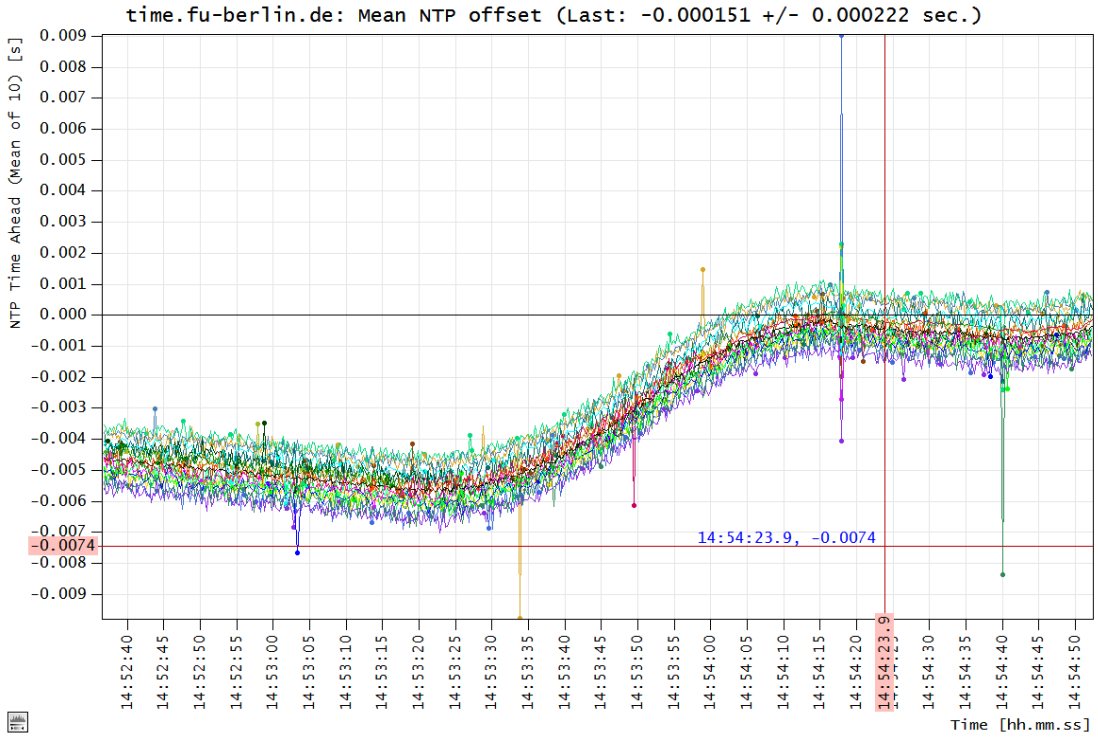
News History:

2017-02-15: Version 2.60 release [Build 0260]

- Version 2.60 introduces NTP server cluster monitoring. This new feature may optionally be enabled by the G_Kernel.exe startup keyword "**cluster**" (e.g. "*G_Kernel.exe gui cluster*". As a result, a set of NTP servers can be monitored simultaneously:



Version 2.6: NTP Offset Tab with a cluster of selected NTP servers.

Members of the NTP cluster are now plotted along with the primary NTP server in the NTP Offset tab. The NTP server cluster can handle up to 60 NTP servers simultaneously.

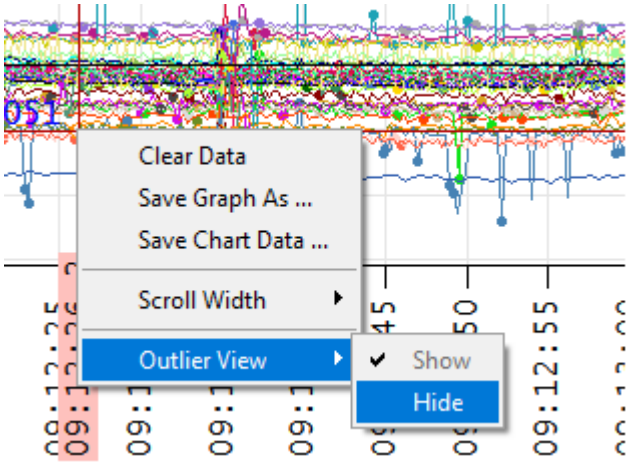
Rejection of unreliable NTP captures has been significantly improved with this version. The filter algorithm has received an adaptive component, which has greatly contributed to much better rejection efficiency. A set of selectable filters can be enabled or disabled from within the GUI (NTP Cluster Configuration menu item) or asynchronously by means of the G_Setup.exe tool (See new G_Setup.exe parameters below). The procedure shown with the graph above is a short drift period followed by a period of local clock disciplining with the primary NTP server as reference. The primary server was time.fu-berlin.de.

Amongst the known view modes (all, scrolled, zoomed), version 2.6 offers showing rejected captures. The *Outlier View* menu item within the chart context menu allows showing or hiding rejected outliers. The graph shown above shows the rejected outliers marked with a dot. Note: The number of shown dot indicators is limited to 1500. Rejected outliers are not discarded but only marked as

rejected. As a result, also *Save Chart Data* will now save the rejected captures to file. However, rejected captures will be marked "rejected" in the file. The default setting of the *Outlier View* is *Hide.*

Keeping the "rejected" data has been a result of filter development. The knowledge of previously rejected captures guides the prediction of rejections for incoming captures.

The *Outlier View* menu item in the graph context menu:



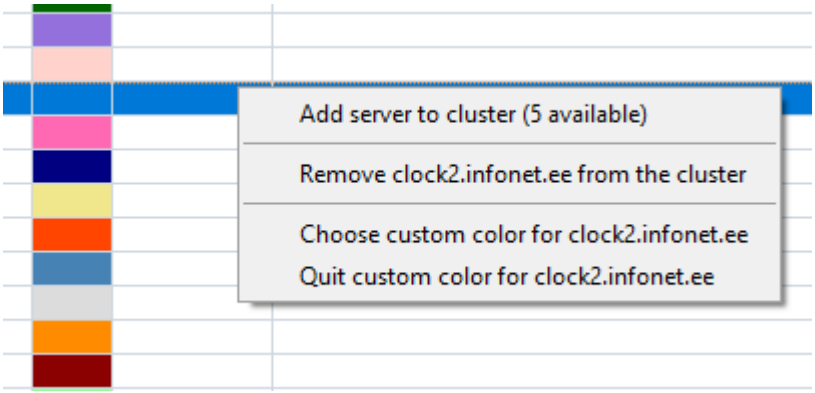
Version 2.6: The Outlier View menu item of the graph context menu.

Cluster members can be configured within the new NTP Server Cluster tab, the NTP Cluster Configuration menu, or the G_Setup.exe tool. This new tab shows "live" information about the configured cluster servers, e.g. name, ip, stratum, round trip delay (RTD), mean RTD (+/- Standard Deviation), and color settings. The color settings are determining the chart color for a specific server. The *color* is system wide and predefined by G_Kernel.exe. This way all instances of G_GUI.exe show the same color for a particular server (default color). However, a custom color can be selected individually within a G_GUI instance. The NTP Server Cluster tab:

Cluster Server	IP	Stratum	Offset [ms]	RTD [ms]	Mean RTD [ms]	Color	Custom Color	Comment
Cluster Mean	Number of server(s): 36		-0.13 +/- 0.71					36 of 36 server(s) contributing to mean
-	87.233.197.123	3	+0.57	28.29	28.50 +/- 0.28			
213.251.53.217	213.251.53.217	2	+1.39	39.39	39.82 +/- 0.56			
mirror	208.53.158.34	3	+2.46	130.84	130.77 +/- 0.57			
ns2.customer-resolver.net	62.116.130.3	2	+0.47	33.78	34.01 +/- 0.28			
91.218.89.74	91.218.89.74	2	+1.33	57.69	56.74 +/- 1.13			
ns1.newsnet.li	[2a00:f8c0:8000::224]:123	3	-2.08	38.43	38.95 +/- 0.55			
re.uni-paderborn.de	131.234.137.64	1	-0.26	28.25	28.11 +/- 0.28			
6.ntp1.pl	[2001:67c:24c:1::20]:123	2	-2.23	47.17	47.38 +/- 0.34			
79.132.231.103.static.edpnet....	79.132.231.103	2	-0.96	60.25	59.80 +/- 0.43			
clock1.infonet.ee	212.7.1.131	2	+3.85	49.95	49.99 +/- 0.42			
gohere.hojmark.net	213.5.39.34	2	-4.32	40.21	40.36 +/- 0.38			
ntp1.pl	91.212.242.20	2	+0.08	38.31	38.62 +/- 0.38			
clock2.infonet.ee	212.7.1.132	2	+3.91	55.32	55.33 +/- 0.30			
ntp2.pl	91.212.242.21	2	-0.17	38.65	38.89 +/- 1.32			
news-archive.icm.edu.pl	193.219.28.147	2	+0.57	35.64	35.66 +/- 0.30			
goodtime.ijs.si	193.2.4.2	2	+1.34	38.95	38.45 +/- 0.33			
ntp1.torix.ca	206.108.0.131	2	-2.71	123.72	123.38 +/- 0.30			
ntp.xpd.se	217.75.106.216	2	-2.27	39.38	39.13 +/- 0.35			
bray.walcz.net	[2a01:258:fffe:f800::1]:123	2	-3.29	49.69	49.89 +/- 0.43			
ntp2.litnet.lt	193.219.61.120	1	+0.21	63.06	63.02 +/- 0.29			
relay-ams.cybertu.be	[2a03:b0c0:0:1010::3:4001...	2	+0.17	32.53	32.72 +/- 0.57			
ns.aksinet.net	195.3.254.2	2	+1.00	72.96	73.23 +/- 0.31			
bray.walcz.net	89.234.64.77	2	-1.65	46.56	47.10 +/- 2.46			
tock.usshc.com	199.102.46.80	1	-0.77	132.04	132.69 +/- 0.39			
tock.usshc.com	199.102.46.73	1	-0.29	133.83	133.94 +/- 0.40			
tock.usshc.com	199.102.46.77	1	-0.86	133.21	133.67 +/- 0.26			
level1e.cs.unc.edu	152.2.133.52	1	+2.80	133.39	133.73 +/- 0.22			
tock.usshc.com	199.102.46.76	1	-0.51	135.04	134.57 +/- 0.51			
clock.sic.hr.net	216.218.254.202	1	-2.30	173.84	173.80 +/- 0.19			

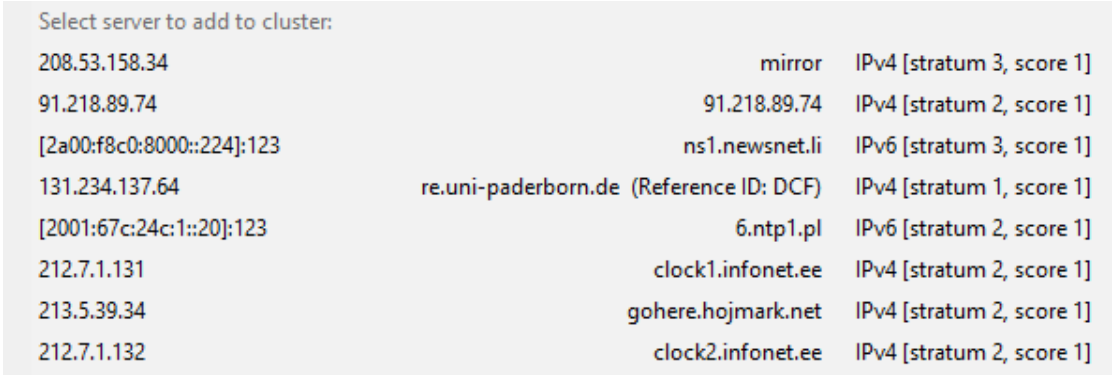
Version 2.6: The NTP Server Cluster tab.

