



User Manual

Version : 16.06.14

Date : 14 June 2016

Copyright Riyosu Pty Ltd

Contents

Section 1. Product Synopsis	3
1.1 Summary	3
Section 2. Installation and Configuration.....	6
2.1 Pre-requisites.	6
2.2 Installation	6
2.3 How to us this document.....	6
Section 3. Analyse	7
3.1 Raw Trace Analysis.....	7
3.1.1 Load.....	7
3.1.2 Analyse.....	8
3.1.3 Raw Data	9
3.2 OSW(ps)	10
3.2.1 Load.....	10
3.2.2 Analyse	11
3.2.3 Raw Data	12
3.3 OSW(mpstat).....	13
3.3.1 Load.....	13
3.3.2 Analyse	14
3.4 OSW(netstat).....	15
3.4.1 Load.....	15
3.4.2 Analyse	16
3.4.3 Raw Data	17
Section 4. Configuration.....	18
4.1 Settings.....	18
4.2 Trace Rules	19
4.2.1 General Trace Statistics.....	19
4.2.2 Wait Event Analysis(Count and Time).....	20
4.2.3 Block Analysis(Count and Time).....	20
4.2.4 OSW(netstat).....	21
4.2.5 OSW(ps)	21

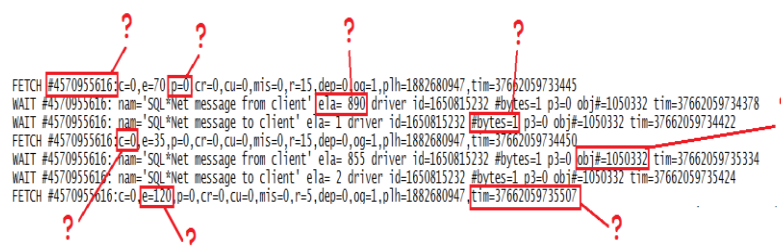
Section 1. Product Synopsis

1.1 Summary

Overview

SPOTO is an intuitive tool used to decipher and graphically present the content of various forms of Oracle trace files. SPOTO can easily demystify and plot all trace file information to a graphical timeline thus providing the DBA with superior insight into complex Oracle tracing.

What does it all mean?



```
FETCH #4570955616: c=0, e=70, p=0, cr=0, cu=0, mis=0, r=15, dep=0, oq=1, plh=1882680947, tim=37662059733445
WAIT #4570955616: nam='SQL*Net message from client' ela= 890 driver id=1650815232 #bytes=1 p3=0 obj#=1050332 tim=37662059734378
WAIT #4570955616: nam='SQL*Net message to client' ela= 1 driver id=1650815232 #bytes=1 p3=0 obj#=1050332 tim=37662059734422
FETCH #4570955616: c=0, e=35, p=0, cr=0, cu=0, mis=0, r=15, dep=0, oq=1, plh=1882680947, tim=37662059734450
WAIT #4570955616: nam='SQL*Net message from client' ela= 855 driver id=1650815232 #bytes=1 p3=0 obj#=1050332 tim=37662059735334
WAIT #4570955616: nam='SQL*Net message to client' ela= 2 driver id=1650815232 #bytes=1 p3=0 obj#=1050332 tim=37662059735424
FETCH #4570955616: c=0, e=120, p=0, cr=0, cu=0, mis=0, r=5, dep=0, oq=1, plh=1882680947, tim=37662059735507
```

Benefits

An Oracle DBA will spend a vast amount of time pouring over the contents of a trace file in attempt to decipher statistics for certain database operations. SPOTO aims to drastically reduce such analysis by extracting and translating all information and presenting it in a manner which is comprehensive and yet simple to understand.

SPOTO focuses on two key types of analysis:

Oracle SQL Trace Files:

This is probably the most important trace file to analyse when you want to determine the execution statistics for any given SQL. However the instrumentation on today's market for performing such a fundamental analysis does not quite cut it. SPOTO aims to provide the Oracle DBA with an insight into all valuable trace file information in one simple glance.

OSW Trace Files (Oracle System Watcher)

OSW trace files provide statistical information for the operating system on which an Oracle database resides. OSW generates important system information which is not captured by standard monitoring tools such as OEM. The content of an OSW trace file is superb. However trying to decipher this information can be quite the challenge. Once again SPOTO rescues the DBA from such a pain stacking task and presents the OSW information in a series of easy to read graphs.

Functional Details

The most common trace file analysed by the Oracle DBA would be the usual SQL query trace file. Granted Oracle's good old trusty TKPROF tool can provide a nice summary of the trace file content but there is just so much more valuable information at play.

Consider the old school method of TKPROF\trace file analysis below:

call	count	cpu	elapsed	disk	query	current	rows
Parse	1	0.00	0.00	0	0	0	0
Execute	1	0.01	0.00	0	0	0	0
Fetch	9	2.23	17.62	6063	6300	0	111
total	11	2.24	17.62	6063	6300	0	111

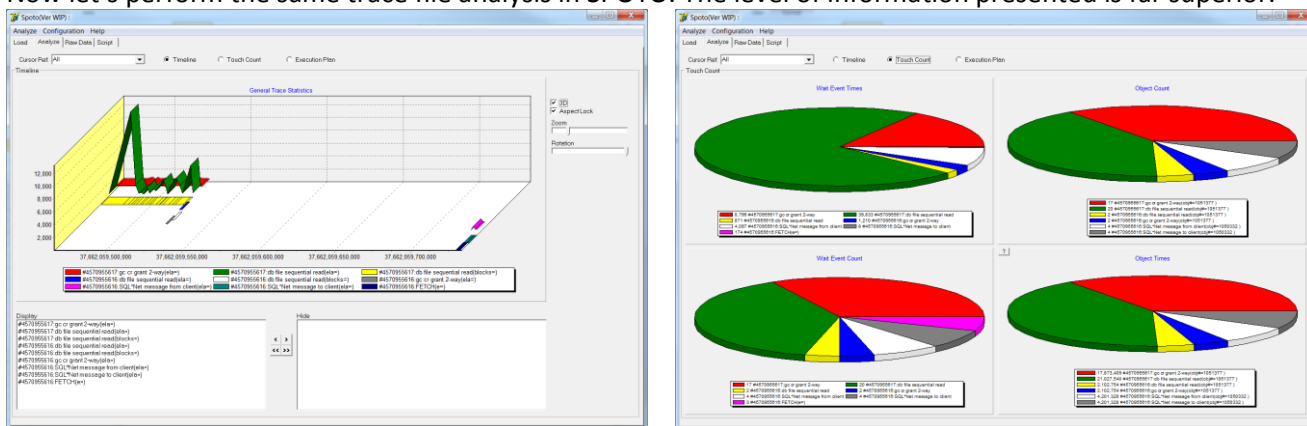
Misses in library cache during parse: 0
Optimizer mode: ALL_ROWS
Parsing user id: 81
Number of plan statistics captured: 1

Rows (1st)	Rows (avg)	Rows (max)	Row Source Operation
111	111	111	HASH GROUP BY (cr=6300 pr=6063 pw=0 time=1761894 us cost=3481 size=10896 card=227)
317856	317856	317856	NESTED LOOPS (cr=6300 pr=6063 pw=0 time=2300206 us cost=3479 size=1194000 card=24875)
109	109	109	INDEX RANGE SCAN PK_PC_IN8 (cr=6 pr=6 pw=0 time=783 us cost=6 size=1976 card=76)(object id 1050332)
317856	317856	317856	PARTITION HASH SINGLE PARTITION: KEY(AP) KEY(AP) (cr=6294 pr=6057 pw=0 time=2508039 us cost=46 size=7194 card=327)
317856	317856	317856	INDEX RANGE SCAN IDXPT_11: KEY(AP) KEY(AP) (cr=6294 pr=6057 pw=0 time=2403690 us cost=46 size=7194 card=327)(object id 1051376)

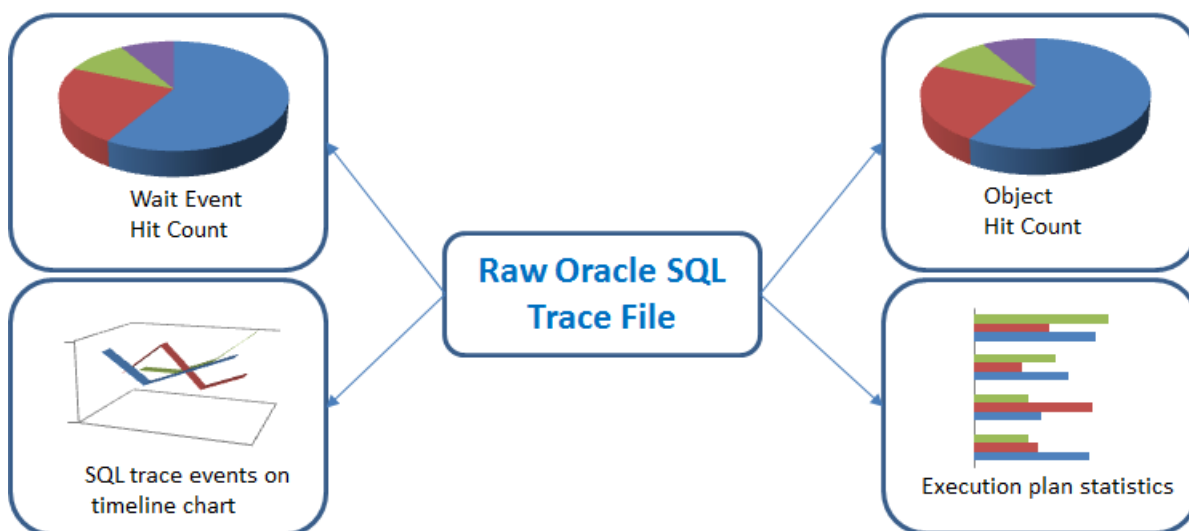
Elapsed times include waiting on following events:

Event waited on	Times waited	Max. Wait	Total Waited
SQL*Net message to client	9	0.00	0.00
Disk File operations I/O	51	0.00	0.00
gc cr grant 2-way	3965	0.01	2.30
db file sequential read	6063	0.03	12.59
gc cr grant congested	7	0.00	0.00
SQL*Net message from client	9	0.00	0.00

Now let's perform the same trace file analysis in SPOTO. The level of information presented is far superior:



SPOTO will analyse the content of an Oracle SQL trace and break it down into many aspects. By plotting the trace file information to various graphs we tap into a more intuitive and comprehensive manner of trace file analysis.



Apart from Oracle SQL trace file analysis SPOTO also has functionality to decipher Oracle System Watcher trace files.

