



splunk®

Travelling in time with Splunk

How to deal with different time zones in global enterprises

Norbert Hamel

nhamel@splunk.com

October 2018

Forward-Looking Statements

During the course of this presentation, we may make forward-looking statements regarding future events or the expected performance of the company. We caution you that such statements reflect our current expectations and estimates based on factors currently known to us and that actual events or results could differ materially. For important factors that may cause actual results to differ from those contained in our forward-looking statements, please review our filings with the SEC.

The forward-looking statements made in this presentation are being made as of the time and date of its live presentation. If reviewed after its live presentation, this presentation may not contain current or accurate information. We do not assume any obligation to update any forward-looking statements we may make. In addition, any information about our roadmap outlines our general product direction and is subject to change at any time without notice. It is for informational purposes only and shall not be incorporated into any contract or other commitment. Splunk undertakes no obligation either to develop the features or functionality described or to include any such feature or functionality in a future release.

Splunk, Splunk>, Listen to Your Data, The Engine for Machine Data, Splunk Cloud, Splunk Light and SPL are trademarks and registered trademarks of Splunk Inc. in the United States and other countries. All other brand names, product names, or trademarks belong to their respective owners. © 2018 Splunk Inc. All rights reserved.

About me

Splunk Professional Services Consultant and Instructor

- ▶ Working with Splunk since 2011
- ▶ Started as customer (Vodafone Germany, Telco)
- ▶ Delivering Splunk Professional services across EMEA since 2014
- ▶ Delivering Splunk training in Germany for 3 years
- ▶ Presented at .conf2013 and several Splunk Live! events



Contents

Topics of this presentation

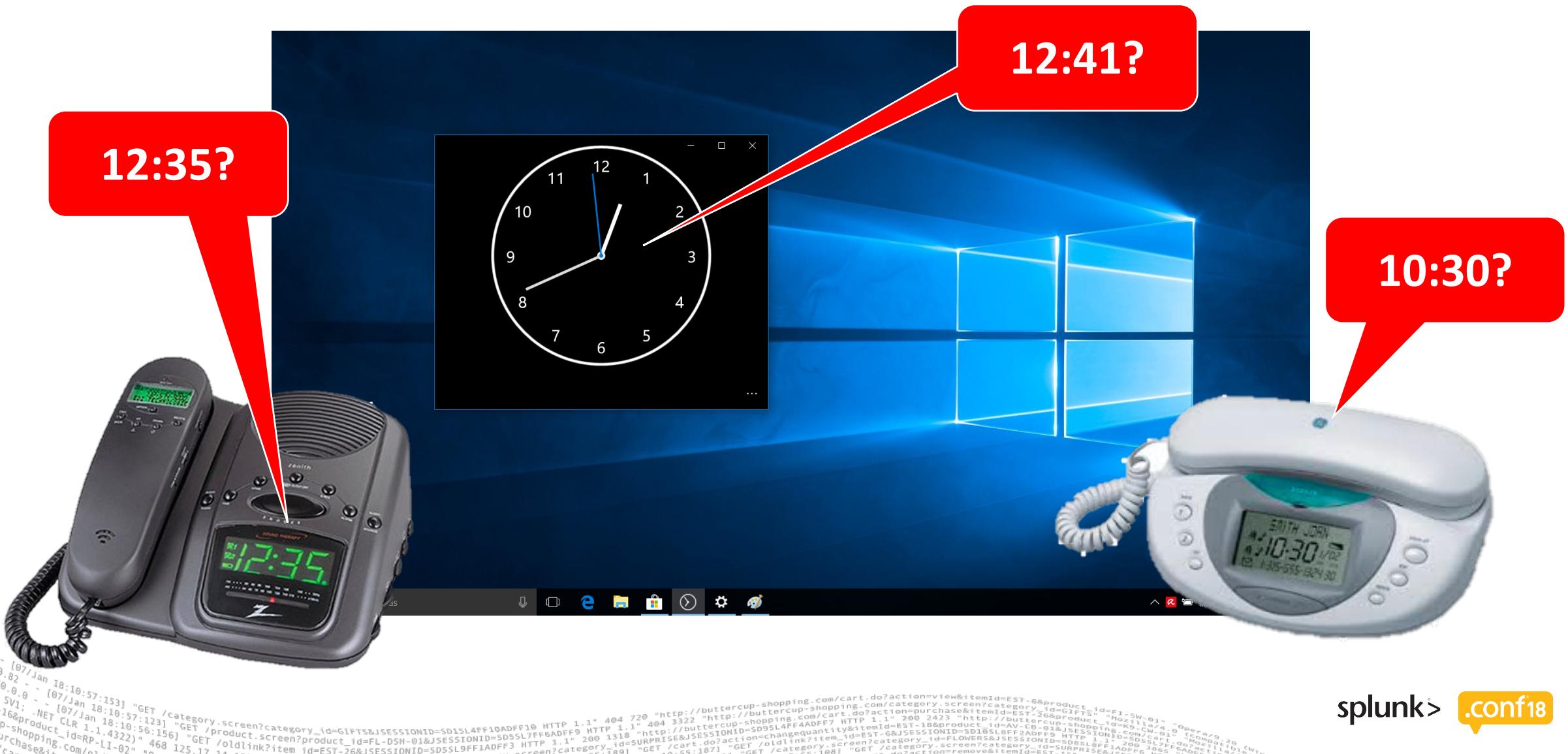
- ▶ About time zones
 - ▶ How Splunk detects time stamp and time zone
 - ▶ Ingest data without or including a proper time zone configuration
 - ▶ Add time zone information to raw events
 - ▶ Using time in searches

