東京大学地震研究所技術研究報告，No. 31，1–12 頁，2025 年.
Technical Research Report, Earthquake Research Institute, The University of Tokyo, No. 31, pp. 1–12, 2025.

論　説

MQTT 対応 WIN によるリアルタイム表示システムの開発

鶴　岡　　弘*†

Development of real-time monitoring system using WIN system with MQTT support

Hiroshi TSURUOKA*†

Abstract

The recent introduction of the raw2mq and mq2raw programs into the WIN system has enabled very simple real-time monitoring of seismic waveform data transmitted in WIN format. These programs also reduce server-side load compared to previous mechanisms and simplify system configuration. Moreover, they allow more stable data transmission and reception with fewer data gaps. It was also demonstrated that real-time analysis and other processing can be performed using only the MQTT package, without relying on the WIN system. Going forward, continued exploration will be necessary to further leverage these programs and adapt them to next-generation data distribution systems.

Key words : WIN system, MQTT, JavaScript, Real-time monitor, Real-time analysis, Data transfer

は　じ　め　に

WIN システム（東京大学地震研究所日本列島モニタリング研究センターウェブサイト）は，マルチチャンネルの地震波形データを取り扱うための処理システムで，UNIX 上で動作する多くのプログラム群から構成されている．日本におけるデータ流通のデファクトスタンダードシステムであると同時に現在においてもなお開発が継続していることも重要である．WIN システムにおいては，WIN フォーマットでのデータ送受信がなされているが，そのフォーマットをリアルタイムに表示できるツールは WIN システムのプログラム群の中に同梱されている．MQTT プロトコルを用いた WIN システム用リアルタイム波形表示システムの開発（鶴岡，2022）では，Web により WIN フォーマットで配信される地震波形データをリアルタイムに表示できることが示された．MQTT は，機械同士が通信を介して情報をやり取りする M2M や家電，自動車など多種多様なモノがインターネットにつながりお互いに情報をやりとりする IoT においてよく利用されるプロトコルであり，シンプル，軽量，省電力という特徴を持っている．なお，このシステムをこれ以後 winq と呼ぶ．ただし，鶴岡（2022）においては，主にクライアント側での負荷に着目しており，サーバ側の負荷やシステムセットアップについてはあまり言及がされていなかった．WIN システム開発においては，最近，WIN システムの開発者である卜部により raw2mq および mq2raw という 2 つのプログラムが追加されており，これらのプログラムを利用することにより，winq において，サーバ側の負荷を軽減できるとともにシステムのセットアップも容易となる．本稿ではそれらを含め，これらのプログラムを利用することにより，データの送受信をより容易にかつ安定してできることも紹介する．さらに，これらのプログラムをリアルタイム解析に活用する方法についても簡単に紹介する．

2025 年 10 月 24 日受付，2025 年 12 月 19 日受理.
† tsuru@eri.u-tokyo.ac.jp
* 東京大学地震研究所日本列島モニタリング研究センター
* Research Center for monitoring Japan Arc, Earthquake Research Institute, The University of Tokyo

winq について

winq において，サーバ側における起動コマンドや Web における配置ファイル等の詳細をまずは説明する．構成としては，クライアント（Web ブラウザ）から Web サーバに接続し，WebSocket を通して MQTT ブローカーに配信される地震波形データにアクセスすることになる．MQTT ブローカー（mosquitto により実装）へ WIN フォーマットの地震波形データを送信するコマンドの例は，

```
% shmdump -tq 11 0252 | wintojson.pl |
    mosquitto_pub -t j/0252 -l -h localhost          (1)
```

となる．ここで 11 は共有メモリの key, 0252 は WIN のチャネル ID で，チャネル毎に shmdump，wintojson.pl（鶴岡

(2022) の表 2 参照) と mosquitto_pub のコマンドが起動されるので，配信したいチャネル数が増えるほどサーバ側のプロセス負荷が線形に増大することになるとともに，これらのプロセスをチャネル毎に起動することは手間でもある．また，Web サーバでは httpd (apache) が起動し，クライアントからのアクセスに対して応答するので，サーバ側の負荷としてカウントすることになる．(1)については，この後に述べる raw2mq コマンドを利用することにより非常に簡単かつ応用が効く形で MQTT ブローカーに WIN フォーマットの地震波形データを送信できるようになる．

raw2mq および mq2raw について

WIN パッケージ v3.0.17 (公開日 2025/7/30) において，raw2mq と mq2raw なる 2 つのプログラムが追加されている．このプログラムは共有メモリに書き込まれている WIN フォーマットデータと MQTT ブローカー間を相互にやり取りするためのプログラムである．チャネルが複数ある場合には，winq においてはチャネル分の (1) のコマンドの起動が必要となるが，raw2mq により

```
% raw2mq -t win -j -n -L ./channels.tbl 12 localhost
  raw2mq.log                                          (2)
```

