Kofax Communication Server

tcpMeter Technical Manual

Version: 10.3.0

Date: 2019-12-13

KOFAX

Table of Contents

Chapter 1

Preface

TCPMeter is a tool for measuring the network bandwidth quality or bandwidth availability between a Line Server (LS1/TC16) or Branchbox/TC15 and TCOSS.

TC15 or TC16 uses a TCP point to point connection to transfer data between TCOSS and TC20 (only TC15). This connection must meet minimum quality requirements, what might be a problem in a WAN, especially when a lot of services are sharing the same route. tcpMeter helps to decide if network-quality meets specific requirements before installation of TC15 or TC16.

In this manual TC15 is used as a synonym for TC15, TS15 and TC17. They all use similar hardware architecture. Given bandwidth limits are valid for all types of fax channels, values for legacy Telex, Coax, Twinax and SNA interfaces are listed in the **BranchBox manual**.

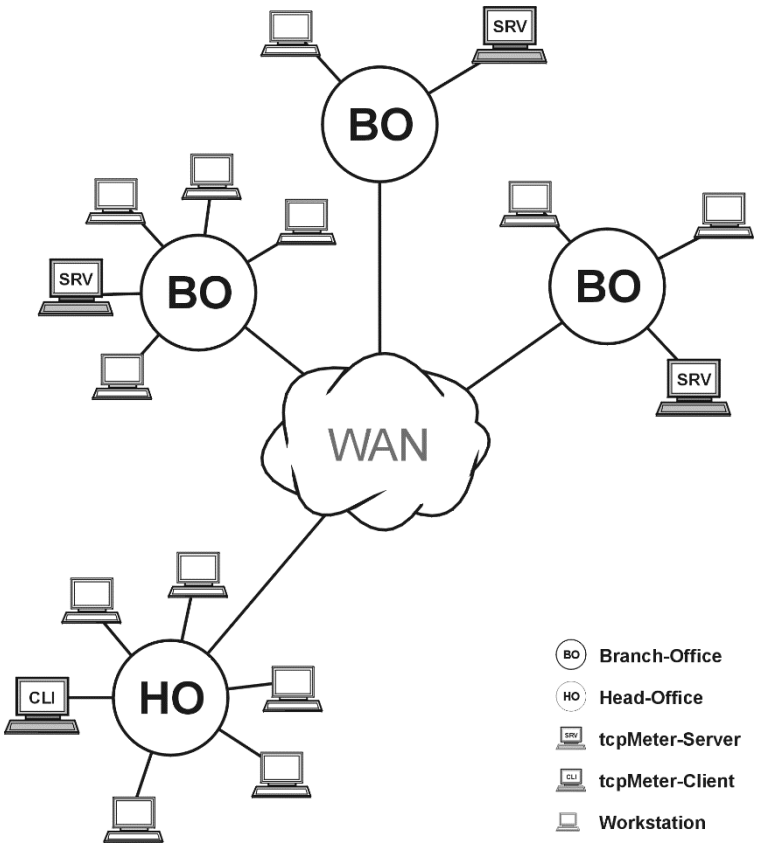
tcpMeter is primary dedicated to be used in WAN's. Usage inside LAN's is possible, but does not make much sense.

The program tcpMeter.exe can act as tcpMeter server or tcpMeter client. Client connects to server, performs metering and logs results to a file. There is another application called tcpMeterEval.exe that reads this file and helps evaluating results.

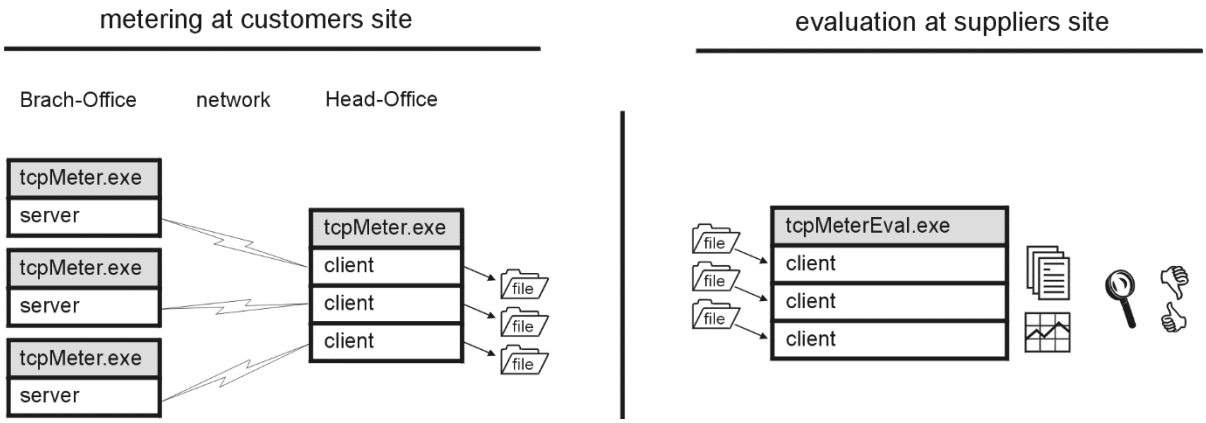
Important The Kofax Communication Server and its components formerly used the name TOPCALL. Some screen shots and texts in this manual may still use the former name.

How It Works with Branchbox

tcpMeter-server is similar to TC15; tcpMeter-client acts similar to TCOSS. Multiple instances of tcpMeter have to be installed to test a network. On the Branch-Office site, where TC15 is / will be located tcpMeter.exe must be installed to act as server. On the Head-Office site, where TCOSS is / will be located tcpMeter must be installed to act as (multiple) client(s).



tcpMeter-client connects to tcpMeter-server and periodically performs metering-tasks, giving information about connection-quality. The results of these tasks are collected in a file (one file per client-server connection, stored local at the client - machine). There is an own executable tcpMeterEval.exe to evaluate the content of these files later.

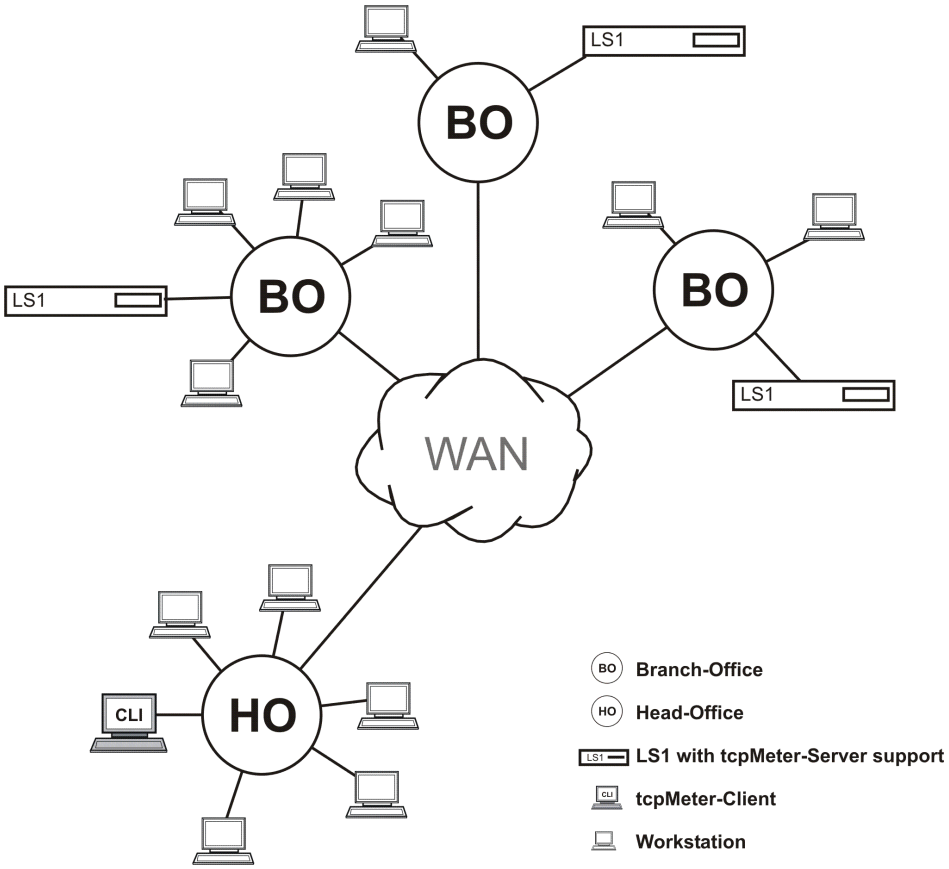


To have an overview on how network-quality differs with time, tcpMeter.exe should collect metering-data at least for one week.

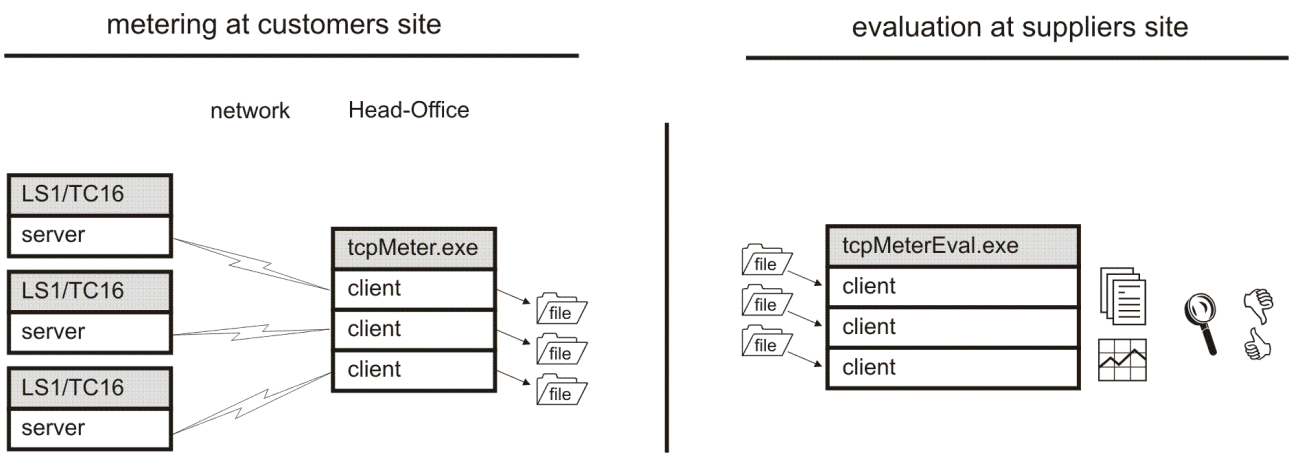
The network load caused by tcpMeter depends on the clients' settings. The amount of disk space needed for logging results too (details later).