About time zones

Why you should care about time zones

Recently, at customer's office

Time lottery



Time zones in global companies

Just an example



San Francisco
2:00
2 AM



New York
5:00
5 AM



London
10:00
10 AM



Frankfurt
11:00
11 AM



New Delhi
14:30
2:30 PM



Shanghai
17:00
5 PM

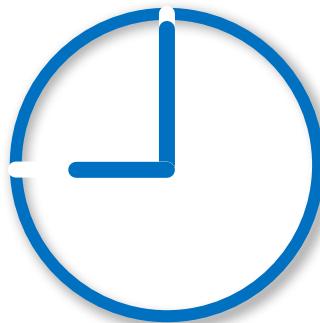
- ▶ Different time zones in regions like AMER, EMEA, APAC, different time zones within countries
 - 4 main TZ in USA – EST, CST, MST and PST
 - 11 in Russia – only 9 between 2010 and 2014
 - 12 time zones in France – world champion not only in football!
 - ▶ Time zones with 30 minutes shift like India
 - ▶ Time zones with / without daylight savings, daylight saving switches on different days
(New York: 11-MAR-2018 to 04-NOV-2018, London 25-MAR-2018 to 28-OCT-2018)

Why do we care?

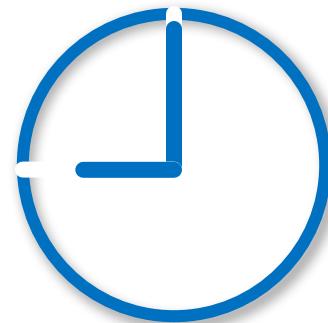


My whole world is UTC

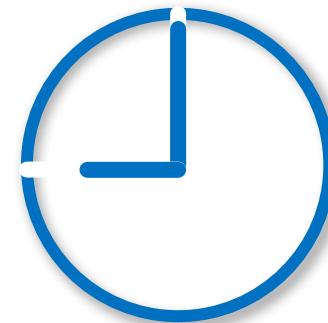
Admin's dirty hack to ignore time zone existence



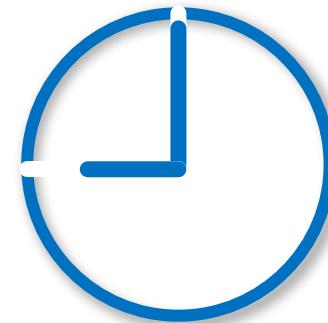
San Francisco
2:00
2 AM



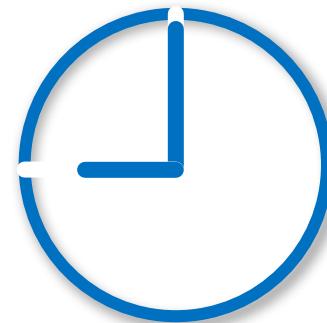
New York
5:00
5 AM



London
10:00
10 AM



Frankfurt
11:00
11 AM



**New Delhi
14:30
2:30 PM**



- ▶ Creating wrong timestamps in logs by intention
 - ▶ Very likely not all systems can be set up like this

How Splunk detects time stamp and time zone

Every event in Splunk needs a bit of time, please!



Refresher: how Splunk detects time stamp

Happens after line breaking, before transforms etc.

