

Gateway Software Status Update 21 Mar 2014

Contents

Topology	qFOCE 2014/2Q test configuration
One Pager	utility reference
Basics	basic gateway concepts
System	system administration
Operation	common operations
Archive	working with data
Examples	archive utility examples

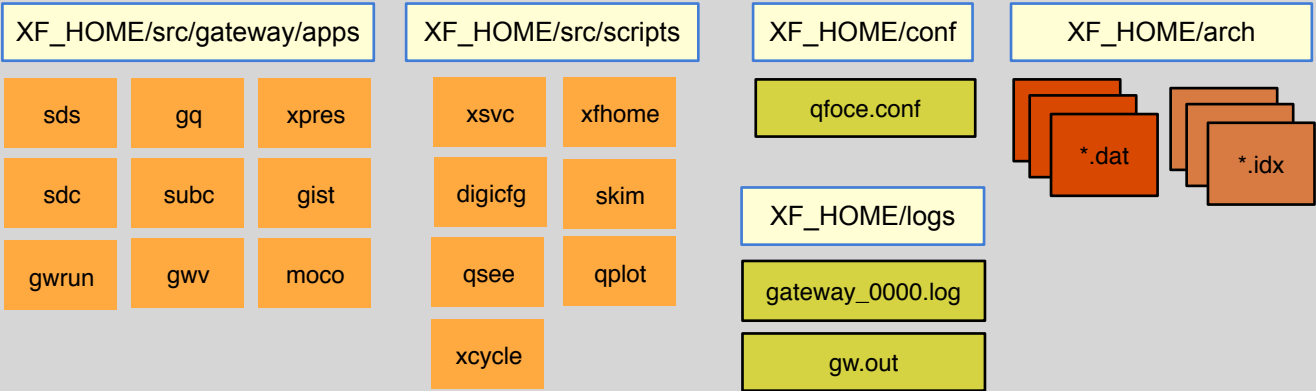
**qFOCE
2014/Q2 test**



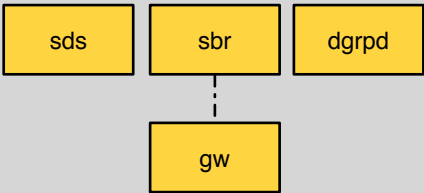
- util
- conf, syslog
- data archive
- app, process

Kelpbed.swfoce.mbari.org

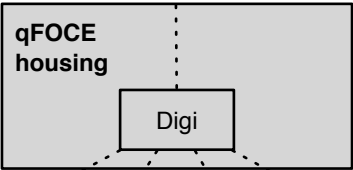
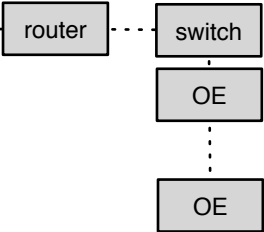
Files



Applications



swfoce.mbari.org



```
xsvc show
xsvc start sds -p
xsvc stop sds
xsvc start sbr
xsvc stop sbr
xsvc start gw -f <config>
xsvc stop gw
```

Gateway Utility Summary

```
<util> -h prints use help
```

```
sudo digicfg -i QF -a <host>
```

```
status           : gq -q svstat
properties        : gq -q svprop
list topics       : gq -q topics
IO properties     : gq -q ioprop
Sampling Properties: gq -q saprop
Debug Settings    : gq -q debug
gq /
gq -s property.sbe18s.0.cal_k=<val>
gq -s property.sbe18s.0.cal_slo=<val>
gq -s property.sbe18s.0.cal_off=<val>
gq -s property.sbe18s.0.cal_cor=<val>
gq -s debug_level.SBE18S=DEBUG
gq -s worker_period_msec.sbe52mp.0.GETD.0=60000

gq -d act.dcraft.0.cspeed=<counts>
gq -d act.sbe52mp.0.pump=fast
```