How it Works with Line Server

LS1/TC16 running a server (activated with LSD Configuration panel, required the version 1.04.03 or higher of module cs1/TC16 on LS1) with same functionality as tcpMeter-Server; tcpMeter-client acts similar to TCOSS. On the Head-Office site, where TCOSS is / will be located tcpMeter must be installed to act as (multiple) client(s).



tcpMeter-client connects to LS1 and periodically performs metering-tasks, giving information about connection-quality. The results of these tasks are collected in a file (one file per client-server connection, stored local at the client-machine). There is an own executable tcpMeterEval.exe to evaluate the content of these files later.



To have an overview on how network-quality differs with time, tcpMeter.exe should collect metering-data at least for one week.

The network-load caused by tcpMeter is depending on the clients-settings; The amount of disk-space needed for logging results too (details later).

Chapter 2

tcpMeter.exe

tcpMeter.exe runs on all currently supported Windows Server versions. This application does not use Windows registry; its configuration is stored in the file tcpMeter.ini located in the same directory as tcpMeter.exe. The application must have write access to tcpMeter.ini.

Note For more details about supported operating systems, refer to *Platform System Manual*.

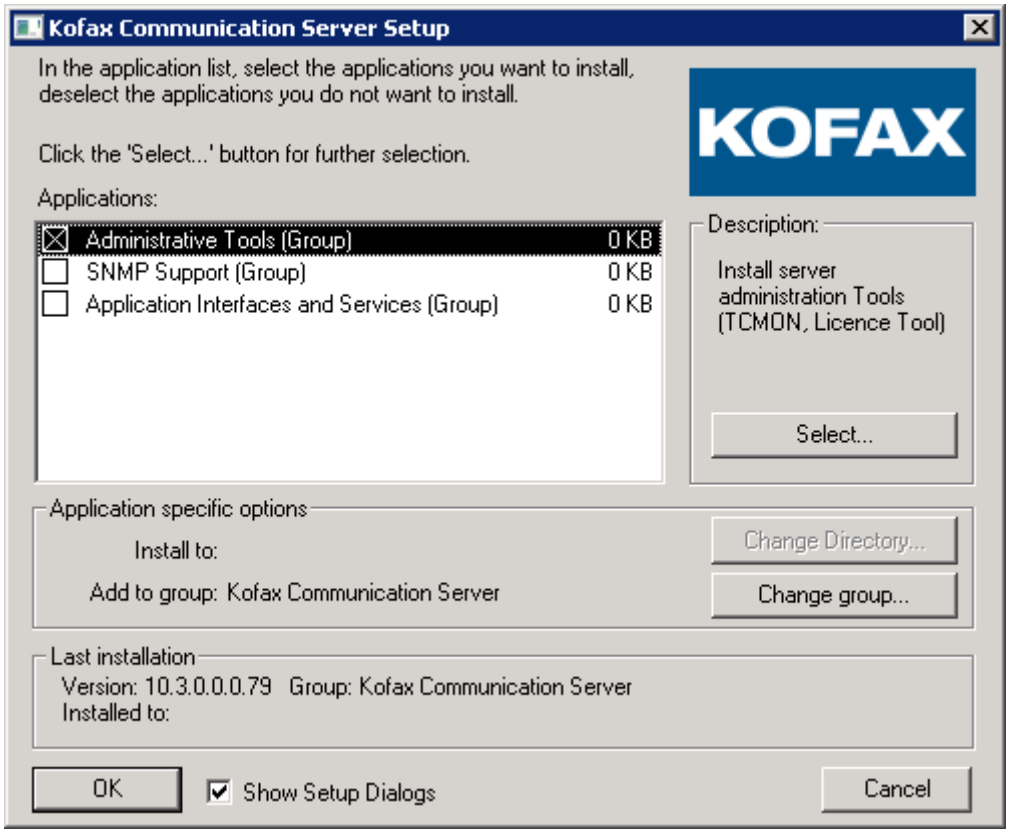
At least 800x600 screen resolution is required.

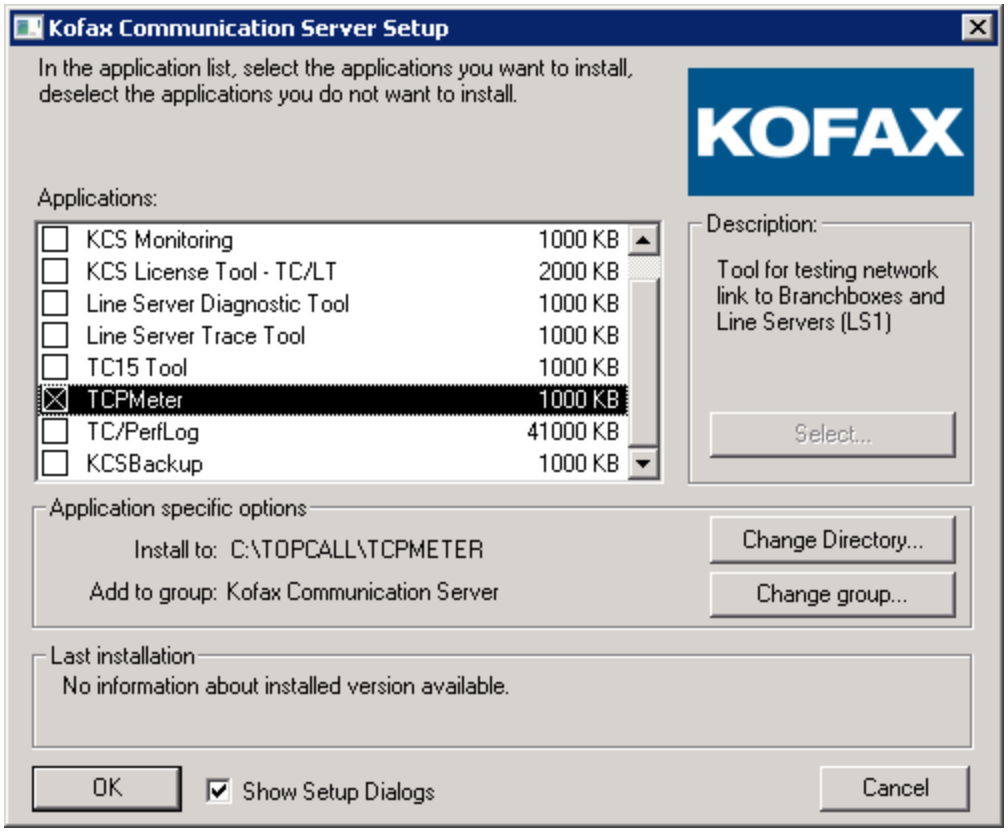
Please pay attention to the tool tip hints appearing when moving the mouse over different window items.

On Windows Server 2008 and later operating systems, telnet is not installed by default. You need to add it as an operating system feature.

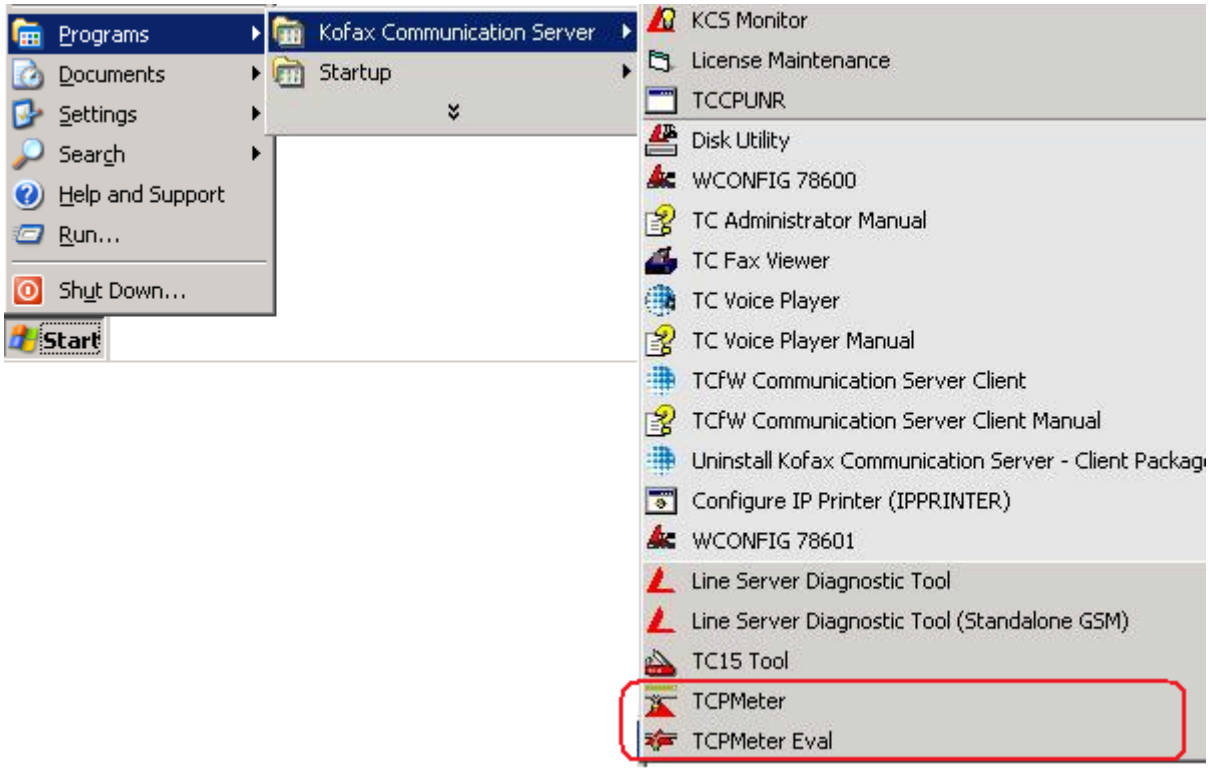
Installation

tcpMeter can be installed with TC/SP 7.62.03 or higher by selecting TCPMeter in the Administrative Tools group in the main screen of the setup.