とするだけで共有メモリに書き込まれているチャネルがまとめて MQTT ブローカーに配信される．(2) における数値 12 は共有メモリの key である．ここで，channels.tbl は複数観測点のチャネル情報が記載された WIN システムにおけるチャネルテーブルで，オプションが -l の場合にはデジタル値で，-L の場合にはこのテーブルに従った物理量値で MQTT ブローカーに配信される．例えば，配信するチャネルが 100 ある場合には (1) の場合はプロセスを 300 起動させる必要があるが，(2) の場合は一つで済むことになる．比較のため，73 チャネルを配信する場合のサーバ側の負荷を 4GB のメモリを搭載した Raspberry Pi 4 Model B 機器において測定した．実際に測定に使ったコマンドは，

```
% top -bn10 -d10| grep Cpu | awk '{printf("%f\n",100-$8);}'
                                                      (3)
```

でこの数値の平均値を負荷とした．(1) の場合は 4.4 で (2) の場合は 0.8 となり，(2) の場合にはサーバの負荷が 1/5 以下となった．チャネル数が多くなるほどこの差が大きくなるので raw2mq コマンドの利用はサーバ側の負荷の軽減につながる．mq2raw については，MQTT ブローカーにある WIN フォーマットデータ (バイナリ) を共有メモリに書き込めるほかバイナリをそのまま標準出力に出力する機能と WIN のテキスト形式で出力する機能がある．コマンドオプションは mq2raw および raw2mq のどちらにおいても豊富に存在する．そのため，raw2mq と mq2raw のマニュアルについてはオンラインマニュアルを参照せずに確認できるように Appendix に載せた．

リアルタイム波形モニタリング

winq において，MQTT ブローカーを活用することにより簡便に WIN フォーマットで配信されている地震波形データをリアルタイム表示することが示されたが，今回はさらにその仕組みが raw2mq プログラムを利用することにより大幅に簡素化されることになる．実際に波形を表示するクライアントとそのためのサーバの概略図と必要なファイル群と実行すべきコマンドを図 1 に示す．リアルタイム波形表示には，WIN システムおよび JavaScript のライブラリである smoothie.js，WebSocket によるリアルタイムデータ受信のためのライブラリの mqttws31.js，smoothie.js へのパラメータ設定のための winq.js および実際の波形表示のための winq.html を参照する．なお，winq.js/winq.html は，https:// wwweic.eri.u-tokyo.ac.jp/repository/winq/ から参照できる．また，winq.html における波形モニター表示においてはオプションを多数設けており，これらにより柔軟なモニター表示が可能である．以下にオプションとその機能を説明する．

- NAME (表示名)

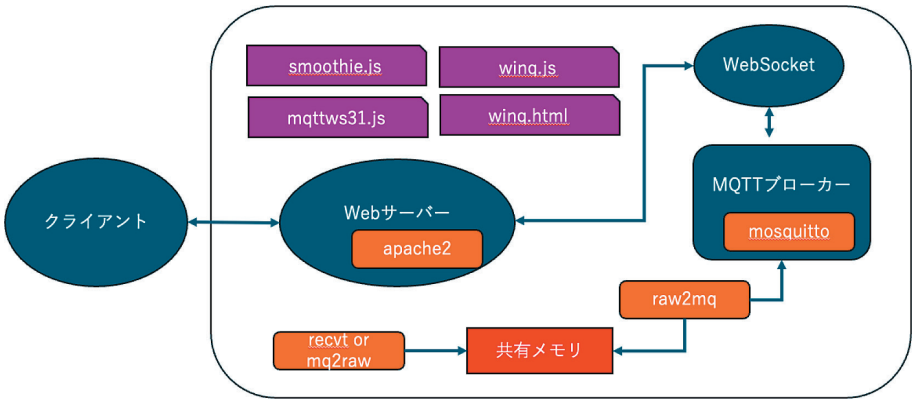

図 1. リアルタイム波形表示システム概念図．アプリケーションをオレンジで，ファイルを紫表示としている．

- CH（表示するチャネル ID あるいはチャネルのトピック名）
- SEC（波形を表示する全体の時間長 : 単位秒）
- GSEC（SEC の長さを GSEC で分割するグリッドラインの時間長）
- WIDTH（iframe におけるウィンドウの幅）
- HEIGHT（iframe におけるウィンドウの高さ）
- MIN/MAX（縦軸の最小と最大，これが設定されない場合には auto スケールで表示される）
- YGRID（縦軸の分割グリッド数）
- COL（表示色，数値として 0–15 が指定可能）
- FPS（表示におけるフレームレート）

