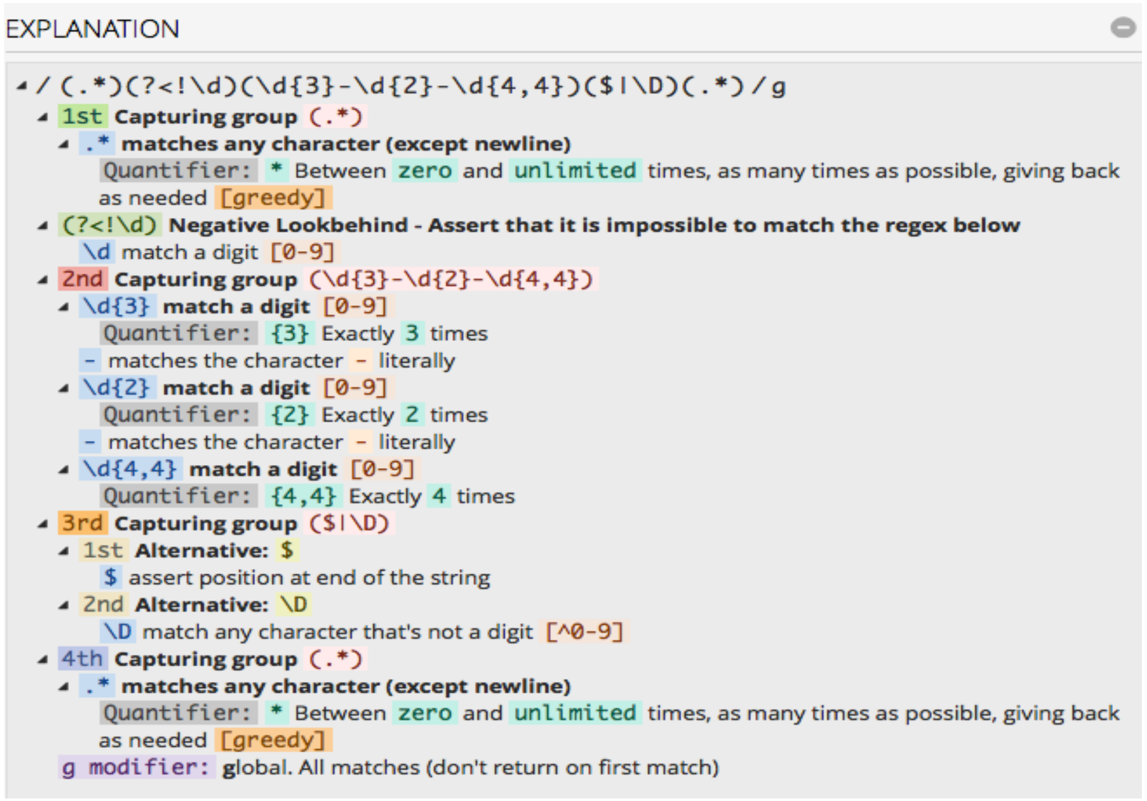
This matches only the SSN:



```
/ (.*)(?<!\d)(\d{3}-\d{2}-\d{4,4})($|\D)(.*)
```

```
My phone number is 800-555-1212.
My social security number is 123-45-6789.
My birthday is 6-11-1958.
I want to order item with part number 871-34-91269.
I want to order item with part number 8871-34-9126.
```

Regex101.com explains



EXPLANATION

```
/ (.*)(?<!\d)(\d{3}-\d{2}-\d{4,4})($|\D)(.*) / g
  1st Capturing group (.*)
    .* matches any character (except newline)
       Quantifier: * Between zero and unlimited times, as many times as possible, giving back
       as needed [greedy]
  (?<!\d) Negative Lookbehind - Assert that it is impossible to match the regex below
     \d match a digit [0-9]
  2nd Capturing group (\d{3}-\d{2}-\d{4,4})
    \d{3} match a digit [0-9]
       Quantifier: {3} Exactly 3 times
     - matches the character - literally
    \d{2} match a digit [0-9]
       Quantifier: {2} Exactly 2 times
     - matches the character - literally
    \d{4,4} match a digit [0-9]
       Quantifier: {4,4} Exactly 4 times
  3rd Capturing group ($|\D)
    1st Alternative: $
       $ assert position at end of the string
    2nd Alternative: \D
       \D match any character that's not a digit [^0-9]
  4th Capturing group (.*)
    .* matches any character (except newline)
       Quantifier: * Between zero and unlimited times, as many times as possible, giving back
       as needed [greedy]
  g modifier: global. All matches (don't return on first match)
```

The entry

```
[noSSN]
REGEX = (?m)^(.*)(?<!\d)(\d{3}-\d{2}-\d{4,4})($|\D)(.*)$
FORMAT = $1###-##-####$3$4
DEST_KEY = _raw
```

Why this REGEX

- Multi-line match, otherwise this will only work on single line events:

  `(?m)`

- Beginning of line and everything up to the last non-digit before our SSN in a capture group.

  `^(.*)`

- Here is the SSN (only match if not preceded by a digit)

  `(?<!\d)(\d{3}-\d{2}-\d{4,4})`

- The first non-digit if not the end of the line in one capture group, then everything from the last non-digit after the SSN to the end of the line in the last capture group

  `($|\D)(.*)$`

Why this FORMAT

- $1 is to be replaced with the first capture group
- $2 would be the SSN, but it isn't included in the FORMAT
- $3 is to be replaced with the single non-digit after the SSN (if it exists)
- $4 is to be replaced with the end capture group
- The SSN is substituted with hashes

```
$1###-##-####$3$4
```

REX and REGEX commands

When to use REX

- You **don't always** want to extract the data
- You want to extract data from a field that is already extracted
- You don't have access to field extractions (permissions, etc.)
- You are in a hurry or you are doing a proof-of-concept

REX example

- **index=ics.eup sourcetype=voice* Description | rex "Description=(?P<description>[^\^]+)" | rex field=description "From (?P<start>.+) to (?P<end>.+?):\s”**

- Aug 2 20:32:28 l13772 CPCMl13772: %local7-2-ALARM: 16$Description= Number of AuthenticationFailed events exceeds configured threshold during configured interval of time 1 within 3 minutes on cluster StandAloneCluster. There are 2 AuthenticationFailed events (up to 30) received during the monitoring interval From Wed Aug 03 10:25:00 PHT 2016 to Wed Aug 03 10:28:00 PHT 2016:   TimeStamp : 8/3/16 10:26 AM LoginFrom : 10.127.34.40 Interface : VMREST UserID : JacobMD AppID : Cisco Tomcat ClusterID : NodeID : APPHMANAOVM001 TimeStamp : Wed Aug 03 10:26:13 PHT 2016 TimeStam::Status=2,cleared^Severity=minor^Acknowledged=no^CUSTOMER=Cisco Prime Collaboration^Private IP Address=10.160.17.24^Default Alarm Name=AuthenticationFailed^Managed Object=10.160.17.24^Managed Object Type=Unity Connection^MODE=2;Alarm ID=343815480^Component=10.160.17.24\x00000