Kofax Communication Server Setup
In the application list, select the applications you want to install, deselect the applications you do not want to install.
Click the 'Select...' button for further selection.
KOFAX
Applications:
Administrative Tools (Group) 0 KB
SNMP Support (Group) 0 KB
Application Interfaces and Services (Group) 0 KB
Description:
Install server administration Tools (TCMON, Licence Tool)
Select...
Application specific options
Install to:
Change Directory...
Add to group: Kofax Communication Server
Change group...
Last installation
Version: 10.3.0.0.0.79 Group: Kofax Communication Server
Installed to:
OK
Show Setup Dialogs
Cancel



When installing it, the program files for TCPMeter and TCPMeter Eval are copied to c:\topcall\tcpmeter and two shortcuts are created in the Kofax Communication Server start menu group.



Server Operation

This section describes the server operation for Branchbox/TC15.

For Branchbox/TC15

tcpMeter-server has to run on a machine in the LAN of the branch-office (where TC15 is / will be located).

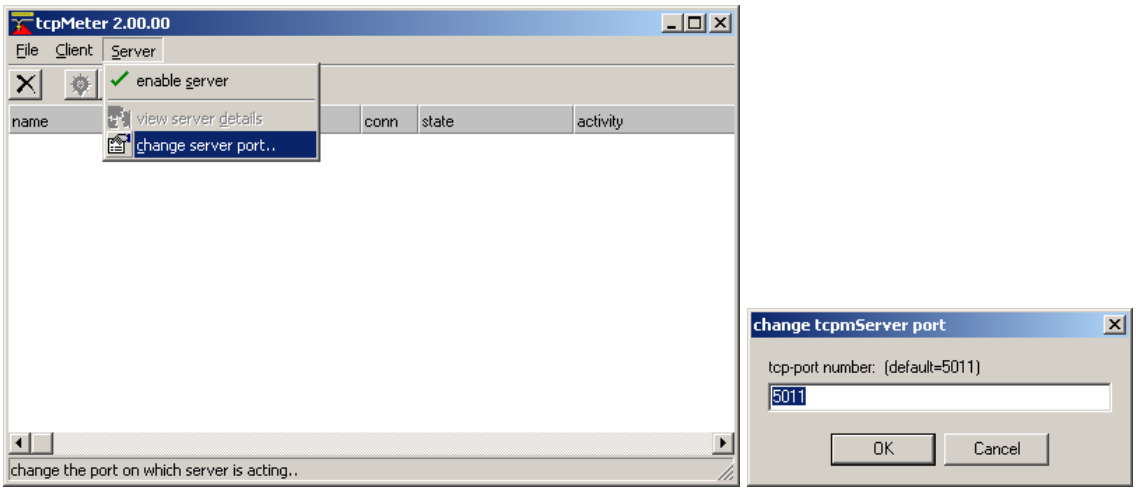
tcpMeter-server accepts incoming TCP-connections from tcpMeter-client. The default port is 5011.

tcpMeter-server(s) should be installed before tcpMeter-client(s).

Step by step procedure:

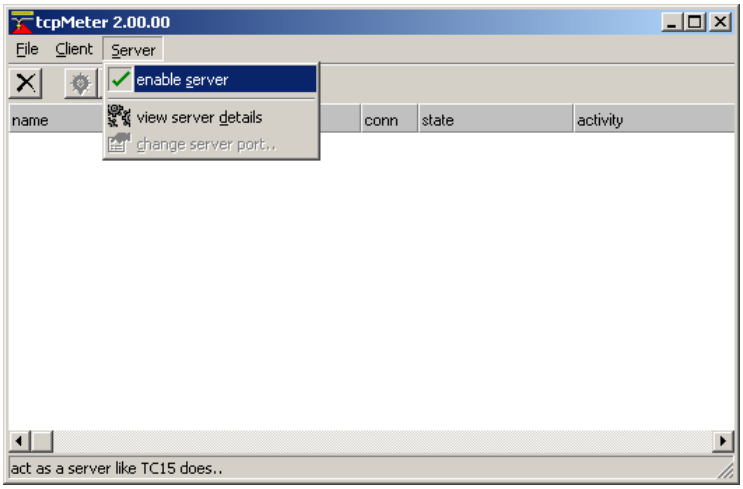
1. Install tcpMeter
   (see chapter Installation)
2. Optionally, set servers port.

There might be the case that customer prefers another port than the default 5011. In this case choose menu-item **change server port** (available if server disabled only) and enter the desired port-number:



3. Enable Server

   start tcpMeter.exe and check menu-item **enable-server**



   Enabling the server results in an error (10048) if server port is already in use by another application or service. tcpMeter.exe saves the server port and enabled/disabled status in tcpMeter.ini. At the next startup of tcpMeter.exe, the server will be enabled on the same port per default.

4. Test tcpMeter-server locally

   choose start-menu **run**, **telnet localhost 5011**

   telnet must be able to connect to tcpMeter-server and must show an output like this:

```
tcpMETER Server 2.xx.xx
(c)2000 TOPCALL International
me: 127.0.0.1:5011 (TC15DEV)
```

```
you:127.0.0.1:1061
```

Now you know that tcpMeter-server is active. Press Enter to close the connection.

- Note down IP settings of local machine and how to reach this machines server-port from the subnet where tcpMeter-client will be located. There might be the need to configure some routers, gateways, firewalls etc. You might need the help of customer's network administrator to find this out. In any way you can test reach ability with telnet as shown above from any point of the network.
- Make sure that tcpMeter-server will permanently run during whole the metering period. Nobody should terminate tcpMeter.exe. Nobody should turn off the machine or log out the current user until metering is completed.
- After metering is completed, one can terminate tcpMeter.exe and delete files tcpMeter.*. tcpMeter-server does not any logging or something of local interest else.

For LS1/TC16

Step by step procedure:

1. Activate tcpMeter-support of LS1 with LSD Configuration panel. (see Line Server 305 Model Manual).

   The port number is fix: **5011**.

2. Test tcpMeter-server

   Select start-menu **run**, **telnet <IP address of LS1> 5011**
   telnet must be able to connect to LS1 and must show an output like this:

```
tcpMETER Server 2.xx.xx
(c)2000 TOPCALL International
me: 0.0.0.0:5011 (LS1ISKO)
you:127.0.0.1:1061
```

   Now you know that tcpMeter-server is active. Press Enter to close connection.

Client Operation

tcpMeter-client has to run on a computer in the LAN of the head-office (where TCOSS is / will be located), e.g. on the TCOSS server itself. Please keep in mind that tcpMeter is a GUI application and therefore needs a user logged in to operate. tcpMeter-servers should be installed before tcpMeter-client.

Install tcpMeter.exe

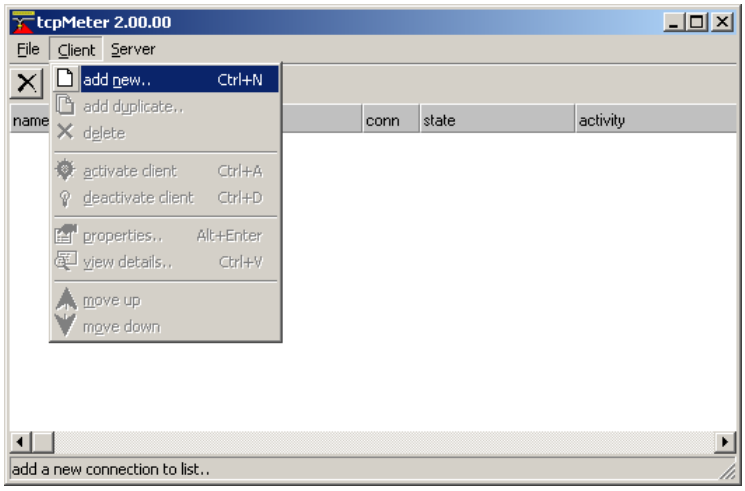
Install tcpMeter (see Installation).

Please make sure that there is enough free disk-space for logging metering results (Refer Tab sheet Timing below for Details).

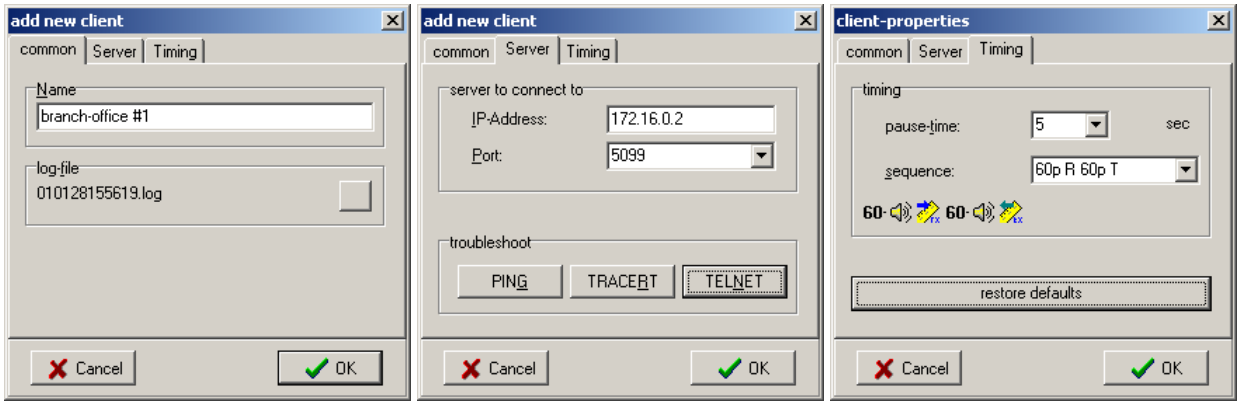
Add / Configure Clients

For each route you want to meter you already have installed a tcpMeter-server at the branch-offices site or LS1 with tcpMeter support. Now you are going to add the corresponding clients. Typically you will set up one client for each server.