- ▶ Time stamp found in event with time AND date
 1. Use rules in props.conf defined for host, source, sourcetype (note precedence)

```
2018-Oct-03 10:00:00 CEST started scanning for client malware
scanned device XHT765RT4: no malware found, status OK (12)
scanned device KJU87RT5H: malware found, status critical (34)
scanned device KU876RTH6: no malware found, status OK (12)
```

```
2018-Oct-03 10:35:00 CEST started scanning for server malware
scanned device HT75RE$Z6: no malware found, status OK (12)
scanned device KIU98H7ZM: malware found, status critical (34)
scanned device ML098UZT7: no malware found, status OK (12)
```

1. Use system time or index

- ▶ Store time stamp in epoch / Unix format including milliseconds (if available)

Recommendation: Always define time extraction

Will reduce errors, will increase performance

► TIME_PREFIX

2018-Oct-03 10:00:00 here is a simple event

Oct 03 10:05:00 syslog1 [code] 2018-Oct-03 10:00:00 original event

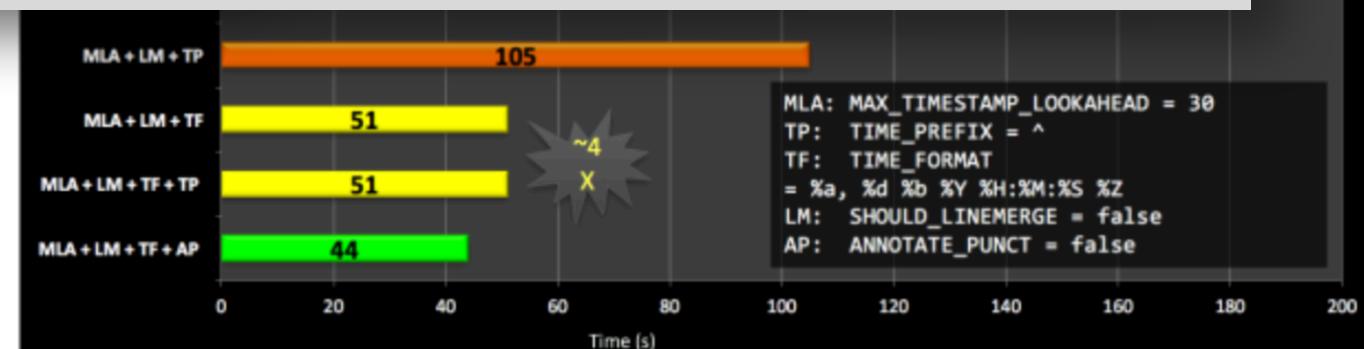
► Here is another event, where the timestamp can not be found in the first 128 characters. By default, this will not be treated as the time of the event 2018-Oct-03 10:00:00 remaining part of the event

- Default 128
- Critical in case time stamp is far away from event start

► TIME_FORMAT

- What is the human readable format of the time stamp (strftime abbreviations)

Critical in case of unusual formats



How Splunk detects time zone

And what if no time zone information is available?

- ▶ Use the time zone in event (for example, PST, -0800)
 - ▶ Use the TZ attribute set in props.conf, if the event matches the host, source, or sourcetype that the stanza specifies.
 - ▶ Use the time zone of Forwarder (Splunk version 6.x)
 - ▶ Use the time zone of the host that indexes the event

2018-Oct-03 10:00:00 PST here is a simple event with time zone info

```
[ host::sf-server* ]
```

TZ = PST

```
[ host::ny-server* ]
```

TZ = EST

Ingest data with or without time zone configuration



Demo: ingest data without time zone

What happens in the processing chain?

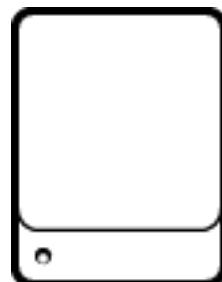


Configure time zone in props.conf

props.conf:

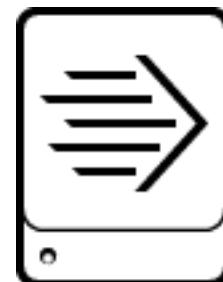
```
[source:::/var/log/mylogs/PST/* ]
```

