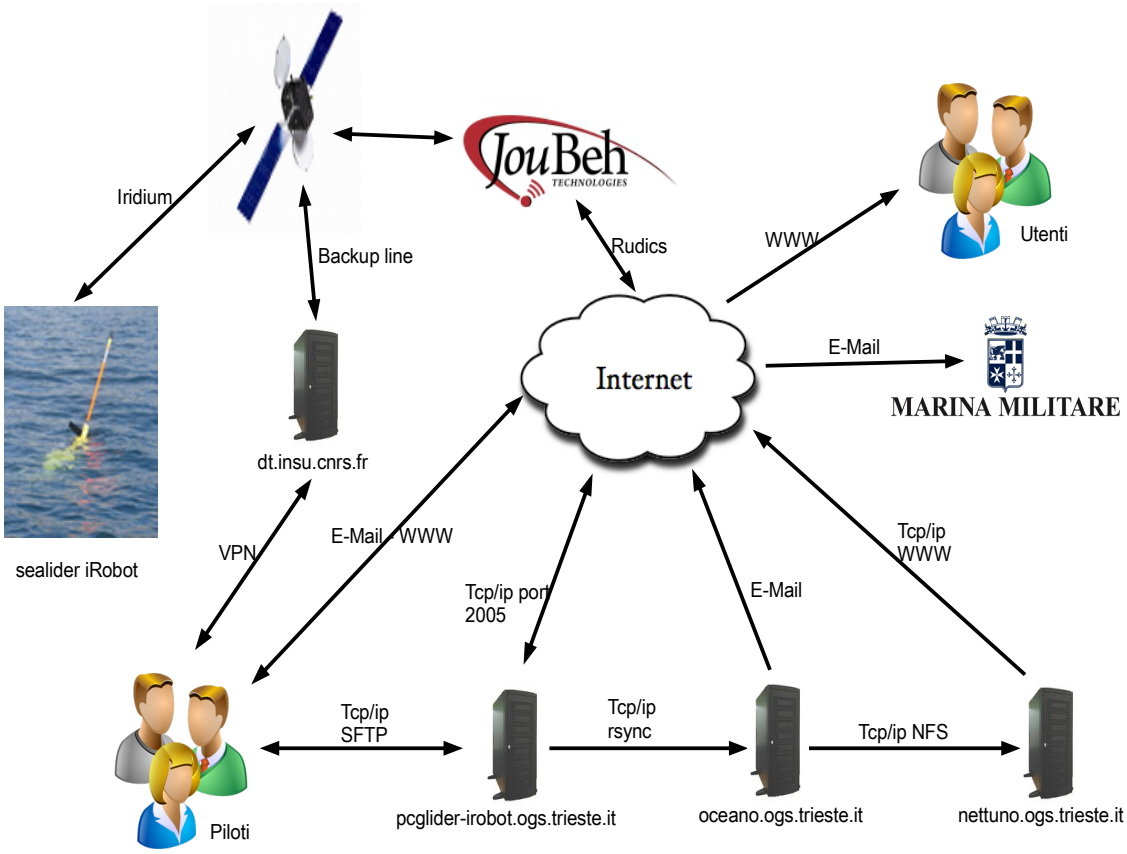
Configurazione e gestione del flusso dati del seaglider iRobot



A. BUSSANI e R. GERIN

Approved by: .....

Dr. Paola Del Negro

INDICE:

Abstract

Il flusso di dati del seaglider iRobot è piuttosto complesso, per questo motivo si rende necessario scegliere ed organizzare in maniera efficiente tutti i servizi e gli script necessari.
Il glider Amerigo (sg554) della iRobot si connette a terra, attraverso un modem iridium, il quale comunica con la basestation (server a terra) attraverso il sevizio RUDICS (fornito da JouBeh, a pagamento).
Il servizio RUDICS permette di creare un tunnel tcp/ip tra i sistemi riceventi della JouBeh e il computer denominato pcglider-irobot.ogs.trieste.it, presso l'OGS (stanza Argo Italy).
All'interno della rete OGS si trovano i server necessari all'elaborazione e alla notifica verso l'esterno dei dati ricevuti.
I server in questione sono: pcglider-irobot.ogs.trieste.it, oceano.ogs.trieste.it, nettuno.ogs.trieste.it.
I server all'interno della rete OGS presentano dei servizi di notifica automatica sulla creazione/eliminazione di file (incron), servizi di controllo del corretto funzionamento temporizzati opportunamente e servizi di sharing di directory (NFS), strumenti che vangono utilizzati in maniera opportuna per poter rendere il flusso di dati operativo ed efficiente.
All'interno del report viene descritto il dettaglio del flusso delle informazioni e le relative configurazioni necessarie ad attuare i servizi di base necessari al funzionamento del sistema.

Descrizione flusso dati

La basestation (pcglider-irobot.ogs.trieste.it) ha un servizio di notifica automatica (incron) sulla cartella dove arrivano i file trasmessi dal glider (servizio RUDICS, ip 140.105.70.99 porta 2005).
Il software proprietario della iRobot si occupa di monitorare la ricezione dei dati e eseguire i principali programmi di decodifica. Tali software si occupano anche di segnalare attraverso la presenza del file .connected nella cartella /home/sg554/ se la connessione tra glider e basestation è effettuata correttamente.
Il demone incrond, nella versione attualmente installata, può monitorare più eventi sulla singola cartella, ma non è possibile specificare diversi script che devono essere eseguiti se l'evento è di un tipo, piuttosto che un'altro. I diversi eventi che possono accadere all'interno di quella specifica cartella, vengono fatti analizzare da un singolo script, il quale si occuperà di scegliere quale altro script eseguire a seconda del tipo di evento che è occorso, lo script che si occupa di gestire quale script eseguire è il seguente:
/root/script_icron_connected

```
[root@pcglider-irobot pilot]# incrontab -l
/home/sg554/ IN_CREATE,IN_DELETE /root/script_icron_connected $# $%
```

Per maggiori informazioni sul servizio incron vedere pagina 20.

Alla creazione del file .connected, la basestation si occupa di trasferire il medesimo file su oceano.ogs.trieste.it (script **/root/script_icron_notify_connection2oceano** richiamato a sua volta da **/root/script_icron_connected** il quale è eseguito tramite incron, evento IN_CREATE -creazione- del file .connected nella directory /home/sg554/),

```
scp /home/sg554/.connected
glider@oceano.ogs.trieste.it:/storage/sire/dati/glider/seaglider/amerigo/actual/
connected
```

Al momento della disconnessione di RUDICS su pcglider-irobot.ogs.trieste.it (evento IN_DELETE del file /home/sg554/.connected) parte lo script **/root/script_icron_disconnected** (richiamato a sua volta da **/root/script_icron_connected**) che trasferisce tramite rsynch tutto il contenuto della directory /home/sg554/ verso il server oceano.ogs.trieste.it nella directory /storage/sire/dati/glider/seaglider/amerigo/actual/

```
rsync -azv -e ssh /home/sg554/
glider@oceano.ogs.trieste.it:/storage/sire/dati/glider/seaglider/amerigo/actual/
2>&1
```

Viene creato il file /home/sg554/transmitted (server pcglider-irobot.ogs.trieste.it) e trasferito su oceano.ogs.trieste.it (questo file serve per far partire la procedura di elaborazione dei file trasferiti sul server oceano.ogs.trieste.it), attraverso rsync.

```
touch /home/sg554/transmitted
DATO2=$(rsync -azv -e ssh /home/sg554/transmitted
glider@oceano.ogs.trieste.it:/storage/sire/dati/glider/seaglider/amerigo/actual/
)
```

Alla fine dei trasferimenti tramite rsync, lo script **/root/script_icron_disconnected** rimuove il file transmitted nella directory /home/sg554/ (non viene toccato il file .connected che viene gestito dai programmi del produttore del glider, iRobot).

```
rm /home/sg554/transmitted
```

Il file /home/sg554/transmitted (server pcglider-irobot.ogs.trieste.it) viene trasferito su oceano.ogs.trieste.it nella directory

```
/storage/sire/dati/glider/seaglider/amerigo/actual/
```