Start tcpMeter.exe and choose menu-item **client – add new**



You will get a dialog where you have to enter client parameters:



Tab Sheet Common

Name simple is an informational string used to identify this client. tcpMeter generates a client-unique filename inside the program-directory where meter-results are stored in. This filename is displayed in the box **log-file**. You might enter another filename using the button in the right.

Tab Sheet Server

Relevant only for Branchbox/TC15!

IP-Address and **Port** are Address and Port of tcpMeter-server where to connect. Depending on the network these could also be address/port of a firewall etc.

To verify connection parameters you should use button **telnet**. If parameters are all right you must be able to connect to tcpMeter server and must see servers own IP-address there (that you noted down during installation of tcpMeter-server).

In the example above 172.16.0.2:5099 is a gateway which forwards the connection to the server 194.112.157.141:5020. According to this, in telnet you will see:

```
tcpMETER Server 2.xx.xx
(c)2000 TOPCALL International
me: 194.112.157.141:5020 (touchpos)
you:213.33.11.77:4096
```

“me” is the local IP-address/port of the server (if multi-homed the one at which the connection was accepted) itself. The string put in between the parenthesis is the hostname of the server.

“you” is the IP-address/port which is connected to the server. If there are no firewalls etc. in the route this would be the IP-address of the machine where you are connecting from. In the above example this is an address of the gateway.

Tab Sheet Timing

For normal you will work with the default timing (coming out of tcpMeter.ini) and there is no reason to change this. On the other hand it might be useful to change timing for special purposes.

pause-time is a time of inactivity between two metering-tasks. After one metering-task has finished client is idle for pause-time seconds, and then performs the next metering task.

Sequence tells then client which kind of metering-tasks is to be performed in which order and how often. At the moment the following metering-task-kinds are supported: **P**ing, meter **R**X, meter **T**X .

Ping determines the network latency [msec]. Ping does not cause much traffic and gives a good impression of networks condition (“ping” is NOT an ICMP-ping, it is only called so because of likeness; Ping uses the same, already opened TCP Connection from tcpMeter-Client to tcpMeter-Server as Meter RX- and Meter TX- tasks do).

Meter RX determines the available bandwidth [kBit/sec] from server to client, using a transmission scheme equivalent to TCOSS fax receives (incoming fax).

Meter TX determines the available bandwidth [kBit/sec] from client to server, using a transmission scheme equivalent to TCOSS fax transmission (outgoing fax).

Both meter RX and meter TX causing network traffic worth mentioning. With default parameters (ini) one RX causes approximate 55 kilobyte from server to client and ca. 4 kb from client to server (same amount, opposite direction for TX).

Syntax Details of Sequence

- One can understand sequence as list of actions. Actions will be performed one after the other, results will be written into clients log-file. Between each action client stays idle for pause-time. After the very last action of list the very first action will be performed.
- P means PING, means metering latency
- R means metering RX, means metering bandwidth available for incoming-fax
- T means metering TX, means metering bandwidth available for outgoing-fax
- Blanks are allowed for better readability but does not change the metering sequence P P is equal PP
- _ (Underscore) does “nothing” – after pause time passed next pause starts immediate..
- Numbers are interpreted as multiplier eg. 3P is equal PPP

- Entries are case insensitive, P is equal p

Example:

2p R _ T will lead to [<ping> <pause> <ping> <pause> <RX> <pause> <pause> <TX> <pause>]…

Required disk space

Depending on the kind and result of one meter-action the line written to clients-log is about 20 to 50 characters long. For a worst-case calculation let's take 40 characters per meter-action. So if pause time is 1 second we would get 40 bytes/sec log-output which is 3.4 MB /day or 24 MB/week (ignoring that meter-action itself is taking time).

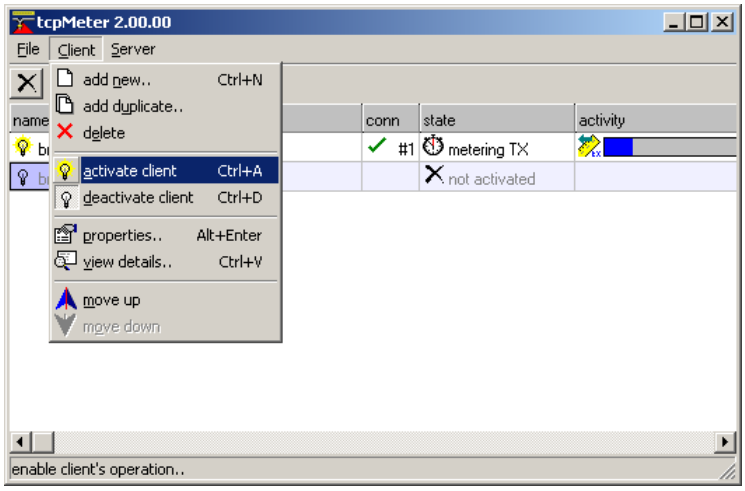
If we are using e.g. 5 seconds pause we only need 1/5th of this meaning ca. 5 MB/week and client.

To avoid killing other tasks, per default tcpMeter stops logging when there is less than 100 MB of available disk space (INI). So if you want to meter e.g. 4 connections for two weeks with 5 sec pause time you should have approximately 4x2 x 24/5 +100 = 138 MB available disk space when start metering. There will be at maximum 384 MB of metering results in 4 files after two weeks.

Activating Client

After adding clients, they are visible in the grid of tcpMeters main-window (one row for each).

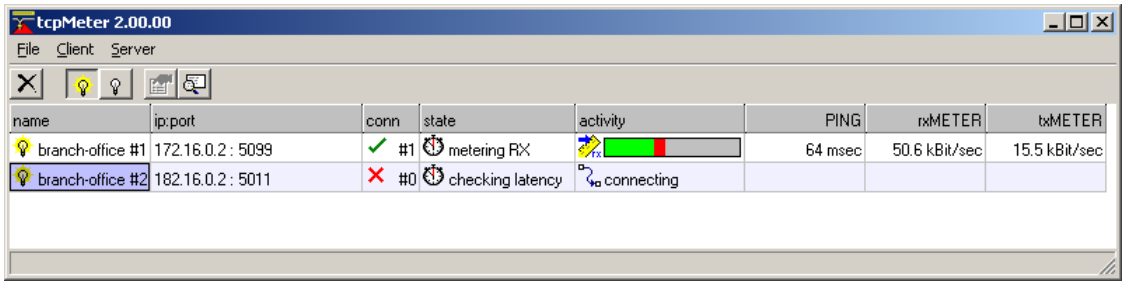
A dark bulb at the very left is indicating that the client is not active. To start a clients operation select the corresponding row and choose menu-item **Client – activate client**.



You can activate / deactivate any client each time. Activation state of client is stored in INI, so that next startup of tcpMeter will recover the last activation state. To change clients-properties it must be disabled. Please keep in mind that multiple activations / deactivations might confuse you during evaluation of the log-files later.

Watch Clients Activity

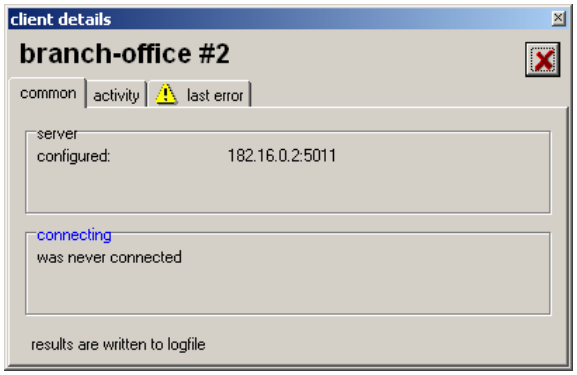
You can see clients' activity in the grid of tcpMeters main-windows. You might have to resize or scroll window to be able to see all columns of grid.

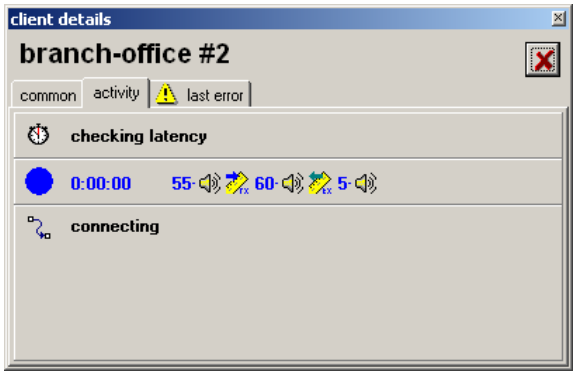


Column **name** simply shows the name you entered in client's properties. Column **ip:port** shows the IP-Address and Port of tcpMeter server entered in client's properties. Column **conn** shows a green hook if connection is established or a red cross if there is no connection. Each successful establishment of a connection is counted and noted beside in this column. E.g. #1 means that the current connection is the very first one which has been made (after activating client) and so there never were a loss of connection.

Columns **state** and **activity** shows what the client actually is doing. In the example above the first client is currently metering RX and about 50% finished. The second client is trying to ping. Therefore he tries to connect. But as we see in column conn he never was able to connect (since activation). Columns **PING**, **rxMeter** and **txMeter** showing the corresponding results of the very last successful metering-action.