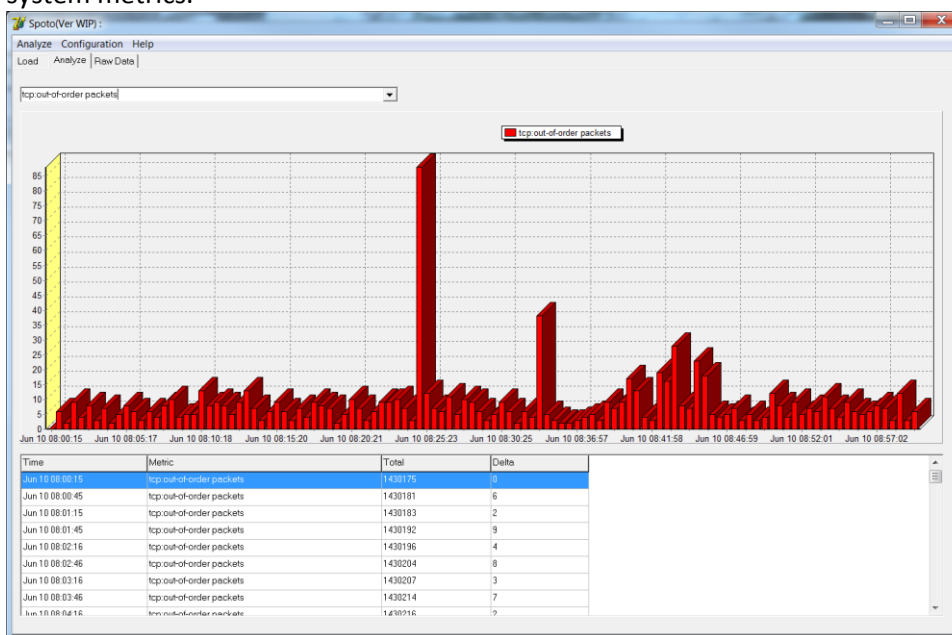
The typical OSW trace file shown below has a labyrinth of information collected over frequent snapshots. Any attempt to try and transform this information into a legible dataset is going to take a considerable amount of time.

```

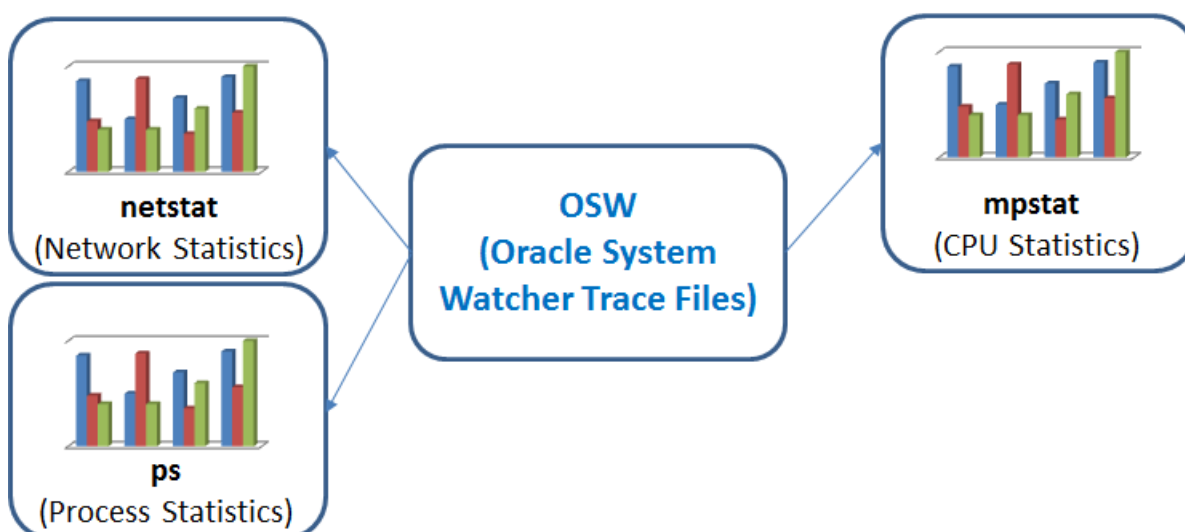
icmp:      160538 calls to icmp_error|
0 errors not generated because old message was icmp
Output histogram:
  echo reply: 149720984
  destination unreachable: 160538
  echo: 75001846
  time stamp reply: 66
  information request reply: 66
0 messages with bad code fields
0 messages < minimum length
0 bad checksums
0 messages with bad length
Input histogram:
  echo reply: 75000800
  destination unreachable: 135074
  echo: 149720984
  time stamp: 66
  information request: 66
  address mask request: 66
149721116 message responses generated

igmp:      0 messages received
0 messages received with too few bytes
0 messages received with bad checksum
0 membership queries received
0 membership queries received with invalid field(s)
0 membership reports received
0 membership reports received with invalid field(s)
0 membership reports received for groups to which we belong
58 membership reports sent
  
```

Now let's run the OSW trace file through SPOTO. Instantly we have a valuable perspective on operating system metrics.



By plotting the OSW trace file information to graphs we instantly open up the value of OSW.



Section 2. Installation and Configuration

2.1 Pre-requisites.

SPOTO does not connect to an Oracle database so no SQL drivers are required. There are no pre-requisites to installing SPOTO.

2.2 Installation

- Copy all SPOTO files into a directory on a local hard drive. For example all files should be placed in c:\spoto
- Edit the db.ini file and ensure the **"METADATA"** parameter equals the full pathname of where the SPOTO files have been placed.

Example:

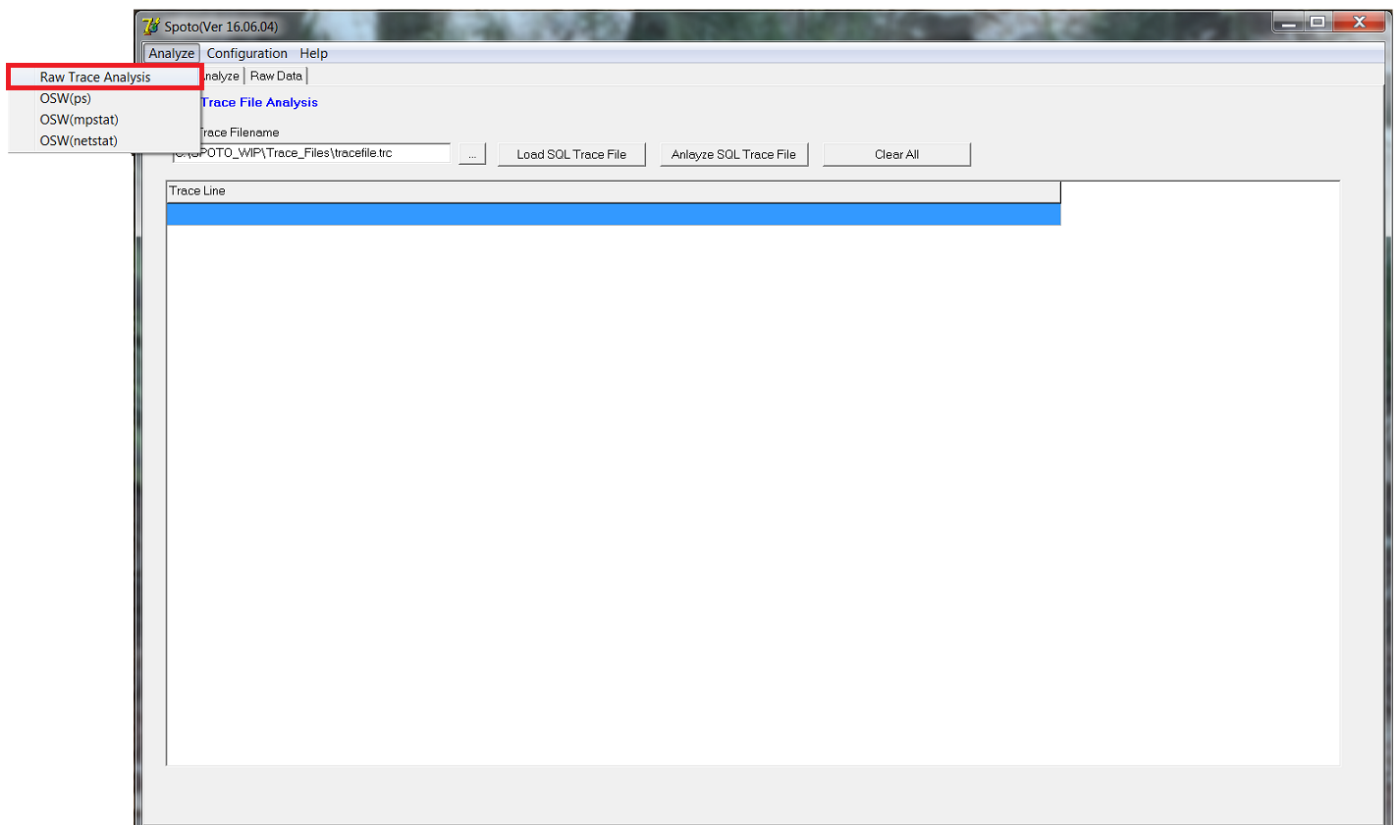
```
[Main]
SPOTODB=ORACLE
METADATA=C:\SPOTO
```

2.3 How to us this document

Each section of this document refers to a specific piece of functionality in SPOTO.

Example:

Section "3.1 Raw Trace Analysis" of this document refers to the highlighted menu item below:



Section 3. Analyse

3.1 Raw Trace Analysis

Raw Oracle SQL trace files can be loaded into SPOTO memory and deciphered into a meaningful set of graphs.