The selected custom color will only apply from after being set. Servers can be also added and be removed from the cluster by the NTP Server Cluster context menu:



Version 2.6: The NTP Server Cluster context menu.

The *Add server to cluster…* menu item allows adding a cluster member from the list of preconfigured NTP servers. The available servers are selectable from a popup list:

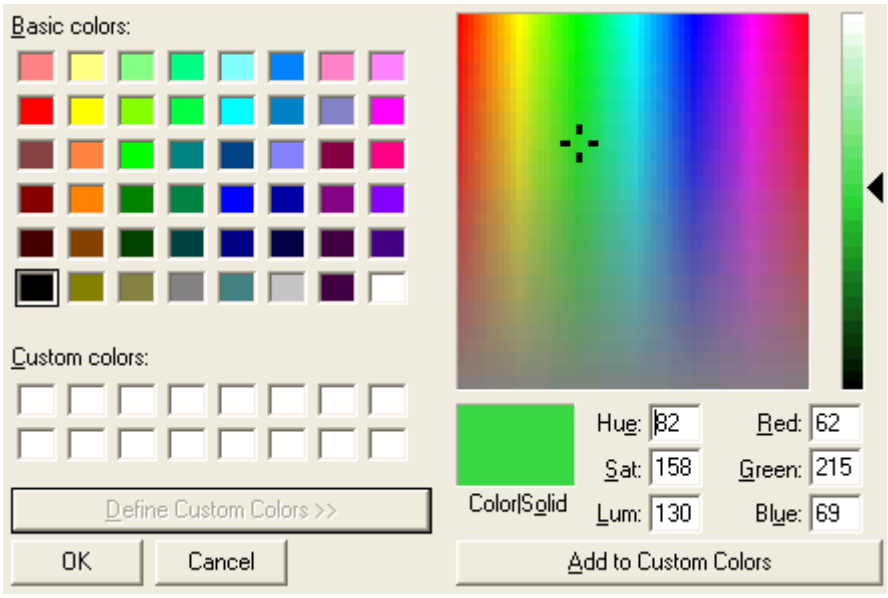


Version 2.6: The Add cluster member popup list.

NTP cluster members can be removed by choosing the *Remove…* menu item. The selected server will be removed from the cluster but it will not be removed from the list of preconfigured servers.

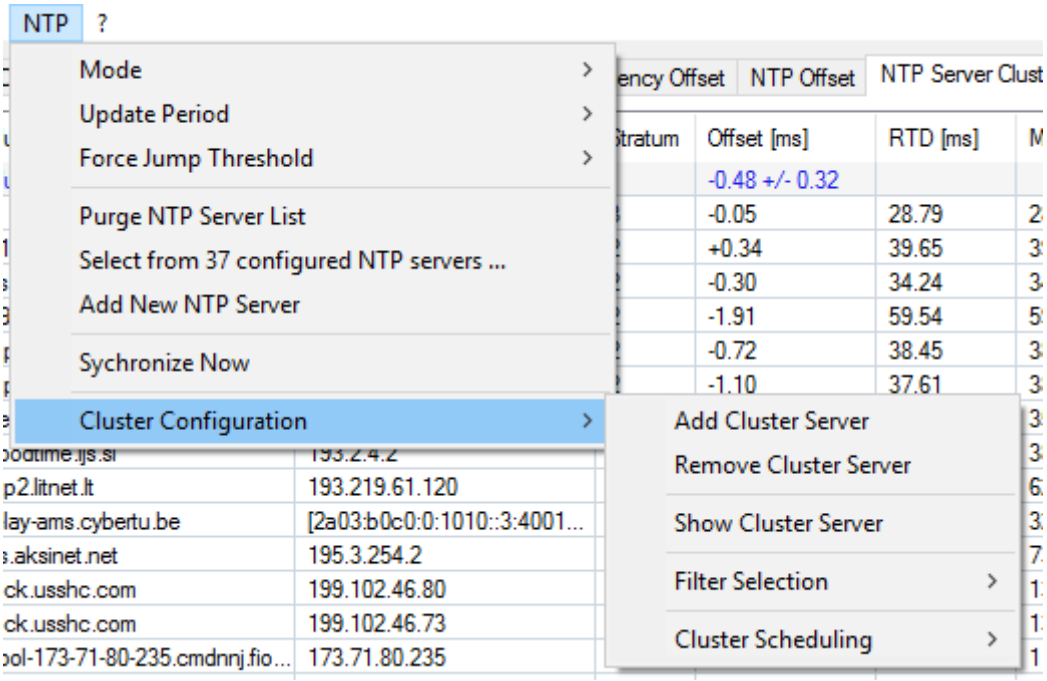
The *Choose custom color* menu item pops up a color chooser dialog box. The chosen custom color will apply for the selected server after being set. A chosen custom color can only be used once. This is audited internally; selecting a color already in use will result in an error message. A few colors are internally disallowed (white, black, and a few other colors). The custom color can be changed at any time. It can also be overridden by a new custom color.

A menu item *Quit custom color…* appears when a custom color is selected for a server. Quitting a custom color will return the chart of the selected server to its predefined default color.



Version 2.6: The color chooser dialog box.

Cluster configuration is done by means of the *NTP Server Cluster tab* context menu as described above or by means of the *Cluster Configuration* submenu in the *NTP* menu:

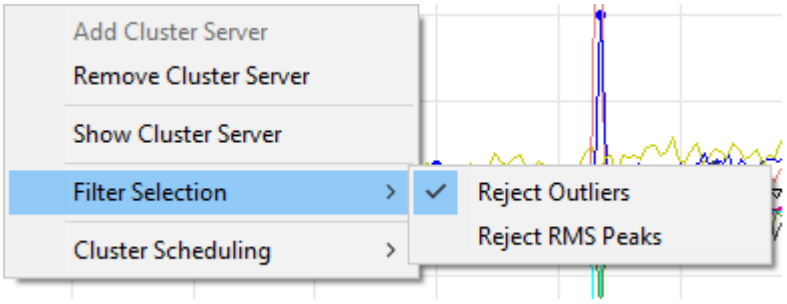


Version 2.6: The *Cluster Configuration* submenu of the *NTP menu.*

Add Cluster Server and *Remove Cluster Server* are implemented as in the *NTP Server Cluster* tab context menu.

Show Custer Server writes a list of current cluster members to the *All Output* tab and thus also into the log file when logging is enabled.

The remaining menu items allow setting the rejection filters and the cluster scheduling. Version 2.6 supports a selection of two filters: The *Reject Outliers* filter and the *Reject RMS Peaks* filter:



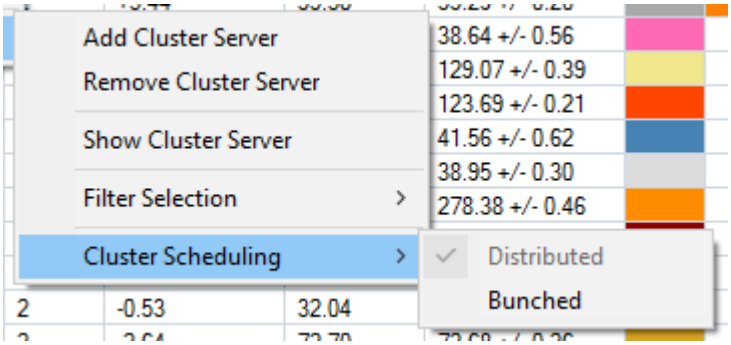
Version 2.6: Cluster Filter Selection.

The *Reject Outliers* filter is based on the evaluation of RTD (Round Trip Delay) and its distribution/variation. This is a very powerful filter which is capable to reject most unwanted NTP captures. As mentioned earlier, it has an adaptive component which allows this filter to learn. It typically takes about 20 captures for the algorithm to converge.

However, heavy network traffic may result in captures beyond the scope of the *Reject Outliers* filter. Such harsh conditions are the field of the *Reject RMS Peaks* filter.

Possible filter settings: Both, each one, or none. The default filter setting is none. Note: The primary server has its own filter. This filter cannot be controlled from outside; it is enabled at all time. Hence, the chart may show rejected captures for the primary server with *Outlier View* = *Show* and *Filter Selection* = *None*.

The scheduling of the NTP captures offers two modes: *Distributed* and *Bunched*.



Version 2.6: Cluster Scheduling.

The primary NTP server determines the NTP capture scheduling for all cluster members. It will trigger first and subsequently the cluster members. The scheduling scheme *Distributed* will schedule the cluster member captures evenly distributed over the selected NTP update period. The *Bunched* mode will schedule the captures in a bunch. The bunch covers an interval matching the shortest selectable NTP update period (250 ms). Within this short interval, the captures are scheduled (again) evenly distributed.

- New keywords:
  - **G_GUI.exe**:
    - **tab=5** starts the GUI with the *NTP Server Cluster* tab.
  - **G_Kernel.exe**:
    - **cluster** starts G_Kernel.exe in cluster mode.
  - **G_Setup.exe**:
    - **server_add[=name/ip]** adds a server to the list of preconfigured NTP servers. Searches a new NTP server automatically with preference to local servers when name/ip is omitted. Note: As of version 2.6. the server is not made the active primary server anymore.
    - **cluster_add[=name/ip]** adds a server to the cluster list. Searches a new NTP server automatically with preference to local servers when name/ip is omitted. If the server exists in the list of preconfigured servers, it is just added to the cluster. If it isn't, it will be added to the list of preconfigured servers first and subsequently added to the cluster.

    Note: Searching a random server is done be means of using the local country code. Multiple attempts to add a new server this way, may result in "Unable to find a new server at this time. Try later…".
    - **cluster_remove=name/ip** removes the server from the cluster. However, it will remain in the list of preconfigured servers.
    - **cluster/filter=[-]filter_name** adds/removes a filter to/from the set of currently used filters. Examples:
      1. **cluster/filter=none** disables all cluster filters.
      2. **cluster/filter=filter_name** adds the filter.
      3. **cluster/filter=-filter_name** removes the filter.

      Supported filter names: "reject_outliers", "reject_rms_peaks".
    - **cluster/schedule=distributed** switches to distributed scheduling.
    - **cluster/schedule=bunched** switches to bunched scheduling.
    - **log="my log text"** logs text to the *All Output* tab. Mind the quotes.
    - **wait_kernel_accurate** causes G_Setup.exe to wait until G_Kernel.exe has finished its initial calibration.
- Fixes:
  - **G_CreateServerSetupScript.exe**:
    - The input keyword for G_Setup.exe has been changed from "ntp=" to "server_add=". Consequently, setup files are only adding servers to the list of preconfigured servers without making them active.