You can get more detailed information about a client choosing menu-item **Client – view details**:

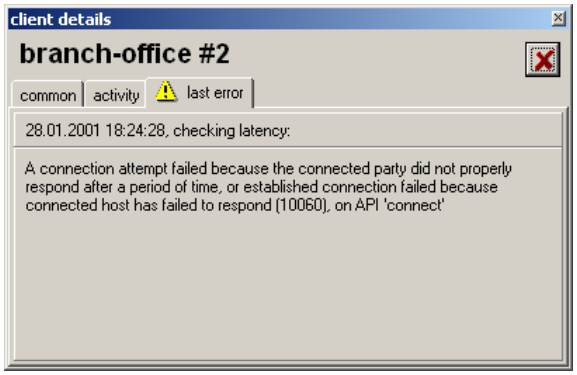




When you move the mouse-cursor over the label containing “results are written to logfile” you can see the name of the file currently used in the tool-tip-hint appearing. By double clicking onto it you might open this file with notepad.

During pause–state you might force pause-end by double click the label containing the remaining pause-time.

You might force a rotation of sequence (without performing the corresponding action) by double clicking onto the sequence-glyph in the right.



The format of the error-message may differ with the operating system. In most cases you will find a value in between parenthesis, which is a result of windows winsock API call WSAGetLastError (wsock32.dll).

Appendix: tcpMeter.ini

The following sample is a shot of tcpMeter.ini after using tcpMeter in the examples above. tcpMeter.ini is a “normal, old-style” INI-file. tcpMeter.exe expects tcpMeter.ini in its own directory (NOT in windows-directory). There must be write-access to tcpMeter.ini (no read-only flag, not usable from CD).

You must not modify tcpMeter.ini as long as tcpMeter.exe is running.

```
[globals]
MinFreeAvailableMB4LOG=100
enterTOS=0
VerifyTOS=0
```

```
TrayIcon=1
```

When there is less than `MinFreeAvailableMB4LOG` [MB] free disk-space tcpMeter-clients stops logging (but still continue executing sequence).

`enterTOS` shows/hides TOS (Type Of Service) related topics in clients-properties dialog.

`VerifyTOS` enables / disables verification of TOS after setting it. Current windows-versions allow requesting different TOS with positive result. But in fact when reading back current used TOS (what is meant with verification) it always reports the same value independent on the requested. Therefore enterTOS and verifyTOS are disabled by default now.

`TrayIcon`=1 enables the usage of the tray-icon inside the task-bars notification area (typical right bottom of screen). When tray-icon is enabled (default) and tcpMeter.exe is minimized it is not shown as button in taskbar (and not in task-list when pressing ALT-TAB) but only as small icon in notification area. This should avoid unintentional closing of tcpMeter.exe by user.

```
[SERVER]
ACTIVE=0
PORT=5011
```

`Active=0` means server will not start up when starting tcpMeter.exe. PORT is the port-number used by server when enabled.

```
[client-default]
name=new client
IP=
PORT=5011
TOS client=-1
TOS server=-1
PAUSE=5
SEQUENCE=60p R 60p T
rxMeter.WindowCount=32
rxMeter.WindowSize=1104
rxMeter.AckSize=84
rxMeter.EvalBase=1024
rxMeter.MeterWindowCount=128
txMeter.WindowCount=32
txMeter.WindowSize=1104
txMeter.AckSize=84
txMeter.EvalBase=1024
txMeter.MeterWindowCount=128
txMeter.FrameSizeServer=21
txMeter.FrameSizeClient=276
txMeter.EvalBaseSize=256
txMeter.MeterFrameCount=200
ping.FrameSizeServer=21
ping.FrameSizeClient=21
ping.ServerSleep=0
ClientActive=0
```

These are the default-properties of newly added client.

`Name, IP, PORT, PAUSE` and `SEQUENCE` are default values for the corresponding input fields of clients properties dialog.

`TOS client` and `TOS server` are the TOS values to be used by server/client. Value –1 means to use operation-systems default value.

The values `rxMeter.xxx` are parameters, which are used for metering RX; `txMeter.xxx` are used for metering TX. Parameters are similar for both directions.

Sender uses <xxMeter.WindowCount> “windows” (which means that as much data-blocks might be pending in parallel). Each data block sent is <xxMeter.WindowSize> bytes of size and will be acknowledged by receiver with <xxMeter.AckSize> bytes.

There are <xxMeter.EvalBase> bytes of really used data transferred in one window. One metering is done over <xxMeter.MeterWindowCount> of windows.

With the values above this means:

200 Windows will be transferred within a single RX/TX from sender to receiver. Each window is sent as 276 bytes from sender to receiver. Receiver acknowledges with 21 bytes each. At maximum 16 windows are pending in parallel. Inside the 276 bytes of window-data there are 256 bytes of actually used data. tcpMeter.exe meters the time needed to transfer all requested windows from the sender to the receiver. Let’s say that this has taken 7 seconds. The transfer-rate then is (200*256) byte / 7 seconds = 7300 bytes/sec = 57 kBit/sec

Previous versions of tcpMeter and LS1 did not support windowing for TX. To be able to run tcpMeter-Client against such a server one may use `txMeter.WindowCount=1`.

The Values ping.xxx are parameters used for metering latency. Clients requests ping with <ping.FrameSizeClient> bytes, server responds with <ping.FrameSizeServer> bytes. Before server responds it will sleep for <ping.ServerSleep> msec. The result of a single ping procedure in LOG is a “RoundTripTime” in [msec] (including bandwidth depending transmission times, latencies and optional <ping.ServerSleep> time).

Per default `ClientActive` is 0, which means that a newly added client must be activated manually.

```
 [cb sequence]
;items listed in combobox 'sequence'
60p R 60p T=default sequence
p=ping only-sequence
r=rxMeter only-sequence
t=txMeter only-sequence

[cb port]
;items listed in combobox 'port'
5011=
5020=

[cb pause]
;items listed in combobox 'pause'
1=
2=
3=
5=
10=
30=
```

Values of sections `cb sequence` are the items of combo-box sequence inside clients properties dialog.

Values of sections `cb port` are the items of combo-box Port inside clients properties dialog.

Values of sections `cb pause` are the items of combo-box Pause inside clients properties dialog.

Values are transferred from INI to corresponding combo-boxes (“customizing”).

```
[clients]
010128155619=branch-office #1
010128175515=branch-office #2
```

The list of clients is stored in section `clients`. For each client there is an entry <section>=<comment>. The client-parameter itself is stored in the referred section. The name of the section is generated from tcpMeter.exe out of current time when user is adding the new client. The same is true for the filename of the log.

```
[010128155619]
name=branch-office #1
logfile=<exe>\010128155619.log
IP=172.16.0.2
PORT=5099
TOS client=-1
TOS server=-1
PAUSE=5
SEQUENCE=60p R 60p T
rxMeter.WindowCount=16
rxMeter.WindowSize=276
rxMeter.AckSize=21
rxMeter.EvalBase=256
rxMeter.MeterWindowCount=200
txMeter.WindowCount=16
txMeter.WindowSize=276
txMeter.AckSize=21
txMeter.EvalBase=256
txMeter.MeterWindowCount=200
ping.FrameSizeServer=21
ping.FrameSizeClient=21
ping.ServerSleep=0
ClientActive=1
```

In this section the parameter of client named “branch-office #1” are stored.

```
[010128175515]
name=branch-office #2
logfile=<exe>\010128175515.log
IP=182.16.0.2
PORT=5011
TOS client=-1
TOS server=-1
PAUSE=5
SEQUENCE=60p R 60p T
rxMeter.WindowCount=16
rxMeter.WindowSize=276
rxMeter.AckSize=21
rxMeter.EvalBase=256
rxMeter.MeterWindowCount=200
txMeter.WindowCount=16
txMeter.WindowSize=276
txMeter.AckSize=21
txMeter.EvalBase=256
txMeter.MeterWindowCount=200
ping.FrameSizeServer=21
ping.FrameSizeClient=21
ping.ServerSleep=0
ClientActive=1
```

In this section the parameter of client named “branch-office #2” are stored.

Appendix: Client Log Files Written by tcpMeter.exe

Each client writes its own log file. Here some typical content.

```
010128 164942 tcpMeter 2.00.00
010128 164942 ACTIVE+
options
 name=branch-office #1  (fast poll)
 logfile=<exe>\010128155619.log
 ip=172.16.0.2
 port=5099
 tos client=-1
 tos server=-1
 pause=3
 sequence=3p R 3p T
 rxmeter.windowcount=16
 rxmeter.windowsize=276
 rxmeter.acksize=21
 rxmeter.evalbase=256
 rxmeter.meterwindowcount=200
 txMeter.WindowCount=16
 txMeter.WindowSize=276
 txMeter.AckSize=21
 txMeter.EvalBase=256
 txMeter.MeterWindowCount=200
 ping.framesizeserver=21
 ping.framesizeclient=21
 ping.serversleep=0
 clientactive=1
010128 164942 tcpMeter 2.00.00 @ 172.16.0.6 (TC15DEV)
010128 164942 CON+ 1
010128 164942 AUT 172.16.0.6:1074 (TC15DEV) => 172.16.0.2:5099 (touchpos)
010128 164943 P 209
010128 164946 P 55
010128 164950 P 61
010128 165001 R 51580 7941 97 603 824
010128 165005 P 56
010128 165008 P 60
010128 165011 P 58
010128 165042 T 15244 26870 119 134 318
010128 165045 P 61
010128 165049 P 56
010128 165052 P 63
010128 165103 R 51730 7918 99 601 770
010128 165107 P 62
...
...
010128 170607 T 15609 26241 121 131 430
010128 170610 P 62
010128 170614 P 61
010128 170617 P 63
010128 170629 R 51303 7984 99 604 888
010128 170632 P 58
010128 170635 P 54
010128 170639 P 57
010128 170708 T 16061 25502 121 128 176
010128 170708 ACTIVE-
```

At client-activation actual software version and client-options are logged. After connection and authentication we can see metering results until the client is deactivated.

```
010128 164950 P 61
```

At 28.1.2001 16:49.50 Ping finished with a result of 61 msec.

```
010128 165001 R 51580 7941 97 603 824
```

At 28.1.20001 16:50.01 rxMeter finished with 51580 bits/second. The whole metering took 7941 msec. The fastest window took 97 msec from sending till acknowledging. Average time between sending and acknowledgment of a window took 603 msec. Maximum window-cycle time was 824 msec.

```
010128 165042 T 15244 26870 119 134 318
```

At 28.1.20001 16:50.42 txMeter finished with 15244 bits/second transmit rate. The whole metering took 26870 msec. The fastest block took 119 msec from sending till acknowledging. Average time between sending and acknowledgment of a block took 134 msec. Maximum block-cycle time was 318 msec.