Examine gateway system logs

```
less logs/gw.out
less logs/gateway_nnnn.log
```

```
subc
subc -t /data/sbe
subc -t /data/sbe18s.0
subc -a <addr> [options]
sdc -t /sdata/tref
```

subc and sds topics are defined in the **streams** section of the gateway config file

```
xpres -d arch -s 1812 -b 2014/06/01T12:00 -e 2014/06/02T12:00
xpres -d arch -s 1812 -B 1394000000 -B 1394012000
skim -M 30 -d arch -s 1812
etx [-u] 1394000000
date --date="2014/03/05T06:13" +%s
date --date='@1394000000'
```

Control Motors (no gateway)

```
moco -c <counts> -t <dev>[,<speed>,<par>,<dbit>,<sbit>]
```

Gateway Basics

Gateway Environment

- XF_HOME must be set
- The following directories must be in the PATH variable
XF_HOME/src/scripts
XF_HOME/src/gateway/apps
- The environment and PATH are set up automatically when you log in as ops
- The environment should be set/changed using
source XF_HOME/src/scripts/xfhome [/path/to/new/XF_HOME]*

* default is current directory

Gateway Utility/Script Conventions

- CLI (command line interface) syntax is lower case
- get help using '-h' option
- most components support verbose output with -V (sometimes -v)

Gateway Clients

Services used to query or change the gateway are presented by the service broker (proxy). Gateway clients request operations through a service API; the operations supported include variations of:

get : request information about configuration, state, status
set : change configuration
do : request some action be taken (e.g. set actuation values)

Operations use **topics** to specify what is to be get, set or done; a topic represents a single configuration or state value, and is generally associated with one data primitive (a number or string). Semantically, a topic has a path, name, type and value; some topics may be changed (via set or do, e.g. driver properties, actuation values), and some are read-only.

Topics

Topics are organized hierarchically, and their syntax resembles that used for file system paths, e.g.

/platform/config/id
/service/status/errors.sbel8s.0

By requesting more or less of a topic path, it is possible to access groups of related information, or a single item.

In general, **topic paths** are used for **GET** operations, while only **topic names** (e.g. **act.dcraft.0.cspeed**) are needed for **SET** and **ACT** operations.

The primary gateway clients are **gq**, **subc** and **sdsc**. Data archive clients include **xpres** and **gist**. These and other utilities are described in below.

Gateway: System

Login

```
ssh ops@kelpbed.swfoce.mbari.org  
ops:brynnner
```

See what components are running

```
xsvc show
```

Start/Stop Digi Ports requires root/sudo

```
sudo digicfg -i QF -a <host>  
sudo digicfg -i QF -a <host> -c uninit
```

Start/Stop SDS

```
xsvc start sds -p  
xsvc stop sds
```

Start sds and sbr before
starting the gateway

These may be left running, and
rarely need to be restarted

Start/Stop SBR

```
xsvc start sbr  
xsvc stop sbr
```

Start/Stop Gateway

```
xsvc start gw [-f <config_file>]  
xsvc stop gw
```

Config file name only.
Must be located in XF_HOME/conf

Kill an unresponsive app

```
ps -ef|grep <app>  
Or  
ps -C <app> -o pid= -o args=
```

Then

```
kill -9 <pid>  
Or  
sudo kill -9 <pid>
```

Gateway: System

Control Motors (gateway NOT RUNNING)

```
moco -c <counts> -t <dev>[,<speed>,<par>,<dbit>,<sbit>]
```

Examine gateway system logs

```
use 'less' or 'xflog' (beaglebone)  
less logs/gw.out  
less logs/gateway_nnnn.out
```

```
use 'xflog' instead of less on beaglebone  
xflog logs/gateway_nnnn.log  
:q! to exit xflog
```