Il trasferimento genera le condizioni per l'esecuzione degli script di elaborazione. Il server oceano.ogs.trieste.it, infatti, è anch'esso equipaggiato con un sistema di notifica della variazione dei file ricevuti nella directory

```
/storage/sire/dati/glider/seaglider/amerigo/actual/
```

ogni creazione di nuovi file in quella cartella, fa partire lo script **/storage/sire/work/glider/script/updateDataglider** (utenza glider)

```
[glider@oceano actual]$ incrontab -l
/storage/sire/dati/glider/seaglider/amerigo/actual/ IN_CREATE
/storage/sire/work/glider/script/incron_amerigo_actual $# $%
```

Lo script **/storage/sire/work/glider/script/incron_amerigo_actual** discrimina nell'evento IN_CREATE (creazione) dei file contenenti la stringa "**transmitted"** oppure **"connected**".
Il file transmitted indica che il trasferimento dei file è concluso, quindi è necessario far partire l'elaborazione di tutti i file che sono presenti.
Il file connected implica la creazione del file connesso

```
touch /storage/sire/work/web/sire/glider/connesso
```

attraverso il controllo sulla sua presenza, effettuato all'interno della pagina php sul server nettuno.ogs.trieste.it http://nettuno.ogs.trieste.it/sire/glider/home.php (accesso ristretto), è possibile visualizzare la situazione riguardante il trasferimento dei dati tra il glider e la basestation (vedi illustrazione 1).

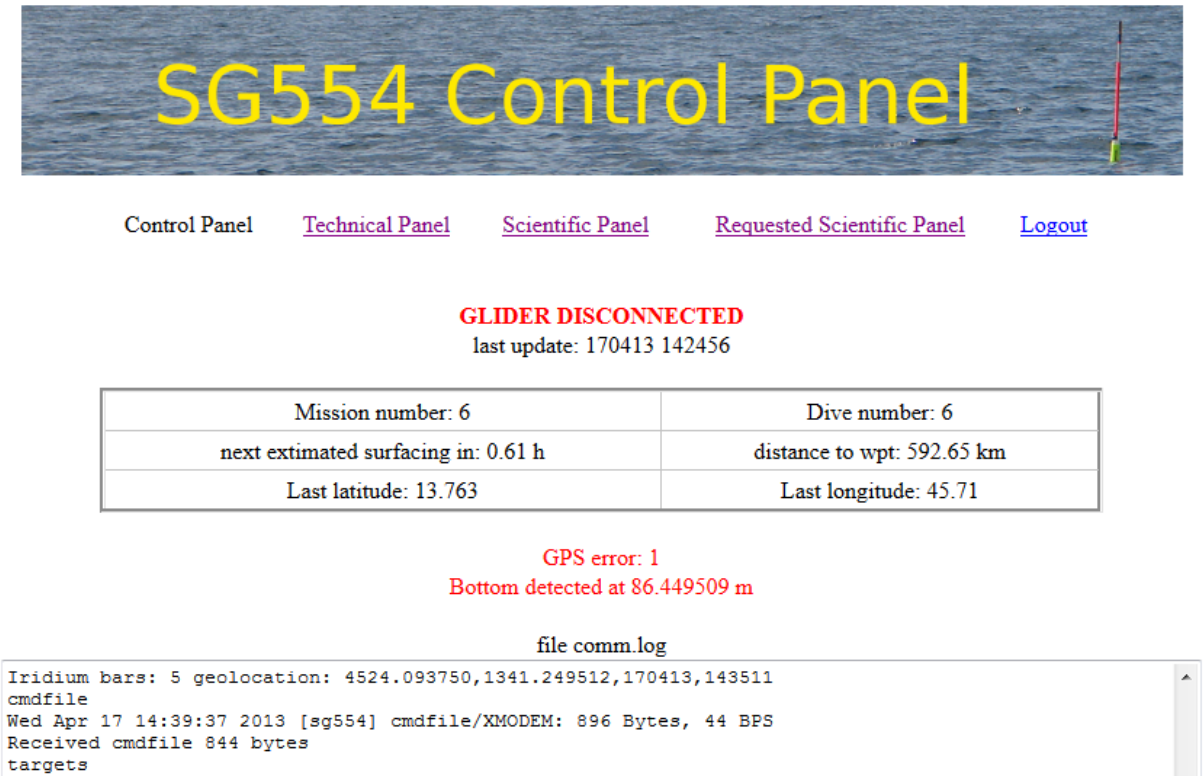


Illustrazione 1

La creazione del file /storage/sire/dati/glider/seaglider/amerigo/actual/transmitted fa eseguire lo script **/storage/sire/work/glider/script/updateDataGlider_rsync**. Tale script provvederà a:

- rimuovere il file /storage/sire/dati/glider/seaglider/amerigo/actual/connected
- Eseguire o script **/storage/sire/work/glider/web/amerigo/actual/crea_dati_log.py** con parametro ogni file che inizi per p554 e termini per .log
- Eseguire **/storage/sire/work/glider/web/amerigo/actual/ordina_dati_log.py** con parametro il file/storage/sire/work/web/sire/glider/dati_log.raw
- Eseguire gli script matlab, i quali provvedono alla generazione delle varie immagini riassuntive

matlab -nodisplay </storage/sire/work/glider/web/amerigo/actual/RG_plot_seaglider_data.m

- Spedire email con i dati di posizione, verso i soggetti interni e/o esterni:
  - se posizioni (lat/lon) non aggiornate rispetto alla precedente email.
    echo "/storage/sire/work/web/sire/glider/dati_log.txt presenta una posizione identica alla precedente, controllare" |`/bin/mail -s "$OGGETTOEMAIL" $EMAIL1 $EMAIL2 $EMAIL3`
  - se posizioni (lat/lon) aggiornate rispetto alla precedente email.
    echo $DATO1 |`/bin/mail -s "$OGGETTOEMAIL" $EMAIL1 $EMAIL2 $EMAIL3 $EMAIL5 $EMAIL6 $EMAIL7 $EMAIL8`

  Legenda:

```
OGGETTOEMAIL="Posizione glider-OGS in data: `date`"
EMAIL1='rgerin@inogs.it'
EMAIL2='pzuppelli@inogs.it'
EMAIL3='abussani@inogs.it'
EMAIL4='antbus@gmail.com'
```

```
#mail altri -> marina.difesa.it
EMAIL5='coanuss@marina.difesa.it'
EMAIL6='paolo.latronico@marina.difesa.it'
EMAIL7='giovanni.mammana@marina.difesa.it'
EMAIL8='mdpt.an.segrop@marina.difesa.it'
```

- Creare file kml (Google Earth) per traiettorie del glider

/storage/sire/work/glider/web/amerigo/actual/crea_kml.py

- Rimuovere i file di connessione/trasferimento
  rm -f /storage/sire/dati/glider/seaglider/amerigo/actual/connected
  rm -f /storage/sire/dati/glider/seaglider/amerigo/actual/transmitted
  rm -f /storage/sire/work/web/sire/glider/dati_log.raw
  rm -f /storage/sire/work/web/sire/glider/connesso
  rm -f /storage/sire/dati/glider/seaglider/amerigo/actual/connected

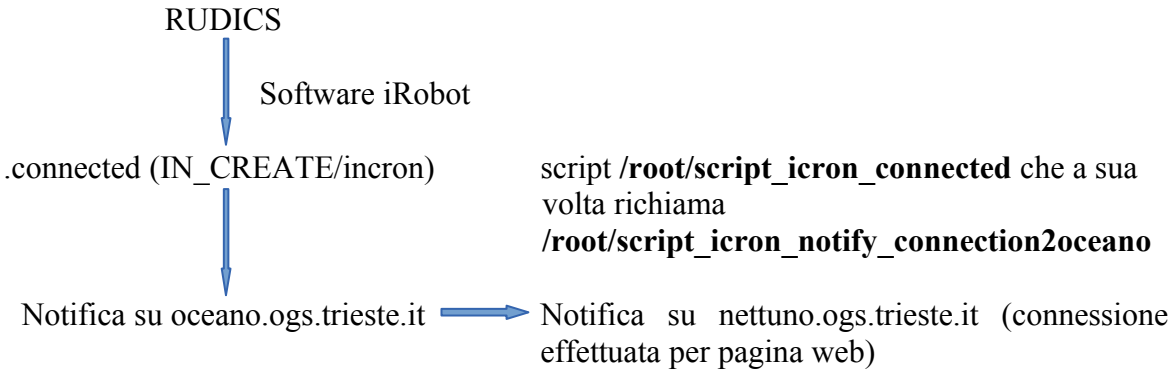
Per maggiori informazioni sui programmi python (*.py) si rimanda a Zuppelli et al. (2013).