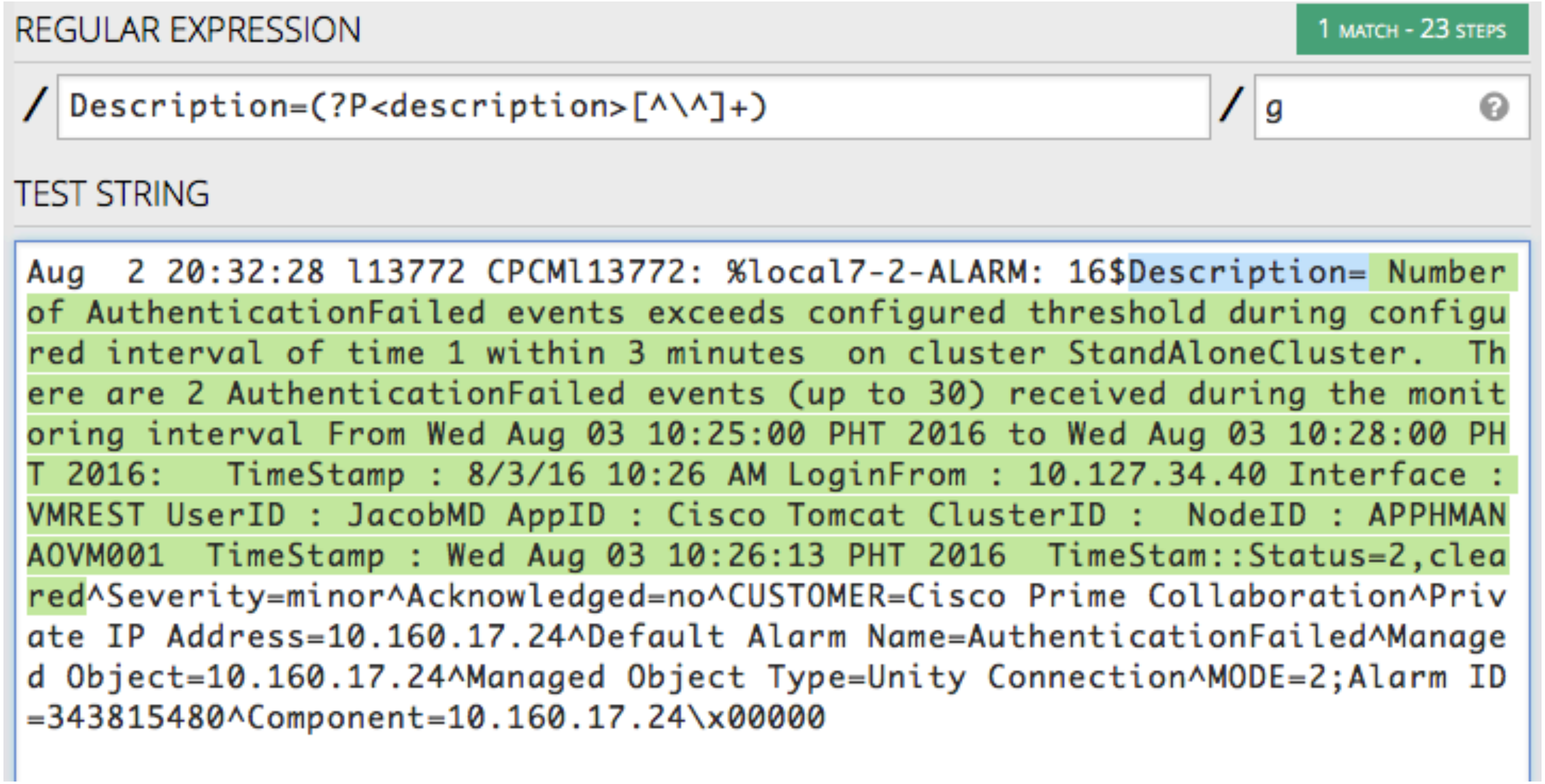
REX commands

- index=ics.eup sourcetype=voice* Description
  | rex "Description=(?P<description>[^\^]+)"
  | rex field=description "From (?P<start>.+) to (?P<end>.+?):\s”

- SYNTAX:
  | rex [field=*fieldname*] “regex”

- Also available:
  | rex mode=sed

First rex – get the description



REGULAR EXPRESSION

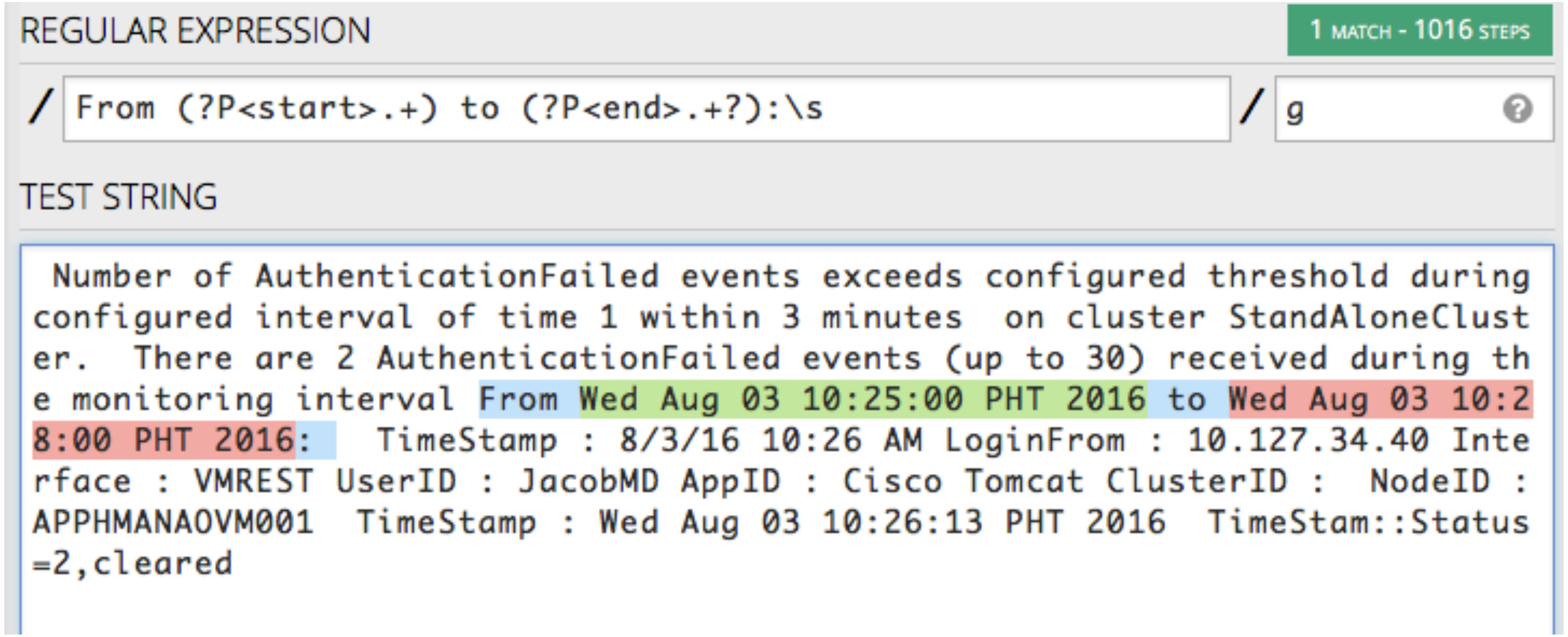
1 MATCH - 23 STEPS

```
/ Description=(?P<description>[^\^]+) / g
```