Check Gateway version

```
gwv
```

Gateway: Operation

Set thruster speed

```
gq -d act.dcraft.0.cspeed=<counts>
```

Typically, motor speed is in the range -1024 to 1024

Set pH cal coefficient

```
gq -s property.sbel8s.0.cal_k=<val>
gq -s property.sbel8s.0.cal_slo=<val>
gq -s property.sbel8s.0.cal_off=<val>
gq -s property.sbel8s.0.cal_cor=<val>
```

gq, subc and sdc use the "-a" options to specify a remote host

Service Queries

```
status           : gq -q svstat
properties        : gq -q svprop
list topics       : gq -q topics
IO properties     : gq -q ioprop
Sampling Properties: gq -q saprop
Debug Settings    : gq -q debug
```

Show all topics/values

```
gq /
```

Show specified topic value(s)

```
gq <topic> [<topic>...]
```

Stream pubsub data

```
[show all]
subc
```

```
[show sbel8, sbe52...]
subc -t /data/sbe
```

```
[show sbel8s.0 only]
subc -t /data/sbel8s.0
```

```
[remote]
subc -a <addr> [options]
```

pubsub and sds topics are defined in the gateway config file

Stream sds data

```
sdc -t /sdata/tref
```

Gateway: Data Archive

Where is the data?

Each instrument or other data source stores its data in a single archive. Archives are kept on the gateway in `XF_HOME/arch` by default.

Each archive consists of one or more files, segmented by size; the segment size may be configured per device in the gateway configuration file. Each log segment consists of a data (.dat) and index (.idx) pair; the index enables fast searches on large archives.

Archive file names reflect the device type, instance, source ID, and archive segment, e.g. `sbcl8s_1_1813_0000.dat`. Do not rename archive files.

In some configurations, the data may also be mirrored to other locations using `rsync` or similar utilities.

How is data stored?

An archive data file (.dat) consists of a binary file header and one or more data records.

Data records consists of a binary record header, plus the record data; record data are stored in their original format (i.e., as originated from the device). The binary headers contains metadata that enable searches, diagnostics, and repairs e.g. `stream_id`, sequence number, data size, etc. Similarly, the index files contain information about segment content.

Since there is one archive per device, the archive may contain multiple streams (record types). Each stream ID corresponds to a specific data format. This is somewhat device-dependent (i.e., up to the driver developer), but best practice is to assign a unique stream ID to records with different number, type and/or order of fields. As a guideline, any data that may be handled transparently by any application may use the same stream ID.

stream ID 0-99 are reserved.

stream ID 0 : metadata record

stream ID 99 : error record

Device data stream ID are defined in `xf-id.h`. Devices may use non-reserved stream IDs as they wish.

Different device **types** (e.g. `sbcl8s` and `sbe52`) may use the same stream ID for different purposes.

Different device **instances** (`sbcl8s.0` and `sbcl8s.1`) must use the same stream ID for the same purpose.

See also:

`src/gateway/core/xf-log.c`

`src/include/xf-log.h`

Gateway: Data Archive

How do I see the data?

Archive contents are viewed using a utility called **xpres**. xpres enables flexible, formatted presentation of all or part of an archive.

For example, it is possible to filter by time and record type. It is also possible to include or exclude various parts of the archive metadata, and offers some control over representation of binary data.

A script named **skim** uses xpres to dump the latest data in the archive. Specify the amount of data in hours, minutes, and/or seconds, and optionally use an xpress config file to format the output.

A utility named **gist** provides an overview of an archive. This can be useful for finding the archive extents (start/end times, number of records...), for example.

XPRES Overview

```
xpres -d arch -s <source ID> ...
```

There are lots of options:

- start, end times (date or epoch time)
- stream ID
- skip modulo n
- formatted output
- Automate complex queries using config file:

```
xpres [options] < myquery.conf
```

See Also: Detailed use info below

xpres Options

Usage: xpres [options...] [< config_file>]

options:

-s: source ID
-t: stream ID
-b: begin time (YYYY/MM/DDTHH:MM:SS)
-e: end time (YYYY/MM/DDTHH:MM:SS)
-B: begin time (epoch sec)
-E: end time (epoch sec)
-d: log directory
-n: skip interval
-o: output format
 e.g., -o <source>,<stream>,"<format specifiers>"[,<mnemonics>...]
 where
 <source>,<stream> : source or stream ID or -1 for all
 <format specifiers> : printf specifiers
 <mnemonics> : mnemonics for data components (see below)