Flusso schematico tra sistemi/processi, cartelle ed eventi

Eventi su pcglider-irobot.ogs.trieste.it

(I) Evento iniziale scatenante la procedura di elaborazione. root@pcglider-irobot.ogs.trieste.it /home/sg554/ IN_CREATE, IN_DELETE

```
root@pcglider-irobot sg554
incrontab -l
/home/sg554/ IN_CREATE,IN_DELETE /root/script_icron_connected $# $%
```




```
DEBUG=2
case $DEBUG in
1) STR_DEBUG=""
;;
2) STR_DEBUG=" >>/tmp/glider.$SUFFIX.txt"
esac;

echo -e "PID: $myPID\tfile: .connected\t\tdate: $DATAEXEC\t errnum:
$ERRNUMBER\tTRY: 1" $STR_DEBUG

$(scp $VERBOSE /home/sg554/.connected
glider@oceano.ogs.trieste.it:/storage/sire/dati/glider/seaglider/amerigo/actual/
connected) $STR_DEBUG
ERRNUMBER=$?
while [ "0$ERRNUMBER" -ne "0" ] && [ "0$TRYOUT" -le "0$NTRYOUT" ] #aggiunto il 0
per prevenire se la var e' vuota
do
$(scp $VERBOSE /home/sg554/.connected
glider@oceano.ogs.trieste.it:/storage/sire/dati/glider/seaglider/amerigo/actual/
connected) $STR_DEBUG
      ERRNUMBER=$?
      echo -e "PID: $myPID\tfile: .connected\t\tdate: $DATAEXEC\t errnum:
$ERRNUMBER\tTRY: $TRYOUT/$NTRYOUT" $STR_DEBUG
      let "TRYOUT += 1"
done
```

La procedura notifica ad oceano.ogs.trieste.it e di conseguenza a nettuno.ogs.trieste.it, che il collegamento tra lo strumento oceanografico glider e la basestation (pcglider-irobot.ogs.trieste.it) è in corso.

(II) Secondo evento. "/home/sg554/.connected IN_DELETE"

RUDICS

Software iRobot

.connected (IN_DELETE/incron) script **/root/script_icron_disconnected**

- sleep 60 (per permettere l'elaborazione da parte del SW iRobot)
- rsync -azv -e ssh /home/sg554/ glider@oceano.ogs.trieste.it:/storage/sire/dati/glider/seaglider/amerigo/actual/
- touch /home/sg554/transmitted
- rsync -azv -e ssh /home/sg554/transmitted glider@oceano.ogs.trieste.it:/storage/sire/dati/glider/seaglider/amerigo/actual/
- rm /home/sg554/transmitted

Trasferimento dei file arrivati nella cartella /home/sg554/ che non erano già stati trasferiti in precedenza (rsync). Creazione del file transmitted. Successivo trasferimento del file transmitted verso oceano.ogs.trieste.it. Eliminazione del file transmitted creato localmente.

Eventi su oceano.ogs.trieste.it

(III) Terzo evento "/storage/sire/dati/glider/seaglider/amerigo/actual/ IN_CREATE"

[glider@oceano actual]$ incrontab -l
/storage/sire/dati/glider/seaglider/amerigo/actual/ IN_CREATE
/storage/sire/work/glider/script/incron_amerigo_actual $# $%

(I) Evento
/storage/sire/dati/glider/seaglider/amerigo/actual/connected

/storage/sire/work/glider/script/incron_amerigo_actual

touch /storage/sire/work/web/sire/glider/connesso

Se il file /storage/sire/work/web/sire/glider/connesso è presente, all'interno della pagina web http://nettuno.ogs.trieste.it/sire/glider/sg554_control_panel.php comparirà la relativa notifica, altrimenti il glider risulterà disconnesso (illustrazione 1).

(IV) Quarto evento. "/storage/sire/dati/glider/seaglider/amerigo/actual/transmitted IN_CREATE"

(II) Evento

/storage/sire/dati/glider/seaglider/amerigo/actual/transmitted (IN_CREATE/incron) script **/root/script_icron_disconnected**

/storage/sire/work/glider/script/updateDataGlider_rsync

- Rimozione /storage/sire/dati/glider/seaglider/amerigo/actual/connected
- Per ciascun file $i in /storage/sire/dati/glider/seaglider/amerigo/actual/p554*.log

```
do
/storage/sire/work/glider/web/amerigo/actual/crea_dati_log.py $i
end
```

- **DATO1**=/storage/sire/work/glider/web/amerigo/actual/ordina_dati_log.py /storage/sire/work/web/sire/glider/dati_log.raw
- matlab /storage/sire/work/glider/web/amerigo/actual/RG_plot_seaglider_data.m
- Se **DATO1** risulta vuoto → spedizione delle email al personale OGS

  Se **DATO1** contiene dati → spedizione delle email verso personale OGS e capitanerie
- Creazione kml: crea_kml.py
- Rimozione /storage/sire/dati/glider/seaglider/amerigo/actual/connected Creato da **(I) Evento**
- Rimozione /storage/sire/dati/glider/seaglider/amerigo/actual/transmitted Creato da **(II) Evento**
- Rimozione /storage/sire/work/web/sire/glider/dati_log.raw Creato da **(IV) Evento**, (programma ordina_dati_log.py)
- Rimozione /storage/sire/work/web/sire/glider/connesso Creato da **(I) Evento**
- Rimozione /storage/sire/dati/glider/seaglider/amerigo/actual/connected Creato da **(I) Evento**

Eventi temporali

(V) Evento (temporale) su pcglider-irobot.ogs.trieste.it

Si è visto sperimentalmente che il servizio incrond non è sufficientemente stabile (sulla distribuzione Fedora). Si è pensato perciò di ricaricarlo automaticamente se non risulta avviato.

Lo script che svolge questo compito è: **/root/auto_restart_incrond**

Viene eseguito ogni 10' attraverso il cron.

```
[root@pcglider-irobot sg554]# crontab -l
*/10 * * * * /root/auto_restart_incrond
```

Lo script **/root/auto_restart_incrond** contiene:

```
#!/bin/bash
#AB20130314

#RIAVVIA="/etc/rc.d/rc.incrond restart"
RIAVVIA="/sbin/service incrond restart

#path assoluta del comando pgrep
PGREP="/usr/bin/pgrep"
#nome del demone
NOMEDEMONE="incrond"
#trova il PID del demone
NUMEROPID=$($PGREP $NOMEDEMONE)

# if che vede se il programma/servizio/demone non e' avviato
if [ $? -ne 0 ]
then
# se non e' avviato lo riavvia
$RIAVVIA
Fi
```

(VI) Evento (temporale) su oceano.ogs.trieste.it

Il servizio incrond apparentemente risulta più stabile rispetto a quello installato sul pcglider-irobot.ogs.trieste.it. Si è pensato comunque di controllare la sua esecuzione in modo da ricaricarlo automaticamente se non risulta avviato.