3.1.1 Load

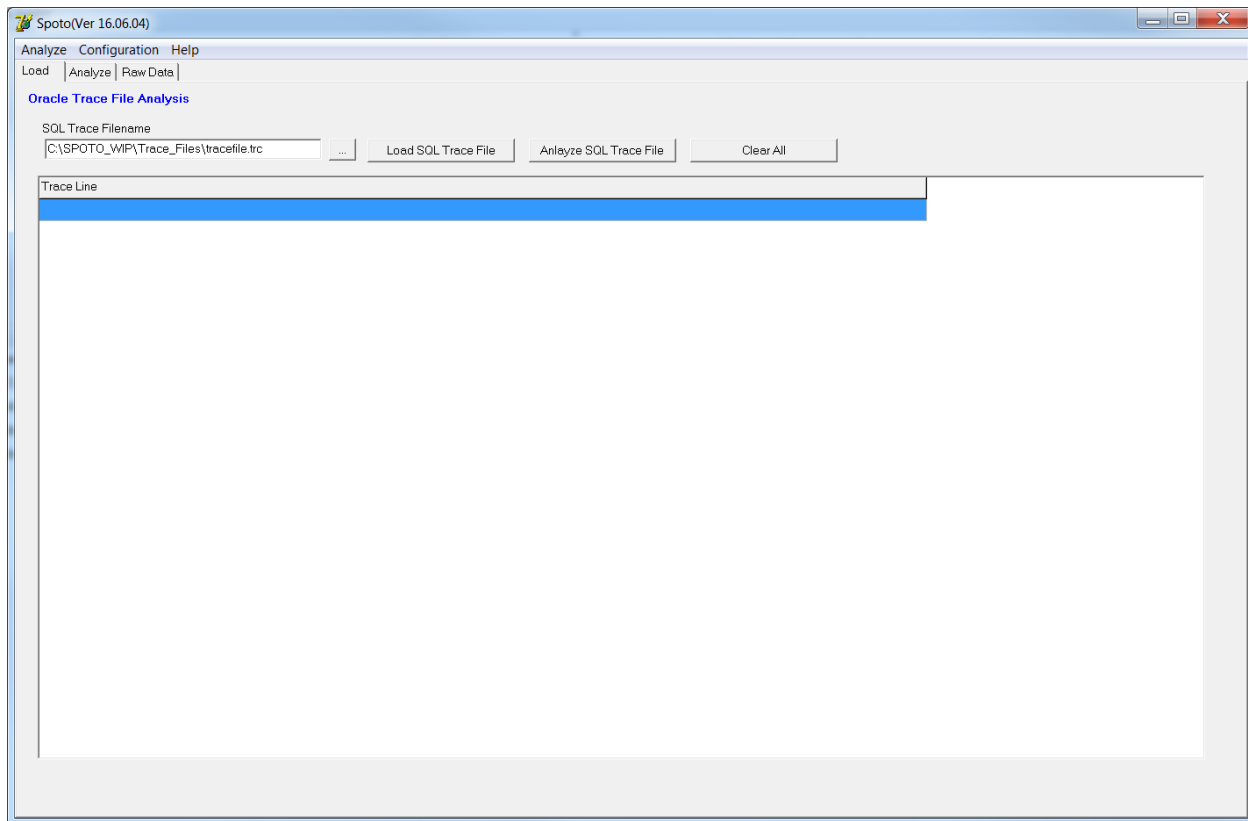


Illustration 3.1.1: Oracle SQL trace file load

Context:

Component ID	Component Type	Description
SQL Trace Filename	Field	Filename of the trace file to load.
LOAD SQL Trace File	Button	Loads the content of the trace file.
Analyse SQL Trace file	Button	Analyses the content of the trace file and plots to a series of graphs.
Clear All	Button	Clears the contents of all grids and graphs.

Instructions, Load and analyse trace File:

Use this method to load the content of an Oracle SQL trace file.

- 1) Enter the filename of the Oracle SQL trace file to load.
- 2) Click the <Load SQL Trace File> button.
- 3) Click the <Analyse SQL Trace File> button.

3.1.2 Analyse

Once the analyse button has been clicked various graphs will be populated with trace file information as per below.

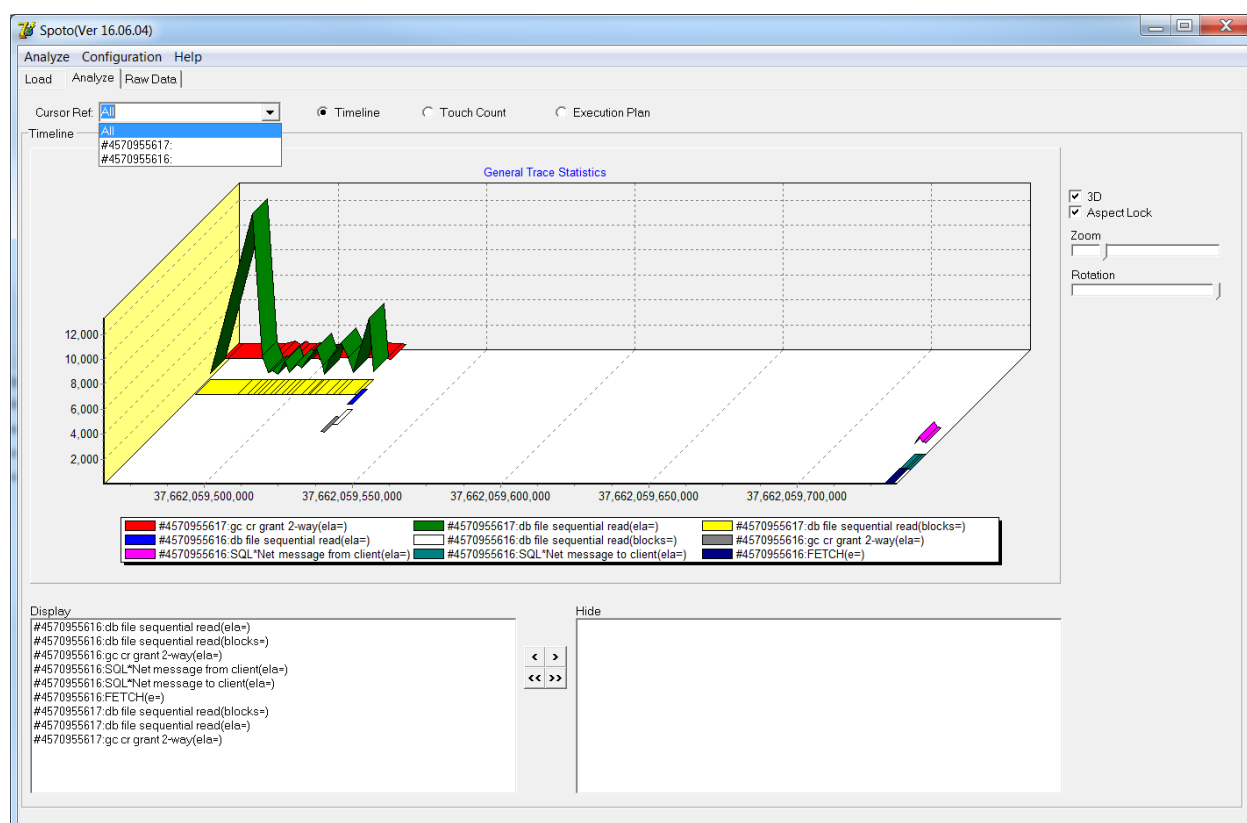


Illustration 3.1.2: Oracle SQL trace analysis

Context:

Component ID	Component Type	Description
Cursor Ref:	Pull down	Trace file information is grouped into cursors. Each different piece of SQL is allocated a distinct cursor reference number. The cursor ref pulldown allows you to view single cursor information of all cursor information on the graph.
TimeLine	Radio Button	Select this radio button to display trace file events on the timeline graph.
Touch Count	Radio Button	Select this radio button to display a summary of trace file event grouped by wait events and objects.
Execution Plan	Radio Button	Select this radio button to display information about an SQL execution plan.
3D	Checkbox	Check to display the graph in 3D mode.
Aspect Lock	Checkbox	Lock the graph into its default aspect position. Disables the zoom and rotation slide bars.
Zoom	Slide Bar	Zoom in\out on the graph.
Rotation	Slide Bar	Rotate the viewing angle of the graph
Display	List Box	List of trace file events to display on the graph.
Hide	List Box	List of trace file events to hide on the graph.

3.1.3 Raw Data

Displays formatted trace file data for the following information:

- General Trace Statistics(Wait Events)
- General Trace Statistics(Fetch Events)
- Trace Summary(Totals of time and hit counts for each wait event)
- Object summary(Totals of time and hit counts for each object)
- Execution plan statistics

Spoto(Ver 16.06.04)										
Analyze Configuration Help										
Load Analyze Raw Data										
General Trace Statistics(Wait)										
Event	Metric Value	Metric Type	Time							
#4570961160:SQL*Net message to client	2	ela+	37662042095852							
#4570961160:SQL*Net message from client	700	ela+	37662042098562							
#4570955616:SQL*Net message to client	5	ela+	37662042106933							
#4570955616:Disk file operations I/O	9	ela+	37662042107050							
#4570955616:gc cr grant 2-way	1057	ela+	37662042108341							
#4570955616:Disk file operations I/O	333	ela+	37662042108691							
#4570955616:db file sequential read	1372	ela+	37662042110374							
#4570955616:db file sequential read	1	block+	37662042110374							
General Trace Statistics(Fetch)										
Event	Metric Value	Time								
#4570955616:FETCH(e=)	17619791	37662059726751								
#4570955616:FETCH(e=)	75	37662059727529								
#4570955616:FETCH(e=)	47	37662059728693								
#4570955616:FETCH(e=)	61	37662059729709								
#4570955616:FETCH(e=)	52	37662059731257								
#4570955616:FETCH(e=)	61	37662059732374								
#4570955616:FETCH(e=)	70	37662059733445								
#4570955616:FETCH(e=)	26	37662059734450								
Trace Summary										
Event	Time	Count								
#4570961160:SQL*Net message to client	2	1								
#4570961160:SQL*Net message from client	700	1								
#4570955616:SQL*Net message to client	24	9								
#4570955616:Disk file operations I/O	4726	51								
#4570955616:gc cr grant 2-way	2309234	3965								
#4570955616:db file sequential read	12590050	6063								
#4570955616:gc cr grant congested	3966	7								
#4570955616:FETCH(e=)	17620312	9								
Object Summary										
Event	Time	Count								
#4570961160:SQL*Net message to client(obj#=13755)	13755	1								
#4570961160:SQL*Net message from client(obj#=13755)	13755	1								
#4570955616:SQL*Net message to client(obj#=13755)	13755	1								
#4570955616:Disk file operations I/O(obj#=13755)	13755	1								
#4570955616:gc cr grant 2-way(obj#=1050332)	6301992	6								
#4570955616:Disk file operations I/O(obj#=1050332)	4201328	4								
#4570955616:db file sequential read(obj#=1050332)	6301992	6								
#4570955616:Disk file operations I/O(obj#=1051377)	48363342	46								
Execution Plan										
Cursor#	Count	Obj#	Operation	Log Reads	Phy Reads	Phy Writes	Time(us)	Cost	Size(Bytes)	Cardinality
#4570955616:	111	0	HASH GROUP BY	6300	6063	0	17619694	3481	10896	227
#4570955616:	317856	0	NESTED LOOPS	6300	6063	0	2300206	3479	1194000	24875
#4570955616:	109	1050332	INDEX RANGE SCAN PK_PC_INFO3	6	6	0	7283	6	1976	76
#4570955616:	317856	0	PARTITION HASH SINGLE PARTITION: KEY(AP) KE 6294	6057	0	0	2508039	46	7194	327
#4570955616:	317856	1051376	INDEX RANGE SCAN IDXPT_CURVE_INST_DT PAR 6294	6057	0	0	2403690	46	7194	327

Illustration 3.1.3: Oracle SQL trace formatted data

3.2 OSW(ps)

OSW trace files can be loaded into SPOTO memory and deciphered into a meaningful set of graphs. OSW files generated via the “ps” command provide information for specific processes.