TZ = PST



San Francisco Application Server

New York Log Server with UF



**London
Heavy
Forwarder**

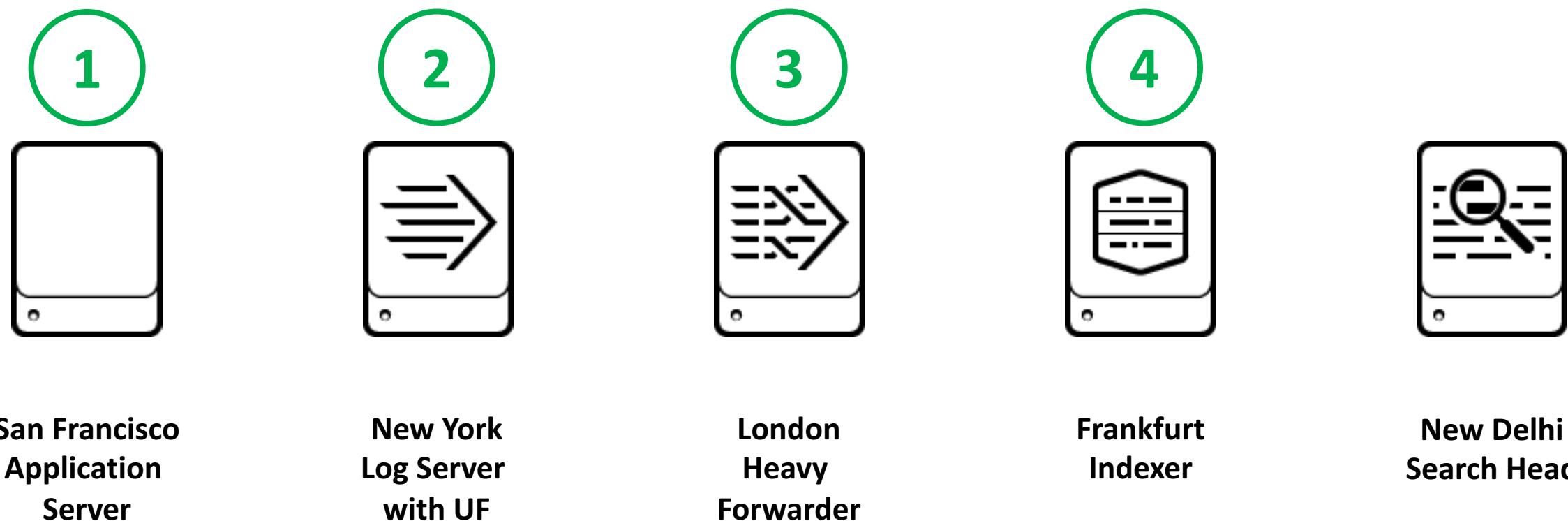


Frankfurt Indexer



New Delhi Search Head

Repeat demo: ingest data without time zone



Bonus: Configure time zone for Splunk DB connect

- ▶ Splunk DB connect does not take care about TZ settings in props.conf
- ▶ Instead, configure time zone for each DB connection separately

db_connections.conf

```
timezone = <time zone identifier>
# optional, default uses JVM time zone
# The identifier could be:
# an offset from UTC/Greenwich, that uses the same offset
# regardless given date-time e.g. +08:00
# an area where a specific set of rules for finding the offset
# from UTC/Greenwich apply e.g. Europe/Paris.
```

Add time zone information to raw events



Add time and time zone using unarchive_cmd

Suitable for static files

- ▶ Some applications generate a single file per message (e.g. XML)
- ▶ File content has no time information, but file name has date and time
- ▶ Splunk cannot retrieve time from file name (source)

`inputs.conf`

```
[monitor:///var/log/incomingData]
index = null

[batch:///var/log/modifiedData]
index = main
sourcetype = modifiedlogs
move_policy = sinkhole
```

`props.conf`

```
[ source:::/var/log/incomingData/* ]
invalid_cause = archive
unarchive_cmd = /path/to/TimestampFromFilename.sh

[ modifiedlogs ]
TIME_PREFIX=^
MAX_TIMESTAMP_LOOKAHEAD = 24
LINE_BREAKER = [ \r\n ]+
SHOULD_LINEMERGE = false
TIME_FORMAT = %Y-%m-%dT%H:%M:%S.%3N
```

Add time and time zone using unarchive_cmd

- ## ► Script Source Code Included in the Appendix slides

```

#!/bin/bash

# Loop over incoming directory
for file in "/var/log/incoming"/*.xml; do

    # Extract the timestamp
    timestamp=$(echo "$file" | sed -n 's/.*\(\([[:digit:]]\{17\}\)\).*/\1/p')

    # Format timestamp into Year-Month-Day Hour:Minute:Second.Subsecond
    timestamp="${timestamp:0:4}-${timestamp:4:2}-${timestamp:6:2}"
    ${timestamp:8:2}:${timestamp:10:2}:${timestamp:12:2}.${timestamp:14:3}"

    # Add timestamp into each line of the current file
    sed -i -e "s/^/$timestamp - /" $file

    # After modification move the file to the modified directory
    mv $file $dst

done