Lo script che svolge questo compito è: **/root/auto_restart_incrond**

E viene eseguito ogni 10' attraverso il cron.

```
[root@oceano ~]# crontab -l
*/10 * * * * /root/auto_restart_incrond
```

Lo script **/root/auto_restart_incrond** contiene:

```
#!/bin/bash
#AB20130314

RIAVVIA="/etc/init.d/incrond status"

#path assoluta del comando pgrep
PGREP="/usr/bin/pgrep"
#nome del demone
NOMEDEMONE="incrond"
#trova il PID del demone
NUMEROPID=$($PGREP $NOMEDEMONE)

# if che vede se il programma/servizio/demone non e' avviato
if [ $? -ne 0 ]
then
# se non e' avviato lo riavvia
$RIAVVIA
fi
```

La variabile ? contiene lo stato di uscita del precedente comando (il comando più recente completato in primo piano).

Riferimenti:

http://tldp.org/LDP/Bash-Beginners-Guide/html/sect_07_01.html

Schema riassuntivo del flusso dati/eventi asincroni (incron)

<table>
<tr><th colspan="2">pcglider-irobot.ogs.trieste.it</th><th colspan="2">oceano.ogs.trieste.it</th><th>nettuno.ogs.trieste.it</th></tr>
<tr><td colspan="2"></td><td colspan="2"></td><td></td></tr>
<tr><td colspan="2">Incron (user: root)</td><td colspan="2">Incron (user glider)</td><td></td></tr>
<tr><td colspan="2"></td><td colspan="2"></td><td></td></tr>
<tr><td colspan="2">/home/sg554/ IN_CREATE,IN_DELETE /root/script_icron_connected $# $%</td><td colspan="2" rowspan="3"></td><td rowspan="3"></td></tr>
<tr><td>IN_CREATE</td><td>IN_DELETE</td></tr>
<tr><td>/ root/script_icron_notify_connection2oceano</td><td>/ root/script_icron_disconnected</td></tr>
<tr><td colspan="2" rowspan="4"></td><td colspan="2">/storage/sire/dati/glider/seaglider/amerigo/actual/ IN_CREATE /storage/sire/work/glider/script/incron_amerigo_actual $# $%</td><td rowspan="2"></td></tr>
<tr><td colspan="2">IN_CREATE</td></tr>
<tr><td>*transmitted*</td><td>*connected*</td><td></td></tr>
<tr><td>/ storage/sire/work/glider/script/updateDataGlider_rsync</td><td colspan="2">touch /storage/sire/work/web/sire/glider/connesso</td></tr>
</table>

Schema riassuntivo del flusso dati/eventi sincroni (cron)

pcglider-irobot.ogs.trieste.it	**oceano.ogs.trieste.it**
Cron (root)	**Cron (root)**
30 4 * * * /usr/sbin/ntpdate -s 140.105.64.5	
*/10 * * * * /root/auto_restart_incrond	*/10 * * * * /root//auto_restart_incrond
cron old	**cron old**
* * * * * /root/script_cron /home/sg554/comm.log	
#/home/sg554/ IN_CLOSE_WRITE scp /home/sg554/$# glider@oceano.ogs.trieste.it:/storage/sire/dati/glider/seaglider/amerigo/actual/ #/home/sg554/ IN_CLOSE_WRITE /root/script_icron $#	#/storage/sire/dati/glider/seaglider/amerigo/actual/ IN_CLOSE_WRITE /storage/sire/work/glider/script/updateDataGlider $#

```
[root@oceano ~]# cat /root/auto_restart_incrond
#!/bin/bash
#AB20130314

RIAVVIA="/etc/init.d/incrond status"

#path assoluta del comando pgrep
PGREP="/usr/bin/pgrep"
#nome del demone
NOMEDEMONE="incrond"
#trova il PID del demone
NUMEROPID=$($PGREP $NOMEDEMONE)

# if che vede se il programma/servizio/demone non e' avviato
if [ $? -ne 0 ]
then
# se non e' avviato lo riavvia
$RIAVVIA
```

```
fi
```

/usr/sbin/ntpdate -s 140.105.64.5

```
man ntpdate
ntpdate(8)                                                                                                    ntpdate(8)

NAME
       ntpdate - set the date and time via NTP

       Disclaimer:  The  functionality  of  this  program is now available in the ntpd program. See the -q command line
       option in the ntpd - Network Time Protocol (NTP) daemon page. After a suitable period of mourning,  the  ntpdate
       program is to be retired from this distribution
```

Il server 140.105.64.5 è il server dedicato al servizio NTP, il suo nome è timeserver.ogs.trieste.it

Nella tabella di riferimento sono stati inseriti (nella riga denominata “cron old”) i precedenti script utilizzati per il trasferimento dei dati: il trasferimento tramite scp dava problemi quando c'erano molti file che dovevano venir trasferiti in un breve intervallo di tempo. Era stata tentata anche la via del controllo sul trasferimento, ma questa metodologia produceva soltanto un aumento della coda dei trasferimenti, aumentando ulteriormente i problemi di trasferimento. Quindi si è scelto di usare il sistema rsync ed effettuando il trasferimento solo quando il glider ha finito di trasmettere e il software proprietario dell'irobot ha finito di elaborare i file che sono arrivati.

Riferimenti per le configurazioni dei pc all'interno della rete OGS:
http://webcc.ogs.trieste.it/ConfiguraPC.html

Riferimenti per i servizi cron
http://www.feelinglinux.com/articles/cron.jsp

Schema degli script di backup/pulizia per la preparazione di una nuova missione

pcglider-irobot.ogs.trieste.it	oceano.ogs.trieste.it	nettuno.ogs.trieste.it
/root/glider_sg554_backup_missione	/storage/sire/work/glider/script/glider_sg554_backup_missione	
/root/glider_sg554_nuova_missione	/storage/sire/work/glider/script/glider_sg554_nuova_missione	

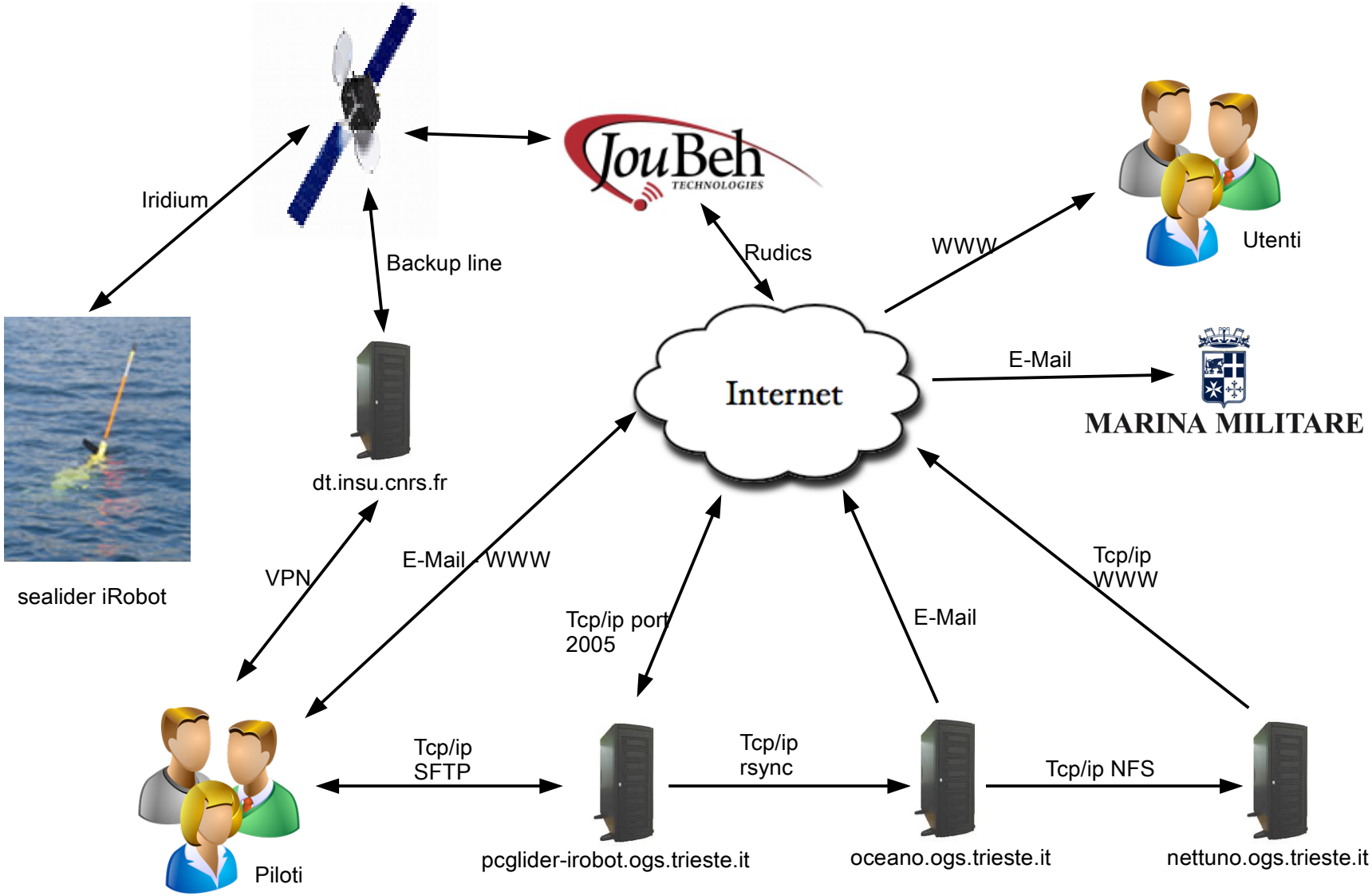
/root/glider_sg554_backup_missione e **/storage/sire/work/glider/script/glider_sg554_backup_missione**
Producono il backup della cartella /home/sg554 e mette i file nella cartella XXX (richiesta dallo script bash e li copia in /home/sg554/mission/XXX