        - Servers whose name cannot be resolved were taken out as of version 2.44. This version takes the exclusion back; these servers are named like their IP.

2016-10-28: Version 2.50 release [Build 0250]

- **G_IO_Service.exe**: Revised queuing, improved performance, and less overhead. Now also handles "future" messages.
- Fixes:
    - **G_Kernel.exe**: Error message "NTP_Service: Bad NTP data, restarting NTP_Service." becomes superfluous.
    - **G_GUI.exe**: Memory leak with ntdll.dll sealed.
    - **G_GUI.exe (x64)**: Erratic termination error fixed.

2016-10-14: Version 2.45 release [Build 0245]

- **G_CreateServerSetupScript.exe**: Improved efficiency with new internal structures. However, the user interface remains unchanged.
- Fixes:
    - **G_GUI.exe**: The chart option "Clear Data" now presents a confirmation request window.
    - **G_CreateServerSetupScript.exe:** Now counting iterations correctly in country code mode.

2016-10-04: Version 2.44 release [Build 0244]

- **G_Kernel.exe** supports a new keyword: **nocalibration**. Starting the G suite with this keyword will force G_Kernel to skip the file time transition calibration during startup and during operation in slow mode.
- Fixes:
    - **G_GUI.exe**: Memory leak within the chart information scheme.
    - **G_GUI.exe**: Automatic scaling of charts after a "clear data" scaled wrong when the menu option was not selected quickly in scroll mode.
    - **G_CreateServerSetupScript.exe**: Only resolvable NTP servers are written to the setup file. An updated version of *NTP_Server_Setup_200.bat* is provided with the package.

2016-09-01: Version 2.43 release [Build 0243]

- **G_GUI.exe:** Switching the **NTP Offset** and **Calibrated Performance Counter Frequency Offset** charts to *scroll-mode* has discarded scrolled out data with previous versions. Hence, the ***Save Chart Data*** option has only written the remaining data to a **csv** file. Even selecting a larger scroll width or disabling the scroll has not recovered the data.

  Version 2.43 introduces caching of scrolled out data in a temporary file. As a consequence, all of the collected data will be kept. Selecting the ***Save Chart Data*** option will restore the scrolled out data from them temporary file and write them in proper sequence with the hold data into the **csv** file. This also applies for the ***auto-scroll*** mode.

  The limit of 10.000.000 points for the chart display (see fixes below) represents approx. 58 days for the **Calibrated Performance Counter Frequency Offset** chart with typically two points/s and approx. 290 days for the **NTP Offset** chart at an NTP update period of 2,5 seconds. Exceeding this range will force the GUI to enter *auto-scroll* mode. The time span (scroll width) of the *auto-scroll* is shown in the chart. The temporary file is created at the location of G_GUI.exe. Its name is composed of the GUIs PID, a name, and the time of creation, followed by a **.tmp**, e.g. "*GUI_12672_NTPO_CHART_DATA_2016-08-29_13_48_48.tmp*" stores the scrolled out data of the NTP Offset chart. These temporary files may appear if required and disappear if no longer used. This scheme allows recording a virtually endless number of chart points. Accordingly, writing the data to a **csv** file may result in very big files.

  Changing the **scroll width** will now always show all available data for the selected scroll width. Nevertheless, the available data may not be enough to entirely fill the selected scroll width. In this case, the chart will keep collecting data until the scroll width is reached and start scrolling thereafter.

- Fixes:
  - **NTP Offset** and **Calibrated Performance Counter Frequency Offset** charts are now limited to 10.000.000 points. The charts will switch into ***auto-scroll*** mode upon reaching this limit. The limit was 2.419.200 points up to version 2.41 and has been taken out in version 2.42. However, after some performance improvements, the limit has now been raised to 10.000.000 points. This new limit suits most platforms and is caused by performance reasons; charts exceeding this limit may cause too high cpu load on less capable platforms.
  - Chart data are now written to a **csv** file following a ***locale-aware*** number formatting.

2016-08-03: Version 2.42 release [Build 0242]

- Fixes:
  - **NTP Offset** chart and **Calibrated Performance Counter Frequency Offset** chart turned into scroll mode at 2419200 chart points. This does not happen anymore.
  - Chart data are now written to a **csv** file with a decimal comma.

2016-07-20: Version 2.41 release [Build 0241]

- Microsoft Visual Studio Community 2015 is a free, fully-featured, and extensible IDE for creating modern applications for Windows. It can be downloaded here. The sample solution **G.sln** is now set up to operate with Visual Studio 2015 (Version 14). Note: Visual Studio may ask to install some additional features (SDK, XP support, etc.) to be compatible with the G suite requirements. However, this is done automatically and possible user intervention is well guided by the Visual Studio update tool.
- Fixes:
  - Kernel.exe in "slow" mode: Infinitely trying to collect file time transition data when the system timer resolution is toggled between 0.5 ms and 1 ms by other applications. The problem occurred with specific drivers, e.g. Bluetooth. G_Kernel.exe now locks the timer resolution to the highest resolution after five failing retries to collect the file time transition data at the preset resolution.
  - Starting NTP services for the first time after the runtime license has expired has caused an infinite wait for calibration accuracy. Fix: Starting NTP services without calibration when the runtime license has expired. This basically means that the NTP services are independent of the performance counter calibration. However, the NTP service will be limited in accuracy due to the use of less accurate local time.
- Supplement: The temporary license, optionally requested during download, lasts until December 31, 2017.

2016-06-30: Version 2.40 release [Build 0240]

- **G_Kernel.exe**: Windows Server 2016 Technical Preview 5 compatible (Builds 14300, 14316, and 14352).
  - Hypervisor detection.
  - Virtual Machine detection and proper adaptation.
  - Option to dump NTP process data into csv file (flags: *dev* and *csv*).
- **G_GUI.exe**: Asynchronous system time adjustments are now indicated be means of purple color.

  Items concerned:
  - *Calibrated Performance Counter Frequency Offset* chart.
  - *NTP Offset* chart.

    - *System Time Adjustment* status line.
- Fixes:
    - Handle leak when NTP server selection changed.
    - Timer resolutions scan for pre VISTA versions.
    - Adjustment gain shown inaccurate for Windows 10.
- Notes:
    - Updated documentation.
    - The Windows timekeeping has undergone changes with Server 2012 R2 and ongoing with Server 2016 to adapt to new international rules and requirements requesting much higher accuracy.

2016-04-06: Version 2.32 release [Build 0232]

- Fixes for system timer resolution issues.

2016-03-30: Version 2.31 release [Build 0231]

- Windows 10 TH2 (Build 10586) compatible.
- Supplements and fixes:
    - License state was reported wrong in "status summary" when no license file was present. Fixed.
    - Timed events lacked accuracy at big time jumps and when "autoadjust" was operating at high gains. Timed events are now also accurate with
        - big jumps in time (Note: Overdue events will signal immediately),
        - "sync_now" requests,
        - "force_jump" events, and
        - toggles into and out of "autoadjust" mode.
    - Corrected ms spin box misbehavior.
    - Persistent toggles to timer resolutions below 1 ms initiated by the OS or other applications (e.g. Windows Media Player) are now treated by locking the timer resolution to maximum resolution during the initial evaluation phase. Long term persistent toggles are eliminated by permanent use of the maximum timer resolution.

2016-01-29: Version 2.30 release [Build 0230]

- A new **NTP** menu item for **G_GUI.exe**: **Purge NTP Server List**. This function purges the internal list of NTP servers. Improved drift estimation.
    - Not responding servers are purged after they failed to respond three times.
    - Specific IPs may result in different server names and vice versa during server setup. Such multiple list entries are purged by removing servers with the lowest score.
    - The purge operation may update the score if it was 0 before.
    - The selected NTP server cannot be purged.

    - Purge is started asynchronously and can only run once. "Purge" and "Add Server" requests are rejected while a purge operation is busy.
    - The entire NTP_Setup functionality remains while a purge is active. However, a server change to a new - yet unlisted - server initiated by G_Setup.exe will terminate the purge operation.
    -

- A new keyword for **G_Setup.exe**: **purge** operates like the **Purge NTP Server List** item in the NTP menu described above.
- A new **?** menu item for **G_GUI.exe**: **Show Status Summary**. This function gathers a status summary and writes it to the All Output tab. The summary contains:
    - Versions, platform, compile date, and startup mode.
    - License information.
    - NTP details.
    - List of clients with their privileges.
- A new keyword for **G_Setup.exe**: **status** operates like the **Show Status Summary** item in the NTP menu described above. However, it also writes the status information to the console.
- A new value for the state field of the **timestamp** structure: TIME_STAMP_LICENSE_EXPIRED (4). The TimeStamp state is set to TIME_STAMP_LICENSE_EXPIRED when the license expires during runtime. The sample code of G_User.c now exhibits it in this way:

```
TimeStamp_TYPE TimeStamp;
GetTimeStamp(&TimeStamp);
if (TimeStamp.State == TIME_STAMP_LICENSE_EXPIRED) {
    fprintf(stdout, "License expired, continuing at default
                     accuracy...\n");
} else {
    fprintf(stdout,"G_Kernel.exe has established calibration,
                    continuing...\n");
}
```

- Note: After the improvement of IPC with version 2.02, the granularity of the functions Time() and GetTimeStamp() is given by the granularity of the performance counter which shows a few 100 ns on a typical "invariant TSC" system. Calls to Time() or GetTimeStamp() only require a few 10 ns, therefore consecutive calls to Time() or GetTimeStamp() may show identical values.
- Supplements and fixes:
    - Revised drift estimation. Note: A suitable NTP update period has to be chosen to enable the automatic local drift estimation. The drift estimation is typically enabled at update periods >= 2500 ms.
    - **G_GUI.exe**: Visual styles enabled.
    - **XP** and/or **32bit**: Resolved reference problem of InterlockedXor (InterlockedXor refers to InterlockedXor64 which is NOT available in XPs Kernel32.dll). XP compatibility healed.
    - **G_CreateServerSetupScript** now adds some "echo" lines into the created setup files to increase its verbosity.

- The GetTimestamp state flag was not updated correctly during the initial calibration phase. Fixed.

2015-10-01: Version 2.20 release [Build 0220]

- Improved drift estimation.
- Updated documentation.

2015-08-26: Version 2.11 release [Build 0211]

- A new **NTP** menu item for **G_GUI.exe**: **Add New NTP Server**. A new NTP server is added to the internal list of servers and made the currently selected server. This functionality does not require NTP to be active. The search of a new server prefers "local" servers by means of applying the current country code whenever available. However, the search is extended to servers outside the country code match, when no new servers can be found quickly. Too frequent use of this option may cause the error *"Unable to find a new server at this time. Try later..."* to appear, because the search is limited to the region. This also depends on the number of NTP servers already stored in the internal list. A retry at a later point in time is likely to be successful.
- A new keyword for **G_Setup.exe**: **add_server** operates like the **Add new NTP server** item in the NTP menu. It adds a new NTP server to the internal list and selects it.