Note:

Quotation marks and escaped characters on the command line must be double escaped:

xpres -d arch -s 300 -o 300,-1,\"%lld,%lld\\n\\",ets,seq

-f: output time format (see strftime, strptime)

-D: debug output

-v: verbose output

-h: print help message

Data Component Mnemonics:

etm: epoch time msec	[%llu]	
ets: epoch time sec	[%llu]	
iso: local time/date	[%s]	(specified by time format using -f option; see strftime)
utm: UTC time/date	[%s]	(specified by time format using -f option)
asc: data buffer ASCII	[%s]	
pas: data buffer ASCII	[%s]	(printable ASCII only, ignore control characters)
hex: data buffer hex	[%s]	
bin: data buffer binary	[%s]	
uue: data buffer uuencoded	[%s]	(not implemented)
pda: parsed data	[%s]	(not implemented)
mdc: metadata cause	[%s]	(not implemented)
src: source ID	[%u]	
pid: platform ID	[%u]	
str: stream ID	[%u]	
stn: stream ID name	[%u]	
seq: sequence number	[%llu]	
dln: data length	[%llu]	
dsz: data record size	[%llu]	
mdr: metadata ref	[%llu]	(not implemented)
pst: statistics	[%s]	(not implemented)

xpres Examples

xpres Examples

```
# Dump entire log
xpres -d arch -s 2001
1394576574919,2001,301,-4.1415,277.1500,0.0000**
1394576577169,2001,301,-4.1415,277.1500,0.0000**
...

# Default, specify time window
xpres -d arch -s 2001 -b 2014/03/11T23 -e 2014/03/11T23:01
1394578801253,2001,301,-4.1419,277.1500,0.0000**
1394578803756,2001,301,-4.1417,277.1500,0.0000**
...
```

Using xpres config files

```
# xpres.conf [may use any valid file name/extension]
-d arch
-s 2001
-n 1
-v
-o 2001,-1,"%llu,%d,%s\n",ets,dln,asc
-o 2001,301,"%s,%llu,%llu,%s\n",utm,ets,dln,asc
-o -1,-1,"%c %llu.%llu,%d,%s,%s\n",ets,etm,dln,hex
-o 999,-1,"%b %llu.%llu,%d,%s\n",ets,etm,dln,bin
-f "%Y-%m-%d %H.%M.%S"
```

Run using options file

```
xpres < conf/lshow_example.conf
2014-03-11 22.22.54,1394576574,25,-4.1415,277.1500,0.0000**
2014-03-11 22.22.57,1394576577,25,-4.1415,277.1500,0.0000**
...
```

override the source

```
# note that it uses the default format, since no format
# for the specified source is defined in the config file
```

```
xpres -s 5001 < conf/lshow_example.conf
1395095518326,5001,701, 0.7024, 22.2956, 0.08, 5.260****
1395095523436,5001,701, 0.7025, 22.2965, 0.09, 4.519****
...
```

skim Examples

skim Examples

Show help message

\$ skim -h

usage: skim

-H <hour> : hours [0]

-M <min> : minutes [0]

-S <sec> : seconds [0]

-s <id> : stream id

-d <arch> : arch directory [/home/headley/host/hg/xfoce-31mar2014/xfoce/arch]

-c <cfg> : config file

-v : verbose output [FALSE]

-h : print this help message

Show last 123 h 48 m of data of source 1812

\$ skim -H 123 -M 48 -s 1812

1401843568881,1812,301,0.0000,2.6751,0.0000,0.0000**

1401843574900,1812,301,0.0000,2.6862,0.0000,0.0000**

1401843580925,1812,301,0.0000,2.6969,0.0000,0.0000**

...