```



Add time zone using props and transforms

- ▶ Objective: add time zone information into events
 - ▶ Event before:
192.168.1.1 [2018-07-26T10:12:34] my log message 200 1234
 - ▶ Event after:
192.168.1.1 [2018-07-26T10:12:34 -0700] my log message 200 1234



```
inputs.conf  
[splunktcp://9991]  
route:has key: linebreaker:parsingQueue
```

```
props.conf
[source:///var/log/mylogswithouttimezone*]
TRANSFORMS-01=add time zone PST
```

```
transforms.conf
[add_time_zone_PST]
SOURCE_KEY = _raw
DEST_KEY = _raw
REGEX = (.*)\](.*)
FORMAT = $1 -0700 $2
```

Add time zone using props and transforms

- ▶ Objective: add time zone information into events
 - ▶ Event before:
192.168.1.1 [2018-07-26T10:12:34] my log message 200 1234
 - ▶ Event after:
192.168.1.1 [2018-07-26T10:12:34 -0700] my log message 200 1234



```
inputs.conf  
[splunktcp://9991]  
route:has key: linebreaker:parsingQueue
```

```
props.conf
[source:///var/log/mylogswithouttimezone*]
TRANSFORMS-01=add time zone PST
```

```
transforms.conf
[add_time_zone_PST]
SOURCE_KEY = _raw
DEST_KEY = _raw
REGEX = (.*)\](.*)
FORMAT = $1 -0700 $2
```

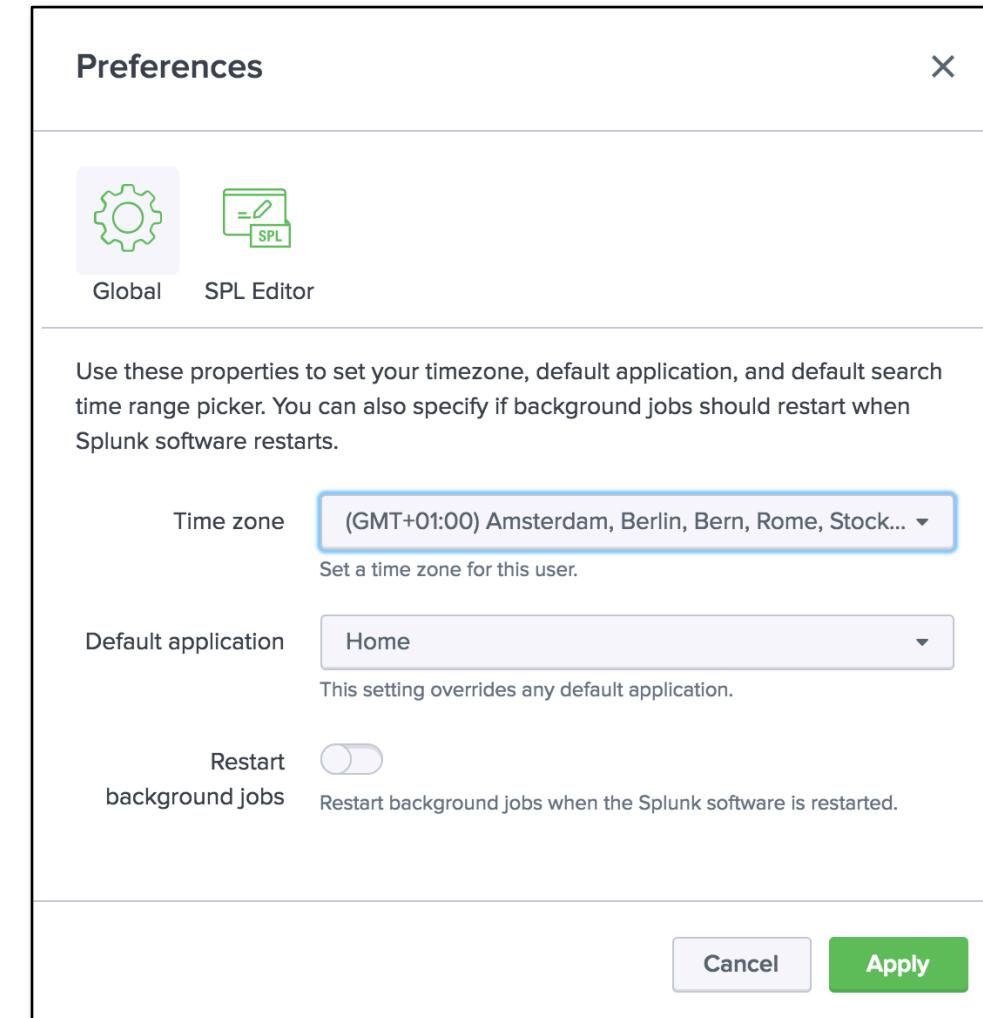
Using time in searches



Searching events in Splunk

Another view on the same events

- ▶ Any Splunk search uses time as 2nd important filter (after index)
- ▶ Search is related to the user's time zone as defined in Splunk Web preferences (can be different from system time)
- ▶ Note that relative historical searches (like "last 24 hours") usually use latest=now(), means results from the future are omitted