Chapter 3

tcpMeterEval.exe

tcpMeterEval.exe is a tool to offline evaluate results metered with tcpMeter-clients. As discussed above tcpMeter-clients are logging results into files. Each client (means connection metered) is creating one file. These files can be read by tcpMeterEval. The content of the files can be visualized in different ways.

Installation

tcpMeterEval.exe is a 32 bit GUI-application designed to run on a Windows computer. tcpMeterEval.exe does not touch registry; configuration is stored in file tcpMeterEval.ini located/generated in the same directory as the exe.

At least 1024x768 pixel, high color is recommended. Please pay attention to the tool tip hints appearing when moving the mouse over different windows items.

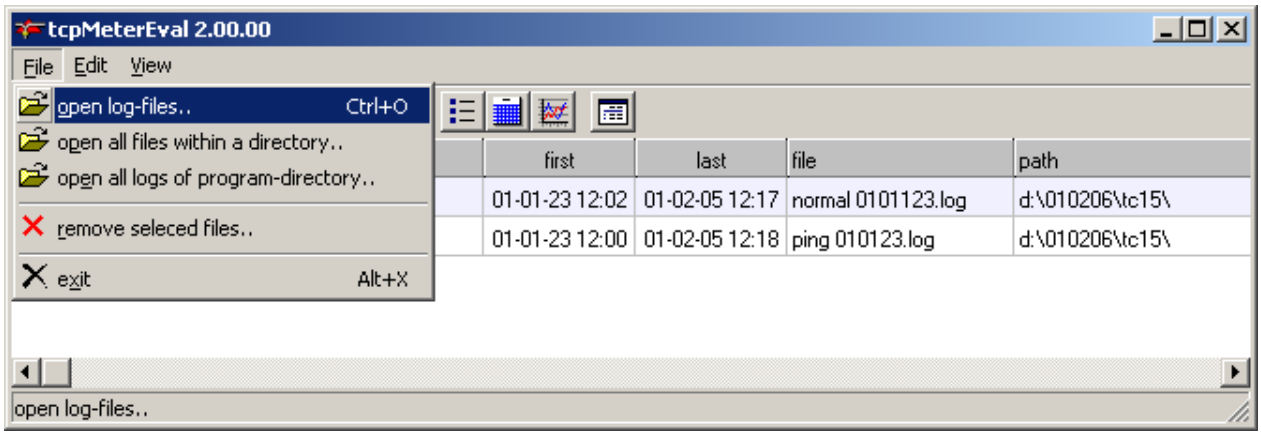
To install tcpMeterEval.exe simply copy files tcpMeterEval.exe and tcpMeterEval.ini to a directory onto local hard disk of target-system. You don’t need administrator rights for this. tcpMeterEval does not require additional hard-disk space.

Always run tcpMeterEval.exe as an administrator.

Evaluating tcpMeter Results

This section describes how to evaluate the tcpMeter results.

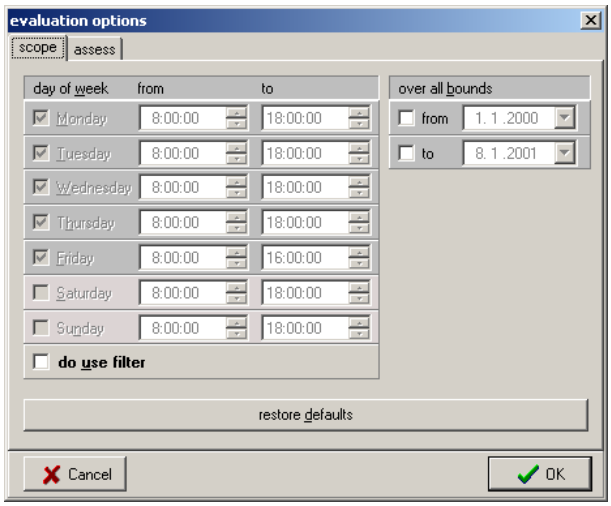
Opening Log Files Written by Client

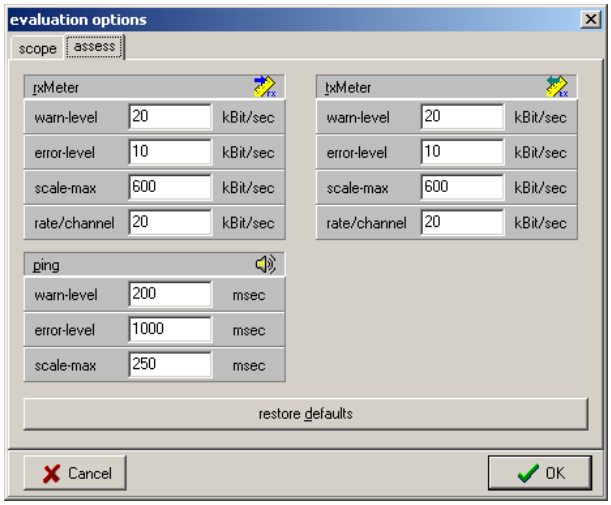


Setting Evaluation Options

The tcpMeterEval tool can be told how to handle / interpret results. You can change these options by selecting **edit – evaluation options** from the menu. Options are loaded from / stored in tcpMeterEval.ini.

Options are common for all clients (global). If you want to assess clients which requires different options, please do this in different sessions (e.g. in another directory with another copy of ini)





Tab Sheet Scope

Scope defines “periods of interest”. Only meter-results captured within scope are displayed / used for calculations.

In the example above no filter is used, which means that all results are “of interest”.

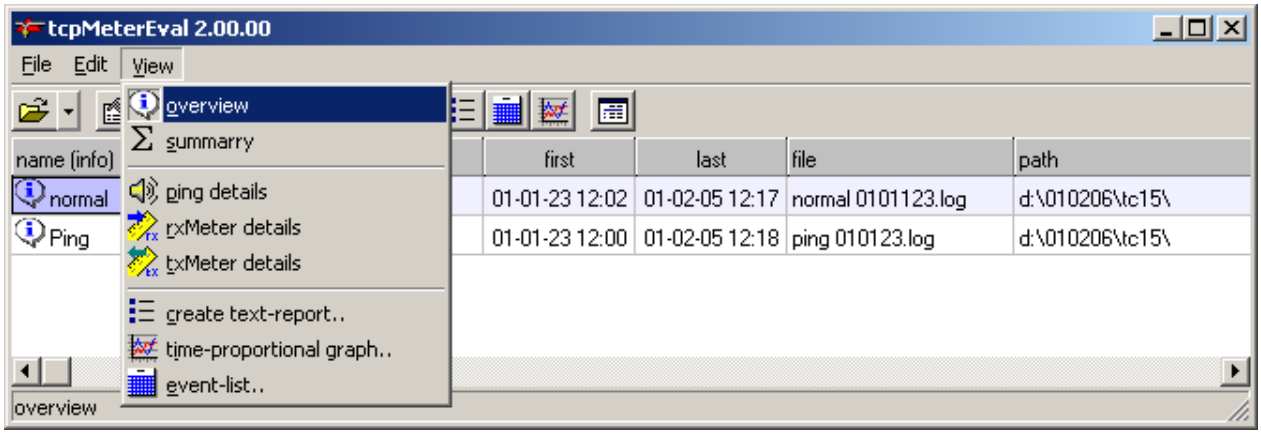
Tab Sheet Assess

Here you have to decide what is “good” and “bad”. For each metering-type there is a **warn-level** and an **error-level**. If the result is worse than the warn-level it is counted as “warning”; if the result is worse than error-level it is counted as “error-result” (please notice that there is a difference between “error” which means that metering failed and “error-result” which means that metering succeeded but the result was bad). The values **rate/channel** are used to calculate the number of available channels out of the metered bandwidth (nChannel=BandWidth/value). The values of **scale-max** are used in graphic to scale values.

You can change options at any time; all current views will be updated with the new options. Options will reflect your expectations. You might modify options from different point of views and have a look how displayed results are affected.

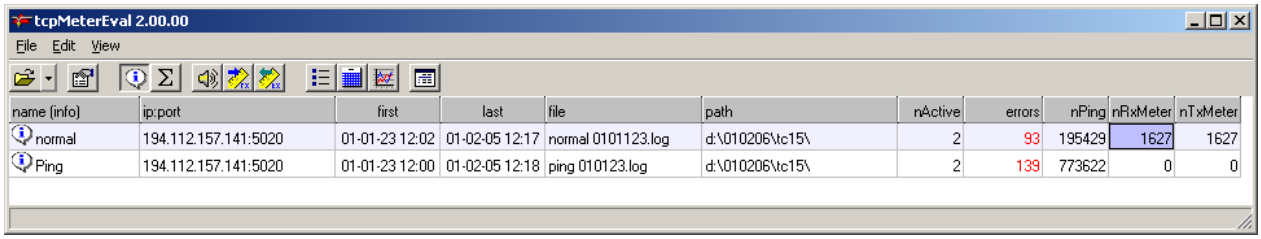
View Modes of Grid

Each line of grid in main-form is showing one client (log file). There are different modes of view for the grid, available in menu View. After selecting a view-mode you might have to enlarge window or scroll window to see all columns. The grid is useful to quickly compare different clients.