/root/glider_sg554_nuova_missione e **/storage/sire/work/glider/script/glider_sg554_nuova_missione**
Eliminano i file in /home/sg554/ tranne sg_calib_constants.m che viene salvato

```
[root@pcglider-irobot sg554]# cat /root/glider_sg554_backup_missione
#!/bin/bash
#AB20130221

SUFFIX=$(date +%F)
DATAEXEC=$(date)
ARGV0=$0 # First argument is shell command (as in C)
ARGV1=$1
myPID=$$
#ERRNUMBER=999999
NTRYOUT=20
TRYOUT=2
MINTIMEDIFF=5
SECS=5
#in secondi

#sleep 1

PATHA="/home/sg554"
FORCE="-vf"
```

```
echo "Inserire la cartella di destinazione, nel formato YYYYMM_nomemissione, nel percorso /home/sg554/mission/"
read RISPOSTA

if [[ $RISPOSTA ]]; then
        echo "Sei sicuro di copiare i dati in: /home/sg554/mission/$RISPOSTA (Y/N)?"
        read RISPOSTA2
        if [[ $RISPOSTA2="y" || $RISPOSTA2="Y" ]]; then
                echo ""
                /bin/mkdir $PATHA/mission/$RISPOSTA
                #$ERR - $? codice di uscita dell'ultima istruzione
                if [[ $? ]];then
                        echo ""
                else
                        echo "$ERR uscita script - backup non effettuato"
#                       exit
                fi
                /bin/cp -vf $PATHA/* $PATHA/mission/$RISPOSTA
                echo -e "\nSe la copia e' andata a buon fine (controllare!) puoi eseguire lo script:"
                echo -e "./root/glider_sg554_nuova_missione\n per cancellare i dati da $PATHA"
                CONTEGGIO=$(ls $PATHA/*|wc)
                CONTEGGIO2=$(ls $PATHA/mission/$RISPOSTA|wc)
                echo -e "Se $CONTEGGIO \t = $CONTEGGIO2\t allora copia eseguita"
        else
                echo "Se vuoi creare un backup, riesegui il programma. Ciao!"
                exit
        fi
else
        echo "La rinuncia e' la cosa corretta, se non si e' sicuri di quello che lo script fa!"
        exit
fi
```

Figura riassuntiva del flusso di dati



Configurazione pcglider-irobot.ogs.trieste.it - incrontab

```
cat /etc/issue
Fedora release 16 (Verne)
```

```
yum update yum
yum install inotify-tools
```

```
Dependencies Resolved

================================================================================
================================================================
 Package                        Arch                              Version
Repository                      Size
================================================================================
================================================================
Installing:
 incron                         x86_64                            0.5.9-2.fc15
fedora                          78 k

Transaction Summary
================================================================================
================================================================
Install  1 Package

Total download size: 78 k
Installed size: 78 k
Is this ok [y/N]: y
Downloading Packages:
incron-0.5.9-2.fc15.x86_64.rpm
|  78 kB     00:00
Running Transaction Check
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing : incron-0.5.9-2.fc15.x86_64
1/1
  Verifying  : incron-0.5.9-2.fc15.x86_64
1/1

Installed:
  incron.x86_64 0:0.5.9-2.fc15

Complete!
```

Per far partire il servizio

```
service incrond start
```

Per farlo caricare all'avvio

```
chkconfig incrond on
```

Per controllare (per ciascun utente) i file/cartelle monitorati e lo script da eseguire

```
incrontab -l
/home/sg554/ IN_CREATE,IN_DELETE /root/script_icron_connected $# $%
```

Per vedere gli eventi che l'incrontab installato prevede:

```
incrontab -t
IN_ACCESS,IN_MODIFY,IN_ATTRIB,IN_CLOSE_WRITE,IN_CLOSE_NOWRITE,IN_OPEN,IN_MOVED_F
ROM,IN_MOVED_TO,IN_CREATE,IN_DELETE,IN_DELETE_SELF,IN_CLOSE,IN_MOVE,IN_ONESHOT,I
N_ALL_EVENTS,IN_DONT_FOLLOW,IN_ONLYDIR,IN_MOVE_SELF
```

Per editare il file incrontab:

```
incrontab -e
```

Per vedere i log di intervento di incrontab, il log "tipico" è

```
/var/log/cron
```

ma può differire dalla distribuzione.

Il file incrontab deve essere composto con determinate caratteristiche sintattiche: i campi devono essere separati dal carattere di spaziatura, separare i campi con il consueto TAB porta a problemi di interpretazione della incrontab da parte del demone incrond.

I tre campi sono, nell'ordine:

- **path** – Il path ASSOLUTO che si desidera monitorare
- **mask** – L'evento che si vuole monitorare. Sono supportati tutte le maschere che sono state descritte prima, più la maschera IN_ALL_EVENTS che, semplicemente, cattura tutti gli eventi (ottima per il debugging)
- **command**- Il comando da eseguire quando viene catturato l'evento. Da notare che in questo comando possiamo utilizzare delle variabili di inotify che ci vengono passate direttamente dal demone. Queste variabili sono:
  - **$@** – Il path monitorato
  - **$#** – Il file sul quale si è scatenato l'evento
  - **$%** – L'evento, in forma testuale, che si è verificato
  - **$&** – L'evento, in forma numerica, che si è verificato
  - **$$** – Il carattere $

Riferimento:
http://archive09.linux.com/feature/144666
http://www.miamammausalinux.org/2011/10/monitorare-directory-e-rispondere-ad-eventi-specifici/

Idee per velocizzare il trasferimento tramite SCP

Il sistema di trasferimento tramite scp è stato abbandonato per i motivi descritti precedentemente, abbandonando quindi questa tipologia per il sistema di trasferimento attraverso rsync.

Per la sicurezza della connessione un manuale dettagliato si trova:
Riferimento:
http://docstore.mik.ua/orelly/networking_2ndEd/ssh/ch11_01.htm#ch11-57817.html

Per problemi di lentezza nell'autenticazione, questo articolo raccoglie in maniera dettagliata i vari passi per risolvere la situazione.
Riferimento:
http://www.walkernews.net/2009/04/06/how-to-fix-scp-and-ssh-login-prompt-is-very-slow-in-linux/

Configurazione del fuso orario con la distribuzione FEDORA 16

Specificare un fuso orario anche se si sta pianificando di utilizzare NTP (Network Time Protocol) per mantenere costante la precisione dell'orologio di sistema.