Searching events in Splunk

Another view on the same events

Representation of _time in user's time zone

The screenshot shows the Splunk interface with the following details:

- Search Bar:** index=_internal | head 10
- Results Summary:** 10 events (before 7/26/18 6:07:59.000)
- Event Sampling:** Event Sampling ▾
- Event List:**
 - Events (10):** Events tab is selected.
 - Format Timeline:** Format Timeline ▾
 - Selection:** Selection
 - Time Scale:** 100 milliseconds per column
 - List View Options:** List ▾, Format, 20 Per Page ▾
 - Selected Fields:** host 1, source 2, sourcetype 2
 - Interesting Fields:** # bytes 5, clientip 1, color 1, component 1
- Raw Event Log:**

	Time	Event
>	7/26/18 6:07:57.351 PM	127.0.0.1 - admin [26/Jul/2018:14:37:57.351 +0200] "GET /en-US/splunkd/_raw/servicesNS/nobody/search/search/jobs/1532608677.344?output_mode=json&_=1532608568855 HTTP/1.1" 200 2367 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/67.0.3396.99 Safari/537.36" - 00e8ac917b5ea28cc9888f25380caa74 5ms host = norberts-mbp.internal source = /Applications/splunk/var/log/splunk/splunkd_ui_access.log sourcetype = splunkd_ui_access
>	7/26/18 6:07:57.342 PM	127.0.0.1 - admin [26/Jul/2018:14:37:57.342 +0200] "GET /en-US/splunkd/_raw/services/search/timelparser?output_mode=json&time=-7d%40h&time=now&output_time_format=%25s.%25Q%7C%25Y-%25m-%25dT%25H%3A%25M%3A%25S.%25Q%253Az&_=1532608568854 HTTP/1.1" 200 109 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/67.0.3396.99 Safari/537.36" - 00e8ac917b5ea28cc9888f25380caa74 1ms host = norberts-mbp.internal source = /Applications/splunk/var/log/splunk/splunkd_ui_access.log sourcetype = splunkd_ui_access

Additional time fields

Another view on the same events

► Default time fields

- date_second, date_minute, date_hour, date_wday, date_mday, date_month, date_year, date_zone
- Generated at index time and stored in tsidx (check with `walk1ex` command)
- Only for events having human readable time stamps
- Extracting values from raw events, means without time zone conversion and without any other changes (props+transforms)
- Does NOT reflect time zone of user when searching

► Index time (_indextime)

- Epoch / Unix format without milliseconds
- Can be used to determine indexing delay and / or time zone issues



How to search in another time zone

- ▶ Set time zone in preferences to the target time zone 
 - ▶ Use starttime and endtime for searches instead of earliest and latest
 - ▶ Use the macro provided in this break out session
 - ▶ Create dashboard with time zone selectors



Use starttime and endtime

- ▶ By default we use earliest and latest in searches
- ▶ You can enter human readable times here, but only in fixed format
%m/%d/%Y:%H:%M:%S
- ▶ Alternatively, we can use starttime and endtime, which allows other formats including time zones:

```
index=_internal
starttime="26/07/2018:00:00:00 PST"
endtime="26/07/2018:02:00:00 PST"
timeformat="%d/%m/%Y:%H:%M:%S %Z"
```

The screenshot shows the Splunk web interface with a search bar containing the query: `index=_internal starttime="26/07/2018:00:00:00 PST" endtime="26/07/2018:02:00:00 PST" timeformat="%d/%m/%Y:%H:%M:%S %Z"`. Below the search bar, it says "9,843 events (7/26/18 10:00:00.000 AM to 7/26/18 12:00:00.000 PM) No Event Sampling". The main area displays a histogram and a table of event details. The table has columns: < Hide Fields, All Fields, i Time, and Event. One row is highlighted, showing the timestamp `7/26/18 11:59:52.381 AM`.

