Some very basic stuff about IceWeb

Glenn Thompson

Jan 9, 2000

ORGANISATION

IceWeb account:

Username: iceweb

password: !drspec

This is setup just like a normal Unix account. User 'iceweb' has an email account, to which any error messages or alarm messages are sent.

The home directory is /home/iceweb. Subdirectories contain (most of) the IceWeb code & data. The most important directories are REAL_TIME_CODE and PARAMETER_FILES. Aliases are set up such that:

cd $RTC

takes you to /home/iceweb/REAL_TIME_CODE, and

cd $PFS

takes you to /home/iceweb/PARAMETER_FILES.

Other important directories are ICEWEB_UTILITIES which contains IceWeb library functions (i.e. useful functions which are used by several different IceWeb programs.

The alias CGI also exists for /usr/local/apache/cgi-bin/iceweb which contains cgi scripts which are required to make some of the IceWeb web pages interactive (most importantly the IceWeb Setup Utility).

REAL TIME CODE

The directory REAL_TIME_CODE contains the code that is run every 10 minutes, for producing 10-minute spectrograms and reduced displacement plots. This code also sends alarms but the alarm system is covered in a separate section below.

This is the most complicated part of the IceWeb system. If you type:

crontab -l

you will see the cronjobs currentlyrunning on the IceWeb computer. The first of these runs every ten minutes and calls /home/iceweb/REAL_TIME_CODE/iceweb.csh. This is a c-shell script which calls the Perl script iceweb.pl. This then calls a C program 'start_matlab_engine' with the argument 'iceweb'. This C program starts Matlab, and tells Matlab to execute the Matlab function 'iceweb.m'. This function calls various subfunctions which plot & save spectrograms and reduced displacement plots, archive data and run the alarm function 'test_thresholds.m'. After this has been done for all the volcanoes listed in the parameter file $PFS/volcanoes.pf, the Matlab engine is closed, the C program ends, and the Perl program takes over again & starts converting postscript files (created by Matlab) to gif files, which are posted on the internal page. This cascade of cronjob to c-shell to Perl to

C to Matlab is, unfortunately necessary, though it is Perl & Matlab that do all the work. The c-shell is needed since otherwise the Perl script will not inherit the environment variables it needs. The C program is needed since Perl cannot start Matlab, but C can. (If we were to invest in the Matlab C-compiler, the Matlab programs could be compiled as C code and called directly from Perl).

PARAMETER FILES

Parameter files are files which contain setup data used by the IceWeb system. These are the parameter files used by IceWeb:

- parameters.pf
- paths.pf - a list of all the paths used by IceWeb - needs to be updated if the code is moved
- volcanoes.pf - a list of which volcanoes IceWeb is implemented for
- Redoubt.pf, Spurr.pf ... - one parameter file for each volcano that list which stations to calculate dr & spectrograms for. Also contains thresholds used by alarm system, and wind station data plotted on dr graph (e.g. cbwind=Cold Bay).

The reason why parameter files are so useful is that they can easily be read by Perl or Matlab programs (the 2 main languages used by IceWeb) using simple interfaces which are part of the Antelope system (which is itself incorporated into Iceworm).

In the fall of 1999, IceWeb was completely rewritten, and rewriting so that it used parameter files was the main motivation behind this. As a result the system is now better organised and more 'Iceworm-compliant' which means it is less likely to break in the future.

Some functions which read these parameter files are in /home/iceweb/ICEWEB_UTILITIES. The following are particularly important:

open_pointer_to_volcano.m

read_iceweb_volcanoes.m

read_iceweb_stations.m

read_thresholds.m

these functions are used in many places in the IceWeb code.

THE ALARM SYSTEM & SETUP UTILITY

Alarm algorithm

The alarm system now implemented is similar to EVA, but can be applied to far more stations. For each station, a threshold dr is set. If dr rises above this, (and the peak frequency of the signal is >0.8Hz) that station triggers. If more than 1 station at a particular volcano triggers, the volcano triggers. If only 1 volcano triggers, an alarm is sent. If several volcanoes trigger, the alarm is assumed to be due to a non-volcanic earthquake.

This alarm algorithm is easier to understand than a previous one which used an STA:LTA, which makes it easier to work with. It was designed this way because during the Shishaldin eruption, Steve McNutt would sometimes say something like "Call me when Dr hits 10cm^2" - this can now be automated.

In theory (and if a lot of assumptions are made) the Dr values for each station at a particular volcano should agree. In practice this is usually not realised, so it is actually necessary to set a different threshold level at each station, rather than just 1 for a particular volcano.

IceWeb setup utility

Threshold levels can be set over the web using the IceWeb Setup utility:

http://www.avo.alaska.edu/Input/iceweb/SETUP

Username: icewebkicksbutt

Password: !drhelispec

Beeper duty people should bookmark this page to make it easily accessible.