Show last 24 h of data of source 1812, use config file

\$ skim -H 24 -s 1812 -c mycfg

2014/07/21T12:34 0.0000,2.6751,0.0000,0.0000

2014/07/21T12:34 0.0000,2.6862,0.0000,0.0000

2014/07/21T12:34 0.0000,2.6969,0.0000,0.0000

...

gist Examples

gist Examples

Show help message

\$ gist -h

Usage: gist [-dhst]

options:

-d: directory

-h: print help message

-s: source ID

-t: stream ID

Show log summary

\$ gist -d arch/ -s 1649

source ID:[1649] segments:[1] first:[0000] last:[0000]

[path : arch/sbe52mp_0_1649_0000.idx]

[platform : 1001]

[source : 1649]

[min_key : 1401843751183]

[max_key : 1401843939336]

[records : 002]

[start date : 2014/06/04T01:02:31]

[end date : 2014/06/04T01:05:39]

Plot Overview

A pair of utility scripts, **qplot** and **qsee**, are provided to generate simple PDF plots from delimited data:

qplot

generates a plot from one or more data files

qsee

automates plot generation, creating data files using **skim**, and invoking **qplot**

Features

- output in PDF and Postscript formats
- use command line and/or config file to specify options
- useful for data quick-look data and performance metrics plots

qplot exposes only a small fraction of the gnuplot API; it is intended for simple diagnostic applications. **qplot** and **qsee** are written in bash, and may be freely modified.

Requirements

bash

GNU plotutils package (<http://www.gnu.org/software/plotutils/>)

There are potentially many, long command line options for **qplot** and **qsee**; place command line options in a text file and read using file redirection or pipes:

```
qplot < myplot.conf
cat myplot.conf | qplot
```

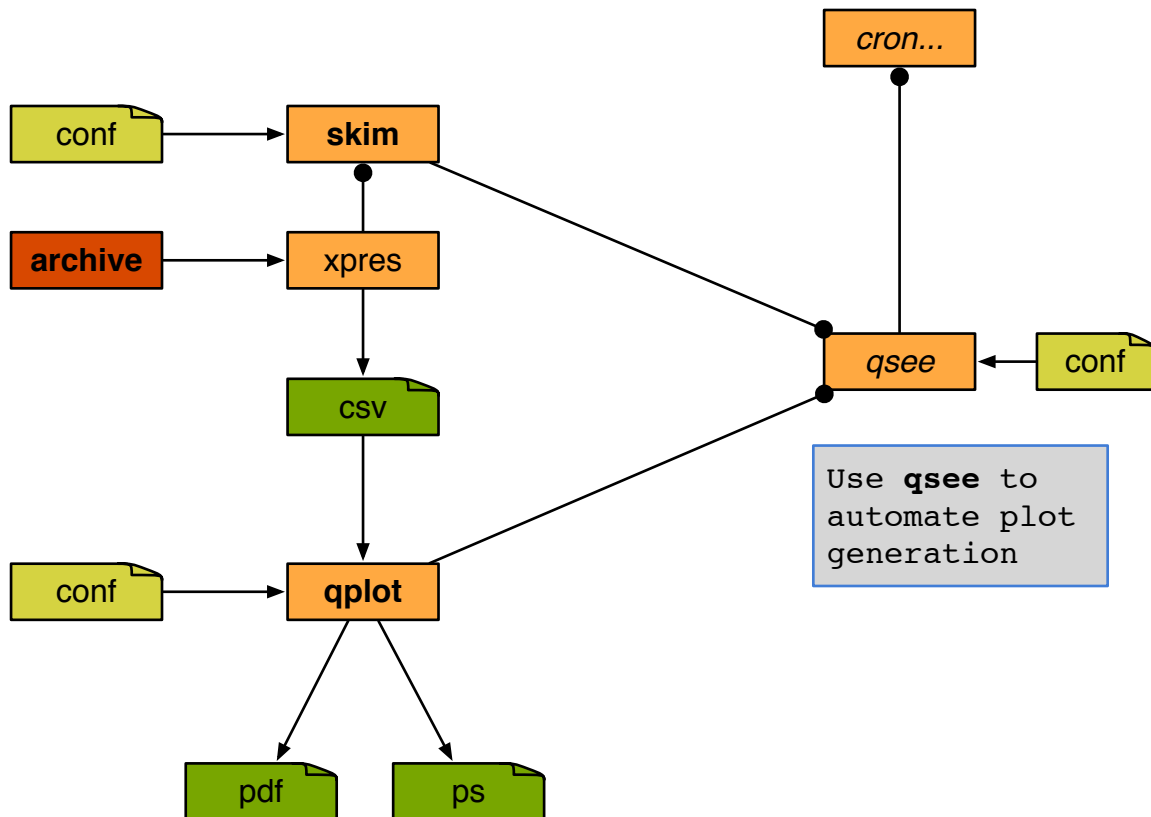
- The files may include one option/value pair per line
- Command line options override config file
- Lines beginning with '#' are ignored, making it possible to include comments or disable options without deleting them