2015-08-07: Version 2.10 release [Build 0210]

- **Windows 10** release version (**Build 10240**) compatible.
- The suite is now **targeted to Windows Vista, 7, 8, 8.1, and 10**. Moreover, it remains compatible with Windows XP.
- Notes:
  - Some security software e.g. virus scanners may delay the start of the suite when it is called for the first time. This may cause parts of the windowstimestamp suite to complain and to terminate with an error window. However, subsequent starts will be successful.
  - Despite the knowledge that GetSystemTimePreciseAsFileTime() misbehaves when a system time adjustment is active (described in the news section of version 1.60), Microsoft has not fixed the inaccuracy during such adjustments with Windows 10. Version 2.10 of the windowstimestamp suite does not use the function GetSystemTimePreciseAsFileTime().

2015-07-29: Version 2.02 release [Build 0202]

- Updated documentation.

2015-07-23: Version 2.02 release [Build 0202]

- Supplements and fixes:
  - G_GUI.exe **(x64)** has suffered a flaw with some data alignment. As a result, maneuvering the hour/minute/second/millisecond spin box across an overflow forced an unintentional termination. Fixed.
  - The inter process communication security scheme has been restructured. This new **lightweight IPC** design has led to a considerable performance advantage. Nonetheless, the x64/x86 interoperability remains fully functional.

2015-07-10: Version 2.01 release [Build 0201]

- G_Setup.exe **timer_resolution** keyword now accepts approximated values for the resolution. This allows easier use of this feature, because the system timer resolutions are calibrated during boot on platforms running Windows above version 7.

  Here is a short excerpt of "*G_Setup.exe query"* to show typical values for available timer resolutions on a typical Windows 8.1 platform:

```
- This platform currently supports 17 different timer resolutions
[100 ns units]:
-    5003 [ 0.5003 ms], thread quantum: 31.5 ms.
-   10007 [ 1.0007 ms], thread quantum: 32.0 ms.
-   20001 [ 2.0001 ms], thread quantum: 32.0 ms.
-   30008 [ 3.0008 ms], thread quantum: 33.0 ms.
-   40002 [ 4.0002 ms], thread quantum: 32.0 ms.
-   50009 [ 5.0009 ms], thread quantum: 35.0 ms (currently active)
-   60003 [ 6.0003 ms], thread quantum: 36.0 ms.
-   70010 [ 7.0010 ms], thread quantum: 35.0 ms.
-   80005 [ 8.0005 ms], thread quantum: 32.0 ms.
-   90012 [ 9.0012 ms], thread quantum: 36.0 ms.
-  100006 [10.0006 ms], thread quantum: 40.0 ms.
-  110000 [11.0000 ms], thread quantum: 33.0 ms.
-  120007 [12.0007 ms], thread quantum: 36.0 ms.
-  130002 [13.0002 ms], thread quantum: 39.0 ms.
-  140009 [14.0009 ms], thread quantum: 42.0 ms.
-  150003 [15.0003 ms], thread quantum: 45.0 ms.
-  156251 [15.6251 ms], thread quantum: 46.9 ms.
```

  The setup procedure now finds the nearest available timer resolution and sets it accordingly. A script file may contain "*G_Setup.exe timer_resolution=5000*" with the result that the resolution is set to 5003 or 0.5003 ms.

- The implementation of the shared memory resource use by **Time()** and **GetTimeStamp()** has been revised and optimized to achieve less overhead. This new scheme has made the mutex protection dispensable, both functions are performed in just a couple of 10 ns.

2015-05-20: Version 2.0 release [Build 0200]

- Windows 10 ("*insider preview*") compatible.

2015-01-07: Version 1.82 release [Build 0182]

- Updated **documentation**: *"Microsecond Resolution Time Services for Windows: Section 2.1.4.3. Timer Periods with Invariant TSC"*
- Fixes for specific Intel 4G mobile processors.

2014-10-13: Version 1.81 release [Build 0181]

- **Full 64-bit support**. Libraries and executables are now available in x86 (32-bit) and x64 (64-bit) versions. 32-bit and 64-bit static or dynamic libraries provide **full interoperability**. Any client (x86 or x64) can operate with any G_Kernel.exe (x86 or x64). Example: G_Kernel.exe (x64 or x86) may operate with an x86 G_GUI.exe and simultaneously with an x64 G_GUI.exe.
- Build version MSC 1800 (Microsoft Visual Studio 12.0). Runtime libraries linked statically, no installation of Visual C++ redistributables required.
- Directory structure of the package adapted to x86/x64 branches. Batch scripts NTP_Server_Setup_200.bat and G_Test.bat need a parameter x86 or x64 to branch into the desired suite.
- Fixes:
    - The handling of specific KoD ("Kiss-o'-Death") messages lead to infinite recovery loops. Fixed.
    - G_Sleep() lasted forever at G_Kernel.exe termination. G_Sleep() will now terminate upon G_Kernel termination.
    - Waitable timed event object handles returned by CreateTimedEvent() never signaled at/after G_Kernel.exe termination. These objects will now signal upon G_Kernel.exe termination. Note: This may cause the waitable objects to signal before the desired time has elapsed in this specific case. However, the service cannot continue anyway after G_Kernel.exe has terminated. The state of G_Kernel.exe may for example be checked by means of the timestamp *State* field. This method was described within the 1.70 release news.
    - The license file may alternatively sit above the x86/x64 binary directories. G_Kernel.exe first tries to find the license file in its directory, in the path, and then at ../.

2014-10-01: Version 1.80 [Build 0180]

- The NTP engine has been extended by a quick test function and a scoring algorithm. NTP servers are quickly diagnosed during setup. As a result, server details like **stratum**, **reference ID/KoD**, and **score** are available to the user. The output of the *NTP_Query()* function, its data structure, and the output of *G_Setup.exe query* has changed accordingly.

  Scoring NTP servers is based on various parameters, some of them are:

  - Duration.
  - Response.
  - Outliers.
  - Stability.

  Version 1.80 assigns **score** values between 0 and 5. A score of 0 is the worst score and indicates a non-responding NTP server. However, specific error conditions may result in negative scores.

- **G_Kernel.exe** automatically configures a *local* NTP server to operate with when no server is specifically requested. In case of errors or discontinuities with a chosen server, the code reverts to substitutional NTP servers. A number of NTP servers may get configured during the course of this substitutional recovery procedure. However, when servers are already configured, the code prefers to revert to the most recently used server showing a score greater than 0.

- **G_CreateServerSetupScript.exe** has been revised and improved. About 200 global NTP servers can be gathered within the first minute of operation. G_CreateServerSetupScript now also performs an initial server test. This way it obtains, similar to G_Kernel.exe, server details like stratum and reference ID/KoD.

  Two new keywords have been added to G_CreateServerSetupScript.exe:

  1. **local** forces local NTP servers to get chosen. This is obtained by means of using the *GetUserGeoID* function. If the user GeoID is not configured, an extended diagnosis of NTP servers will find the appropriate country code information to suit the requested *local* mode. This country code building may take a few minutes.
  2. **max_stratum=n** Testing NTP servers results in knowledge of the servers stratum value. This may be used to filter the search result during the search. Allowed stratum values are 1 to 16. Accordingly, max_stratum=1 will only search for stratum 1 NTP servers, while max_stratum=16 will include any responding server.

  Typing G_CreateServerSetupScript.exe in a console will provide detailed usage information:

```
G_CreateServerSetupScript V1.80 usage:

G_CreateServerSetupScript.exe iterations delay [number [n x
                              countrycode]] [max_stratum=n]
```

```
- iterations: number of pool scans, e.g. 10.
- delay: delay between poolscans in seconds, e.g. 20.
- number optionally specifies a limit for the number of servers
  to create.
- country code, e.g. "US". Multiple codes supported, e.g. "US DE
  SE FI".
  The country code may optionally be "local" to use the local
  country code only.
  "local" option is not allowed in combination with other country
  codes.
  Current local country code: "DE" (Germany).

- max_stratum=1..16 limits the search to servers providing at
  least the specified max_stratum.

Examples: "G_CreateServerSetupScript 10 60
          "G_CreateServerSetupScript 10 60 max_stratum=2"
          "G_CreateServerSetupScript 10 60 GB PL ES"
          "G_CreateServerSetupScript 20 30 12 DE US SE FI"
          "G_CreateServerSetupScript 20 30 local"
          "G_CreateServerSetupScript 20 30 12 max_stratum=2"
          "G_CreateServerSetupScript 20 30 12 PT BR NO
           max_stratum=2"

Note: "Ctrl C" terminates properly with the number of servers
collected at the time of termination.
```

A comprehensive log file is build upon termination. This is an excerpt of such a log file:

```
14:52:24: Total server(s): 28, server hit stats: (hit rank, hits,
country code, address family, stratum, host name, ip)
    .
  27. 1 IE IPv4, stratum=2 "eu-m03.nthweb.co.uk" (46.51.182.47)
  28. 1 IN IPv4, stratum=3 "dnspun.net4india.com" (202.71.140.36)

14:52:24: Total local (DE) "Germany" server(s): 4
  1. 1 DE IPv4, stratum=2 "ntp2.m-online.net" (212.18.3.19)
    .
    .
  4. 1 DE IPv4, stratum=2 "www.danzuck.ch" (46.165.212.204)
14:52:24: Sanity check OK.
14:52:24: End.
```

- The **NTP_Setup()** function remains backwards compatible but has received a few modifications:
  - The host parameter may be NULL or an empty string. This forces the host to remain unchanged during the setup.
  - A period of 0 will force the update period to remain unchanged.
  - Mode = NTP_MODE_STAY (0) will cause the NTP mode to stay unchanged.
- Version 1.80 of **G_GUI.exe** shows stratum and score in the NTP server selection menu.

V1.80 NTP server menu

The availability of server details allows showing parameters like stratum and score in the server selection menu. These parameters are updated automatically. Scoring takes place during operation, score values are updated in the menu accordingly. Some servers even change their stratum every now and then. Also these changes apply to the menu. Stratum 1 servers provide information with their reference ID. The menu shows the ID with stratum 1 servers.