3.2.1 Load

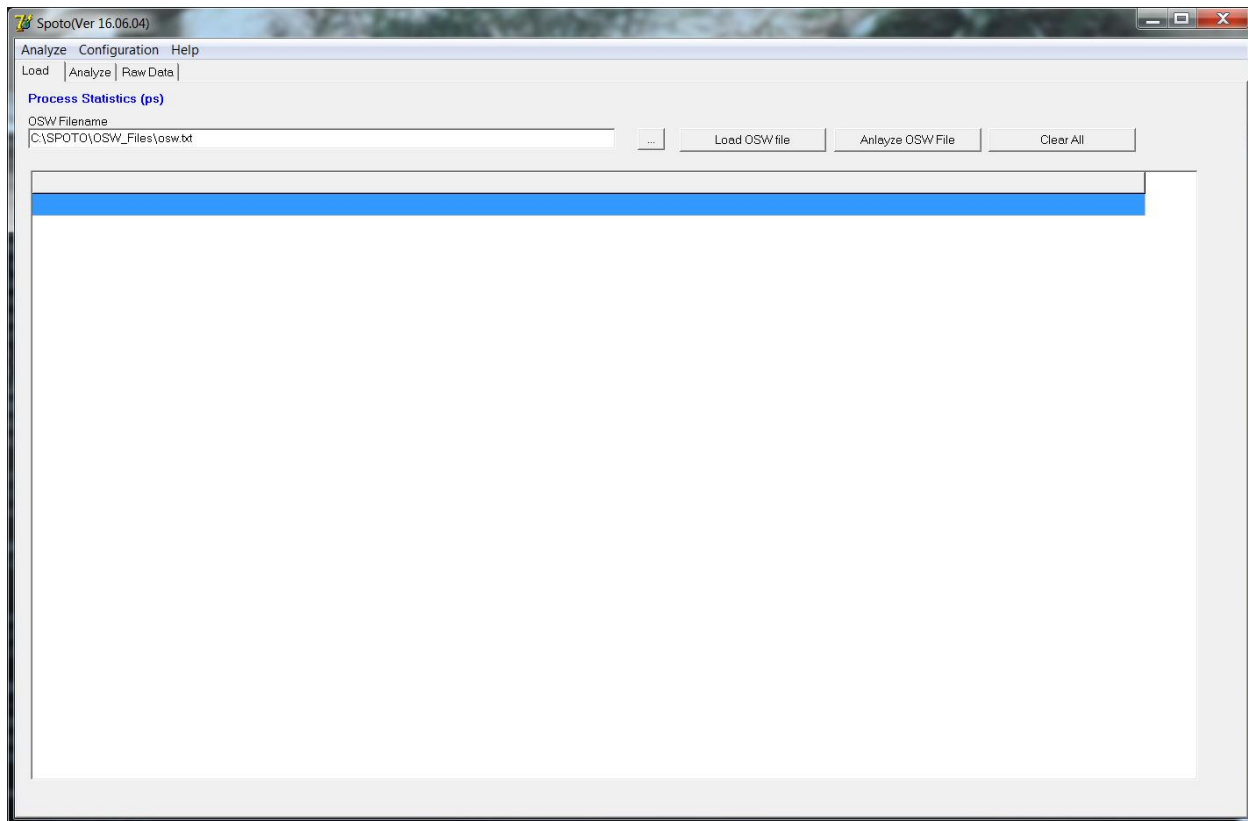


Illustration 3.2.1: OSW(ps) trace file load

Context:

Component ID	Component Type	Description
OSW Filename	Field	Filename of the trace file to load.
LOAD OSW Trace File	Button	Loads the content of the trace file.
Analyse OSW Trace file	Button	Analyses the content of the trace file and plot to graph.
Clear All	Button	Clears the contents of all grids and graphs.

Instructions, Load and analyse trace File:

Use this method to load the content of an OSW trace file.

- 1) Enter the filename of the OSW trace file to load.
- 2) Click the <Load OSW File> button.
- 3) Click the <Analyse OSW File> button.

3.2.2 Analyse

Once the analyse button has been clicked various graphs will be populated with trace file information as per below.

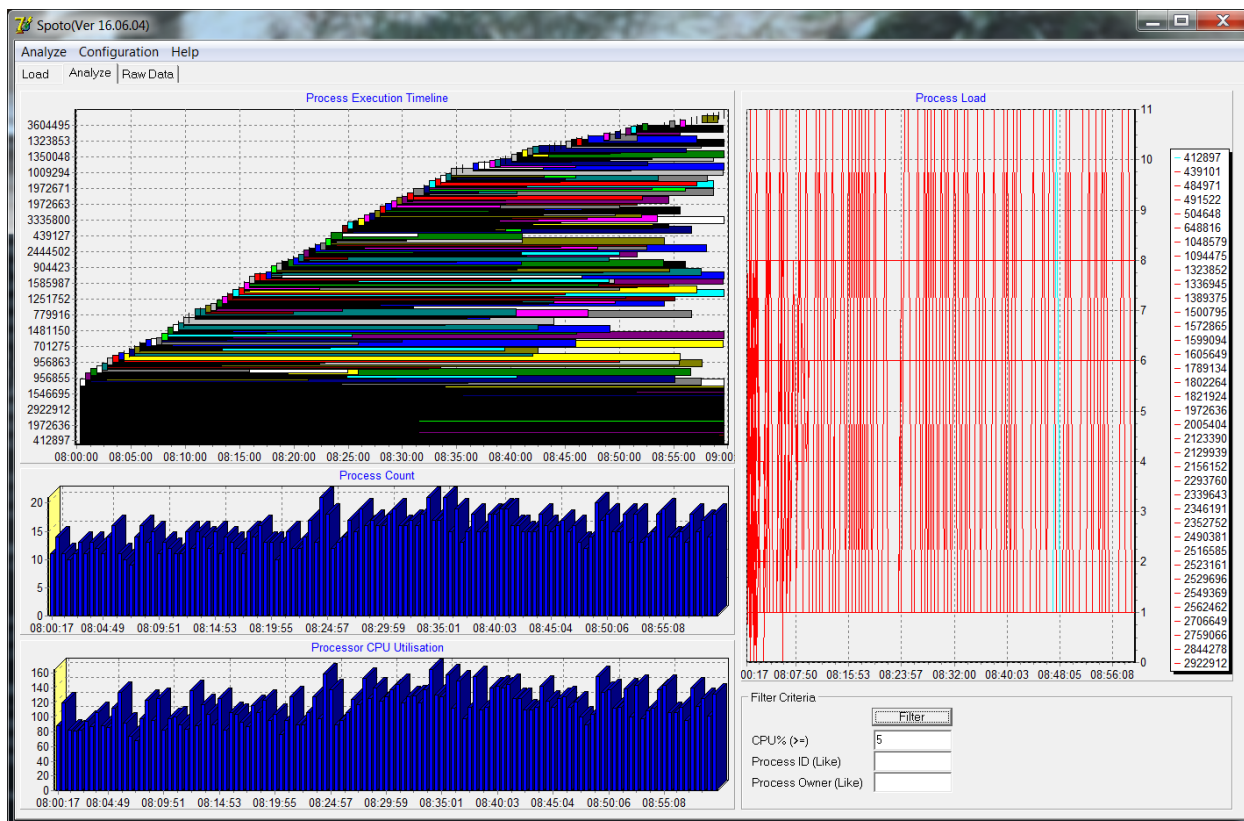


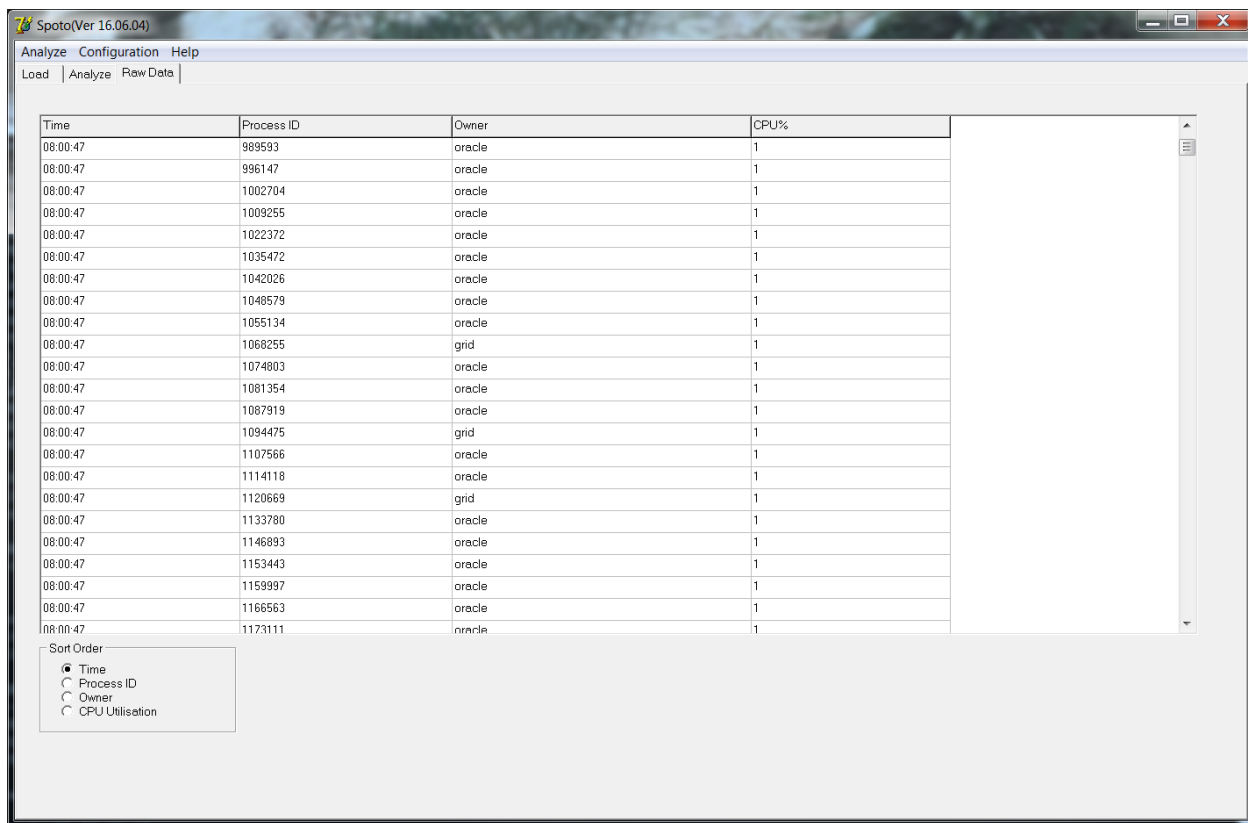
Illustration 3.2.2: OSW(ps) trace file analysis

Context:

Component ID	Component Type	Description
Filter	Button	Filter process graphs based on filter criteria.
CPU%(>=)	Field	Only show information for processes that have ever exceeded a certain amount of CPU consumption (Default is 5%).
Process ID (Like)	Field	Only show information for processes whose process ID's contain this value.
Process Owner (Like)	Field	Only show information for processes whose process owner contains this value.
Process Execution Timeline	Graph	Show all processes that match the filter criteria.
Process Count	Graph	Show number of processes that exceed the CPU% filter for a given point in time.
Process CPU Utilisation	Graph	Show total CPU utilisation for all processes that exceed the CPU% filter for a given point in time.