< Hide Fields	All Fields	i Time	Event
SELECTED FIELDS		> 7/26/18 11:59:52.381 AM	127.0.0.1 - admin [26/Jul/2018:11:59:52.381 +0200] "GET /er/health/splunkd?output_mode=json&_=1532590410747 HTTP/1.1" 200 2423 "http://buttercup-shopping.com/cart.do?action=changequantity&itemID=EST-18&productID=AUTOCAR-SSESSION.COM-01.1.4322" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/67.0.3396.87 Safari/537.36" - 3f736ec5e8eaf7acda4f5753f1a6abcb 1ms
INTERESTING FIELDS			host = norberts-mbp.intern source = /Applications/splunk/var/splunkd_ui_access



Macro to convert times in different time zone

Code provided in appendix

The screenshot shows a Splunk search interface with the following details:

- Search Bar:** Contains the search command:


```
| makeresults
| fields - _time
| eval myearliest=round(strptime("2018-03-25 01:00:00", "%Y-%m-%d %H:%M:%S"),0)
| `timezoned_time(mytime, "%Y-%m-%d %H:%M:%S.%3N", "Europe/London")`
```
- Results Panel:** Shows 1 result from 7/25/18 1:00:00.000 PM to 7/26/18 1:47:54.000 PM. The event details are:

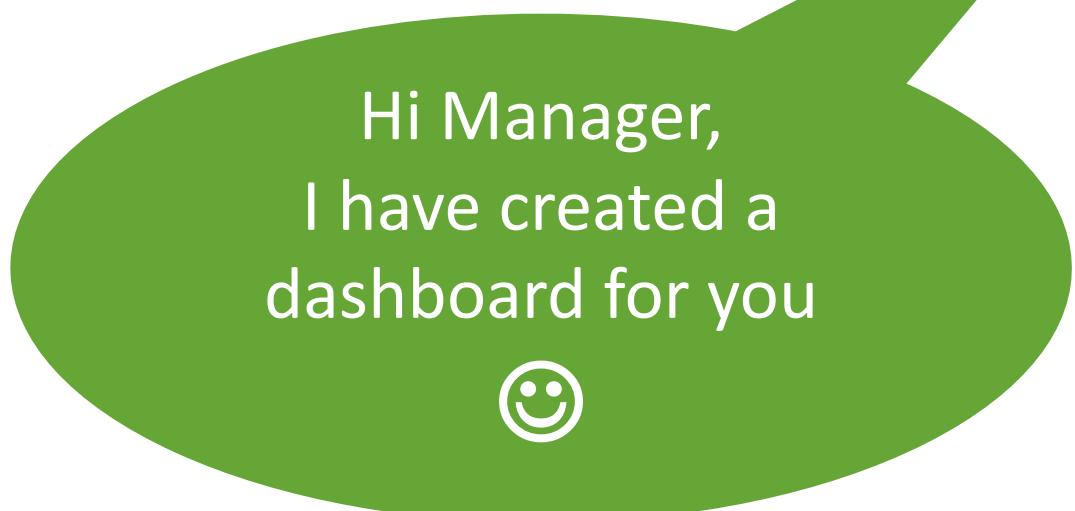
mytime	mytime__timezoned_region	mytime__timezoned_region_spl	mytime__timezoned_user	mytime__timezoned_utc
1521936000	2018-03-25 00:00:00.000 Europe/London	2018-03-25T00:00:00.000+00:00	2018-03-25 01:00:00.000 CET	2018-03-25 00:00:00.000 UTC
- Statistics Tab:** Shows 1 result.
- Visualizations Tab:** Not selected.
- Toolbar:** Includes Save As, Close, Last 24 hours, and a green search button.

Why do we care?

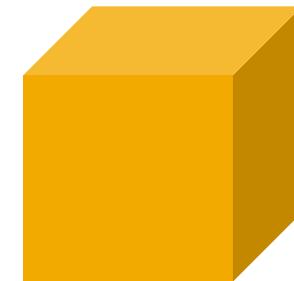


Hi Splunker,
what is our revenue
from yesterday
again?

Germany



Hi Manager,
I have created a
dashboard for you



Thailand



Malaysia



Dashboard with custom time zone selectors

splunk>enterprise App: Search & Reporting ▾

H Administrator ▾ Messages ▾ Settings ▾ Activity ▾ Help ▾ Find

Search Datasets Reports Alerts Dashboards

> Search & Reporting

Revenue by region and day

field1

New York x Berlin x

New Dehli x London x

Singapore x

San Francisco x

field2

2018-10-03 ▾ X

Hide Filters

2018-10-01

2018-10-02

✓ 2018-10-03

2018-10-04

2018-10-05

Revenue as of 2018-10-03 per region

region	revenue
Berlin	50,000
London	750,000
New Dehli	100,000
New York	350,000
San Francisco	500,000
Singapore	300,000

Edit Export ...