TEST STRING

```
Aug  2 20:32:28 l13772 CPCMl13772: %local7-2-ALARM: 16$Description= Number
of AuthenticationFailed events exceeds configured threshold during configu
red interval of time 1 within 3 minutes  on cluster StandAloneCluster.  Th
ere are 2 AuthenticationFailed events (up to 30) received during the monit
oring interval From Wed Aug 03 10:25:00 PHT 2016 to Wed Aug 03 10:28:00 PH
T 2016:   TimeStamp : 8/3/16 10:26 AM LoginFrom : 10.127.34.40 Interface :
VMREST UserID : JacobMD AppID : Cisco Tomcat ClusterID :  NodeID : APPHMAN
AOVM001  TimeStamp : Wed Aug 03 10:26:13 PHT 2016  TimeStam::Status=2,clea
red^Severity=minor^Acknowledged=no^CUSTOMER=Cisco Prime Collaboration^Priv
ate IP Address=10.160.17.24^Default Alarm Name=AuthenticationFailed^Manage
d Object=10.160.17.24^Managed Object Type=Unity Connection^MODE=2;Alarm ID
=343815480^Component=10.160.17.24\x00000
```

Second rex – get the start and end



REGULAR EXPRESSION

1 MATCH - 1016 STEPS

```
/ From (?P<start>.+) to (?P<end>.+?):\s / g
```

TEST STRING

```
 Number of AuthenticationFailed events exceeds configured threshold during
configured interval of time 1 within 3 minutes  on cluster StandAloneClust
er.  There are 2 AuthenticationFailed events (up to 30) received during th
e monitoring interval From Wed Aug 03 10:25:00 PHT 2016 to Wed Aug 03 10:2
8:00 PHT 2016:   TimeStamp : 8/3/16 10:26 AM LoginFrom : 10.127.34.40 Inte
rface : VMREST UserID : JacobMD AppID : Cisco Tomcat ClusterID :  NodeID :
APPHMANAOVM001  TimeStamp : Wed Aug 03 10:26:13 PHT 2016  TimeStam::Status
=2,cleared
```

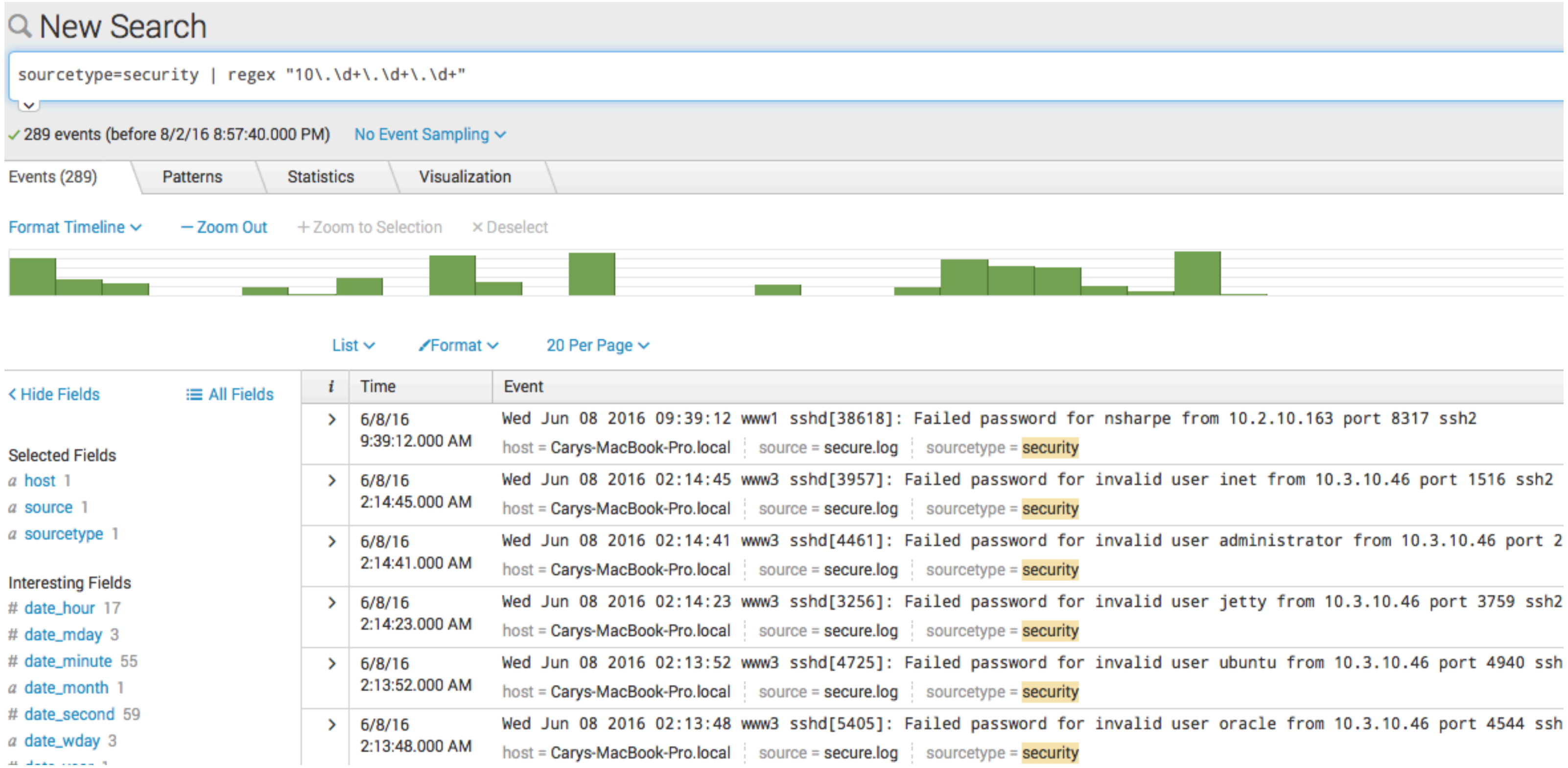
When to use REGEX

- To filter out events/data that you don’t want included in the pipeline

- This is like **search** on steroids, but doesn’t replace **search**

- Only used as a filter

Regex example


New Search

```
sourcetype=security | regex "10\.\d+\.\d+\.\d+"
```

✓ 289 events (before 8/2/16 8:57:40.000 PM) No Event Sampling

Events (289) | Patterns | Statistics | Visualization

Format Timeline · Zoom Out · Zoom to Selection · Deselect

List · Format · 20 Per Page

Hide Fields · All Fields

Selected Fields
- a host 1
- a source 1
- a sourcetype 1

Interesting Fields
- # date_hour 17
- # date_mday 3
- # date_minute 55
- a date_month 1
- # date_second 59
- a date_wday 3

i	Time	Event
>	6/8/16 9:39:12.000 AM	`Wed Jun 08 2016 09:39:12 www1 sshd[38618]: Failed password for nsharpe from 10.2.10.163 port 8317 ssh2` host = Carys-MacBook-Pro.local source = secure.log sourcetype = security
>	6/8/16 2:14:45.000 AM	`Wed Jun 08 2016 02:14:45 www3 sshd[3957]: Failed password for invalid user inet from 10.3.10.46 port 1516 ssh2` host = Carys-MacBook-Pro.local source = secure.log sourcetype = security
>	6/8/16 2:14:41.000 AM	`Wed Jun 08 2016 02:14:41 www3 sshd[4461]: Failed password for invalid user administrator from 10.3.10.46 port 2` host = Carys-MacBook-Pro.local source = secure.log sourcetype = security
>	6/8/16 2:14:23.000 AM	`Wed Jun 08 2016 02:14:23 www3 sshd[3256]: Failed password for invalid user jetty from 10.3.10.46 port 3759 ssh2` host = Carys-MacBook-Pro.local source = secure.log sourcetype = security
>	6/8/16 2:13:52.000 AM	`Wed Jun 08 2016 02:13:52 www3 sshd[4725]: Failed password for invalid user ubuntu from 10.3.10.46 port 4940 ssh` host = Carys-MacBook-Pro.local source = secure.log sourcetype = security
>	6/8/16 2:13:48.000 AM	`Wed Jun 08 2016 02:13:48 www3 sshd[5405]: Failed password for invalid user oracle from 10.3.10.46 port 4544 ssh` host = Carys-MacBook-Pro.local source = secure.log sourcetype = security