- The *NTP_Query_TYPE* data structure for **G_Setup.exe query** and the **NTP_Query()** function has been extended by score, reference ID, stratum, and address family:

```
typedef struct NTP_Query_TYPE {
  unsigned long msg_status;// optional error status
  unsigned long status;    // states NTP_STATUS_INACTIVE (1),
                           // NTP_STATUS_GATHERING (2), or
                           // NTP_STATUS_ACTIVE (3)
  unsigned long mode;                  // modes NTP_MODE_OFF (1),
                           // NTP_MODE_MONITOR (2), or
                           // NTP_MODE_AUTOADJUST (3)
  unsigned long update_period;   // current NTP update period in ms
  char host[MAX_PATH];     // NTP host URL, e.g. "time-a.nist.gov"
  char ip[MAX_PATH];       // NTP host IP as dotted notation
                           // string, e.g. "178.23.124.2"
  int locked;              // locked TRUE when the past three
                           // consecutive NTP synchronizations were
                           // successful
  int wsa_up;              // windows sockets have been started,
                           // the WinSock DLL was initiated
                           // (WSAStartup)
  double offset;           // current NTP offset
  double mean_offset;      // mean NTP offset
  double stddev;           // standard deviation of mean offset
  long long offset_timestamp; // timestamp of the last NTP
                           // synchronization
  int address_family;      // address family
  int score;               // derived score, as of V1.80 0...5
```

```
  unsigned long refid;      // reference identifier
  int stratum;              // stratum
} NTP_Query_TYPE;
```

The result of **G_Setup.exe query**:

```
G_Setup: NTP query:

  - Local time: 2014-08-29 14:06:05.364760.5
  - Mode: NTP AUTOADJUST
  - Selected host: "xxx.xxxxx.xx"
  - Selected IP: "83.169.43.165" (IPv4)
  - Selected update period: 250 ms (00:00:00.250)
  - Score: 2
  - Stratum: 1 (PPS)
  - MeanOffset: +0.000010 +/-0.000117 s

Timer resolution query:

  - This platform currently supports 7 different timer resolutions
  [100 ns units]:
  - 5000   [ 0.5000 ms]
  - 10000  [ 1.0000 ms], thread quantum: 32.0 ms (currently active)
  - 12500  [ 1.2500 ms]
  - 25000  [ 2.5000 ms]
  - 50000  [ 5.0000 ms]
  - 100000 [10.0000 ms]
  - 156000 [15.6000 ms]
```

- Supplements and fixes:
  - IPv6 compatibility was partially lost with version 1.70. Version 1.80 has returned to full compatibility with IPv6.
  - The entire NTP timeout and recovery scheme has been restructured.
  - GetTimeStamp(): Recursion stack overflow fixed.
  - "Kiss-o'-Death" message handling completed.
  - G_GUI.exe: Context menu outside graph area did not perform the selection, fixed.
  - G_GUI.exe: Setting the scroll range will apply immediately and update the graph.

2014-08-06: Version 1.70 [Build 0170]

- A **new security model** allows clients to operate without administrator rights. This applies to all clients including G_GUI.exe and G_Setup.exe. Specific rights are still required for G_Kernel.exe to operate at the desired performance. G_Kernel.exe uses and distributes the available variety of thread priorities along with thread affinities. MSDN states: *If a thread has the REALTIME_PRIORITY_CLASS base class, nPriority can also be -7, -6, -5, -4, -3, 3, 4, 5, or 6*. The documentation does not mention any rights required to obtain these priorities. However, an unprivileged user may obtain REALTIME_PRIORITY_CLASS but fails when trying to acquire the thread priorities listed above.

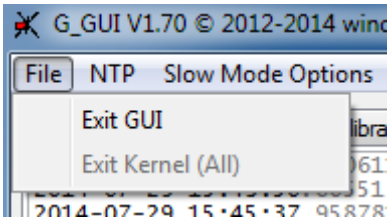
  G_Kernel.exe may also frequently modify the system time. The

SE_SYSTEMTIME_NAME privilege is accessible with administrative rights. Global resources can only be configured and published for use without rights when the creating process has extended rights. All of this lead to the new security model with the following features:

1. **Clients do not need administrator rights.**
2. **Clients may run in separate sessions.**
3. **G_Kernel.exe has to run within a session with extended privileges** (administrator rights).

Note: Clients such as G_GUI.exe and G_Setup.exe are not permitted to send termination requests to G_Kernel.exe when they are running without appropriate rights. All other functions are fully supported without extended rights.

- New or modified G_kernel startup arguments:
  1. **force_jump[=threshold]** The force_jump threshold minimum is reduced to 1 ms. This extended accuracy is accomplished by synchronized system time setting. A threshold of zero disables the force_jump functionality. See the news section for version 1.51 to get more details about the *force_jump* keyword.
- New G_Setup arguments:
  1. **suspend=n** *suspend=1000* suspends the execution of a batch script for 1000 ms. Note: This is only applicable within batch scripts and is NOT using G_Kernel services. Example: *G_setup period=2000 suspend=10000 period=5000 suspend=20000 exit* sets the NTP period to 2 seconds, waits 10 seconds, sets the NTP period to 5 seconds. waits another 20 seconds, and sends a termination request to G_Kernel.
  2. **force_jump=threshold** Allows to asynchronously set the *force_fump* threshold value.
- GUI:
  - G_GUI.exe has received a startup argument **tab=n** Format: "tab=1" to "tab=4", example: *G_GUI.exe tab=4* starts an instance of the GUI with the NTP Offset tab shown. (n = 1 to 4, "All Output" to "NTP Offset")
  - The **ESC** button used to quit the GUI. This feature is now disabled.
  - **Ctrl O** toggles the Hold Output button in the All Output tab.
  - GUI menus:
    1. **File** Menu



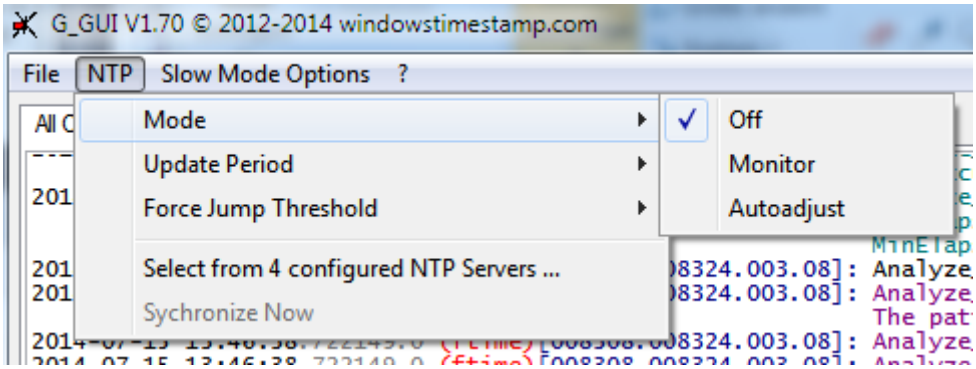
File menu in non-privileged mode

The **File** menu allows exiting the GUI instance (**Exit GUI**). Additionally it allows sending an exit request to G_Kernel.exe (**Exit Kernel (All)**). The latter

is disabled for non-privileged sessions. The GUI's **Stop Kernel** button is disabled accordingly. Only clients on privileged sessions may request G_Kernel termination. As a consequence, G_Kernel.exe will end and all clients will terminate.

2. **NTP** Menu

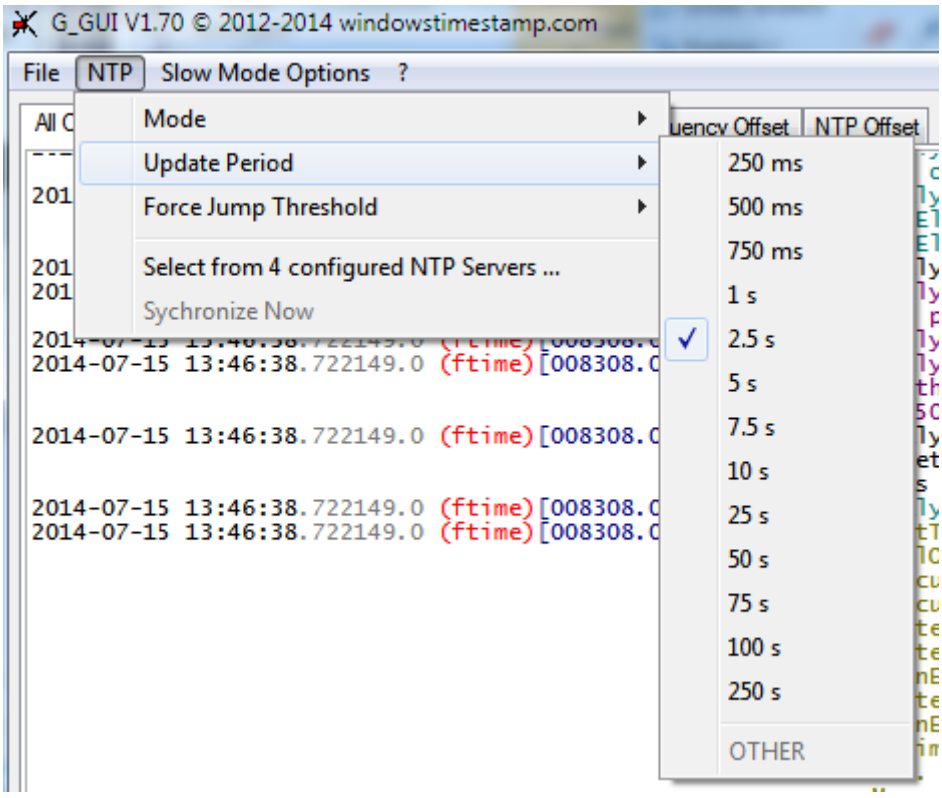
- Submenu **Mode**



NTP mode selection submenu

The **Mode** submenu of the **NTP** menu provides access to the three modes of NTP operation. Mode **Off** disables NTP services, mode **Monitor** enables NTP services, and mode **Autoadjust** holds NTP services enabled while adjusting the system time to closely match the time provided by the selected NTP server. The current mode is checked.

- Submenu **Update Period**

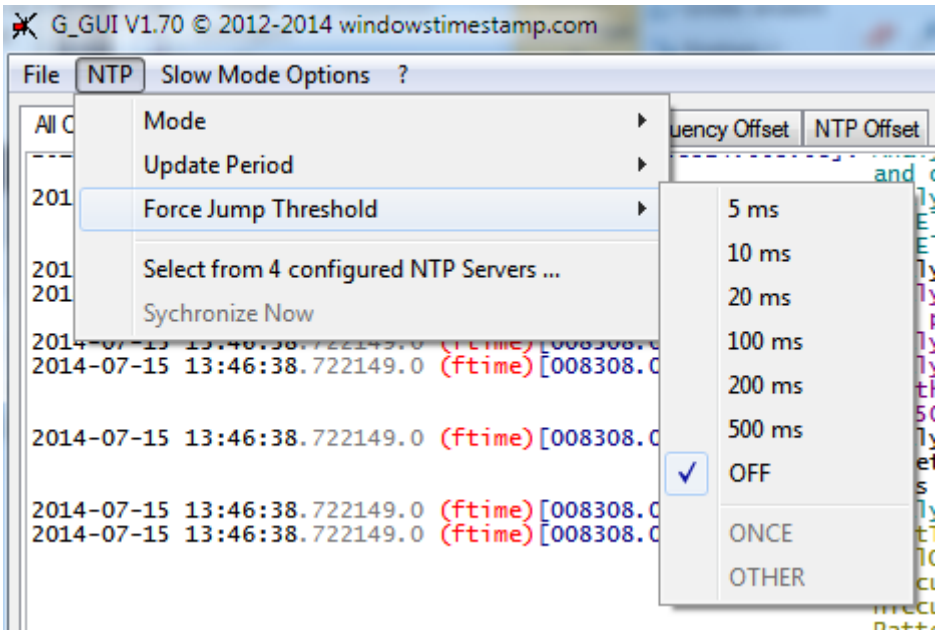


NTP update period selection submenu

The **Update Period** submenu of the **NTP** menu allows selecting an NTP

update period from a given selection of update periods. However, other update periods may be selected by clients using the NTP_Setup() function. The current setting is checked. Values not matching the preset values of this submenu appear at the bottom of the menu (OTHER). The GUI does not provide any means of selecting other update periods. See *G_Setup.exe* and/or the *NTP_Setup()* function for more details about how to set the NTP update period.