Overview

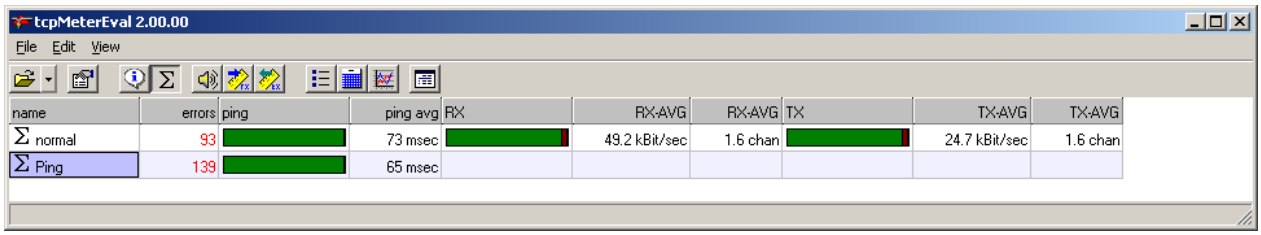
This is the default-mode showing informational columns like filename, etc.:



Column	Meaning
name (info)	the “name” of the client as entered in clients-option; if there is an asterisk added, multiple values was used
ip:port	IP-Address and Port of tcpMeter-server used by this client ; if there is an asterisk added, multiple values was used; the very last used value is displayed
first	date/time of very first event logged in file (yy-dd-mm)
last	date/time of very last event logged in file (yy-dd-mm)
file	name of the log file (at load time, without path)
path	directory where the log file is located (at load time)
nActive	how often client was activated / deactivated (e.g. termination of tcpMeter.exe)
errors	count of errors – means numbers of meter-attempts failed (e.g. no connection etc.)
nPing	count of ping-results in file (number of pings succeeded)
nRxMeter	count of rxMeter-Results in file (number of rxMeter succeeded)
nTxMeter	count of TxMeter-Results in file (number of txMeter succeeded)

Summary

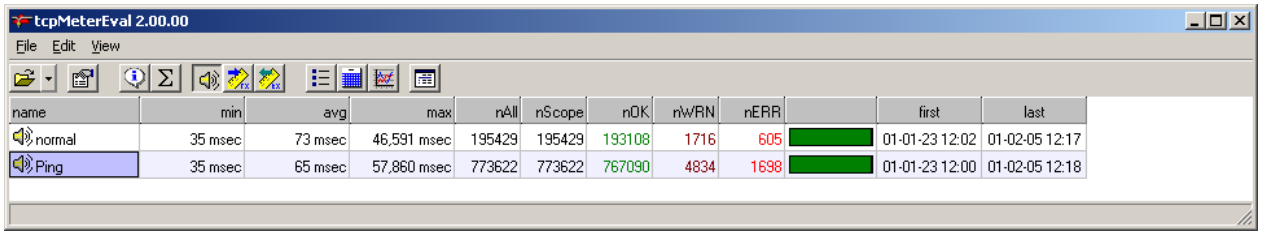
This function gives a quick compare of results in scope:



Column	Meaning
name	the “name” of the client as entered in clients-option
errors	count of all errors in scope – means numbers of meter-attempts failed
ping	a graph showing distribution of results in scope; green range indicating values “in valid range”; dark-red range is indicating “warning-level-results” and red range is indicating values with “error-level-result”
ping avg	the average value of all ping results in scope
RX	distribution of classified rxMeter-results (like ping)
RX-AVG	the average value of all rxMeter results in scope
TX	distribution of classified txMeter-results (like ping)
TX-AVG	the average value of all txMeter results in scope

Ping Details

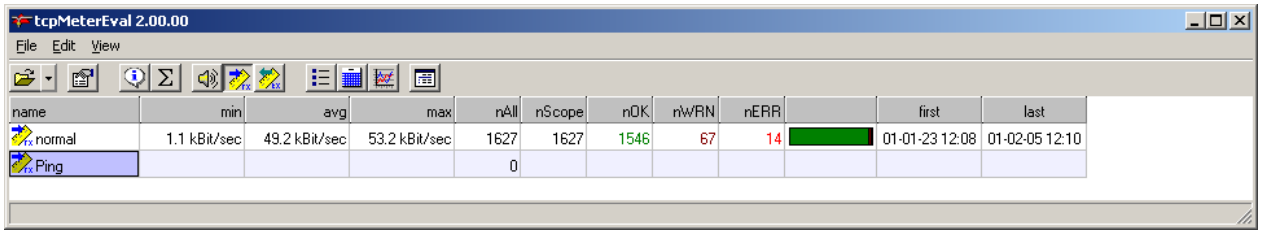
This function gives a quick compare of ping results in scope:



Column	Meaning
name	the “name” of the client as entered in clients-option
min	minimum meter-result
avg	average meter-result
max	maximum meter-result
nAll	count of all pings in file

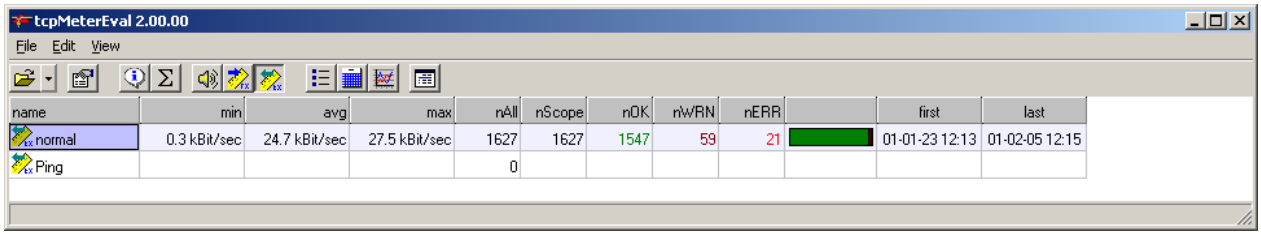
Column	Meaning
nScope	count of pings in scope
nOK	count of results in valid range
nWRN	count of values in warning-range
nERR	count of values in error-range
first	date/time of first ping in scope (yy-mm-dd)
last	date/time of last ping in scope (yy-mm-dd)

rxMeter Details



Columns are equivalent to ping details.

txMeter Details



Columns are equivalent to ping details.

Text Report

By selecting **view – create text-report** from the menu you can generate a textual conclusion appearing in a window. You can view the report in this window, save it to a file, print it out or copy it to the clipboard.

Depending on the events in log file, generating report may take a few moments; changing of evaluation-options will update the report.

There are a few (optional) sections within the report: Common, summary, ping results, rxMeter results, txMeter results, list of connections and list of errors.

Especially the sections list of connections and list of errors could be very long. Most of the content should be self-explanatory.

Example-Report (shortened)

```
------------------------------------------------------------------------------
tcpMeter-report                                        created: 01-02-06 21:17
------------------------------------------------------------------------------
------------------------------------------------------------------------------
common
------------------------------------------------------------------------------
 source:       tcpMeter 2.00.0d @ 10.168.1.87 (witest9) (+)
 client:       normal server:       194.112.157.141:5020
 file:         d:\tcpmeter\eval\normal 0101123.log
 sequence:     60p R 60p T
 pause-time:   5 sec
 first event:  01-01-23 12:02.33
last  event:  01-02-05 12:17.42
------------------------------------------------------------------------------
summary                    over all       in scope   not in scope
------------------------------------------------------------------------------
 number of errors:              93             93              0
 number of pings:           195429         195429              0
 number of rxMeter:           1627           1627              0
 number of txMeter:           1627           1627              0
------------------------------------------------------------------------------
ping results
------------------------------------------------------------------------------
 min. value:                  35 msec
 avg. value:                  73 msec
 max. value:               46591 msec
 in scope:                195429 x          100.0 %
 valid-level:             193108 x           98.8 %
 warning-level:             1716 x            0.9 %  (>  500 msec)
 error-level:                605 x            0.3 %  (> 2000 msec)
------------------------------------------------------------------------------
rxMeter results
------------------------------------------------------------------------------
 min. value:                1.1 kBit/sec     0.0 chan
 avg. value:               49.2 kBit/sec     1.6 chan
 max. value:               53.2 kBit/sec     1.8 chan
 in scope:                 1627 x          100.0 %
 valid-level:              1546 x           95.0 %
 warning-level:              67 x            4.1 %  (< 30 kBit/sec)
 error-level:                14 x            0.9 %  (< 20 kBit/sec)
------------------------------------------------------------------------------
txMeter results
------------------------------------------------------------------------------
 min. value:                0.3 kBit/sec     0.0 chan
 avg. value:               24.7 kBit/sec     1.6 chan
 max. value:               27.5 kBit/sec     1.8 chan
 in scope:                 1627 x          100.0 %
 valid-level:              1547 x           95.1 %
 warning-level:              59 x            3.6 %  (< 15 kBit/sec)
 error-level:                21 x            1.3 %  (< 10 kBit/sec)
------------------------------------------------------------------------------
list of connections
------------------------------------------------------------------------------
  1 01-01-23 12:02 .. 01-01-24 18:52    30:49.57+    18869P      157R      157T
                                         0:00:00-
  2 01-01-24 18:52 .. 01-01-26 21:37    50:45:24+    31924P      266R      266T
 ...
 ...
 41 01-02-04 13:15 .. 01-02-04 13:15     0:00.11+        0P        0R        0T
                                         0:00.05-
 42 01-02-04 13:16 .. 01-02-04 13:18     0:02.07+        5P        0R        0T
                                         0:00.15-
```

```
 43 01-02-04 13:18 .. 01-02-05 08:30    19:11:44+    12238P      102R      102T
                                         0:01:00-
 44 01-02-05 08:31 .. 01-02-05 12:17     3:46.35+     2379P       20R       20T
-------------------------------------------------------------------------------
list of errors
-------------------------------------------------------------------------------
01-01-26 21:37.55+ Read error 64, The specified network name is no longer ava..
01-01-26 21:39.21+ no response..
01-01-26 21:42.09+ no response..
...
...
01-02-04 13:18.08+ no response..
01-02-05 08:30.07+ Read error 64, The specified network name is no longer ava..
01-02-05 08:31.02+ (10060), on API 'connect'
<end of report>
```

list of connections

In this list all established connections are shown. In the example above the first connection was established at January 23th. It was lost at 24th. It was active for 30 hours 59 min. With this connection 18869 ping, 157 rxMeter and 157 txMeter were done.