Breakdown

- Search:
  sourcetype=security | regex "10\.\d+\.\d+\.\d+”
- Only internal (10.*) IP addresses make it though the **regex** filter
- Search produces events, regex then limits those results passed on through the pipeline by a fancy regular expression
- *Yes, there are other way to do this, but this is a regex example*

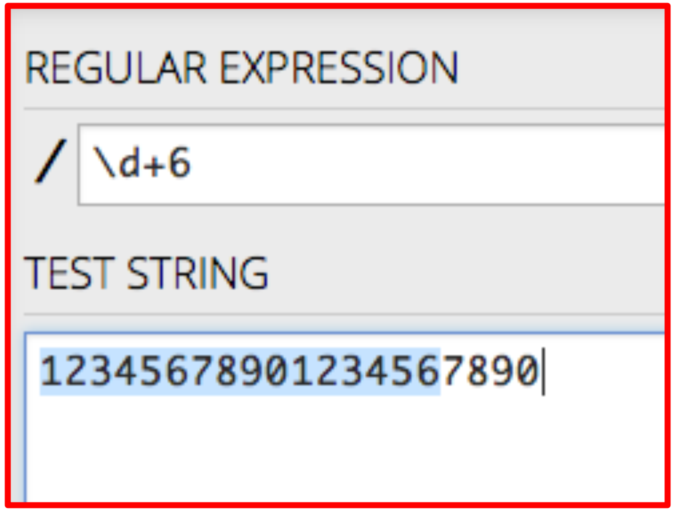
Rex vs Regex

- Use **rex** to extract fields
- Use **regex** to limit results
- Yes, you can use them in the same search:

```
sourcetype=security | rex "from (?P<src_ip>\d+\.\d+\.\d+\.\d+)"
| regex src_ip="(?<!10)\.\d+\.\d+\.\d+"
```

Greedy vs. Lazy matches

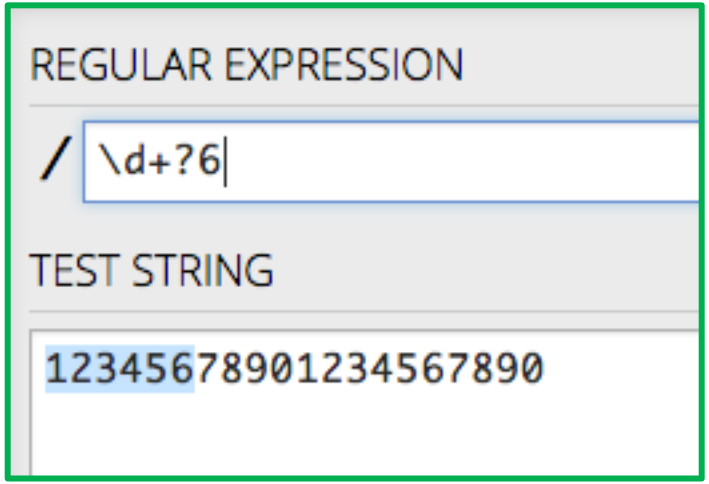
GREEDY vs LAZY


REGULAR EXPRESSION

```
/ \d+6
```

TEST STRING

```
12345678901234567890
```


REGULAR EXPRESSION

```
/ \d+?6
```

TEST STRING

```
12345678901234567890
```

Where is the difference?

- Greedy – Grab as much as you can
- Lazy – Grab as little as you can
- The lazy match will continue only as far as it needs to:

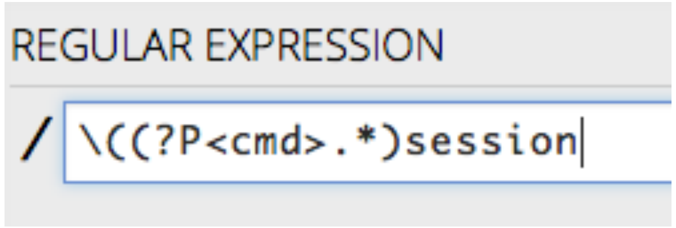
  <.+?> will match <12345>, while

  <.+> will match both <12345> and <12345><67890>

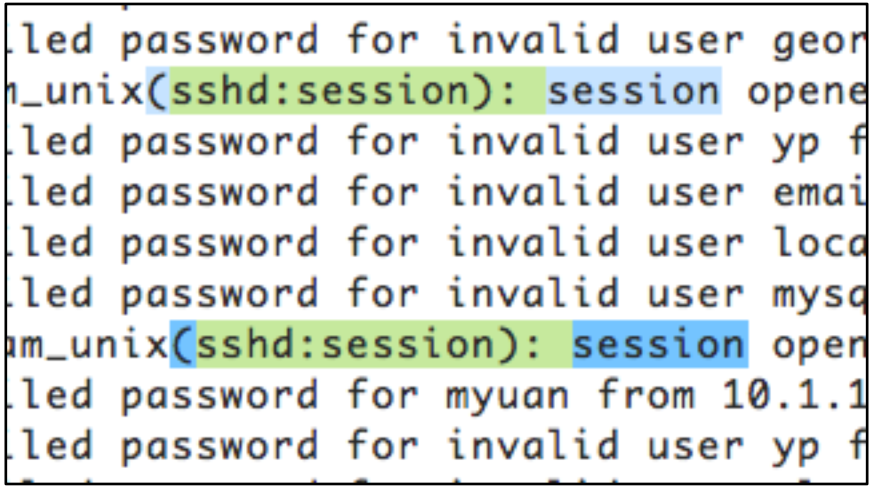
- SYNTAX: place a ? after a * or +

The lazy match only goes to the first instance of a match following the multiple match

GREEDY

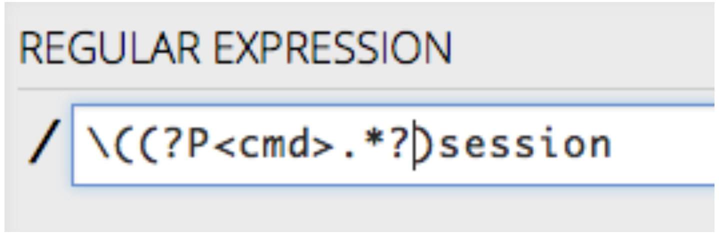

REGULAR EXPRESSION

```
/ \((?P<cmd>.*)session
```



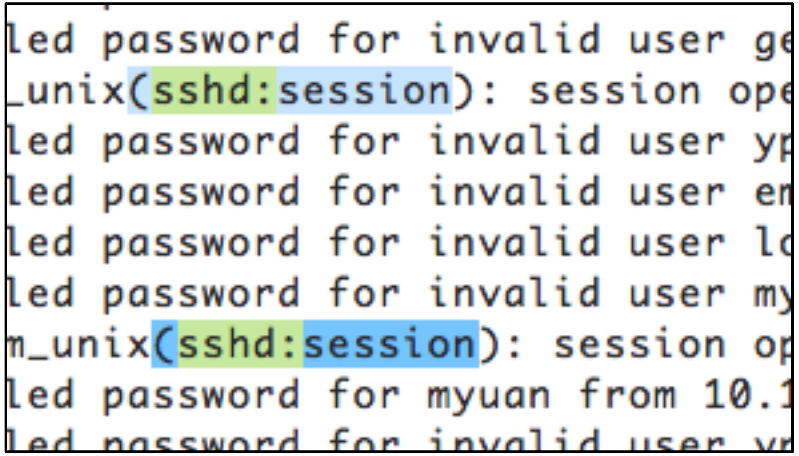
```
.led password for invalid user geor
n_unix(sshd:session): session opene
.led password for invalid user yp f
.led password for invalid user emai
.led password for invalid user loca
.led password for invalid user mysq
am_unix(sshd:session): session open
.led password for myuan from 10.1.1
.led password for invalid user yp f
```