3.2.3 Raw Data

Displays formatted trace file data for the following information:



Time	Process ID	Owner	CPU%
08:00:47	989533	oracle	1
08:00:47	996147	oracle	1
08:00:47	1002704	oracle	1
08:00:47	1009255	oracle	1
08:00:47	1022372	oracle	1
08:00:47	1035472	oracle	1
08:00:47	1042026	oracle	1
08:00:47	1048579	oracle	1
08:00:47	1055134	oracle	1
08:00:47	1068255	grid	1
08:00:47	1074803	oracle	1
08:00:47	1081354	oracle	1
08:00:47	1087919	oracle	1
08:00:47	1094475	grid	1
08:00:47	1107566	oracle	1
08:00:47	1114118	oracle	1
08:00:47	1120669	grid	1
08:00:47	1133780	oracle	1
08:00:47	1146893	oracle	1
08:00:47	1153443	oracle	1
08:00:47	1159997	oracle	1
08:00:47	1166563	oracle	1
08:00:47	1173111	oracle	1

Sort Order

- ☒ Time
- ☐ Process ID
- ☐ Owner
- ☐ CPU Utilisation

Illustration 3.2.3: OSW(ps) formatted data

Context:

Component ID	Component Type	Description
Time	Radio Button	Sort raw data grid by Time.
Process ID	Radio Button	Sort raw data grid by process ID
Owner	Radio Button	Sort raw data grid by process owner.
CPU Utilisation	Radio Button	Sort raw data grid by CPU utilisation.

3.3 OSW(mpstat)

OSW trace files can be loaded into SPOTO memory and deciphered into a meaningful set of graphs. OSW files generated via the “mpstat” command provide information for overall CPU consumption.

3.3.1 Load

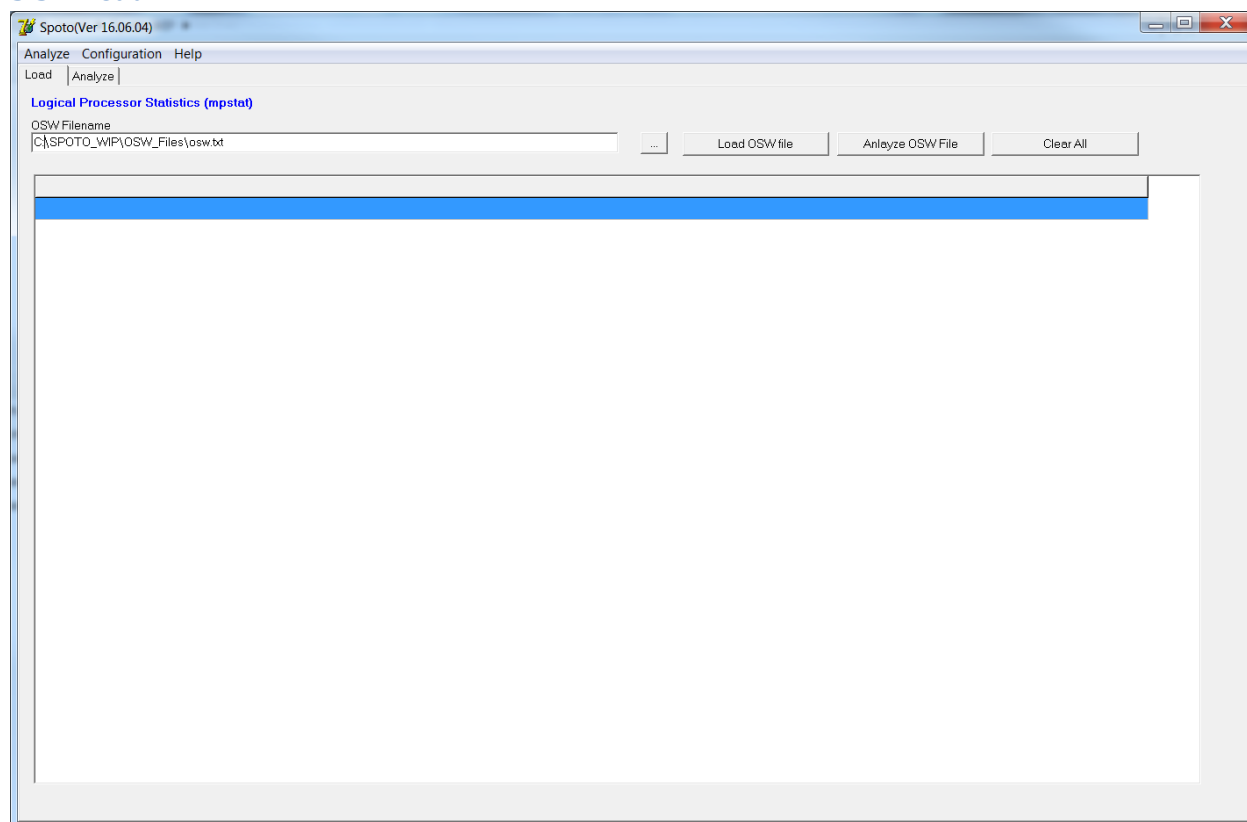


Illustration 3.3.1: OSW(mpstat) trace file load

Context:

Component ID	Component Type	Description
OSW Filename	Field	Filename of the trace file to load.
LOAD OSW Trace File	Button	Loads the content of the trace file.
Analyse OSW Trace file	Button	Analyses the content of the trace file and plot to graph.
Clear All	Button	Clears the contents of all grids and graphs.

Instructions, Load and analyse trace File:

Use this method to load the content of an OSW trace file.

- 1) Enter the filename of the OSW trace file to load.
- 2) Click the <Load OSW File> button.
- 3) Click the <Analyse OSW File> button.

3.3.2 Analyse

Once the analyse button has been clicked the graph will be populated with trace file information as per below.