- Submenu **Force Jump Threshold**



NTP force jump threshold selection submenu

The **Force Jump Threshold** submenu of the **NTP** menu provides a preset selection of thresholds. The system time is corrected instantaneously whenever the NTP offset (modulus) exceeds *Force Jump Threshold*. Other values are not available for setting within the GUI. However, they are shown at the bottom of the submenu (OTHER). A specific setting is only accessible at startup of G_Kernel.exe: ONCE (*force_jump* keyword without *threshold*). This will trigger the instantaneous system time setting only once after startup. Selected values are checked in the submenu. See *G_Setup.exe* for more information about how to set other *threshold* values.

- Submenu **Server Selection**

NTP server selection popup menu

The **Server Selection** submenu of the **NTP** menu operates is three modes:

1. No NTP server configured: The *Server Selection* menu item shows *No NTP server configured*.

2. One NTP server configured: The *Server Selection* menu item shows the server details directly e.g. *65.55.56.206 time.microsoft.akadns.net*.

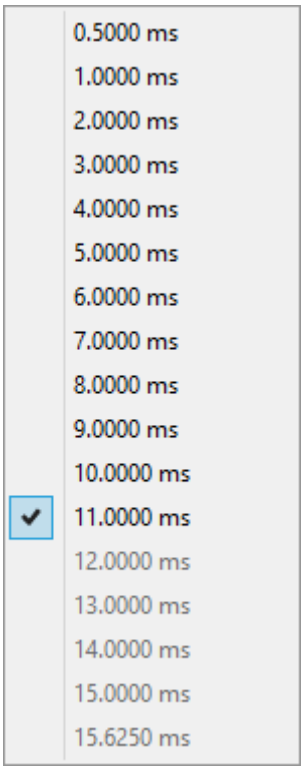
3. Multiple NTP servers configured: The *Server Selection* menu item shows *Select from* n *configured NTP servers...* and pops up a server selection popup menu as shown here with the selection of all configured NTP servers.

- Menu item **Synchronize Now**

  The **Synchronize Now** menu item of the **NTP** menu triggers an instantaneous synchronization of the system time with the NTP server time. This menu item is only enabled when the NTP mode is either *Monitor* or *Autoadjust* and the NTP service is in active state.

3. **Slow Mode Options** Menu

The **Slow Mode Options** is only available when G_Kernel is started with the *slow* keyword. Slow mode is currently only supported on Windows 8 (6.2) and above.



Select Timer Resolution

The **Select Timer Resolution...** submenu of the **Slow Mode Options** allows selecting the system timer resolution. The current timer resolution is checked. Note: The GUI also shows the current timer resolution in the timer resolution status field just above the Stop Kernel button.

The selection is subject to specific restrictions:

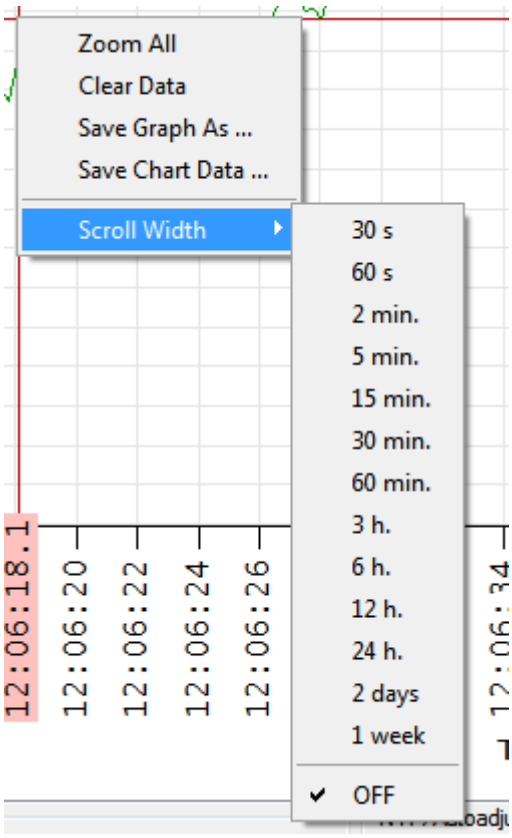
1. The system timer resolution may be set by other applications. It is a system wide setting and is handled by the Windows kernel. The kernel manages a list of timer resolution settings acquired by applications. The highest acquired timer resolution is active. When the application which has acquired the highest resolution releases its timer resolution requirement, the Windows kernel scans the list and reverts to the next highest timer resolution. It may also stay with the same resolution when the same resolution was acquired by other applications.

2. It is not possible to select a timer resolution lower than the currently active timer resolution. As a consequence, some of the timer resolutions shown in the popup menu are disabled.

3. The popup menu is build dynamically. G_Kernel.exe tries to scan all timer resolutions at startup. However, not all resolutions may be available for a scan due to the reasons described above. Therefore G_Kernel.exe tries to complete the list at any resolution change.

4. G_Kernel.exe also evaluates the thread quantum (time slice) for each active timer resolution. The results are shown in the *All Output* tab. They can also be obtained by *G_Setup.exe query*.

- Graph **Context** Menu



Graph context menu

The graph context menu is accessed by the right mouse button and provides the

following options:

1. The **Zoom All** menu item is only available when the display port shows a zoomed area of the data. Selecting **Zoom All** will rescale the graph and show all available data.

2. **Clear Data** will clear all plot data. Warning: Data are not just not shown, they are discarded.

3. The **Save Graph As ...** allows saving the graph as JPEG (.jpg) or Portable Network Graphics (.png) file.

4. **Save Chart Data ...** saves the chart data into a comma separated values (.csv) file.

5. The **Scroll Width** menu item opens the **Scroll Width Selection** submenu. The scroll width setting allows keeping graph data within a specific time span. Warning: Data outside the scroll width will be discarded. The selected scroll width will be checked in the **Scroll Width** submenu and is shown in the x-axis label of the graph. If the scroll width exceeds the time span of the available data, the time span of data is shown without scrolling.

- How to **maneuver** inside ***Calibrated Performance Counter Offset*** and ***NTP Offset*** graphs?

  Zooming and panning is accomplished by mouse input only:

  1. Pressing the left mouse button opens a rubber band box when pressed inside the graph area. Releasing it will finish the rubber band box selection and display the zoomed area. Note: The *Zoom All* menu item is added to the context menu in such a zoomed state.
  2. Scrolling the mouse wheel will zoom in/out symmetrically. Holding the center button of the mouse while scrolling the mouse wheel zooms in/out at the position of the cross hair cursor.
  3. Holding the center button pressed while moving the mouse pans within the data.

  Note: A selected scroll width may move the region of data out of the zoomed viewport.

- Supplements and fixes:
  - Scripting:

    Asynchronous access to the G_Kernel.exe settings was already established by G_Setup.exe. The capabilities of this scheme have been extended by using G_Setup.exe from within batch files.

    1. Example: **G_Test.bat**

       This little script file shows the capabilities of using G_Setup.exe in scripts.

    2. A tool to create server setup files has been added to the package. **G_CreateServerSetupScript.exe** creates a script file named "NTP_Server_Setup_xxxx.bat" in the directory it has been started. The xxxx part of the filename is constructed from input parameters.

Example: *NTP_Server_Setup_de_uk_dk_pl_149.bat* contains 149 NTP servers from Germany, United Kingdom, Denmark, and Poland. A file with 200 worldwide servers is contained in the package (NTP_Server_Setup_200.bat). These setup files do use G_Setup.exe to fill the server list in G_kernel.exe.

```
Usage:
G_CreateServerSetupScript.exe
      iterations delay [number [n x country code]]

- iterations: number of pool scans, e.g. 10.
- delay: delay between pool scans in seconds, e.g. 20.
- number: limit for the number of servers to create.
- country code: e.g. "uk". Multiple country codes supported,
  e.g. "uk de se fi".

G_CreateServerSetupScript runs in 2 phases:

- Phase 1 generates a native list of NTP server pools when no
  country code is supplied.
- Phase 2 scans those pools for new servers in a loop specified
  by "iterations" and "delay"

Example: "G_CreateServerSetupScript 20 30 12 de uk se fi"

Note: "Ctrl C" terminates properly with the number of servers
collected at the time of termination.
```

3. **NTP_Server_Setup_200.bat**

```
:: G_Setup script to configure NTP servers:
:: Note: None of the subsequent G_Setup command lines exceeds
   4095 characters.
::
G_setup ntp=196.25.1.9 ntp=41.248.247.207 ntp=193.108.227.130
        ntp=72.51.27.50 ntp=.....
::
:: Total servers included: 200
```

This file is part of the package and was created using G_CreateServerSetupScript.exe. Note: Windows limits the maximum length of a command line. This limitation is taken into account by G_CreateServerSetupScript.exe. If needed, NTP server listings are split in multiple G_setup commands within the setup file.

- **qfprinf()** now supports output to a *comma separated values* file by using the CSV_FILE (255 << 8) flag. See an example in G_User.c in the samples directory supplied with the package. However, using this requires G_Kernel.exe to be started with the *csv* startup parameter, otherwise the data is lost.
- The **NTP Offset** chart has received two modes of showing the NTP Offset. The

graph enters precision mode at offsets < 1.5 ms. The mean of 10 consecutive captures is displayed in this mode. Precision mode is left when the offset exceeds 2 ms (hysteresis of 0.5 ms). Note: The averaging is reset with a change of the update period.

- Bug fix: The **Clear Data** option in the in chart context menu did not update the chart display, thus no update happened until the next NTP capture occurred. Fixed.
- Note: **InitPipeServices()** is now obsolete. The required pipe initialization is done automatically.
- **Force Jump** now also works in mode AUTOADJUST mode.
- Structures fully support 64 bit alignment (**8 byte packing**). Previous versions required 4 byte packing for compatibility reasons.
- New field in the **timestamp structure**: State

```
typedef struct TimeStamp_TYPE {
  long long Time;                              // 100ns units
  long long ScheduledTime;                     // 100ns units
  double RefinedPerformanceCounterFrequency;   // 1/sec
  int Accuracy;                                // 1ns units
  int State;
} TimeStamp_TYPE;
```

State describes the state of the time calibration. The **GetTimeStamp()** function can be called at any time, no matter whether G_Kernel.exe runs or not. Therefore it is simple to quietly query the state of G_Kernel.exe by means of calling GetTimeStamp().

The *State* member of the TimeStamp_TYPE structure can show the following values:

1. **TIME_STAMP_OFFLINE** **(1)**
   G_Kernel.exe is not running.
2. **TIME_STAMP_AWAITING_CALIBRATION** **(2)**
   G_Kernel.exe is running but has not yet reached calibrated state.
3. **TIME_STAMP_CALIBRATED** **(3)**
   G_Kernel.exe is running in calibrated state.

G_User.c in the samples directory of the package shows how this new member of the TimeStamp_TYPE structure can be used at startup to find out in what state G_Kernel.exe is.