なお，オプションには標準値を設定しており，すべてのオプションを指定する必要はなく，HTML ソースをなるべく省力化できるようになっている．

実　際　例

実際の Web におけるリアルタイム表示とその HTML ファイルの内容をそれぞれ図 2 と表 1 に示した．また，地

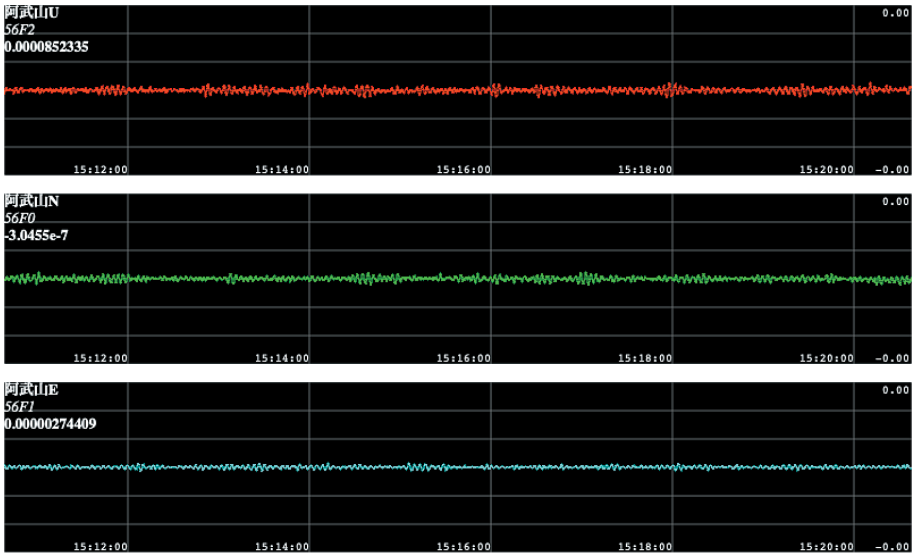

図 2.　リアルタイム表示例．阿武山観測点からの 3 成分の波形表示となっている．

表 1.　html ソース（monitor.html）

```
<head>
<meta HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=UTF-8">
<title>Win Monitor</title>
<style>
iframe {
border: none;
}
</style>
</head>
<body>
<iframe src="winq.html?NAME=阿武山U&CH=56F2&MIN=-1.0e-5&MAX=1.0e-
5&FPS=0&SEC=600&COL=2&WIDTH=800&HEIGHT=150" name="win" width="800"
height="150"></iframe><p>
<iframe src="winq.html?NAME=阿武山N&CH=56F0&MIN=-1.0e-5&MAX=1.0e-
5&FPS=0&SEC=600&COL=4&WIDTH=800&HEIGHT=150" name="win" width="800"
height="150"></iframe><p>
<iframe src="winq.html?NAME=阿武山E&CH=56F1&MIN=-1.0e-5&MAX=1.0e-
5&FPS=0&SEC=600&COL=5&WIDTH=800&HEIGHT=150" name="win" width="800"
height="150"> </iframe>
</body>
</html>
```

震研究所 100 周年にあたり，地震研究所 1 号館 2F ラウンジに地震活動モニタリング展示を今回のリアルタイム表示システムを活用して構築した．図 3 のスナップショットに示されるように簡単に複数観測点からの地震波形データをリアルタイムでモニターすることが実現されている．

波形データにおける送受信 およびリアルタイム解析への活用

これまでの WIN システムにおける波形データの送受信は，図 4 にあるように，send_raw および recvt を使用している．send_raw および recvt は UDP プロトコルによりデータの送受信をしており，セッションを維持していないために再送のための仕組みがあるものの，ネットワーク帯域が十分である場合には問題がないが，そうでない場合には欠測につながるということがあり，自律協調型通信なる ACT プロトコル（森田他，2010）が開発されてきた．一方，raw2mq および mq2raw プログラムは図 5 に示したように MQTT ブローカーを介するが raw2mq は send_raw と同様に送信側，mq2raw は recvt と同様に受信側のプログラムとして置き換えることができる．MQTT ブローカーのプロトコルは TCP であるため，より安定した通信が可能である．また，MQTT ブローカーの利用は，通信部分のプログラム開発を IoT 等の技術を積極的に利用して分離するという方向性を示したことでもある．なお，