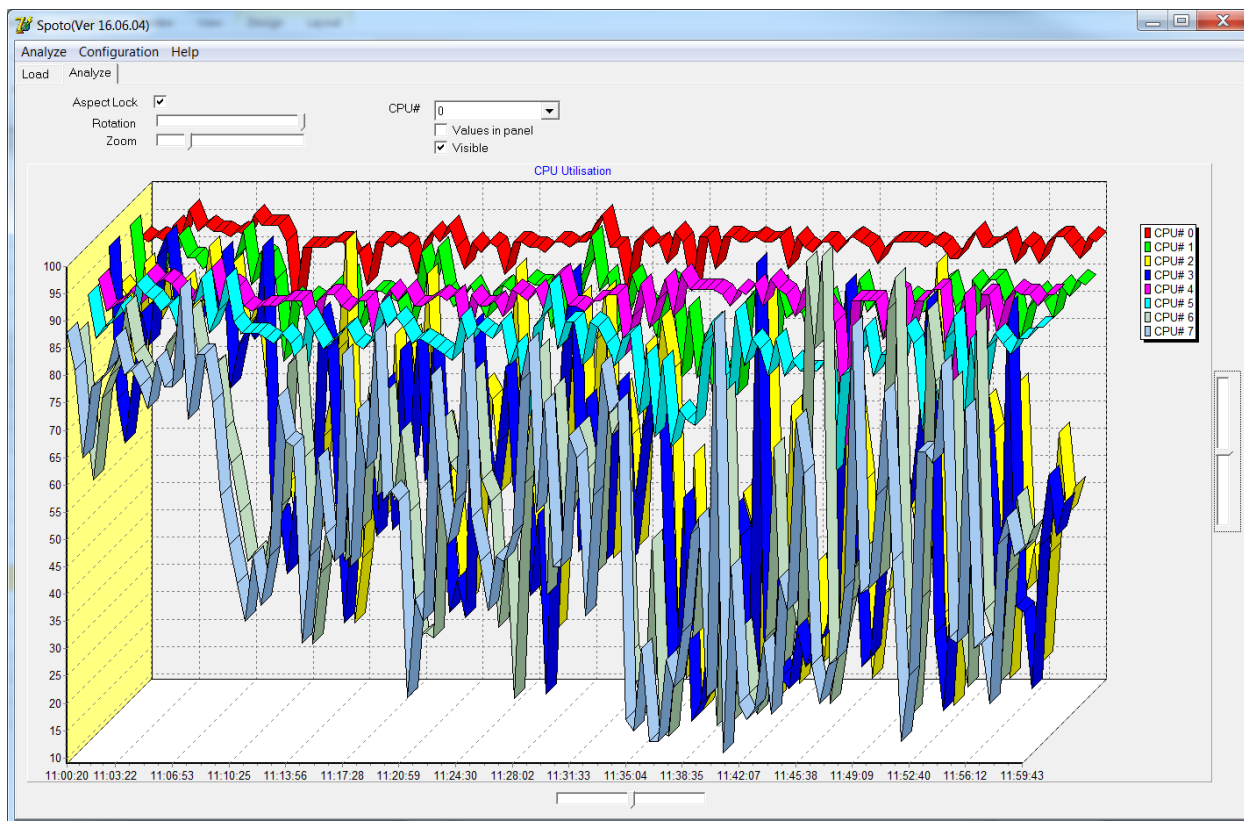


Illustration 3.3.2: OSW(mpstat) trace file analysis

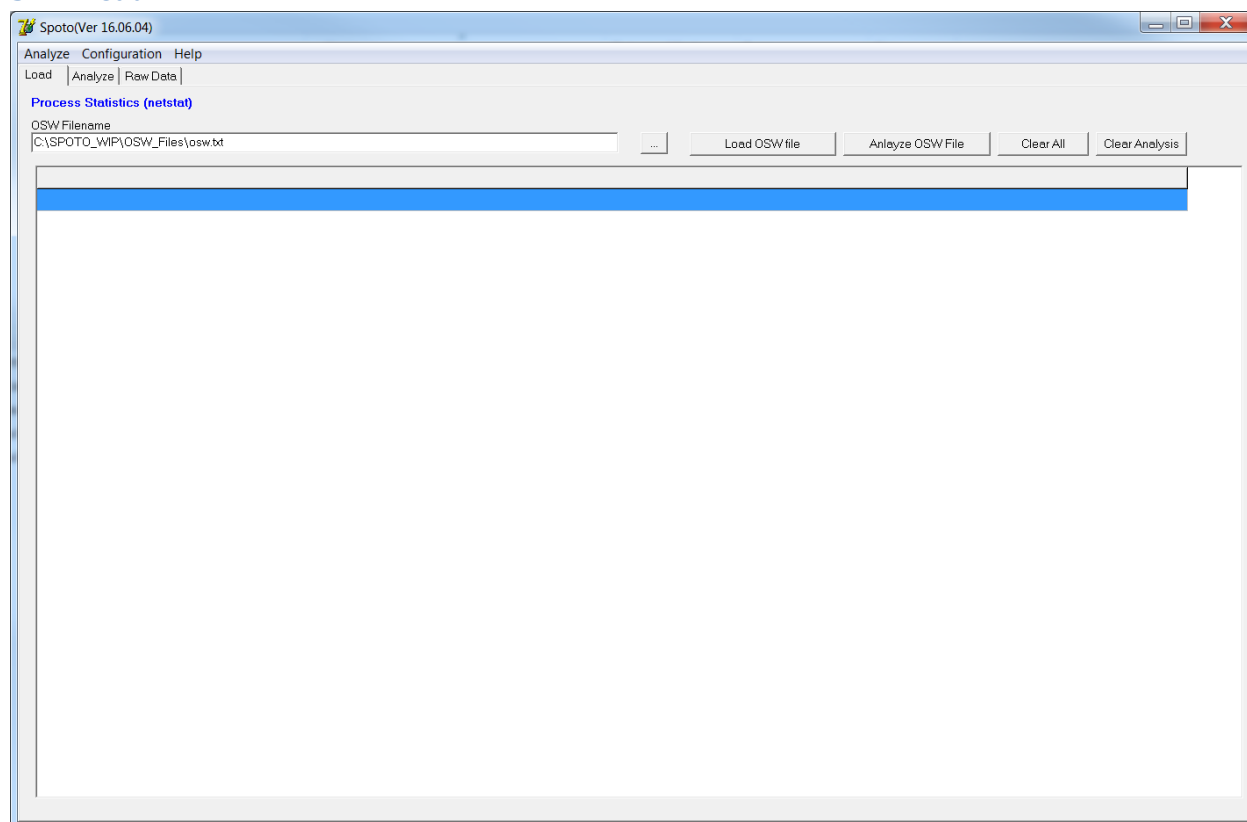
Context:

Component ID	Component Type	Description
Aspect Lock	Checkbox	Lock the graph into its default aspect position. Disables the zoom and rotation slide bars.
Rotation	Slide Bar	Rotate the viewing angle of the graph.
Zoom	Slide Bar	Zoom in\out on the graph.
CPU#	Pull Down	Select the CPU# for whose display attributes to change .
Values in panel	Checkbox	Display CPU# values in graph.
Visible	Checkbox	Check to display CPU# details in graph.

3.4 OSW(netstat)

OSW trace files can be loaded into SPOTO memory and deciphered into a meaningful set of graphs. OSW files generated via the “netstat” command provide information for network statistics.

3.4.1 Load



Context:

Component ID	Component Type	Description
OSW Filename	Field	Filename of the trace file to load.
LOAD OSW Trace File	Button	Loads the content of the trace file.
Analyse OSW Trace file	Button	Analyses the content of the trace file and plot to graph.
Clear All	Button	Clears the contents of all grids and graphs.
Cleat Analysis	Button	Clear only the graphs and raw data.

Instructions, Load and analyse trace File:

Use this method to load the content of an OSW trace file.

- 1) Enter the filename of the OSW trace file to load.
- 2) Click the <Load OSW File> button.
- 3) Click the <Analyse OSW File> button.

3.4.2 Analyse

Once the analyse button has been clicked the graph will be populated with trace file information as per below.

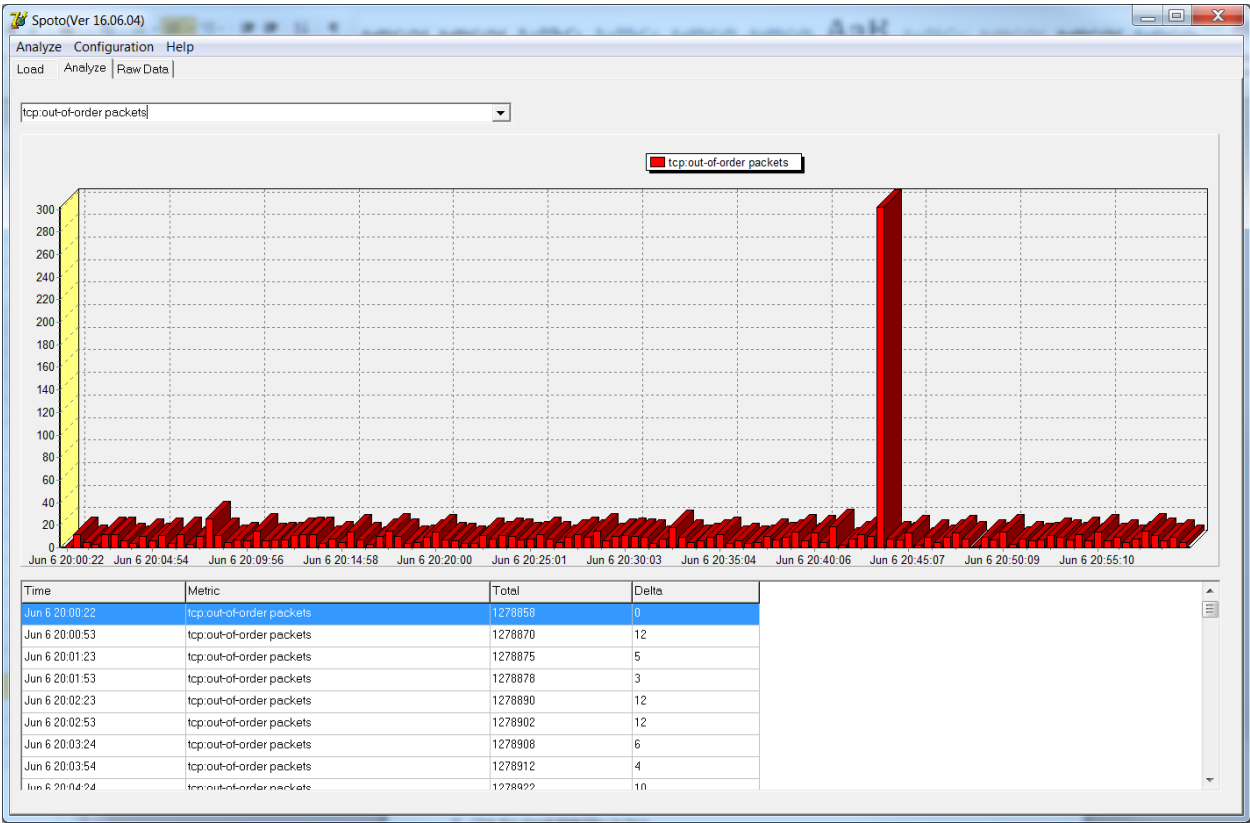


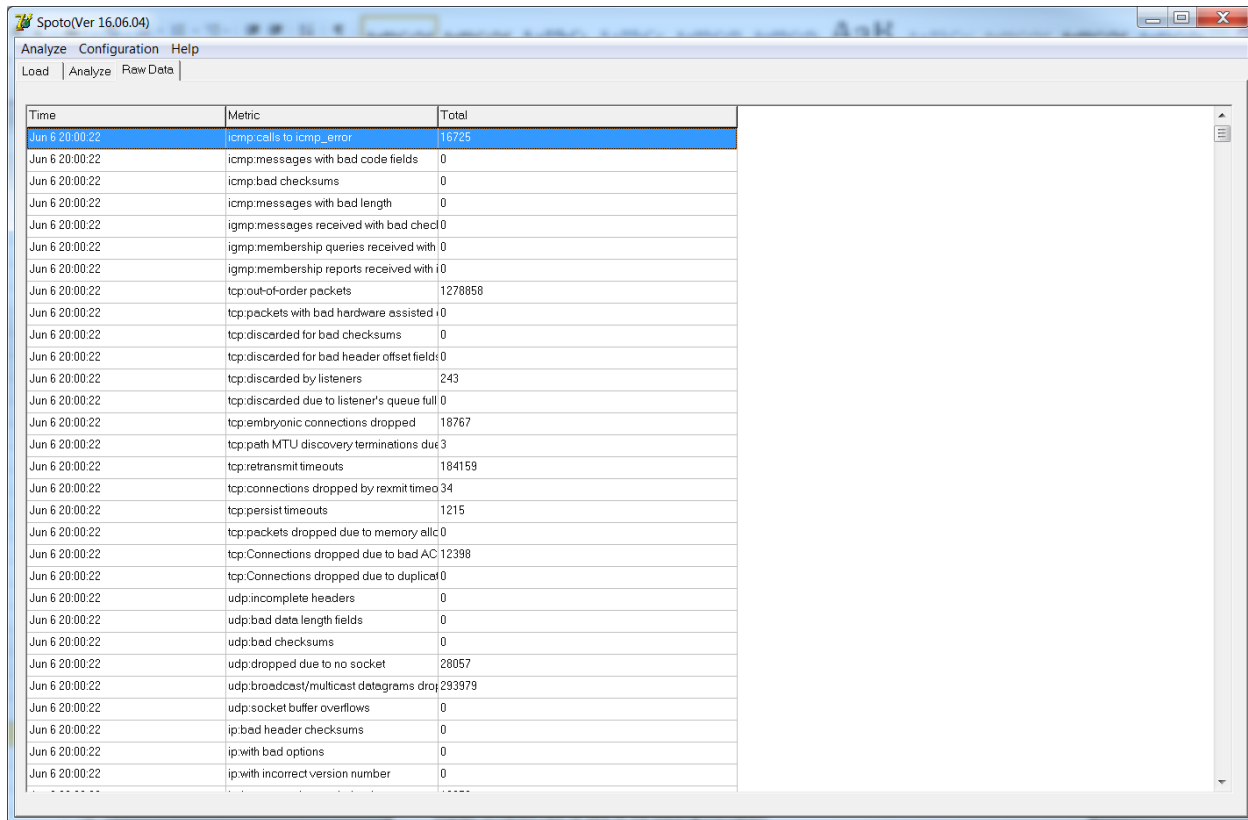
Illustration 3.4.2: OSW(netstat) trace file analysis

Context:

Component ID	Component Type	Description
Network Metric	Pull Down	Select the network metric to show on the graph.

3.4.3 Raw Data

Displays formatted trace file data for all network information.