2014-05-22: Version 1.60 [Build 0160]

- New or modified G_kernel startup arguments:
  1. **slow** starts in adaptive timer resolution mode. The systems timer resolution is not set at startup and the time service adapts to any timer resolution set by other applications. This also includes timer resolutions of 0.5 ms. However, older

Windows versions suffer from unpredictable jumps in time when the system timer resolution is modified ("MSDN: Use caution when calling timeBeginPeriod, as frequent calls can significantly affect the system clock, system power usage, and the scheduler") This particularly applies to Windows XP, but it also applies to Windows VISTA and Windows 7 when a system time adjustment is active. The "affected" system time prohibits precise system time adjustment. Consequently, the minimum Windows version required for "slow" mode is Windows 8 / Windows Server 2012.

- Extensions:
  1. Available supported timer resolutions are reported during startup.
  2. The GUI shows the current timer resolution in ms.
- New or modified G_Setup arguments:
  1. **timer_resolution=n** modifies the system timer resolution. This keyword is only available in *"slow"* mode, the Windows version restrictions for *"slow"* mode apply also for the *"timer_resolution"* keyword. Example: *"G_Setup timer_resolution=100000"* sets the timer resolution to 10ms (Use the *"query"* keyword obtain the available resolutions).
  2. **sleep=n** suspends the execution of G_Setup arguments for n milliseconds. Example: *"G_Setup period=500 sleep=15000 sync_now"* sets the NTP period to 500 ms, waits 15 seconds, and synchronizes to an NTP server.
  3. **query** has additional output:

```
NTP query:
- Local time: 2014-05-14 17:10:03.618096.9
- Mode: NTP MONITOR
- Update period: 620 ms (00:00:00.620)
- Host: "time.fu-berlin.de"
- IP: "130.133.1.10"
- StdDev: +/-0.003753 s
- MeanOffset: +0.386032 s
- Locked: 1
- WSA_up: 1
- Offset: 0.377741
- OffsetTimestamp: 2014-05-14 17:10:03.012835.7
Timer resolution query:
- This platform currently supports 6 different timer
  resolutions [100 ns units]:
- 5000 [ 0.5000 ms]
- 10000 [ 1.0000 ms], thread quantum: 32.0 ms (currently active
  and acquired by G_Kernel.exe).
- 12500 [ 1.2500 ms]
- 25000 [ 2.5000 ms]
- 50000 [ 5.0000 ms], thread quantum: 35.0 ms.
- 100000 [10.0000 ms], thread quantum: 40.0 ms.
- Lower resolutions are currently unavailable for detection.
```

Note: The thread quantum (time slice) values are only available on systems with more than one logical processor and only after the associated timer resolutions have been active. Another process/thread may prevent to scan all timer

resolutions by acquiring them. This results in *"Lower resolutions are currently unavailable for detection".*

- Notes:
  - The Windows 8 / 8.1 internal system time adjustment threshold is +/- 2 seconds. NTP offsets exceeding +/- 2 seconds will cause an instantaneous setting of the system time according to the NTP offset. No adjustment will take place for such big deviations. As a result, the maximum observed adjustment gain Windows applies is approx. 350 ppm. The contribution of such a gain to the error of the current version of **GetSystemTimePreciseAsFileTime** is: 350 µs/s = 0.35 µs/ms or approx. 5.5 µs in a 15.6250 ms slice. This considerable error has to be taken into account when using **GetSystemTimePreciseAsFileTime** while a system time adjustment is active. The G_Kernel time service does not use the function **GetSystemTimePreciseAsFileTime**.
- Supplements and fixes:
  - *force_jump = 0* was allowed nonsense. Fixed, now rejected as invalid.
  - NTP startup now shows the active update period.
  - *SetTimedEvent* periodic event was facing a limitation *"being in fast mode for more than 10 sec"* on multicore platforms. This error condition only applies to single processor non-hyperthreading platforms, fixed.
  - Invalid DumpMessages termination fixed.
  - Console mode on Windows 8 fixed.
  - Analyze_FFT stuck on specific platforms, fixed.
  - Invalid BatchTimer transit time on single core, fixed.
  - *G_Sleep()* with large arguments e.g. 300 sec resulted in wrong delays, fixed.

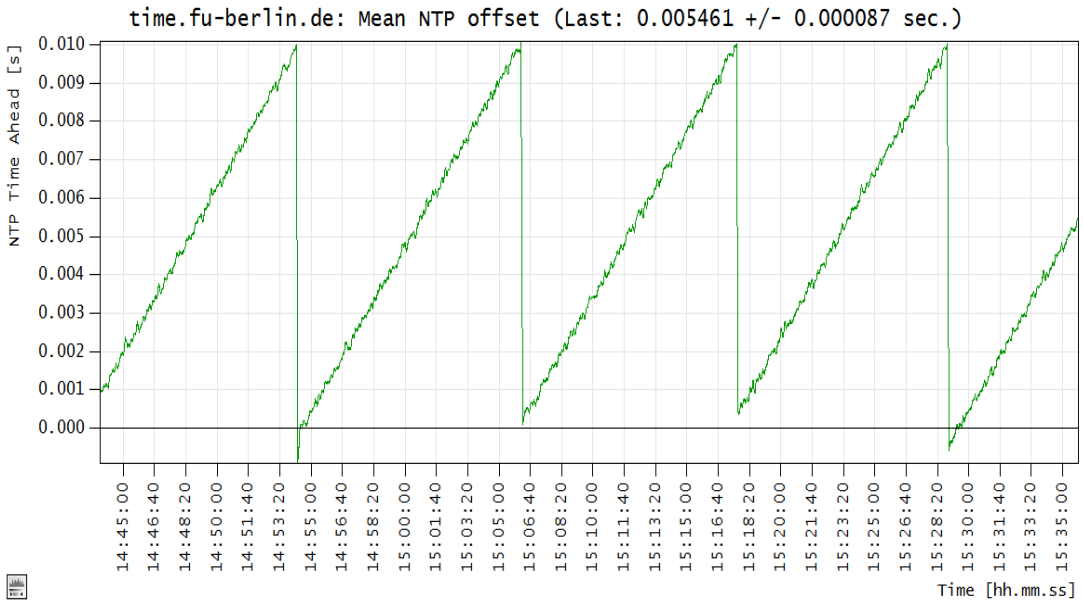
2014-04-03: Version 1.51 [Build 0151]

- New or modified G_kernel startup arguments:
  1. **max_ntp_restarts=n** allows to configure the maximum number of restarts of the NTP service. Restarts of the NTP service typically occur when an NTP server turns out to be unreliable. A value of n=0 will configure an endless amount of restarts. This may be advisable for unreliable networks. This value persists during the runtime.
  2. **force_jump[=threshold]** the force_jump now takes an optional threshold parameter in milliseconds. This configures the package to force an instantaneous system time adjustment whenever the NTP offset in milliseconds is beyond the threshold. The keyword *force_jump* may still be used without a threshold. This will force an instantaneous system time correction once with the first NTP read. The absolute value of the offset is compared to the threshold (e.g. a force_jump=10 will cause an instantaneous adjustment for offsets > 10 ms and for offsets < -10 ms). The threshold value persists during the runtime. Note: Mode autoadjust disables adjustments based on threshold.

     Remarks: The instantaneous synchronizations initiated by "*force_jump*",

"*force_jump=threshold*", or "*G_Setup sync_now*" are not as accurate as the system time adjustment done by *Autoadjust*. This reason is primarily the granularity of the system time. The effect is particularly noticeable on Windows XP platforms which do not change the system time granularity with different settings of the multimedia timer resolution. Only post XP versions are optionally operating at granularities in the 1 ms range. The minimum allowed threshold is 5 ms; however, using such small values of threshold on Windows XP may lead to continuous instantaneous adjustments because the instantaneous adjustment may not be capable to achieve an offset below 5 ms.

The following graph shows the offset of the system time to the time provided by an NTP server. The startup parameter force_jump=10 causes an instantaneous system time setting when the offset reaches the threshold of 10 ms.



NTP offset while operating with startup parameter force_jump=10 (Windows 7)

2014-03-22: Updated documentation

- Section *2.1.4.2. Desktop Applications:*

  GetSystemTimePreciseAsFileTime() of [*Microsecond Resolution Time Services for Windows*] sheds some light on the implications of a system time adjustment to the function *GetSystemTimePreciseAsFileTime()*.

- Associated pdf file updated.

2014-03-01: Version 1.50 [Build 0150]

- New G_kernel startup argument:

  1. **force_jump** forces a prompt adjustment of the system time to match the first NTP read after startup. This may produce a severe discontinuity in time. Particularly negative jumps in time can cause other software to get in trouble.

- Improved NTP outlier rejection while large system time adjustments are active.
- Altering the NTP period significantly during operation has caused the rejection algorithm to reject too many NTP reads for some time. This sometimes led to the error *"Max. number of consecutive rejections exceeded."*. Fixed.
- New G_Setup arguments:
  1. **exit** ends G_Kernel.exe. An alternative to the "Exit" button of the GUI.
  2. **sync_now** forces an instantaneous synchronization of system time and NTP time. This is only applicable with modes *NTP* and *Autoadjust* enabled and locked. Note: This is similar to the startup argument *force_jump*, note the comment given for *force_jump*. The accuracy of this synchronization is not as good as the accuracy achieved with *Autoadjust* because it is based on a single NTP capture.
- Proper termination of G_Kernel by either the GUI *Exit* button or the G_Setup *exit* keyword will not pop up windows with remaining messages any longer.
- Updated documentation.
- Note: As of February 2014, Microsoft has released a more detailed look into Windows timekeeping: "MSDN: Acquiring high-resolution time stamps."

2014-01-15: Version 1.40 [Build 0140]

- IPv6 support (*RFC 2460, Internet Protocol, Version 6 (IPv6) Specification*).

2013-12-18: Version 1.32 [Build 0132]

- Some fixes to G_Setup, DumpMessage, G_IO_Server, and location of license. G_Setup now correctly translates pool servers "official" names and responds friendly when executed from a non administrator console. G_IO_Server has undergone a major revision, particularly the console part was restructured. Internal messages will not appear in the message dump any longer. Performance/CPU occupancy reduced for G_IO_Server.

2013-11-30: Version 1.31 [Build 0131]

- NTP Version 4 (RFC 5905) is now the preferred NTP version. The package still supports NTP Version 3 (RFC 1305) and reverts to it when selected NTP servers don't support version 4.
- Higher accuracy with reduced CPU occupancy in autoadjust mode.
- G_Setup.exe tool added to the package. G_Setup.exe allows to asynchronously modify operational parameters of the package from the console.

  G_Setup Version 1.31 usage:

  Arguments:

  1. *"help"* this text

2. *"query"* queries some current NTP parameters.
3. *"ntp=pool.ntp.org"* sets the NTP server host.
4. *"ntp=176.74.176.179"* sets the NTP server IP (e.g. IP of www.time.windows.com).
5. *"period=2300"*sets the NTP update period in ms (supported periods: 250 ms to 300000 ms).
6. *"mode=off"* sets the mode of operation(Supported modes: off, monitor, autoadjust).

Examples:

1. *"G_Setup query"* queries the current NTP configuration.
2. *"G_Setup ntp=time.windows.com period=2500 mode=monitor"* establishes NTP monitoring with "time.windows.com" at an update period of 2500 ms.
3. *"G_Setup ntp=176.74.176.179 period=250 mode=autoadjust"* establishes NTP autoadjusting with "176.74.176.179" at an update period of 250 ms.
4. *"G_Setup mode=off"* disables the NTP functionality.