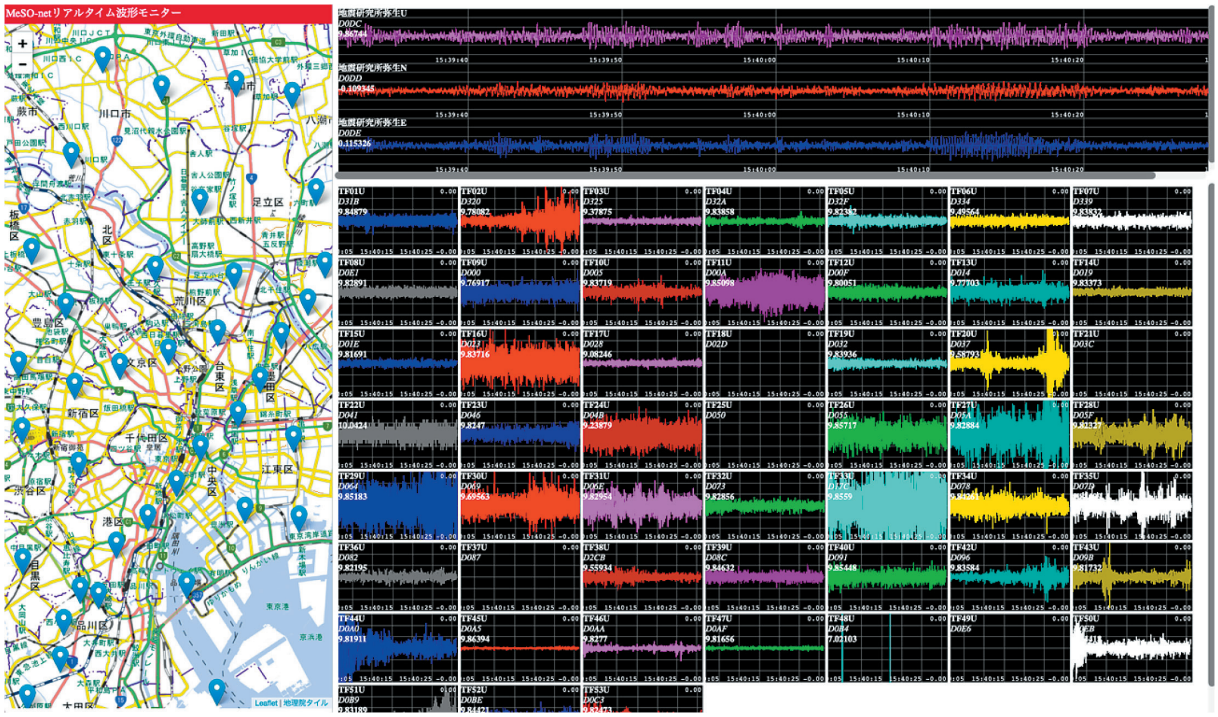

図 3.　地震活動モニタリングにおける複数観測点のリアルタイム波形表示例．首都圏に配置されている MeSO-net 観測網の波形表示となっている

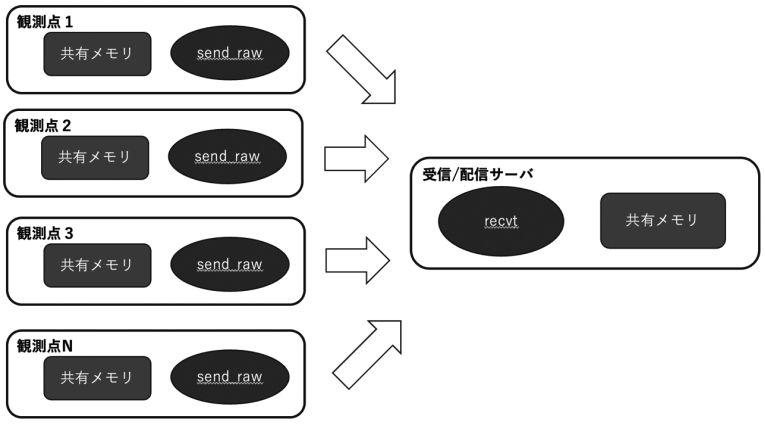

図 4.　これまでの WIN システムにおける送受信概念図．データの送信は send_raw，データの受信は recvt を起動する．

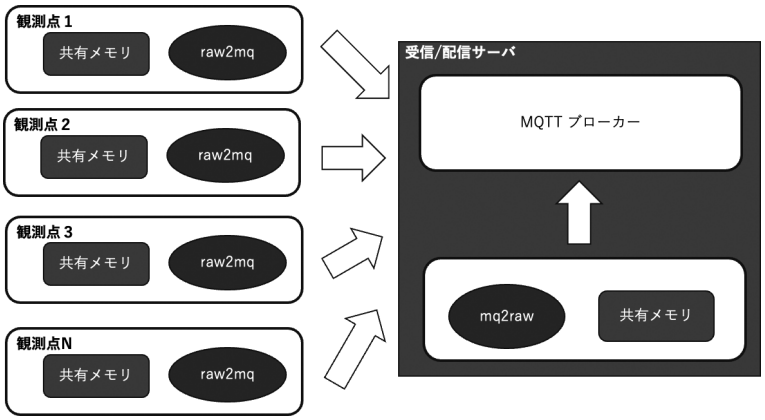

図 5. raw2mq および mq2raw を使った際の送受信概念図．データの送信は raw2mq，データの受信は mq2raw を起動する．

MQTT はパブリッシュ / サブスクライブモデルを採用しており，トピック毎に送受信データが分離されるという大きなメリットがある．さらに send_raw および recvt ではスイッチやネットワークインタフェースの相性問題等の回避のためのオプションが存在しており，どのオプションおよびパラメータ値を使うかについてデータ送受信の経験等も必要とされていた．一方, raw2mq/mq2raw については，MQTT ブローカーに接続するための手続きのためのオプション等が必要になるものの実質的には共有メモリキーと共有メモリサイズのみを設定すればよく，より使いやすくなったと言える．また，MQTT ブローカーをネットワーク上の任意に配置できるため，用途に応じて柔軟なネットワーク構成ができるようになったこともこれらのプログラムを利用する利点であると思われる．

データ流通におけるチャネル管理においては，データを複数機関において共有する場合にはチャネル ID をユニークにつけるということが必要となり，また，WIN システムにおいては波形フォーマットの制約によりチャネル ID 数が 16bit（65536 まで具体的には 16 進数で 0X0000 から 0XFFFF まで）に制限されるという現実がある．一方，建物強震観測においては，被災度判定等は建物毎にデータが収集できればよいということがあり，建物毎であればチャネル ID を同一にできるとシステムの構築が容易になる場合がある．この実現方法としては，これまでの send_raw/recvt の場合には，送信する UDP のポートを変更するなどが考えられるが，ファイアウォール等の設定をネットワークの管理部門に依頼するケースなどでは設定や設定変更等に時間を要する場合が考えられる．一方，raw2mq/mq2raw の場合には，MQTT ブローカーに送るトピック名を変更するだけでこれが実現できる．また，観測点のステータス情報などを別の UDP ポートに送り，監視しているが，raw2mq/mq2raw であれば，トピック名を別で設定すれば同じポートに送ることができ，統一したデータ送受信が可能となる．