Sono disponibili diversi modi per selezionare il fuso orario:

1. Usando il mouse, fate clic sulla mappa interattiva per selezionare una città specifica, (contrassegnata da un punto giallo). Comparirà una X rossa che indica la scelta.
2. Il fuso orario può anche essere selezionato tramite un elenco posto nella parte inferiore della schermata. Usando il mouse, cliccare sulla mappa per evidenziare la scelta.
3. (Per le distribuzioni linux, riga da comando).

Per evitare problemi è comodo eseguire il backup del precedente tempo locale.

```
mv /etc/localtime  /etc/localtime-old
```

Creare un collegamento simbolico per il fuso orario appropriato ad /etc/localtime oppure copiare il file relativo.

```
ln -sf /usr/share/zoneinfo/Europe/Amsterdam /etc/localtime
```

oppure usando il copy

```
cp /usr/share/zoneinfo/Europe/Rome/ /etc/localtime
```

Se Fedora è l'unico sistema operativo sul computer, selezionare UTC. L'orologio di sistema è una parte dell'hardware sul computer. Fedora utilizza le impostazioni di fuso orario per determinare lo sfasamento fra l'ora locale e l'UTC dell'orologio di sistema. Questo comportamento è universale per i sistemi operativi stile UNIX.

Riferimento:
http://docs.fedoraproject.org/it-IT/Fedora/11/html/Installation_Guide/s1-timezone-x86.html

Configurazione del server NTP con la distribuzione FEDORA 16

Per tenere aggiornato il tempo del server è necessario usare il protocollo NTP. All'interno della rete interna dell'OGS esiste un particolare server dedicato a questo scopo:
timeserver.ogs.trieste.it con indirizzo 140.105.64.5

Nel file

```
/etc/ntp.conf
```
inserire il server ntp dell'OGS
```
server 140.105.64.5 iburst
```

per l'aggiornamento a riga di comando:
```
ntpdate -q 140.105.64.5
```

Per attivare il servizio all'avvio:
```
systemctl enable ntpdate.service
```

Link di riferimento:
http://docs.fedoraproject.org/en-US/Fedora/15/html/Deployment_Guide/sect-Date_and_Time_Configuration-Command_Line_Configuration-Network_Time_Protocol.html

```
date
Fri Mar  1 11:03:26 CET 2013
```

Riferimento:
http://webcc.ogs.trieste.it/ConfiguraPC.html
http://it.wikipedia.org/wiki/Network_Time_Protocol

Configurazione pcglider-irobot.ogs.trieste.it – rudics tunneling

La configurazione della porta necessaria al collegamento con il servizio rudics si trova nel file:
/etc/services

iRobotRudicsd 2005/tcp # iRobotRudics daemon
La porta aperta verso l'estero è appunto la 2500

Configurazione pcglider-irobot.ogs.trieste.it – fail2ban

Per bloccare le ripetute richieste di accesso da parte di virus/script malevoli o hacker, è necessario installare un sistema di blocco per un numero di accessi superiori ad una determinata quantità.

Fail2ban (http://www.fail2ban.org/) è un software che è nato per permettere di bloccare gli host che stanno tentando di effettuare un attacco di brute force via SSH..
Questo genere di attacco si basa su continui tentativi di accesso, provando o l'username root con password probabili (come password, root, toor e così via) oppure tramite coppie di username/password conosciute (create, ad esempio, da malware o da rootkit)

Fail2ban controlla ciascuna riga dei log nel lasso di tempo che è stato prefissato.

Dettaglio installazione fail2ban
```
yum install fail2ban
```

```
Loaded plugins: langpacks, presto, refresh-packagekit
updates/metalink
|  27 kB     00:00
Resolving Dependencies
--> Running transaction check
---> Package fail2ban.noarch 0:0.8.4-27.fc16 will be installed
--> Processing Dependency: python-inotify for package: fail2ban-0.8.4-
27.fc16.noarch
--> Processing Dependency: shorewall for package: fail2ban-0.8.4-27.fc16.noarch
--> Running transaction check
---> Package python-inotify.noarch 0:0.9.4-1.fc16 will be installed
---> Package shorewall.noarch 0:4.5.1-1.fc16 will be installed
--> Processing Dependency: shorewall-core = 4.5.1-1.fc16 for package: shorewall-
4.5.1-1.fc16.noarch
--> Processing Dependency: perl(Digest::SHA1) for package: shorewall-4.5.1-
1.fc16.noarch
--> Running transaction check
---> Package perl-Digest-SHA1.x86_64 0:2.13-3.fc16 will be installed
---> Package shorewall-core.noarch 0:4.5.1-1.fc16 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
================================================
 Package                              Arch                          Version
Repository                     Size
================================================================================
================================================
Installing:
 fail2ban                             noarch                        0.8.4-27.fc16
fedora                        126 k
Installing for dependencies:
 perl-Digest-SHA1                     x86_64                        2.13-3.fc16
fedora                         48 k
 python-inotify                       noarch                        0.9.4-1.fc16
updates                        48 k
 shorewall                            noarch                        4.5.1-1.fc16
updates                       462 k
 shorewall-core                       noarch                        4.5.1-1.fc16
updates                        55 k

Transaction Summary
================================================================================
================================================
Install  1 Package (+4 Dependent packages)

Total download size: 740 k
Installed size: 2.4 M
Is this ok [y/N]: y
Downloading Packages:
(1/5): fail2ban-0.8.4-27.fc16.noarch.rpm
| 126 kB     00:00
(2/5): perl-Digest-SHA1-2.13-3.fc16.x86_64.rpm
|  48 kB     00:00
(3/5): python-inotify-0.9.4-1.fc16.noarch.rpm
|  48 kB     00:00
```

```
(4/5): shorewall-4.5.1-1.fc16.noarch.rpm
| 462 kB     00:00
(5/5): shorewall-core-4.5.1-1.fc16.noarch.rpm
|  55 kB     00:00
--------------------------------------------------------------------------------
--------------------------------------------------
Total
772 kB/s | 740 kB     00:00
Running Transaction Check
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing : perl-Digest-SHA1-2.13-3.fc16.x86_64
1/5
  Installing : python-inotify-0.9.4-1.fc16.noarch
2/5
  Installing : shorewall-core-4.5.1-1.fc16.noarch
3/5
  Installing : shorewall-4.5.1-1.fc16.noarch
4/5
  Installing : fail2ban-0.8.4-27.fc16.noarch
5/5
  Verifying  : fail2ban-0.8.4-27.fc16.noarch
1/5
  Verifying  : shorewall-core-4.5.1-1.fc16.noarch
2/5
  Verifying  : shorewall-4.5.1-1.fc16.noarch
3/5
  Verifying  : python-inotify-0.9.4-1.fc16.noarch
4/5
  Verifying  : perl-Digest-SHA1-2.13-3.fc16.x86_64
5/5

Installed:
  fail2ban.noarch 0:0.8.4-27.fc16

Dependency Installed:
  perl-Digest-SHA1.x86_64 0:2.13-3.fc16         python-inotify.noarch 0:0.9.4-
1.fc16         shorewall.noarch 0:4.5.1-1.fc16
  shorewall-core.noarch 0:4.5.1-1.fc16

Complete!
```

Per farlo caricare all'avvio

```
chkconfig --levels 235 fail2ban on
```

Per far partire il servizio (dipende dal tipo di distribuzione installata)

```
/etc/init.d/fail2ban start
```

oppure

```
service fail2ban start
```

Per farlo caricare all'avvio

```
chkconfig incrond on
```

Tutti i file di configurazione di trovano in:

```
/etc/fail2ban
```

i fallimenti di accesso sono registrati in:

```
 /var/log/secure

sargassi.ogs.trieste.it  user=pilot
Feb 14 12:09:17 pcglider-irobot sshd[15227]: Failed password for pilot from
140.105.70.146 port 7320 ssh2
Feb 14 12:09:24 pcglider-irobot sshd[15227]: Failed password for pilot from
140.105.70.146 port 7320 ssh2
Feb 14 12:09:27 pcglider-irobot sshd[15227]: Failed password for pilot from
140.105.70.146 port 7320 ssh2
Feb 14 12:09:31 pcglider-irobot sshd[15227]: Failed password for pilot from
140.105.70.146 port 7320 ssh2
```