Time	Metric	Total
Jun 6 20:00:22	icmp:calls to icmp_error	16725
Jun 6 20:00:22	icmp:messages with bad code fields	0
Jun 6 20:00:22	icmp:bad checksums	0
Jun 6 20:00:22	icmp:messages with bad length	0
Jun 6 20:00:22	igmp:messages received with bad check	0
Jun 6 20:00:22	igmp:membership queries received with	0
Jun 6 20:00:22	igmp:membership reports received with	0
Jun 6 20:00:22	tcp:out-of-order packets	1278858
Jun 6 20:00:22	tcp:packets with bad hardware assisted	0
Jun 6 20:00:22	tcp:discarded for bad checksums	0
Jun 6 20:00:22	tcp:discarded for bad header offset field	0
Jun 6 20:00:22	tcp:discarded by listeners	243
Jun 6 20:00:22	tcp:discarded due to listener's queue full	0
Jun 6 20:00:22	tcp:embryonic connections dropped	18767
Jun 6 20:00:22	tcp:path MTU discovery terminations due	3
Jun 6 20:00:22	tcp:retransmit timeouts	184159
Jun 6 20:00:22	tcp:connections dropped by rexmit timeo	34
Jun 6 20:00:22	tcp:persist timeouts	1215
Jun 6 20:00:22	tcp:packets dropped due to memory alloc	0
Jun 6 20:00:22	tcp:Connections dropped due to bad ACK	12398
Jun 6 20:00:22	tcp:Connections dropped due to duplicate	0
Jun 6 20:00:22	udp:incomplete headers	0
Jun 6 20:00:22	udp:bad data length fields	0
Jun 6 20:00:22	udp:bad checksums	0
Jun 6 20:00:22	udp:dropped due to no socket	28057
Jun 6 20:00:22	udp:broadcast/multicast datagrams drop	293979
Jun 6 20:00:22	udp:socket buffer overflows	0
Jun 6 20:00:22	ip:bad header checksums	0
Jun 6 20:00:22	ip:with bad options	0
Jun 6 20:00:22	ip:with incorrect version number	0

Illustration 3.4.3: OSW(netstat) trace file analysis

Section 4. Configuration

4.1 Settings

Default directory to locate specific types of trace files.

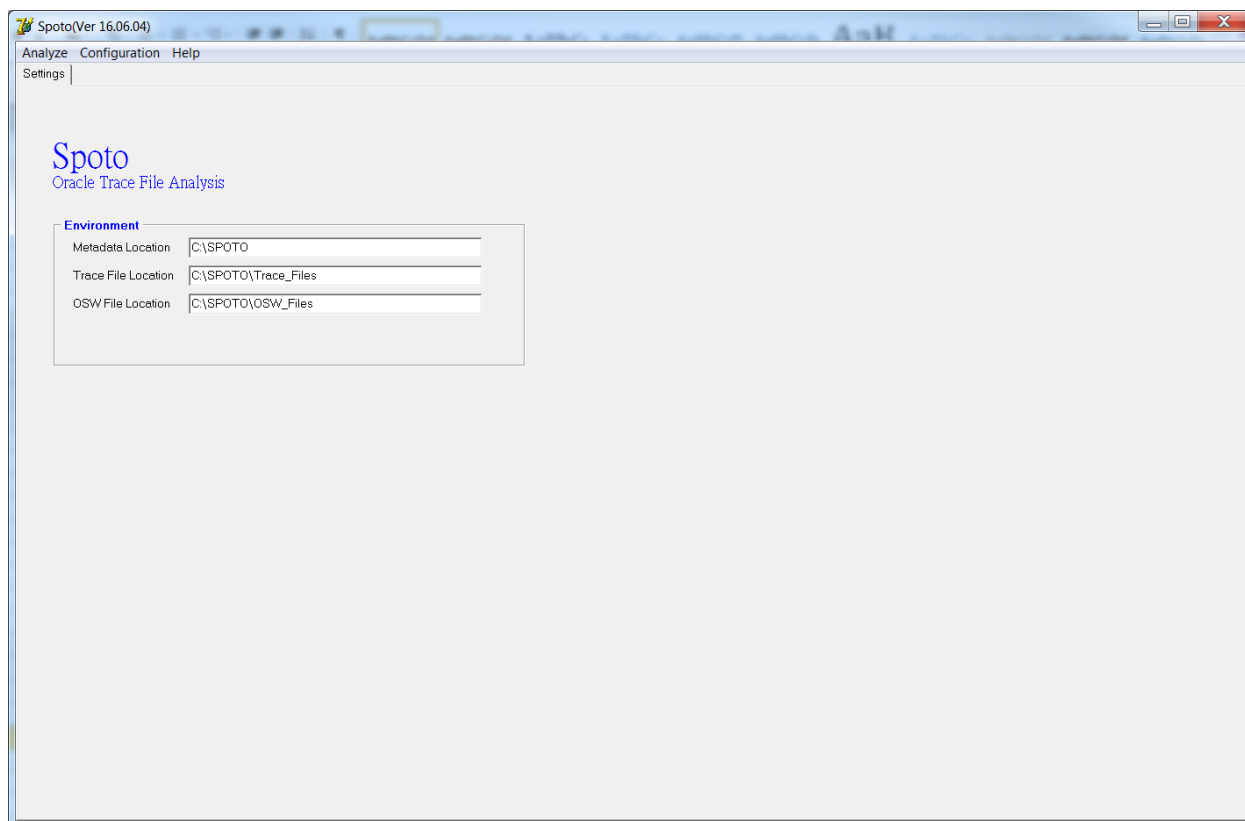


Illustration 4.1: Configuration Settings

4.2 Trace Rules

Configures the method in which trace files are analysed.

****Whilst it is not recommend the trace rule files can be altered by modifying the values in the tracerules.ini file located in the root level directory of SPOTO.**

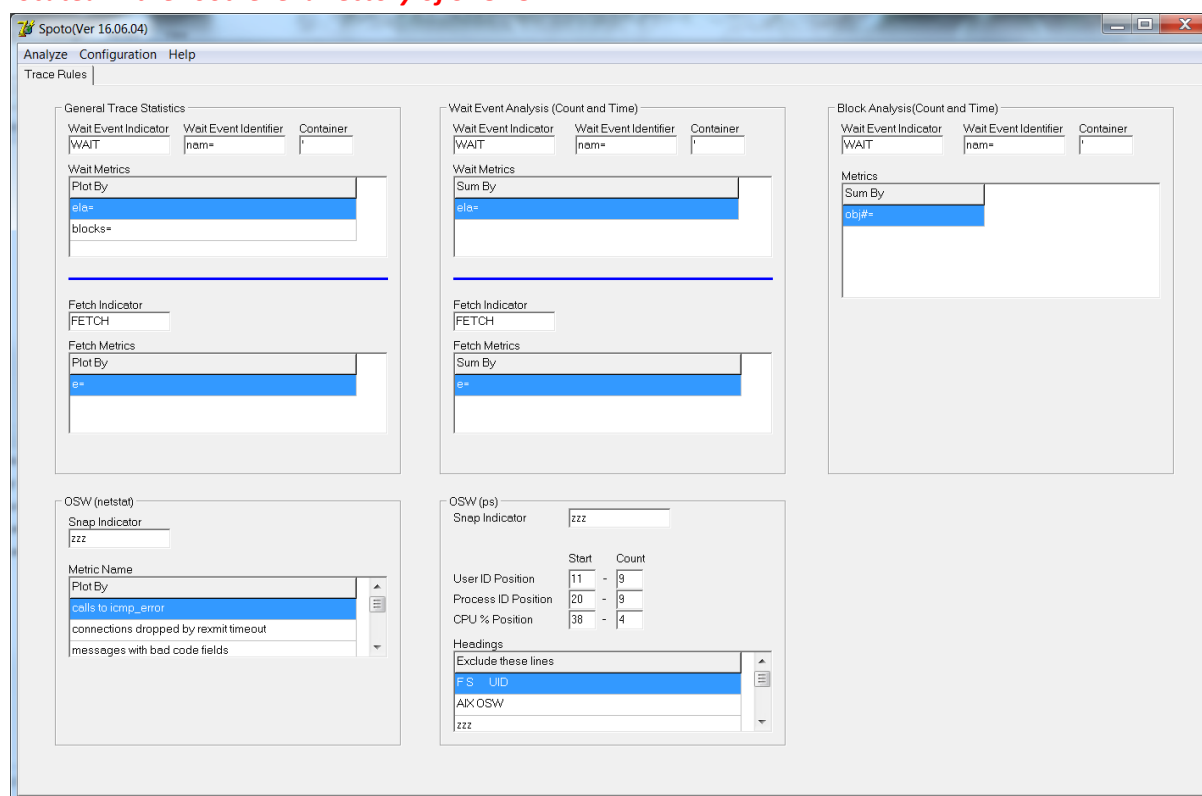


Illustration 4.2: Trace Rules

4.2.1 General Trace Statistics

The lines in a trace file are deciphered based on a set of preconfigured trace rules as shown below.

WAIT EVENT INDICATOR	Look for this sub-string at the start of the trace line. Identifies a WAIT event.
WAIT EVENT IDENTIFIER	This sub-string designates how the wait event type is prefixed.
CONTAINER:	The wait event name can be found in a container marked with these values.
WAIT METRICS	Looks for this string in the trace line. The value applicable to this metric is the value which will be plotted in the graphs.
FETCH INDICATOR	Look for this sub-string at the start of the trace line. Identifies a FETCH event.
FETCH METRICS	Looks for this string in the trace line. The value applicable to this metric is the value which will be plotted in the graphs.

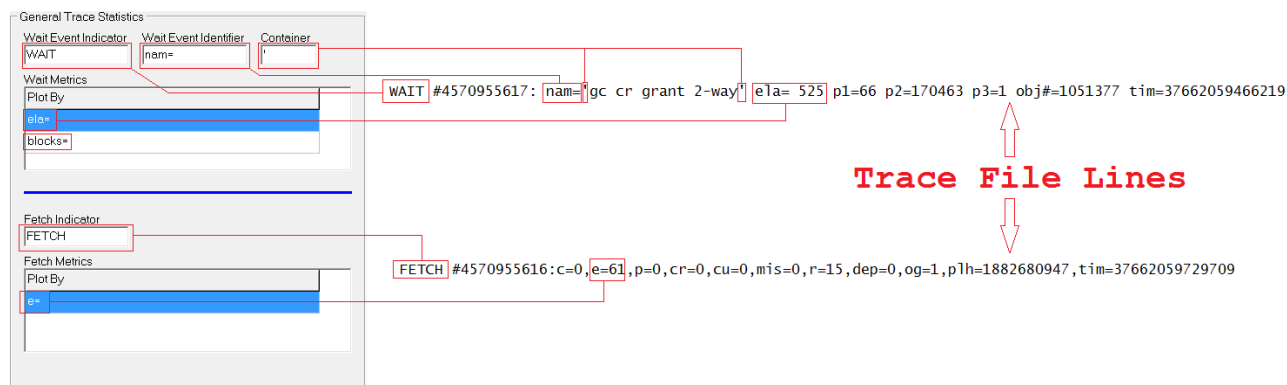


Illustration 4.2.1 Trace Rule mapping to trace file

4.2.2 Wait Event Analysis(Count and Time)

The lines in a trace file are deciphered based on a set of preconfigured trace rules as shown below.