最後に, リアルタイム解析のプラットフォームとしては，WIN システムでは shmdump コマンドにおける WIN テキストフォーマットによる標準出力機能を使うということで実現できる．ただし, 出力は，デジタル値であるため，チャネルテーブルを参照して物理値に変換する部分が必須となっていた．一方，raw2mq により物理値として MQTT ブローカーに WIN テキストフォーマットあるいは WIN json フォーマットで配信すれば，WIN システムをインストールしていない端末でも，MQTT におけるサブスクライブコマンド等（MQTT ブローカー毎に異なるが Mosquitto の場合には mosquitto_sub コマンド）により，リアルタイムでの出力を取得することでリアルタイム解析が可能となる．例えば，3 チャネル分（チャネル ID は 0252，0253，0254）の WIN テキストフォーマットのリアルタイムデータを受信するサンプルは

% mosquito_sub -u anonymous -P readonly -t win/a/
0252 -t win/a/0253 -t win/a/0254 -h localhost. (4)

となる．

結論および今後の展望

WIN システムに新たに追加された raw2mq および mq2raw プログラム，MQTT プロトコルの標準的な実装である mosquitto，リアルタイムにデータを表示する JavaScript のライブラリである smoothie.js および WebScoket により MQTT ブローカーに接続する mqttws31.js を活用して WIN フォーマットに対応したリアルタイム波形表示システムを開発した．また，これらのプログラムがこれまでの recvt/send_raw に変わりデータの送受信をより容易にかつ安定して実現できることについても説明した．また，MQTT におけるトピックを活用することにより MQTT ブローカー内においては WIN フォーマットにおけるチャネル制限を回避できることも示した．今後においては，MQTT ブローカーのさらなる活用が可能となる機能強化（例えば，wdisk に代わる波形データの

保存プログラムの開発）を図っていくことが重要と考えている．

謝　辞：技術研究報告編集委員会と 2 名の匿名査読者には，有益なご指摘を頂きました．ここに記して感謝申し上げます．

文　　献

東京大学地震研究所日本列島モニタリング研究センターウェブサイト，WIN システム https://wwweic.eri.u-tokyo.ac.jp/WIN/.（参照 2025-10-10）

Eclipse Mosquitto ウェブサイト，Mosquito，https://mosquitto.org.（参照 2025-10-10）

Smoothie Charts ウェブサイト，http://smoothiecharts.org.（参照 2025-10-10）

鶴岡　弘，2003，WIN システム用波形モニターツールの活用，東京大学地震研究所技術研究報告，9，14-19.