Per modifiche sui tentativi di accesso e la durata, è necessario modificare lo script

```
/etc/fail2ban/jail.conf

[DEFAULT]

# "ignoreip" can be an IP address, a CIDR mask or a DNS host. Fail2ban will not
# ban a host which matches an address in this list. Several addresses can be
# defined using space separator. #localhost sargassi morto
ignoreip = 127.0.0.1 #140.105.70.146 140.105.70.25

# "bantime" is the number of seconds that a host is banned.
bantime  = 600

# A host is banned if it has generated "maxretry" during the last "findtime"
# seconds.
findtime  = 600

# "maxretry" is the number of failures before a host get banned.
Maxretry = 3

[ssh-iptables]

enabled  = true
filter   = sshd
action   = iptables[name=SSH, port=ssh, protocol=tcp]
           sendmail-whois[name=SSH, dest=root, sender=fail2ban@example.com]
logpath  = /var/log/secure
maxretry = 5

[proftpd-iptables]

enabled  = false
filter   = proftpd
action   = iptables[name=ProFTPD, port=ftp, protocol=tcp]
           sendmail-whois[name=ProFTPD, dest=you@example.com]
logpath  = /var/log/proftpd/proftpd.log
maxretry = 6
```

Per controllare le espressioni regolari che intervengono per recuperare le informazioni dai vari log

```
/etc/fail2ban/filter.d/sshd.conf

failregex = ^%(__prefix_line)s(?:error: PAM: )?Authentication failure for .*
from <HOST>\s*$
            ^%(__prefix_line)s(?:error: PAM: )?User not known to the underlying
authentication module for .* from <HOST>\s*$
            ^%(__prefix_line)sFailed (?:password|publickey) for .* from
<HOST>(?: port \d*)?(?: ssh\d*)?$
            ^%(__prefix_line)sROOT LOGIN REFUSED.* FROM <HOST>\s*$
            ^%(__prefix_line)s[iI](?:llegal|nvalid) user .* from <HOST>\s*$
            ^%(__prefix_line)sUser \S+ from <HOST> not allowed because not
listed in AllowUsers$
            ^%(__prefix_line)sauthentication failure; logname=\S* uid=\S*
euid=\S* tty=\S* ruser=\S* rhost=<HOST>(?:\s+user=.*)?\s*$
            ^%(__prefix_line)srefused connect from \S+ \(<HOST>\)\s*$
            ^%(__prefix_line)sAddress <HOST> .* POSSIBLE BREAK-IN ATTEMPT!*\s*$
            ^%(__prefix_line)sUser \S+ from <HOST> not allowed because none of
user's groups are listed in AllowGroups$
```

Per i test sulle espressioni regolari usate, è necessario usare questo comando:

```
fail2ban-regex /var/log/secure.log /etc/fail2ban/filter.d/sshd.conf

`- Number of matches:
   [1] 0 match(es)
   [2] 0 match(es)
   [3] 0 match(es)
   [4] 0 match(es)
   [5] 0 match(es)
   [6] 0 match(es)
   [7] 0 match(es)
   [8] 0 match(es)
   [9] 0 match(es)
   [10] 0 match(es)
```

Se l'espressione regolare è corretta ma il demone non blocca ancora i tentativi di intrusione, è necessario controllare il tempo corrente (e relativo fuso orario) del pc usato.

Riferimento:
http://guide.debianizzati.org/index.php/Fail2ban#Introduzione
http://www.howtoforge.com/preventing-brute-force-attacks-with-fail2ban-on-fedora9

Test fail2ban

Apertura porta 22 da parte del firewall dell'OGS

pcglider-irobot fail2ban lo stesso giorno dopo aver aperto la porta 22 all'esterno, email dell'utente root (client di posta mutt):

```
Date: Thu, 21 Feb 2013 02:06:57 +0100
```

```
From: Fail2Ban <fail2ban@example.com>
To: root@pcglider-irobot.ogs.trieste.it
Subject: [Fail2Ban] SSH: banned 46.165.196.43

Hi,

The IP 46.165.196.43 has just been banned by Fail2Ban after
5 attempts against SSH
```

Ulteriori tentativi di connessione da quell'ip (all'interno del lasso temporale configurato) risultano impossibili, come se il computer fosse chiuso (non vengono accettate connessioni da quell' ip).

Le variazioni di durata, ricerca all'interno del lasso temporale e loro quantità ammessa, si trovano all'interno del file /etc/fail2ban/jail.conf (come precedentemente descritto) e gli ultimi settaggi al momento della stesura di questo documento sono i seguenti:

```
# "bantime" is the number of seconds that a host is banned.
bantime  = 3600

# A host is banned if it has generated "maxretry" during the last "findtime"
# seconds.
findtime  = 3600

# "maxretry" is the number of failures before a host get banned.
maxretry = 3
```

Installazione VPN (su pc dei piloti)

Per una descrizione completa si rimanda a Bussani (2013).

Configurazione oceano.ogs.trieste.it

L'utilizzo di yum e del servizio di aggiornamenti attivo rende l'installazione praticamente automatica.

```
cat /etc/issue
Red Hat Enterprise Linux Server release 5.8 (Tikanga)
```

Per l'installazione:

```
yum install incron
yum install inotify-tools
```

Packages Installed:
incron-0.5.5-2.el5.x86_64
inotify-tools-3.14-1.el5.i386

Monitoraggio in tempo reale su oceano.ogs.trieste.it e pcglider-irobot.ogs.trieste.it

```
tail -f /tmp/gliderYYYYMMGG.log
```

dove YYYYMMGG è l'anno il mese e il giorno della trasmissione.

Utilizzare scp e/o rsync, senza l'immissione della password

Quando è necessario utilizzare scp oppure rsync per copiare i file tra host diversi, viene richiesto l'accesso (login e password) da parte del sistema di destinazione.
La digitazione della password diventa impossibile quando scp e rsync vengono usati in script automatici (cron e incron). Per ovviare a questo problema esiste la possibilità di usare una coppia di chiavi opportunamente generata.

Attraverso rsync è possibile trasferire file tra 2 hosts, hosts sorgente (host_src) ed e host destinazione (host_dest), senza preoccuparsi della direzione dei file da copiare.

Sull'host sorgente eseguire questo comando con l'utente che userà uno dei seguenti comandi: scp/ssh/rsync

```
$ ssh-keygen -t rsa
```

Questo richiederà una passphrase. Non introdurre niente, e premere invio. Questo genererà un identificativo (chiave privata) e una chiave pubblica. Non divulgare la chiave privata con nessuno.
Ssh-keygen mostra dove questo file viene salvato, di default ~/.ssh/id_rsa.pub

La chiave pubblica è stata salvata in <your_home_dir>/.ssh/id_rsa.pub

E' necessario trasferire il file id_rsa.pub file al host_dest, via ftp, scp, rsync o qualsiasi altro metodo.
Sull'host_dest, loggarsi come utente remoto con il quale pianificate di effettuare il trasferimento attraverso scp, ssh o rsync verso l'host_src
Copiare il contenuto di id_rsa.pub verso ~/.ssh/authorized_keys

```
    $ cat id_rsa.pub >>~/.ssh/authorized_keys
    $ chmod 700 ~/.ssh/authorized_keys
```

Se questo file non esiste, il comando sopra indicato lo creerà. E' necessario rimuovere i permessi per gli altri di leggere il file.
Perchè prevenire la lettura di quei file se questa è una chiave pubblica? Il possessore della chiave può distribuirlo solo agli utenti “trusted”, ottenere questa chiave senza il consenso sarebbe un problema di sicurezza.