See Also

sample config files in `src/scripts/qsee`

Plot Process

Use **skim**/xpres and **qplot** to manually generate plots



Use **qsee** to automate plot generation

qplot

```
$ qplot -h
```

qplot: Generate plots from one or more delimited text files

usage: qplot [options] file...

Options:

-d	: debug (keep temporary files)	[*]
-o <ofile>	: output file	[qplot_out]
-r <min Y>	: y-range min	[*]
-R <max Y>	: y-range max	[*]
-i <min X>	: x-range min	[*]
-I <max X>	: x-range max	[*]
-T <title>	: plot title	[]
-Y <title>	: Y axis title	[]
-X <title>	: X axis title	[Timestamp (epoch sec)]
-t <fmt>	: time input format	[%s]
-x <fmt>	: x axis (time) format	["%m/%d/%y %H:%M:%S"]
-s <sep>	: data separator	[,]
-y <pspec>	: file,xcol,N,col-1,title-1,...col-N,title-N	
-V	: verbose output	[FALSE]
-h	: print this help message	

qplot output is written to plot-output/pdf and/or plot-output/ps

qplot Example

Here is skim/xpres configuration.

- specifies formats for 3 devices (1812, 1813, 1649)
- record format: timestamp, data_size, data(printable ascii)
- uses time format "%s" (epoch seconds)

xp-qfoce.conf

```
-d /path/to/xfhome/arch
-o 1812,-1,"%llu,%d,%s\n",ets,dln,pas
-o 1813,301,"%llu,%d,%s\n",ets,dln,pas
-o 1649,701,"%llu,%d,%s\n",ets,dln,pas
-f "%s"
```

xpres -o option details

source ID stream ID (-1: default) printf format... ...element names (see xpres help)

-o 1812,-1,"%llu,%d,%s\n",ets,dln,pas

Generate CSV data for qplot (last 24 hour) for two devices:

```
cd $XF_HOME/tmp
mkdir qsout
skim -H 24 -s 1812 -t 301 -c xp-qfoce.conf > qsout/1812-301.dat
skim -H 24 -s 1813 -t 301 -c xp-qfoce.conf > qsout/1813-301.dat
```

Here is a qplot configuration:

qp-ph.conf

```
#-d
#-V
-o ph-qplot
-T "pH vs Time"
-X "Timestamp (epoch msec)"
-Y "pH"
-t "%s"
-y qsout/1812-301.dat,1,1,3,"1812-ph"
-y qsout/1813-301.dat,1,1,3,"1813-ph"
```

Use col 1 for X axis data

plot 1 field, col 3 show as "1812-ph" (may use multiple Y columns)

Multiple -y options allowed

Generate the plot:

```
qplot < qp-ph.conf
```

output:

- plot-output/pdf/ph-qplot.pdf
- plot-output/ps/ph-qplot.ps