WAIT EVENT INDICATOR	Look for this sub-string at the start of the trace line. Identifies a WAIT event.
WAIT EVENT IDENTIFIER	This sub-string designates how the wait event type is prefixed.
CONTAINER:	The wait event name can be found in a container marked with these values.
WAIT METRICS	Looks for this string in the trace line. The value applicable to this metric is the value which will be plotted in the graphs.
FETCH INDICATOR	Look for this sub-string at the start of the trace line. Identifies a FETCH event.
FETCH METRICS	Looks for this string in the OSW trace line. The value applicable to this metric is the value which will be plotted in the graphs.

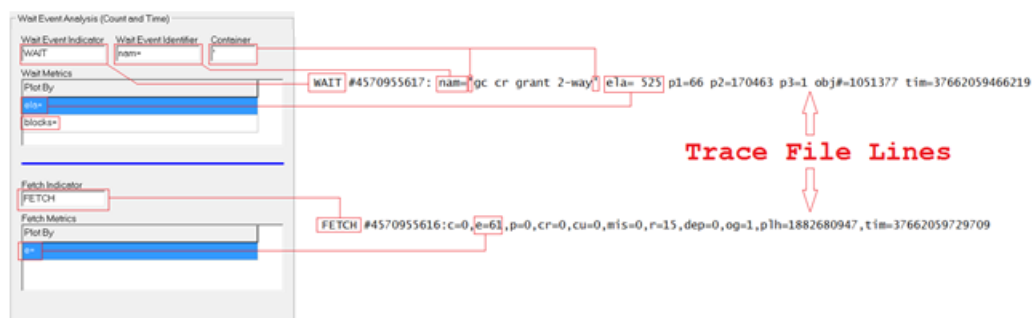


Illustration 4.2.2 Trace Rule mapping to trace file

4.2.3 Block Analysis(Count and Time)

The lines in a trace file are deciphered based on a set of preconfigured trace rules as shown below.

WAIT EVENT INDICATOR	Look for this sub-string at the start of the trace line. Identifies a WAIT event.
WAIT EVENT IDENTIFIER	This sub-string designates how the wait event type is prefixed.
CONTAINER:	The wait event name can be found in a container marked with these values.
METRICS	Look for this string in the trace line. The value applicable to this metric is the value which will be plotted in the graphs.

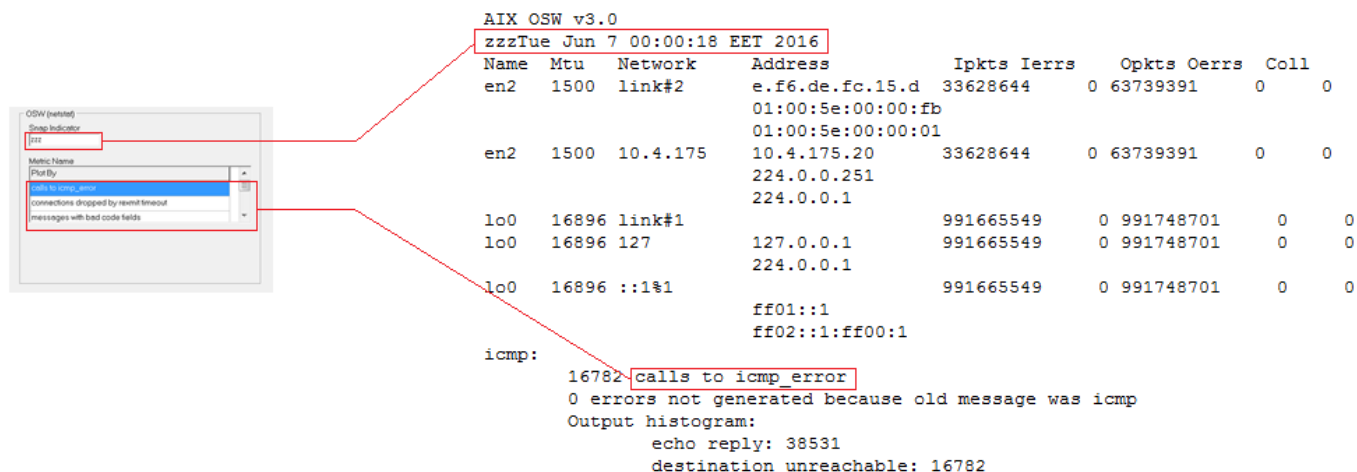


Illustration 4.2.3 Trace Rule mapping to trace file

4.2.4 OSW(netstat)

The lines in a trace file are deciphered based on a set of preconfigured trace rules as shown below.

Snap Indicator	Look for this sub-string at the start of the OSW trace line which indicates the OSW snapshot time.
Metric Name	Looks for this string in the OSW trace line. The value applicable to this metric is the value which will be plotted in the graphs.



AIX OSW v3.0

zzzTue Jun 7 00:00:18 EET 2016

Name	Mtu	Network	Address	Ipkts	Ierrs	Opkts	Oerrs	Coll
en2	1500	link#2	e.f6.de.fc.15.d	33628644	0	63739391	0	0
			01:00:5e:00:00:fb					
			01:00:5e:00:00:01					
en2	1500	10.4.175	10.4.175.20	33628644	0	63739391	0	0
			224.0.0.251					
			224.0.0.1					
lo0	16896	link#1		991665549	0	991748701	0	0
lo0	16896	127	127.0.0.1	991665549	0	991748701	0	0
			224.0.0.1					
lo0	16896	::1		991665549	0	991748701	0	0
			ff01::1					
			ff02::1:ff00:1					

icmp:

16782 calls to icmp_error

0 errors not generated because old message was icmp

Output histogram:

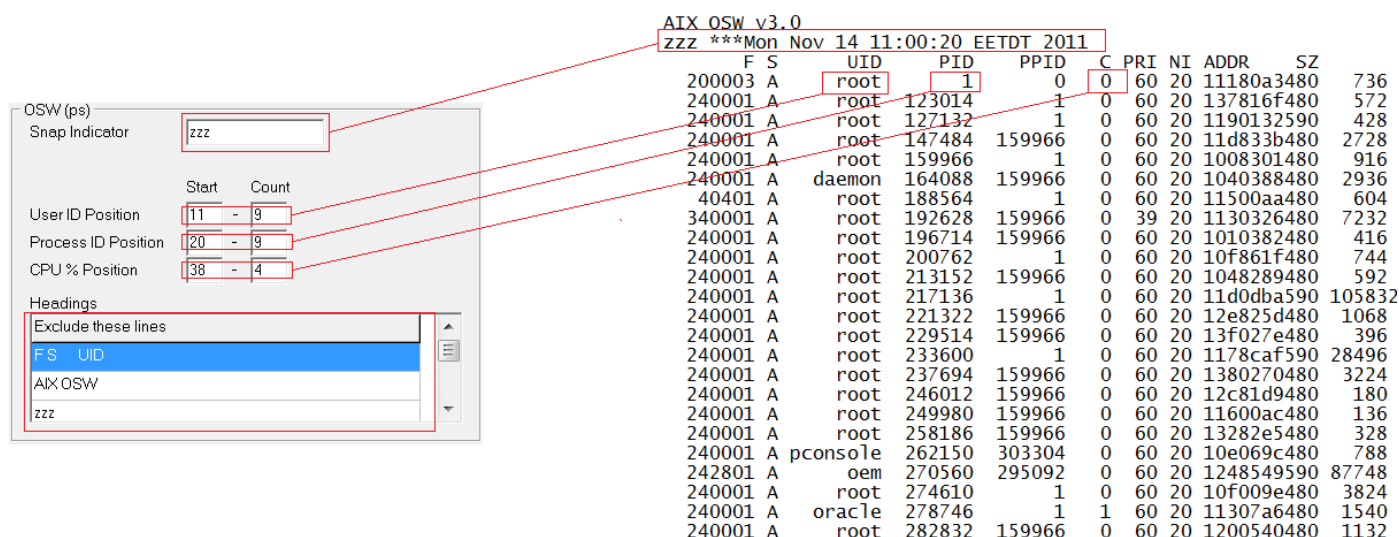
echo reply: 38531

destination unreachable: 16782

4.2.5 OSW(ps)

The lines in a trace file are deciphered based on a set of preconfigured trace rules as shown below.

Snap Indicator	Look for this sub-string at the start of the OSW trace line which indicates the OSW snapshot time.
User ID Position	Position and length of the user ID in the trace line.
Process ID Position	Position and length of the process ID in the trace line.
CPU % Position	Position and length of the CPU % in the trace line.
Heading	Any trace line that contains any of these strings will not be analysed.



AIX OSW v3.0

zzz ***Mon Nov 14 11:00:20 EETDT 2011

F	S	UID	PID	PPID	C	PRI	NI	ADDR	SZ
200003	A	root	1	0	0	60	20	11180a3480	736
240001	A	root	123014	1	0	60	20	137816f480	572
240001	A	root	127132	1	0	60	20	1190132590	428
240001	A	root	147484	159966	0	60	20	11d833b480	2728
240001	A	root	159966	1	0	60	20	1008301480	916
240001	A	daemon	164088	159966	0	60	20	1040388480	2936
40401	A	root	188564	1	0	60	20	11500aa480	604
340001	A	root	192628	159966	0	39	20	1130326480	7232
240001	A	root	196714	159966	0	60	20	1010382480	416
240001	A	root	200762	1	0	60	20	10f861f480	744
240001	A	root	213152	159966	0	60	20	1048289480	592
240001	A	root	217136	1	0	60	20	11d0dba590	105832
240001	A	root	221322	159966	0	60	20	12e825d480	1068
240001	A	root	229514	159966	0	60	20	13f027e480	396
240001	A	root	233600	1	0	60	20	1178caf590	28496
240001	A	root	237694	159966	0	60	20	1380270480	3224
240001	A	root	246012	159966	0	60	20	12c81d9480	180
240001	A	root	249980	159966	0	60	20	11600ac480	136
240001	A	root	258186	159966	0	60	20	13282e5480	328
240001	A	pconsole	262150	303304	0	60	20	10e069c480	788
242801	A	oem	270560	295092	0	60	20	1248549590	87748
240001	A	root	274610	1	0	60	20	10f009e480	3824
240001	A	oracle	278746	1	1	60	20	11307a6480	1540
240001	A	root	282832	159966	0	60	20	1200540480	1132