森田裕一・酒井慎一・中川茂樹・笠原敬司・平田　直・鏡　弘道・加藤拓弥・佐藤峰司，2010，首都圏地震観測網（MeSO-net）のデータ伝送方式について―自律協調型データ送信手順（ACT protocol）の開発―，東京大学地震研究所彙報，84，89-105.

鶴岡　弘，2022，MQTT プロトコルを用いた WIN システム用リアルタイム波形表示システムの開発，東京大学地震研究所技術研究報告，28，1-5.

Appendix

raw2mq(1W)

名称

raw2mq - RAW 形式データを MQTT ブローカーへ送信

形式

raw2mq [-agjnsD] [-t topic] [-p port] [-q qos] [-i id] [-c cafile] [-d capath] [-C certfile] [-k keyfile] [-u user] [-P pswd] [-l/-L chfile] inkey host [logfile]

解説

raw2mq はホスト host の MQTT ブローカーに接続し、キー inkey の共有メモリ・セグメントに巡回的に書き込まれている WIN 形式のデータを送信 (publish) します。 MQTT メッセージのトピックはデフォルトで "win/data" です。共有メモリ inkey 上の形式は、時間順整列の有無・書き込み時刻の有無・ ブロック末尾のブロック長の有無、のいずれも問いません。 共有メモリ上の形式については order(1W), recvt(1W) を参照してください。共有メモリ上の 1 ブロックからブロック長情報と書き込み時刻情報を 除いた部分 （すなわち通常はタイムスタンプと波形データからなる 1 秒分）が 1 つの MQTT メッセージのペイロードとして送信されます （ただしオプション -a または -j が指定された場合は下記参照)。

ログファイル名 logfile を指定すると、ここに動作ログがとられ、指定しないとログ情報は標準出力に 送られます(ただし、daemon モードで動いている時は syslogd(8) に送られます)。 ログファイルは書き込みのたび毎にオープン／クローズされます。

raw2mq は、HUP シグナルを受けると ログファイル logfile に流量情報を書き出します。これには起動時、または前回 HUP シグナルを受けたときからの、メッセージ数、バイト数、毎秒メッセージ数、毎秒バイト数が含まれます。

raw2mq は特定の WIN チャネル番号のデータのみを選別して送信することは できません（ただしオプション -a または -j が -l または -L と同時に指定された場合は下記参照)。 送信前にチャネル番号で選別するには、raw_raw(1W) 等が使えます。

raw2mq は、引数なしで起動すると簡単な使用法を表示します。

オプション

-s SSL/TLS 化されたブローカーに接続します。

-t topic

送信する MQTT メッセージのトピックを topic に設定します。トピックは任意の文字列で、'/'で区切って階層化することができます。 トピックの詳細については MQTT の資料を見てください。

-p port

MQTT ブローカー host の TCP ポート番号を port に指定します。デフォルトは 1883 です。ただしオプション -s で SSL/TLS 化した場合のデフォルトは 8883 です。

-q qos

MQTT ブローカーとの間の通信品質(QoS)を qos に設定します。設定できる値は 0,1,2 のいずれかで、デフォルトは 0 です。 QoS の意味については MQTT の資料を見てください。

-a ASCII 形式で出力します。このときデータは 1 秒×1 チャネル毎の MQTT メッセージとして、 チャネル毎に別々のトピック名がついて、 shmdump(1W) -tq の形式で出力されます。 このデータを MQTT クライアントで受信(subscript)してパイプで shmx(1W) に入力することにより、 望む 1 つまたは複数のチャネルのリアルタイム波形表示をすることができます。 このときのトピック名は topic ではなく、topic/a/CHID （CHID は 16 進数 4 桁のチャネル番号） になります。この場合に限り、 topic は空文字列("")または空白(" ")でもよく、そのときトピック名は a/CHID になります。 -a オプションは-j オプションと同時に指定することができます。

-j JSON 形式で出力します。 このときデータは 1 秒分×1 チャネル毎の MQTT メッセージとして、 チャネル毎に別々のトピック名を持ち、 winmonitor.js / smoothie.js によるリアルタイム波形表示システムで受信(subscript)表示することができる JSON 形式です。 このときのトピック名は topic ではなく、 topic/j/CHID （CHID は 16 進数 4 桁のチャネル番号）になります。この場合に限り、 topic は空文字列("")または空白(" ")でもよく、そのときトピック名は j/CHID になります。 -j オプションは-a オプションと同時に指定することができます。

-u user

-P pswd

MQTT ブローカーがユーザー認証を必要とする場合、ユーザー名 user とパスワード pswd をそれぞれ設定します。

-l chfile

-j または-a オプションと共に指定されたとき、 WIN 形式のチャネル表ファイル chfile を読んで、その中にあるチャネルだけを出力します。 チャネル表ファイルの形式については win(1W) を参照してください。 出力の際のトピック名はチャネル毎に topic/j/STN/CMP または topic/a/STN/CMP 等になり（STN はチャネル表から得た観測点コード、CMP は同じく成分名）、 データ中の CHID 部分も同様に STN/CMP になります。