LAZY



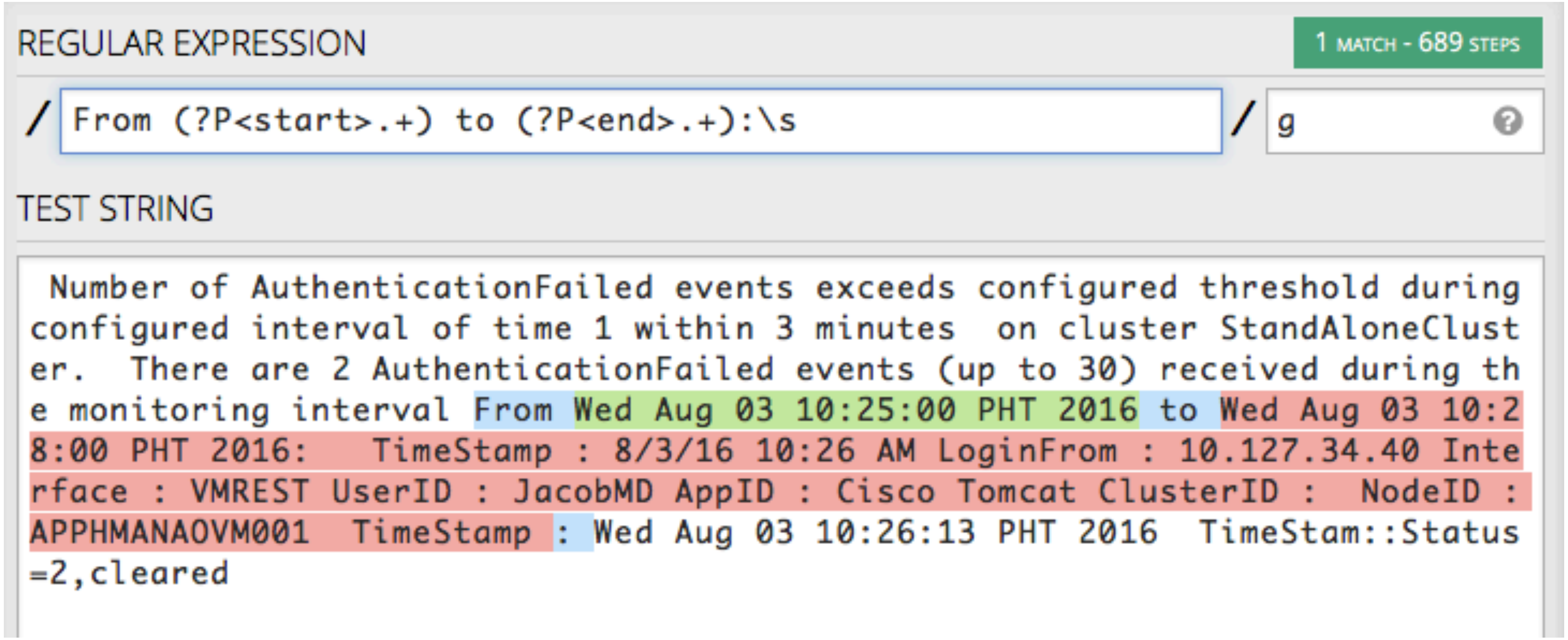
REGULAR EXPRESSION

```
/ \((?P<cmd>.*?)session
```



```
led password for invalid user ge
_unix(sshd:session): session ope
led password for invalid user yp
led password for invalid user en
led password for invalid user lo
led password for invalid user my
m_unix(sshd:session): session op
led password for myuan from 10.1
led password for invalid user yp
```

Second Look – Greedy



REGULAR EXPRESSION

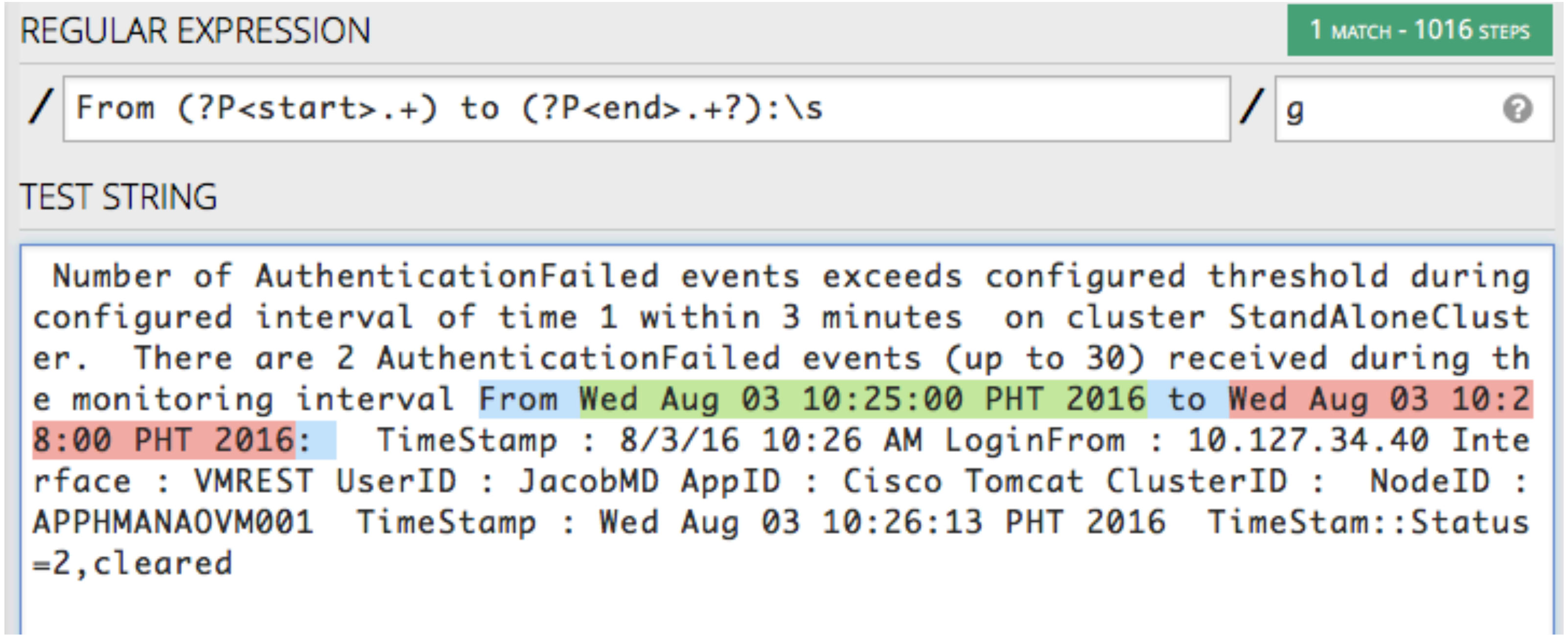
1 MATCH - 689 STEPS

```
/ From (?P<start>.+) to (?P<end>.+):\s / g
```