Times with trailing plus are duration connected; Times with trailing minus are duration disconnected.

list of errors

In this list all errors are shown. Each failed meter-attempt is one error; So e.g. if the network is down and pause-time is 5 sec an error is generated each 5 sec.

There might be a lot of different cryptic error messages; Some of them depend on the operating system at which tcpMeter-client acted. You will often find a value in between parenthesis, which is a result of windows winsock API call WSAGetLastError (wsock32.dll).

Here some typical error strings, created on NT4:

(10060), on API 'connect' : “connection timed out” (WSAETIMEDOUT). A connection could not established because of timeout.

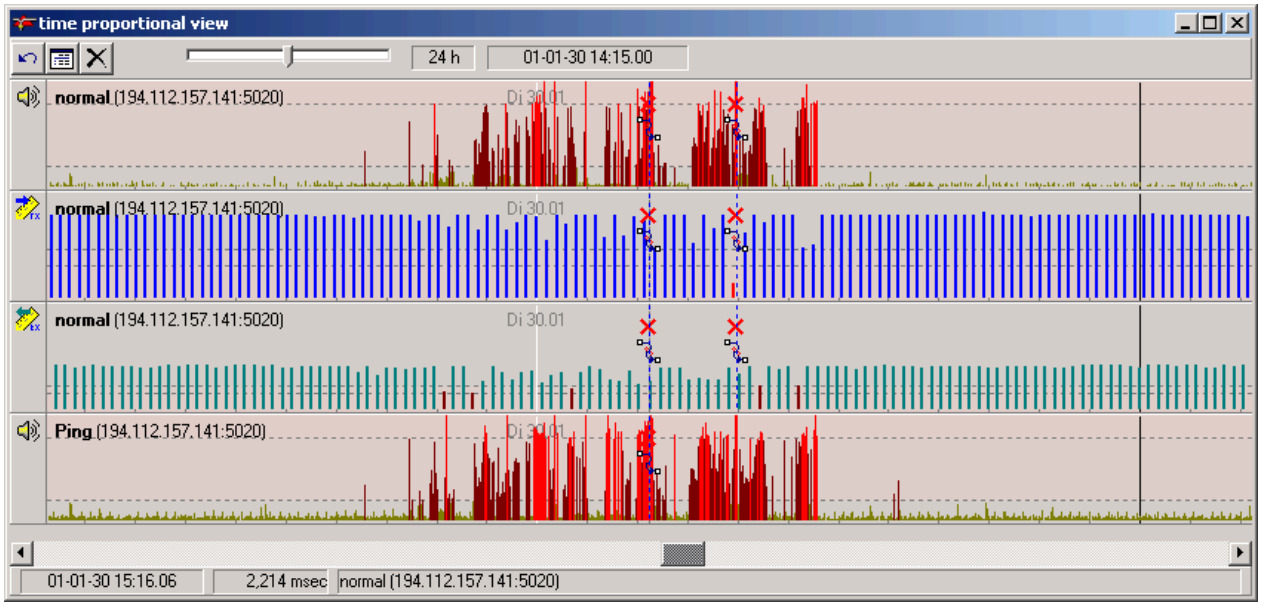
no response.. during metering there was no server-response within one minute – the connection is very bad; tcpMeter-client has terminated the connection.

Read error 64, The specified network name is no longer available: during metering the connection was lost; either tcpMeter-server was stopped or the network is down.

bad server (no hello): a connection could be established, but client did not receive the “server-hello” (the one you can see with telnet). Either there is a wrong server (e.g. FTP etc) or (more often) the connection is as slow as tcpMeter-client thinks there is no proper server at the other side.

Graph

With menu-item **view – time proportional graph** you will get a window, showing meter-results in a graph. Each time you choose this function you will open a new window containing results of actually selected clients (you might select multiple clients in grid).



Each track is dedicated to a client and shows a result-type of this client. You might remove / modify track in its context-menu by clicking with right mouse-button onto it.

You may resize track at its left panel; When holding down ctrl-key during resizing you will set the same size to all tracks of the window.

The vertical scale (result) in tracks is coming out of evaluation-options. The horizontal scale (time) might be adjusted with the trackbar at windows top. The range is form 0.5 to 48 h visible range.

Inside a track midnight is shown as black vertical line; Noon is shown as white vertical line.

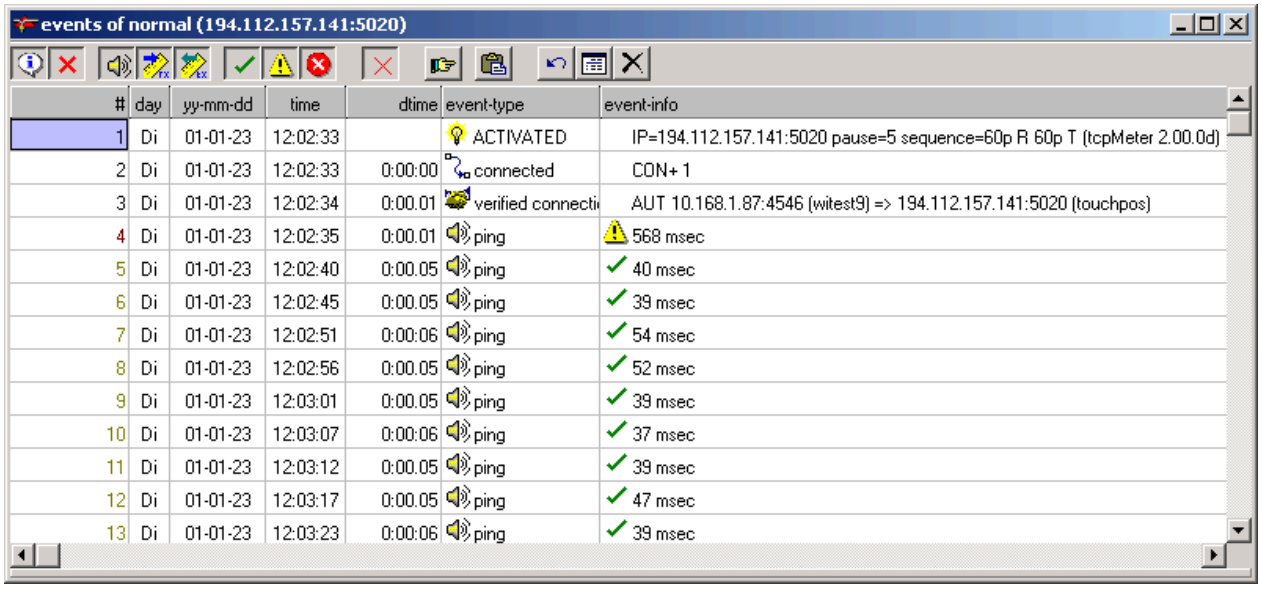
Meter-results are shown as vertical bars, the color is depending on the results classification (valid, warning, error). You can explore values by moving the mouse over the value and watching the bottom line.

Miscellaneous events like connect, disconnect, error etc. are displayed as glyphs.

By double-clicking onto a position inside a track, the corresponding time will be moved to the center of view; If there are already any other windows (graph or event) open they will move to the same time. This is helpful to locate events for a certain time seen in the graph.

Event List

By selecting View – event list you will open an event-window for the client focused in grid. Each time you choose this function you will open a new event-window. Use event list is to explore details or if you exactly want to know what happened.



#	day	yy-mm-dd	time	dtime	event-type	event-info
1	Di	01-01-23	12:02:33		ACTIVATED	IP=194.112.157.141:5020 pause=5 sequence=60p R 60p T (tcpMeter 2.00.0d)
2	Di	01-01-23	12:02:33	0:00:00	connected	CON+ 1
3	Di	01-01-23	12:02:34	0:00.01	verified connecti	AUT 10.168.1.87:4546 (witest9) => 194.112.157.141:5020 (touchpos)
4	Di	01-01-23	12:02:35	0:00.01	ping	568 msec
5	Di	01-01-23	12:02:40	0:00.05	ping	40 msec
6	Di	01-01-23	12:02:45	0:00.05	ping	39 msec
7	Di	01-01-23	12:02:51	0:00:06	ping	54 msec
8	Di	01-01-23	12:02:56	0:00.05	ping	52 msec
9	Di	01-01-23	12:03:01	0:00.05	ping	39 msec
10	Di	01-01-23	12:03:07	0:00:06	ping	37 msec
11	Di	01-01-23	12:03:12	0:00.05	ping	39 msec
12	Di	01-01-23	12:03:17	0:00.05	ping	47 msec
13	Di	01-01-23	12:03:23	0:00:06	ping	39 msec

Each event (meter results, and other things logged by client) is listed. By pressing down / releasing the buttons at the top you can filter which events you want to see. Per default you will see all events. You might e.g. decide to see “ping results with warning level” only etc.

Please have a look to the tool-tip hints (place mouse over button) or simply try what’s happening.

By double-clicking onto an event you will move view-center of eventually opened graphs to the corresponding time.

Additional Hints

This section describes additional hints for using tcpMeter.

Handling of Multiple Windows

tcpMeterEval uses different / multiple windows (report, events, graph). You can open multiple windows in parallel. Such windows may cover any other including the applications main-window. Each window contains a button “windows-list” and “switch to main-form” to help you navigating through the windows.

General used date-format is yy-mm-dd.