-L chfile

オプション-l とほぼ同じですが、 出力されるのはチャネル表に従って物理量に変換された後の値です。またトピック名はチャネル毎に topic/J/STN/CMP または topic/A/STN/CMP 等になり、データ中の CHID 部分も同様に STN/CMP になります。

-n オプション-l または-L が指定されている場合でも、トピック名の STN/CMP の部分を CHID にします。

-c cafile

-d capath

SSL/TLS 対応のブローカーの場合、CA(認証局)証明書である PEM 形式のファイル cafile またはそれを内容として持つディレクトリ capath のいずれかを指定します。 デフォルトでそれぞれ /etc/ssl/certs と/etc/ssl/cert.pem が使用され (試され)ますが、これらがいずれもが有効ではない場合は指定が必要です。

-C certfile

-k keyfile

SSL/TLS 対応のブローカーがクライアント認証を必要とする場合、 PEM 形式の証明書ファイル certfile と PEM 形式の秘密鍵ファイル keyfile の２つをそれぞれ指定します。

-D daemon モードで起動します。

-g デバッグ情報を出力します。

使用例

キー11 の共有メモリに流れているデータを、ASCII 形式で、 チャネルごとに別々のトピックで、ホスト host の MQTT ブローカーへ 送信(publish)します。トピックとして "" を指定したので、 トピック名は a/CHID になります。MQTT のユーザー名 user、パスワード pass です。

```
raw2mq -aj -t "" -u user -P pass 11 host
```

送信されたデータのうち 9100,9101,9102 の３チャネルを MQTT クライアントで受信 (subscript)して、shmx(1W) でリアルタイム波形表示します。

```
mosquitto_sub -h host -u user -P pass -t a/9100 -t a/9101 -t a/9102 |￥
  shmx 9100 9101 9102
```

web ブラウザで下のような URL をアクセスすると winmonitor.js/smoothie.js による チャネル 9100 のリアルタイム波形表示が見えるかもしれません。

```
http://host/winq.html?CH=9100&FPS=0&SEC=10&COL=1&WIDTH=1000&HEIGHT=200
```

mq2raw(1W)

名称

mq2raw - MQTT ブローカーから RAW 形式データを受信

形式

mq2raw [-abgsD] [-t topic] [-p port] [-q qos] [-i id] [-c cafile] [-d capath] [-C certfile] [-k keyfile] [-u user] [-P pswd] host outkey shmsize [logfile]

mq2raw -a/-b [-gsD] [-t topic] [-p port] [-q qos] [-i id] [-c cafile] [-d capath] [-C certfile] [-k keyfile] [-u user] [-P pswd] host [logfile] (-a/-b を指定したときは outkey と shmsize を省略可)

mq2rawd [-abgs] [-t topic] [-p port] [-q qos] [-i id] [-c cafile] [-d capath] [-C certfile] [-k keyfile] [-u user] [-P pswd] host outkey shmsize [logfile]

mq2rawd -a/-b [-gs] [-t topic] [-p port] [-q qos] [-i id] [-c cafile] [-d capath] [-C certfile] [-k keyfile] [-u user] [-P pswd] host [logfile] (-a/-b を指定したときは outkey と shmsize を省略可)

解説

mq2raw は、ホスト host の MQTT ブローカーに接続して WIN 形式データを受信(subscribe) し、 それをキー outkey で与えられる共有メモリ・セグメントに巡回的に書き込みます。 ただしオプション-a/-b を指定した場合には出力先が共有メモリではなく標準出力になります。 受信する MQTT メッセージのトピックはデフォルトで "win/data" です。

共有メモリ・キーは 32 ビットの整数値です。 outkey をもつ共有メモリ・セグメントが存在しない場合は、大きさ shmsize (KB) の共有メモリ・セグメントが作られます。すでに存在している場合は、 そのサイズが shmsize (KB) よりも小さいとエラーになります。 オプション-a/-b を指定した場合には outkey と shmsize は省略可能です。