This is the preferred way to change thresholds/stations/volcanoes for IceWeb (the other way is to edit parameter files directly).

Again, this design was motivated by the needs of the beeper duty staff, who decided that if there was to be any kind of IceWeb-based alarm system connected to the beeper, they would need a way to control the settings of the alarm system remotelyvia the Web.

Example 1: Tremor increasing

Lets say tremor is increasing and when Dr>5 cm^2, an alarm is sent. However, the beeper duty person might decide that this is not too serious, and that he does not wish to be awaken again unless Dr>10 cm^2. So he logs in from home & resets the threshold level to 10 cm^2 over the Web.

If he could not do this he might receive a series of alarms (for Dr>5 cm^2) which would keep him up all night. This is much better than the currently used EVA system, since he cannot change EVA settings unless he drives into the lab.

Example 2: False alarms

If threshold levels are adequately tuned the false alarm rate should be acceptably low. However, if telemetry noise is particularly bad one night, it is conceivable a beeper duty person might receive a series of false alarms. This would be extremely irritating, but at least the beeper duty person can log in from home & turn the offending stations off (he can

figure out which are the offending stations by looking at the 10-minute spectrograms - the same stations are used).

Can this alarm system be made practical?

It would be extremely valuable for AVO to have an effective alarm system for all seismically monitored volcanoes connected to a beeper, since this would provide 24 hour coverage (volcanoes never sleep).

The major difficultly in designing an effective alarm algorithm is that it can be fooled by several types of noise (telemetry, station outage, storm, wind). However, there is no reason why the algorithm presently used should not be (at least) as effective as the EVA system, since it is similar in design.

The real problem is the overhead such a system would place on beeper duty people. Given that spectrograms & dr plots usually give us sufficient warning of a volcanic crisis, and when a volcanic crisis is in progress the lab is usually manned 24 hours a day, this overhead may be too much. This overhead depends on the number of false alarms produced by the system and the amount of tuning (of thresholds) required.

Implementation of the alarm system

The alarm system is not at present active (although alarms are sent by email to user 'iceweb'). Some sort of plan for the testing & implementation of this alarm system needs to be developed. For example:

1. Initially someone (I would recommend Gordon) would have to 'tune' the threshold levels at each station such that they only trigger when something interesting happens, but do so before something catastrophic occurs. Initial tuning should be based on Dr data over some suitably long interval (the program dr_plot_request in the directory / home/iceweb/ANALYSIS will be helpful for making plots of archived Dr data). Stations that are dead should be switched off (ideally removed from IceWeb altogether).The system could be left in this 'test' mode for some time, with alarms being sent only to user 'iceweb' (this is the current default). The performance of the system should privately be reported to Steve & fine tuning (based on the alarms sent) should be done.
2. If the performance is satisfactory, there should be a meeting with beeper duty people, and the alarm system should be explained & discussed. In particular they should be shown how to use the IceWeb Setup Utility to change thresholds & stations used. Perhaps then a phase will follow during which alarms are emailed to all beeper duty staff and they are encouraged to give feedback.
3. Finally, the user 'beeper' could be added to the alarm email list, when performance is judged to be reasonable & beeper duty staff are comfortable with the alarm system.

Notes:

- An email sent to user 'beeper' should automatically be sent to the beeper carried by the beeper duty person. This can be easily tested (and needs to be). If it doesn't work, tell Kent.
- To add the users 'steve', 'beeper' and 'guy' etc to the alarm email list, open the file

$PFS/parameters.pf & edit the part which says:

```
# alarms
alarm_list &Tbl{
        iceweb
}
```

so that is instead reads:

```
# alarms
alarm_list &Tbl{
        iceweb
        steve
        beeper
        guy
}
```

and save the file.

Installing IceWeb on the new computer [Page 1 of 2]

Two-machine setup

For reliability reasons, it would be better to split IceWeb into two parts, with the 10 minute spectrograms and dr plots running on the new machine, and stuff that runs on a different timescale (once per hour or once per day) on Ukinrek. Actually the processes on Ukinrek should not impact Ukinrek much at all - perhaps an average of 2 minutes of CPU time per hour, so Ukinrek could still be used as a machine for a staff member. If this is considered to be impractical, then follow the instructions below for a 1-machine setup.

1. Log on as iceweb (password !drspec)
2. rlogin to new computer
3. cd /home/iceweb/CRONS
4. crontab newcomputer.cron
5. rlogin to Ukinrek
6. cd /home/iceweb/CRONS
7. crontab ukinrek.cron
8. send an email message to cheetah_thompson@hotmail.com giving name of the new machine & saying a 2 machine setup is being used

This setup will put the most important stuff - the 10 minute spectrograms & dr plots - on the new computer. Less important stuff will run on Ukinrek. This way, the less important stuff doesn't compete with the 10 minute stuff, with the result that the 10 minute stuff runs more reliably.