Di default il demone ssh non permette all'utente root di loggarsi. Se è necessario che l'utente root acceda verso host_dest è necessario editare

```
/etc/ssh/sshd_config
```

e cambiare l'opzione da

```
PermitRootLogin no
```

a

```
PermitRootLogin yes
```

Non dimenticarsi di far ripartire il demone sshd

```
/etc/init.d/sshd restart
```

Ora si è pronti ad eseguire scp, ssh e rsync collegando host_src verso host_dest, senza la richiesta della password.

Se è stato fatto tutto correttamente ora dovrebbe essere possibile eseguire scp, ssh e rsync sul host_src.

Attenzione che verrà richiesta la password se si eseguono i comandi su host_dest con la connessione verso host_src. Eseguendo i passi precedenti (generare la chiave pubblica su host_dest e copiare la chiave pubblica su host-src), si ha pronta un installazione a due vie!

Su pcglider-irobot.ogs.trieste.it

```
ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
/root/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
d6:f5:f0:e6:e2:2d:e6:93:8c:30:7a:86:7d:5c:d4:74 root@pcglider-
irobot.ogs.trieste.it
The key's randomart image is:
+--[ RSA 2048]----+
|                 |
|               . E|
|            oo . |
|          . ..+.  |
|         S ..  +  |
|        .o   .o   |
|        + + +...  |
|       o + +.*o   |
|        o . ooo.  |
+-----------------+
```

```
[root@pcglider-irobot .ssh]#scp /home/sg554/st0026lu.x01
glider@oceano.ogs.trieste.it:/storage/sire/dati/glider/seaglider/amerigo/actual/
st0026lu.x01
100% 1024     1.0KB/s   00:00
```

Riferimento:
https://blogs.oracle.com/jkini/entry/how_to_scp_scp_and
http://www.spaghettilinux.org/tips-tricks/usare-ssh-per-gestire-un-server-remoto-senza-inserire-sempre-la-password

Per iniziare una nuova missione

Due script (per server) si occupano di eseguire il backup dei vecchi dati e di ripristinare le opportune cartelle allo stato iniziale.

Su pcglider-irobot.ogs.trieste.it **/root/glider_sg554_backup_missione** fa il backup della cartella /home/sg554 copiando i file nella cartella /home/sg554/mission/XXX (richiesta dallo script bash).

/root/glider_sg554_nuova_missione elimina il file in /home/sg554/, tranne sg_calib_constants.m

Dopo questo script la cartella risulta pulita e quindi è possibile ricevere nuovi dati.

Su oceano.ogs.trieste.it gli script da eseguire sono:
/storage/sire/work/glider/script/glider_sg554_backup_missione
/storage/sire/work/glider/script/glider_sg554_nuova_missione

Test dei flussi dati – Spedizione dati dal glider

Dopo aver connesso il glider attraverso il cavo seriale (connettore alla base dell'antenna). E aver aperto il glider facendo passare la calamita sul punto ON sul fianco del glider. Dal terminale (9600 baud) si potrà vedere il menù.
Per la spedizione dei dati è necessario entrare

```
2 → HW
```

```
8 → Modem
```

```
6 → Exercise upload data
```

or

```
7 → Exercise upload data self test only
```

Il glider tiene conto dei dati che precedentemente sono già stati spediti, è consigliabile quindi (per testare il flusso dati) copiare all'interno della cartella del pcglider-irobot /home/sg554/ tutti i file di una precedente missione.

Malfunzionamento del servizio incrond e relativa risoluzione/workaround

Con l'uso si è notato che potevano avvenire dei crash relativi al demone incrond (il software nel 2013 risultava ancora non stabile)

Versione installata su oceano.ogs.trieste.it

```
yum info incron
Installed Packages
Name        : incron
Arch        : x86_64
Version     : 0.5.5
Release     : 2.el5
Size        : 280 k
Repo        : installed
Summary     : Inotify cron system
URL         : http://inotify.aiken.cz
License     : GPL
```

```
Description: This program is an "inotify cron" system.
           : It consists of a daemon and a table manipulator.
           : You can use it a similar way as the regular cron.
           : The difference is that the inotify cron handles
           : filesystem events rather than time periods.
```

Versione installata su pcglider-irobot.ogs.trieste.it

```
yum info incron
Installed Packages
Name        : incron
Arch        : x86_64
Version     : 0.5.9
Release     : 2.fc15
Size        : 234 k
Repo        : installed
From repo   : fedora
Summary     : Inotify cron system
URL         : http://inotify.aiken.cz
License     : GPLv2
Description : This program is an "inotify cron" system.
            : It consists of a daemon and a table manipulator.
            : You can use it a similar way as the regular cron.
            : The difference is that the inotify cron handles
            : filesystem events rather than time periods.
```

Il crash viene loggato nei log relativi ai cron (cron e incron)

```
grep unhandled /var/log/cron
Mar 12 18:30:27 pcglider-irobot incrond[12250]: *** unhandled exception occurred
***
```

Per ripristinare il servizio (su fedora 16):

```
service incrond restart
```

Visto che senza il servizio incron il flusso si interrompe è necessario avere una possibilità automatica di notifica/ripristino del servizio.

Lo script di riavvio **/root/auto_restart_incrond**, è inserito in cron con frequenza 10 minuti (se la discesa del glider e relativa risalita è di 30 minuti, la frequenza dovrebbe essere meno della metà).

```
cat /root/auto_restart_incrond
#!/bin/bash
#AB20130314

RIAVVIA="/sbin/service incrond restart"

#path assoluta del comando pgrep
PGREP="/usr/bin/pgrep"
#nome del demone
NOMEDEMONE="incrond"
#trova il PID del demone
NUMEROPID=$($PGREP $NOMEDEMONE)

# if che vede se il programma/servizio/demone non e' avviato
if [ $? -ne 0 ]
```

```
then
# se non e' avviato lo riavvia
$RIAVVIA
fi
```

```
[root@pcglider-irobot ~]# crontab -l
30 4 * * *    /usr/sbin/ntpdate -s 140.105.64.5
* * * * * /root/script_cron /home/sg554/comm.log >/dev/null
*/10 * * * * /root/auto_restart_incrond
```

Riferimenti/Bibliografia/Relazioni

http://nettuno.ogs.trieste.it/sire/glider/Seaglider%20User%20Guide%20-%20Rev.%20C.pdf
http://nettuno.ogs.trieste.it/sire/glider/appunti_seattle_elena.pdf
http://nettuno.ogs.trieste.it/sire/glider/appunti_seattle_riccardo.pdf
http://tldp.org/LDP/Bash-Beginners-Guide/html/sect_07_01.html
http://webcc.ogs.trieste.it/ConfiguraPC.html
http://www.feelinglinux.com/articles/cron.jsp
http://archive09.linux.com/feature/144666
http://www.miamammausalinux.org/2011/10/monitorare-directory-e-rispondere-ad-eventi-specifici/
http://docstore.mik.ua/orelly/networking_2ndEd/ssh/ch11_01.htm#ch11-57817.html
http://www.walkernews.net/2009/04/06/how-to-fix-scp-and-ssh-login-prompt-is-very-slow-in-linux/
http://docs.fedoraproject.org/it-IT/Fedora/11/html/Installation_Guide/s1-timezone-x86.html
http://webcc.ogs.trieste.it/ConfiguraPC.html
http://it.wikipedia.org/wiki/Network_Time_Protocol
http://guide.debianizzati.org/index.php/Fail2ban#Introduzione
http://www.howtoforge.com/preventing-brute-force-attacks-with-fail2ban-on-fedora9
https://blogs.oracle.com/jkini/entry/how_to_scp_scp_and
http://www.spaghettilinux.org/tips-tricks/usare-ssh-per-gestire-un-server-remoto-senza-inserire-sempre-la-password

Bussani A., 2013. Installazione OPENVPN da ogs.trieste.it a dt.insu.cnrs.fr per backup pc di comando glider. Rel. OGS 2013/18 Sez. OCE 10 SIRE

Zuppelli P., 2013 - in fase di stesura/rilascio