ブローカーから受信された 1 メッセージのペイロードが 共有メモリ上の 1 ブロックになります。 その形式は、書き込み時刻付き・ブロック末尾のブロック長付きです。 共有メモリ上の形式については order(1W), recvt(1W) を 参照してください。

mq2rawd は daemon モードで起動します。

ログファイル名 logfile を指定すると、ここに動作ログがとられ、指定しないとログ情報は標準出力 (オプシ

ョン-a/-b を指定した場合に限り標準エラー出力） に送られます。ただし daemon モードで動いている時は syslogd(8) に送られます。 ログファイルは書き込みのたび毎にオープン／クローズされます。

mq2raw は、HUP シグナルを受けると ログファイル logfile に流量情報を書き出します。これには起動時、または 前回 HUP シグナルを受けたときからの、メッセージ数、バイト数、毎秒メッセージ数、毎秒バイト数が含まれます。

mq2raw は特定の WIN チャネル番号のデータのみを選別して受信することは できません。受信後にチャネル番号で選別するには、raw_raw(1W) 等が使えます。

mq2raw は、引数なしで起動すると簡単な使用法を表示します。

オプション

-a ASCII 形式（shmdump(1W) -tq の形式）で標準出力へ書き出します。 このデータをパイプで shmx(1W) に入力することにより、 望む１つまたは複数のチャネルのリアルタイム波形表示をすることができます。 -a が指定されたとき、２つ目と３つ目の引数 outkey と shmsize は省略可能で、それらを省略した場合は logfile がさらに省略可能な２つ目の引数になります。

-b バイナリ形式で標準出力へ書き出します。これは WIN フォーマットの 波形ファイルをそのままダンプ(cat)したのと同じで形式で、 shmdump(1W)が、入力共有メモリキーとして " - " を指定されたときに 期待する形式です。したがってこのデータをパイプで shmdump(1W)に 入力して利用することができます。 -b が指定されたとき、２つ目と３つ目の引数 outkey と shmsize は省略可能で、それらを省略した場合は logfile がさらに省略可能な２つ目の引数になります。

-s SSL/TLS 化されたブローカーに接続します。

-t topic

受信するべき MQTT メッセージのトピックを topic に設定します。トピックは任意の文字列で、'/'で区切って階層化することができます。 １つまたは複数の階層をそれぞれワイルドカード '+'と'#' で表現することにより、 複数のトピックを受信指定することもできます。トピックの詳細については MQTT の資料を見てください。

-p port

MQTT ブローカー host の TCP ポート番号を port に指定します。デフォルトは 1883 です。ただしオプション -s で SSL/TLS 化した場合のデフォルトは 8883 です。

-q qos

MQTT ブローカーとの間の通信品質(QoS)を qos に設定します。設定できる値は 0,1,2 のいずれかで、デフォルトは 0 です。 QoS の意味については MQTT の資料を見てください。

-u user

-P pswd

MQTT ブローカーがユーザー認証を必要とする場合、ユーザー名 user とパスワード pswd をそれぞれ設定します。

-c cafile

-d capath

SSL/TLS 対応のブローカーの場合、CA(認証局)証明書である　PEM 形式のファイル cafile またはそれを内容として持つディレクトリ capath のいずれかを指定します。　デフォルトでそれぞれ　/etc/ssl/certs と/etc/ssl/cert.pem　が使用され (試され)ますが、これらがいずれもが有効ではない場合は指定が必要です。

-C certfile

-k keyfile

SSL/TLS 対応のブローカーがクライアント認証を必要とする場合、 PEM 形式の証明書ファイル　certfile と　PEM 形式の秘密鍵ファイル　keyfile の２つをそれぞれ指定します。

-D　daemon モードで起動します。

-g　デバッグ情報を出力します。 ここで出力されるトピック・QoS・メッセージ ID は受信された 各メッセージに付いてきたものです。

使用例

ホスト　my ブローカー の MQTT ブローカーから WIN データを受信してキー 21 の 共有メモリ (大きさ 1000KB)に書き出します。　MQTT のユーザー名 user、パスワード pass です。

```
mq2raw -u user -P pass myブローカー 21 1000
```

このとき、並行して WIN データパケットを UDP ポート　7000 で受信して キー 11 の共有メモリに書き出しているとします。

```
recvt 7000 11 1000
```

これら２系統のデータを１つの共有メモリに合流させるには、 次の３つの方法があります。

(1)　sendt_raw 21 localhost 7000

MQTT 経由のデータを UDP に転送して共有メモリ 11 にまとめる。

(2)　raw2mq -u user -P pass 11 myブローカー

UDP 経由のデータを MQTT に転送して共有メモリ 21 にまとめる。

(3)　rawmix 11 21 31 1000

共有メモリ 11 と 21 のデータを別の共有メモリ 31 にまとめる。

ホスト　my ブローカー　の MQTT ブローカーから WIN データを受信して、shmx(1W)　で
チャネル番号 9100,9102,9103 の 3 チャネルの波形をモニターします。

```
mq2raw -a myブローカー -u user -P pass | shmx 9100 9101 9102
```

上と同様ですが、バイナリ/ASCII 変換は shmdump(1W) で行います。

```
mq2raw -b myブローカー -u user -P pass | shmdump -tq - 9100 9101 9102 |shmx 9100 9101
9102
```