Revenue as of 2018-10-03 per region

revenue

Thank You

**Don't forget to rate this session
in the .conf18 mobile app**



Appendix



Add time and time zone using unarchive_cmd

Script source code

```
#!/bin/bash

# Loop over incoming directory
for file in "/var/log/incoming"/*.xml; do

    # Extract the timestamp
    timestamp=$(echo "$file" | sed -n 's/.*\(\[[[:digit:]]\{17\}\)\.*/\1/p')

    # Format timestamp into Year-Month-Day Hour:Minute:Second.Subsecond
    timestamp="${timestamp:0:4}-${timestamp:4:2}-${timestamp:6:2}"
    ${timestamp:8:2}:${timestamp:10:2}:${timestamp:12:2}.${timestamp:14:3}"

    # Add timestamp into each line of the current file
    sed -i -e "s/^/$timestamp - /" $file

    # After modification move the file to the modified directory
    mv $file $dst

done
```

done

How Splunk detects year, if missing

Sub title

- ▶ Get either the date of the last event current system time
 - ▶ Perform some logical checks
 - ▶ Assign the year to the time stamp
 - ▶ Check if that falls into the allowed time range
(MAX_DAYS_AGO – 2000 days, MAX_DAYS_HENCE – 2 days)
 - ▶ Examples:
 - New event with time stamp 26 Jun retrieved on May 26, 2017 will end up as 26-JUN-2016
 - New event with time stamp 10 Apr retrieved on May 26, 2017 will end up as 10-APR-2018, which is outside of the allowed range, therefore the current date 26-MAY-2017 is used

Time shift during DST switch

timestamp	timestring	delta
1521921600	2018-03-24 20:00:00 Europe/London	3600
1521925200	2018-03-24 21:00:00 Europe/London	3600
1521928800	2018-03-24 22:00:00 Europe/London	3600
1521932400	2018-03-24 23:00:00 Europe/London	3600
1521936000	2018-03-25 00:00:00 Europe/London	3600
1521939600	2018-03-25 02:00:00 Europe/London	3600
1521943200	2018-03-25 03:00:00 Europe/London	3600
1521946800	2018-03-25 04:00:00 Europe/London	3600
1521950400	2018-03-25 05:00:00 Europe/London	3600
1521954000	2018-03-25 06:00:00 Europe/London	3600
1521957600	2018-03-25 07:00:00 Europe/London	3600
1521961200	2018-03-25 08:00:00 Europe/London	3600
1521964800	2018-03-25 09:00:00 Europe/London	3600
1521968400	2018-03-25 10:00:00 Europe/London	3600

Macro to convert times in different time zone

```

eval _ma_dateformat="$target_timeformat"
| eval _ma_locTZ="$target_timezone"
| eval _ma_timestamp=$utc_timestamp
| eval _ma_usrTime=strftime(_ma_timestamp, _ma_dateformat." %Z")
| eval _ma_utcTime=strftime(_ma_timestamp - (strptime(strftime(_ma_timestamp, _ma_dateformat."Z"),
_ma_dateformat.%Z)-strptime(strftime(_ma_timestamp, _ma_dateformat), _ma_dateformat)), _ma_dateformat." UTC")
| eval _ma_locTime=strftime(2*_ma_timestamp - strptime(strftime(_ma_timestamp, _ma_dateformat."%Z"),
._ma_locTZ),_ma_dateformat." %Z"),_ma_dateformat)." "._ma_locTZ
| eval _ma_locOffsetSec=strptime(_ma_locTime,_ma_dateformat)-strptime(_ma_utcTime,_ma_dateformat)
| eval _ma_locOffsetNot=round(_ma_locOffsetSec/3600,2), _ma_locOffsetNot;if(_ma_locOffsetSec <
0,_ma_locOffsetNot,"+"._ma_locOffsetNot)
| rex field=_ma_locOffsetNot "^(?<_ma_rexP>\+|-)(?<_ma_rexH>\d+)\.(?<_ma_rexM>\d+)$"
| eval _ma_rexM;if(_ma_rexM>0,60*(_ma_rexM/100),_ma_rexM)
| eval _ma_rexH;if(len(_ma_rexH)<2,"0"._ma_rexH,_ma_rexH)
| eval _ma_locOffsetNot=_ma_rexP._ma_rexH.":"._ma_rexM
| eval _ma_locTimeSPL=strftime(strptime(_ma_locTime,_ma_dateformat),"%Y-%m-%dT%H:%M:%S.%3Q"._ma_locOffsetNot)
| rename _ma_usrTime AS "$utc_timestamp$__timezoned_user" _ma_utcTime AS "$utc_timestamp$__timezoned_utc"
_ma_locTime AS "$utc_timestamp$__timezoned_region" _ma_locTimeSPL AS "$utc_timestamp$__timezoned_region_spl"
fields - ma *

```