Remarks:
Arguments may be supplied in any order in upper or lower case letters. Parameters are updated asynchronously and may not take effect immediately. Operational details shall be looked at in the GUI or the log file. Arguments are processed in the supplied order, multiple occurrences of arguments are allowed.

- Update on NTP_Query function (introduced with version 1.22) return values: If the function succeeds, the return value is nonzero. If the function fails, the return value is zero. To get extended error information, call GetLastError.
- The dynamic link library (DLL) provides a subset of functions of the static library. New functions wrapped into the DLL:

1. `void DumpMessages(LPSTR lpMsg, int DumpFlag);`

   with *DumpFlag* taking on of the two values:

   ```
   #define DUMP_FLAG_CONSOLE1 << 0
   #define DUMP_FLAG_WINDOW1  << 1
   ```

   and *lpMsg* as a pointer to a string identifying the calling process.

2. ```
   int GetMMTimerResolution(
       MMtimerResolution_TYPE * MMtimerResolution);
   ```

   This function updates the content of the MMtimerResolution structure:

   ```
   typedef struct MMtimerResolution_TYPE {
     DWORD MMminRes;       // minimum timer resolution
     DWORD MMmaxRes;       // maximum timer resolution
     DWORD MMcurRes;       // current timer resolution
   } MMtimerResolution_TYPE;
   ```
   ```

- Functions supported by the dynamic link library have the following name convention: Function names are extended by an underscore ("_"). Example: The static function *Time()* can be loaded from the DLL by specifying the load name *Time_()*.

2013-10-30: Version 1.30 [Build 0130]

- Windows 8.1 / Server 2012 R2 compatibility.

  Windows 8.1 and its server variant have undergone modifications of the timekeeping. The timekeeping granularity has returned to 15.625 ms for specific platforms (*TimeIncrement* of the function GetSystemTimeAdjustment() and *MinimumResolution* of the function NtQueryTimerResolution()). However, the system time adjustment is not bound to this granularity since Windows 8.0.

- **csv** and **log** files are shared files. They may be visited during operation (write protected). The **csv** is overridden with each start of the suite.
- Local drift estimates are only provided for NTP update periods larger than twice the *BestTime* period evaluated by G_Kernel.exe. (Look for "*BestTimerPeriod: BestTime*" in the log file.) The estimate can typically be forced by a period > 2300 ms.
- Updated documentation: *Microsecond Resolution Time Services for Windows* and *Part II: Adjustment of System Time*.
- Associated pdf files updated.

2013-10-01: Version 1.22 [Build 0122]

- New or modified library functions:

  1. `DWORD G_Sleep(LONGLONG Delay);`

     is now also available with the dynamic link library (DLL). See news 2013-06-07 for details.

  2. A new function to query NTP details:

     ```
     int NTP_Query(NTP_Query_TYPE * ntp_query);
     ```

     This function updates the content of the ntp_query structure:

     ```
     typedef struct NTP_Query_TYPE {
       DWORD msg_status;    // optional error status
       DWORD status;        // NTP_STATUS_INACTIVE, NTP_STATUS_GATHERING,
                            // or NTP_STATUS_ACTIVE
       DWORD mode;          // NTP_MODE_OFF, NTP_MODE_MONITOR, or
                            // NTP_MODE_AUTOADJUST
       DWORD update_period; // current update period in ms
       char host[MAX_PATH]; // host URL, e.g. "time.windows.com"
       char ip[MAX_PATH];   // host IP as dotted notation string, e.g.
                            // "178.23.124.2"
       BOOL locked;         // the past three consecutive NTP
                            // synchronizations were successful
       BOOL wsa_up;         // WinSock DLL was initiated
       double offset;       // current NTP offset
       double mean_offset;  // mean NTP offset
     ```

```
    double stddev;       // standard deviation of mean offset
    long long offset_timestamp;// timestamp of last NTP
                         // synchronization
} NTP_Query_TYPE;
```

NTP_Query asynchronously queries current NTP service parameters.

- New G_kernel startup arguments:

  1. **csv** dumps a csv file containing the collected filetime transitions during the initial analysis. Files are stored in /csv directory where the package resides. This directory is created automatically if it does not exist. Previous versions established csv as default startup configuration.
  2. **nolog** disables the output to a log file.

  Example:

  ```
  G_Kernel.exe silent nolog ntp=pool.ntp.org period=2500 autoadjust
  ```

  This example starts the package without any output or log file, synchronizing the system time to an NTP server with an update period of 2.5 seconds.

2013-08-06: Version 1.21 [Build 0121]

- Windows 8 / Server 2012 compatibility.
- Updated documentation: *Microsecond Resolution Time Services for Windows*.
- Updated documentation: *Part II: Adjustment of System Time*.
- Associated pdf files updated.

2013-07-05: Microsoft Visual C++ Redistribution Package

- The Microsoft Visual C++ 2010 Redistributable Package *vcredist_x86.exe* has been added to the zip archive. It may alternatively also be downloaded from *Microsoft Visual C++ 2010 SP1 Redistributable Package (x86)*.

2013-06-28: Updated pdf files

- Links activated in pdf documents.

2013-06-07: Part II: Adjustment of System Time

- The documentation has been extended by a detailed description on system time adjustments. See the second part of the description: *Part II: Adjustment of System Time*.

2013-06-07: Version 1.2 [Build 0120]

- Build in NTP client functionality. The GUI provides a checkbox to enable/disable NTP monitoring.
- New *NTP Offset* tab to plot the offset of the NTP server time vs. the local time.

- Full support of system time changes and the ability to perform system time adjustments. The system time is optionally adjusted automatically by a new "autoadjust" function. A checkbox has been added to the GUI to establish access to this new feature.

- The corrected performance counter frequency is now shown in the *Calibrated Performance Counter Frequency Offset* tab. The values are now normalized to the default performance counter frequency (given by QueryPerformanceFrequency()). The offset (correction) is given in ppm to ease the interpretability. One ppm deviation of the performance counter frequency results in a deviation of 1 µs/s.
- New or modified library functions:
  1. `SetTimedEvent(...)`

     has received a limitation for its argument *TimerPeriod* for single core platforms: TimerPeriod now needs be at least 2 times the *ActualResolution* returned by the function NtQueryTimerResolution. Smaller values of *TimerPeriod* are rejected as invalid argument on single core platforms.

  2. A new wait function for timed events:

     ```
     DWORD WaitForSingleTimedEvent(    HANDLE hEvent,
                                       DWORD dwMilliseconds,
                                       long long * pTimeNow = NULL);
     ```

     WaitForSingleTimedEvent executes the WaitForSingleObject function in a capsule with raised priority. It is up to the user to alternatively also use WaitForSingleObject but it is not guaranteed that WaitForSingleObject returns in timely manner when other threads of equal or higher priority are running. However, *pTimeNow* is set within the capsule and therefore represents the actual time at which the event occurred. Additionally the processor affinity is adjusted inside the capsule to obtain best results. The static library also contains a function MsgWaitForMultipleObjectsOrTimedEvents which refers to the function MsgWaitForMultipleObjects in a similar way. (See G_User.c in the samples directory for more details on how to use the WaitForSingleTimedEvent function.)

  3. Access to the new functionality of NTP services and autoadjust are provided by the function

     ```
     int NTP_Setup(   LPCTSTR lpNtpHostName,
                      DWORD dwNTPPeriod,
                      DWORD dwNTPMode);
     ```

     NTP_Setup is an asynchronous function which updates the parameters of the NTP/Autoadjust functionality. *lpNtpHostName* specifies the host name of the NTP server (e.g. "time.windows.com"). The function also takes an ASCII string in internet standard dotted-decimal format as returned by inet_ntoa (e.g. "192.168.16.0") as "host name". *dwNTPPeriod* sets the update period of the NTP service. Currently the range between 0.25 s to 300 s (250 - 300000) is supported. Values outside this range are rejected and reported as invalid arguments.

     NTP_Setup currently supports three modes of operation:

- NTP_MODE_OFF (1) disables all NTP services.
- NTP_MODE_MONITOR (2) enables monitoring of an NTP server.
- NTP_MODE_AUTOADJUST (3) enables monitoring of an NTP server and auto adjusts the local system time.

The function will return TRUE upon successful update of the parameters.

Example:

```
if (!NTP_Setup("time.windows.com",350,NTP_MODE_AUTOADJUST))
HandleTheNTP_SetupError();
```

Such a call to NTP_Setup will establish a 350 milliseconds update of the NTP server time of one of the servers in the "time.windows.com" NTP server pool and will start the continuous adjustment of the local clock.

4. A shortcut to a more complex task: The G_Sleep function.

   ```
   DWORD G_Sleep(LONGLONG Delay);
   ```

   This function accepts *Delay* in 100 ns units. At this stage *Delay* is expected to have a positive value. *Delay* may also be 0. Unlike the common Windows Sleep() function, G_Sleep(0) will NOT relinquish the reminder of the threads time slice to other threads. A successful call to G_Sleep will return TRUE.

   Remarks: The G_Sleep service is initialized once per process with the first call. Therefore it is advised to make a call (e.g. G_Sleep(0)) ahead of any time critical task. G_Sleep can only conquer the sub-milliseconds range by means of busy waits. However, the system wide structure allows to only having one busy section for all executing G_Sleep instances and this section is only busy for at most the millisecond ahead of the expiration of the desired delay. Multicore platforms don't show any significant performance drawbacks. A careful use of G_Sleep is advised on single core platforms. Consecutive calls to G_Sleep with delays less than 2 times the *ActualResolution* returned by the function NtQueryTimerResolution (approx. 2-3 ms) may put the hardware in a close to 100% busy state at high priority. As a consequence, the user interface may get less responsive.

- New G_kernel startup arguments:

  1. **ntp[=*Hostname/Address*]** starts the monitoring of the network time. The *Hostname* or the *Address* as an ASCII string in internet standard dotted-decimal format (e.g. "192.168.16.0") can be supplied to the ntp keyword to select a specific network time provider. The NTP monitoring starts at a default period of 5000 ms.
  2. **period=*UpdatePeriod*** The period keyword allows to select the NTP *UpdatePeriod* within 250 ms to 300000 ms (300 sec.). The default period is 5000 ms.
  3. **autoadjust** starts the service with the autoadjustment enabled. Note: The autoadjust keyword must be accompanied by the ntp keyword.

     Example:

     ```
     G_Kernel.exe gui ntp=pool.ntp.org period=250 autoadjust
     ```

This example starts with a single GUI instance, updating the network time from one of the servers in the pool of pool.ntp.org at a rate of 4 updates per second with autoadjustment enabled.

2012-12-22: Version 1.1 [Build 0110]

- Improved support of dynamic system time changes. A continuous applied system time change e.g. done by NTP (Network Time Protocol) or the operating system is now treated as a temporarily change of the performance counter frequency. A system time change temporarily modifies the rate of progress of time. The time interval is also the reference for the value of the performance counter frequency. Therefore its value changes accordingly and allows proper high resolution timing also when a system time change is applied.

...