TEST STRING

```
 Number of AuthenticationFailed events exceeds configured threshold during
configured interval of time 1 within 3 minutes  on cluster StandAloneClust
er.  There are 2 AuthenticationFailed events (up to 30) received during th
e monitoring interval From Wed Aug 03 10:25:00 PHT 2016 to Wed Aug 03 10:2
8:00 PHT 2016:   TimeStamp : 8/3/16 10:26 AM LoginFrom : 10.127.34.40 Inte
rface : VMREST UserID : JacobMD AppID : Cisco Tomcat ClusterID :  NodeID :
APPHMANAOVM001  TimeStamp : Wed Aug 03 10:26:13 PHT 2016  TimeStam::Status
=2,cleared
```

Second Look - Lazy



REGULAR EXPRESSION

1 MATCH - 1016 STEPS

```
/ From (?P<start>.+) to (?P<end>.+?):\s / g
```

TEST STRING

```
 Number of AuthenticationFailed events exceeds configured threshold during
configured interval of time 1 within 3 minutes  on cluster StandAloneClust
er.  There are 2 AuthenticationFailed events (up to 30) received during th
e monitoring interval From Wed Aug 03 10:25:00 PHT 2016 to Wed Aug 03 10:2
8:00 PHT 2016:   TimeStamp : 8/3/16 10:26 AM LoginFrom : 10.127.34.40 Inte
rface : VMREST UserID : JacobMD AppID : Cisco Tomcat ClusterID :  NodeID :
APPHMANAOVM001  TimeStamp : Wed Aug 03 10:26:13 PHT 2016  TimeStam::Status
=2,cleared
```

Choose your path wisely

- Greedy may cross long segments
- Lazy may stop prematurely
- Try it on various data sets to make sure it will do what you want

Other Notes

Performance considerations

- Many field extractions can be costly
- Complex regular expressions can be costly
- Use the Job Inspector to see if there is a difference in doing one complex field extraction vs. many simple field extractions

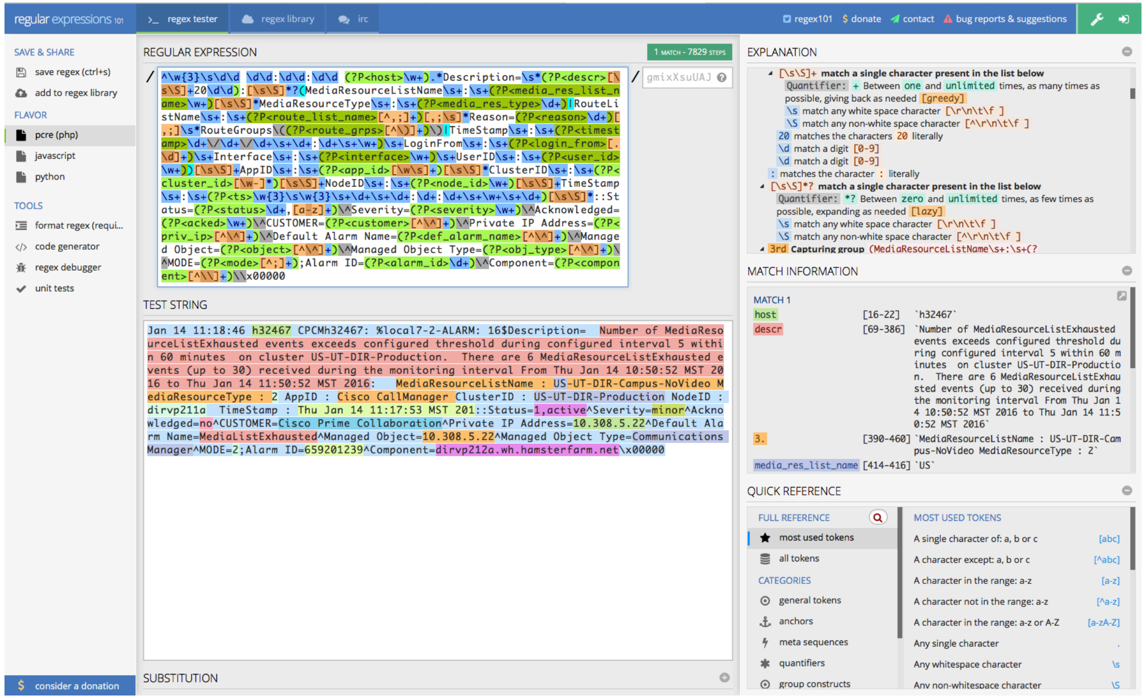
Maintenance considerations

- The complex field extractions may be easier to maintain than multiple simple extractions
- Your own field extractions will probably be easier to maintain than those created using the Field Extraction Tool

Tools

Regex Web Page

http://regex101.com/



regular expressions 101 | regex tester | regex library | irc | regex101 | $ donate | contact | bug reports & suggestions

SAVE & SHARE
- save regex (ctrl+s)
- add to regex library

FLAVOR
- pcre (php)
- javascript
- python

TOOLS
- format regex (requi...
- code generator
- regex debugger
- unit tests

$ consider a donation

REGULAR EXPRESSION — 1 MATCH - 7829 STEPS

```
/^\w{3}\s\d\d \d\d:\d\d:\d\d (?P<host>\w+).*Description=\s*(?P<descr>[\s\S]+20\d\d):[\s\S]*?(MediaResourceListName\s+:\s+(?P<media_res_list_name>\w+)[\s\S]*MediaResourceType\s+:\s+(?P<media_res_type>\d+)|RouteListName\s+:\s+(?P<route_list_name>[^,;]+)[,;\s]*Reason=(?P<reason>\d+)[,;]\s*RouteGroups\((?P<route_grps>[^\)]+)\)|TimeStamp\s+:\s+(?P<timestamp>\d+\/\d+\/\d+\s+\d+:\d+\s+\w+)\s+LoginFrom\s+:\s+(?P<login_from>[.\d]+)\s+Interface\s+:\s+(?P<interface>\w+)\s+UserID\s+:\s+(?P<user_id>\w+))[\s\S]+AppID\s+:\s+(?P<app_id>[\w\s]+)[\s\S]*ClusterID\s+:\s+(?P<cluster_id>[\w-]*)[\s\S]+NodeID\s+:\s+(?P<node_id>\w+)[\s\S]+TimeStamp\s+:\s+(?P<ts>\w{3}\s\w{3}\s+\d+\s+\d+:\d+:\d+\s+\w+\s+\d+)[\s\S]*::Status=(?P<status>\d+,[a-z]+)\^Severity=(?P<severity>\w+)\^Acknowledged=(?P<acked>\w+)\^CUSTOMER=(?P<customer>[^\^]+)\^Private IP Address=(?P<priv_ip>[^\^]+)\^Default Alarm Name=(?P<def_alarm_name>[^\^]+)\^Managed Object=(?P<object>[^\^]+)\^Managed Object Type=(?P<obj_type>[^\^]+)\^MODE=(?P<mode>[^;]+);Alarm ID=(?P<alarm_id>\d+)\^Component=(?P<component>[^\\]+)\\x00000/ gmixXsuUAJ
```