Example: daily spectrograms are updated once per hour. If these were run on the same machine as the 10 minute stuff, then once out of every 6 runs, both processes would compete for CPU time, with the result that the 10 minute stuff would take longer than it normally does. If it took more than 10 minutes, then it would 'collide' with the next run of the 10 minute stuff - so some spectrograms and dr plots would not be produced, i.e. reliability of the system would decrease. Ideally all processes that run on a different time scales (e.g. every 10 minutes, every hour, every day) should run on different machines.

Installing IceWeb on the new computer [Page 2 of 2]

One-machine setup

Follow these instructions ONLY IF the two-machine setup described above is impractical. With this setup all cronjobs will be run on the new computer (none on Ukinrek) with less important cronjobs 'niced' so they do not compete with the 10 minute stuff.

1. Log on as iceweb (password !drspec)
2. rlogin to new computer
3. cd /home/iceweb/CRONS
4. crontab current.cron (this runs the cronjobs on the new machine)
5. rlogin to Ukinrek
6. crontab -r (this removes cronjobs from Ukinrek)
7. send an email message to cheetah_thompson@hotmail.com giving name of the new machine & saying a 1 machine setup is being used

Troubleshooting

Past problems (in order of frequency) with IceWeb have been due to:

1. Upgrades to Iceworm/Antelope or Matlab (happens far too often - every 3 months or so)
2. Iceworm problems which prevent source data appearing in correct place at the correct time (happens occasionally, perhaps every few months)
3. Network problems, which impact data retrieval from Ice/Earlybird or data writing to Kiska (gif images, html pages)

The first problem is often fatal & requires debugging and recoding (of the Antelope-Matlab interface, or the get_iceworm_data_for_station function in / home/iceweb/REAL_TIME_CODE/iceweb.m.

IceWeb is a useful diagnostic tool. Since the web plots are looked at quite frequently they often give us the first notice that one of the above problems has occurred. If some web plots work and others don't, that information can help pinpoint the exact cause. Error me

Automated email error messages

IceWeb has been designed such that error notification messages are sent to user 'iceweb' whenever a serious error is detected. These messages also help to diagnose the problem. The most serious message is 'IceWeb has not run for > 2 hours'.

Important questions:

- Are the IceWeb computers down? (if so, IceWeb will not run)
- Are the cronjobs still running (log in and type crontab -l)
- What error messages were sent to user 'iceweb'?
- Are the 10 minute spectrograms still being updated every 10 minutes?
- Are the reduced displacement plots still being updated every 10 minutes?
- Are the current day daily spectrograms still being updated every hour?

If there is a problem with the 10 minute spectrograms or dr plots:

1. log in as iceweb
2. cd $RTC
3. start matlab
4. run the function iceweb (just type 'iceweb' at the Matlab prompt & note any errors displayed (there should be none - warning messages are not errors)

Do spectrograms & dr plots appear on the screen? If so, problem is either with Perl script 'iceweb.pl' or with the C program 'start_matlab_engine'. Test this by running iceweb.pl (from the Unix prompt). If you get a lot of messages saying that certain postscript files do

not exist, it is likely that start_matlab_engine has failed due to a Unix upgrade - so recompile start_matlab_engine. The source code is in /home/iceweb/ICEWEB_UTILITIES/src/start_matlab_engine. Change to that directory, run 'make' and then copy the machine code file (start_matlab_engine) to /home/iceweb/ICEWEB_UTILITIES (overwriting the version already there). If still no luck, email cheetah_thompson@hotmail.com.

The TESTING directory

The directory /home/iceweb/TESTING is useful for troubleshooting. The file last_ran is 'touched' everytime iceweb.pl runs successfully (the script which makes 10 minute spectrograms & dr plots - but nothing else) so by doing an 'ls -l' you can see when this last ran.

The file 'run_time' records how long each execution of 'iceweb.pl' takes (if successful). To see these data, start matlab and type:

> load run_time

> plot(run_time,'.')

> ylabel('run time (s)')

This will give a graph of how run time has varied with time. Run time should be comfortably below 600 s. If the run time has exceeded this for a few hours or longer, there may be a network problem. The solution may be to temporarily take some volcanoes off iceweb.

If run time has exceeded 600 s for a long period of time (days, weeks etc) it indicates the computer is overloaded & some volcanoes must be removed permanently or a faster computer used. Ideally the run_time should be around 400s, which should allow IceWeb to keep running when minor network problems (fairly common) occur. To insure against more serious network problems, run_time (under normal conditions) needs to be less, but the more conservative you are, the more expensive computer you need to buy to run the program in a shorter amount of time.