TEST STRING

```
Jan 14 11:18:46 h32467 CPCMh32467: %local7-2-ALARM: 16$Description=  Number of MediaResourceListExhausted events exceeds configured threshold during configured interval 5 within 60 minutes  on cluster US-UT-DIR-Production.  There are 6 MediaResourceListExhausted events (up to 30) received during the monitoring interval From Thu Jan 14 10:50:52 MST 2016 to Thu Jan 14 11:50:52 MST 2016:   MediaResourceListName : US-UT-DIR-Campus-NoVideo MediaResourceType : 2 AppID : Cisco CallManager ClusterID : US-UT-DIR-Production NodeID : dirvp211a  TimeStamp : Thu Jan 14 11:17:53 MST 201::Status=1,active^Severity=minor^Acknowledged=no^CUSTOMER=Cisco Prime Collaboration^Private IP Address=10.308.5.22^Default Alarm Name=MediaListExhausted^Managed Object=10.308.5.22^Managed Object Type=Communications Manager^MODE=2;Alarm ID=659201239^Component=dirvp212a.wh.hamsterfarm.net\x00000
```

SUBSTITUTION

EXPLANATION

- `[\s\S]+` **match a single character present in the list below**
  - Quantifier: `+` Between `one` and `unlimited` times, as many times as possible, giving back as needed `[greedy]`
  - `\s` match any white space character `[\r\n\t\f ]`
  - `\S` match any non-white space character `[^\r\n\t\f ]`
- `20` matches the characters `20` literally
- `\d` match a digit `[0-9]`
- `\d` match a digit `[0-9]`
- `:` matches the character `:` literally
- `[\s\S]*?` **match a single character present in the list below**
  - Quantifier: `*?` Between `zero` and `unlimited` times, as few times as possible, expanding as needed `[lazy]`
  - `\s` match any white space character `[\r\n\t\f ]`
  - `\S` match any non-white space character `[^\r\n\t\f ]`
- 3rd **Capturing group** `(MediaResourceListName\s+:\s+(?`

MATCH INFORMATION

```
MATCH 1
host                [16-22]   `h32467`
descr               [69-386]  `Number of MediaResourceListExhausted
                              events exceeds configured threshold du
                              ring configured interval 5 within 60 m
                              inutes  on cluster US-UT-DIR-Productio
                              n.  There are 6 MediaResourceListExhau
                              sted events (up to 30) received during
                              the monitoring interval From Thu Jan 1
                              4 10:50:52 MST 2016 to Thu Jan 14 11:5
                              0:52 MST 2016`
3.                  [390-460] `MediaResourceListName : US-UT-DIR-Cam
                              pus-NoVideo MediaResourceType : 2`
media_res_list_name [414-416] `US`
```

QUICK REFERENCE

FULL REFERENCE
- most used tokens
- all tokens

CATEGORIES
- general tokens
- anchors
- meta sequences
- quantifiers
- group constructs

MOST USED TOKENS	
A single character of: a, b or c	[abc]
A character except: a, b or c	[^abc]
A character in the range: a-z	[a-z]
A character not in the range: a-z	[^a-z]
A character in the range: a-z or A-Z	[a-zA-Z]
Any single character	.
Any whitespace character	\s
Any non-whitespace character	\S

Other relevant presentations

- **Become a regular expressions ninja and unlock your Splunk potential**

  - by *Gabriel Vasseur*

Splunk Answers and Docs

Learn from others – ask questions – get answers

- http://answers.splunk.com/

Splunk Documentation

- https://docs.splunk.com/Documentation/Splunk/6.4.3/Knowledge/AboutSplunkregularexpressions

Questions?

Contact Me

- cary.petterborg@ldschurch.org
- carypetterborg@gmail.com
- YouTube Channel: **The Splunk Hacker**

THANK YOU

Optional Content: Multiple dates with the same name

Social Media Feed Problem

Props.conf file must get the right timestamp, which is very difficult

- Twitter feeds in JSON format have multiple **postedTime** fields
- The first **postedTime** isn’t necessarily the most recent
- You can’t use the **postedTime** JSON key to determine the right timestamp
- You can’t rely on the first timestamp to be the right one

What do you do?

Original twitter feed:

{"id":"tag:search.twitter.com,2005:754700573774151685","objectType":"activity","actor":{"objectType":"person","id":"id:twitter.com:4889755594","link":"http://www.twitter.com/searcherdonate","displayName":"Вероника Гусева","**postedTime**":"2016-02-08T20:33:04.000Z","image":"https://abs.twimg.com/sticky/default_profile_images/default_profile_3_normal.png","summary":null,"links":[{"href":null,"rel":"me"}],"friendsCount":3,"followersCount":81,"listedCount":27,"statusesCount":26831,"twitterTimeZone":"Pacific Time (US & Canada)","verified":false,"utcOffset":"-25200","preferredUsername":"searcherdonate","languages":["ru"],"favoritesCount":0},"verb":"post","**postedTime**":"2016-07-17T15:33:33.000Z","generator":{"displayName":"searcherdonate","link":"http://searcherdonate.com"},"provider":{"objectType":"service","displayName":"Twitter","link":"http://www.twitter.com"},"link":"http://twitter.com/searcherdonate/statuses/754700573774151685","body":"Donate Car for Tax Credit\n\nhttps://t.co/0XtalQOyZa https://t.co/YRGGzqgIRg","object":{"objectType":"note","id":"object:search.twitter.com,2005:754700573774151685","summary":"Donate Car for Tax Credit\n\nhttps://t.co/0XtalQOyZa https://t.co/YRGGzqgIRg","link":"http://twitter.com/searcherdonate/statuses/754700573774151685","**postedTime**":"2016-07-17T15:33:33.000Z"},"favoritesCount":0,"twitter_entities":{"hashtags":[],"urls":[{"url":"https://t.co/0XtalQOyZa","expanded_url":"http://searcherdonate.com/2016/07/17/donate-car-for-tax-credit-475/","display_url":"searcherdonate.com/2016/07/17/don…","indices":[27,50]}],"user_mentions":[],"symbols":[],"media":[{"id":754700571177783296,"id_str":"754700571177783296","indices":[51,74],"media_url":"http://pbs.twimg.com/media/Cnk8K8DWEAAU9Zw.jpg","media_url_https":"https://pbs.twimg.com/media/Cnk8K8DWEAAU9Zw.jpg","url":"https://t.co/YRGGzqgIRg","display_url":"pic.twitter.com/YRGGzqgIRg","expanded_url":"http://twitter.com/searcherdonate/status/754700573774151685/photo/1","type":"photo","sizes":{"medium":{"w":480,"h":360,"resize":"fit"},"small":{"w":480,"h":360,"resize":"fit"},"thumb":{"w":150,"h":150,"resize":"crop"},"large":{"w":480,"h":360,"resize":"fit"}}}]},"twitter_extended_entities":{"media":[{"id":754700571177783296,"id_str":"754700571177783296","indices":[51,74],"media_url":"http://pbs.twimg.com/media/Cnk8K8DWEAAU9Zw.jpg","media_url_https":"https://pbs.twimg.com/media/Cnk8K8DWEAAU9Zw.jpg","url":"https://t.co/YRGGzqgIRg","display_url":"pic.twitter.com/YRGGzqgIRg","expanded_url":"http://twitter.com/searcherdonate/status/754700573774151685/photo/1","type":"photo","sizes":{"medium":{"w":480,"h":360,"resize":"fit"},"small":{"w":480,"h":360,"resize":"fit"},"thumb":{"w":150,"h":150,"resize":"crop"},"large":{"w":480,"h":360,"resize":"fit"}}}]},"twitter_filter_level":"low","twitter_lang":"en","retweetCount":0,"gnip":{"matching_rules":[{"value":"Donate","tag":null}],"urls":[{"url":"https://t.co/0XtalQOyZa","expanded_url":"http://searcherdonate.com/2016/07/17/donate-car-for-tax-credit-475/","expanded_status":200},{"url":"https://t.co/YRGGzqgIRg","expanded_url":"http://twitter.com/searcherdonate/status/754700573774151685/photo/1","expanded_status":200}],"klout_score":42,"language":{"value":"en"}}}

Examination of dates:

Note newest date for postedTime:

- "displayName":"Вероника Гусева","**postedTime**":"2016-02-08T20:33:04.000Z”
- "verb":"post","**postedTime**":"**2016-07-17T15:33:33.000Z**”
- "link":"http://twitter.com/searcherdonate/statuses/754700573774151685","**postedTime**":"2016-07-17T15:33:33.000Z”

Before latest **postedTime** comes **“verb”:”post”**

Original tweet feed:

{"id":"tag:search.twitter.com,2005:754693294882295810","objectType":"activity","actor":{"objectType":"person","id":"id:twitter.com:2987589649","link":"http://www.twitter.com/lauramessner14","dis
playName":"laura.the.escaper","**postedTime**":"2015-01-17T22:22:12.000Z","image":"https://pbs.twimg.com/profile_images/713992821766758400/3HDVKsXR_normal.jpg","summary":"forever escaper
• MDE is my Everything • Chris Ryan is the cutest • YouTube is my life • Amanda is my potato • [follow me on insta
laura_messner14]","links":[{"href":null,"rel":"me"}],"friendsCount":1534,"followersCount":1281,"listedCount":24,"statusesCount":55756,"twitterTimeZone":null,"verified":false,"utcOffset":null,"preferre
dUsername":"lauramessner14","languages":["en"],"favoritesCount":3244},"verb":"share","**postedTime**":"2016-07-17T15:04:37.000Z","generator":{"displayName":"Twitter for
iPhone","link":"http://twitter.com/download/iphone"},"provider":{"objectType":"service","displayName":"Twitter","link":"http://www.twitter.com"},"link":"http://twitter.com/lauramessner14/statuses
/754693294882295810","body":"RT @VansWarpedTour: Holmdel, donate 3 canned goods, $5, or a used cell phone at the @feedourchildren tent and get express entry!
https://t.…","object":{"id":"tag:search.twitter.com,2005:754646851442253824","objectType":"activity","actor":{"objectType":"person","id":"id:twitter.com:17503591","link":"http://www.twitter.com/Va
nsWarpedTour","displayName":"Vans Warped Tour","**postedTime**":"2008-11-20T03:36:43.000Z","image":"https://pbs.twimg.com/profile_images/638426457211965442/-
WWswoJQ_normal.jpg","summary":"On the road through August 13th. Tickets available now and complete lineup details, Warped 101 and more can be found on the Warped
site","links":[{"href":"http://www.vanswarpedtour.com","rel":"me"}],"friendsCount":118,"followersCount":939364,"listedCount":3417,"statusesCount":24198,"twitterTimeZone":"Pacific Time (US &
Canada)","verified":true,"utcOffset":"-25200","preferredUsername":"VansWarpedTour","languages":["en"],"favoritesCount":1912},"verb":"post","**postedTime**":"2016-07-
17T12:00:04.000Z","generator":{"displayName":"Twitter
Ads","link":"https://ads.twitter.com"},"provider":{"objectType":"service","displayName":"Twitter","link":"http://www.twitter.com"},"link":"http://twitter.com/VansWarpedTour/statuses/754646851442
253824","body":"Holmdel, donate 3 canned goods, $5, or a used cell phone at the @feedourchildren tent and get express entry!
https://t.co/n6UrMrM6Oq","object":{"objectType":"note","id":"object:search.twitter.com,2005:754646851442253824","summary":"Holmdel, donate 3 canned goods, $5, or a used cell phone at the
@feedourchildren tent and get express entry! https://t.co/n6UrMrM6Oq","link":"http://twitter.com/VansWarpedTour/statuses/754646851442253824","**postedTime**":"2016-07-
17T12:00:04.000Z"},"favoritesCount":221,"twitter_entities":{"hashtags":[],"urls":[{"url":"https://t.co/n6UrMrM6Oq","expanded_url":"http://smarturl.it/FeedOurChildrenNow","display_url":"smarturl.it/
FeedOurChildre…","indices":[109,132]}],"user_mentions":[{"screen_name":"FeedOurChildren","name":"FeedOurChildrenNOW!","id":116930972,"id_str":"116930972","indices":[64,80]}],"symbols":[]},"t
witter_filter_level":"low","twitter_lang":"en"},"favoritesCount":0,"twitter_entities":{"hashtags":[],"urls":[{"url":"https://t.co/n6UrMrM6Oq","expanded_url":"http://smarturl.it/FeedOurChildrenNow","d
isplay_url":"smarturl.it/FeedOurChildre…","indices":[139,140]}],"user_mentions":[{"screen_name":"VansWarpedTour","name":"Vans Warped
Tour","id":17503591,"id_str":"17503591","indices":[3,18]},{"screen_name":"FeedOurChildren","name":"FeedOurChildrenNOW!","id":116930972,"id_str":"116930972","indices":[84,100]}],"symbols":[]},"
twitter_filter_level":"low","twitter_lang":"en","retweetCount":61,"gnip":{"matching_rules":[{"value":"Donate","tag":null}],"urls":[{"url":"https://t.co/n6UrMrM6Oq","expanded_url":"https://www.faceb
ook.com/warpedtour/photos/p.10153673105303435/10153673105303435/?type=3&theater","expanded_status":403}],"klout_score":43,"language":{"value":"en"}}}

Examination of dates:

Note newest date for postedTime:

- “displayName":"laura.the.escaper","**postedTime**":"2015-01-17T22:22:12.000Z"
- “verb":"share","**postedTime**":"**2016-07-17T15:04:37.000Z**”
- ,"displayName":"Vans Warped Tour","**postedTime**":"2008-11-20T03:36:43.000Z”
- "verb":"post","**postedTime**":"2016-07-17T12:00:04.000Z”
- "link":"http://twitter.com/VansWarpedTour/statuses/754646851442253824","**postedTime**":"2016-07-17T12:00:04.000Z”

All retweeted items have **“verb”:”share”** before **postedTime**, which is the latest time for retweets

Sed – the stream editor

```
sed  \
-e 's/"verb":"share","postedTime":"/"verb":"share","truePostedTime":"/' \
-e '/truePostedTime/!s/"verb":"post","postedTime":"/"verb":"post","truePostedTime":"/' \
-e '/^$/d’
```

- You need to change the right postedTime to another name
  - **postedTime** -> **truePostedTime**
- Modify for the retweeted **postedTime** first
- Then if there is no **truePostedTime**, modify for original post
- Then get rid of blank lines (not discussed, but helpful)

Use Unbuffered Output

- Buffered output will break lines at something like 4096 bytes.
- This will make Splunk break events up improperly
- Definitely breaks JSON formatting
- Use unbuffered output to make the entire event be written/output

Curl & Sed Unbuffered

Shortened to make it easier to read:

```
curl -N -s --compressed –u$USERINFO $GNIPSTREAM | \
sed –u \
 -e \ 's/"share","postedTime"/"share","truePostedTime"/' -e \
'/truePostedTime/!s/"post","postedTime"/"post","truePostedTime"
/' -e '/^$/d' \
>>/app/gnip/log/inputstream.log 2>>/app/gnip/log/errors
```